Computational Complexity of Robust Bilevel Optimization Problems

Dissertation
zur Erlangung des akademischen Grades einer
Doktorin der Naturwissenschaften
(Dr. rer. nat.)

Der Fakultät für Mathematik der
Technischen Universität Dortmund
vorgelegt von

Dorothee Henke

im Juli 2025

Dissertation

Computational Complexity of Robust Bilevel Optimization Problems

Fakultät für Mathematik

Technische Universität Dortmund

Erstgutachter: Prof. Dr. Christoph Buchheim

Zweitgutachterin: Prof. Dr. Britta Peis

Tag der mündlichen Prüfung: 17. September 2025

Abstract

Bilevel optimization is concerned with hierarchical optimization problems that involve two decision makers, a leader and a follower. Each player's decision influences the other player's optimization problem. Therefore, the leader has to anticipate the follower's reaction when taking the first decision, which can make the bilevel problem very hard to solve. In terms of computational complexity theory, combinatorial bilevel optimization problems are typically on the second level of the polynomial-time hierarchy.

In this thesis, we study three relatively easy and natural combinatorial bilevel optimization problems. The Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem can be solved in polynomial time and the Bilevel Minimum Spanning Tree Problem is $\mathcal{NP}$-equivalent. In all of them, the leader and the follower each control a subset of some set of items, and they build a feasible solution of the underlying single-level problem together, while optimizing different objective functions.

In case of the Bilevel Minimum Spanning Tree Problem, we also investigate the approximability and the special case where the number of follower's edges is small, and we characterize the complexity of problem variants involving bottleneck objective functions.

Besides the complexity of bilevel optimization problems, we also study the additional complexity that uncertainty introduces in bilevel problems. In our setting of robust bilevel optimization, we assume that the leader does not have full information about the follower's objective function and optimizes for the worst case among the possible follower's reactions. We consider several types of uncertainty sets that are typically used in robust optimization, such as discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty, and study the effect that the robustness has on the underlying bilevel problem, in terms of its computational complexity.

We present complexity results for a class of robust linear bilevel problems with discrete leader's and continuous follower's decisions and we thoroughly study robust versions of the polynomial-time solvable Bilevel Selection Problem and Bilevel Continuous Knapsack Problem. Some of them are still solvable in polynomial time, while others are $\mathcal{NP}$-hard. These results reveal several differences between robust bilevel optimization and robust single-level optimization. For example, in the former case, interval uncertainty is not trivial and an uncertainty set can usually not be replaced by its convex hull without changing the problem.

Zusammenfassung

Bilevel-Optimierung befasst sich mit hierarchischen Optimierungsproblemen, in denen zwei Personen, ein *Leader* und ein *Follower*, Entscheidungen treffen. Die Entscheidung jeder Person beeinflusst auch das Optimierungsproblem der anderen Person. Aufgrund dessen muss der Leader bereits beim Treffen der ersten Entscheidung die Reaktion des Followers antizipieren, was die Lösung des Bilevel-Problems sehr schwierig machen kann. Komplexitätstheoretisch lässt sich sagen, dass sich kombinatorische Bilevel-Optimierungsprobleme typischerweise in der zweiten Ebene der Polynomialzeithierarchie befinden.

In dieser Arbeit untersuchen wir drei relativ einfache und natürliche kombinatorische Bilevel-Optimierungsprobleme. Das Bilevel Selection Problem und das Bilevel Continuous Knapsack Problem können in polynomieller Zeit gelöst werden und das Bilevel Minimum Spanning Tree Problem ist $\mathcal{NP}$-äquivalent. Diese Probleme haben gemeinsam, dass Leader und Follower jeweils eine Teilmenge einer gewissen Menge von Elementen kontrollieren und dass beide gemeinsam eine zulässige Lösung des zugrunde liegenden einstufigen Problems erzeugen, wobei sie verschiedene Zielfunktionen optimieren.

Im Fall des Bilevel Minimum Spanning Tree Problems untersuchen wir außerdem die Approximierbarkeit sowie den Spezialfall, in dem die Anzahl der Follower-Kanten klein ist. Weiterhin charakterisieren wir die Komplexität von Problemvarianten, die Engpass-Zielfunktionen beinhalten.

Neben der Komplexität von Bilevel-Optimierungsproblemen untersuchen wir auch die zusätzliche Komplexität, die Unsicherheit in Bilevel-Problemen erzeugt. In unserer Definition von robuster Bilevel-Optimierung nehmen wir an, dass der Leader keine vollständigen Informationen über die Zielfunktion des Followers hat und den schlechtesten Fall unter den möglichen Reaktionen des Followers optimiert. Wir betrachten mehrere Arten von Unsicherheitsmengen, die üblicherweise in der robusten Optimierung verwendet werden, wie diskrete Unsicherheit, Intervall-Unsicherheit und diskret unkorrelierte Unsicherheit, und untersuchen den komplexitätstheoretischen Effekt, den die Robustheit auf das zugrunde liegende Bilevel-Problem hat.

Wir stellen Komplexitätsergebnisse für eine Klasse von robusten linearen Bilevel-Problemen mit diskreten Leader- und stetigen Follower-Entscheidungen vor und führen eine gründliche Analyse von robusten Varianten des Bilevel Selection Problems und des Bilevel Continuous Knapsack Problems, die beide ohne Unsicherheit in polynomieller Zeit lösbar sind, durch. Manche der robusten Varianten sind weiterhin in polynomieller Zeit lösbar, während andere $\mathcal{NP}$-schwer sind. Die Ergebnisse zeigen mehrere Unterschiede zwischen robuster Bilevel-Optimierung und robuster einstufiger Optimierung auf. Beispielsweise ist Intervallunsicherheit in der robusten Bilevel-Optimierung nicht trivial und eine Unsicherheitsmenge kann im Allgemeinen nicht durch ihre konvexe Hülle ersetzt werden, ohne das Problem zu verändern.

Acknowledgments

This thesis would not have been possible without the great work and support of many people.

First and foremost, I would like to thank my advisor Christoph Buchheim for introducing and leading me into the topics of my research as well as into the world of academia, for many hours of jointly exploring interesting research questions and discussing details, and for all his encouragement and advice throughout the last years.

I thank my co-authors Christoph Buchheim and Felix Hommelsheim for their dedication and our collaboration on the articles this thesis is based on, as well as Kristina Hartmann for the work in her master's thesis that provided the basis of the research presented in Chapter 3.

Moreover, I am thankful to everyone who proofread parts of this thesis, in particular Christoph Buchheim, Alexandra Grütering, and Maja Hügging. My thanks also go to Britta Peis for agreeing to be the second reviewer of this thesis.

I gratefully acknowledge the financial support by the Deutsche Forschungsgemeinschaft (German Research Foundation) within the project *Strategic planning of seaport hinterland networks with focus on LCL shipment consolidation in gateways*.

I would like to express my gratitude to Marc Goerigk for offering me a postdoc position, for trusting and supporting me, even though I had not finished my thesis yet, for new perspectives, and for the interesting research we have started to work on together.

A special thanks goes to all my former and current colleagues both in Dortmund and in Passau for providing a pleasant working atmosphere and contributing to me feeling at home in both places now. Thank you all for your company and conversations, for lunch and coffee breaks and social events.

I am grateful that I have already had the opportunity to work on so many different research projects with many different collaborators. I thank all of them for the joint research and for everything I learned from them. Moreover, I thank Bettina Klinz and Lasse Wulf for enabling an enjoyable and productive research stay in Graz.

During the last year, I had many encouraging and insightful conversations with my mentor Rebecca Waldecker, thank you too.

Last but not least, I thank my family and friends for being on my side during the time of working on this thesis. Most of all, I thank Ludger for his incredibly huge support and patience.

Contents

1 Introduction **1**

2 Preliminaries **5**
2.1 Computational Complexity Theory . . . 5
2.1.1 Basic Notions . . . 6
2.1.2 Approximation Algorithms . . . 13
2.1.3 Parameterized Complexity . . . 14
2.1.4 The Polynomial-Time Hierarchy . . . 15
2.2 Bilevel Optimization . . . 17
2.2.1 Basic Definitions . . . 18
2.2.2 Literature Overview . . . 22
2.3 Robust Optimization . . . 26
2.3.1 Basic Definitions . . . 27
2.3.2 Types of Uncertainty Sets . . . 28
2.3.3 General Observations . . . 31
2.3.4 Literature Overview of Robust Combinatorial Optimization Problems 33
2.4 Bilevel Optimization Under Uncertainty . . . 34
2.4.1 Introduction to Robust Bilevel Optimization with Uncertain Follower's Objective . . . 34
2.4.2 Complexity of Robust Bilevel Optimization with Uncertain Follower's Objective . . . 36

3 The Bilevel Selection Problem **45**
3.1 The Selection Problem . . . 45
3.2 Definition and Variants of the Bilevel Selection Problem . . . 47
3.2.1 The Basic Problem Variant . . . 47
3.2.2 Assumptions About the Item Sets . . . 50
3.2.3 Assumptions About Binary and Continuous Decisions . . . 52
3.2.4 Assumptions About the Capacity Constraint . . . 52
3.2.5 Equivalence Between Problem Variants . . . 53
3.3 The Bilevel Selection Problem Without Uncertainty . . . 59
3.3.1 A Polynomial-Time Algorithm . . . 59
3.3.2 Structure of the Leader's Objective Function Under Assumption (disj) 62
3.4 The Robust Bilevel Selection Problem . . . 64
3.4.1 The Robust Bilevel Selection Problem Under Assumption (disj) . . . 68
3.4.2 The Robust Bilevel Selection Problem Under Assumption (sub) . . . 95
3.5 Conclusion . . . 109

4 The Bilevel Continuous Knapsack Problem **111**
4.1 The Knapsack Problem . . . 112

4.2 Definition and Variants of the Bilevel Knapsack Problem . . . 114
4.2.1 Basic Properties of the Bilevel Continuous Knapsack Problem . . . 116
4.2.2 Discussion of the Variants of the Bilevel Knapsack Problem . . . 117
4.2.3 Complexity of the Bilevel Knapsack Problem Under Assumptions (bb) and ($\leq'$) and Related Literature . . . 121
4.3 The Bilevel Continuous Knapsack Problem Without Uncertainty . . . 123
4.4 The Robust Bilevel Continuous Knapsack Problem . . . 127
4.4.1 Discrete Uncertainty . . . 132
4.4.2 Interval Uncertainty . . . 133
4.4.3 Discrete Uncorrelated Uncertainty . . . 140
4.4.4 Simplicial Uncertainty . . . 152
4.4.5 Norm Uncertainty . . . 157
4.5 Conclusion and Related Problems . . . 159

5 The Bilevel Minimum Spanning Tree Problem 161
5.1 Graph Notions and the Minimum Spanning Tree Problem . . . 162
5.2 Introduction of the Bilevel Minimum Spanning Tree Problem . . . 163
5.2.1 Problem Definition and Basic Properties . . . 163
5.2.2 Applications . . . 166
5.2.3 Related Work and Overview of Results . . . 167
5.3 Restricted Problem Variants . . . 168
5.4 Equivalence Between Problem Variants . . . 170
5.5 Main Hardness Results . . . 175
5.6 Fixed-Parameter Tractability . . . 180
5.7 An Approximation Algorithm . . . 186
5.8 Problem Variants with Bottleneck Objectives . . . 188
5.8.1 Introduction of the Bottleneck Problem Variants . . . 188
5.8.2 Tractability Results . . . 190
5.8.3 Hardness Results . . . 192
5.9 Conclusion . . . 195

6 Conclusions, Generalizations, and Outlook 197
6.1 The Bilevel Minimum Matroid Basis Problem . . . 198
6.2 Complexity of Robust Bilevel Optimization . . . 202
6.3 Open Questions . . . 204
6.3.1 Open Questions From Chapter 3 . . . 204
6.3.2 Open Questions From Chapter 4 . . . 205
6.3.3 Open Questions From Chapter 5 . . . 206
6.3.4 Other Open Questions . . . 207

Bibliography 209

1 Introduction

In this thesis, we combine the three research fields of *bilevel optimization*, *robust optimization*, and *computational complexity theory*.

Classical optimization problems involve only a single decision maker, which is not a good model for many real-world situations. Some settings are more appropriately modeled through *bilevel optimization*. Here, two decision makers each have their own optimization problem with constraints and an objective function, but each of the problems may involve also the other player's decision variables. The players are arranged in a hierarchical order and, accordingly, called the *leader* and the *follower*. First, the leader takes her decision, and after that, the follower reacts to this decision, which he considers fixed. The follower's decision in turn has an impact on the leader's objective function value. In order to optimize her objective function, the leader therefore has to anticipate the follower's reaction.

Robust optimization is one of several concepts for addressing uncertainty in decision making. Also here, the goal is to achieve more realistic models than by classical optimization problems. Some problem parameters are assumed to be unknown at the time when the decision is taken and only revealed later. However, a set of possible values for the unknown parameters is given and the worst case among these scenarios is to be optimized.

In particular, including uncertainty naturally suggests itself in bilevel optimization, since it appears to be a strong assumption that both decision makers have full knowledge of each other's data. More specifically, it might be more realistic that the leader is uncertain about the precise follower's preferences, i.e., the follower's objective function. Therefore, we study robust bilevel optimization problems where we take this type of uncertainty in the interaction between the two decision makers into account.

Due to their hierarchical structure, bilevel optimization problems, both with and without uncertainty, can be very hard to solve. In this thesis, we are interested in quantifying their complexity. Our focus will be on *combinatorial optimization*, where a useful tool for describing the hardness of problems is *computational complexity theory*.

Since the literature on the computational complexity of combinatorial bilevel optimization problems is still relatively limited, we contribute results on three bilevel problems that are natural extensions of fundamental single-level combinatorial problems. The Bilevel Selection Problem, the Bilevel Continuous Knapsack Problem, and the Bilevel Minimum Spanning Tree Problem are all *partitioned-items bilevel problems*. Here, a set of items is partitioned into leader's items and follower's items, and both players have to build a feasible solution to the underlying classical combinatorial optimization problem, while each decision maker has their own cost function on the items.

Bilevel problems (with discrete decisions and linear objectives and constraints) naturally fall into the second level of the polynomial-time hierarchy. But specific bilevel optimization problems can also be easier than that. We show that the Bilevel Minimum Spanning Tree Problem is $\mathcal{NP}$-equivalent, while the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem are even solvable in polynomial time. In this sense, all three problems are easier than general bilevel optimization problems.

A reason for our interest in relatively easy bilevel problems lies in the main question we are approaching with this thesis: How much more complex does a bilevel optimization problem become with added uncertainty? While bilevel optimization under uncertainty is a recently growing area of research, the perspective of computational complexity theory is still underrepresented. We try to enhance the understanding of the complexity of robust bilevel optimization. To this end, we start the complexity studies with the two polynomial-time solvable bilevel optimization problems mentioned above, the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem. We investigate several variants of them and obtain complexity results for their robust versions for different types of uncertainty sets. It will turn out that some of the robust problems can still be solved in polynomial time, while others are $\mathcal{NP}$-hard. This gives some insight into the effect that the uncertainty has on the bilevel problems. Moreover, we will draw conclusions about properties of robustness in bilevel problems, in contrast to single-level robust optimization. In addition to our studies of specific bilevel problems, we also present some results on the complexity of robust bilevel optimization for a more general class of underlying bilevel problems, in which the leader takes a discrete decision and the follower takes a continuous decision.

In the following, we give an overview of the contents of each of the chapters in this thesis and describe their relation to the publications the thesis is based on.

In Chapter 2, we introduce the preliminaries required for the research that is presented in this thesis. We cover the relevant notions and concepts as well as an overview of foundational and related literature regarding the topics of computational complexity theory, bilevel optimization, and robust optimization. We then turn to bilevel optimization under uncertainty, where, after a short introduction, we present complexity results for a general class of robust bilevel optimization problems in Section 2.4.2. We investigate discrete uncertainty and interval uncertainty, and our results show that both types of uncertainty can add complexity to the underlying bilevel problem, but behave differently than for single-level robust optimization. The results in Section 2.4.2 have been published and presented in a very similar way in [BHH21].

Chapter 3 deals with the Bilevel Selection Problem and its uncertain version, the Robust Bilevel Selection Problem. We introduce and compare several different problem variants and obtain a variety of complexity results for them. For all variants, we show that the Bilevel Selection Problem without uncertainty can be solved in polynomial time. In the Robust Bilevel Selection Problem, we investigate discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty. For some of the problem variants, we are able to develop polynomial-time algorithms in all three robust settings. For some of the more general problem variants, we present an $\mathcal{NP}$-hardness proof, a 2-approximation algorithm, and an exponential-time algorithm.

All main results that are presented in Chapter 3 have been published in [Hen25]. However, the discussion of problem variants is more extensive in this thesis, and we give more details on some proofs and remarks. Abstracts of this work have appeared in the Dagstuhl Report [BBHH23] and the Oberwolfach Report [LSW23]. Some of the results have been obtained jointly with Kristina Hartmann and have already appeared, in a similar form, in her master's thesis [Har21], which the author of this thesis co-supervised.

In Chapter 4, we study the Bilevel Continuous Knapsack Problem and the Robust Bilevel Continuous Knapsack Problem, which can be seen as generalizations of some problem variants from Chapter 3. Again, we obtain complexity results for the problem without uncertainty as well as for several robust versions, with different types of uncertainty

sets. While the algorithmic results for the certain problem and for the robust problems with discrete uncertainty and interval uncertainty from Chapter 3 can be generalized, discrete uncorrelated uncertainty as well as budgeted, polytopal, and ellipsoidal uncertainty turn out to lead to $\mathcal{NP}$-hard problems. However, a generalization of a polynomial-time algorithm from Chapter 3 gives a pseudopolynomial-time algorithm in case of discrete uncorrelated uncertainty.

Most of the main results in Chapter 4 have been published in [BH22], and the results regarding the pseudopolynomial-time algorithm in Section 4.4.3.2 have appeared in [Hen25]. Again, some discussion on problem variants, some proofs and remarks contain more details in this thesis than in the mentioned articles. In addition, we consider a slightly more general problem version in this thesis and thus strengthen the algorithmic results by showing that they also hold here. Moreover, the results are presented in a different way in Chapter 4 than in [BH22]: While [Hen25] has appeared after [BH22] and therefore presents the results for the Robust Bilevel Selection Problem as an extension of the ones in [BH22], we choose the opposite route here and build the results for the Robust Bilevel Continuous Knapsack Problem on top of the ones for the corresponding variant of the Bilevel Selection Problem. An extended abstract of parts of [BH22] has been published in [BH19b].

Chapter 5 is about the Bilevel Minimum Spanning Tree Problem. Here, we investigate the computational complexity of several different variants of the problem without uncertainty. We show that the main problem variant as well as several special cases of it are $\mathcal{NP}$-hard, and that the problem can be $(|V(G)|-1)$-approximated. We consider the setting where the number of edges controlled by the follower is fixed, where we show that a special case of the problem can be solved in polynomial time, the general problem can be 2-approximated, and we establish a relation to the Shortest Vertex-Disjoint Paths Problem. Finally, we completely characterize the complexity of all problem variants where at least one decision maker has a bottleneck instead of a sum objective function. Chapter 5 is based on and presented very similarly as in [BHH22].

The final Chapter 6 summarizes the most important findings of this thesis, in particular regarding the complexity of robust bilevel optimization. We also discuss connections between and possible generalizations of the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem, which can both be seen as special cases of the Bilevel Minimum Matroid Basis Problem. Finally, we discuss open questions that arise from the results in this thesis.

Besides the four articles this thesis is based on, the author has also been involved in research on several related topics. The results from these projects have appeared in [BHI22; HW25; HLST25a; HLST25b] and will be discussed in the appropriate sections of this thesis; see Remark 2.23 and Sections 2.2.2.2 and 4.5.

2 Preliminaries

This chapter gives an overview of the basic concepts and terminology that will be used throughout this thesis, and provides some background and references to the literature. First, Section 2.1 introduces the notions from computational complexity theory that we will work with. Section 2.2 deals with our central topic; it defines bilevel optimization problems and reviews the literature and some known results from this field. The other main area that is relevant for the work in this thesis is optimization under uncertainty, in particular robust optimization, which is introduced in Section 2.3. Finally, Section 2.4 deals with the combination of the two concepts, i.e., bilevel optimization under uncertainty. After a general overview, Section 2.4.2 contains some original research on the complexity of robust bilevel optimization problems, which has also appeared in [BHH21].

Notation. We denote by $\mathbb{N}$ the set of all natural numbers (without 0), by $\mathbb{Z}$ the set of all integers, by $\mathbb{Q}$ the set of all rational numbers, and by $\mathbb{R}$ the set of all real numbers. Moreover, we set $\mathbb{N}_0 \coloneqq \mathbb{N} \cup \{0\}$ and, for $\mathbb{K} \in \{\mathbb{Q}, \mathbb{R}\}$, $\mathbb{K}_{\geq 0} \coloneqq \{r \in \mathbb{K} \mid r \geq 0\}$, and $\mathbb{K}_{>0} \coloneqq \{r \in \mathbb{K} \mid r > 0\}$. For a number $r \in \mathbb{R}$, we write $\lfloor r \rfloor = \max\{z \in \mathbb{Z} \mid z \leq r\}$ and $\lceil r \rceil = \min\{z \in \mathbb{Z} \mid z \geq r\}$ for the integers that are closest to r.

Notation. For numbers $a, b \in \mathbb{R}$, we denote by $(a, b) \coloneqq \{r \in \mathbb{R} \mid a < r < b\}$ and $[a, b] \coloneqq \{r \in \mathbb{R} \mid a \leq r \leq b\}$ the open and closed interval between a and b, respectively.

Notation. For a finite set $\mathcal{E}$ and a set $A \subseteq \mathbb{R}$, we denote by $A^{\mathcal{E}}$ the set of all functions $c \colon \mathcal{E} \to A$. We will usually treat its elements as functions, e.g., by writing $c(e)$ for $e \in \mathcal{E}$. Sometimes, in particular when it comes to variables $x \in A^{\mathcal{E}}$, we will also treat them as vectors indexed by the elements of $\mathcal{E}$ instead and write x_e for $e \in \mathcal{E}$.

Notation. For a finite set $\mathcal{E}$, a function $c \colon \mathcal{E} \to \mathbb{R}$, and a set $X \subseteq \mathcal{E}$, we write $c(X) = \sum_{e \in X} c(e)$.

Notation. For sets A, B, a subset $A' \subseteq A$, and a function $c \colon A \to B$, we denote by $c \restriction A'$ the function $c' \colon A' \to B$ that is the restriction of c to A', i.e., $c'(a) = c(a)$ for all $a \in A'$.

2.1 Computational Complexity Theory

A main goal of this thesis is to investigate how hard some specific types of optimization problems are to solve. In order to formally talk about this, we first need to introduce some basics from computational complexity theory. This section gives an overview of terms that will be used later on, however, sometimes being a bit informal. For a more thorough introduction, we refer to the literature mentioned in the beginning of each of the following sections.

Section 2.1.1 introduces the basic notions of complexity theory, such as algorithms, problems, and the complexity classes $\mathcal{P}$ and $\mathcal{NP}$. Section 2.1.2 deals with some terms regarding approximation algorithms for optimization problems. Finally, Sections 2.1.3

and 2.1.4 introduce further complexity classes, which help to describe more aspects of the complexity of a problem, beyond $\mathcal{P}$ and $\mathcal{NP}$. In particular, we will work with the concepts of parameterized complexity and the polynomial-time hierarchy. The latter is of fundamental relevance when dealing with problems involving several decision makers, such as bilevel optimization.

2.1.1 Basic Notions

This section mostly follows the presentation of the basic complexity concepts in [KV18]. For a complete and more formal introduction, we refer to [KV18, Chapters 1.2, 15] and also to, e.g., [GJ79], [GLS93, Chapter 1], and [Sch03, Chapter 4].

2.1.1.1 Algorithms and Running Times

An *algorithm* consists of a set of possible inputs and some instructions that can be uniquely interpreted, for every feasible input, as a finite sequence of elementary steps that lead to some output. In general, the *input* and the *output* can consist of any mathematical objects. In an implementation of the algorithm on a computer, they are usually encoded as binary numbers, i.e., finite sequences of 0's and 1's *(bits)*. Hence, it is necessary that the mathematical objects actually have such a finite representation. For example, a number $z \in \mathbb{Z}$ can be encoded in binary using $\lceil \log_2(|z| + 1) \rceil + 1$ bits, and for a rational number, the numerator and the denominator are encoded separately. On the other hand, arbitrary real numbers cannot be represented exactly on a computer. Therefore, we will mostly follow a usual convention in the literature and work with rational numbers in the description of algorithms, i.e., numbers in inputs and outputs of algorithms (and also problems later on) are usually assumed to be rational. This is not a severe restriction because every irrational number can be approximated arbitrarily well by a rational number, and has to be approximated on a computer anyway. However, also sets of real numbers will sometimes occur, e.g., when defining continuous variables or objective functions. In such cases, dealing with reals is more common and convenient, and, under reasonable assumptions, it is still true that the corresponding algorithms only need to work with rational numbers. See, e.g., [GLS93, section 1.3] for more details.

The *size* of an algorithm's input I, also denoted by $\text{size}(I)$, is the length of its binary representation. Relative to this input size, algorithms are usually sought to be fast. In order to make this more precise, the *running time* of an algorithm is formalized by a function that gives, for every possible input size, an upper bound on the number of elementary steps the algorithm performs. The definition of an *elementary step*, and therefore also the running time, depends on the underlying machine model. We will work with the popular model of a *random-access machine (RAM)*, for which arithmetic operations on integers can be performed in a constant number of elementary steps. While alternative machine models (such as Turing machines) can lead to different running times, the difference is assumed to be not too large: According to the informal conjecture of the *extended Church-Turing thesis*, all "reasonable" machine models (of a classical, non-quantum computer) can simulate each other with a polynomially-bounded overhead in running time and space. For more details on machine models, we refer to [Emd90].

Since the exact running time of an algorithm depends on details of the implementation and the machine model, one usually ignores constant factors when stating running times.

This can be formalized using the so-called $\mathcal{O}$-*notation*.

Definition 2.1. Given a set D and two functions $f, g\colon D \to \mathbb{R}_{\geq 0}$, we say that f *is (in)* $\mathcal{O}(g)$ if there are constants $\alpha, \beta \in \mathbb{R}_{>0}$ such that $f(z) \leq \alpha g(z) + \beta$ for all $z \in D$.

Using this notation, we will often make statements about an algorithm's running time, as a function of its input size n, being, for example, $\mathcal{O}(n^2)$ or $\mathcal{O}(2^n)$. Sometimes, the most relevant information is whether an algorithm runs in *polynomial time*, i.e., in $\mathcal{O}(n^k)$ for some constant $k \in \mathbb{N}$ (while using at most $\mathcal{O}(n^k)$ bits of storage). Such an algorithm is also called *efficient*. Note that this notion is independent of the machine model according to the extended Church-Turing thesis.

Algorithms, as introduced above, can be used to solve *computational problems*. A computational problem consists of a set $\mathcal{I}$ of *instances* and a function $f\colon \mathcal{I} \to \mathcal{Z}$ for some set $\mathcal{Z}$ of possible outputs. An algorithm *solves* this problem, or *computes* the function f, if it accepts all instances $I \in \mathcal{I}$ as possible inputs and, after a finite running time, outputs the corresponding value $f(I)$. Like the inputs and outputs of algorithms, also instances and possible outputs of computational problems are usually assumed to have finite representations in binary numbers. A computational problem is called *tractable* if there is an efficient algorithm that solves it.

We are mostly interested in solving specific types of computational problems, namely decision and optimization problems, which will be introduced in Sections 2.1.1.2 and 2.1.1.3.

2.1.1.2 Decision Problems and Basic Complexity Classes

Informally, a decision problem is any (mathematically well-defined) question that can be answered by "no" or "yes", or, equivalently, a computational problem with possible output values $\mathcal{Z} = \{0, 1\}$ (corresponding to "no" and "yes"). More formally, it can be defined as follows.

Definition 2.2. An *alphabet* is a finite set of size at least two. Given an alphabet A, a *string* over A is a finite sequence of elements of A. We denote the set of all strings over A by A^*. A *language* over A is a subset of A^*. A *decision problem* $P = (\mathcal{I}, \mathcal{Y})$ consists of a language $\mathcal{I}$ over some alphabet A such that $\mathcal{I}$ is decidable in polynomial time (i.e., there is an algorithm that runs in polynomial time and correctly outputs, for any input $I \in A^*$, whether $I \in \mathcal{I}$ or not) and a subset $\mathcal{Y} \subseteq \mathcal{I}$. The elements of $\mathcal{I}$ are called *instances* of P, the elements of $\mathcal{Y}$ are *yes-instances*, and the elements of $\mathcal{I} \setminus \mathcal{Y}$ are *no-instances*.

Note that Definition 2.2 requires that one can check in polynomial time whether some I is an instance of the problem or not. However, the problem's actual question, namely whether an instance is a yes- or a no-instance, can be much more complex to answer. We say that a decision problem is *solved* by some algorithm if this algorithm accepts any instance $I \in \mathcal{I}$ as a possible input and correctly outputs "yes" or "no" (usually encoded as 1 and 0) if the input is a yes- or a no-instance, respectively. An important question is which decision problems can be solved efficiently, i.e., in polynomial time.

Definition 2.3. The class $\mathcal{P}$ consists of all decision problems that can be solved by a polynomial-time algorithm.

For many problems, it is not known whether they belong to $\mathcal{P}$. A relevant weaker notion is the class $\mathcal{NP}$, which contains problems that cannot necessarily be solved in polynomial time, but for which a given solution can be checked in polynomial time.

Definition 2.4. The class $\mathcal{NP}$ consists of all decision problems $P = (\mathcal{I}, \mathcal{Y})$ for which there are a polynomial p and a decision problem $P' = (\mathcal{I}', \mathcal{Y}')$ that is contained in $\mathcal{P}$ and satisfies:

$$\mathcal{I}' = \{(I, c) \mid I \in \mathcal{I}, c \in \{0,1\}^{\lfloor p(\text{size}(I))\rfloor}\}$$
$$\mathcal{Y} = \{I \in \mathcal{I} \mid \text{there is a string } c \in \{0,1\}^{\lfloor p(\text{size}(I))\rfloor} \text{ with } (I, c) \in \mathcal{Y}'\}$$

A string c with $(I, c) \in \mathcal{Y}'$ is called a *certificate* for I.

In other words, with the help of some reasonable polynomial-size certificate attached to each (yes-)instance, it can be verified in polynomial time that a given yes-instance is indeed a yes-instance. For questions that ask for the existence of a mathematical object with certain properties (e.g., a Hamiltonian cycle in a graph or a partition of a set of numbers), this object can usually be directly encoded in the certificate, and the required properties can be checked in polynomial time. $\mathcal{NP}$ stands for "nondeterministically polynomial" and refers to an equivalent definition of $\mathcal{NP}$ via the existence of polynomial-time nondeterministic algorithms for solving decision problems. Informally, these algorithms can be imagined to "guess" (i.e., randomly generate) certificates and thus to correctly identify a yes-instance as a yes-instance if the guess was correct, and, correctly or incorrectly, return "no" otherwise.

Clearly, all problems in $\mathcal{P}$ are also contained in $\mathcal{NP}$. A major open question of theoretical computer science is whether $\mathcal{P} = \mathcal{NP}$ holds; the widely assumed conjecture is that this is not the case. In order to further classify the complexity of decision problems and, in particular, to formalize that one problem is not easier than another one, one makes use of the notion of polynomial transformations (also known as Karp reductions) between decision problems.

Definition 2.5. Given two decision problems $P_1 = (\mathcal{I}_1, \mathcal{Y}_1)$ and $P_2 = (\mathcal{I}_2, \mathcal{Y}_2)$, we say that P_1 *polynomially transforms* to P_2 if there is a function $f\colon \mathcal{I}_1 \to \mathcal{I}_2$ that can be computed in polynomial time and satisfies $f(I_1) \in \mathcal{Y}_2$ for all $I_1 \in \mathcal{Y}_1$ and $f(I_1) \in \mathcal{I}_2 \setminus \mathcal{Y}_2$ for all $I_1 \in \mathcal{I}_1 \setminus \mathcal{Y}_1$.

This means that yes-instances are transformed to yes-instances and no-instances are transformed to no-instances. Using this notion, we can formally define that a problem is at least as hard as all other problems in $\mathcal{NP}$.

Definition 2.6. A decision problem P is *$\mathcal{NP}$-complete* if it is contained in $\mathcal{NP}$ and all other problems in $\mathcal{NP}$ polynomially transform to P.

For proving the $\mathcal{NP}$-completeness of some decision problem $P \in \mathcal{NP}$, it suffices to transform some $\mathcal{NP}$-complete problem P' to P because the relation defined by polynomial transformations is clearly transitive. On the other hand, finding a polynomial-time algorithm for some $\mathcal{NP}$-complete problem would yield polynomial-time algorithms for all problems in $\mathcal{NP}$ and hence imply $\mathcal{P} = \mathcal{NP}$.

In order to prove $\mathcal{NP}$-completeness by a polynomial transformation from another $\mathcal{NP}$-complete problem, historically, some "first" $\mathcal{NP}$-complete problem with a different proof was needed. This problem is the well-known Satisfiability Problem and the fact that it is $\mathcal{NP}$-complete is known as *Cook's theorem* [Coo71]. We will work with this problem and extensions of it in Sections 2.1.4 and 2.4.2.

Satisfiability Problem
Instance: A Boolean formula $f\colon \{0,1\}^n \to \{0,1\}$
Task: Decide whether there exists a satisfying assignment, i.e., a $y \in \{0,1\}^n$ such that $f(y) = 1$.

Remark 2.7. One usually assumes that the Boolean formula is given in *conjunctive normal form*, i.e., as a conjunction of disjunctions of literals, where each *literal* corresponds to either a binary variable or the negation of a binary variable. Correspondingly, it is often written as a set of *clauses* each of which is a set of literals. For example, a Boolean formula $f\colon \{0,1\}^3 \to \{0,1\}$ could be defined by $f(y_1, y_2, y_3) = (y_1 \vee y_2 \vee \overline{y_3}) \wedge (\overline{y_1} \vee y_3) \wedge (\overline{y_2})$. In this example, a satisfying assignment is $(y_1, y_2, y_3) = (1, 0, 1)$. In particular, the Satisfiability Problem for Boolean formulas given in conjunctive normal form is still $\mathcal{NP}$-complete.

On the other hand, one can consider Boolean formulas in *disjunctive normal form*, i.e., as a disjunction of conjunctions of literals. Every Boolean formula can be written equivalently in conjunctive normal form and in disjunctive normal form, but, in both cases, the reformulation might result in a representation of exponential size. Therefore, it can be important to distinguish between conjunctive normal form and disjunctive normal form for complexity theoretic questions. In fact, if the input of the Satisfiability Problem is given in disjunctive normal form, the problem is easy to solve in linear time: We iterate over all conjunctions until we find one that can be satisfied, i.e., that does not contain both a variable and its negation. If such a conjunction is not found, we have verified that the formula is not satisfiable.

We will later discuss extensions of the Satisfiability Problem where we also have to be careful if the Boolean formula is given in conjunctive or disjunctive normal form; see Remark 2.19.

With the Satisfiability Problem as a starting point, a great number of decision problems could be proved to be $\mathcal{NP}$-complete; see [GJ79] for a collection of many fundamental $\mathcal{NP}$-complete problems.

Since the definition of $\mathcal{NP}$ is not symmetric with respect to yes- and no-instances, it can be useful to consider also complements of decision problems.

Definition 2.8. For a decision problem $P = (\mathcal{I}, \mathcal{Y})$, its *complement* is the decision problem $(\mathcal{I}, \mathcal{I} \setminus \mathcal{Y})$. The class *co-$\mathcal{NP}$* consists of all decision problems whose complements are contained in $\mathcal{NP}$. A decision problem P is *co-$\mathcal{NP}$-complete* if it is contained in co-$\mathcal{NP}$ and all other problems in co-$\mathcal{NP}$ polynomially transform to P.

It is an open question whether $\mathcal{NP} =$ co-$\mathcal{NP}$; the common conjecture is that this is not the case. If the answer is indeed "no", then no co-$\mathcal{NP}$-complete problem is contained in $\mathcal{NP}$.

So far, we have only introduced complexity classes for decision problems. Later on, we will mostly be interested in optimization problems, which can be seen as another special case of computational problems. We will define and discuss optimization problems in Section 2.1.1.3. In order to speak of their complexity as well, we will now introduce complexity notions for general computational problems. As a generalization of polynomial transformations (see Definition 2.5), we define polynomial reductions (also known as Turing reductions) for computational problems.

Definition 2.9. Given two computational problems P_1 and P_2, we say that P_1 *polynomially reduces* to P_2 if there is a polynomial-time oracle algorithm for P_1 that uses an oracle for P_2.

By an *oracle algorithm* we mean an algorithm that uses another algorithm (the *oracle*) as a subroutine, i.e., it may compute some feasible input of the oracle, call the oracle with this input, and continue working with the oracle's output. The oracle can also be called several times. We assume that the oracle solves the required computational problem correctly and count every oracle call as a single step in the running time of the main algorithm. Sometimes, when talking about oracle algorithms, we will also include the running time of the oracle as a parameter when stating the running time of the main algorithm; see, e.g., Theorem 3.25.

The complexity of computational problems relative to $\mathcal{NP}$ can now be characterized as follows.

Definition 2.10. A computational problem P is $\mathcal{NP}$*-hard* if all problems in $\mathcal{NP}$ polynomially reduce to P. A computational problem P is $\mathcal{NP}$*-easy* if P polynomially reduces to a problem in $\mathcal{NP}$. A computational problem P is $\mathcal{NP}$*-equivalent* if it is both $\mathcal{NP}$-hard and $\mathcal{NP}$-easy.

Similarly to the comment after Definition 2.6 above, $\mathcal{NP}$-hardness is usually proved by a polynomial reduction from some $\mathcal{NP}$-complete (or co-$\mathcal{NP}$-complete) problem. While we usually cannot expect to find a polynomial-time algorithm for an $\mathcal{NP}$-hard problem (as this would imply $\mathcal{P} = \mathcal{NP}$), an algorithm running in pseudopolynomial time exists for some of these problems (e.g., the Knapsack Problem, see Section 4.1).

Definition 2.11. An algorithm whose input consists of natural numbers runs in *pseudo-polynomial time* if its running time, as a function not only in the input size but also in the largest number of the input, is bounded by a polynomial in these two variables.

Such an algorithm can be fast in practice if instances are known to contain only small numbers. For this reason, it is useful to differentiate the complexity notions further as follows.

Definition 2.12. A computational problem P whose instances consist of natural numbers is *strongly* $\mathcal{NP}$*-hard* if there is a polynomial p such that the restriction of P to instances I in which all numbers are bounded by $p(\text{size}(I))$ is $\mathcal{NP}$-hard. If P is a decision problem that lies in $\mathcal{NP}$ and the above restriction of it is $\mathcal{NP}$-complete, P is *strongly* $\mathcal{NP}$*-complete*.

If a computational problem is $\mathcal{NP}$-hard (or $\mathcal{NP}$-complete), but not (known to be) strongly $\mathcal{NP}$-hard (or strongly $\mathcal{NP}$-complete), it is also called *weakly* $\mathcal{NP}$*-hard* (or *weakly* $\mathcal{NP}$*-complete*). Unless $\mathcal{P} = \mathcal{NP}$, there are no polynomial-time algorithms for $\mathcal{NP}$-hard problems and no pseudopolynomial-time algorithms for strongly $\mathcal{NP}$-hard problems.

2.1.1.3 Optimization Problems

Finally, we define what we mean by an optimization problem in this thesis. It can be interpreted as a special case of a computational problem; hence, the previously defined terms, such as $\mathcal{NP}$-hardness, are also applicable to optimization problems.

Definition 2.13. An *optimization problem* consists of a set $\mathcal{I}$ of *instances* that is a language over some alphabet A and decidable in polynomial time (see also Definition 2.2), a *feasible set* $\mathcal{X}_I$ of *feasible solutions* for every $I \in \mathcal{I}$, an *objective function* $\mathcal{G}\colon \{(I,x) \mid I \in \mathcal{I}, x \in \mathcal{X}_I\} \to \mathbb{R}$, and the information goal $\in \{\min, \max\}$ whether the problem is a *minimization problem* or a *maximization problem*. We say that an instance $I \in \mathcal{I}$ is *feasible* if it has a feasible solution, i.e., if $\mathcal{X}_I \neq \emptyset$. In short, we write $\mathrm{OPT}(I) = \mathrm{goal}\{\mathcal{G}(I,x) \mid x \in \mathcal{X}_I\}$ or

$$\begin{aligned} &\mathrm{goal}\ \mathcal{G}(I,x) \\ &\text{s.t. } x \in \mathcal{X}_I \end{aligned}$$

for the optimization problem and also for its *optimum value*, given some instance $I \in \mathcal{I}$. An *optimum solution* of an instance $I \in \mathcal{I}$ is a feasible solution $x^* \in \mathcal{X}_I$ that attains the optimum value, i.e., with $x^* \in \mathrm{arggoal}\{\mathcal{G}(I,x) \mid x \in \mathcal{X}_I\} = \{x \in \mathcal{X}_I \mid \mathcal{G}(I,x) = \mathrm{OPT}(I)\}$.

Note that, formally, we would need to define the optimum value by $\inf\{\mathcal{G}(I,x) \mid x \in \mathcal{X}_I\}$ or $\sup\{\mathcal{G}(I,x) \mid x \in \mathcal{X}_I\}$ here because it might happen that the optimum value is not attained; see also below. However, following common practice, we will always write "min" and "max".

One often distinguishes *discrete optimization problems* and *continuous optimization problems*, depending on whether the sets of feasible solutions and therefore the variables that can be used to represent them are discrete or continuous. Also a mixture of both is possible. In a *combinatorial optimization problem*, the set of feasible solutions is typically finite and described by a combinatorial structure, e.g., a graph. Fundamental examples of combinatorial optimization problems are the Selection Problem (see Section 3.1) and the Minimum Spanning Tree Problem (see Section 5.1), which form the basis of large parts of this work.

Another important class of optimization problems are *linear programs*. They can be written using continuous variables, linear objective functions and linear inequality constraints. The feasible set of a linear program is a *polyhedron* and it can be shown that, if the problem is feasible and bounded, it always has an optimum solution that is an extreme point of this polyhedron. A bounded polyhedron is called a *polytope*. Linear programs can be solved using the well-known simplex method, which is not guaranteed to run in polynomial time but works well in practice, or using the polynomial-time ellipsoid method. For an introduction to the theory of linear programming, we refer to [GLS93]. Whenever we deal with continuous optimization problems in this thesis, they will be (bilevel variants of) linear programs; see Sections 2.4.2, 3.2.3, and 4.2.

The discrete optimization problems that correspond to linear programs can be written using integer or binary variables, linear objective functions and linear inequality constraints. We will call them *integer linear programs* and *binary linear programs*. They are $\mathcal{NP}$-hard in general, as many $\mathcal{NP}$-hard combinatorial optimization problems can be modeled as binary linear programs. The *linear programming relaxation* of an integer linear program is the linear program that is obtained by omitting the integrality constraints. As a linear program, the relaxation is easier to solve and can be useful to get bounds on the optimum value of the original integer linear program.

Given an instance I of an optimization problem, there are the following possible outcomes:

- The instance is infeasible, i.e., $\mathcal{X}_I = \emptyset$.
- The instance is unbounded, i.e., the set $\{\mathcal{G}(I,x) \mid x \in \mathcal{X}_I\}$ is not bounded from below (above), in case of a minimization (maximization) problem.

- The instance has an optimum value but no optimum solution, i.e., the set $\{\mathcal{G}(I,x) \mid x \in \mathcal{X}_I\}$ has an infimum (supremum) but no minimum (maximum), in case of a minimization (maximization) problem.
- The instance has an optimum value and a (not necessarily unique) optimum solution.

In terms of a computational problem or an algorithm solving the problem, for a given instance, the task is to distinguish between these four cases and return the optimum value and an optimum solution if they exist. An algorithm that solves an optimization problem in this sense is also called an *exact algorithm*. In contrast to that, we will define approximation algorithms in Section 2.1.2 that do not necessarily solve the problem to optimality, but return a feasible solution with some guarantee regarding its quality compared to the optimum value.

In a combinatorial optimization problem with a finite set of feasible solutions for every instance, only two of the above four cases can occur: Either an instance is infeasible or there is an optimum solution. Large parts of this thesis will deal with such problems. Also in linear programming, the third case is excluded.

However, also more complicated (bilevel) optimization problems with continuous variables will occur and, while all of our problems are still bounded, we will discuss examples where the optimum value is not attained; see, e.g., Lemma 3.12. In bilevel optimization, one has to be particularly careful about this issue because nonconvex sets may occur in settings in which this is usually not the case for single-level optimization problems, such as linear programming; see also Section 2.2.1.

Another issue that needs to be discussed for optimization problems as defined above is that the instances and the output of a computational problem need to have a finite representation in binary numbers and should, in particular, not contain arbitrary real numbers; see Section 2.1.1.1. For combinatorial optimization problems, this is often easily achieved by requiring all objective function values to be rationals, which is usually not a restriction, and we will always do this when defining a combinatorial optimization problem in this thesis. For general optimization problems, one has to be more careful. In order to obtain a well-defined computational problem, which can possibly be solved exactly by an algorithm having a finite running time, we need to require that, besides the instances, also optimum values and optimum solutions, as well as all values that an algorithm might need to handle in order to find an optimum, have some finite representation. This is usually the case and implicitly clear for the optimization problems in this thesis, even if they are defined or discussed using continuous variables and real objective functions. For example, in order to work with linear programs according to this definition, it usually suffices that their parameters are rational numbers; see [GLS93].

Remark 2.14. When defining (combinatorial) optimization problems whose complexity is to be analyzed, it can be reasonable to assume (and this is done in [KV18, Definition 15.35]) that the feasibility of a solution can be checked in polynomial time and that the objective function can be computed in polynomial time. Then every such optimization problem is $\mathcal{NP}$-easy [KV18, Theorem 15.36]. However, we do not make these assumptions here because they are often not satisfied for bilevel optimization problems with which we will be concerned in this thesis. In fact, bilevel optimization problems are not $\mathcal{NP}$-easy in general, unless $\mathcal{NP} = \Sigma_2^{\mathrm{p}}$; see Section 2.2.2.1.

2.1.2 Approximation Algorithms

When solving an $\mathcal{NP}$-hard optimization problem, we cannot expect to find an efficient exact algorithm, as this would imply $\mathcal{P} = \mathcal{NP}$. There are two ways to deal with this situation: Either we solve the problem using an exact algorithm whose running time is not polynomial, or we relax the goal of solving the problem exactly, i.e., we aim for a polynomial-time algorithm returning a feasible solution that should be as good as possible, although we cannot guarantee it to be optimum. In practice, finding a reasonably good solution often suffices. Then it is still interesting to quantify the quality of such a solution. For an approximation algorithm, we can bound how much worse than the optimum value the returned solution is at most. This section only deals with the basic concepts regarding approximation algorithms that are used in this thesis and is based on [KV18, Chapter 16] and [WS11, Chapter 16]; for a more thorough introduction, we refer to these textbooks.

Bounding the absolute difference of the solution values is not possible in many cases. Hence, one usually tries to bound the relative error the algorithm makes compared to the optimum value. This notion is only reasonable if the solution values that are compared have the same sign. Therefore, we only consider approximation algorithms for optimization problems with nonnegative objective functions.

Definition 2.15. Let P be an optimization problem whose objective function is nonnegative for all instances, and let $k \in \mathbb{R}$ be some number with $k \geq 1$. A *k-approximation algorithm* for P is an algorithm that accepts all instances of P as possible inputs, has polynomial running time and, for every feasible and bounded instance I of P, returns a feasible solution This thesis was only possible because of the great work and support of many people. whose objective function value $\mathcal{A}(I)$ satisfies $1/k \operatorname{OPT}(I) \leq \mathcal{A}(I) \leq k \operatorname{OPT}(I)$. We also say that the algorithm has an *approximation factor* or *approximation guarantee* of k.

The definition captures both minimization and maximization problems, while the first inequality is only relevant for maximization problems and the second inequality is only relevant for minimization problems. Note that 1-approximation algorithms are exact and that, for instances I with $\operatorname{OPT}(I) = 0$, every approximation algorithm must return an optimum solution.

As an extension of Definition 2.15, one can also allow the approximation factor to be a function of the instance instead of a constant. We will see such an approximation algorithm for the Bilevel Minimum Spanning Tree Problem in Theorem 5.37. Typically, such an approximation is weaker than one with a constant factor, as the factor grows with the size of the instance.

In the best case, a small approximation factor (i.e., close to 1) can be achieved. On the other hand, for some $\mathcal{NP}$-hard optimization problems, it is even very hard to approximate them well. More formally, it can be $\mathcal{NP}$-hard to approximate a problem within some factor, i.e., finding an approximation algorithm with this or a better factor would prove $\mathcal{P} = \mathcal{NP}$. In order to prove such results on the hardness of approximation, *approximation-preserving reductions* are useful. In contrast to general polynomial reductions (see Definition 2.9), they do not only imply that a polynomial-time algorithm for problem P_2 could be used to construct a polynomial-time algorithm for problem P_1, but even that a k-approximation algorithm for problem P_2 would imply a k-approximation algorithm for problem P_1. More generally, one can also consider reductions that turn a k-approximation algorithm for P_2 into an $f(k)$-approximation algorithm for P_1, for some appropriate function $f\colon \mathbb{R} \to \mathbb{R}$.

For more formal definitions, we refer to [KV18, Chapter 16] and [WS11, Chapter 16].

Very strong inapproximability results can sometimes be obtained as follows. If it is $\mathcal{NP}$-hard to decide whether the optimum value of some optimization problem is zero or strictly positive, then the existence of any approximation algorithm (i.e., with any constant or non-constant approximation factor) is excluded, unless $\mathcal{P} = \mathcal{NP}$. Indeed, any approximation algorithm for such a problem would need to return a feasible solution having value zero or a positive value, respectively, in the two cases. This argument will be used, e.g., in Theorem 5.45.

2.1.3 Parameterized Complexity

When dealing with $\mathcal{NP}$-hard optimization problems, we can not only try to approximate their optimum value, as described in the previous section, but we can also look for exact algorithms, without requiring polynomial running times. To this end, it can be reasonable to study running times more carefully and relative to specific characteristics of the input and not only to the total input size. In particular, only one parameter of an instance might be responsible for an exponential running time, and in case this parameter is small in practice, the running times might be very good. This idea leads to the concept of parameterized complexity. This section is based on [Cyg+15, Chapter 1.1], and we again only focus on the concepts required for the results in this thesis; see [Cyg+15] for a complete introduction into the theory of parameterized complexity.

Definition 2.16. A *parameterized problem* is a computational problem $(\mathcal{I}, f)$ together with a *parameter* $k\colon \mathcal{I} \to \mathbb{N}$. A parameterized problem $(\mathcal{I}, f, k)$ is *fixed-parameter tractable ($\mathcal{FPT}$)* if there are a computable function $g\colon \mathbb{N} \to \mathbb{N}$, a constant $c \in \mathbb{N}$, and an algorithm solving $(\mathcal{I}, f)$ whose running time is bounded by $g(k(I))(\text{size}(I) + k(I))^c$ for every instance $I \in \mathcal{I}$. A parameterized problem $(\mathcal{I}, f, k)$ is *slice-wise polynomial ($\mathcal{XP}$)* if there are computable functions $g, h\colon \mathbb{N} \to \mathbb{N}$ and an algorithm solving $(\mathcal{I}, f)$ whose running time is bounded by $g(k(I))(\text{size}(I) + k(I))^{h(k(I))}$ for every instance $I \in \mathcal{I}$. We also say that the problem $(\mathcal{I}, f)$ is *fixed-parameter tractable* or *slice-wise polynomial in k*.

The parameter can describe, for example, the size of some part of the instance or the size of an optimum solution. It may occur in some (usually superpolynomial) factor $g(k(I))$ in the running time and, in case of slice-wise polynomial problems, also in the exponent $h(k(I))$ of the polynomial part of the running time. The term $\text{size}(I) + k(I)$ can be imagined as the instance size of the parameterized problem, where the parameter is seen as a part of the instance and encoded in unary. The functions g and h are usually assumed to be nondecreasing, which is also reasonable in practice.

Clearly, every parameterized problem that is fixed-parameter tractable is also slice-wise polynomial. In both cases, the running time is polynomial for every fixed value of the parameter, but the exponent of the polynomial may change with different parameter values in the latter case.

The two approaches presented in Sections 2.1.2 and 2.1.3 can also be combined, resulting in algorithms that compute an approximate solution to some optimization problem in the sense of Definition 2.15, while having a running time that may depend on a parameter as in Definition 2.16. For example, in Theorem 5.36, we will show a fixed-parameter tractability result for the problem of 2-approximating the Bilevel Minimum Spanning Tree Problem.

2.1.4 The Polynomial-Time Hierarchy

Beyond the complexity classes $\mathcal{P}$, $\mathcal{NP}$, and co-$\mathcal{NP}$ introduced in Section 2.1.1, one can define a whole *polynomial-time hierarchy* of higher complexity classes that extends these three notions. These higher levels are useful for classifying decision and optimization problems having a more complicated structure. In particular, bilevel optimization problems (see Section 2.2) naturally fall into the second level of the hierarchy. The notions and basic results concerning the polynomial-time hierarchy are due to Meyer, Stockmeyer, and Wrathall [MS72; SM73; Sto76; Wra76]. We also refer to [GJ79, Chapter 7.2] and [Joh11; Woe21] for an introduction to the topic.

The polynomial-time hierarchy consists of the classes Δ_k^{p}, Σ_k^{p}, and Π_k^{p}, for all $k \in \mathbb{N}_0$, which generalize the classes $\mathcal{P}$, $\mathcal{NP}$, and co-$\mathcal{NP}$ (see Section 2.1.1.2) in the following way:

Definition 2.17. Let $\Delta_0^{\mathrm{p}} = \Sigma_0^{\mathrm{p}} = \Pi_0^{\mathrm{p}} = \mathcal{P}$. For $k \in \mathbb{N}$, we define recursively: The class Δ_k^{p} consists of all decision problems that can be solved by a polynomial-time oracle algorithm using an oracle for some problem in $\Sigma_{k-1}^{\mathrm{p}}$. The class Σ_k^{p} consists of all decision problems P for which there are a polynomial p and a decision problem P' that satisfies the same conditions as in Definition 2.4, but, instead of being contained in $\mathcal{P}$, can be solved by a polynomial-time oracle algorithm using an oracle for some problem in $\Sigma_{k-1}^{\mathrm{p}}$. The class Π_k^{p} consists of all decision problems whose complements are contained in Σ_k^{p}.

The first stage of the polynomial-time hierarchy is consistent with the notions introduced in Section 2.1.1.2: We have $\Delta_1^{\mathrm{p}} = \mathcal{P}$, $\Sigma_1^{\mathrm{p}} = \mathcal{NP}$, and $\Pi_1^{\mathrm{p}} = \text{co-}\mathcal{NP}$. The definition of Σ_k^{p} works with certificates, analogously to the definition of $\mathcal{NP}$. However, we do not require that the certificate can be verified in polynomial time here, but the task of verifying the certificate may require an oracle for a problem in $\Sigma_{k-1}^{\mathrm{p}}$. Sometimes, the relations in Definition 2.17 are also written using the notation $\Delta_k^{\mathrm{p}} = \mathcal{P}^{\Sigma_{k-1}^{\mathrm{p}}}$, $\Sigma_k^{\mathrm{p}} = \mathcal{NP}^{\Sigma_{k-1}^{\mathrm{p}}}$, and $\Pi_k^{\mathrm{p}} = \text{co-}\mathcal{NP}^{\Sigma_{k-1}^{\mathrm{p}}}$.

The hierarchy satisfies, for all $k \in \mathbb{N}_0$, the fundamental inclusions

$$\Delta_k^{\mathrm{p}} \subseteq \Delta_{k+1}^{\mathrm{p}}, \quad \Sigma_k^{\mathrm{p}} \subseteq \Sigma_{k+1}^{\mathrm{p}}, \quad \Pi_k^{\mathrm{p}} \subseteq \Pi_{k+1}^{\mathrm{p}}, \quad \Sigma_k^{\mathrm{p}} \cup \Pi_k^{\mathrm{p}} \subseteq \Delta_{k+1}^{\mathrm{p}} \subseteq \Sigma_{k+1}^{\mathrm{p}} \cup \Pi_{k+1}^{\mathrm{p}}.$$

Related to the question whether $\mathcal{P} = \mathcal{NP}$, it is not known if the hierarchy collapses at any stage $k \in \mathbb{N}_0$, i.e., whether $\Sigma_k^{\mathrm{p}} = \Sigma_{k+1}^{\mathrm{p}}$. If this is the case, then the hierarchy collapses also above the stage k, i.e., $\Sigma_k^{\mathrm{p}} = \Sigma_j^{\mathrm{p}}$ for all $j \geq k$ [Sto76, Theorem 3.2]. In turn, this also implies that, in order to prove $\mathcal{P} \neq \mathcal{NP}$, it would suffice to show that $\mathcal{P} \neq \Sigma_k^{\mathrm{p}}$ for some $k \in \mathbb{N}$ [Sto76, Corollary 3.3]. Generalizing the question whether $\mathcal{NP} = \text{co-}\mathcal{NP}$, there is also the open question if $\Sigma_k^{\mathrm{p}} = \Pi_k^{\mathrm{p}}$ for any $k \in \mathbb{N}$. Again, if this is the case at some stage k, the hierarchy collapses above that stage, i.e., $\Sigma_k^{\mathrm{p}} = \Sigma_j^{\mathrm{p}} = \Pi_j^{\mathrm{p}}$ for all $j \geq k$ [Sto76, Theorem 3.2]. On the other hand, with the current knowledge, it could be possible that the hierarchy is finite, i.e., that there are proper inclusions up to some stage k, and a collapse above that stage. However, the typical conjecture, along the lines of $\mathcal{P} \neq \mathcal{NP}$, is that the polynomial-time hierarchy is infinite.

An equivalent definition of the classes Σ_k^{p} and Π_k^{p} uses quantified expressions as follows [Wra76, Theorem 3]. For $k \in \mathbb{N}_0$, the class Σ_k^{p} consists of all decision problems that can be written as $\exists x_1 \forall x_2 \ldots Q x_k f(x_1, \ldots, x_k)$, where $Q = \exists$ if k is odd and $Q = \forall$ if k is even, and f can be evaluated in polynomial time. Analogously, the class Π_k^{p} consists of all decision problems that can be written as $\forall x_1 \exists x_2 \ldots Q x_k f(x_1, \ldots, x_k)$, where $Q = \forall$ if k is odd and $Q = \exists$ if k is even, and f can be evaluated in polynomial time. In both cases, the

quantifiers $\exists$ and $\forall$ alternate, starting with $\exists$ in case of Σ_k^{p} and with $\forall$ in case of Π_k^{p}. To be precise, one also has to require that all strings $x_1, \ldots, x_k$ have polynomially-bounded length; see [Wra76] for a more formal definition.

This alternative definition is closely related to some fundamental and natural decision problems that lie in Σ_k^{p} and Π_k^{p}, respectively. They are extensions of the $\mathcal{NP}$-complete Satisfiability Problem (see Section 2.1.1.2) that are equipped with additional alternating quantifications, analogously to the definition in the previous paragraph. We define the $\exists\forall$-Quantified Satisfiability Problem, which is contained in Σ_2^{p} and which we will work with in Section 2.4.2, as follows.

$\exists\forall$-Quantified Satisfiability Problem
Instance: A Boolean formula $f\colon \{0,1\}^p \times \{0,1\}^n \to \{0,1\}$
Task: Decide whether there exists some $x \in \{0,1\}^p$ such that $f(x,y) = 1$ for all $y \in \{0,1\}^n$.

Note that the Satisfiability Problem can be seen as a special case of the $\exists\forall$-Quantified Satisfiability Problem by setting $n = 0$. However, one has to be careful about the complexity of these problems if the Boolean formula is given in conjunctive or disjunctive normal form; see Remark 2.19 below.

In order to speak of decision problems being at least as hard as the problems in some complexity class (of the polynomial-time hierarchy), we can directly generalize the definition of $\mathcal{NP}$-completeness (see Definition 2.6):

Definition 2.18. Let $\mathcal{C}$ be some class of decision problems (such as the ones introduced in Definition 2.17). A decision problem P is *$\mathcal{C}$-complete* if it is contained in $\mathcal{C}$ and all other problems in $\mathcal{C}$ polynomially transform to P.

Definition 2.18 is reasonable for the complexity classes of the polynomial-time hierarchy because, as the classes $\mathcal{NP}$ and co-$\mathcal{NP}$, they are closed under the operation given by polynomial transformations. Again, $\mathcal{C}$-completeness of a problem is typically proved using a polynomial transformation from some $\mathcal{C}$-complete problem.

While the Satisfiability Problem was the first problem that was proved to be $\mathcal{NP}$-complete, the quantified versions of the Satisfiability Problem introduced above play a similar role in the polynomial-time hierarchy. In fact, they are Σ_k^{p}-complete or Π_k^{p}-complete for the appropriate $k \in \mathbb{N}$ [MS72; Wra76]. In particular, the $\exists\forall$-Quantified Satisfiability Problem is Σ_2^{p}-complete and its complement, in which the existential and the universal quantification are exchanged, is Π_2^{p}-complete.

Remark 2.19. Recall from Remark 2.7 that the Satisfiability Problem is $\mathcal{NP}$-complete if the Boolean formula is given in conjunctive normal form, while it is solvable in polynomial time if the Boolean formula is given in disjunctive normal form. Moreover, by De Morgan duality, the negation of a Boolean formula in conjunctive normal form is directly given in disjunctive normal form, and vice versa. Accordingly, the complement of the Satisfiability Problem, when it is stated in the form "Decide whether $f(y) = 1$ holds for all $y \in \{0,1\}^n$", is co-$\mathcal{NP}$-complete if the Boolean formula is given in disjunctive normal form and solvable in polynomial time if the Boolean formula is given in conjunctive normal form. This directly generalizes to the higher classes of the polynomial-time hierarchy. For the second level of the hierarchy, it means that the $\exists\forall$-Quantified Satisfiability Problem, as defined above, remains Σ_2^{p}-complete if the Boolean formula is given in

disjunctive normal form, while it is contained in $\mathcal{NP}$ if the Boolean formula is given in conjunctive normal form. This is in line with the fact that the inner problem, i.e., the universal quantification, is equivalent to the complement of the Satisfiability Problem. Analogously, the complement of the ∃∀-Quantified Satisfiability Problem remains Π_2^{p}-complete if the Boolean formula is given in conjunctive normal form. For more details, see, e.g., [Wra76].

As mentioned above, the Satisfiability Problem can be seen as the special case of the ∃∀-Quantified Satisfiability Problem where the inner problem is trivial (i.e., $n = 0$). This is true for general Boolean formulas, but not if, following the above observations, we restrict the Satisfiability Problem to Boolean formulas given in conjunctive normal form and the ∃∀-Quantified Satisfiability Problem to Boolean formulas given in disjunctive normal form. However, under these restrictions, it is still possible to polynomially transform the Satisfiability Problem to the ∃∀-Quantified Satisfiability Problem (using a more complicated construction) because the ∃∀-Quantified Satisfiability Problem is Σ_2^{p}-complete and the Satisfiability Problem lies in $\mathcal{NP}$ and therefore in Σ_2^{p}.

For the complexity results that we will prove using quantified variants of the Satisfiability Problem (see Section 2.4.2.1), we will consider general Boolean formulas in order to avoid the above discussion.

Analogously to Definition 2.10, we can also classify general computational problems, and in particular optimization problems, relative to the complexity classes of the polynomial-time hierarchy:

Definition 2.20. Let $\mathcal{C}$ be some class of decision problems (such as the ones introduced in Definition 2.17). A computational problem P is *$\mathcal{C}$-hard* if all problems in $\mathcal{C}$ polynomially reduce to P. A computational problem P is *$\mathcal{C}$-easy* if P polynomially reduces to a problem in $\mathcal{C}$. A computational problem P is *$\mathcal{C}$-equivalent* if it is both $\mathcal{C}$-hard and $\mathcal{C}$-easy.

In order to prove the $\mathcal{C}$-hardness of a problem, again, one usually describes a polynomial reduction from some $\mathcal{C}$-complete problem. In Sections 2.2.2.1 and 2.4.2, we will discuss the Σ_2^{p}-hardness of bilevel optimization problems and also make use of the Σ_2^{p}-completeness of the ∃∀-Quantified Satisfiability Problem, e.g., in the proof of Theorem 2.33. For more problems that belong to higher levels of the polynomial-time hierarchy, see, e.g., [SU02a; SU02b; Woe21; GW25a; GW24; GW25b].

We finally emphasize an important practical implication of Σ_2^{p}-hardness results [Woe21]: Assuming $\Sigma_2^{\mathrm{p}} \neq \mathcal{NP}$, a Σ_2^{p}-hard problem cannot be formulated as an integer linear program of polynomial size, and therefore, we cannot expect to solve such a problem using a standard solver for integer linear programs. Hence, Σ_2^{p}-hardness does not only mean that slow running times are to be expected when solving the problem, but also that new techniques or problem-specific algorithms might be required to solve the problem at all.

2.2 Bilevel Optimization

In classical optimization problems, there is one decision maker whose aim it is to select a feasible solution that is optimum with respect to some objective function (see Definition 2.13). *Bilevel optimization* considers an important extension of this concept in which there are two decision makers, called the *leader* and the *follower*. Each of them has their own optimization problem (i.e., decision variables, constraints, and objective function),

and they select their solutions one after the other, in a hierarchical order: First the leader, then the follower. The key feature is that both decisions also have an impact on the other decision maker's optimization problem. This can concern the set of feasible solutions or the objective function. This interdependence makes bilevel optimization problems very difficult in general.

This section introduces bilevel optimization problems, which constitute the central concept of the work in this thesis. We will first deal with basic definitions and observations regarding bilevel optimization problems in Section 2.2.1, and then discuss literature and central results in the field in Section 2.2.2. The focus here lies on complexity results and bilevel combinatorial optimization problems in the literature.

2.2.1 Basic Definitions

As mentioned above, a bilevel optimization problem involves two decision makers, a leader and a follower, who make their decisions in a hierarchical order. Hence, from the follower's perspective, the leader's decision is already fixed and parameterizes the follower's optimization problem, i.e., for a fixed leader's decision, the follower's problem is a "usual" optimization problem. However, the more interesting and challenging perspective is the one of the leader whose goal it is to solve her optimization problem not only with respect to the "usual" constraints, but also while anticipating the follower's reaction to her solution. Mathematically speaking, the optimality of the follower's solution is a constraint in the leader's optimization problem.

Notation. We will always refer to the leader using "she/her" and to the follower using "he/him/his".

Remark 2.21. In the (mathematical) literature on bilevel optimization, the leader's and follower's optimization problem are often called *upper-level* and *lower-level* problem. The interpretation as two decision makers that are called *leader* and *follower* is more common in the game-theoretic context, where the concept corresponding to bilevel optimization is known under the name of a *Stackelberg game*; see also Section 2.2.2. However, also for our purposes, it will be very useful and illustrative to imagine two persons who solve their respective optimization problems and interact according to the bilevel problem structure. Therefore, we will use the terms *leader's* and *follower's problem*, *solution* etc. throughout.

We will now give a more formal definition of a bilevel optimization problem. It loosely follows the common introductory literature, such as [CMS07; DKPK15; BS21], but deviates in some formal details and notation, being adapted to our combinatorial and complexity-theoretic perspective. In particular, our definition is compatible with our definition of an optimization problem (see Definition 2.13) in the sense that bilevel optimization problems are a special type of optimization problems. Hence, the complexity-theoretic notions introduced in Section 2.1 for optimization problems are also applicable to bilevel optimization problems.

For the sake of a simpler exposition, we will introduce the leader's and the follower's problem as maximization problems, but of course they can also be minimization problems.

To formalize the feasible leader's and follower's solutions, respectively, let $\mathcal{L}$ and $\mathcal{F}_x$ for all $x \in \mathcal{L}$ be sets, and define $\mathcal{F} := \bigcup_{x \in \mathcal{L}} \mathcal{F}_x$. The leader's and follower's objective functions are given as $\mathcal{C}, \mathcal{D} \colon \mathcal{L} \times \mathcal{F} \to \mathbb{R}$. An appropriate description of these sets and objective

functions constitutes an instance of a bilevel optimization problem. Usually, the follower's set $\mathcal{F}_x$ depends on the leader's solution $x \in \mathcal{L}$ in a way that is easy to describe, e.g., in terms of constraints depending linearly on x, such that we do not need to require that every set $\mathcal{F}_x$ is stored separately (and, accordingly, we do not need to require $\mathcal{L}$ to be finite).

The *follower's problem* is defined, for every fixed leader's solution $x \in \mathcal{L}$, as:

$$\begin{aligned} &\max_y \mathcal{D}(x, y) \\ &\text{s. t. } y \in \mathcal{F}_x \end{aligned} \tag{F}$$

For every fixed $x \in \mathcal{L}$, we denote the set of optimum solutions of the follower's problem by

$$\mathcal{F}_x^* := \operatorname{argmax}\{\mathcal{D}(x, y) \mid y \in \mathcal{F}_x\}.$$

The follower's problem is a well-defined optimization problem according to Definition 2.13. As discussed in Section 2.1.1.3, it could be infeasible, unbounded, or not attaining its optimum value; then $\mathcal{F}_x^* = \emptyset$. The usual case, however, is that the follower's problem has one or several optimum solutions.

The leader's task is now to optimize $\mathcal{C}(x, y)$ under the constraints $x \in \mathcal{L}$ and $y \in \mathcal{F}_x^*$. An aspect which is still imprecise is the question how the follower's optimum solution $y \in \mathcal{F}_x^*$ in the leader's problem is selected if the follower's problem has several optimum solutions, i.e., if $|\mathcal{F}_x^*| > 1$. While different solutions $y \in \mathcal{F}_x^*$ lead to the same follower's objective value $\mathcal{D}(x, y)$ by definition, different choices could have a large impact on the leader's objective $\mathcal{C}(x, y)$. The two most common ways to handle this issue are to adopt the so-called *optimistic* or *pessimistic* setting, respectively. If the leader is optimistic, she assumes that the follower will select y in favor of her objective function in case there is no unique optimum follower's solution. On the other hand, in the pessimistic setting, the leader assumes that the follower will select $y \in \mathcal{F}_x^*$ in the worst possible way with respect to her objective function.

Formally, the resulting two variants of the *bilevel optimization problem* (or *leader's problem*) can now be written as follows.

$$\max\{\mathcal{C}(x, y) \mid x \in \mathcal{L}, y \in \mathcal{F}_x^*\} \tag{L-opt}$$

$$\max\left\{\mathcal{C}(x, y) \,\middle|\, x \in \mathcal{L}, y \in \operatorname{argmin}\{\mathcal{C}(x, y') \mid y' \in \mathcal{F}_x^*\}\right\} \tag{L-pess}$$

Remark 2.22. In the strict interpretation according to Definition 2.13, a *feasible follower's solution* is some $y \in \mathcal{F}_x$ and an *optimum follower's solution* is some $y \in \mathcal{F}_x^*$, while a *feasible* or *optimum leader's solution* is a pair $(x, y) \in \mathcal{L} \times \mathcal{F}$. However, depending on the problem and on the discussed questions, a less strict usage of these notions can be useful.

First, the follower's problem might or might not be seen to contain the optimization of the leader's objective function as a secondary criterion according to the optimistic or the pessimistic setting. That is, the follower's problem, for fixed $x \in \mathcal{L}$, could also be defined as $\max\{\mathcal{C}(x, y) \mid y \in \mathcal{F}_x^*\}$ or $\min\{\mathcal{C}(x, y) \mid y \in \mathcal{F}_x^*\}$, in the optimistic or in the pessimistic setting, respectively, instead of as (F). In other words, the follower's objective and (the opposite of) the leader's objective are optimized in a lexicographic manner. We will see that, for the bilevel optimization problems discussed in the following chapters, there is no relevant difference between the two possible definitions regarding the complexity of or algorithms for the follower's problems. Moreover, the optimistic and the pessimistic

setting can often be handled very similarly for our problems. Therefore, we do not have to distinguish the two problem variants for large parts of our considerations and only discuss their differences where necessary.

Furthermore, we will typically use the term *(feasible or optimum) leader's solution* only for the leader's part of the decision, i.e., for some $x \in \mathcal{L}$ instead of a pair $(x, y) \in \mathcal{L} \times \mathcal{F}$. We call such a solution $x \in \mathcal{L}$ *feasible* if $\mathcal{F}_x^* \neq \emptyset$, i.e., if the corresponding follower's problem has an optimum solution. We emphasize that this also makes it part of the leader's task to ensure that the resulting follower's problem is feasible. Given some feasible $x \in \mathcal{L}$, we also say that a (or the) corresponding optimum follower's solution $y \in \mathcal{F}_x^*$ (possibly also optimum including optimism or pessimism) is a (or the) *follower's response* to x. Accordingly, an *optimum* leader's solution is a feasible leader's solution $x \in \mathcal{L}$ such that, together with a follower's response to it, it is optimum for the leader's problem.

In some of the bilevel optimization problems in this thesis (the Bilevel Selection Problem in Chapter 3 and the Bilevel Minimum Spanning Tree Problem in Chapter 5), leader's and follower's solutions are sets X and Y (of items or edges), and the constraint that the two players have to satisfy concerns the union $X \cup Y$. In these cases, we will sometimes also talk of $X \cup Y$ as a solution of the overall problem.

We will often write bilevel optimization problems in the following form, which is convenient and easier to read with respect to the bilevel structure, compared with (L-opt) and (L-pess):

$$\begin{aligned} &\max_{x} \ \mathcal{C}(x, y) \\ &\text{s.t. } x \in \mathcal{L} \\ &\qquad y \in \operatorname*{argmax}_{y'} \mathcal{D}(x, y') \\ &\qquad\qquad \text{s.t. } y' \in \mathcal{F}_x \end{aligned} \tag{L}$$

This formulation is again imprecise with respect to the handling of non-unique optimum follower's solutions, but can easily be made precise: For the optimistic setting, replace $\max_x$ by $\max_{x,y}$, and for the pessimistic setting write:

$$\begin{aligned} &\max_{x \in \mathcal{L}} \ \min_{y} \ \mathcal{C}(x, y) \\ &\qquad \text{s.t. } y \in \operatorname*{argmax}_{y'} \mathcal{D}(x, y') \\ &\qquad\qquad \text{s.t. } y' \in \mathcal{F}_x \end{aligned}$$

However, for simplicity of notation, we will often use the ambiguous formulation (L) and keep in mind that there are an optimistic and a pessimistic problem version that might need to be handled differently, depending on the specific problem.

Even in settings which are relatively simple in single-level optimization, one has to be careful in bilevel optimization. For instance, there are bilevel optimization problems that turn out to have unexpected properties, such as not having a solution that attains their optimum value; we will see such a case in Lemma 3.12. We refer to, e.g., [VSJ96] for more examples of such a behavior in bilevel optimization problems with continuous leader's variables and integer follower's variables.

Moreover, the set of feasible (leader's) solutions (x, y) is nonconvex in general, even for the bilevel equivalents of linear programs, i.e., bilevel optimization problems with

continuous variables and leader's and follower's constraints that depend linearly on these variables. Later on, we will also call these problems *bilevel linear programs*. In fact, bilevel linear programs are already $\mathcal{NP}$-hard. On the other hand, the leader's problem is usually $\mathcal{NP}$-easy if the follower's problem is tractable. However, if the follower's problem is an $\mathcal{NP}$-hard optimization problem itself, then the leader's problem typically falls into the second level of the polynomial-time hierarchy (see Section 2.1.4). We refer to Section 2.2.2.1 for more details on these complexity results.

Remark 2.23. In contrast to the introducing paragraph of Section 2.2 in which we mentioned that both decisions might impact the feasibility or the objective function of the other decision maker's problem, we made a simplifying assumption in our formal definition of a bilevel optimization problem: The follower's optimum solution $y \in \mathcal{F}_x^*$ occurs in the leader's objective function $\mathcal{C}(x, y)$, but not in the leader's constraints $x \in \mathcal{L}$. In the problem formulation, it would be easy to allow also for so-called *coupling constraints*, i.e., constraints in the leader's problem that involve both the leader's and the follower's decision variables: We could redefine $\mathcal{L}$ such that the leader's constraints are of the form $(x, y) \in \mathcal{L}$.

However, bilevel optimization problems with coupling constraints are, in some sense, more complicated than those without coupling constraints. For instance, the feasible set of a bilevel linear program with coupling constraints can be disconnected, while it is always connected in the setting without coupling constraints; see [Ben89; Kle21; HLST25a]. In addition, the follower's choice $y \in \mathcal{F}_x^*$ among his optimum solutions (in the pessimistic setting) might not only affect the leader's objective function value now, but even the feasibility of her solution $x \in \mathcal{L}$; see, e.g., [BS21, Definition 3.5]. In [MD06], it is shown that it is not allowed in general to shift constraints from the leader's to the follower's problem.

On the other hand, there is no difference in complexity between bilevel optimization with and without coupling constraints; see [HLST25a]. Moreover, in [HLST25a; HLST25b], we present polynomial-sized reformulations of general optimistic and pessimistic bilevel linear programs with coupling constraints as respective problems without coupling constraints, which are equivalent on the level of optimum solutions.

Since the bilevel optimization problems that we will be concerned with in this thesis do not contain any coupling constraints, we omit them also in the general formulations here.

Remark 2.24. As a natural extension of bilevel optimization problems, one can consider optimization problems with more than two levels, also called *multilevel optimization problems*. The levels still correspond to different interacting decision makers that are arranged in a hierarchy, i.e., the decision makers can be assumed to decide one after the other. The task of the highest-level decision maker now involves anticipating the reactions to her decision in the entire sequence of lower-level problems, while optimizing her own objective. Obviously, more levels in the hierarchy of decisions might come with an increase in complexity. Indeed, the computational complexity of general classes of multilevel optimization problems depends on the number of levels and is closely related to the polynomial-time hierarchy (see Section 2.1.4); see also Section 2.2.2.1.

As another extension of bilevel optimization, one can consider multi-leader-follower games, where several decision makers interact in some structure involving a bilevel hierarchy and Nash equilibria; see [AS20].

2.2.2 Literature Overview

As the first occurrence of bilevel optimization in the literature, one usually mentions the seminal work of the economist Heinrich von Stackelberg [Sta34]. Therefore, in economics and game theory, the concept is often called *Stackelberg game* and the corresponding solution a *Stackelberg equilibrium*. In the mathematical optimization community, the concept first appeared in [BM73]. Since then, bilevel optimization problems have been studied from several different perspectives, and numerous articles, surveys and books have appeared on the topic. For a brief overview of the milestones in the history of bilevel optimization, we refer to [BS21, Section 1.3]. A recent overview of more than 1000 references on the topic of bilevel optimization can be found in [Dem20]. For general introductions to the topic, we also refer to [CMS07; DKPK15; BS21].

Our focus in this thesis lies on the perspective of combinatorial optimization and computational complexity. Compared with the majority of the previous work on bilevel optimization, the literature on these aspects of bilevel optimization is still relatively limited. We will discuss literature that is concerned with the computational complexity of bilevel optimization problems in Section 2.2.2.1 and research on bilevel versions of combinatorial optimization problems in Section 2.2.2.2. Within the field of bilevel optimization, the literature on these two aspects can be seen as the closest related work to the research presented in this thesis.

Other important research directions concerning bilevel optimization comprise, for example, studying convexity properties or optimality conditions of bilevel optimization problems, general solution approaches, often using single-level reformulations of bilevel optimization problems and branch-and-bound algorithms, and applications in various fields, e.g., gas and electricity networks [DKPK15; Dem20]. These research aspects mainly come from the nonlinear optimization community or are mostly concerned with improving the practical solvability of bilevel optimization problems. For an overview of solution methods for bilevel optimization problems, see, e.g., [SMD18; KLLS21].

2.2.2.1 Computational Complexity

The first complexity results for bilevel optimization problems were obtained by Jeroslow in [Jer85]. He considered multilevel optimization problems and classified them into the polynomial-time hierarchy. The main results are that general k-level optimization problems with linear objectives and constraints are $\Sigma^{\mathrm{p}}_{k-1}$-hard in case of continuous variables and Σ^{p}_{k}-hard in case of binary variables, for $k \in \mathbb{N}$. This is in line with single-level linear programs being polynomial-time solvable and integer linear programs being $\mathcal{NP}$-hard. For bilevel optimization problems, we have $\mathcal{NP}$-hardness in the continuous case and Σ^{p}_{2}-hardness in the binary case.

Shorter $\mathcal{NP}$-hardness proofs for bilevel (continuous) linear programs were obtained in [Bar91; BB90; Bla92]. The same problem was shown to be even strongly $\mathcal{NP}$-hard in [Den98; HJS92]. Also checking whether a given point is a local minimum is an $\mathcal{NP}$-hard problem [VSJ94]. We also refer to [DKW98, Section 2.4] for a detailed discussion of some of these hardness results. See also [MS05] for an overview of hardness results and proofs for bilevel linear programs. In contrast, bilevel linear programs become polynomial-time solvable when the number of follower's variables is constant [LS95; Den98].

While several different reductions were used to prove the hardness results mentioned in the previous paragraph, the following intuitive argument illustrates the core idea; see, e.g., [VSJ96; AHJS97; MS05]. It is well-known that binary linear programs are $\mathcal{NP}$-hard in general, where the hardness is due to the binarity constraints, since linear programs with continuous variables are solvable in polynomial time. On the other hand, when we consider bilevel linear programs, the additional hardness compared with single-level linear programs comes from the hierarchical structure. In a hardness proof, the binarity constraints of a binary linear program can be modeled into the second level of a bilevel linear program as follows. A binary variable $x \in \{0, 1\}$ is modeled using a continuous leader's variable $x \in [0, 1]$ and an additional variable $y \in [0, 1]$ that is controlled by the follower. The follower maximizes y, while having to respect the constraints $y \leq x$ and $y \leq 1 - x$. Hence, the follower's optimum value for y models the deviation of x from the closest binary value, i.e., $y = \min\{x, 1 - x\}$. When we allow for coupling constraints (see Remark 2.23), the follower's optimum value of y can be directly forced to be 0 by the leader's constraint $y = 0$. Due to the follower's problem's structure, this is equivalent to forcing x to be binary. When coupling constraints are not allowed, the same idea can be applied in a more indirect way: Add the term My to the leader's objective function (assuming that the leader's problem is a minimization problem), where M is a constant that is large enough, depending on the problem's parameters. Therefore, the leader's primary goal is to minimize y, which basically forces y to be 0 and thus again x to be binary. For more details, in particular regarding the choice of M, we refer to [VSJ96]; see also [HLST25a]. We will use similar arguments also in Section 2.4.2; see Remarks 2.31 and 2.35 and the proof of Theorem 2.33.

Due to this idea of shifting binarity that occurs in the leader's problem into the follower's problem, there is usually no difference in the complexity of bilevel optimization problems with discrete leader's variables compared to those with continuous leader's variables. On the other hand, when the follower has discrete variables, the bilevel problem can be expected to be one level harder than in the case when the follower has continuous variables; see the results of [Jer85] mentioned above. The latter observation is also confirmed by the fact that bilevel linear programs are $\mathcal{NP}$-easy and thus in fact easier than Σ_2^{p}-hard bilevel optimization problems with discrete follower's variables, unless $\Sigma_2^{\mathrm{p}} = \mathcal{NP}$ [Buc23]. In Lemma 2.32, we will see a simple special case of this result, concerning the case of binary leader's variables.

Turning to the complexity of discrete bilevel optimization problems again, we also refer to [Joh11, Theorem 3.1.1] for a shorter proof of the Σ_2^{p}-completeness of bilevel binary linear programming, and to [Woe21] for illustrations and intuitions regarding the Σ_2^{p}-hardness of bilevel optimization problems and also other classes of problems. As mentioned in the end of Section 2.1.4, Σ_2^{p}-hard (bilevel) problems can be expected to be hard to deal with also in practice because they cannot be formulated as polynomial-size integer linear programs, unless $\Sigma_2^{\mathrm{p}} = \mathcal{NP}$.

While all complexity results mentioned so far are for bilevel optimization problems with linear objective functions, one can also look at bottleneck objectives. This typically means that the maximum instead of the sum of the costs of the items that are selected is minimized; see also Sections 2.2.2.2 and 5.8. In [DKW98], Dudás, Klinz, and Woeginger extended the results of [Jer85] to (continuous) multilevel bottleneck problems with linear constraints. They showed that these problems are on the same levels of the polynomial-time hierarchy as bilevel linear programs, i.e., that a k-level problem of this type is again $\Sigma_{k-1}^{\mathrm{p}}$-hard, for $k \in \mathbb{N}$.

In this thesis, we will mostly deal with the computational complexity of specific bilevel optimization problems. The Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem, which are the basic versions of the problems we study in Chapters 3 and 4, are solvable in polynomial time. In light of the complexity results mentioned above, this is unusual for bilevel optimization problems. However, this is the reason why we consider them to be useful starting points for studying the additional complexity induced by uncertainty in bilevel optimization problems; see Section 2.4. On the other hand, in Chapter 5, we will prove that the Bilevel Minimum Spanning Tree Problem is $\mathcal{NP}$-hard, even without uncertainty. In Section 4.2, we will review several bilevel versions of the $\mathcal{NP}$-hard Knapsack Problem that have been previously studied. They are typical examples of Σ_2^{p}-hard bilevel optimization problems. A few more complexity results for other bilevel combinatorial optimization problems will be mentioned in the next section.

2.2.2.2 Bilevel Combinatorial Optimization Problems

Although bilevel optimization is a concept that has attracted much more attention in other areas of optimization, there is also a variety of research on bilevel combinatorial optimization problems. This section provides an insight into the literature in this area, while not being a complete list and focusing on the works that are close to the research in this thesis.

The basic combinatorial versions of the three bilevel optimization problems that we will study in this thesis – the Bilevel Selection Problem in Chapter 3, the Bilevel Knapsack Problem in Chapter 4, and the Bilevel Minimum Spanning Tree Problem in Chapter 5 – all have a similar structure: There is a set of items (or edges in a graph in case of the Bilevel Minimum Spanning Tree Problem), each of which is owned by the leader or by the follower (or both). The two decision makers' common task is to select items in order to build some combinatorial structure (such as a spanning tree in the given graph). Each of them can only select items from their own set and has a different cost function on the items that they aim to optimize. In [HW25], we propose to call problems of this structure *partitioned-items bilevel problems*. For a precise definition of the problems studied in this thesis, see Sections 3.2, 4.2, and 5.2.1. The Bilevel Knapsack Problem and the Bilevel Minimum Spanning Tree Problem (and variants of them) have been studied before; we refer to Sections 4.2.3 and 5.2.3 for an overview of previous literature on these problems.

In [HW25], we have also studied two partitioned-items bilevel variants of the shortest path problem regarding their computational complexity. In this *bilevel shortest path problem*, like in the Bilevel Minimum Spanning Tree Problem, each of the two decision makers controls a subset of the set of edges in a graph and selects some of them, according to their personal edge weights. The solution that the two players build together is a path between two specified vertices. In the strong path completion variant, all edges selected by the leader have to be part of the path, while in the weak path completion variant, the follower may ignore some of them. We investigated both directed and undirected graphs, as well as directed acyclic graphs, and completely classified the computational complexity of all these problem variants. In the weak path completion variant, the follower's problem is solvable in polynomial time and the leader's problem is $\mathcal{NP}$-hard. In case of strong path completion, the follower's problem is already $\mathcal{NP}$-hard and the leader's problem was shown to be Σ_2^{p}-hard. Only in a directed acyclic graph and with strong path completion,

the leader's and the follower's problem can be solved in polynomial time. Finally, we studied the case where the leader's edge set is of constant size and, in particular, showed that the problem is then equivalent to the shortest-k-cycle problem in case of strong path completion and undirected graphs.

Another partitioned-items bilevel combinatorial optimization problem is the *bilevel assignment problem*. Again, each decision maker owns a subset of the set of edges in a graph. The common goal here is to construct a perfect matching. The computational complexity of the bilevel assignment problem has been investigated in [GK09; FMW22]. Not only the usual sum objective functions, where the sum of the costs of all edges has to be paid, are considered, but also bottleneck objective functions, where only the cost of the most expensive edge is to be minimized. Each of the two decision makers may have a sum or a bottleneck objective and, in addition, the optimistic and the pessimistic case can be distinguished for every combination. Gassner and Klinz proved for seven of the resulting eight settings that the bilevel assignment problem is $\mathcal{NP}$-hard; the optimistic problem version in which both decision makers have bottleneck objectives remained open [GK09]. This gap was closed by Fischer, Muluk, and Woeginger who showed that also the remaining case is $\mathcal{NP}$-hard. Moreover, their proof applies to all eight cases, while in [GK09], several different reductions were used. In Chapter 5, we will classify the computational complexity of the corresponding eight variants of the Bilevel Minimum Spanning Tree Problem. In contrast to the bilevel assignment problem, we will see that some cases are $\mathcal{NP}$-hard, while others are polynomial-time solvable; see Table 5.1 for an overview. A solution approach for the bilevel assignment problem, based on a branch-and-bound algorithm, as well as polynomial-time algorithms for special cases of the problem, were developed by Beheshti, Prokopyev, and Pasiliao in [BPP16].

Another widely studied class of bilevel combinatorial optimization problems are *price-setting problems*. A typical example of a price-setting problem is the *toll optimization problem*: The follower models one or several drivers each solving a shortest path problem in a graph, while the leader can charge tolls as additional costs on some of the edges for the follower. The tradeoff here is that the leader aims at maximizing her own profit from the tolls she sets, but should not discourage the follower from using her edges by making them too expensive to use. The toll optimization problem has been shown to be $\mathcal{NP}$-hard by Labbé, Marcotte, and Savard in [LMS98]. They also identify special cases that can be solved in polynomial time and provide a mixed integer programming formulation of the general problem. More generally, price-setting problems can be described as bilevel optimization problems in which the leader determines or influences some of the costs that appear in the follower's objective function, and she receives profits from the costs paid by the follower in his response. This concept is mostly applied to various network problems, with applications in, e.g., transportation and telecommunications; see [MS05; Hoe08; LV13] for surveys and further results on network price-setting problems. Moreover, a price-setting problem in which the follower's problem is a Minimum Spanning Tree Problem – which is therefore another bilevel variant of the Minimum Spanning Tree Problem, besides the Bilevel Minimum Spanning Tree Problem we deal with in Chapter 5 – has been investigated in several articles; see [LPP21] and the references therein.

Another class of bilevel optimization problems in which the leader can influence the follower's objective function is the one of *(partial) inverse problems*. Here, the leader's goal is to make some given (partial) solution of the follower's problem optimum for the follower, while minimizing the adjustments made to his objective function. Many problem

variants can be distinguished, e.g., regarding the allowed ranges for the adjustments the leader can make and the norm in which they are measured in her objective function. For an overview on the topic of inverse combinatorial problems, we refer to the surveys [Heu04; DM14]. Usually, the complete inverse problem corresponding to a polynomial-time solvable underlying optimization problem is again solvable in polynomial time [AO01], while partial inverse problems can be harder than that. The complexity of many complete and partial inverse variants of the Minimum Spanning Tree Problem, again inluding sum and bottleneck objectives, has been analyzed, e.g., in [ZXM97; SAO99; AO00; CDYZ08; Gas09; LZD18]. Some variants are shown to be solvable in polynomial time, while $\mathcal{NP}$-hardness proofs are provided for others. Polynomial-time algorithms for different variants of a partial inverse matroid problem are presented, e.g., in [DMM03; ZLLD16; LZL16].

Note that, in the setting of robust bilevel optimization that we will introduce in Section 2.4, the adversary's problem has some similarity with a price-setting problem or an inverse problem, as he also determines parameters of the follower's objective function. However, the adversary's own objective depends only indirectly on the parameters he sets, since it is given by the opposite of the leader's objective function value of the resulting follower's solution.

Finally, we mention the class of *interdiction problems*. Typically, but not necessarily, the follower's problem is again some network problem, such as a shortest path or a network flow problem. The leader can remove some vertices or edges from the network in order to harm the follower as much as possible, i.e., the leader's and the follower's objective functions are opposite. Hence, interdiction problems are a very special type of bilevel optimization problems and also have some similarity with robust optimization problems (see Section 2.3), which can be seen as problems of two decision makers with opposite objectives as well. Interdiction problems have been studied extensively and have often been shown to be $\mathcal{NP}$-hard; see, e.g., [Isr99; DeN11; SPG13] and the references therein. In Section 4.2.3, we will discuss different bilevel versions of the Knapsack Problem, one of which is also an interdiction problem. Interdiction variants of the Minimum Spanning Tree Problem have been investigated, e.g., in [LC93; FS99; WF25b]. In [GW25a; GW25b], it is shown that interdiction variants of many classical $\mathcal{NP}$-hard combinatorial optimization problems are Σ_2^{p}-hard.

A few complexity and algorithmic results on bilevel scheduling problems can be found, e.g., in [KW96; KK12].

2.3 Robust Optimization

In the presentation so far, one assumes that the instance of an optimization problem, consisting of parameters of the constraints and the objective function, is precisely known when solving the problem. However, when the problem comes from an application, the decision maker might not have all the information, for example, because problem parameters are just estimates from historical data and their real values will only be revealed in the future.

The two main concepts to model such uncertainties are *robust optimization* and *stochastic optimization*. In stochastic optimization, one assumes that a probability distribution of the possible values that the uncertain parameters can attain is known. Typically, the goal is then to optimize the expected value of the objective function. Robust optimization deals

with the uncertainty in a more conservative way by aiming at optimizing the worst-case outcome. For this, one does not need to know a probability distribution of the uncertain parameters, but instead works with *uncertainty sets* containing all scenarios that are considered to be possible or reasonably likely. Depending on the application and the given data, the robust or the stochastic approach can be more appropriate.

In this thesis, we will only consider robust optimization problems. However, the idea of combining bilevel optimization and robust optimization, which we will describe in Section 2.4, can also be used to obtain stochastic bilevel optimization models in a similar fashion. In [BHI22], we studied such a stochastic version of a bilevel optimization problem. For an introduction to stochastic optimization, we refer to, e.g., [BL11].

The idea of robust optimization was first introduced by Soyster in [Soy73]. Over the time, different types of uncertainty sets and different robust optimization paradigms were introduced and studied extensively. Among the large amount of ideas and results on robust optimization, this section will focus only on the aspects that are directly relevant for the results in this thesis, which again correspond mainly to the combinatorial and complexity-theoretic perspective. Our presentation is mostly based on the survey [BK18]. For a more thorough introduction to the field of robust optimization and an overview of important literature, we also refer to [BK18] and the references therein, to the book [GH24], which focuses on combinatorial problems as well, and to the books [KY97; BEN09].

After an introduction to the general concept of robust optimization in Section 2.3.1, we will discuss different types of uncertainty sets in Section 2.3.2. Section 2.3.3 will then give an overview of further basic facts on robust optimization that are relevant in the context of this thesis. Finally, we review some selected literature on specific robust combinatorial optimization problems in Section 2.3.4.

2.3.1 Basic Definitions

As already mentioned above, in robust optimization, the worst-case outcome over all possible scenarios is optimized. Depending on whether the uncertain parameters are in the objective or in the constraints of the original optimization problem, this can mean two different things: In the former case, the feasibility of solutions is not affected by the uncertainty, and the objective is of the form "min max" or "max min". In the latter case however, a solution might be feasible in some scenarios and infeasible in others. The idea of hedging against the worst case then consequently means that a solution is only considered to be feasible for the robust problem if it is feasible in all scenarios. Our focus will lie on uncertainty in objective functions, such that we do not have to deal with feasibility issues introduced by the uncertainty.

Without loss of generality, we write all (underlying certain) optimization problems in this section as maximization problems again.

Consider an optimization problem of the form $\max\{\mathcal{C}(x) \mid x \in \mathcal{X}\}$ with a feasible set $\mathcal{X}$ and an objective function $\mathcal{C}\colon \mathcal{X} \to \mathbb{R}$. We now assume that (some parameter of) the objective function is uncertain, and a nonempty uncertainty set $\mathcal{U}$ consisting of all possible objective functions $\mathcal{C}\colon \mathcal{X} \to \mathbb{R}$ is given. The so-called *robust counterpart* of the problem is then given by

$$\max_{x \in \mathcal{X}} \min_{\mathcal{C} \in \mathcal{U}} \mathcal{C}(x).$$

We also call a problem of this form a *robust optimization problem* and the original optimization problem the *underlying certain problem* of its robust counterpart.

Under reasonable assumptions for the uncertainty set and its representation in the input, the robust problem can again be interpreted as an optimization problem according to Definition 2.13, where the objective function now involves selecting the worst case from the uncertainty set. Note that, like when considering continuous variables in optimization problems, we allow for general real numbers occurring in the uncertainty sets, although they cannot be represented exactly in a computer; see also Sections 2.1.1.1 and 2.1.1.3. However, when discussing specific uncertainty sets, we will typically assume that the parameters that define them are given by rational numbers, which usually suffices to obtain well-defined optimization problems.

Similarly to bilevel optimization (see Section 2.2), also a robust optimization problem can be understood as the interplay of two decision makers: First, the main decision maker selects a solution $x \in \mathcal{X}$, and second, her *adversary* selects a scenario $\mathcal{C} \in \mathcal{U}$ that results in the worst-case objective value $\mathcal{C}(x)$. Similarly to the follower's problem in bilevel optimization, the *adversary's problem* $\min_{\mathcal{C}\in\mathcal{U}} \mathcal{C}(x)$, for every fixed $x \in \mathcal{X}$, is again an optimization problem. In fact, bilevel optimization and robust optimization are closely related; see Remark 2.29.

All optimization problems in this thesis have a finite number of variables, indexed by some finite set $\mathcal{E}$, and a linear weight function $c\colon \mathcal{E} \to \mathbb{R}$. Accordingly, we can usually write them as

$$\max \{\textstyle\sum_{e\in\mathcal{E}} c(e)x_e \mid x \in \mathcal{X}\} \tag{R-UC}$$

(or $\max\{\min_{e\in\mathcal{E}} c(e)x_e \mid x \in \mathcal{X}\}$ in case of a bottleneck objective; see, e.g, Section 5.8) for some feasible set $\mathcal{X} \subseteq \mathbb{R}^{\mathcal{E}}$. In this setting and therefore also in the remainder of this section, we will write the uncertainty sets as $\mathcal{U} \subseteq \mathbb{R}^{\mathcal{E}}$, containing all possible weight functions $c\colon \mathcal{E} \to \mathbb{R}$. The robust counterpart of (R-UC) is then

$$\max \{\min \{\textstyle\sum_{e\in\mathcal{E}} c(e)x_e \mid c \in \mathcal{U}\} \mid x \in \mathcal{X}\}. \tag{R}$$

The properties and the complexity of a robust optimization problem highly depend on the structure of the uncertainty set $\mathcal{U}$. In the following section, we will discuss several common types of uncertainty sets.

2.3.2 Types of Uncertainty Sets

We will now introduce common types of uncertainty sets, which we will study in the context of robust bilevel optimization problems in Sections 2.4.2, 3.4, and 4.4, and review complexity results for the corresponding robust (single-level) optimization problems. When we later speak of *types of uncertainty sets*, we usually mean the classes of possible uncertainty structures distinguished in the following.

In general, it is often reasonable to assume that the uncertainty set is bounded (which usually is not a restriction; see [BK18, Section 2.5]) and closed, i.e., compact. In particular, this guarantees that the adversary's problem in (R) always has an optimum solution.

2.3.2.1 Discrete Uncertainty

A *discrete uncertainty set* consists of a finite number of scenarios that are explicitly given as part of the input of the robust optimization problem, i.e., a finite set $\mathcal{U} \subseteq \mathbb{Q}^{\mathcal{E}}$. In practice, such an uncertainty set can be built, for instance, of scenarios that have been observed in the past.

Typically, the robust counterpart subject to discrete uncertainty is significantly harder than the underlying certain optimization problem. In particular, even the robust counterpart of the trivial optimization problem $\max\{c^\top x \mid x \in \{0,1\}^n\}$ with an uncertainty set $\mathcal{U} \subseteq \mathbb{Q}^n$ consisting of only two scenarios is $\mathcal{NP}$-hard [BK18, Theorem 1]. The same holds true in case of the underlying certain problem being one of several polynomial-time solvable combinatorial optimization problems, such as the Minimum Spanning Tree Problem or the shortest path problem. However, for many of these problems, there is a pseudopolynomial-time algorithm whenever the number of scenarios is a fixed constant; see [ABV05]. For an unbounded number of scenarios however, the robust counterpart of all mentioned problems is strongly $\mathcal{NP}$-hard; see [BK18, Section 3.1.1] for more details.

In Section 2.4.2.2, we will show that such complexity results can also be transferred to robust bilevel problems. However, most of the more specific robust bilevel problems that we will study in Sections 3.4 and 4.4 turn out to be easier than that. We will also see a situation where a constant number of scenarios does not only enable a pseudopolynomial-time algorithm, but even a polynomial-time algorithm, while the problem is strongly $\mathcal{NP}$-hard for an unbounded number of scenarios; see Section 6.2 for more details.

2.3.2.2 Interval Uncertainty

In an *interval uncertainty set*, every parameter may independently vary within an interval, i.e., the uncertainty set is of the form $\mathcal{U} = \prod_{e\in\mathcal{E}}[c^-(e), c^+(e)]$ for some functions $c^-, c^+ \in \mathbb{Q}^{\mathcal{E}}$ that satisfy $c^-(e) \leq c^+(e)$ for all $e \in \mathcal{E}$.

Usually, interval uncertainty is not very interesting to study because it can easily be eliminated as follows. When all feasible solutions $x \in \mathcal{X}$ satisfy $x \geq 0$ (as it is typically the case, e.g., in combinatorial optimization), then the worst-case scenario $c \in \mathcal{U}$ is always given by $c = c^-$, independently of the choice $x \in \mathcal{X}$. Therefore, the robust optimization problem (R) is equivalent to the certain optimization problem

$$\max\{\textstyle\sum_{e\in\mathcal{E}} c^-(e)x_e \mid x \in \mathcal{X}\}$$

in this case. In particular, interval uncertainty does not introduce any additional complexity compared with the underlying certain problem in this setting.

However, in the robust bilevel setting that we will consider later on (see Section 2.4.1), interval uncertainty cannot be tackled in such a trivial way. In fact, we will see in Theorem 2.33 that interval uncertainty can add an additional level of complexity to a bilevel optimization problem; see also Section 2.4.2.3. For the specific robust bilevel problems in this thesis, we present problem-specific algorithms for the setting of interval uncertainty. We will investigate several variants of the Robust Bilevel Selection Problem as well as the Robust Bilevel Continuous Knapsack Problem with interval uncertainty in Sections 3.4.1.1.2, 3.4.1.2.2, and 3.4.2.1.2 and Section 4.4.2.

2.3.2.3 Discrete Uncorrelated Uncertainty

Another discrete type of uncertainty sets, which we call *discrete uncorrelated uncertainty*, shares the idea of interval uncertainty that each parameter can vary independently in some set, which now is a finite set instead of an interval. Hence, a discrete uncorrelated uncertainty set is of the form $\mathcal{U} = \prod_{e\in\mathcal{E}} \mathcal{U}_e$ with finite sets $\mathcal{U}_e \subseteq \mathbb{Q}$ that are given in the input, for all $e \in \mathcal{E}$.

The resulting set $\mathcal{U}$ is clearly a finite set again, like a discrete uncertainty set; see Section 2.3.2.1. But, in contrast to a discrete uncertainty set, its entries are not stored explicitly, which can make a difference in terms of complexity. In fact, the cardinality $|\mathcal{U}| = \prod_{e\in\mathcal{E}}|\mathcal{U}_e|$ of the uncertainty set is typically exponential in the number $\sum_{e\in\mathcal{E}}|\mathcal{U}_e|$ of parameters that describe it; therefore, the scenarios cannot be enumerated efficiently.

Like interval uncertainty (see Section 2.3.2.2), discrete uncorrelated uncertainty can often be eliminated in a trivial way since the values of c can vary independently: If all feasible solutions are nonnegative, the robust optimization problem is again equivalent to the underlying certain problem with $c \in \mathbb{Q}^{\mathcal{E}}$ defined by $c(e) = \min\ \mathcal{U}_e$ for all $e \in \mathcal{E}$.

Moreover, an uncertainty set can usually be replaced by its convex hull, resulting in an equivalent problem; see Lemma 2.26. Hence, discrete uncorrelated uncertainty is also equivalent to interval uncertainty in this sense because the convex hull of a discrete uncorrelated uncertainty set is an interval uncertainty set.

Again, we will see that the above, and in particular Lemma 2.26, does not hold in our robust bilevel setting, which makes it interesting to study discrete uncorrelated uncertainty on its own here. In fact, discrete uncorrelated uncertainty turns out to be equivalent to interval uncertainty also in case of one variant of the Robust Bilevel Selection Problem (see Section 3.4.1.1.3), while this is not the case for other variants of the Robust Bilevel Selection Problem and for the Robust Bilevel Continuous Knapsack Problem. In the latter case, interval uncertainty results in a polynomial-time solvable robust problem, but discrete uncorrelated uncertainty makes the problem $\mathcal{NP}$-hard; see Sections 4.4.2 and 4.4.3.1 and also Section 6.2.

2.3.2.4 Budgeted Uncertainty

Interval uncertainty and discrete uncorrelated uncertainty are very pessimistic because they assume that all parameters may take their worst-case values independently. In order to establish a less conservative setting, Bertsimas and Sim introduced the so-called *budgeted uncertainty* or *Gamma uncertainty* in [BS04]. Here, the most extreme parts of an interval uncertainty set are cut off by bounding the total deviation from the midpoints of the intervals. There are several different models, e.g., either the number of parameters that deviate from their midpoint or the sum of all deviations can be bounded (*discrete budgeted uncertainty* or *continuous budgeted uncertainty*; see [GH24, Chapter 3.2]). In the strictly robust setting, these models do not introduce additional complexity to the underlying certain problem, as the robust problem can be solved by a polynomial-time oracle algorithm that uses an algorithm for the underlying certain problem as an oracle. The situation gets more complicated, e.g., in case of two-stage robustness (see Remark 2.28). We refer to [BK18, Section 2.2] for more details.

In Section 4.4.4, we will prove that the Robust Bilevel Continuous Knapsack Problem with a special case of continuous budgeted uncertainty, which we call *simplicial uncertainty*, is $\mathcal{NP}$-hard and thus harder than the underlying certain Bilevel Continuous Knapsack Problem.

2.3.2.5 Polytopal Uncertainty

As a generalization of (some variants of) budgeted uncertainty, one can also consider general *polytopal uncertainty sets*. Such a set can be given by an inner description or by an outer

description, i.e., either as a convex hull of its vertices or by linear inequalities. Due to Lemma 2.26, in case of the inner description, polytopal uncertainty is equivalent to discrete uncertainty. This directly implies that also polytopal uncertainty typically increases the complexity of the underlying certain problem. The same appears to be true also in case the uncertainty set is given by an outer description; see [BK18, Section 3.1.2] for more details.

For the Robust Bilevel Continuous Knapsack Problem, the $\mathcal{NP}$-hardness results for simplicial uncertainty mentioned in Section 2.3.2.4 can also be formulated for a polytopal uncertainty set that is either given by an inner description or given by an outer description; see Section 4.4.4. Hence, also polytopal uncertainty can increase the complexity of bilevel optimization problems when robustness is added to them in the way we describe in Section 2.4.1.

2.3.2.6 Ellipsoidal Uncertainty

Another commonly studied type of uncertainty is the one of *ellipsoidal uncertainty sets*. This concept is closely related to the assumption that the uncertain parameters are normally distributed in reality; see [BK18, Section 2.3] for more details. One typically considers *uncorrelated ellipsoidal uncertainty* as a special case, corresponding to an axis-parallel ellipsoid, and to a normal distribution with a diagonal covariance matrix, i.e., no correlation between the single parameters in the distribution. For many tractable underlying certain problems, the robust counterpart under uncorrelated ellipsoidal uncertainty has been shown to be tractable as well; see [BK18, Section 3.2.2]. On the other hand, general ellipsoidal uncertainty often produces $\mathcal{NP}$-hard robust counterparts; see [BK18, Section 3.1.3].

For the Robust Bilevel Continuous Knapsack Problem, we will prove in Section 4.4.5 that even uncorrelated ellipsoidal uncertainty turns the tractable underlying certain problem into an $\mathcal{NP}$-hard problem, thus highlighting another difference of our robust bilevel setting to the typical situation in single-level optimization under strict robustness; see also Section 6.2.

2.3.2.7 Norm Uncertainty

Some of the uncertainty sets introduced so far can be seen as special cases of the more general *norm uncertainty sets*, where the uncertain parameter vector may deviate from some expected scenario within a given radius, and the deviation is measured in some given norm. Ellipsoidal uncertainty (see Section 2.3.2.6) corresponds to the 2-norm, while interval uncertainty (see Section 2.3.2.2) can be modeled using the ∞-norm. The budget constraint in some variants of budgeted uncertainty, in particular *simplicial uncertainty* (see Section 2.3.2.4), can be written using a 1-norm.

Our $\mathcal{NP}$-hardness results for the Robust Bilevel Continuous Knapsack Problem with simplicial uncertainty mentioned in Section 2.3.2.4 can be generalized to the setting of p-norm uncertainty, for every fixed $p \in [1, \infty)$; see Section 4.4.5. This matches the situation that results for ellipsoidal uncertainty can often be generalized to p-norms [BK18, Section 2.4].

2.3.3 General Observations

A central fact in robust optimization is that replacing an uncertainty set by its convex hull usually does not change the problem. For this, first recall the definition of the convex hull:

Definition 2.25. Let $\mathcal{E}$ be a finite set. Given a number $k \in \mathbb{N}$ and vectors $x_1, \dots, x_k \in \mathbb{R}^{\mathcal{E}}$, a *convex combination* of $x_1, \dots, x_k$ is a vector $x = \sum_{i=1}^{k} \lambda_i x_i$ with $\lambda_1, \dots, \lambda_k \in \mathbb{R}_{\geq 0}$ satisfying $\sum_{i=1}^{k} \lambda_i = 1$. For a set $A \subseteq \mathbb{R}^{\mathcal{E}}$, the *convex hull* $\operatorname{conv}(A)$ of A is the set of all convex combinations of points in A.

The convex hull of a finite set of vectors is a polytope. We refer to [KV18, Chapter 3.5] for more details and further basic results on convex hulls and polytopes. In our model (R), it is easy to see that the uncertainty set can be replaced by its convex hull:

Lemma 2.26. *In a robust optimization problem of the form (R), one can replace the uncertainty set $\mathcal{U}$ by its convex hull $\operatorname{conv}(\mathcal{U})$, without changing the optimum value or the set of optimum solutions of the problem.*

Proof. The adversary's problem $\min\{\sum_{e \in \mathcal{E}} c(e) x_e \mid c \in \mathcal{U}\}$, for a fixed vector $x \in \mathcal{X}$, corresponds to an optimization problem with feasible set $\mathcal{U}$ and linear objective function $x \in \mathbb{R}^{\mathcal{E}}$. (Remember that we use functions on finite sets and vectors interchangeably; see the notation introduced in the beginning of Chapter 2.) Since a linear objective function evaluated on a convex combination of points cannot be better than the best value of any of these points, this problem clearly has the same optimum value as the same problem with feasible set $\operatorname{conv}(\mathcal{U})$. Hence, for every $x \in \mathcal{X}$, the problem (R) has the same objective function value as the problem (R) with $\operatorname{conv}(\mathcal{U})$ instead of $\mathcal{U}$. Thus, the optimum value and the set of optimum solutions do not change. □

As mentioned above, this implies that some types of uncertainty sets are equivalent to each other. In particular, interval uncertainty corresponds to the convex hulls of discrete uncorrelated uncertainty sets, and the convex hull of a discrete uncertainty set forms a polytope.

However, Lemma 2.26, and hence these equivalences between types of uncertainty sets, do not hold for robustness concepts other than the strict robustness in (R) in general. In fact, some of our results for robust bilevel optimization problems imply that this is not the case for the robustness concept that we will introduce in Section 2.4.1; see Section 2.4.2.3.

Remark 2.27. When defining the uncertainty sets so far, we assumed that all parameters of the objective function, i.e., all values $c(e)$ for $e \in \mathcal{E}$, might be uncertain. In practice, only some of them might vary, while others are fixed in all scenarios. Most of the types of uncertainty sets introduced above allow for some fixed dimensions by definition. In particular, for some $e \in \mathcal{E}$, the corresponding parameter $c(e)$ can be fixed to a constant $\gamma \in \mathbb{Q}$ for all scenarios in a discrete uncertainty set, in an interval uncertainty set or a budgeted uncertainty set (by setting $c^-(e) = c^+(e) = \gamma$), in a discrete uncorrelated uncertainty set (by setting $\mathcal{U}_e = \{\gamma\}$) or in a polytopal uncertainty set (by setting $c(e) = \gamma$ in the description of the polytope). However, ellipsoidal and general norm uncertainty sets, as introduced above, do not necessarily allow for fixing some dimension. We will still assume that this is possible and that, e.g., the ellipsoid is only defined on some dimensions in this case, i.e., that the uncertainty set has the form $\mathcal{U} = \overline{\mathcal{U}} \times \{\overline{c}\}$, where $\overline{\mathcal{U}} \subseteq \mathbb{R}^{\mathcal{E} \setminus \overline{\mathcal{E}}}$ is an ellipsoid, and $\overline{c} \in \mathbb{Q}^{\overline{\mathcal{E}}}$ models the parameters that are not uncertain, where $\overline{\mathcal{E}} \subseteq \mathcal{E}$ corresponds to the dimensions containing fixed values.

Remark 2.28. The robustness concept considered so far is also known as *strict robustness* and only forms a small part of the current theory of robust optimization. More advanced

robustness concepts comprise, e.g., two-stage adjustable, recoverable, and min–max regret robustness. For an introduction to these approaches, we refer to, e.g., [GH24] and [BK18, Section 4]. As we will combine robustness with bilevel optimization problems and this results in much more complicated problems anyway (see Section 2.4), we only focus on strict robustness in this thesis. However, also the other robustness concepts mentioned above are interesting in our context because they can add additional stages of complexity to the underlying certain problem as well; see, e.g., [Woe21, Section 5] and [DW10; GLW23].

Remark 2.29. As already indicated in Section 2.3.1, bilevel optimization and (strict) robust optimization are closely related in their structure, as both concepts deal with two stages of decisions that are made one after the other in a hierarchical manner. Bilevel optimization, however, is more flexible than robust optimization in some aspects: Leader and follower can have completely different objective functions, while the adversary in robust optimization has exactly the opposite objective of the main decision maker by definition. In addition, the adversary's feasible set in robust optimization is restricted to special structures of uncertainty sets and does often not depend on the decisions made in the first stage, whereas the follower's constraints in bilevel optimization typically contain the leader's variables.

The recent work [GKST25] investigates the close connection of bilevel and robust optimization (including several different robustness concepts; see Remark 2.28), and gives reformulations of robust problems as bilevel problems, and vice versa. Regarding the computational complexity of the involved problem classes, there are still many open questions. For robust combinatorial optimization problems of type (R), it is known that the robust counterpart of a tractable underlying certain problem is $\mathcal{NP}$-hard in many cases; see Sections 2.3.2 and 2.3.4. On the other hand, the adversary's problem is solvable in polynomial time for the typical uncertainty sets, which implies that the robust problem is $\mathcal{NP}$-easy, analogously to Lemma 2.32 for bilevel optimization problems with binary leader's variables and a tractable follower's problem. However, in case of uncertainty sets inducing $\mathcal{NP}$-hard adversary's problems, robust optimization problems can be Σ_2^{p}-hard, even in the strict robustness setting [CS20]. This is in line with the Σ_2^{p}-hardness of bilevel optimization problems with an $\mathcal{NP}$-hard follower's problem; see Section 2.2.2.1.

We also refer to [Woe21], where Woeginger discusses several classes of optimization problems that naturally fall into the second level of the polynomial-time hierarchy, including bilevel and robust optimization problems.

2.3.4 Literature Overview of Robust Combinatorial Optimization Problems

The robust counterparts of many specific (combinatorial) optimization problems have been investigated in the literature. We only mention a small selection of results here, with a focus on strict robustness (in contrast to the robustness concepts mentioned in Remark 2.28), complexity results, and the underlying certain problems that are closest to the ones we deal with in this thesis. For a broader overview and more results, we refer to, e.g., [KY97; ABV09; KZ16; BK18; GH24].

The robust counterparts of the Selection Problem (see Section 3.1) and of the Minimum Spanning Tree Problem (see Section 5.1) with discrete uncertainty are typical examples of the situation described in Section 2.3.2.1: They are weakly $\mathcal{NP}$-hard already in case of two scenarios and strongly $\mathcal{NP}$-hard in case of an unbounded number

of scenarios; see [Ave01; KZ09] and [KY97, Chapter 3.3]. For uncorrelated ellipsoidal uncertainty, the robust counterparts of the Selection Problem and of the Minimum Spanning Tree Problem (and of the general Minimum Matroid Basis Problem; see Section 6.1) are solvable in polynomial time; see [Nik10].

In case of the Knapsack Problem (see Section 4.1), already the certain problem is weakly $\mathcal{NP}$-hard and admits a pseudopolynomial-time algorithm. The robust counterpart in case of discrete uncertainty with a constant number of scenarios is still solvable in pseudopolynomial time. In case of an unbounded number of scenarios however, the robust Knapsack Problem is again strongly $\mathcal{NP}$-hard. For more details, see [KY97, Chapter 3.7] and [Yu96; ABV09]. For the case of budgeted uncertainty, a pseudopolynomial-time dynamic programming algorithm for the robust Knapsack Problem is described in [MPS13].

2.4 Bilevel Optimization Under Uncertainty

In bilevel optimization, as introduced in Section 2.2, one assumes that the leader has full knowledge of the follower's problem when making her own decision. In a practical setting in which leader and follower are actually two distinct decision makers, this assumption might not be very realistic. It rather seems to be very natural here to adopt some concept of optimization under uncertainty, as the ones mentioned in Section 2.3, and assume that the leader is uncertain about some parameters of the follower's problem. Generally speaking, this idea can be applied to any part of the follower's problem and using any way of modeling uncertainty. In this thesis, we will focus on uncertainty in the parameters of the follower's objective function and on the (strict) robust optimization approach (see Section 2.3), investigating different types of uncertainty sets. We will next formalize this concept in Section 2.4.1 and then present some complexity results for general robust bilevel optimization problems in Section 2.4.2.

2.4.1 Introduction to Robust Bilevel Optimization with Uncertain Follower's Objective

Consider the notation for bilevel optimization problems from Section 2.2.1 again. We now add robust uncertainty concerning the follower's objective function $\mathcal{D}$, in the sense of Section 2.3.1, to the leader's problem (L). That is, when $\mathcal{U}$ denotes the uncertainty set containing all possible realizations of $\mathcal{D}$, we add $\min_{\mathcal{D}\in\mathcal{U}}$ in front of the leader's objective $\mathcal{C}(x,y)$:

$$\begin{aligned} \max_{x\in\mathcal{L}} \ \min_{\mathcal{D}\in\mathcal{U}} \ & \mathcal{C}(x,y) \\ \text{s.t. } & y \in \operatorname*{argmax}_{y'} \mathcal{D}(x,y') \\ & \text{s.t. } y' \in \mathcal{F}_x \end{aligned} \tag{R-L}$$

As in single-level robust optimization, the leader optimizes her worst case over all scenarios in $\mathcal{U}$, where each scenario now corresponds to a different follower's objective function and thus to a possibly different follower's reaction to the same leader's choice. The situation can also be illustrated by considering three decision makers: The *leader* first chooses a solution $x \in \mathcal{L}$, then an *adversary* (of the leader) chooses an objective function $\mathcal{D} \in \mathcal{U}$ for the follower, and the *follower* finally computes an optimum solution $y \in \mathcal{F}_x$ according to

this objective function $\mathcal{D}$ and depending on the leader's choice x. The leader's aim is to optimize her own objective value $\mathcal{C}(x, y)$, which depends on the follower's choice y, while the adversary has the opposite objective.

The adversary's problem, i.e., the problem of evaluating the leader's objective function in (R-L), for a fixed leader's solution $x \in \mathcal{L}$, can be written as:

$$\begin{aligned} \min_{\mathcal{D}} \ & \mathcal{C}(x, y) \\ \text{s.t. } & \mathcal{D} \in \mathcal{U} \\ & y \in \operatorname*{argmax}_{y'} \mathcal{D}(x, y') \\ & \qquad \text{s.t. } y' \in \mathcal{F}_x \end{aligned} \tag{A}$$

This problem turns out to be a bilevel optimization problem again, however with the leader's constraints having a very specific structure now, depending only on the uncertainty set $\mathcal{U}$.

The follower's problem (F) in the robust bilevel setting is the same as in Section 2.2.1 and needs to be considered for every fixed leader's solution $x \in \mathcal{L}$ and adversary's decision $\mathcal{D} \in \mathcal{U}$.

Like in certain bilevel optimization, for a precise problem definition, one has to clarify the follower's behavior in case his optimum solution is not unique. In fact, both (R-L) and (A) are imprecise as stated above when it comes to the selection of y. As before, one can adopt the optimistic or the pessimistic approach; see (L-opt) and (L-pess) in Section 2.2.1. Note that the consideration of the leader's objective as a secondary criterion in the follower's problem now takes place after the adversary's decision in the sequence of decisions outlined above.

Let $\mathcal{F}^*_{x,\mathcal{D}}$ denote the set of optimum solutions of the follower's problem (F), for every fixed $x \in \mathcal{L}$ and $\mathcal{D} \in \mathcal{U}$. The pessimistic version of the robust bilevel optimization problem can now be written as

$$\max \left\{ \min\{\mathcal{C}(x, y) \mid \mathcal{D} \in \mathcal{U}, y \in \mathcal{F}^*_{x,\mathcal{D}}\} \,\middle|\, x \in \mathcal{L} \right\} \tag{R-L-pess}$$

and the optimistic version as

$$\max \left\{ \min \left\{ \mathcal{C}(x, y) \,\middle|\, \mathcal{D} \in \mathcal{U}, y \in \operatorname{argmax}\{\mathcal{C}(x, y') \mid y' \in \mathcal{F}^*_{x,\mathcal{D}}\} \right\} \,\middle|\, x \in \mathcal{L} \right\}. \tag{R-L-opt}$$

Note that, in contrast to certain bilevel optimization, the optimistic setting is more complicated now than the pessimistic one. In the optimistic version (L-opt) of the certain bilevel optimization problem, the optimism can be seen as being integrated into the leader's decision, while the pessimism adds another level to the formulation in (L-opt). Similarly, in robust bilevel optimization, the pessimism can be seen to be part of the adversary's decision in (R-L-pess), and the optimism adds another stage in (R-L-opt). The pessimistic assumption can also be considered to be more natural here because adopting the robust optimization approach corresponds to a conservative and pessimistic attitude of the leader anyway.

Remark 2.30. Since the adversary can influence only the follower's objective but not his constraints, the feasibility of the follower's problem still depends only on the leader's and not on the adversary's decision in our model. If there were also uncertain parameters in the follower's constraints, the leader would need to ensure that the follower's problem is feasible for all possible realizations, similarly to single-level robust optimization with uncertainty in the constraints. But also in our setting (R-L), it could happen that the

follower's problem does not have an optimum solution for each realization (e.g., due to the follower's feasible set not being compact), and the leader also has to avoid this situation. However, this does not apply to the problems considered in this thesis.

A very recent and comprehensive overview of the area of bilevel optimization under uncertainty is given by Beck, Ljubić, and Schmidt in the survey [BLS23]. It covers both stochastic and robust approaches to model uncertainty, and it classifies various ways of where and how uncertainty can come into play in a bilevel optimization problem. In contrast to a *wait-and-see follower* as in our setting, one can also consider a *here-and-now follower* who decides after the leader but before the uncertainty is realized. The latter corresponds to a robust and thus more complex follower's problem, while the uncertainty has no direct influence on the leader. On the other hand, the wait-and-see follower models the natural situation in which the leader is uncertain about the follower's data or preferences when anticipating his reaction. It is also a very interesting setting as we will see that results from single-level robust optimization cannot be applied directly. Moreover, apart from the assumption of *data uncertainty* described so far, the interaction between the two decision makers offers several additional types of so-called *decision uncertainty*; for instance, the follower can be uncertain when he observes the leader's solution, or the follower might not be able to obtain an optimum but only a reasonably good solution, again imposing uncertainty for the leader.

A main goal of this thesis is to improve the understanding of robust bilevel optimization problems of type (R-L) and, in particular, of the additional complexity that this way of respecting uncertainty introduces in bilevel optimization problems. In the next section, we will obtain some complexity results for a general class of robust bilevel optimization problems. In Chapters 3 and 4, the concept will be applied to specific bilevel optimization problems, namely the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem, and we will again study the computational complexity of the resulting robust bilevel optimization problems. The complexity of the corresponding stochastic version of the Bilevel Continuous Knapsack Problem is investigated in [BHI22].

2.4.2 Complexity of Robust Bilevel Optimization with Uncertain Follower's Objective

In this section, our aim is to investigate the computational complexity of robust bilevel optimization problems having the form of (R-L) relative to the complexity of the corresponding certain problem (L), with respect to different types of uncertainty sets. The results of this section have already appeared in [BHH21].

Here we focus on the setting where we have $\mathcal{L} \subseteq \{0,1\}^p$ for some $p \in \mathbb{N}$, i.e., the leader's problem can be seen as a combinatorial optimization problem, while the follower's problem is a linear program with only the right-hand side depending (affine-linearly) on the leader's decision x. More formally, we assume that there are $m, n \in \mathbb{N}$, $A \in \mathbb{Q}^{m \times n}$, $B \in \mathbb{Q}^{m \times p}$, and $b \in \mathbb{Q}^m$ such that $\mathcal{F}_x = \{y \in \mathbb{R}^n \mid Ay \leq Bx + b\}$ for all $x \in \mathcal{L}$. For simplicity, we assume throughout that $\mathcal{F}_x$ is nonempty and bounded for all $x \in \mathcal{L}$, so that an optimal follower's response always exists. Moreover, we restrict ourselves to the case in which both objective functions depend linearly on the follower's decision but do not depend on the leader's decision. Therefore, we will write them as $\mathcal{C}(x,y) = c^\top y$ and $\mathcal{D}(x,y) = d^\top y$

for all $x \in \mathcal{L}$ and $y \in \mathcal{F}$, with vectors $c, d \in \mathbb{Q}^n$. Like in the problem formulation (R) in Section 2.3.1, the uncertainty set $\mathcal{U}$ will be interpreted as a set of vectors d instead of a set of functions $\mathcal{D}$, i.e., $\mathcal{U} \subseteq \mathbb{R}^n$, throughout this section.

In summary, the robust bilevel optimization problem (R-L) with these assumptions can now be written as

$$\begin{aligned} \max_{x \in \mathcal{L}} \ & \min_{d \in \mathcal{U}} c^\top y \\ & \text{s.t. } y \in \operatorname*{argmax}_{y' \in \mathbb{R}^n} d^\top y' \\ & \qquad \text{s.t. } Ay' \leq Bx + b. \end{aligned} \tag{R-L$'$}$$

Correspondingly, the adversary's problem (A), for a fixed leader's solution $x \in \mathcal{L}$, becomes

$$\begin{aligned} \min_{d} \ & c^\top y \\ & \text{s.t. } d \in \mathcal{U} \\ & \quad y \in \operatorname*{argmax}_{y' \in \mathbb{R}^n} d^\top y' \\ & \qquad \text{s.t. } Ay' \leq Bx + b, \end{aligned} \tag{A$'$}$$

and the follower's problem (F), for fixed $x \in \mathcal{L}$ and $d \in \mathcal{U}$, now reads

$$\begin{aligned} \max_{y \in \mathbb{R}^n} \ & d^\top y \\ & \text{s.t. } Ay \leq Bx + b. \end{aligned} \tag{F$'$}$$

Corresponding to (L), we can formulate the underlying certain problem of (R-L$'$), for a fixed scenario $d \in \mathbb{Q}^n$, as

$$\begin{aligned} \max_{x \in \mathcal{L}} \ & c^\top y \\ & \text{s.t. } y \in \operatorname*{argmax}_{y' \in \mathbb{R}^n} d^\top y' \\ & \qquad \text{s.t. } Ay' \leq Bx + b. \end{aligned} \tag{L$'$}$$

Remark 2.31. The assumption that the leader's and the follower's objective functions do not depend on the leader's decision is not a restriction compared to the setting of objectives having the form $\mathcal{C}(x, y) = c_x^\top x + c_y^\top y$ and $\mathcal{D}(x, y) = d_x^\top x + d_y^\top y$: In case of the follower's objective, the term $d_x^\top x$ is constant from the follower's perspective anyway, and in case of the leader's objective, the term $c_x^\top x$ can be modeled using an additional follower's variable z together with an additional follower's constraint $z = c_x^\top x$.

Also the assumption that the leader's variables are discrete is not a restriction because binary leader's variables can be modeled using continuous leader's and follower's variables, as outlined in Section 2.2.2.1. This implies that the hardness results we obtain in this section also hold for bilevel (continuous) linear programs that are often considered in the literature; see Section 2.2.2.1. For the result of Theorem 2.33, we explain the argument in more detail again in Remark 2.35.

For given x and d, the follower's problem (F$'$), as a linear program, is always tractable. This holds not only in terms of finding *any* optimum solution, but also if the task includes finding an optimum solution that is best or worst possible for the leader (see Remark 2.22).

This amounts to optimizing two linear objective functions lexicographically, which can be done efficiently [SS83]. This can be used to prove that the leader's optimization problem (L′) is $\mathcal{NP}$-easy. Indeed, its decision variant (i.e., the question whether some given objective value can be achieved or exceeded by a feasible solution) belongs to $\mathcal{NP}$, the corresponding feasible solution $x \in \mathcal{L} \subseteq \{0,1\}^p$ being the certificate. A polynomial reduction from an optimization problem of such a form to its decision variant can be achieved using binary search; see, e.g., [KV18, Proposition 15.44]. This shows:

Lemma 2.32. *The leader's problem (L′) is $\mathcal{NP}$-easy.*

Note that the same result can also be shown for bilevel linear programs, while the certificate has to be chosen more carefully in case of continuous leader's variables [Buc23]; see also Section 2.2.2.1.

Similarly to the proof of Lemma 2.32, it can be verified that the robust problem (R-L′) can be at most one level harder, in the polynomial-time hierarchy, than the adversary's problem (A′), i.e., the evaluation of the objective function of (R-L′). The complexity of (R-L′) and (A′) strongly depends on the uncertainty set $\mathcal{U}$. We will see that (R-L′) can be Σ_2^{p}-hard and (A′) $\mathcal{NP}$-hard in case of interval uncertainty (Theorem 2.33), while in case of discrete uncertainty, (A′) is tractable and thus (R-L′) is $\mathcal{NP}$-easy (Theorem 2.39). This is a remarkable difference to classical (single-level) robust optimization where interval uncertainty in the objective function does not make a problem harder, while many tractable combinatorial optimization problems have an $\mathcal{NP}$-hard robust counterpart in case of discrete uncertainty; see Sections 2.3.2.2 and 2.3.2.1.

2.4.2.1 Interval Uncertainty

We start by considering the case of interval uncertainty, i.e., uncertainty sets of the form $\mathcal{U} = [d_1^-, d_1^+] \times \cdots \times [d_n^-, d_n^+]$ for given $d^-, d^+ \in \mathbb{Q}^n$ with $d^- \leq d^+$. We will prove that, in this case, the robust counterpart (R-L′) is significantly harder than the underlying problem (L′) in general, while the adversary's problem (A′) is significantly harder than the follower's problem (F′).

For the reductions, we will use the $\mathcal{NP}$-complete Satisfiability Problem (see Section 2.1.1.2) and the Σ_2^{p}-complete $\exists\forall$-Quantified Satisfiability Problem (see Section 2.1.4). For both problems, we may assume that the Boolean formula is given in the input by a recursive definition using negations and bivariate conjunctions and disjunctions. For instance, every Boolean formula that is given in conjunctive or disjunctive normal form with multivariate conjunctions and disjunctions can be equivalently written in this representation with only a polynomial overhead. However, in the following, we do not assume the Boolean formulas to be given either in conjunctive or disjunctive normal form such that we do not have to switch between the two depending on the current level of the polynomial-time hierarchy and on the question if we are dealing with the original problems or their complements; see Remarks 2.7 and 2.19.

Theorem 2.33. *In case of interval uncertainty, the robust counterpart (R-L′) is Σ_2^{p}-hard and the adversary's problem (A′) is $\mathcal{NP}$-hard, even if $\mathcal{L} = \{0,1\}^p$.*

Proof. We prove the Σ_2^{p}-hardness of (R-L′) by a polynomial reduction from the Σ_2^{p}-complete $\exists\forall$-Quantified Satisfiability Problem (see Section 2.1.4). Since (R-L′) can be at

most one level harder than (A′), as mentioned in the introduction of Section 2.4.2, the $\mathcal{NP}$-hardness of (A′) directly follows. Furthermore, a polynomial reduction from the complement of the Satisfiability Problem (which is co-$\mathcal{NP}$-complete; see Section 2.1.1.2) to (A′) is implicitly contained in our reduction from the ∃∀-Quantified Satisfiability Problem to (R-L′).

For a given Boolean formula $f\colon \{0,1\}^p \times \{0,1\}^n \to \{0,1\}$, consider the robust bilevel problem

$$\begin{aligned} \max_{x\in\{0,1\}^p} \ \min_{d\in[-1,1]^n} \ & f(x,y) \\ & \text{s.t. } y \in \operatorname*{argmax}_{y'} d^\top y' \\ & \qquad\qquad \text{s.t. } y' \in [0,1]^n. \end{aligned} \tag{2.1}$$

This problem can be linearized in the standard way: Introduce a new variable with domain $[0,1]$ for each intermediate term in the recursive definition of f, including f itself, and model the corresponding nonlinear relations between all variables by appropriate linear inequalities (for conjunctions and disjunctions) or linear equations (for negations); see, e.g., [Jer85]. In order to obtain a reformulation of (2.1) in the form of (R-L′) with $\mathcal{L} = \{0,1\}^p$, we add all new variables and linear constraints to the follower's problem. For all newly introduced variables, we set the corresponding entries of d^- and d^+ to 0, such that they are not affected by the uncertainty; see also Remark 2.27.

We first consider the optimistic setting. Assume that f is a no-instance of the ∃∀-Quantified Satisfiability Problem. Then, for all leader's solutions $x \in \{0,1\}^p$, there exists some $y \in \{0,1\}^n$ such that $f(x,y) = 0$. By construction of (2.1), the adversary can enforce any follower's response $y \in \{0,1\}^n$ by appropriately choosing $d \in \{-1,1\}^n$. In that case, since x and y are binary, the linearization forces all additional variables to the corresponding binary values as well, including the variable representing the leader's objective function. This shows that the optimum value of (2.1) is 0 in this case.

If f is a yes-instance, there exists an $x \in \{0,1\}^p$ such that $f(x,y) = 1$ for all $y \in \{0,1\}^n$. Then we claim that this leader's solution x yields an objective value of 1 in the linearization of (2.1). Indeed, for every adversary's choice d, there is an optimum follower's solution such that y is binary. By construction, each such solution leads to $f(x,y) = 1$. Since we adopt the optimistic approach, the follower will thus select some solution y with $f(x,y) = 1$.

Finally, assume that we adopt the pessimistic approach. In this case, we need to modify the construction of (2.1) as follows. For every $i \in \{1,\dots,n\}$, we introduce a new follower's variable $\overline{y}_i$, along with linear constraints

$$\overline{y}_i \geq 0, \quad \overline{y}_i \leq y_i, \quad \overline{y}_i \leq 1 - y_i.$$

Each new variable $\overline{y}_i$ has coefficient M in the leader's objective, where $M \geq 3$ is at least the number of atomic terms in f (i.e., the total number of occurrences of all Boolean variables in the given representation of f), and a certain coefficient of 1 in the follower's objective. This ensures that $\overline{y}_i = \min\{y_i, 1 - y_i\}$ holds in every optimum follower's solution, i.e., the variable $\overline{y}_i$ models the deviation of y_i from the closest binary value.

The resulting follower's objective $d_i y_i + \overline{y}_i$, in every dimension $i \in \{1,\dots,n\}$, depends on the adversary's choice d_i and always consists of two linear pieces, as a function in y_i; see Figure 2.1. If the adversary chooses $d \in \{-1,1\}^n$, the follower's optimum solutions satisfy $y_i = \overline{y}_i \in [0, 1/2]$ or $y_i = 1 - \overline{y}_i \in [1/2, 1]$, respectively. By setting all y_i to binary values, the

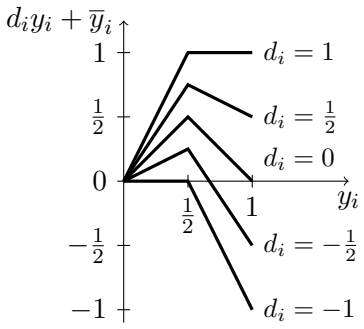


Figure 2.1. Illustration of the follower's objective function in the pessimistic setting in the proof of Theorem 2.33. For a fixed dimension $i \in \{1, \dots, n\}$, the value $d_i y_i + \overline{y}_i$ is displayed in dependence of the value $y_i \in [0, 1]$ the follower selects, for different values of d_i chosen by the adversary.

follower can achieve a leader's objective value of at most 1 then, due to the pessimistic assumption. If the adversary chooses some $d_i \in (-1, 1)$, then, by construction, the value of $(y_i, \overline{y}_i)$ in any optimum follower's solution is $(1/2, 1/2)$, so that the leader's objective is at least $M/2 > 1$, which is never optimum for the adversary. Hence, the adversary must always choose $d \in \{-1, 1\}^n$.

For a no-instance f and any $x \in \{0, 1\}^p$, the adversary can now choose $d \in \{-1, 1\}^n$ such that the follower has a binary vector $y \in \{0, 1\}^n$ in his set of optimum solutions for which $f(x, y) = 0$ holds. By the pessimistic assumption, the follower actually chooses y such that the leader's objective value is 0.

Assume, on the contrary, that f is a yes-instance, and consider some leader's solution $x \in \{0, 1\}^p$ such that $f(x, y) = 1$ for all $y \in \{0, 1\}^n$. Any adversary's choice $d \in \{-1, 1\}^n$, together with any binary follower's solution y then results in an objective value of 1 for the leader. Since we adopt the pessimistic approach, it remains to show that the follower cannot achieve a smaller objective value for the leader by choosing y_i nonbinary and exploiting the resulting flexibility when setting the linearization variables. However, one can verify that this is not possible because the gain in the variable representing $f(x, y)$ would be exceeded by the punishment due to the terms $M\overline{y}_i$, by definition of M; see, e.g., [Jer85, Lemma 4.1]. Thus, the follower selects $y \in \{0, 1\}^n$ and the leader's optimum value is indeed 1. □

Note that the underlying certain problem of (2.1) is $\mathcal{NP}$-equivalent. Indeed, it is $\mathcal{NP}$-hard even when $n = 0$, i.e., when the follower only controls the additional variables introduced for linearizing f, which are uniquely determined by the leader's solution $x \in \{0, 1\}^p$. The problem is then equivalent to the Satisfiability Problem. On the other hand, the problem is $\mathcal{NP}$-easy, for any n; see Lemma 2.32.

Remark 2.34. In the proof of Theorem 2.33, we argued that, if all players in the robust problem decide optimally, all variables can be assumed to attain binary values. However, for the underlying certain problem with a fixed follower's objective d, this is not necessarily the case. More precisely, the optimum follower's choice as well as the leader's optimum value in the underlying certain problem might not be binary if $n \neq 0$, for some values of d. In particular, if some or all coefficients in the follower's objective are 0, the follower may select any values in $[0, 1]$ for the corresponding variables and is only guided by the

optimistic or pessimistic assumption. This can make it optimal for him to exploit the behavior of the linearization in case of nonbinary variables. As an example, consider the simple case of no leader's and one follower's variable and the Boolean formula $y \vee \overline{y}$, in the optimistic setting. In the linearization, it is modeled by an additional follower's variable $z \in [0,1]$ and additional follower's constraints $z \geq y$ and $z \geq 1 - y$. Obviously, the leader's objective z is then maximized by setting $y = z = 1/2$.

Remark 2.35. As indicated in Remark 2.31, in order to strengthen the statement of Theorem 2.33, the integrality constraints in the leader's problem can be relaxed without changing the result, using the idea explained in Section 2.2.2.1. For clarification, we repeat the argument here for the proof of Theorem 2.33, as it uses a similar construction as the proof in the pessimistic case (and [Jer85]): For each $i \in \{1, \dots, p\}$, we can introduce a new follower's variable $\overline{x}_i \in [0,1]$ with coefficient $-M$ in the leader's objective, where $M \geq 3$ is again at least the number of atomic terms in f, with certain coefficient 1 in the follower's objective, and with additional follower's constraints $\overline{x}_i \geq 0$, $\overline{x}_i \leq x_i$, and $\overline{x}_i \leq 1 - x_i$. This ensures $\overline{x}_i = \min\{x_i, 1 - x_i\}$ in any optimum follower's solution and, by the choice of M, that $x \in \{0,1\}^p$ holds in any optimum leader's solution. We obtain a problem of type (R-L′) again, but without integrality constraints. Note that this construction does not interfere with the one in the proof of Theorem 2.33 in the pessimistic case because the latter is relevant only in case of a yes-instance, whereas the former is relevant only in case of a no-instance.

If, on the other hand, binarity constraints were allowed both in the leader's and in the follower's problem, a similar construction as in Theorem 2.33 would actually yield a Σ_3^{p}-hard problem, in the optimistic setting. For this, consider the Σ_3^{p}-complete

∃∀∃-Quantified Satisfiability Problem
Instance: A Boolean formula $f\colon \{0,1\}^p \times \{0,1\}^n \times \{0,1\}^q \to \{0,1\}$
Task: Decide whether there exists some $x \in \{0,1\}^p$ such that for all $y \in \{0,1\}^n$ there exists some $z \in \{0,1\}^q$ with $f(x,y,z) = 1$.

One could reduce the ∃∀∃-Quantified Satisfiability Problem to (R-L′) by considering two separate sets of variables in the follower's problem, the first one corresponding to y and being "controlled" by the adversary (i.e., with $d_i^- = -1$ and $d_i^+ = 1$) and the second one corresponding to z and being "controlled" by the follower (i.e., with $d_i^- = d_i^+ = 0$). The adversary's problem is then equivalent to the complement of the ∃∀-Quantified Satisfiability Problem and thus Σ_2^{p}-hard. The underlying certain problem is equivalent to the Satisfiability Problem, similarly as in Theorem 2.33, and thus $\mathcal{NP}$-equivalent. However, in contrast to the setting of Theorem 2.33, the same holds for the follower's problem here. Hence, the increase in complexity of the robust problem relative to the underlying certain problem is strictly greater here than in Theorem 2.33, while it remains the same relative to the follower's problem.

Remark 2.36. The result of Theorem 2.33 does not only hold for interval uncertainty sets, but also for many other uncertainty sets $\mathcal{U}$ where each entry of d can be chosen to be positive or negative independently of each other, i.e., where $\mathcal{U}$ contains a point in the interior of every orthant of $\mathbb{R}^n$. In particular, the proof of Theorem 2.33 can be easily adapted for the cases of discrete uncorrelated uncertainty (see Section 2.3.2.3) and of uncertainty sets which contain the origin in their interior.

2.4.2.2 Discrete Uncertainty

We next address the case of a discrete uncertainty set $\mathcal{U}$. For this case, we first show that robust counterparts of bilevel problems are at least as hard as robust counterparts of single-level problems, for each underlying feasible set of the leader.

Lemma 2.37. *For any class of underlying sets $X \subseteq \{0,1\}^p$, the robust counterpart of linear optimization over X subject to discrete uncertainty can be polynomially reduced to a problem of type (R-L′), with the same number of scenarios.*

Proof. Starting from the single-level robust counterpart

$$\begin{aligned} \max_{x} \ & \min_{\overline{c}\in\overline{\mathcal{U}}} \ \overline{c}^\top x \\ \text{s.t. } & x \in X \end{aligned} \tag{2.2}$$

of linear optimization over some set $X \subseteq \{0,1\}^p$, with $\overline{\mathcal{U}} = \{\overline{c}_1, \dots, \overline{c}_l\} \subseteq \mathbb{Q}^p$, we construct the robust bilevel problem

$$\begin{aligned} \max_{x\in X} \ & \min_{d\in\mathcal{U}} \ y \\ & \text{s.t. } (y,z) \in \operatorname*{argmax}_{y',z'} \ d^\top z' \\ & \qquad \text{s.t. } y' = \sum_{j=1}^{l}\sum_{i=1}^{p} \overline{c}_{ji} x_i z'_j \\ & \qquad\qquad y' \in \mathbb{R}, \quad z' \in \mathbb{R}^l_{\geq 0}, \quad \sum_{j=1}^{l} z'_j = 1 \end{aligned} \tag{2.3}$$

with uncertainty set $\mathcal{U} = \{e_1, \dots, e_l\}$, where e_j denotes the j-th unit vector of $\mathbb{R}^l$, for $j \in \{1, \dots, l\}$. When the adversary chooses $d = e_j$ for some $j \in \{1, \dots, l\}$, the follower's unique optimum solution consists of $z = e_j$ and $y = \overline{c}_j^\top x$. Thus, (2.2) and (2.3) are equivalent.

It remains to show that we can linearize (2.3) and thus turn it into the form required in (R-L′). For this, we introduce follower's variables u_{ji} and constraints

$$u_{ji} \geq 0, \ u_{ji} \geq x_i + z'_j - 1, \ u_{ji} \leq x_i, \ u_{ji} \leq z'_j$$

for all $j \in \{1, \dots, l\}$ and $i \in \{1, \dots, p\}$, and replace the first constraint in the follower's problem by

$$y' = \sum_{j=1}^{l}\sum_{i=1}^{p} \overline{c}_{ji} u_{ji}.$$

For every $d \in \mathcal{U}$, the unique optimum follower's solution satisfies $z \in \{0,1\}^l$. Hence, this linearization ensures $u_{ji} = x_i z_j$ and thus yields a problem equivalent to (2.3), having the form of (R-L′). Formally, all scenarios $d \in \mathcal{U}$ must be extended by zeros in the entries corresponding to the variables y' and u; see also Remark 2.27. □

Note that all optimum follower's solutions in the proof of Lemma 2.37 are uniquely determined, so that the result holds for both the optimistic and the pessimistic setting.

The robust counterpart of a single-level combinatorial optimization problem under discrete uncertainty often turns out to be $\mathcal{NP}$-hard even for two scenarios, and strongly $\mathcal{NP}$-hard when the number of scenarios is not fixed; this holds true even for unrestricted binary optimization [BK18]; see also Section 2.3.2.1. Together with Lemma 2.37, this immediately implies

Corollary 2.38. *Problem (R-L′) with $\mathcal{L} = \{0,1\}^p$ is $\mathcal{NP}$-hard for $|\mathcal{U}| = 2$ and strongly $\mathcal{NP}$-hard for finite $\mathcal{U}$, even when (L′) is solvable in polynomial time.*

On the other hand, the complexity of (R-L′) can be bounded from above as follows: As mentioned in the introduction of Section 2.4.2, (R-L′) can be at most one level harder than (A′). Moreover, the adversary's problem (A′) can be polynomially reduced to the follower's problem (F′) in case of discrete uncertainty by enumerating all scenarios and solving (F′) for each of them. For our setting, in which (F′) is solvable in polynomial time, this immediately implies the following result:

Theorem 2.39. *Let $\mathcal{U}$ be a finite set, given explicitly as part of the input. Then the adversary's problem (A′) is solvable in polynomial time and the robust problem (R-L′) is $\mathcal{NP}$-easy.*

When replacing the uncertainty set $\mathcal{U}$ in Theorem 2.39 by its convex hull, we obtain a significantly harder problem:

Theorem 2.40. *Let $\mathcal{U}$ be defined as the convex hull of a finite set of vectors, which are given explicitly as input. Then the robust counterpart (R-L′) is Σ_2^p-hard and the adversary's problem (A′) is $\mathcal{NP}$-hard, even if $\mathcal{L} = \{0,1\}^p$.*

Proof. In the proof of Theorem 2.33, we have seen that the linearized version of (2.1) is Σ_2^{p}-hard for uncertainty sets of the form $[-1,1]^n \times \{\overline{d}\}$, where $\overline{d}$ is a vector collecting all certain entries of d, corresponding to linearization variables and to artificial variables $\overline{y}$ in the pessimistic case; see also Remark 2.27. Similarly to Remark 2.36, it is easy to verify that the problem does not change when replacing $[-1,1]^n$ by a superset, e.g., by the simplex $\operatorname{conv}\{-e, -e+2ne_1, \dots, -e+2ne_n\}$, where e is the all-ones vector and e_i is the i-th unit vector of $\mathbb{R}^n$, for $i \in \{1,\dots,n\}$. Since the latter is defined by polynomially many vertices, which can all be extended by $\overline{d}$, we obtain the desired result for (R-L′). The $\mathcal{NP}$-hardness of (A′) again follows directly from the Σ_2^{p}-hardness of (R-L′). □

2.4.2.3 Comparison to Single-Level Robust Optimization

We have seen that bilevel optimization problems become significantly harder when considering uncertain follower's objective functions in a robust optimization approach. However, our results highlight a fundamental difference between the interval uncertainty case and the case of discrete uncertainty. Indeed, the construction in Theorem 2.33 yields a problem where (R-L′) is Σ_2^{p}-hard and (A′) is $\mathcal{NP}$-hard, while both problems are at least one level easier in the setting of Theorem 2.39. In this sense, interval uncertainty renders bilevel optimization problems significantly harder than discrete uncertainty, in contrast to classical single-level robust optimization (see Sections 2.3.2.1 and 2.3.2.2). However, Theorem 2.33 and Corollary 2.38 show that (R-L′) can be one level harder than (L′) for both interval and discrete uncertainty.

Moreover, Theorems 2.39 and 2.40 together imply that the complexity of both the robust counterpart (R-L′) and the adversary's problem (A′) may increase when replacing $\mathcal{U}$ by its convex hull. This again is in contrast to the single-level case, where, typically, replacing $\mathcal{U}$ by its convex hull essentially does not change the robust counterpart; see Lemma 2.26.

In Chapters 3 and 4, we will investigate the complexity of specific robust bilevel optimization problems, namely the Robust Bilevel Selection Problem and the Robust Bilevel Continuous Knapsack Problem, for different types of uncertainty sets. For a comparison of the obtained results to the ones in this section, see Section 6.2.

3 The Bilevel Selection Problem

This chapter deals with one of the easiest bilevel combinatorial optimization problems one can think of: Leader and follower each select a number of items, where a common number of items to select in total is given, and each of the two players maximizes the total value of the selected items, according to different item values. After a short introduction to the single-level problem version in Section 3.1, we will define the Bilevel Selection Problem formally and discuss several problem variants and assumptions one can consider in Section 3.2. Since the single-level Selection Problem is solvable in polynomial time, it is not surprising that the Bilevel Selection Problem is not on the second level of the polynomial-time hierarchy. However, it is not obvious whether it is polynomial-time solvable or $\mathcal{NP}$-hard in the general setting. In Section 3.3, we will prove that the Bilevel Selection Problem, and all its variants, can indeed be solved in polynomial time. Moreover, precisely because the Bilevel Selection Problem is relatively easy, we consider it to be a useful starting point for a better understanding of the additional complexity induced by uncertainty in bilevel optimization problems. We will look at the robust setting in Section 3.4, and we will see that uncertainty indeed makes the problem more complicated, sometimes in the sense that it requires more effort to develop a polynomial-time algorithm, but sometimes also in the sense of $\mathcal{NP}$-hardness. Our results are in line with and add to the complexity results obtained for general robust bilevel optimization problems in Section 2.4.2; see also Section 6.2.

This chapter also serves as a common basis for Chapters 4 and 5 because the problems considered there are both related to certain variants of the Bilevel Selection Problem. Therefore, many intuitions and results from this chapter will also be useful there. In particular, for the Bilevel Continuous Knapsack Problem in Chapter 4, many results from this chapter can be generalized. For the Bilevel Minimum Spanning Tree Problem in Chapter 5, it will turn out that it is already $\mathcal{NP}$-hard without uncertainty, and we will not investigate uncertain variants of it. However, for the certain problem, the basic considerations in Sections 3.2 and 3.3 will still be a useful preparation. For more details on the relation between the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem, we refer to Section 6.1.

As mentioned in Chapter 1, most of the contents of this chapter can also be found in [Hen25]. A stochastic version of the Bilevel Selection Problem has been investigated in the master's thesis [Irm21] and is also closely related to the stochastic version of the Bilevel Continuous Knapsack Problem in [BHI22]; see also Section 4.5.

3.1 The Selection Problem

We first define the (single-level) Selection Problem as follows:

Selection Problem
Instance: A finite set $\mathcal{E}$ of items, a number $b \in \{0, \dots, |\mathcal{E}|\}$, and item values $c\colon \mathcal{E} \to \mathbb{Q}$
Task: Find a subset $X \subseteq \mathcal{E}$ such that $|X| = b$ and $c(X)$ is maximum.

This problem is easy to solve in polynomial time because it is clearly optimal to select b items that have the highest values of c, for example, by sorting the items in $\mathcal{E}$ by nonincreasing values of c and then selecting the first b items of this order. This algorithm has running time $\mathcal{O}(n \log n)$, where $n = |\mathcal{E}|$. However, explicitly sorting the items is not even necessary in order to solve the Selection Problem: It suffices to find the b-th best item $e \in \mathcal{E}$ with respect to c, which amounts to computing a weighted median, and then select the set of all items with values at least $c(e)$. This can be done in linear time $\mathcal{O}(n)$ [BFPRT73]; see also [KV18, Chapter 17.1].

For the bilevel optimization problems investigated in this chapter, the Selection Problem will occur as a substructure, and we will often use an algorithm for the Selection Problem as a subroutine. We will mostly use the approach of first sorting the items in time $\mathcal{O}(n \log n)$, although it is the less efficient way to solve the Selection Problem. The reason for this is that we often solve several instances of the Selection Problem, with the same sets of items and the same item values, but with different numbers b, during an algorithm. In this case, it can be more efficient to sort the items once in the beginning and use the obtained order several times; see, e.g., Remark 3.19.

In addition to the Selection Problem as defined above, also a continuous version of the problem will play an important role in the following. It can be seen as the linear programming relaxation of the straightforward integer linear programming formulation of the Selection Problem, in which the selection of items for the solution X is modeled by binary variables $x \in \{0,1\}^{\mathcal{E}}$. We define the continuous problem version as follows:

Continuous Selection Problem

Instance: A finite set $\mathcal{E}$ of items, a number $b \in \mathbb{Q} \cap [0, |\mathcal{E}|]$, and item values $c\colon \mathcal{E} \to \mathbb{Q}$

Task: Find a vector $x \in [0,1]^{\mathcal{E}}$ such that $\sum_{e \in \mathcal{E}} x_e = b$ and $\sum_{e \in \mathcal{E}} c(e) x_e$ is maximum.

Note that we allow not only the decision variables, but also the given number b to be non-integer in this definition.

Clearly, the Continuous Selection Problem can be solved in polynomial time as a linear program. However, like the binary variant, it can also be solved directly in a greedy way. In fact, an optimum solution of the Continuous Selection Problem is given by an optimum solution of the corresponding Selection Problem in case b is an integer. In general, an optimum solution of the Continuous Selection Problem is given by setting $x_e = 1$ for the $\lfloor b \rfloor$ best items $e \in \mathcal{E}$ according to their values $c(e)$, setting $x_e = b - \lfloor b \rfloor$ for the next item in the order, and finally setting $x_e = 0$ for all remaining items.

Remark 3.1. For the Selection Problem as well as the Continuous Selection Problem, there might not be a unique optimum solution in case there are several items $e \in \mathcal{E}$ with the same value $c(e)$, or, more precisely, if the b-th (or $\lceil b \rceil$-th) best item with respect to c is not unique. This will become relevant when we consider bilevel optimization problems with follower's problems being a Selection Problem or a Continuous Selection Problem because we then have to distinguish between the optimistic and the pessimistic setting; see Remark 3.2. However, when solving a Selection Problem or a Continuous Selection Problem, we will always assume that the simple greedy algorithm described above is applied, with respect to some fixed nonincreasing order of the values $c(e)$. In particular, for the Continuous Selection Problem, we will always assume that a solution with at most one non-integer variable is returned, although there might be optimum solutions with more than one fractional value as well.

Moreover, also the variants of the Selection Problem and the Continuous Selection Problem will occur in which the cardinality $|X|$ and the total weight $\sum_{e\in\mathcal{E}} x_e$, respectively, is not required to equal b, but only to be bounded by b from above, i.e., the constraints $|X| = b$ and $\sum_{e\in\mathcal{E}} x_e = b$ are replaced by $|X| \leq b$ and $\sum_{e\in\mathcal{E}} x_e \leq b$, respectively. From the algorithmic perspective, this does not make a big difference: In these variants, it is never optimal to select an item of negative value, but the greedy approach on all other items is still the best strategy. Hence, these problem variants can be reduced to the ones defined above by replacing $\mathcal{E}$ by $\mathcal{E}_{\geq 0} := \{e \in \mathcal{E} \mid c(e) \geq 0\}$ and b by $\min\{b, |\mathcal{E}_{\geq 0}|\}$. (Equivalently, we could also use $c(e) > 0$ instead of $c(e) \geq 0$ in the definition of $\mathcal{E}_{\geq 0}$.) Alternatively, instead of removing items from $\mathcal{E}$ and possibly decreasing the value of b, we can add $\max\{0, b - |\mathcal{E}_{\geq 0}|\}$ dummy items e to $\mathcal{E}$, for which we set $c(e) = 0$. An optimum solution will then use these items to fill the remaining capacity if required. The value of b does not have to be changed for this construction, which is why it will be more convenient in the bilevel setting later; see Lemma 3.16.

A reduction in the other direction is given by adding some large enough $M \in \mathbb{Q}_{\geq 0}$ to the item values $c(e)$ for all $e \in \mathcal{E}$ such that all $c(e)$ become positive. This does not change the greedy order in which the items are selected, and having only positive item values leads to every optimum solution filling the capacity completely, although it is only required not to exceed it. The objective function values then only differ by the constant term Mb.

Note that, in the Continuous Selection Problem with inequality constraint, it is still true that, if b is an integer, any optimum solution of the corresponding Selection Problem with inequality constraint is also optimum for the Continuous Selection Problem, and that there is always an optimum solution with at most one fractional value in case b is not an integer.

3.2 Definition and Variants of the Bilevel Selection Problem

From now on, we will work with bilevel versions of the Selection Problem and the Continuous Selection Problem defined in Section 3.1. In this section, we will first define a basic variant of the Bilevel Selection Problem in Section 3.2.1, and then introduce and discuss modifications and assumptions that can be made to this basic problem variant in Sections 3.2.2, 3.2.3, and 3.2.4. Finally, in Section 3.2.5, we deal with all of these problem variants in more detail and prove several equivalences between them. This section is rather technical and not essential for understanding the remainder of the chapter. Therefore, readers who would like to proceed with the actual algorithmic results on the Bilevel Selection Problem and its robust versions in Sections 3.3 and 3.4 as quickly as possible may consider skipping Section 3.2.5.

3.2.1 The Basic Problem Variant

We define the main variant of the problem to investigate in this chapter as follows:

Bilevel Selection Problem

Instance: Finite leader's and follower's item sets $\mathcal{E}_l$ and $\mathcal{E}_f$, respectively, a number $b \in \{0, \dots, |\mathcal{E}_l \cup \mathcal{E}_f|\}$, and leader's and follower's item values $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$ and $d\colon \mathcal{E}_f \to \mathbb{Q}$, respectively

Task: Find a subset $X \subseteq \mathcal{E}_l$ for the leader such that $c(X \cup Y)$ is maximum, where $Y \subseteq \mathcal{E}_f \setminus X$ is a follower's subset that satisfies $|X \cup Y| = b$ and maximizes $d(Y)$.

Note that we do not require the sets $\mathcal{E}_l$ and $\mathcal{E}_f$ to be disjoint in general. However, we will discuss this assumption in Section 3.2.2 and distinguish the general problem setting from the special case in which the two sets are disjoint in large parts of this chapter.

Notation. We will denote the cardinalities of the item sets by $n_l = |\mathcal{E}_l|$, $n_f = |\mathcal{E}_f|$, and $n = |\mathcal{E}_l \cup \mathcal{E}_f|$ throughout this chapter.

As a bilevel optimization problem of type (L) (see Section 2.2.1), the Bilevel Selection Problem can be written as follows:

$$\begin{aligned} &\max_{X} \ c(X \cup Y) \\ &\text{s.t. } X \subseteq \mathcal{E}_l \\ &\quad Y \in \operatorname*{argmax}_{Y'} \ d(Y') \\ &\qquad \text{s.t. } Y' \subseteq \mathcal{E}_f \setminus X \\ &\qquad\quad |X \cup Y'| = b \end{aligned} \tag{BSP1}$$

The leader can be imagined to make her choice $X \subseteq \mathcal{E}_l$ before the follower and, at the same time, anticipate the follower's optimum solution $Y \subseteq \mathcal{E}_f \setminus X$, while optimizing her objective function c on all items selected by the leader or the follower. Leader's subsets $X \subseteq \mathcal{E}_l$ that do not allow for a feasible follower's response, i.e., for some $Y' \subseteq \mathcal{E}_f \setminus X$ such that $|X \cup Y'| = b$, are considered infeasible for the leader's problem; see also Remark 2.22. However, the requirement $b \in \{0, \dots, |\mathcal{E}_l \cup \mathcal{E}_f|\}$ ensures that a feasible (leader's and corresponding follower's) solution always exists.

From the follower's perspective, a feasible leader's solution $X \subseteq \mathcal{E}_l$ is already fixed and the task is to select a subset Y' of $b - |X|$ items from the set $\mathcal{E}_f \setminus X$, while optimizing the total value $d(Y')$ of the selected items. This amounts to a standard single-level Selection Problem and can thus be solved in a greedy way by sorting the follower's items by their values d, or by computing a weighted median; see Section 3.1.

Remark 3.2. Both the definition of the Bilevel Selection Problem and the formulation in (BSP1) are imprecise regarding the potential non-uniqueness of optimum follower's solutions. As usual, one can distinguish the optimistic and the pessimistic setting; see Section 2.2.1. For the follower's greedy algorithm, this means that c is used as a secondary criterion when non-uniqueness arises in sorting the follower's items with respect to d. More precisely, the required order can be defined such that an item $e_1 \in \mathcal{E}_f$ precedes an item $e_2 \in \mathcal{E}_f$ if either $d(e_1) > d(e_2)$ or $d(e_1) = d(e_2)$ and $c(e_1) > c(e_2)$ [$c(e_1) < c(e_2)$] in the optimistic [pessimistic] setting.

This already indicates that, for the Bilevel Selection Problem, there is not a significant difference between the two settings from an algorithmic perspective. In fact, as soon as there is any fixed order according to which the follower selects his items greedily, the problem is well-defined and the follower's optimum solution can always be assumed to be unique. Therefore, we will often implicitly assume that a fixed follower's greedy order is given, such that it will usually not be required to discuss the optimistic and the pessimistic setting in detail.

In fact, from the leader's perspective, only this order of the follower's items is relevant, but not the precise follower's item values. Accordingly, when constructing an instance, we will sometimes only define a follower's greedy order instead of a function d; see, e.g.,

Theorem 3.58 and Examples 3.61 and 3.66. Observe that, given a follower's greedy order, an appropriate function d (with polynomial-size values) can easily be constructed in polynomial time, e.g., by setting its values to subsequent natural numbers in this order.

Remark 3.3. Note that the follower's objective function considers only the items selected by the follower, while the leader's values of all items selected by either player appear in the leader's objective. This is also due to the fact that the leader's solution X is fixed from the follower's perspective. If there were additional follower's item values for the items in X, they would only amount to a constant term in his objective and would therefore not change the problem. A similar situation has also been discussed in Remark 2.31.

For some purposes, in particular when we look at continuous problem variants later on (see Section 3.2.3), the following equivalent formulation of (BSP1) as a bilevel integer linear program will be more convenient:

$$\begin{aligned}
&\max_{x} \sum_{e\in\mathcal{E}_l} c(e)x_e + \sum_{e\in\mathcal{E}_f} c(e)y_e \\
&\text{s.t. } x \in \{0,1\}^{\mathcal{E}_l} \\
&\qquad y \in \operatorname*{argmax}_{y'} \sum_{e\in\mathcal{E}_f} d(e)y'_e \\
&\qquad\qquad \text{s.t. } x_e + y'_e \le 1 \qquad\qquad \forall e \in \mathcal{E}_l \cap \mathcal{E}_f \\
&\qquad\qquad\qquad \sum_{e\in\mathcal{E}_l} x_e + \sum_{e\in\mathcal{E}_f} y'_e = b \\
&\qquad\qquad\qquad y' \in \{0,1\}^{\mathcal{E}_f}
\end{aligned} \tag{BSP2}$$

Possible leader's and follower's solutions $X \subseteq \mathcal{E}_l$ and $Y' \subseteq \mathcal{E}_f$ in (BSP1) are represented in (BSP2) by the binary variables $x \in \{0,1\}^{\mathcal{E}_l}$ and $y' \in \{0,1\}^{\mathcal{E}_f}$, using the straightforward interpretation that $x_e = 1$ if and only if $e \in X$, for each $e \in \mathcal{E}_l$, and $y'_e = 1$ if and only if $e \in Y'$, for each $e \in \mathcal{E}_f$. The constraint that the follower's solution must be disjoint from the leader's one ($Y' \subseteq \mathcal{E}_f \setminus X$ in (BSP1)) is modeled by the inequalities $x_e + y'_e \le 1$ for all $e \in \mathcal{E}_l \cap \mathcal{E}_f$ here.

We will next consider several variants of the Bilevel Selection Problem because we will investigate generalizations of different of these variants afterwards in Chapters 4 and 5. As a common basis for these chapters, the goal of this chapter is to provide a complete picture of all reasonable problem variants of the Bilevel Selection Problem.

Remark 3.4. In addition to the problem variants discussed next, one could also think about how the problem changes when only allowing for, e.g., nonnegative leader's and/or follower's item values, in contrast to arbitrary rational values as introduced so far. In fact, the setting of arbitrary item values can be easily reduced to any of the restricted settings by adding appropriate constants $C \in \mathbb{Q}_{>0}$ and $D \in \mathbb{Q}_{>0}$ to all leader's and follower's item values, respectively. The resulting problem is equivalent because, by the capacity constraint $|X \cup Y'| = b$, this changes both the leader's and the follower's objective function value of any given solution (in the latter case assuming that some fixed leader's solution X is given) only by a constant. Note that this holds for all problem variants that will be introduced in Sections 3.2.2 and 3.2.3, while the alternative Assumption ($\le$) about the capacity constraint in Section 3.2.4 changes the follower's behavior regarding negative item

values. However, also in this case, all problem variants arising from assumptions about the nonnegativity of item values are equivalent; see Remark 3.17 for details.

For the rest of the chapter, we will usually work with arbitrary rational item values. Only for the approximation algorithm of Theorem 3.56, we assume nonnegative item costs (in the minimization version of the Bilevel Selection Problem, see also Remark 3.5), which is necessary for a reasonable approximation result. Note that the construction described above does not preserve any approximation guarantees.

Remark 3.5. The Selection Problem and the Bilevel Selection Problem have been introduced as maximization problems above, but could analogously be defined as minimization problems, i.e., the decision makers could minimize the total costs of their selection of items instead of maximizing the total value. Note that [Har21; Hen25] actually consider the minimization variant of the Bilevel Selection Problem. In this thesis, the maximization variant is presented because it aligns better with the Bilevel Knapsack Problem in Chapter 4. Almost all of our arguments and approaches are valid for both versions of the problem. Only the approximation result in Theorem 3.56 appears to be specific to the minimization version and will therefore be formulated for that. Since nonnegative item costs have to be assumed for this result, minimization and maximization cannot be exchanged as easily as for other results (see also Remark 3.4).

3.2.2 Assumptions About the Item Sets

First, there are the following four assumptions we can make about the leader's and the follower's items sets:

(arb) arbitrary sets $\mathcal{E}_l$ and $\mathcal{E}_f$
(sub) $\mathcal{E}_l \subseteq \mathcal{E}_f$
(disj) $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$
(uni) $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$ and $c(e) = \gamma$ for all $e \in \mathcal{E}_l$, where $\gamma \in \mathbb{Q}$

Assumption (sub) can be seen as reasonable and convenient because it ensures that there is always a feasible follower's solution, no matter which (at most b) items the leader has selected. In the general setting, the leader might have to be careful about selecting enough items from $\mathcal{E}_l \setminus \mathcal{E}_f$ in order to get a feasible solution. However, we will show in Lemma 3.15 that Assumptions (arb) and (sub) are actually equivalent, i.e., that Assumption (sub) does not simplify the problem significantly. Therefore, we will usually, without loss of generality, restrict ourselves to the setting of Assumption (sub).

On the other hand, in Assumption (disj), the two players are assumed to control disjoint sets of items. It will turn out that this assumption actually simplifies the problem in many settings. For this reason, we will often distinguish two cases corresponding to Assumptions (sub) and (disj) later on in this chapter.

Assumption (uni) is a special case of Assumption (disj). Therefore, the algorithmic results that we will obtain for the Bilevel Selection Problem under Assumption (disj) can also be applied to the Bilevel Selection Problem under Assumption (uni), and we will usually not discuss this problem variant separately. However, some of our negative results even hold in the setting of Assumption (uni); see Lemma 3.12 and Examples 3.41 and 3.50. Moreover, the Bilevel Selection Problem under Assumption (uni) is also interesting on its own because it is equivalent to another natural bilevel variant of the Selection Problem, which will be demonstrated in the following.

As an alternative bilevel variant of the Selection Problem, we introduce a bilevel optimization problem in which the leader does not select any items herself, but instead determines the number b_f of items to be selected by the follower directly via a variable. More formally, in addition to a finite set $\mathcal{E}_f$ of follower's items and leader's and follower's item values $c, d\colon \mathcal{E}_f \to \mathbb{Q}$, respectively, we are given bounds $b_f^-, b_f^+ \in \{0, \dots, n_f\}$ and a number $\gamma \in \mathbb{Q}$. The problem can then be stated as follows:

$$\begin{aligned} \max_{b_f} \quad & -\gamma b_f + \sum_{e \in \mathcal{E}_f} c(e) y_e \\ \text{s.t. } & b_f \in \{b_f^-, \dots, b_f^+\} \\ & y \in \operatorname*{argmax}_{y'} \sum_{e \in \mathcal{E}_f} d(e) y'_e \\ & \qquad \text{s.t. } \sum_{e \in \mathcal{E}_f} y'_e = b_f \\ & \qquad\quad y' \in \{0,1\}^{\mathcal{E}_f} \end{aligned} \tag{BSP-b_f}$$

The intuition behind this formulation is that the leader provides the follower with some capacity that he can use according to his preferences. The leader's objective function then consists of the term $-\gamma b_f$, i.e., some cost γ she has to pay for each unit of capacity provided, and, as before, the leader's item values of the items selected by the follower.

An analogous bilevel variant of the Continuous Knapsack Problem, in which the leader determines the capacity for the follower's Continuous Knapsack Problem, and its robust version were studied in [BH22]; see Sections 4.2.2 and 4.4 for more details.

We now show that the problem (BSP-b_f) is indeed equivalent to the Bilevel Selection Problem under Assumption (uni).

Lemma 3.6. *The bilevel optimization problem stated in (BSP-b_f) and the Bilevel Selection Problem under Assumption (uni) can be linearly reduced to each other.*

Proof. In the Bilevel Selection Problem under Assumption (uni), the leader's only relevant decision amounts to the number $b_l \in \{0, \dots, n_l\}$ of items she selects, which contributes a value of γb_l to her objective function and determines the number $b_f = b - b_l$ of items to be selected by the follower. In order to formally reduce the problem to the problem (BSP-b_f), we can reformulate the leader's problem in terms of choosing the number b_f directly and set $b_f^- = \max\{0, b - n_l\}$ and $b_f^+ = \min\{n_f, b\}$. Note that these bounds exactly correspond to the leader's choices to which the follower can react in a feasible way. Then the feasible solutions of the Bilevel Selection Problem and of the problem (BSP-b_f) directly correspond to each other and yield the same leader's objective values in both problems, up to the constant γb.

For the reduction in the opposite direction, define $b = b_f^+$ and $n_l = b - b_f^-$, i.e., introduce a set $\mathcal{E}_l$ of n_l leader's items each having a leader's value of γ, and use the same observation. □

Remark 3.7. Very similarly to Lemma 3.6, one can show that the Bilevel Selection Problem under Assumption (uni), together with any of the assumptions introduced in the subsequent Sections 3.2.3 and 3.2.4, is equivalent to a corresponding modified version of the problem in (BSP-b_f).

Remark 3.8. In the formulation (BSP-b_f), and hence also in the Bilevel Selection Problem with Assumption (uni), we can even assume, without loss of generality, that $\gamma = 0$. Indeed, the cost γ for each unit of capacity can be incorporated into the leader's values of the follower's items. More precisely, we can replace γ by $\gamma' = 0$ and c by c' without changing the problem, where c' is defined by $c'(e) = c(e) - \gamma$ for all $e \in \mathcal{E}_f$.

3.2.3 Assumptions About Binary and Continuous Decisions

In the formulation (BSP2), an obvious modification of the problem is to relax the integrality in the leader's or the follower's decision, i.e., to require only $x \in [0,1]^{\mathcal{E}_l}$ or $y' \in [0,1]^{\mathcal{E}_f}$ such that the leader and/or the follower are allowed to choose fractions of items, as in the Continuous Selection Problem compared to the Selection Problem in Section 3.1. This leads to four combinations of binary/continuous decisions in the leader's/follower's problem that one could investigate:
(bb) $x \in \{0,1\}^{\mathcal{E}_l}$ and $y' \in \{0,1\}^{\mathcal{E}_f}$
(cc) $x \in [0,1]^{\mathcal{E}_l}$ and $y' \in [0,1]^{\mathcal{E}_f}$
(bc) $x \in \{0,1\}^{\mathcal{E}_l}$ and $y' \in [0,1]^{\mathcal{E}_f}$
(cb) $x \in [0,1]^{\mathcal{E}_l}$ and $y' \in \{0,1\}^{\mathcal{E}_f}$
We will see that Assumptions (bc) and (cb) are not very reasonable or interesting to consider further because they are either equivalent to Assumption (bb) or lead to bilevel optimization problems whose optimum value might not be attained; see Lemmas 3.10, 3.11, and 3.12. Therefore, we will focus on the more natural Assumptions (bb) and (cc) for the rest of the chapter then. In the setting without uncertainty, the two are equivalent (see Lemma 3.14), but under uncertainty, this is not true (see, e.g., Remark 3.24). Therefore, we will usually distinguish the problem variants corresponding to Assumptions (bb) and (cc) in Section 3.4.

Remark 3.9. In the setting in which at least one decision maker controls continuous variables, one could also allow the total capacity b to be an arbitrary rational number in $[0, |\mathcal{E}_l \cup \mathcal{E}_f|]$ instead of an integer, like in the definition of the Continuous Selection Problem in Section 3.1. However, for the sake of a simpler exposition, we will only consider integers b throughout this chapter. By dealing with some additional technical details, the algorithms we describe for the Bilevel Selection Problem and the Robust Bilevel Selection Problem under Assumption (cc) can be extended to arbitrary numbers $b \in \mathbb{Q} \cap [0, |\mathcal{E}_l \cup \mathcal{E}_f|]$. However, note that, even if b is an integer, the capacity $b - \sum_{e \in \mathcal{E}_l} x_e$ that remains for the follower after the leader's decision might be non-integral in case the leader's decision is continuous.

3.2.4 Assumptions About the Capacity Constraint

Another possible modification of the problem is to replace the "$= b$" in the follower's constraint by "$\leq b$", i.e., the two possible constraints are:
(=) $\sum_{e \in \mathcal{E}_l} x_e + \sum_{e \in \mathcal{E}_f} y'_e = b$
($\leq$) $\sum_{e \in \mathcal{E}_l} x_e + \sum_{e \in \mathcal{E}_f} y'_e \leq b$

In the setting of Assumption ($\leq$), the follower is not forced to use the provided capacity completely if this would mean to select items having a negative value for him. See Section 3.1 for a discussion of these two variants and their equivalence in the single-level setting. In a similar way, we will prove in Lemma 3.16 that Assumptions (=) and ($\leq$) are equivalent also

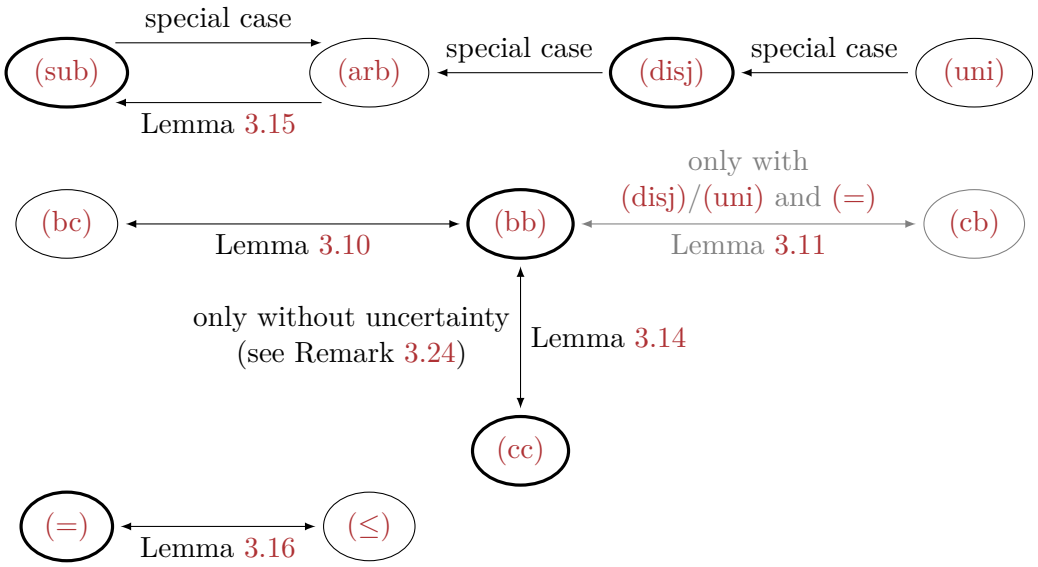


Figure 3.1. Overview of the relations and equivalences between problem variants of the Bilevel Selection Problem. The assumptions marked with a thick ellipse will be used and distinguished in the later sections of this chapter.

in the bilevel setting. Therefore, we will focus on the variant introduced in Section 3.2.1, i.e., Assumption (=), later on. This is also because our arguments often become easier when the number of items in a solution is known in advance.

3.2.5 Equivalence Between Problem Variants

In Sections 3.2.2, 3.2.3, and 3.2.4, we have introduced three "dimensions" of problem variants ($4 \cdot 4 \cdot 2 = 32$ variants in total) of the Bilevel Selection Problem, some of which are equivalent and some of which will turn out to change the problem significantly, in particular when it comes to the Robust Bilevel Selection Problem that we will investigate in Section 3.4. This section will discuss the less interesting of these problem variants and their relations. In particular, we will prove that some of the problem variants are equivalent, even when fixing the remaining "dimensions" in an arbitrary way. This allows us, without loss of generality, to focus on a smaller number of problem variants that will be distinguished in the remainder of the chapter. An overview of the equivalences proved in this section is given in Figure 3.1. Readers that are not interested in the technical details of the equivalences may skip this section and continue with Section 3.3.

First, we investigate Assumptions (bc), (bb), (cc), and (cb) about binary and continuous decisions further. We start with some observations about the follower's problem.

As noted in Section 3.2.1, the follower's problem in the basic problem variant, i.e., under Assumption (bb), corresponds to a single-level Selection Problem with the additional complication of respecting the leader's objective function in the greedy order according to the optimistic or the pessimistic setting. If the follower's decision variables are continuous instead, i.e., under Assumption (bc), the follower's problem becomes a Continuous Selection Problem, in which the optimistic or the pessimistic setting must be respected in the same fashion. The same holds under Assumption (cc) in case of Assumptions (disj)

and (uni), while here the capacity the leader leaves to the follower is not necessarily an integer. In contrast, under Assumptions (arb) and (sub) and Assumption (cc), the situation is slightly more complicated; see the explanations before Lemma 3.14 and in Section 3.4.2.2 for more details.

As mentioned in Section 3.1, the Continuous Selection Problem always has an optimum solution that is binary if b is an integer or has only one fractional value otherwise. Hence, the same also holds for the follower's problem now, in the optimistic as well as in the pessimistic setting. Moreover, if we go from Assumption (=) to Assumption (≤), it clearly remains true that there is always a binary optimum solution if the capacity is an integer and there is at most one fractional variable otherwise, like in the single-level setting in Section 3.1.

These observations directly imply:

Lemma 3.10. *The Bilevel Selection Problem under Assumption (bc) and the Bilevel Selection Problem under Assumption (bb) can be linearly reduced to each other, while preserving any of the Assumptions (arb), (sub), (disj), and (uni) and any of the Assumptions (=) and (≤).*

Proof. When the leader chooses a binary solution, the follower can always be assumed to do so as well, even in the setting of Assumption (bc), by the above considerations about the follower's problem, because the capacity given to him by the leader is an integer. This shows that both problem variants yield the same leader's (and follower's) objective function value for any given leader's solution. □

We now turn to the case of Assumption (cb). Here we have to distinguish between Assumptions (arb), (sub), (disj), and (uni) and also between Assumptions (=) and (≤). In some cases, Assumption (cb) is again equivalent to Assumption (bb):

Lemma 3.11. *The Bilevel Selection Problem under Assumption (cb) and the Bilevel Selection Problem under Assumption (bb) can be linearly reduced to each other, while preserving any of the Assumptions (disj) and (uni) as well as Assumption (=).*

Proof. Given an integer b and the fact that the follower has to make a binary decision in the setting of Assumption (cb), the leader has to use an integer number of items in total as well, because of the equality constraint given by Assumption (=). Like in the binary setting (see Algorithm 3.1), it is clearly optimal, for any fixed splitting of b into integer numbers b_l and b_f of leader's and follower's items, respectively, that each of the two players selects their number of items greedily from their own item set, i.e., each player solves a single-level Selection Problem or Continuous Selection Problem, respectively. Again by the considerations about the Continuous Selection Problem in Section 3.1, this directly implies the existence of an optimum leader's solution that is binary for the problem under Assumption (cb). □

If we are in the setting of Assumption (≤) instead, the above proof does not work since the leader does not have to select an integer number of items in total in order to ensure feasibility. Moreover, the idea of splitting b and applying separate greedy algorithms for both players is based on $\mathcal{E}_l$ and $\mathcal{E}_f$ being disjoint. (However, we will see in Theorem 3.20 later that it can also work under Assumptions (arb) and (sub).) In fact, in any of the cases that are not covered by Lemma 3.11, Assumption (cb) is not equivalent to Assumption (bb) in general. It can even happen that there is no optimum solution for the leader's problem:

Lemma 3.12. *The* Bilevel Selection Problem *under Assumption (cb) does not necessarily admit an optimum solution in case of any of the Assumptions (arb), (sub), and (≤).*

Proof. An example for the case of Assumption (sub) (and hence Assumption (arb)) and Assumption (=) is given as follows: let $\mathcal{E}_l = \{e_1, e_2\}$, $\mathcal{E}_f = \{e_1, e_2, e_3\}$, $b = 2$, $c(e_1) = 0$, $c(e_2) = c(e_3) = 1$, $d(e_1) = 2$, $d(e_2) = d(e_3) = 1$. For the leader, the overall solution consisting of e_2 and e_3 would be best possible, with an objective value of 2. However, a solution without e_1 is not possible because the follower prefers e_1 over e_2 and e_3 and the leader cannot select e_3 herself. But, as the leader is allowed to make continuous decisions and the follower is not, it suffices that the leader selects a small fraction of e_1 in order to "block" it for the follower. However, the leader has to select an integer number of items in total in order to ensure feasibility. This can be achieved by selecting a corresponding fraction of e_2. More precisely, for every $\varepsilon \in (0, 1]$, the leader's solution $x = (\varepsilon, 1 - \varepsilon, 0)$ induces the follower's solution $y = (0, 0, 1)$ and hence a leader's objective value of $2 - \varepsilon$. Thus, it can get arbitrarily close to 2, but cannot attain 2. This implies that the problem does not have an optimum leader's solution.

If we replace Assumption (=) by Assumption (≤), the same example with the same leader's solutions is also valid, as the follower fills the capacity of $b = 2$ also under Assumption (≤) due to his item values being positive.

But also Assumption (≤) in combination with Assumption (uni) (and hence Assumption (disj)) can lead to the same behavior: let $\mathcal{E}_l = \{e_1\}$, $\mathcal{E}_f = \{e_2\}$, $b = 1$, $c(e_1) = c(e_2) = -1$, $d(e_2) = 1$. Here, the leader can achieve an objective value of $-\varepsilon$, for any $\varepsilon \in (0, 1]$, by selecting a fraction of ε of item e_1, but she cannot achieve an objective value of 0 because the follower will select item e_2 as soon as the leader selects nothing. □

Remark 3.13. The issue described in Lemma 3.12 is specific to the setting of Assumption (cb) because it introduces an unnatural discontinuity into the leader's objective. Under Assumptions (bc) and (bb), this cannot happen because the leader's set of feasible solutions is finite. In the setting of Assumption (cc), we will see that the leader's objective can be written as a continuous function such that the issue of Lemma 3.12 is also excluded; see Section 3.3.2 and, similarly, the beginning of Section 3.4.1.2 for the uncertain setting with Assumption (disj).

Because of Lemmas 3.11 and 3.12, we will not investigate Assumption (cb) any further. Because of Lemma 3.10, also Assumption (bc) is not interesting to consider separately. Therefore, we will focus on Assumptions (bb) and (cc) and distinguish between them in the subsequent sections of this chapter.

In the basic setting of the Bilevel Selection Problem without uncertainty, it turns out that also Assumptions (bb) and (cc) are equivalent; see Lemma 3.14. However, this will change in the robust settings, as we will see in Remark 3.24 and Example 3.41. Therefore, it still makes sense to distinguish the two cases then. On the other hand, Lemma 3.10 remains valid also under uncertainty; see Lemma 3.23.

For the following result, we will need some insights about the follower's problem. In the beginning of Section 3.2.5, we have already discussed that the follower's problem corresponds to a single-level Selection Problem or Continuous Selection Problem in most problem variants and what this means for its optimum solutions. However, we have to be more careful in the setting of Assumptions (arb) and (sub) together with

Assumption (cc). In this case, there might be follower's items occurring in his problem that were already selected fractionally by the leader. Therefore, the follower's problem is not a Continuous Selection Problem, but a Continuous Knapsack Problem (see also Section 4.1), i.e., the items may have different sizes, corresponding to the remaining fractions of the items in $\mathcal{E}_l \cap \mathcal{E}_f$ after removing the leader's selection. This will be explained in more detail in Section 3.4.2.2. As the Continuous Knapsack Problem can be solved in a greedy way as well (see Section 4.1), the follower's problem has the same properties here as explained above for the case in which it corresponds to a Continuous Selection Problem. In particular, it always has an optimum solution with at most one fractional variable. Then, after translating the follower's solution back to the original problem, additional fractional variables can occur in the follower's solution. However, this can only happen for items $e \in \mathcal{E}_l \cap \mathcal{E}_f$ which are selected entirely in the overall solution, while one part is selected by the leader and the remaining one by the follower, i.e., $x_e + y_e = 1$. Hence, it remains true that there is at most one item that is selected fractionally by the follower and in the overall solution.

With these observations, we can now prove the equivalence of Assumptions (bb) and (cc).

Lemma 3.14. *The Bilevel Selection Problem under Assumption (bb) and the Bilevel Selection Problem under Assumption (cc) can be linearly reduced to each other, while preserving any of the Assumptions (arb), (sub), (disj), and (uni) and any of the Assumptions (=) and (≤).*

Proof. First consider the setting of Assumptions (disj) and (uni) and Assumption (cc), together with any of the Assumptions (=) and (≤). Then the only relevant interaction between leader and follower is again the number of items selected by the leader, i.e., $b_l := \sum_{e \in \mathcal{E}_l} x_e$; see also the proof of Lemma 3.11. However, b_l does not need to be an integer anymore. We have already seen above that the follower, and in the setting of Assumptions (disj) and (uni) also the leader, can be assumed to make greedy decisions on their own item sets such that there is always an optimum solution in which each of them selects at most one fractional value. If the leader's solution is binary and hence b_l is an integer, then the corresponding follower's solution can be assumed to be binary as well, like in Lemma 3.10. Now suppose that the leader selects some item $e_1 \in \mathcal{E}_l$ fractionally. Then the follower either chooses a binary solution (under Assumption (≤), if he is not interested in filling his capacity completely) or a solution with a fractional item $e_2 \in \mathcal{E}_f$ that satisfies $x_{e_1} + y_{e_2} = 1$. Now consider two ways of rounding the leader's solution to a binary one: either set $x_{e_1} = 0$ or $x_{e_1} = 1$, while not changing the other x-variables. The corresponding follower's solution either remains the same if it was already binary or it changes only the value of y_{e_2}, namely to $y_{e_2} = 1$ or $y_{e_2} = 0$, respectively. In both cases, the original solution is a convex combination of these two solutions, and hence all three solutions have the same objective value if the original solution was optimum.

Now we turn to the setting of Assumptions (arb) and (sub). Consider an optimum leader's solution under Assumption (cc). Without loss of generality, we may assume that there are no items $e \in \mathcal{E}_l \cap \mathcal{E}_f$ each player has selected a fraction of because the leader could increase the value of her variable x_e by the value of the follower's variable y_e, which does not change anything else in the follower's solution, apart from decreasing y_e to 0, and hence does not change the overall solution. Now we can replace $\mathcal{E}_l$ and $\mathcal{E}_f$ by the supports $\mathcal{E}_l'$ and $\mathcal{E}_f'$ of x and y, respectively, and apply the same arguments as in the setting of Assumption (disj) above in order to construct a binary solution having the same leader's objective function value as the original solution.

Hence, even under Assumption (cc), there is always an optimum solution that is also valid for the problem under Assumption (bb). □

Next, we show that Assumptions (arb) and (sub) are equivalent in all cases in which an optimum solution is guaranteed to exist, i.e., only excluding the case of Assumption (cb) (see Lemma 3.12). Therefore, we will usually distinguish only between Assumptions (sub) and (disj) later on.

Lemma 3.15. *The* Bilevel Selection Problem *under Assumption (arb) can be linearly reduced to the* Bilevel Selection Problem *under Assumption (sub), while preserving any of the Assumptions (bc), (bb), and (cc) and any of the Assumptions (=) and (≤).*

Proof. First suppose that we are in the setting of Assumptions (bb) and (=). From an instance satisfying Assumption (arb), one can create an equivalent instance satisfying Assumption (sub) by adding all items in $\mathcal{E}_l \setminus \mathcal{E}_f$ to $\mathcal{E}_f$ and setting $d(e) = -M$ for all these items e, where $M \in \mathbb{Q}_{>0}$ is chosen large enough such that $-M < d(e)$ holds for all items e that were in $\mathcal{E}_f$ in the original instance. In a solution of the new instance, the follower will always prefer his original items over the ones added by the construction. Hence, given any feasible leader's solution in the original instance, it is also a feasible leader's solution in the new instance, resulting in the same follower's solution and the same leader's objective value because the follower has no need to take one of his new items. On the other hand, there can be feasible leader's solutions in the new instance that are infeasible in the original instance. However, the leader can easily obtain a solution in the original instance by adding all follower's new items that he has selected to her solution. The rest of the follower's solution and therefore the overall solution and the leader's objective value remain the same. This proves the lemma when fixing Assumptions (bb) and (=).

In case the leader is allowed to select fractions of items, i.e., under Assumption (cc), the same proof applies because, in the second part, the leader can still add (possibly fractions of) items from the follower's solution to her own solution. But also in case of Assumption (bc), where the leader is not allowed to select fractions of items, but the follower is, this is not a problem because the follower always has an optimum solution that is binary; see Lemma 3.10.

If we are in the case of Assumption (≤) instead of Assumption (=), the follower will never select any of the new items since they have negative value for him and he is not required to fill the capacity. Hence, the sets of feasible leader's solutions in the two instances are even the same now and the second part of the proof above is not needed. More generally, under Assumption (≤), Assumption (sub) is not a restriction as long as negative follower's item values are allowed because such items can always be removed from $\mathcal{E}_f$ without changing the problem. Note that this holds under any of the Assumptions (bc), (bb), and (cc). □

Finally, we prove that Assumptions (=) and (≤) are equivalent. The main ideas here are similar to the ones for the equivalence of the corresponding assumptions in the single-level Selection Problem; see Section 3.1.

Lemma 3.16. *The* Bilevel Selection Problem *under Assumption (=) and the* Bilevel Selection Problem *under Assumption (≤) can be linearly reduced to each other, while preserving any of the Assumptions (arb), (sub), (disj), and (uni) and any of the Assumptions (bc), (bb), and (cc).*

Proof. We first give a reduction from Assumption (≤) to Assumption (=). Consider an instance of the Bilevel Selection Problem under Assumption (≤) and define $\mathcal{E}_f^{\geq 0} = \{e \in \mathcal{E} \mid d(e) > 0 \text{ or } d(e) = 0 \text{ and } c(e) \geq 0\}$ in the optimistic setting, replacing $c(e) \geq 0$ by $c(e) \leq 0$ in the pessimistic setting. This is the set of items that can occur in an optimum follower's solution under Assumption (≤), while the items from $\mathcal{E}_f \setminus \mathcal{E}_f^{\geq 0}$ would only worsen his objective, and he is never required to select any of them in order to satisfy the capacity constraint. We now construct an instance for the Bilevel Selection Problem under Assumption (=) by adding $\max\{0, b - |\mathcal{E}_f^{\geq 0}|\}$ new items e to $\mathcal{E}_f$, for which we set $d(e) = c(e) = 0$. In the follower's greedy order of $\mathcal{E}_f$, the new items now precede any item from $\mathcal{E} \setminus \mathcal{E}_f^{\geq 0}$, ensuring that none of the items in $\mathcal{E} \setminus \mathcal{E}_f^{\geq 0}$ will be selected by the follower, even in the setting of Assumption (=), in which the follower has to fill the capacity completely. Moreover, the dummy items make sure that, also in the setting of Assumption (=), the follower always has a feasible solution consisting of items from $\mathcal{E}_f^{\geq 0}$ and dummy items, no matter what the leader's solution is. Hence, any feasible leader's solution of the initial instance is now also a feasible leader's solution for the new instance, and the other way around, resulting in the same leader's and follower's objective function values.

For the other direction, consider an instance of the Bilevel Selection Problem under Assumption (=). We add some large enough value $M \in \mathbb{Q}_{\geq 0}$ to each follower's item value $d(e)$, for $e \in \mathcal{E}_f$, such that all values $d(e)$ become positive. This will lead to the follower filling the capacity b completely even if we switch to Assumption (≤). However, we have to be careful about the feasibility of leader's solutions now. Every feasible leader's solution of the original instance is still a feasible solution, and the corresponding follower's optimum solution is the same because the greedy order does not change and all items now have positive value for him. Hence, the follower's objective function value only changes by a constant and the leader's objective function value does not change. However, there can be feasible leader's solutions in the new instance that are infeasible in the original instance because the follower does not have enough items left to fill the remaining capacity completely. Note that this can only happen if $|\mathcal{E}_f| < b$. Such leader's solutions cannot be made infeasible in the new instance directly, but we can make them unattractive for the leader by adding $b - |\mathcal{E}_f|$ new items e to $\mathcal{E}_f$, for which we set $d(e) = \varepsilon$ and $c(e) = \alpha$, where $\varepsilon \in \mathbb{Q}_{>0}$ is chosen such that it is smaller than any other follower's item value (including the M), and $\alpha \in \mathbb{Q}$ such that it is smaller than any other leader's item value. This does not change anything when considering leader's solutions that are feasible for both problems because the follower will not select any of the new items then. However, if the leader chooses a solution that is infeasible for the original instance under Assumption (=), then the follower will select at least one of the new items, which makes the corresponding leader's objective value worse than the one of the leader's solution in which she selects an arbitrary additional item from $\mathcal{E}_l \setminus \mathcal{E}_f$ because this replaces the item that has leader's item value α in the overall solution then. Note that there must always be some item in $\mathcal{E}_l \setminus \mathcal{E}_f$ left since we assume $b \leq |\mathcal{E}_l \cup \mathcal{E}_f|$. Hence, a leader's solution which makes the follower select one of the new items is never optimum in the new instance. Therefore, any optimum leader's solution in the new instance is also feasible and optimum in the original instance.

Both constructions clearly preserve Assumptions (arb), (sub), (disj), and (uni) as we only add new items to $\mathcal{E}_f$. Moreover, the arguments are also independent of whether leader and follower make binary or continuous decisions. For the setting of Assumption (cc), note that the construction that replaces a follower's item having leader's item value α by an arbitrary

leader's item works in the same way for replacing a fraction of such a follower's item if required. Hence, both reductions also preserve Assumptions (bc), (bb), and (cc). □

Remark 3.17. For the case of Assumption (=), we have discussed in Remark 3.4 that assuming all leader's and/or follower's item values to be nonnegative leads to an equivalent problem. Under Assumption (≤), the reduction of Remark 3.4, which shifts all item values by a constant, does not work because the number of items in a solution is not fixed. However, (follower's) item values being nonnegative is closely related to the difference between Assumptions (=) and (≤), and, in fact, it now follows from Lemma 3.16 that all such assumptions about the nonnegativity of item values are still equivalent. More precisely, a reduction from the setting of Assumption (≤) and arbitrary rational item values to the one of Assumption (≤) and nonnegative leader's and/or follower's item values is given by first reducing to Assumption (=) as in the proof of Lemma 3.16, then shifting item values as necessary as in Remark 3.4, and finally reducing back to Assumption (≤) as in the proof of Lemma 3.16. Note that the reduction back becomes much easier because infeasibility of leader's solutions due to $|\mathcal{E}_f| < b$ is not possible after the first reduction anyway. Moreover, it clearly preserves also leader's item values being nonnegative.

Summing up, only four problem variants remain that will be interesting to distinguish in the following, namely the ones corresponding to any combinations of Assumptions (sub) and (disj) with Assumptions (bb) and (cc). In the setting without uncertainty, which is investigated next in Section 3.3, even Assumptions (bb) and (cc) are equivalent by Lemma 3.14 such that we will only distinguish Assumptions (sub) and (disj).

3.3 The Bilevel Selection Problem Without Uncertainty

In the previous section, we have defined and discussed several variants of the Bilevel Selection Problem, and we have shown that most of them are equivalent such that we only have to distinguish Assumptions (sub) and (disj) about the relation of the leader's and the follower's item sets here. This section will introduce a polynomial-time algorithm for the Bilevel Selection Problem in Section 3.3.1 that can be applied to all problem variants, using the results from the previous section. In Section 3.3.2, we will then investigate the structure of the leader's objective function in the setting of Assumptions (disj) and (cc) in detail because this gives some insights that will also be useful for the uncertain problem variants later on.

3.3.1 A Polynomial-Time Algorithm

We first focus on the follower's problem. As already noted in Section 3.2.1, the follower solves a single-level Selection Problem, which can be done in linear time by selecting the best follower's items according to his (and possibly the leader's, in case of non-uniqueness) item values. From the follower's perspective, there is no difference regarding Assumptions (sub) and (disj) because the leader's solution X is already fixed and the item set the follower chooses from is given by $\mathcal{E}_f \setminus X$, regardless of the relation between $\mathcal{E}_l$ and $\mathcal{E}_f$.

We now turn to the leader's perspective and distinguish Assumptions (sub) and (disj). In case of Assumption (disj), as already observed in Section 3.2.5, the greedy strategy of Algorithm 3.1 is optimal:

Algorithm 3.1. Algorithm for the Bilevel Selection Problem under Assumptions (bb) and (=) and any of the Assumptions (sub) and (disj)

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $n_l = |\mathcal{E}_l|$, $n_f = |\mathcal{E}_f|$, and $n = |\mathcal{E}_l \cup \mathcal{E}_f|$, $b \in \{0, \dots, n\}$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, $d\colon \mathcal{E}_f \to \mathbb{Q}$

Output : an optimal solution (X, Y) of the Bilevel Selection Problem under Assumptions (bb) and (=)

1 compute a bijection $p_l\colon \{1, \dots, n_l\} \to \mathcal{E}_l$ such that $c(p_l(1)) \geq \dots \geq c(p_l(n_l))$

2 compute a bijection $p_f\colon \{1, \dots, n_f\} \to \mathcal{E}_f$ such that $d(p_f(1)) \geq \dots \geq d(p_f(n_f))$ and $c(p_f(i)) \geq c(p_f(i+1))$ [or $c(p_f(i)) \leq c(p_f(i+1))$] for all $i \in \{1, \dots, n_f - 1\}$ with $d(p_f(i)) = d(p_f(i+1))$ in the optimistic [pessimistic] setting

3 $b_l^- \coloneqq \max\{0, b - n_f\}$

4 $b_l^+ \coloneqq \min\{b, n_l\}$

5 **for** $b_l = b_l^-, \dots, b_l^+$ **do**

6 $\quad X_{b_l} \coloneqq \{p_l(1), \dots, p_l(b_l)\}$

7 $\quad b_f \coloneqq b - b_l$

8 $\quad Y_{b_l} \coloneqq \{p_f(1), \dots, p_f(m)\} \setminus X_{b_l}$, where $m \in \{0, \dots, n_f\}$ is chosen such that $|Y_{b_l}| = b_f$

9 **return** $\operatorname{argmax}\{c(X \cup Y) \mid (X, Y) \in \{(X_{b_l^-}, Y_{b_l^-}), \dots, (X_{b_l^+}, Y_{b_l^+})\}\}$

Theorem 3.18. *Algorithm 3.1 solves the Bilevel Selection Problem under Assumption (disj) in time $\mathcal{O}(n \log n)$.*

Proof. From the follower's perspective, only the number $|X|$ of items selected by the leader is relevant, not the specific leader's solution X, because the item set in the follower's Selection Problem is always $\mathcal{E}_f$ in case of disjoint sets $\mathcal{E}_l$ and $\mathcal{E}_f$, and only the number $b - |X|$ of items to select from $\mathcal{E}_f$ is influenced by the leader. Hence, for any fixed cardinality of the leader's solution, selecting the corresponding number of items greedily from her item set $\mathcal{E}_l$ is optimal for the leader. The algorithm now enumerates all feasible numbers b_l of leader's items to select, together with the corresponding greedy leader's and follower's solutions, respectively. Note that the order of the follower's items determined by p_f correctly represents the optimistic and the pessimistic setting; see also Remark 3.2. Moreover, b_l^- and b_l^+ are defined such that exactly the numbers b_l are enumerated that lead to feasible solutions, given that exactly b items have to be selected in total.

The running time of Algorithm 3.1 is dominated by sorting the leader's and follower's items in Lines 1 and 2, which can each be done in time $\mathcal{O}(n \log n)$. Indeed, the loop in Lines 5 to 9 can be implemented to run in linear time by updating X_{b_l} and Y_{b_l} in every iteration. Note that $\mathcal{E}_l$ and $\mathcal{E}_f$, and hence also X_{b_l} and $\mathcal{E}_f$, are disjoint under Assumption (disj) such that Y_{b_l} is simply chosen as $\{p_f(1), \dots, p_f(b_f)\}$ in Line 8. □

Remark 3.19. As discussed in Section 3.1, a single-level Selection Problem can also be solved in linear time, without the need to sort all items. Hence, a single iteration of the loop in Algorithm 3.1 could be implemented to run in linear time, even without Lines 1 and 2. However, the overall algorithm becomes faster in general if we precompute the orders of the two item sets, as this results in a running time of $\mathcal{O}(n \log n)$ instead of $\mathcal{O}(n^2)$. The reason is that we can use the same orders for all iterations, which correspond to Selection Problems on the same item sets, but with different capacities.

Next, we will show that, also in case of Assumption (sub), the Bilevel Selection Problem can be solved in polynomial time. In fact, the same Algorithm 3.1 solves the leader's problem optimally, although this is less obvious than in case of Assumption (disj):

Theorem 3.20. *Algorithm 3.1 solves the Bilevel Selection Problem under Assumption (sub) in time* $\mathcal{O}(n \log n)$.

Proof. We first argue that the analysis of the running time from the proof of Theorem 3.18 is also valid here. Indeed, it is still true that the loop in Lines 5 to 8 can be implemented to run in linear time by updating X_{b_l} and Y_{b_l} in every iteration. Here, the new follower's solution Y_{b_l} is obtained from Y_{b_l-1} by either removing the item $p_l(b_l)$ that was just added to the leader's solution if $p_l(b_l) \in Y_{b_l-1}$, or otherwise removing the item $p_f(m)$ from the previous iteration, i.e., decreasing m by one.

It remains to show that Algorithm 3.1 is still correct under Assumption (sub). Note that, under Assumption (sub), all sets $X \subseteq \mathcal{E}_l$ with $|X| \leq b$ are feasible leader's solutions because the follower is always able to select the remaining number of items, assuming $b \leq n = n_f$. In particular, we always have $b_l^- = 0$.

Let p_l be the bijection computed in Line 1 of Algorithm 3.1. Given some set $X \subseteq \mathcal{E}_l$, we define

$$i_1(X) \coloneqq \begin{cases} 0 & \text{if } X = \emptyset, \\ \max\{i \in \{1, \dots, n_l\} : p_l(i) \in X\} & \text{otherwise,} \end{cases}$$

$$i_2(X) \coloneqq \begin{cases} n_l + 1 & \text{if } X = \mathcal{E}_l, \\ \min\{i \in \{1, \dots, n_l\} : p_l(i) \notin X\} & \text{otherwise,} \end{cases}$$

and look at the difference $D(X) \coloneqq i_1(X) - i_2(X)$. We first observe that, for all leader's solutions X_{b_l} in Algorithm 3.1, we have $D(X_{b_l}) = b_l - (b_l + 1) = -1$, and for all feasible leader's solutions $X \subseteq \mathcal{E}_l$ that do not have this structure, $D(X) > 0$ holds.

Let X be an optimum leader's solution with minimal value $D(X)$. If $D(X) = -1$, Algorithm 3.1 inspects the solution X and thus returns an optimum solution. Otherwise, we will construct another optimum solution $\overline{X}$ with $D(\overline{X}) < D(X)$, contradicting the assumption that $D(X)$ is minimal. Hence, we will assume $D(X) > 0$ from now on.

Let $e_1 \coloneqq p_l(i_1(X)) \in X$ and $e_2 \coloneqq p_l(i_2(X)) \in \mathcal{E}_l \setminus X$, which is well-defined because $\emptyset \subsetneq X \subsetneq \mathcal{E}_l$ in the case where X is feasible but not considered by the algorithm. As we assume $D(X) > 0$, we have $i_1(X) > i_2(X)$ and hence $c(e_1) \geq c(e_2)$. We consider three operations that can be applied to X:

- removing e_1,
- adding e_2, and
- both removing e_1 and adding e_2.

Note that all three operations reduce the value $D(X)$: Removing e_1 decreases $i_1(X)$ and does not change $i_2(X)$ if $i_1(X) > i_2(X)$ was true before. Analogously, adding e_2 increases $i_2(X)$ and does not change $i_1(X)$. When performing both actions one after the other, $D(X)$ is decreased twice if $D(X) > 0$ still holds after the first one. Otherwise, $D(X) = -1$ already holds after the first action and still after the second one.

We now distinguish four cases and show that, in each of them, one of the three operations leads to an optimum solution again. Let Y denote the optimum follower's choice corresponding to X.

Case 1: $e_2 \in Y$. In this case, $|X| < b$ holds and hence, it is still feasible for the leader to choose $X \cup \{e_2\}$. This leads to the follower choosing $Y \setminus \{e_2\}$ since he can select one item less, while one of the items he selected before is not available anymore, so the selection of the other items is still optimum. Thus, the overall solution $X \cup Y$ does not change and $X \cup \{e_2\}$ is still optimum for the leader.

From now on, we assume $e_2 \notin Y$. For the following cases, consider the optimum follower's solution $\overline{Y}$ corresponding to the leader's choice $\overline{X} = X \setminus \{e_1\}$. The follower chooses the same items as before and one additional one, which we call $\bar{e}$, i.e., $\overline{Y} = Y \cup \{\bar{e}\}$.

Case 2: $\bar{e} = e_1$. The overall solution $X \cup Y$ does not change when the leader removes e_1 from her solution X because the follower will take it instead. Hence, $\overline{X}$ is optimum for the leader.

Case 3: $\bar{e} = e_2$. This means that, in the overall solution $X \cup Y$, the item e_1 is replaced by e_2 when the leader chooses $X \setminus \{e_1\}$ instead of X. As $c(e_1) \geq c(e_2)$, this does not make the solution worse for the leader, so it is still optimum.

Case 4: $\bar{e} \notin \{e_1, e_2\}$. Hence, when the leader chooses $\overline{X}$, neither e_1 nor e_2 occur in the overall solution. If the leader adds e_2 to her solution $\overline{X}$ now, the follower will leave out $\bar{e}$ again, but not change his solution apart from that, i.e., he will choose Y again. Therefore, the leader choosing $X \setminus \{e_1\} \cup \{e_2\}$ leads to an overall solution $X \cup Y \setminus \{e_1\} \cup \{e_2\}$, which is still optimum for the leader due to $c(e_1) \geq c(e_2)$. □

3.3.2 Structure of the Leader's Objective Function Under Assumption (disj)

We have already seen in Lemma 3.14 that the Bilevel Selection Problems (without uncertainty) under Assumptions (bb) and (cc) are equivalent such that we can apply Algorithm 3.1 also to the continuous problem variant. However, we will now give a second proof of this result in the case of Assumption (disj) that is based on investigating the structure of the leader's objective function in more detail. This is useful for a better intuition as well as in view of the upcoming uncertain versions of the Bilevel Selection Problem under Assumptions (disj) and (cc), which we will study in Section 3.4.1.2. Note that the following considerations only hold under Assumption (disj), while the situation of Assumption (sub) is more complicated; see Section 3.4.2.2.

In the setting of Assumptions (disj) and (cc), it is still clear that, for any fixed splitting of the capacity b into capacities b_l and b_f (which are not necessarily integers now) for the leader and the follower, respectively, it is optimal for each of the players to choose their solution greedily, as in the single-level Continuous Selection Problem; see Lemma 3.14. We will now look at the leader's objective function value resulting from any feasible choice of b_l.

First focus on the leader only solving her own Continuous Selection Problem on $\mathcal{E}_l$ with a varying capacity $b_l \in [b_l^-, b_l^+]$, where $b_l^- = \max\{0, b - n_f\}$ and $b_l^+ = \min\{b, n_l\}$ are defined as in Lines 3 and 4 of Algorithm 3.1. Note that, as in the setting of Assumption (bb) above, feasible solutions for the bilevel problem arise from exactly these choices of b_l. The order of the items in $\mathcal{E}_l$ can be assumed to be fixed, independent of b_l, and given by sorting the item values $c(e)$ nonincreasingly. For any b_l, the leader selects a *fractional prefix* (see also Definition 3.45) of this order, i.e., she selects the first $\lfloor b_l \rfloor$ items of the order completely and a fraction $b_l - \lfloor b_l \rfloor$ of the next one; see also Section 3.1. For the total value of the selected items, this means that it changes linearly between any two integer values of b_l, where the slope of this linear segment is given by the current fractional item's value. At an integer b_l, the function proceeds continuously, but the slope changes to the next item's value.

This results in a continuous piecewise linear function with vertices at integer values of b_l and linear segments corresponding to the items. Because of the greedy order of the items, the slopes of the linear segments are sorted nonincreasingly. Hence, the function is concave. An example of such a function is displayed in Figure 3.2a. We denote the function that is constructed in this way by $\mathrm{PLF}(\mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$. The data that are required to define it are the set $\mathcal{E}_l$ of items, the function $c\colon \mathcal{E}_l \to \mathbb{Q}$ of item values that we sort the items by, and the range $[b_l^-, b_l^+]$ of the piecewise linear function, corresponding to the range of capacities that we allow. When writing $\mathrm{PLF}(\mathcal{E}, c', b^-, b^+)$ for a finite set $\mathcal{E}$, a function $c'\colon \mathcal{E} \to \mathbb{Q}$, and numbers $b^-, b^+ \in \mathbb{N}_0$, we always assume that $0 \leq b^- \leq b^+ \leq |\mathcal{E}|$, i.e., that the piecewise linear function is well-defined on the whole range. This function $\mathrm{PLF}(\mathcal{E}, c', b^-, b^+)$ can be computed in $\mathcal{O}(n' \log n')$ time, where $n' = |\mathcal{E}|$, because we need to sort the items $e \in \mathcal{E}$ by their values $c'(e)$, and it can be stored as a list of its vertices, which are given by the function's values in the integers in the range $[b_l^-, b_l^+]$.

For the follower, the setting is analogous: He solves a Continuous Selection Problem on $\mathcal{E}_f$ with varying capacity $b_f \in [b_f^-, b_f^+]$, where $b_f^- = b - b_l^+ = \max\{0, b - n_l\}$ and $b_f^+ = b - b_l^- = \min\{b, n_f\}$. We can assume that the items in $\mathcal{E}_f$ are sorted by their follower's item values $d(e)$ in nonincreasing order (and, in case of non-uniqueness, additionally respecting the leader's item values according to the optimistic or the pessimistic setting; see Remark 3.2). Therefore, his objective function, viewed as a function of b_f, has the same structural properties as the part of the leader's objective function that results from the items selected by the leader, as described above. For an example, see Figure 3.2b. Following the notation introduced above, this function can be denoted by $\mathrm{PLF}(\mathcal{E}_f, (d, \pm c \restriction \mathcal{E}_f), b_f^-, b_f^+)$. By $(d, \pm c \restriction \mathcal{E}_f)$ we mean that the items $e \in \mathcal{E}_f$ are sorted by their values $d(e)$ and, as a secondary criterion, by the values $c(e)$, in a nonincreasing or a nondecreasing order, in the optimistic or the pessimistic setting, respectively.

However, we are mainly interested in the contribution the follower's solution makes to the leader's objective function. This means that we need to take the leader's instead of the follower's item values of the follower's items into account for the leader's objective function, while their order still depends on the follower's item values. This does not change that the function is continuous and piecewise linear with vertices at integer values of b_f, but the slopes of the linear segments now correspond to the leader's item values. Therefore, the function is not concave anymore in general. See Figure 3.2c for the resulting function in the example.

Moreover, in order to combine the two parts of the leader's objective function, we need to use $b_l = b - b_f$ as a parameter instead of b_f, also for the function described in the previous paragraph. This corresponds to mirroring it such that left and right are swapped. In terms of the follower's greedy algorithm, we can imagine that he starts at $b_l = b_l^-$ by selecting his $b_f^+ = b - b_l^-$ best items and then removes them greedily in the reverse order. The transformed function in the example is displayed in Figure 3.2d. We will denote this function by $\overline{\mathrm{PLF}}(\mathcal{E}_f, (d, \pm c \restriction \mathcal{E}_f), c \restriction \mathcal{E}_f, b_l^-, b_l^+, b)$. When writing $\overline{\mathrm{PLF}}(\mathcal{E}, (d', \pm c'), c'', b^-, b^+, b')$, the required data now consist of the finite set $\mathcal{E}$ of items and their values we sort by, in the form $(d', \pm c')$ for functions $d', c'\colon \mathcal{E} \to \mathbb{Q}$ as above, the function $c''\colon \mathcal{E} \to \mathbb{Q}$ of item values that we use to determine the slopes of the linear segments, the range $[b^-, b^+]$ of the piecewise linear function, with $b^-, b^+ \in \mathbb{N}_0$, and the total capacity $b' \in \mathbb{N}_0$ that is required to perform the "swapping" correctly. In order to get a well-defined function, we assume $0 \leq b' - b^+ \leq b' - b^- \leq |\mathcal{E}|$ here. As above, this piecewise linear function can be computed in $\mathcal{O}(n' \log n')$ time, where $n' = |\mathcal{E}|$.

Finally, the leader's objective, as a function of b_l, is given by the sum of the two functions that describe her objective function values resulting from the greedy choices of herself and the follower, respectively. As a sum of two such functions, it is also a continuous piecewise linear function with vertices at integer values of b_l. See Figure 3.2e for the total leader's objective function in the example. Given the two functions as described above, their sum can be computed in $\mathcal{O}(n)$ time.

Optimum leader's solutions now correspond to maxima of this function. Because of its continuous piecewise linear structure, the function always has a maximum at an integer value of b_l. Together with integer optimum solutions of the resulting leader's and follower's Continuous Selection Problems, this yields an optimum solution of the Bilevel Selection Problem under Assumption (cc) that is also valid for Assumption (bb), as in the proof of Lemma 3.14.

Remark 3.21. In the special case of Assumption (uni), the leader's item values of all leader's items are equal. Hence, the part of the leader's objective function that depends only on her own items (see Figure 3.2a) simplifies to being a linear function with slope γ. However, the part of the leader's objective function that depends on the follower's items is not restricted any further than in case of Assumption (disj). Thus, the overall leader's objective function can still be any continuous piecewise linear function with vertices at integer values. Therefore, the case of Assumption (uni) does not seem to be easier to deal with than the case of Assumption (disj) in general.

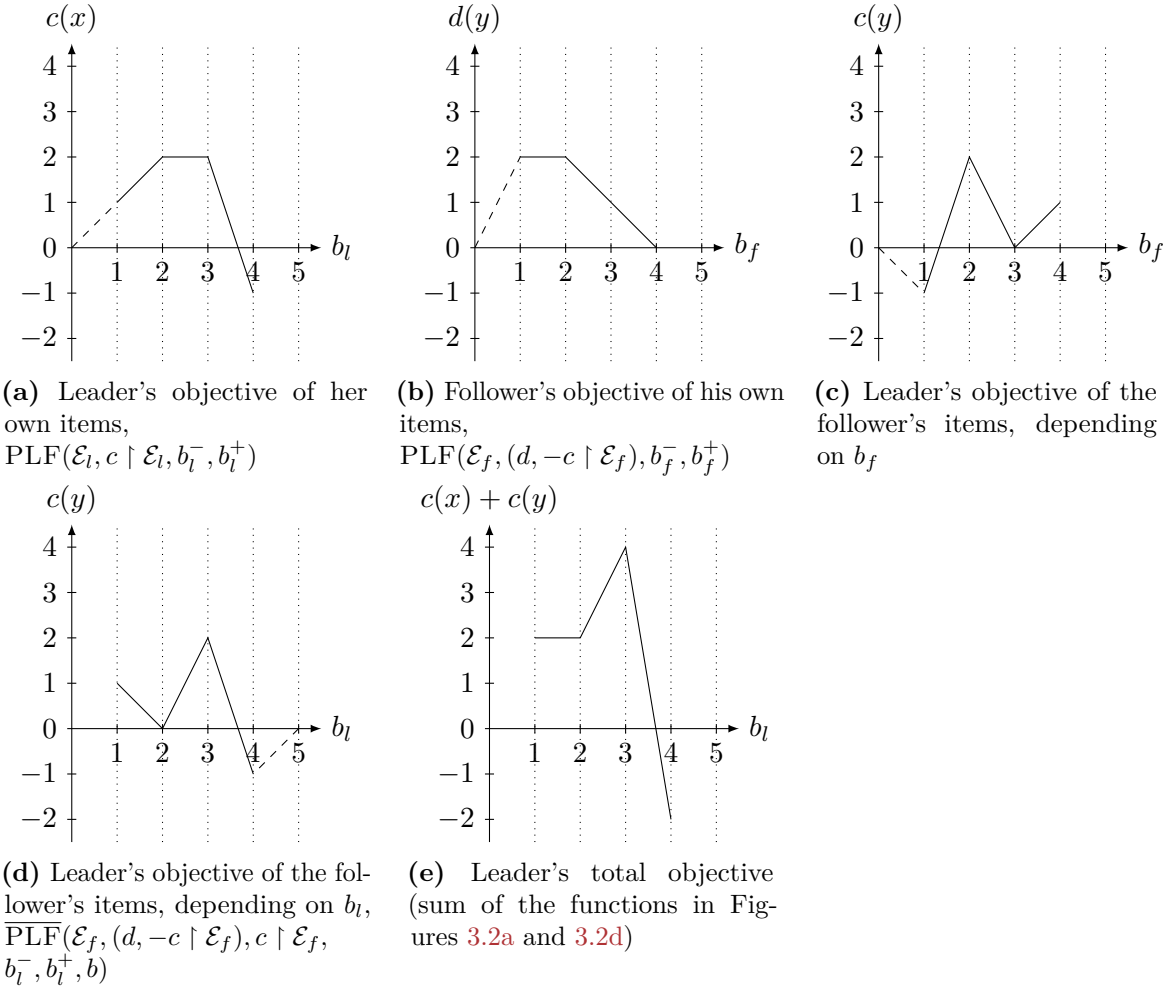
Furthermore, note that the decision of how to split the total capacity b into leader's and follower's capacities b_l and b_f, respectively, is also modeled directly in the alternative formulation (BSP-b_f) of the Bilevel Selection Problem under Assumption (uni) that is discussed in Lemma 3.6.

3.4 The Robust Bilevel Selection Problem

In this section, we start to investigate the Bilevel Selection Problem under uncertainty. More precisely, we assume that the follower's objective function is uncertain from the leader's perspective, and apply the concept of robust optimization, i.e., we assume that an uncertainty set $\mathcal{U} \subseteq \mathbb{R}^{\mathcal{E}_f}$ is given, from which an adversary (of the leader) draws a function $d\colon \mathcal{E}_f \to \mathbb{R}$ of follower's item values such that the result is as bad as possible for the leader. See Sections 2.3 and 2.4.1 for more details on the concept of robust optimization in general and in the context of bilevel optimization problems with uncertain follower's objectives. As an extension of the formulation in (BSP1), the Robust Bilevel Selection Problem can be written as follows:

$$\begin{aligned} \max_{X \subseteq \mathcal{E}_l} \ \min_{d \in \mathcal{U}} \ & c(X \cup Y) \\ \text{s.t. } Y \in \operatorname*{argmax}_{Y'} \ & d(Y') \\ \text{s.t. } & Y' \subseteq \mathcal{E}_f \setminus X \\ & |X \cup Y'| = b \end{aligned} \tag{RBSP}$$

Accordingly, the worst case over $\mathcal{U}$ can also be inserted into the other problem formulations and variants introduced in Section 3.3. Note that all these variants of the Robust Bilevel Selection Problem can be interpreted as special cases of the general problem formulation (R-L) in Section 2.4.1.



(a) Leader's objective of her own items, $\mathrm{PLF}(\mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$

(b) Follower's objective of his own items, $\mathrm{PLF}(\mathcal{E}_f, (d, -c \restriction \mathcal{E}_f), b_f^-, b_f^+)$

(c) Leader's objective of the follower's items, depending on b_f

(d) Leader's objective of the follower's items, depending on b_l, $\overline{\mathrm{PLF}}(\mathcal{E}_f, (d, -c \restriction \mathcal{E}_f), c \restriction \mathcal{E}_f, b_l^-, b_l^+, b)$

(e) Leader's total objective (sum of the functions in Figures 3.2a and 3.2d)

Figure 3.2. Example of leader's and follower's objective functions of the Bilevel Selection Problem under Assumptions (disj), (cc), and (=), depending on the splitting of the total capacity b into leader's and follower's capacities b_l and b_f, respectively. Here, $\mathcal{E}_l = \{e_1, e_2, e_3, e_4\}$, $\mathcal{E}_f = \{e_5, e_6, e_7, e_8\}$, $b = 5$, $c(e_1) = 1$, $c(e_2) = 1$, $c(e_3) = 0$, $c(e_4) = -3$, $c(e_5) = -1$, $c(e_6) = 3$, $c(e_7) = -2$, $c(e_8) = 1$, $d(e_5) = 2$, $d(e_6) = 0$, $d(e_7) = -1$, and $d(e_8) = -1$. We adopt the pessimistic setting, which is reflected in the order of the items e_7 and e_8.
The terms $c(x)$, $c(y)$, and $d(y)$ are used as shortcuts for the objective function values $\sum_{e \in \mathcal{E}_l} c(e)x_e$, $\sum_{e \in \mathcal{E}_f} c(e)y_e$, and $\sum_{e \in \mathcal{E}_f} d(e)y_e$, where $x \in [0,1]^{\mathcal{E}_l}$ and $y \in [0,1]^{\mathcal{E}_f}$ are optimum leader's and follower's solutions for given values of b_l and b_f, respectively.
The dashed linear segments are parts of the functions that correspond to infeasible overall solutions because both leader and follower have to select at least one item in order to reach the total desired capacity of $b = 5$. More formally, we have $b_l^- = b_f^- = 1$ and $b_l^+ = b_f^+ = 4$ here.
See Section 3.3.2 for more details on the structure of the displayed functions.

The situation of the three decision makers can now be imagined as follows: From the follower's perspective, nothing changes compared to the version without uncertainty, as the leader's and the adversary's decisions are already fixed at this point. Hence, the follower still solves a single-level Selection Problem. The adversary chooses an objective function d for the follower from a given uncertainty set $\mathcal{U}$. We observe that, by doing so, he influences the order of the follower's items that the follower will then use for his greedy solution. The adversary's objective is to choose d such that the resulting follower's optimum solution Y has the worst possible value $c(Y)$ for the leader. Finally, the leader's task is to anticipate all of this when choosing her own solution X in the beginning such that the resulting overall set $X \cup Y$ of selected items has the maximum possible value $c(X \cup Y)$ for her.

Remark 3.22. From the leader's and the adversary's perspective, there are only finitely many possible outcomes the adversary can achieve, independently of the type of uncertainty set. Indeed, this is true even if the uncertainty set is not finite because only the greedy order of the follower's items that results from an adversary's decision is relevant for the adversary's and the leader's objective function (see also Remark 3.2), and there are only finitely many (although usually exponentially many) possible orders. We will make use of this observation, in particular, when considering interval uncertainty in Sections 3.4.1.1.2 and 3.4.1.2.2.

The adversary's impact on the leader heavily depends on the type of the uncertainty set $\mathcal{U}$. In fact, uncertainty sets of a specific structure can lead to specific structures of the possible follower's greedy orders and therefore of the worst-case follower's reactions that the leader has to take into account. In the subsequent sections, we will investigate the Robust Bilevel Selection Problem for discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty; see Section 2.3.2 for an introduction to the different types of uncertainty sets.

In addition to different types of uncertainty sets, we can distinguish the same problem variants of the Robust Bilevel Selection Problem as for the Bilevel Selection Problem without uncertainty in Section 3.2. However, most of the equivalences proved in Section 3.2.5 still hold in the robust setting:

Lemma 3.23. *Lemmas 3.10, 3.15, and 3.16 also hold for the Robust Bilevel Selection Problem with any fixed type of uncertainty set.*

Proof. For Lemma 3.10, the assertion is clear because, in every scenario, the follower can still be assumed to choose a binary solution when the leader does so.

In the reductions in the proofs of Lemmas 3.15 and 3.16, new follower's items are added to the instances, and their follower's item values are defined. We will check that the proofs still work if we define appropriate follower's item values for the new items that are independent of the scenario; see also Remark 2.27.

In the proof of Lemma 3.15, the newly defined follower's item values must be smaller than all other follower's item values such that the follower never selects the corresponding items when given a leader's solution that is feasible for the original instance. This still works in the robust setting if we define $M \in \mathbb{Q}_{>0}$ to be large enough such that $-M < d(e)$ holds for all scenarios $d \in \mathcal{U}$ and all items e that were in $\mathcal{E}_f$ in the original instance. The proof of Lemma 3.15 further argues that every feasible leader's solution of the new instance

can be transformed into a feasible leader's solution of the original instance, having the same objective function value, by adding new items that were selected by the follower to the leader's solution instead. Here we have to be careful because, in general, the worst-case scenario might change when the leader changes her solution. However, by construction, the new follower's items are sorted after all original follower's items in his greedy order of every scenario. Hence, when the leader takes over one of these items from the follower, the follower's solution changes in the same way in every scenario, namely by just removing this item. Thus, the adversary's solution and therefore the leader's objective function value can indeed be assumed to be fixed under this specific change of the leader's solution, also in the robust setting.

Next consider the first reduction in the proof of Lemma 3.16. The construction here adds $b - |\mathcal{E}_f^{\geq 0}|$ new items to $\mathcal{E}_f$ and sets their follower's item values to zero. In the robust setting, we can again define the follower's item values to be zero in every scenario. Regarding the number of new items, note that the definition of $\mathcal{E}_f^{\geq 0}$ depends on d and the set can thus be different in different scenarios. However, it is just important that the number of new items is large enough such that the argument works for every scenario. Hence, it suffices to consider the scenario in which $|\mathcal{E}_f^{\geq 0}|$ is minimum when determining the number of new items (or just add b new items).

Finally, turn to the second reduction in the proof of Lemma 3.16. When shifting the follower's item values such that they become positive, it now suffices to choose $M \in \mathbb{Q}_{\geq 0}$ large enough such that all item values in all scenarios become positive. Moreover, the construction adds new items to $\mathcal{E}_f$ whose follower's item values are set such that they are smaller than all other follower's item values. Again, we can now set this value $\varepsilon \in \mathbb{Q}_{>0}$ such that it is smaller than all other follower's item values in all scenarios. Then the leader taking an additional item, starting from a leader's solution that is infeasible for the original instance, changes the corresponding follower's solution in the same manner in every scenario, like in the adapted version of the proof of Lemma 3.15 above. This again implies that the worst-case scenario does not change under this kind of change of the leader's solution, and thus the overall proof is still valid in the robust setting. □

Remark 3.24. Lemma 3.14 cannot be transferred to the robust setting. In contrast to the setting without uncertainty in Section 3.3, the binary and the continuous version (Assumptions (bb) and (cc)) are not equivalent anymore, in the sense that the continuous version does not necessarily have a binary optimum solution, as Example 3.41 will show. We will see that this even holds under Assumption (uni), which is a special case of Assumption (disj). Using Lemmas 3.15 and 3.23, the example can also be transferred to the setting of Assumption (sub). For an intuition, remember that the leader's objective function of the Bilevel Selection Problem without uncertainty under Assumption (cc) is given by a continuous piecewise linear function with vertices at integer values; see Section 3.3.2. This function always attains a maximum in an integer point, corresponding to a binary solution of the Bilevel Selection Problem. In the setting of Assumption (disj), it will turn out that the leader's objective can still be written as a continuous piecewise linear function in the uncertain settings. However, the uncertainty makes the structure of this function more complex such that, in particular, its vertices are not only at integers anymore. This implies that the Robust Bilevel Selection Problem under Assumption (cc) does not necessarily have a binary optimum solution. For more details, see Section 3.4.1.2.

By these considerations, it is reasonable to distinguish four problem variants in the following, namely any combination of Assumptions (sub) and (disj) with Assumptions (bb) and (cc). Some statements hold for several of the problem variants, while we will also see significant differences between them.

We will now start to focus on the easier problem variant of Assumption (disj), before discussing the more complex one of Assumption (sub) in Section 3.4.2.

3.4.1 The Robust Bilevel Selection Problem Under Assumption (disj)

The setting of Assumption (disj), i.e., disjoint leader's and follower's item sets, was already easier to understand than the one of Assumption (sub) in the Bilevel Selection Problem without uncertainty, although the same algorithm was able to solve both variants of the problem; see Section 3.3.1. In particular, it was clear there that the capacity b has to be split between leader and follower somehow and that each of the players solves a single-level Selection Problem or Continuous Selection Problem on their own set of items then. Hence, the algorithm mainly needed to determine (by enumeration) how the leader can split the capacity optimally.

In the robust setting, this basic idea remains valid. The leader can only influence the follower's decision via the capacity b_l she uses, which determines the capacity $b_f = b - b_l$ to be filled by the follower. Therefore, for a fixed choice of b_l, the leader's decision again boils down to a single-level Selection Problem or Continuous Selection Problem that she can solve in a greedy way.

On the follower's side of the problem, now also the adversary comes into play. We do not only solve one single-level Selection Problem or Continuous Selection Problem for the follower, but solve the adversary's problem of determining a follower's objective function that results in the worst possible follower's solution for the leader. A main question in this section is therefore the one of how to solve the adversary's problem given a fixed follower's capacity b_f, which is determined by a fixed leader's solution. We will show that this adversary's problem can be solved in polynomial time for discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty; see Theorems 3.27 and 3.34 and Corollary 3.38 for the case of Assumption (bb) and Theorems 3.43, 3.47, and 3.51 for the case of Assumption (cc).

From the leader's point of view, it remains to determine the best way to split the capacity b into leader's and follower's capacities b_l and b_f, respectively. Under Assumption (bb), this is not more complicated than in the setting without uncertainty because we still know that b_l and b_f must be integers in any feasible solution. Therefore, we can just enumerate these at most $b+1$ options for b_l and b_f. This implies a polynomial-time algorithm for the leader's problem whenever we can solve the adversary's problem in polynomial time; see Theorem 3.25.

In the setting of Assumption (cc), we have to be more careful. Indeed, as mentioned in Remark 3.24, Lemma 3.14 does not hold anymore in the robust setting such that it does not suffice to enumerate integer values b_l in general. This will be discussed in more detail in Section 3.4.1.2. However, there is still a close connection between the leader's and the adversary's problem, although we cannot formulate it as a black-box reduction as in Theorem 3.25 for Assumption (bb). In fact, it will turn out that, also in this setting, the leader's poblem is solvable in polynomial time for discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty; see Theorems 3.44, 3.48, and 3.52. For proving this,

it will be useful to view the leader's objective function as a piecewise linear function again, as in Section 3.3.2, and generalize ideas used for solving the corresponding adversary's problems.

Based on these insights, we will now study the Robust Bilevel Selection Problem under Assumption (disj) in detail for our three specific types of uncertainty sets. In Section 3.4.1.1, we focus on the setting of Assumption (bb), in which, by Theorem 3.25, it suffices to develop algorithms for the adversary's problem. In Section 3.4.1.2, we then develop algorithms for the adversary's and the leader's problem in the setting of Assumption (cc).

3.4.1.1 The Robust Bilevel Selection Problem Under Assumptions (disj) and (bb)

We have already argued above that, in the setting of Assumptions (disj) and (bb), the leader's problem can be solved in polynomial time whenever the adversary's problem can be solved in polynomial time. More formally, we show the following statement:

Theorem 3.25. *Consider the Robust Bilevel Selection Problem under Assumptions (disj) and (bb), and with any type of uncertainty set. Suppose that the corresponding adversary's problem can be solved in time at most $A(I)$, given an instance I of the Robust Bilevel Selection Problem[1] together with any leader's solution. Then the Robust Bilevel Selection Problem, given an instance I with n items, can be solved in time $\mathcal{O}(n \log n + nA(I))$.*

Proof. The leader enumerates all feasible values of b_l that determine the capacity used by herself. These are clearly the integer values in the same range as in Algorithm 3.1, and their number is at most $n+1$. For each of them, the leader solves a single-level Selection Problem on $\mathcal{E}_l$ with capacity b_l in order to determine her solution, and, in addition, solves the adversary's problem for this fixed leader's choice. The former can be done in time $\mathcal{O}(n)$ (see Section 3.1) and the latter in time at most $A(I)$. The best solution computed in this way clearly gives an optimum leader's solution.

For computing all potential leader's solutions, this algorithm requires a running time of $\mathcal{O}(n)$ in each of the $\mathcal{O}(n)$ iterations, i.e., a total running time of $\mathcal{O}(n^2)$. This part of the algorithm's total running time can be reduced to $\mathcal{O}(n \log n)$ by pre-sorting the leader's items once in the beginning of the algorithm, as in Algorithm 3.1; see also Remark 3.19. □

Remark 3.26. The running time $A(I)$ for solving the adversary's problem cannot be expected to be faster than $\mathcal{O}(n)$, which is the time required to solve the follower's problem for a given scenario. Therefore, the second part of the running time $\mathcal{O}(n \log n + nA(I))$ stated in Theorem 3.25 will always dominate the first one, and we could also write the overall running time as $\mathcal{O}(nA(I))$. However, when applying Theorem 3.25 later on, we will not use it as a black box and improve on this running time anyway; see Corollaries 3.28, 3.35, and 3.39.

Because of Theorem 3.25, the aim of this section is to develop polynomial-time algorithms for the adversary's problem of the Robust Bilevel Selection Problem under Assumptions (disj) and (bb), with different types of uncertainty sets. In particular, we will

[1] Note that parts of the structure of the instance I as well as the function A may depend on the type of uncertainty set.

study the settings of discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty, in Sections 3.4.1.1.1, 3.4.1.1.2, and 3.4.1.1.3, respectively.

Note that, although we focus on the setting of Assumption (disj) in the current section, the following algorithmic results about the adversary's problem (Theorems 3.27 and 3.34 and Corollary 3.38) are also stated for the more general setting of Assumption (sub) because there is no difference between the two from the adversary's perspective. Therefore, we can directly apply them later in Section 3.4.2.1 as well.

3.4.1.1.1 Discrete Uncertainty First, we focus on the case of discrete uncertainty, where the uncertainty set $\mathcal{U} \subseteq \mathbb{Q}^{\mathcal{E}_f}$ is a finite set that is explicitly given in the input. In this case, we will show in Theorem 3.27 that the adversary's problem can be solved in polynomial time. Together with Theorem 3.25, this implies that also the leader's problem can be solved in polynomial time; see Corollary 3.28.

Theorem 3.27. *For any fixed feasible leader's solution of the* Robust Bilevel Selection Problem *under any of the Assumptions (sub) and (disj), together with Assumption (bb), and with a discrete uncertainty set $\mathcal{U}$, the adversary's problem can be solved in time $\mathcal{O}(|\mathcal{U}|n)$.*

Proof. We first note that the adversary's problem is independent of Assumptions (sub) and (disj) because a leader's solution $X \subseteq \mathcal{E}_l$ is already fixed and only the remaining follower's item set $\mathcal{E}_f \setminus X$ is relevant. Moreover, let $b_f = b - b_l$ be the capacity that is to be filled by the follower, where $b_l = |X|$ is the capacity the given leader's solution X uses. The adversary's task is to choose follower's item values $d \in \mathcal{U}$ such that the resulting follower's solution is worst possible for the leader. We know that the follower's problem, for fixed item values, is a single-level Selection Problem on $\mathcal{E}_f \setminus X$ with capacity b_f. Such a problem can be solved in linear time; see Sections 3.1, 3.2.1, and 3.2.5. When the adversary enumerates all scenarios in $\mathcal{U}$ and computes the resulting follower's solution, together with its leader's objective value, for each of them, he thus achieves a running time of $\mathcal{O}(|\mathcal{U}|n)$. □

Note that Theorem 3.27 is analogous to the assertion in Theorem 2.39 that, also in more general robust bilevel optimization problems with discrete uncertainty, the adversary's problem is solvable in polynomial time if the follower's problem is solvable in polynomial time, since the adversary can always enumerate all scnearios and solve the corresponding follower's problems.

A polynomial-time leader's algorithm can be directly derived from Theorems 3.27 and 3.25. However, compared to using the two statements as black boxes, its running time can be improved from $\mathcal{O}(|\mathcal{U}|n^2)$ to $\mathcal{O}(|\mathcal{U}|n \log n)$, in a similar way as for the Bilevel Selection Problem without uncertainty; see Remark 3.19.

Corollary 3.28. *The* Robust Bilevel Selection Problem *under Assumptions (disj) and (bb), and with a discrete uncertainty set $\mathcal{U}$, can be solved in time $\mathcal{O}(|\mathcal{U}|n \log n)$.*

Proof. The existence of a polynomial-time algorithm follows from Theorems 3.27 and 3.25. The desired running time of this algorithm can be achieved by implementing it as a variant of Algorithm 3.1: Precompute the bijection p_l and bijections p_f^d for all scenarios $d \in \mathcal{U}$. This takes time $\mathcal{O}(|\mathcal{U}|n \log n)$. The enumeration of values b_l and the computation of the corresponding leader's solutions X_{b_l} is done as in Algorithm 3.1. To determine the

worst-case follower's solution Y_{b_l} in every iteration, we compute $Y_{b_l}^d$ from p_f^d for each scenario $d \in \mathcal{U}$ and choose one with the minimum value $c(Y_{b_l}^d)$. Each of the $\mathcal{O}(n)$ iterations of the loop can thus be implemented to run in time $\mathcal{O}(|\mathcal{U}|)$. □

Note that, while the adversary's problem can be handled in the same way for Assumptions (sub) and (disj) according to Theorem 3.27, the polynomial-time algorithm of Corollary 3.28 only applies to Assumption (disj). In fact, we will see in Theorem 3.58 that Assumption (sub) renders the Robust Bilevel Selection Problem with discrete uncertainty $\mathcal{NP}$-hard.

3.4.1.1.2 Interval Uncertainty In the case of interval uncertainty, i.e., when the uncertainty set has the form $\mathcal{U} = \prod_{e \in \mathcal{E}_f} [d^-(e), d^+(e)]$ with given functions $d^-, d^+ \in \mathbb{Q}^{\mathcal{E}_f}$ that satisfy $d^-(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$, we again develop a polynomial-time algorithm for the adversary's problem, which can then be translated into a polynomial-time algorithm for the leader's problem using Theorem 3.25.

The scenarios arising from an uncertainty set in the Robust Bilevel Selection Problem affect the greedy order of the follower's items the follower then uses to determine his solution. Recall from Remark 3.22 that, for the adversary and the leader, only this order resulting from a scenario is relevant, but not the actual follower's item values. For this reason, we will study which orders can arise from the scenarios in a given uncertainty set. For an interval uncertainty set, we observe that the possible orders correspond to an interval order; see Lemma 3.32. We make this more formal in the following. For more details on interval orders, we refer to, e.g., [Möh89] and the references therein.

Definition 3.29. Let $\mathcal{E}$ be a finite set and $\prec \subseteq \mathcal{E} \times \mathcal{E}$ a binary relation on $\mathcal{E}$, written as $e_1 \prec e_2$ if $(e_1, e_2) \in \prec$ for $e_1, e_2 \in \mathcal{E}$. The relation $\prec$ is called a *(strict) partial order* on $\mathcal{E}$ if it is *antireflexive*, *asymmetric*, and *transitive*, i.e., if it satisfies the following properties for all $e_1, e_2, e_3 \in \mathcal{E}$:

- not $e_1 \prec e_1$
- if $e_1 \prec e_2$, then not $e_2 \prec e_1$
- if $e_1 \prec e_2$ and $e_2 \prec e_3$, then $e_1 \prec e_3$

The pair $(\mathcal{E}, \prec)$ is called a *partially ordered set*.

The *predecessors* of some $e_1 \in \mathcal{E}$, with respect to the partial order, are the items $e_2 \in \mathcal{E}$ with $e_2 \prec e_1$. Two items $e_1, e_2 \in \mathcal{E}$ are *comparable* if we have $e_1 \prec e_2$ or $e_2 \prec e_1$, and *incomparable* otherwise. A *prefix* of a partial order $\prec$ on a set $\mathcal{E}$ is a subset $\mathcal{E}^0 \subseteq \mathcal{E}$ such that, for all $e_1 \in \mathcal{E}^0$ and $e_2 \in \mathcal{E} \setminus \mathcal{E}^0$, either $e_1 \prec e_2$ or e_1 and e_2 are incomparable, i.e., $e_2 \prec e_1$ does not hold.

A *linear order* is a partial order $\prec$ on a set $\mathcal{E}$ in which every two distinct items are comparable. A *linear extension* of a partial order $\prec$ on a set $\mathcal{E}$ is a linear order $\prec_L$ on $\mathcal{E}$ that satisfies $\prec \subseteq \prec_L$.

Observe that, given a partial order $\prec$ on a set $\mathcal{E}$, a subset $\mathcal{E}^0 \subseteq \mathcal{E}$ is a prefix of $\prec$ if and only if there is a linear extension $\prec_L$ of $\prec$ such that $\mathcal{E}^0$ is a prefix of $\prec_L$.

Definition 3.30. A partial order $\prec$ on a finite set $\mathcal{E}$ is called an *interval order* if there is a set $\mathcal{I}$ of closed real intervals, together with a bijection $\mathcal{E} \to \mathcal{I}$ that maps every $e \in \mathcal{E}$ to the interval $[e^-, e^+] \in \mathcal{I}$, such that, for all $e_1, e_2 \in \mathcal{E}$, we have $e_1 \prec e_2$ if and only if $e_1^+ < e_2^-$.

Clearly, every binary relation that is defined via a set of intervals as in Definition 3.30 is a partial order. Note that, while Definition 3.30 uses a representation with closed intervals, interval orders could be equivalently defined in terms of open intervals or a mixture of both; see, e.g., [SST16, Proposition 3].

We will now formally study the relation between interval uncertainty sets and follower's optimum solutions in the Robust Bilevel Selection Problem, which is strongly connected to interval orders. For the sake of a simpler exposition, we assume in the following that there are no one-point intersections between intervals involved in the uncertainty set $\mathcal{U}$. More precisely, we assume that, for all $e_1, e_2 \in \mathcal{E}_f$, we have $d^-(e_1) \neq d^+(e_2)$. However, the case in which one-point intersections may occur can also be handled. It just requires to deal with some technical details related to the optimistic and the pessimistic setting more carefully; see Remark 3.36.

Definition 3.31. Given an interval uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} [d^-(e), d^+(e)]$, we define a binary relation $\prec_{\mathcal{U}}$ on $\mathcal{E}_f$ such that, for $e_1, e_2 \in \mathcal{E}_f$, we have $e_1 \prec_{\mathcal{U}} e_2$ if and only if $d^-(e_1) > d^+(e_2)$.

Lemma 3.32. *The relation $\prec_{\mathcal{U}}$ in Definition 3.31 is an interval order.*

Proof. The relation $\prec_{\mathcal{U}}$ is the inverse of the one defined via the intervals $[d^-(e), d^+(e)]$ as in Definition 3.30. Hence, it is the inverse of an interval order. It is easy to check that this is also an interval order, e.g., by defining it via the intervals $[-d^+(e), -d^-(e)]$ instead. □

Lemma 3.33. *In the Robust Bilevel Selection Problem under Assumption (bb) with an interval uncertainty set $\mathcal{U}$, the set of greedy orders of the follower's items that arise from the scenarios in $\mathcal{U}$ is exactly the set of linear extensions of the interval order $\prec_{\mathcal{U}}$ on $\mathcal{E}_f$. Moreover, the set of follower's optimum solutions, given a fixed feasible leader's solution $X \subseteq \mathcal{E}_l$ and any scenario in $\mathcal{U}$, is exactly the set of prefixes $\mathcal{E}_f^0$ of the restriction of $\prec_{\mathcal{U}}$ to $\mathcal{E}_f \setminus X$ that satisfy $|\mathcal{E}_f^0| = b - |X|$.*

Proof. For fixed leader's and follower's item values c and d, respectively, the follower sorts the items as in Line 2 of Algorithm 3.1, i.e., by decreasing follower's item values and, in case of equal follower's item values, by the leader's item values, depending on whether we are in the optimistic or in the pessimistic setting. If two intervals appearing in $\mathcal{U}$ are disjoint, then the order of the corresponding two follower's items in the greedy order is clearly the same in every scenario. This aligns with the definition of $\prec_{\mathcal{U}}$ and thus implies that every greedy order is indeed a linear extension of $\prec_{\mathcal{U}}$.

On the other hand, every linear extension $\prec_L$ of $\prec_{\mathcal{U}}$ can be achieved as a follower's greedy order by the adversary selecting the follower's item values accordingly: First define $\varepsilon = D/n_f$, where $D = \min_{e_1, e_2 \in \mathcal{E}_f} |d^+(e_1) - d^-(e_2)|$ is the length of the shortest intersection of two intervals (which is positive by our assumption that there are no one-point intersections). We iterate over the items in $\mathcal{E}_f$ in the order given by $\prec_L$. For the first item $e \in \mathcal{E}_f$, we set $d(e) = d^+(e)$. Now consider another iteration that deals with an item $e \in \mathcal{E}_f$, and let $e' \in \mathcal{E}_f$ be the item from the previous iteration. Set $d(e) = d^+(e)$ if $d^+(e) < d(e')$, and $d(e) = d(e') - \varepsilon$ otherwise. The latter is always valid because we have $d(e) < d(e') \leq d^+(e)$ by assumption and $d(e) \geq d^-(e)$ by the definition of ε. This results in an adversary's choice $d \in \mathcal{U}$ that induces a follower's greedy order exactly corresponding to $\prec_L$. Note that this is independent of whether we are in the optimistic or the pessimistic setting because d is constructed such that no two item values are equal.

For every fixed feasible leader's solution $X \subseteq \mathcal{E}_l$ and every $d \in \mathcal{U}$, the follower sorts the items in $\mathcal{E}_f \setminus X$ according to the linear extension of $\prec_{\mathcal{U}}$ corresponding to d, and selects the first $b - |X|$ items as his optimum solution. Since the prefixes of a partial order are exactly the prefixes of the linear extensions of the partial order, this proves also the second part of the assertion. □

Note that Lemma 3.33 is independent of the Assumptions (disj) and (sub), and also the algorithm for the adversary's problem, which is based on it and which we will describe in Algorithm 3.2 and Theorem 3.34, again works for both cases. Moreover, the idea of Lemma 3.33 can also be applied to Assumption (cc); see Lemma 3.46.

Using Lemma 3.33, we could now compute the interval order and its linear extensions explicitly in order to solve the adversary's problem. However, this would not give a polynomial-time algorithm in general because the number of linear extensions can be exponential. Nevertheless, we can solve the adversary's problem in polynomial time, as we show in Theorem 3.34.

For this, we will use that the adversary's problem can be seen as a special case of the *precedence constraint knapsack problem* or *partially ordered knapsack problem*. This is a binary Knapsack Problem (see Section 4.1), where additionally a partial order on the items is given and it is only allowed to pack an item into the knapsack if all its predecessors are also selected; see, e.g., [KPP04, Chapter 13.2]. For the special case where the partial order is an interval order, Woeginger described a pseudopolynomial-time algorithm; see [Woe03, Lemma 11]. There the problem is formulated in a scheduling context and is called *good initial set*. The algorithm is based on the idea that every initial set (i.e., every prefix of the interval order) consists of

- a *head*, i.e., the item whose interval has the rightmost left endpoint among the set,
- all predecessors of the head in the interval order, and
- some subset of the items whose intervals contain the left endpoint of the head in their interior,

assuming that all interval endpoints are pairwise distinct. The algorithm iterates over all items as possible heads, and looks for the optimal subset of the items whose intervals contain the left endpoint of the head in their interior that results in an initial set satisfying the capacity constraint. Since these items are incomparable to each other in the interval order, each subproblem is equivalent to an ordinary Knapsack Problem and can be solved in pseudopolynomial time using dynamic programming (see also Section 4.1).

In our case, the subproblem is a Selection Problem instead of a general Knapsack Problem, i.e., the item sizes are all 1, and it can thus be solved in linear time; see Section 3.1. In our variant as an adversary's problem of the Robust Bilevel Selection Problem, this idea results in Algorithm 3.2. The possible heads $\overline{e} \in \mathcal{E}$ are enumerated in the loop in Lines 7 to 13. The set of the head's predecessors is given by $\mathcal{E}^-_{\overline{e}}$ in Line 8. In Line 11, an appropriate subroutine is called for solving a Selection Problem in linear time. The input of the subroutine consists of a finite item set, a number of items to select and a function of rational item values; see also the definition of the Selection Problem in Section 3.1. Note that the subroutine is called (in Line 11) with the negated item values $-c$, i.e., it actually solves the minimization variant of the Selection Problem with the original item values c. This aligns with the adversary's goal of minimizing the leader's objective function value. Further note that the conditions in Line 10 ensure that $b_f - |\mathcal{E}^-_{\overline{e}}| \in \{0, \dots, |\mathcal{E}^0_{\overline{e}}|\}$ holds. Hence, the instance passed to the subroutine always has a feasible and therefore an optimum solution.

Algorithm 3.2. Algorithm for the adversary's problem of the Robust Bilevel Selection Problem under Assumptions (bb) and (=), and with interval uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$, $b \in \{0, \dots, |\mathcal{E}_l \cup \mathcal{E}_f|\}$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, an interval uncertainty set $\mathcal{U}$ given by $d^-, d^+\colon \mathcal{E}_f \to \mathbb{Q}$ with $d^-(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$, a feasible leader's solution $X \subseteq \mathcal{E}_l$

Output : an optimum adversary's solution $d \in \mathcal{U}$ (i.e., $d\colon \mathcal{E}_f \to \mathbb{Q}$ with $d^-(e) \leq d(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$) of the Robust Bilevel Selection Problem under Assumptions (bb) and (=)

1 $\mathcal{E} \coloneqq \mathcal{E}_f \setminus X$
2 **if** $\mathcal{E} = \emptyset$ **then**
3 $Y_{\overline{e}} \coloneqq \emptyset$
4 **else**
5 $b_f \coloneqq b - |X|$
6 $\overline{\mathcal{E}} \coloneqq \emptyset$
7 **for** $\overline{e} \in \mathcal{E}$ **do**
8 $\mathcal{E}^-_{\overline{e}} \coloneqq \{e \in \mathcal{E} \mid d^-(e) > d^+(\overline{e})\}$
9 $\mathcal{E}^0_{\overline{e}} \coloneqq \{e \in \mathcal{E} \mid d^-(e) \leq d^+(\overline{e}) \leq d^+(e)\}$
10 **if** $|\mathcal{E}^-_{\overline{e}}| \leq b_f$ *and* $|\mathcal{E}^0_{\overline{e}}| \geq b_f - |\mathcal{E}^-_{\overline{e}}|$ **then**
11 $Y^0_{\overline{e}} \coloneqq$ Selection Problem$(\mathcal{E}^0_{\overline{e}}, b_f - |\mathcal{E}^-_{\overline{e}}|, -c \restriction \mathcal{E}^0_{\overline{e}})$
12 $Y_{\overline{e}} \coloneqq \mathcal{E}^-_{\overline{e}} \cup Y^0_{\overline{e}}$
13 $\overline{\mathcal{E}} \coloneqq \overline{\mathcal{E}} \cup \{\overline{e}\}$
14 select $\overline{e} \in \operatorname{argmin}\{c(Y_e) \mid e \in \overline{\mathcal{E}}\}$ arbitrarily
15 **return** $d\colon \mathcal{E}_f \to \mathbb{Q}$ *with* $d(e) \coloneqq d^+(e)$ *for all* $e \in Y_{\overline{e}}$ *and* $d(e) \coloneqq d^-(e)$ *for all* $e \in \mathcal{E}_f \setminus Y_{\overline{e}}$

Theorem 3.34. *For any fixed feasible leader's solution of the Robust Bilevel Selection Problem under any of the Assumptions (sub) and (disj), together with Assumption (bb), and with interval uncertainty, Algorithm 3.2 solves the adversary's problem in time* $\mathcal{O}(n^2)$.

Proof. Note that the result is independent of Assumptions (sub) and (disj) because the relation between $\mathcal{E}_l$ and $\mathcal{E}_f$ does not matter after the leader's solution is fixed.

If the algorithm terminates with $Y_{\overline{e}} = \emptyset$ from Line 3, any choice of $d \in \mathcal{U}$ is optimum for the adversary because the follower has no items to choose from anyway. As we assume that X is a feasible leader's solution, this means that we already have $|X| = b$.

Otherwise, we now assume that $\overline{\mathcal{E}} \neq \emptyset$ holds in the end of the algorithm. We will later show that this is indeed always true. Let $\overline{e}$ and d be defined as in Lines 14 and 15, respectively.

We claim that $Y_{\overline{e}}$ is the optimum follower's solution when the leader's solution X and the adversary's choice d are fixed. The set $\mathcal{E}^-_{\overline{e}}$ defined in Line 8 is the set of predecessors of $\overline{e}$ in the interval order $\prec_{\mathcal{U}}$, while the items in $\mathcal{E}^0_{\overline{e}}$ (Line 9) are incomparable to $\overline{e}$ and to each other because the corresponding intervals all contain the point $d^+(\overline{e})$. Since $Y_{\overline{e}}$ consists of all items in $\mathcal{E}^-_{\overline{e}}$ and a subset of $\mathcal{E}^0_{\overline{e}}$, it is a prefix of $\prec_{\mathcal{U}}$ restricted to $\mathcal{E}$, i.e., for every item in $Y_{\overline{e}}$, also all its predecessors are in $Y_{\overline{e}}$. Moreover, we have $|Y_{\overline{e}}| = b_f = b - |X|$ by construction and by the correctness of the subroutine for the Selection Problem. It

then follows from Lemma 3.33 that the adversary can enforce $Y_{\overline{e}}$ by choosing an appropriate scenario from $\mathcal{U}$. Since we do not care about a precise linear extension, but only about some that has $Y_{\overline{e}}$ as a prefix, this can also be done in an easier way than in the proof of Lemma 3.33, namely by choosing $d \in \mathcal{U}$ as in Line 15 of the algorithm. Indeed, this is a simple way to place the items in $Y_{\overline{e}}$ before the ones in $\mathcal{E} \setminus Y_{\overline{e}}$ in a linear extension of $\prec_{\mathcal{U}}$, by selecting the rightmost possible value $d^+(e)$ for each item $e \in Y_{\overline{e}}$ and the leftmost possible value $d^-(e)$ for each of the remaining items $e \in \mathcal{E} \setminus Y_{\overline{e}}$.

It remains to prove the optimality of the returned solution d. For this, let $Y \subseteq \mathcal{E}$ be an optimum follower's solution corresponding to an optimum adversary's choice. Let $\overline{e} \in Y$ such that $d^+(\overline{e})$ is minimum. Since Y must be a prefix of $\prec_{\mathcal{U}}$ restricted to $\mathcal{E} = \mathcal{E}_f \setminus X$ by Lemma 3.33, all predecessors of $\overline{e}$ are in Y as well, i.e., $\mathcal{E}^-_{\overline{e}} \subsetneq Y$. Moreover, by the choice of $\overline{e}$, we have $Y \setminus \mathcal{E}^-_{\overline{e}} \subseteq \mathcal{E}^0_{\overline{e}}$. Since we also have $|Y \setminus \mathcal{E}^-_{\overline{e}}| = b_f - |\mathcal{E}^-_{\overline{e}}|$ by the feasibility of Y, this set is a feasible solution of the Selection Problem that is solved in Line 11. It must also be an optimum solution because otherwise an optimum solution would lead to a better solution for the adversary. More precisely, every feasible solution of the Selection Problem in Line 11, together with $\mathcal{E}^-_{\overline{e}}$, has cardinality b_f and forms a prefix of $\prec_{\mathcal{U}}$, by the same argument as above. Hence, the adversary can enforce it as an optimum follower's solution by Lemma 3.33. In particular, this holds for the optimum solution of the Selection Problem in Line 11, which leads to the follower's solution of this type that has the minimal value for the leader and is thus best possible for the adversary. Thus, Y is exactly the follower's solution that is enforced in iteration $\overline{e}$ or has the same leader's objective function value as it. This proves the optimality of the returned solution and also shows that $\overline{\mathcal{E}} \neq \emptyset$ indeed holds in the end of the algorithm.

The algorithm performs $|\mathcal{E}| \in \mathcal{O}(n)$ iterations. Each iteration needs linear time to build the sets $\mathcal{E}^-_{\overline{e}}$ and $\mathcal{E}^0_{\overline{e}}$ and to solve the Selection Problem. This gives a total running time of $\mathcal{O}(n^2)$. □

Together with Theorem 3.25, this directly implies that also the leader's problem of the Robust Bilevel Selection Problem with interval uncertainty, under Assumptions (disj) and (bb), can be solved in polynomial time. However, the running time can again be improved, as in Corollary 3.28:

Corollary 3.35. *The Robust Bilevel Selection Problem under Assumptions (disj) and (bb), and with interval uncertainty, can be solved in time* $\mathcal{O}(n^2 \log n)$.

Proof. The existence of a polynomial-time algorithm directly follows from Theorems 3.34 and 3.25. These results imply a running time of $\mathcal{O}(n^3)$. However, this can be improved to $\mathcal{O}(n^2 \log n)$ by presorting not only the leader's, but also the follower's items. More precisely, we compute the sets $\mathcal{E}^-_{\overline{e}}$ and $\mathcal{E}^0_{\overline{e}}$, together with an order of the items in $\mathcal{E}^0_{\overline{e}}$ by their leader's item values c, for all $\overline{e} \in \mathcal{E}_f$, in the beginning of the algorithm, before enumerating the splittings of the capacity b into leader's and follower's capacities b_l and b_f, respectively. This requires a running time of $\mathcal{O}(n^2 \log n)$ and enables to implement every iteration of the loop in Algorithm 3.2 to run in constant time. Thus, the overall running time is $\mathcal{O}(n^2 \log n)$. □

For the sake of completeness, we will finally study the technical complications arising from allowing for one-point intersections between intervals occuring in $\mathcal{U}$.

Remark 3.36. So far in this section, we assumed that the intervals occurring in $\mathcal{U}$ have no one-point intersections. This is convenient because the adversary always has the freedom to set all follower's item values to distinct values and still achieve any linear extension of $\prec_{\mathcal{U}}$; see the proof of Lemma 3.33. In fact, Lemma 3.33 is not true anymore without this assumption. As an extreme example, consider three follower's items $\mathcal{E}_f = \{e_1, e_2, e_3\}$ that all have the same fixed follower's item value, i.e., $d^-(e_1) = d^+(e_1) = d^-(e_2) = d^+(e_2) = d^-(e_3) = d^+(e_3)$. The corresponding interval order $\prec_{\mathcal{U}}$ has no relations then and hence, all six orders of the three items are linear extensions of $\prec_{\mathcal{U}}$. However, the adversary has no choice and the follower's greedy order only depends on the leader's item values c and the optimistic or the pessimistic setting. For instance, if $c(e_1) = 1$, $c(e_2) = 2$, and $c(e_3) = 3$ hold, then the follower's order of the items will be e_3, e_2, e_1 in the optimistic and e_1, e_2, e_3 in the pessimistic setting.

However, in the pessimistic setting, Algorithm 3.2 still works, and Theorem 3.34 and Corollary 3.35 are still true. The reason for this is that the adversary does not have to enforce arbitrary linear extensions of $\prec_{\mathcal{U}}$, but only some of those that have the desired prefix. Moreover, the adversary and the follower have the same interest in case of follower's item values of several items being the same. More precisely, the choice of d in Line 15 of Algorithm 3.2 still makes the follower choose the desired solution $Y_{\overline{e}}$: Assume that $d^+(e_1) = d^-(e_2)$ holds for some $e_1 \in Y_{\overline{e}}$ and $e_2 \in \mathcal{E} \setminus Y_{\overline{e}}$. Then both e_1 and e_2 must lie in $\mathcal{E}^0_{\overline{e}}$, and $c(e_1) \leq c(e_2)$ must hold because e_1 was selected by the Selection Problem subroutine, but e_2 was not. Hence, we may assume that the follower also selects e_1 before e_2 in his greedy order when the adversary chooses d.

The optimistic setting is more complicated because the adversary's and the follower's interests in case of non-uniqueness do not align. However, slightly adjusting the definitions of $\mathcal{E}^-_{\overline{e}}$ and $\mathcal{E}^0_{\overline{e}}$ suffices to make the algorithm work also in the optimistic setting:

1. If $d^-(\overline{e}) < d^+(\overline{e})$, then set

$$\mathcal{E}^-_{\overline{e}} \coloneqq \{e \in \mathcal{E} \mid d^-(e) \geq d^+(\overline{e})\}$$
$$\mathcal{E}^0_{\overline{e}} \coloneqq \{e \in \mathcal{E} \mid d^-(e) < d^+(\overline{e}) \leq d^+(e)\}.$$

   In contrast to Algorithm 3.2, we moved the items $e \in \mathcal{E}$ with $d^-(e) = d^+(\overline{e})$ from $\mathcal{E}^0_{\overline{e}}$ to $\mathcal{E}^-_{\overline{e}}$. In fact, every such item e can be assumed to be selected before $\overline{e}$ in an optimum follower's solution corresponding to an optimum adversary's solution: Either the adversary prefers e preceding $\overline{e}$ (i.e., $c(e) < c(\overline{e})$) and he can also enforce this order then, e.g., by choosing $d(\overline{e}) = d^+(\overline{e}) - \varepsilon$ for an appropriate $\varepsilon \in \mathbb{Q}_{>0}$, or we have $c(e) \geq c(\overline{e})$ and we may assume that the follower selects e before $\overline{e}$, no matter what the adversary does.

2. If $d^-(\overline{e}) = d^+(\overline{e})$, then set

$$\mathcal{E}^-_{\overline{e}} \coloneqq \{e \in \mathcal{E} \mid d^+(e) > d^-(e) \geq d^+(\overline{e}) \text{ or } d^+(e) = d^-(e) > d^+(\overline{e}) \text{ or } [d^+(e) = d^-(e) = d^+(\overline{e}) \text{ and } c(e) > c(\overline{e})]\}$$
$$\mathcal{E}^0_{\overline{e}} \coloneqq \{\overline{e}\} \cup \{e \in \mathcal{E} \mid d^-(e) < d^+(\overline{e}) < d^+(e)\}.$$

   In contrast to the previous case, we now removed the items $e \in \mathcal{E}$ with $d^-(e) = d^+(e) = d^+(\overline{e})$ and $c(e) \leq c(\overline{e})$ from $\mathcal{E}^-_{\overline{e}}$ because these items can be assumed to be selected after $\overline{e}$ by the follower, and the adversary has no influence on them. Moreover, we removed the items $e \in \mathcal{E}$ with $d^-(e) < d^+(e) = d^+(\overline{e})$ from $\mathcal{E}^0_{\overline{e}}$ because these items can be assumed to be selected after $\overline{e}$ by the follower, by a similar argument as in the previous case.

With these ideas, one can show that Algorithm 3.2 with the given modifications solves the adversary's problem correctly in the optimistic setting.

3.4.1.1.3 Discrete Uncorrelated Uncertainty We now turn to the setting of discrete uncorrelated uncertainty, in which there is a finite number of possible follower's item values for each follower's item, independently of each other, i.e., $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$ with finite sets $\mathcal{U}_e \subseteq \mathbb{Q}$ for all $e \in \mathcal{E}_f$.

For every such uncertainty set, its convex hull is an interval uncertainty set and can be written as $\operatorname{conv}(\mathcal{U}) = \prod_{e \in \mathcal{E}_f} [d^-(e), d^+(e)]$, where $d^-(e)$ and $d^+(e)$ are the minimum and maximum value in $\mathcal{U}_e$, respectively, for all $e \in \mathcal{E}_f$. Recall that, in our setting of robust bilevel optimization problems, one cannot replace an uncertainty set by its convex hull in general without changing the problem; see Section 2.4.2.3. However, for the Robust Bilevel Selection Problem under Assumption (bb), we will see in the following that a discrete uncorrelated uncertainty set can still be replaced by its convex hull. We emphasize that this is very specific to the Bilevel Selection Problem under Assumption (bb) and that already the Bilevel Selection Problem under Assumption (cc) does not have this property, as we will see in Example 3.50.

First observe that the adversary's problem with $\operatorname{conv}(\mathcal{U})$ is a relaxation of the one with $\mathcal{U}$. Moreover, the optimum adversary's solutions that are computed by Algorithm 3.2 in the case of interval uncertainty always attain an endpoint of each of the intervals. Hence, when replacing a discrete uncorrelated uncertainty set $\mathcal{U}$ by its convex hull as described above and solve the resulting problem with interval uncertainty using Algorithm 3.2, then only the endpoints of the intervals $[d^-(e), d^+(e)]$ are relevant for the optimum adversary's solutions. As these points are contained in the original sets $\mathcal{U}_e$, this implies that the computed adversary's solutions are also valid, and therefore also optimum, for the original problem with discrete uncorrelated uncertainty. Hence, the two types of uncertainty sets are equivalent for the Robust Bilevel Selection Problem under Assumption (bb), and we have shown:

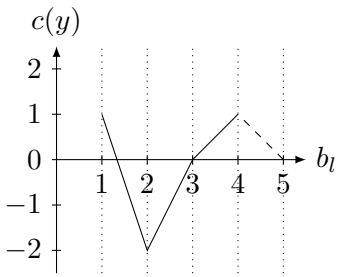
Theorem 3.37. *The Robust Bilevel Selection Problem under any of the Assumptions (sub) and (disj), together with Assumption (bb), and with discrete uncorrelated uncertainty, can be linearly reduced to the corresponding problem with interval uncertainty. The same holds for the adversary's problems, for any fixed feasible leader's solution.*

This implies that Theorem 3.34 and Corollary 3.35 also hold in the setting of discrete uncorrelated uncertainty:

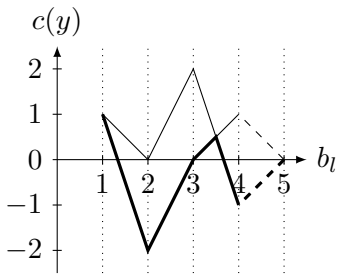
Corollary 3.38. *For any fixed feasible leader's solution of the Robust Bilevel Selection Problem under any of the Assumptions (sub) and (disj), together with Assumption (bb), and with discrete uncorrelated uncertainty, the adversary's problem can be solved in time $\mathcal{O}(n^2)$.*

Corollary 3.39. *The Robust Bilevel Selection Problem under Assumptions (disj) and (bb), and with discrete uncorrelated uncertainty, can be solved in time $\mathcal{O}(n^2 \log n)$.*

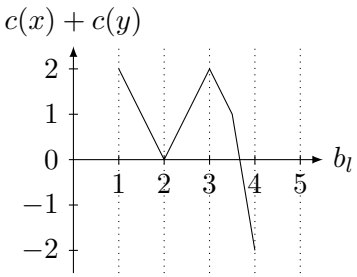
Remark 3.40. The set of greedy orders of the follower's items that the adversary can enforce cannot be described in terms of partial orders here, as it was the case for interval uncertainty; see Lemma 3.33. As an example, let $\mathcal{E}_f = \{e_1, e_2, e_3\}$ with $\mathcal{U}_{e_1} = \{1, 4\}$, $\mathcal{U}_{e_2} = \{2\}$, and $\mathcal{U}_{e_3} = \{3\}$. Then the only relation that is true for every possible follower's greedy order is that e_3 precedes e_2, but the linear extension $e_3 \prec e_1 \prec e_2$ of this partial



(a) Leader's objective of the follower's items, in scenario d_2, $\overline{\text{PLF}}(\mathcal{E}_f, (d_2, -c \restriction \mathcal{E}_f), c \restriction \mathcal{E}_f, b_l^-, b_l^+, b)$



(b) Leader's objective of the follower's items, in the worst case (pointwise minimum of the functions in Figures 3.2d and 3.3a)



(c) Leader's total objective (sum of the functions in Figures 3.2a and 3.3b)

Figure 3.3. Example of the leader's objective function of the Robust Bilevel Selection Problem under Assumptions (disj), (cc), and (=), and with discrete uncertainty, depending on the leader's capacity b_l. Here, $\mathcal{E}_l = \{e_1, e_2, e_3, e_4\}$, $\mathcal{E}_f = \{e_5, e_6, e_7, e_8\}$, $b = 5$, $c(e_1) = 1$, $c(e_2) = 1$, $c(e_3) = 0$, $c(e_4) = -3$, $c(e_5) = -1$, $c(e_6) = 3$, $c(e_7) = -2$, $c(e_8) = 1$, and $\mathcal{U} = \{d_1, d_2\}$ with $d_1(e_5) = 2$, $d_1(e_6) = 0$, $d_1(e_7) = -1$, $d_1(e_8) = -1$, $d_2(e_5) = 1$, $d_2(e_6) = -3$, $d_2(e_7) = 0$, and $d_2(e_8) = 3$. This is the same instance as in Figure 3.2 without uncertainty, but now with an additional second scenario d_2. We again adopt the pessimistic setting, which is reflected in the order of the items e_7 and e_8 in scenario d_1.
See the beginning of Section 3.4.1.2, and also Section 3.3.2 and Figure 3.2, for more details on the structure of the displayed functions.

order cannot be enforced. However, every prefix of $e_3 \prec e_1 \prec e_2$ is also the prefix of one of the two linear orders that the adversary can produce. In fact, the sets of optimum follower's solutions that the adversary can enforce, for any fixed feasible leader's solution, are always the same for $\mathcal{U}$ and $\text{conv}(\mathcal{U})$, which is an intuitive reason why the two types of uncertainty sets are equivalent.

3.4.1.2 The Robust Bilevel Selection Problem Under Assumptions (disj) and (cc)

We now turn to the setting of continuous leader's and follower's decisions, i.e., Assumption (cc). Accordingly, we now use the notation of (BSP2) instead of the one of (BSP1) and (RBSP) that we used in Section 3.4.1.1. That is, we now formulate the leader's and follower's decisions in terms of continuous variables instead of sets.

As already mentioned in Remark 3.24, the setting of Assumption (cc) is not equivalent to the one of Assumption (bb) for the Robust Bilevel Selection Problem anymore, in contrast to the case without uncertainty. We will demonstrate this in Example 3.41.

Before, we need to make some general structural observations about the Robust Bilevel Selection Problem under Assumptions (disj) and (cc). In Section 3.3.2, we have investigated the structure of the leader's objective function of the Bilevel Selection Problem under Assumptions (disj) and (cc), which turned out to be a continuous piecewise linear function in the parameter b_l, the capacity used by the leader (see Figure 3.2e). It was composed as a sum of a function describing the leader's value of the items selected by herself ("PLF"; see Figure 3.2a) and a function describing the leader's value of the items selected by

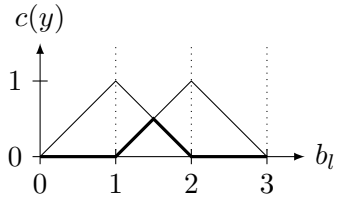


Figure 3.4. The leader's objective function of the Robust Bilevel Selection Problem instance described in Example 3.41

the follower ("$\overline{\text{PLF}}$"; see Figure 3.2d). The former function is not affected by the uncertainty when we turn to the Robust Bilevel Selection Problem, whereas the latter can look different in every scenario. Recall that the linear segments of this function correspond to the selected items, where their slopes are given by the leader's item values. Hence, when the adversary's decision affects the greedy order of the follower's items, this means that the order of the linear segments in the function changes (while the function still remains continuous). See Figure 3.3a for an example of such a function in a different scenario, i.e., only the order of the linear segments is different compared to the function in Figure 3.2d.

For understanding the overall leader's objective function in the robust setting, remember the order in which the three decisions are made: First, the leader determines b_l and her own selection of items, which corresponds to the first function (Figure 3.2a). Second, the adversary chooses a scenario and, with that, one of the other piecewise linear functions (Figures 3.2d and 3.3a in our example). The follower's decision is already implicit when the adversary's decision is interpreted as choosing one of the piecewise linear functions. Since the adversary decides after b_l is fixed, the adversary's and follower's contribution to the leader's objective function is the pointwise minimum of the piecewise linear functions corresponding to the scenarios; see Figure 3.3b. The overall leader's objective function can thus be written as the sum of a piecewise linear function and a pointwise minimum of the other piecewise linear functions; see Figure 3.3c. Again, optimum leader's solutions correspond to maxima of this function.

One can already see in Figures 3.3b and 3.3c that the pointwise minimum coming from the robustness can result in vertices of the piecewise linear functions being not necessarily at integers. We will now show that, for this reason, there might be no optimum solution that is binary, in contrast to the setting without uncertainty. Hence, Assumptions (bb) and (cc) are not equivalent in the robust setting.

Example 3.41. Consider the Robust Bilevel Selection Problem under Assumptions (uni), (cc), and (=), and with discrete uncertainty. Let $\mathcal{E}_l = \{e_1, e_2, e_3\}$, $\mathcal{E}_f = \{e_4, e_5, e_6\}$, $b = 3$, $c(e) = 0$ for all $e \in \mathcal{E}_l$, $c(e_4) = 1$, $c(e_5) = -1$, $c(e_6) = 0$, and $\mathcal{U} = \{d_1, d_2\}$ with $d_1(e_4) = 2$, $d_1(e_5) = 1$, $d_1(e_6) = 0$, $d_2(e_4) = 1$, $d_2(e_5) = 0$, and $d_2(e_6) = 2$. Effectively, the leader can give any capacity $b_f = b - b_l \in [0, 3]$ to the follower by choosing any fraction $b_l \in [0, 3]$ of her items, which have objective function value zero; see also Lemma 3.6. Depending on the follower's objective chosen by the adversary, the order of the follower's items in his greedy algorithm is either e_4, e_5, e_6 or e_6, e_4, e_5. As explained in Section 3.3.2 and above, each of these orders corresponds to a piecewise linear function from the leader's perspective, and the leader's objective function is now the pointwise minimum of these; see Figure 3.4.

One can easily see that the maximum of the leader's objective function is at $b_l = 3/2$, where the two piecewise linear functions corresponding to the two scenarios intersect, i.e., give the same leader's objective value of $1/2$. In contrast, for every binary leader's decision, corresponding to $b_l \in \{0, 1, 2, 3\}$, the adversary can choose $d \in \mathcal{U}$ (namely d_2 for $b_l \in \{0, 1\}$ and d_1 for $b_l \in \{2, 3\}$) such that the leader's objective value is always 0. This shows that the optimum leader's solution in the problem variant under Assumption (cc) is not necessarily binary in case of discrete uncertainty. Hence, Assumptions (bb) and (cc) are indeed not equivalent for the Robust Bilevel Selection Problem.

This observation confirms that it is necessary to examine the case of Assumption (cc) separately, in addition to the one of Assumption (bb). However, we will see that, for the types of uncertainty sets we investigate here, the adversary's and the leader's problems can still be solved in polynomial time in case of Assumptions (disj) and (cc), using similar ideas as in Section 3.4.1.1 for the case of Assumptions (disj) and (bb). We will again discuss discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty, in Sections 3.4.1.2.1, 3.4.1.2.2, and 3.4.1.2.3, respectively. For the setting of discrete uncorrelated uncertainty, we will see that it is not equivalent to interval uncertainty anymore, in contrast to the case of Assumption (bb) (see Theorem 3.37), making it necessary to develop a modified approach.

Remark 3.42. Constructing the leader's objective function as described above, using the pointwise minimum of piecewise linear functions corresponding to the scenarios, directly leads to a polynomial-time solution algorithm for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with discrete uncertainty; see Theorem 3.44. Since, for every type of uncertainty set, even if $\mathcal{U}$ is not finite, there are only finitely many possible follower's greedy orders (see also Remark 3.22), the leader's objective function can always be seen as the pointwise minimum of finitely many piecewise linear functions. If an algorithm is available that enumerates all possible follower's greedy orders that can arise from $\mathcal{U}$, then the problem can be solved using the same approach as for discrete uncertainty. However, the number of possible follower's greedy orders is not polynomial in general, such that the resulting algorithm is usually not efficient. For the cases of interval uncertainty and discrete uncorrelated uncertainty, we will present efficient algorithms in Sections 3.4.1.2.2 and 3.4.1.2.3. These algorithms avoid enumerating all possible follower's greedy orders explicitly, using similar ideas as in Section 3.4.1.1.2.

3.4.1.2.1 Discrete Uncertainty For the case of discrete uncertainty, in which the uncertainty set is a finite set that is explicitly given in the input, we have seen in Theorem 3.27 that the adversary's problem can be solved in polynomial time for the setting of Assumption (bb), by enumerating all scenarios. We now show that the same is true for the setting of Assumption (cc). Note that, in contrast to Theorem 3.27, the following result is only for Assumption (disj), while the case of Assumption (sub) is more complicated here; see Section 3.4.2.2.

Theorem 3.43. *For any fixed feasible leader's solution of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with a discrete uncertainty set $\mathcal{U}$, the adversary's problem can be solved in time $\mathcal{O}(|\mathcal{U}|n)$.*

Proof. Let $b_l = \sum_{e \in \mathcal{E}_l} x_e$ be the capacity that the given leader's solution $x \in [0, 1]^{\mathcal{E}_l}$ uses. Then $b_f = b - b_l$ is the capacity to be filled by the follower. Note that, in contrast to

Theorem 3.27, b_l and b_f are not necessarily integers now. However, by Assumption (disj), it is still true that the only interaction between leader and follower is via the capacities they use. Hence, the follower's task is to solve a single-level Continuous Selection Problem on $\mathcal{E}_f$ with capacity b_f, with respect to the item values chosen by the adversary. As in Theorem 3.27, the adversary can simply enumerate all scenarios in $\mathcal{U}$ and solve the follower's problem for each of them. This can again be done in a running time of $\mathcal{O}(|\mathcal{U}|n)$. □

As a black box, like in Theorem 3.25 and Corollary 3.28 for the setting of Assumption (bb), Theorem 3.43 does not suffice here for constructing an algorithm for the leader's problem because there are infinitely many feasible choices of b_l and Example 3.41 shows that it is indeed not enough to consider only integers b_l. However, the observations about the leader's objective function above directly imply a polynomial-time algorithm in case of a discrete uncertainty set, using piecewise linear functions:

Theorem 3.44. *The Robust Bilevel Selection Problem unter Assumptions (disj) and (cc), and with a discrete uncertainty set $\mathcal{U}$, can be solved in time $\mathcal{O}(|\mathcal{U}|n\log(|\mathcal{U}|n))$.*

Proof. As described in the beginning of Section 3.4.1.2, the leader's objective function can be written as a sum of two continuous piecewise linear functions one of which is a pointwise minimum of $|\mathcal{U}|$ piecewise linear functions. In order to solve the leader's problem, we can compute all these functions explicitly and maximize the resulting continuous piecewise linear function by evaluating it in all its vertices. This leads to an optimum capacity b_l the leader should choose, together with the corresponding leader's greedy solution.

Regarding the running time, first observe that the function corresponding to the items selected by the leader and all functions corresponding to the items selected by the follower in the different scenarios can be computed in total time $\mathcal{O}(|\mathcal{U}|n\log n)$ by sorting the items according to their corresponding values. The pointwise minimum of the $|\mathcal{U}|$ piecewise linear functions with $\mathcal{O}(n)$ vertices each can be computed in time $\mathcal{O}(|\mathcal{U}|n\log(|\mathcal{U}|))$: Since the vertices of the original functions are all at integers, each of the $\mathcal{O}(n)$ sections between two subsequent integer values of b_l can be considered separately. For each of these sections, we need to compute the pointwise minimum of $|\mathcal{U}|$ many linear segments. This can be done in time $\mathcal{O}(|\mathcal{U}|\log(|\mathcal{U}|))$, e.g., using [Her89], and results in a piecewise linear function with $\mathcal{O}(|\mathcal{U}|)$ many linear segments. The total pointwise minimum can therefore be computed in time $\mathcal{O}(|\mathcal{U}|n\log(|\mathcal{U}|))$ and consists of $\mathcal{O}(|\mathcal{U}|n)$ many linear segments. Therefore, all computations, including the final sum of the two piecewise linear functions as well as determining its maximum, can be performed in a total running time of $\mathcal{O}(|\mathcal{U}|n\log(|\mathcal{U}|n))$. □

3.4.1.2.2 Interval Uncertainty In the setting of interval uncertainty, almost the same ideas as in Section 3.4.1.1.2 from the case of Assumption (bb) also apply to the case of Assumption (cc). In particular, the reasoning of Lemma 3.33 is independent of these assumptions, and also the idea of the precedence constraint knapsack problem from [Woe03] is still applicable. However, as we are allowed to select fractions of items now, we have to use an algorithm for the Continuous Selection Problem instead of the Selection Problem as a subroutine. Note that the situation of Assumption (sub) is again more complicated than the one of Assumption (disj), even for the adversary's problem; see Section 3.4.2.2 for more details.

Moreover, a technical complication arises from the fact that, if a fractional solution is selected, it becomes important which item can be the last one of the *initial set* because this will be the item that is selected fractionally by the follower afterwards. In order to formalize this, we introduce the term of a *fractional prefix* in Definition 3.45 and extend Lemma 3.33 to the continuous setting; see Lemma 3.46. Recall Definition 3.29 for the notions of partial orders and prefixes.

Definition 3.45. Let $(\mathcal{E}, \prec)$ be a partially ordered set. A *fractional prefix* of $\prec$ is a triple $(\mathcal{E}^0, e, \lambda)$ where $\mathcal{E}^0 \subseteq \mathcal{E}$ is a prefix of $\prec$ with $\mathcal{E}^0 \neq \emptyset$, $e \in \mathcal{E}^0$ is an item such that $\mathcal{E}^0 \setminus \{e\}$ is also a prefix of $\prec$, and $\lambda \in [0, 1)$. In addition, also the entire set $\mathcal{E}$ is considered to be a fractional prefix of $\prec$. We denote the set of all fractional prefixes of $\prec$ by $\mathcal{F}_\prec$, and we define $f_\prec \colon \mathcal{F}_\prec \to [0,1]^\mathcal{E}$ by $f_\prec(\mathcal{E}^0, e, \lambda) = z$ with $z_{e'} = 1$ for all $e' \in \mathcal{E}^0 \setminus \{e\}$, $z_e = \lambda$, and $z_{e'} = 0$ for all $e' \in \mathcal{E} \setminus \mathcal{E}^0$, for all fractional prefixes $(\mathcal{E}^0, e, \lambda) \in \mathcal{F}_\prec$, and $f_\prec(\mathcal{E}) = z$ with $z_e = 1$ for all $e \in \mathcal{E}$, for the fractional prefix $\mathcal{E} \in \mathcal{F}_\prec$. The *cardinality* of a fractional prefix $F \in \mathcal{F}_\prec$ is defined as $|F| \coloneqq \sum_{e \in \mathcal{E}} (f_\prec(F))_e$. For a fractional prefix $F = (\mathcal{E}^0, e, \lambda)$, we also say that e is the *last item* of F.

While the order of the items in a prefix is usually not fixed or not relevant, the idea of a fractional prefix $(\mathcal{E}^0, e, \lambda)$ is that the final item e of a prefix $\mathcal{E}^0$ is fixed and only a fraction λ of it is considered to be part of the fractional prefix. The function $f_\prec$ maps a fractional prefix to a vector that follows the idea of a characteristic function. Note that the cardinality of the special fractional prefix $\mathcal{E}$ coincides with the usual cardinality of the set $\mathcal{E}$. Otherwise, the cardinality of a fractional prefix $(\mathcal{E}^0, e, \lambda)$ denotes the number of items involved in it, while the last item e is counted only fractionally, according to λ.

Observe that, like in the case of prefixes (see Section 3.4.1.1.2), every fractional prefix of a given partial order $\prec$ on a set $\mathcal{E}$ is also a fractional prefix of some linear extension of $\prec$, and vice versa.

Next, we will apply this notion to the interval order $\prec_\mathcal{U}$ corresponding to the uncertainty set $\mathcal{U}$; see Definition 3.31 and Lemma 3.32. As in Section 3.4.1.1.2, for simplicity, we assume throughout this section that there are no one-point intersections between intervals involved in $\mathcal{U}$, i.e., that, for all $e_1, e_2 \in \mathcal{E}_f$, we have $d^-(e_1) \neq d^+(e_2)$. As explained in Remark 3.36 for the setting of Assumption (bb), one can handle the case with one-point intersections also here by taking care of the technical complications arising from the optimistic or the pessimistic assumption. In particular, as in Section 3.4.1.1.2, the algorithms that are stated in this section are valid for the pessimistic setting, but might need to be adjusted according to Remark 3.36 for the optimistic setting.

We now derive the following statement for the Robust Bilevel Selection Problem under Assumption (cc), in analogy to Lemma 3.33 for the case of Assumption (bb):

Lemma 3.46. *In the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with an interval uncertainty set $\mathcal{U}$, the set of greedy orders of the follower's items that arise from the scenarios in $\mathcal{U}$ is exactly the set of linear extensions of the interval order $\prec_\mathcal{U}$ on $\mathcal{E}_f$. Moreover, the set of follower's optimum solutions, given a fixed feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$ and any scenario in $\mathcal{U}$, is exactly the set $\{f_{\prec_\mathcal{U}}(F) \mid F \in \mathcal{F}_{\prec_\mathcal{U}}, |F| = b - \sum_{e \in \mathcal{E}_l} x_e\}$.*

Proof. The first statement is exactly the same as in Lemma 3.33 and its proof is independent of the different assumptions that we have made here and above. We now turn to the second statement, which is also very similar to Lemma 3.33.

Algorithm 3.3. Algorithm for the adversary's problem of the Robust Bilevel Selection Problem under Assumptions (disj), (cc), and (=), and with interval uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_f \neq \emptyset$ and $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $b \in \{0, \dots, |\mathcal{E}_l \cup \mathcal{E}_f|\}$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, an interval uncertainty set $\mathcal{U}$ given by $d^-, d^+\colon \mathcal{E}_f \to \mathbb{Q}$ with $d^-(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$, a feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$

Output : an optimum adversary's solution $d \in \mathcal{U}$ (i.e., $d\colon \mathcal{E}_f \to \mathbb{R}$ with $d^-(e) \leq d(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$) of the Robust Bilevel Selection Problem under Assumptions (cc) and (=)

1 $b_f \coloneqq b - \sum_{e \in \mathcal{E}_l} x_e$

2 $\overline{\mathcal{E}} \coloneqq \emptyset$

3 **for** $\overline{e} \in \mathcal{E}_f$ **do**

4 $\quad \mathcal{E}_{\overline{e}}^- \coloneqq \{e \in \mathcal{E}_f \mid d^-(e) > d^+(\overline{e})\}$

5 $\quad \mathcal{E}_{\overline{e}}^0 \coloneqq \{e \in \mathcal{E}_f \mid d^-(e) \leq d^+(\overline{e}) \leq d^+(e)\}$

6 $\quad$ **if** $|\mathcal{E}_{\overline{e}}^-| \leq b_f$ *and* $|\mathcal{E}_{\overline{e}}^0| \geq b_f - |\mathcal{E}_{\overline{e}}^-|$ **then**

7 $\quad\quad y^{\overline{e},0} \coloneqq$ Continuous Selection Problem$(\mathcal{E}_{\overline{e}}^0, b_f - |\mathcal{E}_{\overline{e}}^-|, -c \restriction \mathcal{E}_{\overline{e}}^0)$

8 $\quad\quad$ define $y^{\overline{e}} \in [0,1]^{\mathcal{E}_f}$ by $y_e^{\overline{e}} \coloneqq 1$ for all $e \in \mathcal{E}_{\overline{e}}^-$, $y_e^{\overline{e}} \coloneqq y_e^{\overline{e},0}$ for all $e \in \mathcal{E}_{\overline{e}}^0$, and $y_e^{\overline{e}} \coloneqq 0$ for all $e \in \mathcal{E}_f \setminus (\mathcal{E}_{\overline{e}}^- \cup \mathcal{E}_{\overline{e}}^0)$

9 $\quad\quad \overline{\mathcal{E}} \coloneqq \overline{\mathcal{E}} \cup \{\overline{e}\}$

10 select $\overline{e} \in \operatorname{argmin}\{\sum_{e' \in \mathcal{E}_f} c(e') y_{e'}^e \mid e \in \overline{\mathcal{E}}\}$ arbitrarily

11 $\varepsilon \coloneqq \frac{1}{2} \min_{e \in \mathcal{E}_{\overline{e}}^0} (d^+(\overline{e}) - d^-(e))$

12 **return** $d\colon \mathcal{E}_f \to \mathbb{Q}$ *with* $d(e) \coloneqq d^+(e)$ *for all* $e \in \{e' \in \mathcal{E}_f \mid y_{e'}^{\overline{e}} = 1\}$, $d(e) \coloneqq d^-(e)$ *for all* $e \in \{e' \in \mathcal{E}_f \mid y_{e'}^{\overline{e}} = 0\}$, *and* $d(e) \coloneqq d^+(\overline{e}) - \varepsilon$ *for the unique (if it exists)* $e \in \mathcal{E}_f$ *with* $y_e^{\overline{e}} \in (0,1)$

Let $x \in [0,1]^{\mathcal{E}_l}$ be a fixed feasible leader's solution and $d \in \mathcal{U}$ a scenario. The follower then solves a Continuous Selection Problem (see Section 3.1), i.e., he sorts the items in $\mathcal{E}_f$ according to the linear extension $\prec_d$ of $\prec_{\mathcal{U}}$ corresponding to d and selects the first $\lfloor b_f \rfloor$ items and a fraction $b_f - \lfloor b_f \rfloor$ of the next item as his optimum solution, where $b_f = b - \sum_{e \in \mathcal{E}_l} x_e$. This follower's solution is of the form $f_{\prec_d}(F)$ for the unique fractional prefix $F \in \mathcal{F}_{\prec_d}$ with $|F| = b - \sum_{e \in \mathcal{E}_l} x_e$. Since the fractional prefixes of a partial order are exactly the fractional prefixes of the linear extensions of the partial order, this concludes the proof. □

We can now translate the idea of Algorithm 3.2 to the continuous setting and show how to solve the adversary's problem here in Algorithm 3.3 and Theorem 3.47. Observe that, as before, the conditions in Line 6 ensure that the instance passed to the subroutine solving a Continuous Selection Problem in Line 7 has a feasible and therefore an optimum solution; see also Section 3.1. We may assume that the subroutine uses the standard greedy algorithm described in Section 3.1, and, in particular, that it always returns a solution with at most one nonbinary value; see also Remarks 3.1 and 3.2. This assumption ensures that there is indeed at most one $e \in \mathcal{E}_f$ with $y_e^{\overline{e}} \in (0,1)$ in Line 12 of the algorithm.

Theorem 3.47. *For any fixed feasible leader's solution of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with interval uncertainty, Algorithm 3.3 solves the adversary's problem in time* $\mathcal{O}(n^2)$.

Proof. First note that we can make the assumption $\mathcal{E}_f \neq \emptyset$ for the input without loss of generality because the adversary and the follower have nothing to decide otherwise. We first assume that $\overline{\mathcal{E}} \neq \emptyset$ holds in the end of the algorithm, and see that this is indeed true in the end of this proof. Let $\overline{e}$ and d be defined as in Lines 10 and 12, respectively.

We claim that $y^{\overline{e}}$ is the optimum follower's solution when the leader's solution x and the adversary's choice d are fixed. Analogously to Algorithm 3.2 and the proof of Theorem 3.34, the sets $\mathcal{E}_{\overline{e}}^-$ and $\mathcal{E}_{\overline{e}}^0$ defined in Lines 4 and 5, respectively, are the set of predecessors of $\overline{e}$ and an appropriate set of items that are incomparable to $\overline{e}$ and to each other, with respect to the interval order $\prec_{\mathcal{U}}$.

By the definition of $y^{\overline{e}}$ in Line 8 and by the correctness of the subroutine for the Continuous Selection Problem we call in Line 7, we may assume that $y^{\overline{e}}$ corresponds (in the sense of $f_{\prec_{\mathcal{U}}}$; see Definition 3.45) to a fractional prefix of $\prec_{\mathcal{U}}$ of cardinality $\sum_{e \in \mathcal{E}_f} y_e^{\overline{e}} = b_f$. By Lemma 3.46, the adversary can enforce the follower's solution $y^{\overline{e}}$ by choosing an appropriate scenario from $\mathcal{U}$. Again, this can be done in an easier way than the one described in Lemma 3.33 because the adversary does not need to enforce a fixed linear extension, but only some linear extension having the given fractional prefix. In particular, this is achieved by the choice $d \in \mathcal{U}$ that is made in Line 12 because it ensures that both $\{e \in \mathcal{E}_f \mid y_e^{\overline{e}} = 1\}$ and $\{e \in \mathcal{E}_f \mid y_e^{\overline{e}} > 0\}$ are prefixes of the corresponding linear extension of $\prec_{\mathcal{U}}$.

We now prove that the returned adversary's solution d is optimum. First observe that, in case $b_f \in \{0, |\mathcal{E}_f|\}$, any adversary's solution is optimum because there is only a single feasible follower's solution, namely $y_e = 0$ for all $e \in \mathcal{E}_f$ or $y_e = 1$ for all $e \in \mathcal{E}_f$, respectively. For the rest of the proof, we assume that we are in none of these cases. Let $y \in [0,1]^{\mathcal{E}_f}$ be an optimum follower's solution corresponding to an optimum adversary's choice and let $\overline{e} \in \{e \in \mathcal{E}_f \mid y_e > 0\}$ such that $d^+(\overline{e})$ is minimum. By Lemma 3.46, y is the image of a fractional prefix $(\mathcal{E}_y, e_y, \lambda_y)$ of $\prec_{\mathcal{U}}$ under $f_{\prec_{\mathcal{U}}}$. By the properties of a fractional prefix, we have $\mathcal{E}_{\overline{e}}^- \subseteq \mathcal{E}_y$, and by the choice of $\overline{e}$, we have $\mathcal{E}_y \setminus \mathcal{E}_{\overline{e}}^- \subseteq \mathcal{E}_{\overline{e}}^0$. As y is also a feasible follower's solution, $|(\mathcal{E}_y, e_y, \lambda_y)| = b_f$, and hence, $f_{\prec_{\mathcal{U}} | \mathcal{E}_{\overline{e}}^0 \times \mathcal{E}_{\overline{e}}^0}(\mathcal{E}_y \setminus \mathcal{E}_{\overline{e}}^-, e_y, \lambda_y)$ is a feasible solution for the Continuous Selection Problem in Line 7. It must be optimum as well because a better adversary's solution could be constructed from it otherwise, analogously to the proof of Theorem 3.34. Thus, y is the follower's solution that is enforced in iteration $\overline{e}$ or has the same leader's objective function value as it. This proves the optimality of the returned solution and also shows that $\overline{\mathcal{E}} \neq \emptyset$ is satisfied in the end of the algorithm.

Analogously to Theorem 3.34, the running time is given by $|\mathcal{E}_f| \in \mathcal{O}(n)$ iterations each of which can be implemented to run in linear time. □

Observe that, if b_f is an integer, which is, in particular, the case if the leader has taken a binary decision, then Algorithm 3.3 coincides with Algorithm 3.2 for the setting of Assumption (bb). Indeed, the subroutine solving the Continuous Selection Problem is only called for integer capacities then, and hence solves a binary Selection Problem; see also Section 3.1.

In contrast to Corollary 3.35, we cannot directly derive an algorithm for the leader's problem from Algorithm 3.3 and Theorem 3.47 here because it does not suffice to enumerate integer values of b_l under Assumption (cc); see also Example 3.50. However, the adversary's algorithm can be generalized to an algorithm for the leader by combining it with the idea of representing the leader's objective function by piecewise linear functions; see Section 3.3.2 and the beginning of Section 3.4.1.2. In fact, one could enumerate all linear extensions of the interval order $\prec_{\mathcal{U}}$, determine the corresponding piecewise linear functions, and

Algorithm 3.4. Algorithm for the leader's problem of the Robust Bilevel Selection Problem under Assumptions (disj), (cc), and (=), and with interval uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_f \neq \emptyset$ and $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $b \in \{0, \dots, |\mathcal{E}_l \cup \mathcal{E}_f|\}$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, an interval uncertainty set $\mathcal{U}$ given by $d^-, d^+\colon \mathcal{E}_f \to \mathbb{Q}$ with $d^-(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$

Output : an optimum leader's solution $x \in [0,1]^{\mathcal{E}_l}$ of the Robust Bilevel Selection Problem under Assumptions (cc) and (=)

1 $b_l^- := \max\{0, b - |\mathcal{E}_f|\}$, $b_f^+ := b - b_l^-$
2 $b_l^+ := \min\{b, |\mathcal{E}_l|\}$, $b_f^- := b - b_l^+$
3 compute $g_l := \mathrm{PLF}(\mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$
4 $\overline{\mathcal{E}} := \emptyset$
5 **for** $\bar{e} \in \mathcal{E}_f$ **do**
6 $\mathcal{E}_{\bar{e}}^- := \{e \in \mathcal{E}_f \mid d^-(e) > d^+(\bar{e})\}$
7 $\mathcal{E}_{\bar{e}}^0 := \{e \in \mathcal{E}_f \mid d^-(e) \leq d^+(\bar{e}) \leq d^+(e)\}$
8 **if** $|\mathcal{E}_{\bar{e}}^-| \leq b_f^+$ *and* $|\mathcal{E}_{\bar{e}}^0| \geq b_f^- - |\mathcal{E}_{\bar{e}}^-|$ **then**
9 $\overline{b_f^-} := \max\{0, b_f^- - |\mathcal{E}_{\bar{e}}^-|\}$
10 $\overline{b_f^+} := \min\{b_f^+ - |\mathcal{E}_{\bar{e}}^-|, |\mathcal{E}_{\bar{e}}^0|\}$
11 compute $g_f^{\bar{e}} := \overline{\mathrm{PLF}}(\mathcal{E}_{\bar{e}}^0, -c \restriction \mathcal{E}_{\bar{e}}^0, c \restriction \mathcal{E}_{\bar{e}}^0, b - \overline{b_f^+}, b - \overline{b_f^-}, b)$
12 compute $g_f^{\bar{e}} := g_f^{\bar{e}} + (-|\mathcal{E}_{\bar{e}}^-|, c(\mathcal{E}_{\bar{e}}^-))$ (i.e., shift the function $g_f^{\bar{e}}$ by $-|\mathcal{E}_{\bar{e}}^-|$ in x direction and by $c(\mathcal{E}_{\bar{e}}^-)$ in y direction)
13 $\overline{\mathcal{E}} := \overline{\mathcal{E}} \cup \{\bar{e}\}$
14 compute $g_f := \min\{g_f^{\bar{e}} \mid \bar{e} \in \overline{\mathcal{E}}\}$ (i.e., the pointwise minimum)
15 compute $g := g_l + g_f$
16 compute some $b_l^* \in \operatorname{argmax}\{g(b_l) \mid b_l \in [b_l^-, b_l^+]\}$
17 **return** Continuous Selection Problem$(\mathcal{E}_l, b_l^*, c \restriction \mathcal{E}_l)$

apply the same procedure as in the case of discrete uncertainty to combine them; see Section 3.4.1.2.1. However, this would not give a polynomial running time in general, as the number of linear extensions can be exponential; see also Remark 3.42. But combining this approach with the ideas of Algorithm 3.3 provides a strategy that only looks at a small number of linear segments that might contribute to the pointwise minimum. This results in Algorithm 3.4. Recall the definitions of PLF and $\overline{\mathrm{PLF}}$ from Section 3.3.2 (with Figure 3.2); see also the beginning of Section 3.4.1.2 (with Figure 3.3).

Theorem 3.48. *Algorithm 3.4 solves the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with interval uncertainty, in time* $\mathcal{O}(n^2 \log n)$.

Proof. Again, we can make the assumption $\mathcal{E}_f \neq \emptyset$ for the input without loss of generality: Otherwise, assuming a feasible instance, we have $b_l^- = b_l^+ = b$, so the returned value Continuous Selection Problem$(\mathcal{E}_l, b, c \restriction \mathcal{E}_l)$ is clearly the leader's optimum solution.

First note that the function g_l computed in Line 3 is well-defined because $0 \leq b_l^- \leq b_l^+ \leq |\mathcal{E}_l|$, and defined on the range $[b_l^-, b_l^+]$. This is the range of possible capacities b_l that the leader can use in a feasible solution. The function g_l describes the part of the leader's

objective function that represents the optimum value she can achieve by a selection of her own items when using a capacity of b_l. We refer to Section 3.3.2 and the beginning of Section 3.4.1.2 for more details. For the optimum value $b_l^* \in [b_l^-, b_l^+]$, an optimum leader's solution is given by Continuous Selection Problem$(\mathcal{E}_l, b_l^*, c \restriction \mathcal{E}_l)$, as returned by the algorithm in Line 17. It remains to show that the value b_l^* derived by the algorithm is indeed optimum.

We claim that, for every fixed value $b_l \in [b_l^-, b_l^+]$, the function g computed in Line 15 correctly describes the best leader's objective function value that is possible if the leader uses a capacity of b_l and the follower uses a capacity of $b_f = b - b_l$. By the considerations about g_l above, this is equivalent to the function g_f defined in Line 14 correctly describing the best objective function values the adversary can achieve for fixed values of b_l. Hence, we claim that, for every fixed value $b_l \in [b_l^-, b_l^+]$, the value of the adversary's solution returned by Algorithm 3.3, with any given feasible leader's solution x using capacity b_l, is precisely $g_f(b_l)$. Recall that, under Assumption (disj), from the adversary's and the follower's point of view, only the capacity b_l occupied by the leader's solution is relevant, but not the precise leader's solution x. The correctness of Algorithm 3.4 then follows from the correctness of Algorithm 3.3, which we proved in Theorem 3.47.

Let $b_l \in [b_l^-, b_l^+]$ be a fixed leader's capacity and $b_f = b - b_l \in [b_f^-, b_f^+]$ the corresponding follower's capacity. Fix some item $\overline{e} \in \mathcal{E}_f$ and consider the corresponding iterations of the loop in Line 3 of Algorithm 3.3 and of the loop in Line 5 of Algorithm 3.4. The sets $\mathcal{E}_{\overline{e}}^-$ and $\mathcal{E}_{\overline{e}}^0$ are defined in the same way in both algorithms. It can be checked that the conditions in Line 6 of Algorithm 3.3 are satisfied for b_f if and only if the conditions in Line 8 of Algorithm 3.4 are satisfied and the function $g_f^{\overline{e}}$ is defined in $b - b_f$, in the end of the iteration. For the latter, note that the range of the function $g_f^{\overline{e}}$ according to Lines 11 and 12, using the definitions of $\overline{b_f^-}$ and $\overline{b_f^+}$ in Lines 9 and 10, is

$$\left[b - \overline{b_f^+} - |\mathcal{E}_{\overline{e}}^-|, b - \overline{b_f^-} - |\mathcal{E}_{\overline{e}}^-|\right] = \left[\max\{b - b_f^+, b - |\mathcal{E}_{\overline{e}}^0| - |\mathcal{E}_{\overline{e}}^-|\}, \min\{b - |\mathcal{E}_{\overline{e}}^-|, b - b_f^-\}\right],$$

and that this range contains $b - b_f$ for some given $b_f \in [b_f^-, b_f^+]$ if and only if $b_f \in [|\mathcal{E}_{\overline{e}}^-|, |\mathcal{E}_{\overline{e}}^0| + |\mathcal{E}_{\overline{e}}^-|]$ is satisfied, which is exactly the condition in Line 6 of Algorithm 3.3. Hence, either in both or in none of the two algorithms, the case of b_f and $\overline{e}$ contributes to the computations after the loop. We now demonstrate that, if the above conditions are satisfied, then the values contributed by this case are also the same in both algorithms. The corresponding follower's solution $y^{\overline{e}}$ defined in Line 8 of Algorithm 3.3 is given by selecting all items in $\mathcal{E}_{\overline{e}}^-$ and a part of the items in $\mathcal{E}_{\overline{e}}^0$ according to the Continuous Selection Problem with the values $-c$ and the remaining capacity $b_f - |\mathcal{E}_{\overline{e}}^-|$ (see Line 7). The total value $\sum_{e \in \mathcal{E}_f} c(e) y_e^{\overline{e}}$ of this solution is then considered in the minimum in Line 10. The function $g_f^{\overline{e}}$ in Algorithm 3.4 describes exactly the same: The piecewise linear function defined in Line 11 describes the total value c of the follower's solutions that are selections from $\mathcal{E}_{\overline{e}}^0$ according to the values $-c$, depending on the capacity $b_l = b - b_f$ that the leader uses (see also Section 3.3.2), and the shift in Line 12 adjusts this function corresponding to the additional selection of all items in $\mathcal{E}_{\overline{e}}^-$. Therefore, the value $g_f^{\overline{e}}(b_l) = g_f^{\overline{e}}(b - b_f)$ considered in the pointwise minimum in Line 14 is exactly the same as the value of the solution described above that contributes to the minimum in Line 10 of Algorithm 3.3. This implies that, for every $b_l \in [b_l^-, b_l^+]$, the value $g_f(b_l)$ is exactly the leader's value of an optimum follower's solution corresponding to an optimum adversary's decision, as returned by Algorithm 3.3. This concludes the proof of the correctness of Algorithm 3.4.

For the running time, observe that each of the piecewise linear functions g_l (Line 3) and $g_f^{\bar{e}}$ (Line 11) can be computed in time $\mathcal{O}(n \log n)$; see Section 3.3.2. As in the proof of Theorem 3.44, the pointwise minimum of the at most n piecewise linear functions $g_f^{\bar{e}}$ (Line 14), each consisting of at most n linear segments, can be computed in time $\mathcal{O}(n^2 \log n)$, using [Her89]. Again as in the proof of Theorem 3.44, the sum with the piecewise linear function g_l with at most n linear segments and the maximum of the resulting piecewise linear function (Lines 15 and 16) can be computed in time $\mathcal{O}(n^2 \log n)$. □

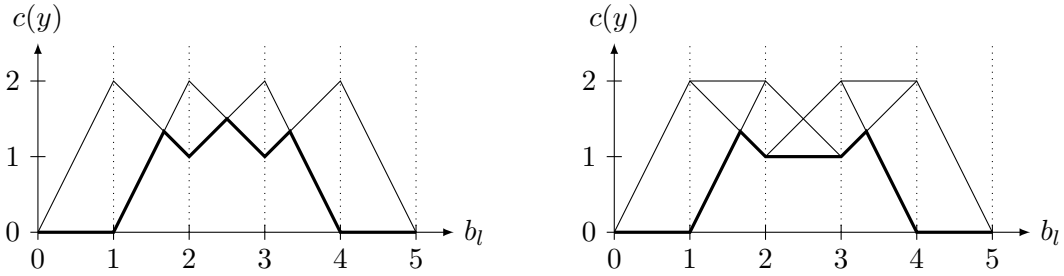
Remark 3.49. In the proof of Theorem 3.48, we have shown that the function g computed by Algorithm 3.4 is the function of optimum leader's objective values depending on the parameter b_l, which is the capacity the leader uses, for all feasible values of b_l. Therefore, it is the same function as the one described in the beginning of Section 3.4.1.2.

As noted above and in Remark 3.42, it is not clear for general uncertainty sets that the function can be computed in polynomial time because it was defined using the pointwise minimum of an exponential number of piecewise linear functions. In general, it might even be possible that it consists of an exponential number of linear segments. The polynomial-time computation of this function in Algorithm 3.4 now also shows that the function indeed consists of only polynomially many linear segments in case of interval uncertainty.

On the other hand, from the construction in Algorithm 3.4, it is not obvious that the function g is continuous because the functions $g_f^{\bar{e}}$ are defined on different ranges such that the pointwise minimum might lead to jumps. However, given the original definition of the function in the beginning of Section 3.4.1.2, it is clearly continuous.

3.4.1.2.3 Discrete Uncorrelated Uncertainty In the setting of Assumption (bb), we have shown that discrete uncorrelated uncertainty is equivalent to interval uncertainty, and we could thus use the same algorithms as for interval uncertainty also in this case; see Theorem 3.37. In particular, the optimum adversary's solutions computed by Algorithm 3.2 for interval uncertainty were always attained at an endpoint of each interval. The latter is not true anymore in Algorithm 3.3, the variant of this algorithm that solves the adversary's problem in case of Assumption (cc). In fact, the algorithm sets all values $d(e)$ to endpoints of the respective intervals, except for the one that corresponds to the unique item the follower is supposed to select only a fraction of. The special adversary's choice of this value is necessary because this item has to be guaranteed to be the last one the follower selects, i.e., the last item in the corresponding fractional prefix; see Definition 3.45 and Lemma 3.46. We show in Example 3.50 that it can in fact be necessary for the adversary to select one value to be in the middle of its interval in order to achieve his optimum value. Hence, the solution returned by Algorithm 3.3 cannot be assumed to be feasible for the adversary's problem in case of discrete uncorrelated uncertainty in general. Therefore, we cannot solve the Robust Bilevel Selection Problem with discrete uncorrelated uncertainty by simply replacing the uncertainty set by the interval uncertainty set that is its convex hull, as in Theorem 3.37.

Example 3.50. Consider the Robust Bilevel Selection Problem under Assumptions (uni), (cc), and (=). We compare the settings of interval uncertainty and discrete uncorrelated uncertainty for the following instance. Let $\mathcal{E}_l = \{e_1, e_2, e_3, e_4, e_5\}$, $\mathcal{E}_f = \{e_6, e_7, e_8, e_9, e_{10}\}$, $b = 5$, $c(e) = 0$ for all $e \in \mathcal{E}_l$, $c(e_6) = 2$, $c(e_7) = -1$, $c(e_8) = 1$, $c(e_9) = -2$, $c(e_{10}) = 0$, and $\mathcal{U} = \{5\} \times \{4\} \times \{3\} \times \{2\} \times \{1, 6\}$. (We write this informally



(a) Discrete uncorrelated uncertainty set $\mathcal{U}$ **(b)** Interval uncertainty set $\text{conv}(\mathcal{U})$

Figure 3.5. The leader's objective function of the Robust Bilevel Selection Problem instances described in Example 3.50

instead of $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$ with $\mathcal{U}_{e_6} = \{5\}$, $\mathcal{U}_{e_7} = \{4\}$, etc.) Then the leader can effectively give any capacity $b_f = b - b_l \in [0, 5]$ to the follower by selecting any fraction $b_l \in [0, 5]$ of her items; see also Lemma 3.6 and Example 3.41.

Given the uncertainty set $\mathcal{U}$, the adversary can choose between two scenarios that result in the greedy orders $e_6, e_7, e_8, e_9, e_{10}$ and $e_{10}, e_6, e_7, e_8, e_9$ for the follower, respectively. As explained in the beginning of Section 3.4.1.2, this results in a leader's objective function that is the pointwise minimum of the corresponding two piecewise linear functions, which are shown in Figure 3.5a. The leader's optimum solution here is to select $b_l = 5/2$ of her own items, resulting in an objective function value of $3/2$ in either of the two scenarios.

On the other hand, consider the convex hull of the discrete uncorrelated uncertainty set $\mathcal{U}$ above, which is the interval uncertainty set $\text{conv}(\mathcal{U}) = \{5\} \times \{4\} \times \{3\} \times \{2\} \times [1, 6]$. This uncertainty set gives the adversary the choice between the five follower's greedy orders in which e_{10} is placed anywhere between the other items in their fixed order e_6, e_7, e_8, e_9. This results in the leader's objective function depicted in Figure 3.5b and in leader's optimum solutions corresponding to $b_l = 5/3$ and $b_l = 10/3$ with an objective function value of $4/3$.

This example first shows again that Assumptions (bb) and (cc) are not equivalent for the Robust Bilevel Selection Problem, in contrast to the problem without uncertainty, because the optimum solutions are not attained at binary solutions. Example 3.41 has already demonstrated this for the case of a discrete uncertainty set, and we now have seen that this is the case also for interval uncertainty and discrete uncorrelated uncertainty.

Moreover, the example shows that interval uncertainty and discrete uncorrelated uncertainty are not equivalent for the Robust Bilevel Selection Problem under Assumption (cc), in contrast to the setting of Assumption (bb); see Section 3.4.1.1.3. In particular, consider a leader's solution corresponding to $b_l = 5/2$, which is optimum for the leader in case of the discrete uncorrelated uncertainty set $\mathcal{U}$. From the adversary's point of view, both scenarios in $\mathcal{U}$ are equally good. However, with $\text{conv}(\mathcal{U})$, he can achieve a better solution for himself, i.e., a worse objective function value for the leader, by choosing $d(e_{10}) \in [3, 4]$, which is not possible in $\mathcal{U}$. This shows that an optimum adversary's solution with respect to interval uncertainty might need to contain a value that does not lie at an endpoint of its respective interval.

Although Example 3.50 shows that we cannot use Algorithms 3.3 and 3.4 for the setting of discrete uncorrelated uncertainty, one can still solve the Robust Bilevel Selection Problem and its adversary's problem in polynomial time, using similar ideas as for

interval uncertainty. In order to demonstrate this, we first study the adversary's problem. Assume that a feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$ is given. As before, this defines the capacity $b_f = b - \sum_{e \in \mathcal{E}_l} x_e$ the follower has to fill.

First note that, in case b_f is an integer, the follower can be assumed to always make a binary decision because he solves a Continuous Selection Problem with an integer capacity; see Section 3.1. Therefore, we can handle the adversary's problem like in case of Assumption (bb), i.e., apply Algorithm 3.2; see Theorem 3.37 and Corollary 3.38. For the general case where b_f is not an integer, we develop a modified version of this algorithm and consider the follower's solution as being composed of a binary selection of $\lfloor b_f \rfloor$ items and an additional item of which the follower selects a fraction of $b_f - \lfloor b_f \rfloor$.

Recall that, in Algorithms 3.2, 3.3, and 3.4, we iterate over all possible *heads* $\overline{e}$ of the (fractional) prefix corresponding to the follower's solution and use the right endpoint $d^+(\overline{e})$ of its interval as a reference point for defining the item sets $\mathcal{E}^-_{\overline{e}}$ and $\mathcal{E}^0_{\overline{e}}$ from which the adversary builds a worst-case follower's solution for the current iteration. Algorithm 3.3 assumes that any item in $\mathcal{E}^0_{\overline{e}}$ can be made the fractional item by selecting the follower's item value $d^+(\overline{e}) - \varepsilon$ for it. On a related note, for a number of intervals that have a proper intersection, the adversary can enforce any order of the corresponding items. When we turn to discrete uncorrelated uncertainty, the adversary's choice is more limited and fewer follower's greedy orders are possible; see also Remark 3.40. Regarding the possible fractional items, this is a significant limitation. Therefore, we have to take a closer look at the specific follower's item values the adversary can select for the fractional item.

Instead of the head, which is not necessarily the fractional item in Algorithm 3.3, we now guess the item $e^* \in \mathcal{E}_f$ that the follower selects only a fraction of, together with the follower's item value $\delta^* \in \mathcal{U}_{e^*}$ the adversary assigns to it. Note that there are $\sum_{e \in \mathcal{E}_f} |\mathcal{U}_e|$ such guesses to consider in total. The remaining adversary's problem is then similar to the problem solved in an iteration of Algorithm 3.2, i.e., it has to be decided which $\lfloor b_f \rfloor$ items the follower selects before the fractional item. This can again be solved as a binary Selection Problem, while we now use δ^* as a reference point in the definition of the item sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$. Among all possible choices of (e^*, δ^*), the adversary selects one that yields the worst leader's value of the resulting follower's solution. This strategy leads to Algorithm 3.5.

Note that, for the sake of a simpler presentation, some technical details that might be relevant for the optimistic problem version are neglected in Algorithm 3.5 and in the following; see Remark 3.55 for details.

Similarly to the proofs of Theorems 3.34 and 3.47, we can now show:

Theorem 3.51. *For any fixed feasible leader's solution of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with a discrete uncorrelated uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$, Algorithm 3.5 solves the adversary's problem in time $\mathcal{O}(nu)$, where $u = \sum_{e \in \mathcal{E}_f} |\mathcal{U}_e|$.*

Proof. If the algorithm terminates from Line 3, any choice of $d \in \mathcal{U}$ is optimum for the adversary because the follower has only one trivial feasible solution anyway, namely either selecting no items or selecting all items from $\mathcal{E}_f$.

We first assume that $\overline{\mathcal{E}} \neq \emptyset$ holds in the end of the algorithm, and see that this is indeed true in the end of this proof. Let e^*, δ^*, and d be defined as in Lines 15 and 16, respectively.

We claim that the follower's solution that selects the items in Y_{e^*,δ^*} and a fraction of $b_f - \lfloor b_f \rfloor$ of the item e^* is optimum for the follower when the leader's solution x

Algorithm 3.5. Algorithm for the adversary's problem of the Robust Bilevel Selection Problem under Assumptions (disj), (cc), and (=), and with discrete uncorrelated uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $b \in \{0, \dots, |\mathcal{E}_l \cup \mathcal{E}_f|\}$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, a discrete uncorrelated uncertainty set $\mathcal{U}$ given by a finite set $\mathcal{U}_e \subseteq \mathbb{Q}$ for every $e \in \mathcal{E}_f$, a feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$

Output : an optimum adversary's solution $d \in \mathcal{U}$ (i.e., a value $d(e) \in \mathcal{U}_e$ for every $e \in \mathcal{E}_f$) of the Robust Bilevel Selection Problem under Assumptions (cc) and (=)

1 $b_f \coloneqq b - \sum_{e \in \mathcal{E}_l} x_e$
2 **if** $b_f = 0$ *or* $b_f = |\mathcal{E}_f|$ **then**
3 **return** *an arbitrary* $d \in \mathcal{U}$
4 **for** $e \in \mathcal{E}_f$ **do**
5 $d^-(e) \coloneqq \min \mathcal{U}_e$, $d^+(e) \coloneqq \max \mathcal{U}_e$
6 $\overline{\mathcal{E}} \coloneqq \emptyset$
7 **for** $e^* \in \mathcal{E}_f$ **do**
8 **for** $\delta^* \in \mathcal{U}_{e^*}$ **do**
9 $\mathcal{E}^-_{e^*,\delta^*} \coloneqq \{e \in \mathcal{E}_f \mid d^-(e) > \delta^*\}$
10 $\mathcal{E}^0_{e^*,\delta^*} \coloneqq \{e \in \mathcal{E}_f \mid d^-(e) \le \delta^* \le d^+(e)\} \setminus \{e^*\}$
11 **if** $|\mathcal{E}^-_{e^*,\delta^*}| \le \lfloor b_f \rfloor$ *and* $|\mathcal{E}^0_{e^*,\delta^*}| \ge \lfloor b_f \rfloor - |\mathcal{E}^-_{e^*,\delta^*}|$ **then**
12 $Y^0_{e^*,\delta^*} \coloneqq$ Selection Problem$(\mathcal{E}^0_{e^*,\delta^*}, \lfloor b_f \rfloor - |\mathcal{E}^-_{e^*,\delta^*}|, -c \restriction \mathcal{E}^0_{e^*,\delta^*})$
13 $Y_{e^*,\delta^*} \coloneqq \mathcal{E}^-_{e^*,\delta^*} \cup Y^0_{e^*,\delta^*}$
14 $\overline{\mathcal{E}} \coloneqq \overline{\mathcal{E}} \cup \{(e^*, \delta^*)\}$
15 select $(e^*, \delta^*) \in \operatorname{argmin}\{c(Y_{e,\delta}) + c(e) \cdot (b_f - \lfloor b_f \rfloor) \mid (e,\delta) \in \overline{\mathcal{E}}\}$ arbitrarily
16 **return** $d\colon \mathcal{E}_f \to \mathbb{Q}$ *with* $d(e^*) \coloneqq \delta^*$, $d(e) \coloneqq d^+(e)$ *for all* $e \in Y_{e^*,\delta^*}$ *and* $d(e) \coloneqq d^-(e)$ *for all* $e \in \mathcal{E}_f \setminus (Y_{e^*,\delta^*} \cup \{e^*\})$

and the adversary's choice d are fixed. Denote this follower's solution by the vector $y^{e^*,\delta^*} \in [0,1]^{\mathcal{E}_f}$, i.e., $y_e^{e^*,\delta^*} = 1$ for all $e \in Y_{e^*,\delta^*}$, $y_{e^*}^{e^*,\delta^*} = b_f - \lfloor b_f \rfloor$, and $y_e^{e^*,\delta^*} = 0$ for all $e \in \mathcal{E}_f \setminus (Y_{e^*,\delta^*} \cup \{e^*\})$. The follower's solution y^{e^*,δ^*} is clearly feasible, as we have $\sum_{e \in \mathcal{E}_f} y_e^{e^*,\delta^*} = b_f$, using the correctness of the subroutine for the Selection Problem we call in Line 12. Similarly to Algorithms 3.2 and 3.3 and the proofs of Theorems 3.34 and 3.47, the sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$ defined in Lines 9 and 10, respectively, are the set of items that the follower always sorts before item e^* in his greedy order, given any $d' \in \mathcal{U}$ with $d'(e^*) = \delta^*$, and the set of items that the follower might sort before or after item e^* in such a greedy order, depending on the adversary's choice of their follower's item values. Accordingly, given the item values d returned by the algorithm, y^{e^*,δ^*} is optimum for the follower, since $d(e) \ge \delta^* = d(e^*)$ for all $e \in \mathcal{E}_f$ with $y_e^{e^*,\delta^*} = 1$ and $d(e) \le \delta^* = d(e^*)$ for all $e \in \mathcal{E}_f$ with $y_e^{e^*,\delta^*} = 0$.

We now prove that the returned adversary's solution d is optimum. Let $y \in [0,1]^{\mathcal{E}_f}$ be an optimum follower's solution corresponding to an optimum adversary's choice $\overline{d}$ and let $\overline{e}$ be the fractional item, or, if the solution y is binary, let $\overline{e} \in \{e \in \mathcal{E}_f \mid y_e = 0\}$ such that $\overline{d}(e)$ is maximum. This is valid because $b_f < |\mathcal{E}_f|$ holds after Line 3 and because we may

assume that y has been selected greedily according to an order given by $\overline{d}$ and therefore has at most one fractional item; see Remarks 3.1 and 3.2. Let $\overline{\delta} = \overline{d}(\overline{e})$. The follower's greedy choice implies that $\mathcal{E}^-_{\overline{e},\overline{\delta}} \subseteq \{e \in \mathcal{E}_f \mid y_e = 1\}$ and $\{e \in \mathcal{E}_f \mid y_e = 1\} \setminus \mathcal{E}^-_{\overline{e},\overline{\delta}} \subseteq \mathcal{E}^0_{\overline{e},\overline{\delta}}$. As y is a feasible follower's solution, the set $\{e \in \mathcal{E}_f \mid y_e = 1\}$ has cardinality $\lfloor b_f \rfloor$. Hence, $\{e \in \mathcal{E}_f \mid y_e = 1\} \setminus \mathcal{E}^-_{\overline{e},\overline{\delta}}$ is a feasible solution for the Selection Problem in Line 12. It must be optimum as well because a better adversary's solution could be constructed from an optimum solution of the Selection Problem otherwise, analogously to the proof of Theorem 3.34. Thus, y is a follower's solution that is enforced in iteration $(\overline{e}, \overline{\delta})$ or has the same leader's objective function value as it. This proves the optimality of the returned solution and also shows that $\overline{\mathcal{E}} \neq \emptyset$ is satisfied in the end of the algorithm.

The running time is given by $u = \sum_{e \in \mathcal{E}_f} |\mathcal{U}_e|$ iterations each of which can be implemented to run in time $\mathcal{O}(n)$. □

Observe that, if b_f is an integer (and we are not in one of the trivial cases $b_f = 0$ and $b_f = |\mathcal{E}_f|$), then Algorithm 3.5 selects a fraction of $b_f - \lfloor b_f \rfloor = 0$ of the item e^*. Hence, the algorithm examines binary prefixes, very similarly to Algorithm 3.2, the only difference being that more possible choices of the reference point δ^* are considered here, which does not change the result in this case.

In order to devise an algorithm for the leader's problem, in case of interval uncertainty, we combined the approach used for solving the adversary's problem with the idea to represent the leader's objective function by piecewise linear functions; see Algorithm 3.4. Something similar can be done here in order to generalize Algorithm 3.5 to an algorithm that solves the leader's problem. This results in Algorithm 3.6.

The computation of the function g_l (Line 6), representing the leader's costs of the items she selects herself, and the final steps to determine an optimum solution (Lines 22 to 24) are as in Algorithm 3.4; they are independent of the type of uncertainty set. The function g_f, representing the leader's costs of the items selected by the follower, is now computed by a generalization of Algorithm 3.5. Indeed, for every fixed follower's capacity $b_f \in [b_f^-, b_f^+]$, the value $g_f^{b_f^*}(b_f)$, for $b_f^* = \lfloor b_f \rfloor$ (or $b_f^* = b_f - 1$ in case $b_f = b_f^+$), and therefore the value $g_f(b_f)$, corresponds to the worst leader's value of a follower's selection that the adversary can achieve. The approach to determine this value is like in Algorithm 3.5: All possible fractional items $e^* \in \mathcal{E}_f$ and all possible adversary's choices $\delta^* \in \mathcal{U}_{e^*}$ are enumerated and a Selection Problem is solved in each iteration. This is independent of the precise fraction $b_f - \lfloor b_f \rfloor \in [0, 1)$ according to which the follower selects the fractional item. Therefore, we consider a linear segment corresponding to all possible fractions in Line 18. The worst case among all possible choices of (e^*, δ^*), which can be chosen individually for each b_f by the adversary, now corresponds to the pointwise minimum of these linear segments (Line 20). Finally, we handle each integer value b_f^* separately, similarly to the problem version of Assumption (bb). On a related note, observe that we solve a binary Selection Problem as a subroutine in Algorithms 3.5 and 3.6, and not a Continuous Selection Problem as in Algorithms 3.3 and 3.4; see also Remark 3.54 below.

Using the ideas explained above, we can prove, similarly to Theorems 3.48 and 3.51:

Theorem 3.52. *Algorithm 3.6 solves the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with a discrete uncorrelated uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$, in time $\mathcal{O}(nu \log(nu))$, where $u = \sum_{e \in \mathcal{E}_f} |\mathcal{U}_e|$.*

Algorithm 3.6. Algorithm for the leader's problem of the Robust Bilevel Selection Problem under Assumptions (disj), (cc), and (=), and with discrete uncorrelated uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $b \in \{0, \dots, |\mathcal{E}_l \cup \mathcal{E}_f|\}$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, a discrete uncorrelated uncertainty set $\mathcal{U}$ given by a finite set $\mathcal{U}_e \subseteq \mathbb{Q}$ for every $e \in \mathcal{E}_f$

Output : an optimum leader's solution $x \in [0,1]^{\mathcal{E}_l}$ of the Robust Bilevel Selection Problem under Assumptions (cc) and (=)

$b_l^- \coloneqq \max\{0, b - |\mathcal{E}_f|\}$, $b_f^+ \coloneqq b - b_l^-$
$b_l^+ \coloneqq \min\{b, |\mathcal{E}_l|\}$, $b_f^- \coloneqq b - b_l^+$
if $b_l^- = b_l^+$ **then**
  $b_l^* \coloneqq b_l^-$
else
  compute $g_l \coloneqq \mathrm{PLF}(\mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$
  for $e \in \mathcal{E}_f$ **do**
    $d^-(e) \coloneqq \min \mathcal{U}_e$, $d^+(e) \coloneqq \max \mathcal{U}_e$
  for $b_f^* \in \{b_f^-, \dots, b_f^+ - 1\}$ **do**
    $\overline{\mathcal{E}} \coloneqq \emptyset$
    for $e^* \in \mathcal{E}_f$ **do**
      for $\delta^* \in \mathcal{U}_{e^*}$ **do**
        $\mathcal{E}^-_{e^*,\delta^*} \coloneqq \{e \in \mathcal{E}_f \mid d^-(e) > \delta^*\}$
        $\mathcal{E}^0_{e^*,\delta^*} \coloneqq \{e \in \mathcal{E}_f \mid d^-(e) \le \delta^* \le d^+(e)\} \setminus \{e^*\}$
        if $|\mathcal{E}^-_{e^*,\delta^*}| \le b_f^*$ *and* $|\mathcal{E}^0_{e^*,\delta^*}| \ge b_f^* - |\mathcal{E}^-_{e^*,\delta^*}|$ **then**
          $Y^0_{b_f^*,e^*,\delta^*} \coloneqq$ Selection Problem$(\mathcal{E}^0_{e^*,\delta^*}, b_f^* - |\mathcal{E}^-_{e^*,\delta^*}|, -c \restriction \mathcal{E}^0_{e^*,\delta^*})$
          $Y_{b_f^*,e^*,\delta^*} \coloneqq \mathcal{E}^-_{e^*,\delta^*} \cup Y^0_{b_f^*,e^*,\delta^*}$
          $g_f^{b_f^*,e^*,\delta^*} \coloneqq$ linear segment from $(b - b_f^* - 1, c(Y_{b_f^*,e^*,\delta^*}) + c(e^*))$ to $(b - b_f^*, c(Y_{b_f^*,e^*,\delta^*}))$
          $\overline{\mathcal{E}} \coloneqq \overline{\mathcal{E}} \cup \{(e^*, \delta^*)\}$
    compute $g_f^{b_f^*} \coloneqq \min\{g_f^{b_f^*,e^*,\delta^*} \mid (e^*, \delta^*) \in \overline{\mathcal{E}}\}$ (i.e., the pointwise minimum)
  join all $g_f^{b_f^*}$ (with range $[b - b_f^* - 1, b - b_f^*]$) to a function g_f with range $[b_l^-, b_l^+]$
  compute $g \coloneqq g_l + g_f$
  compute some $b_l^* \in \operatorname{argmax}\{g(b_l) \mid b_l \in [b_l^-, b_l^+]\}$
return Continuous Selection Problem$(\mathcal{E}_l, b_l^*, c \restriction \mathcal{E}_l)$

Proof. First observe that, if $b_l^- = b_l^+$, then there is only one way to distribute the total capacity of b among the two players, such that Lines 3, 4, and 24 lead to an optimum leader's solution in this case. From now on, we assume $b_l^- < b_l^+$. Note that each of the loops has at least one iteration then.

Analogously to the proof of Theorem 3.48, the function g_l computed in Line 6, describing the part of the leader's objective function that represents the optimum value she can achieve by a selection of her own items when using a capacity of b_l, is well-defined and has the range $[b_l^-, b_l^+]$. Again, for the optimum value $b_l^* \in [b_l^-, b_l^+]$, an optimum leader's solution is given by Continuous Selection Problem$(\mathcal{E}_l, b_l^*, c \restriction \mathcal{E}_l)$, as returned by the algorithm in Line 24. Therefore, it remains to show that the value b_l^* derived by the algorithm is indeed optimum.

Again as in the proof of Theorem 3.48, we will show that for every fixed value $b_l \in [b_l^-, b_l^+]$, the function g computed in Line 22 correctly describes the best leader's objective function value that is possible if the leader uses a capacity of b_l and the follower uses a capacity of $b_f = b - b_l$. Hence, it remains to prove that, for every fixed value $b_l \in [b_l^-, b_l^+]$, the value of the adversary's solution returned by Algorithm 3.5, with any given feasible leader's solution x using capacity b_l, is precisely $g_f(b_l)$. The correctness of Algorithm 3.6 then follows from Theorem 3.51.

Let $b_l \in [b_l^-, b_l^+]$ be some fixed leader's capacity, $b_f = b - b_l \in [b_f^-, b_f^+]$, and $b_f^* = \lfloor b_f \rfloor$. Fix some $e^* \in \mathcal{E}_f$ and $\delta^* \in \mathcal{U}_{e^*}$ and consider the corresponding iterations of the loops in Lines 7 and 8 of Algorithm 3.5 and of the loops in Lines 9, 11, and 12 of Algorithm 3.6. The sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$ are defined in the same way in both algorithms. Also the conditions in Line 11 of Algorithm 3.5 and in Line 15 of Algorithm 3.6 are the same. Therefore, if the conditions are satisfied, the algorithms compute the same values as well, in the following sense. Firstly, the set Y_{e^*,δ^*} in Line 13 of Algorithm 3.5 can be assumed to be the same as the set $Y_{b_f^*,e^*,\delta^*}$ in Line 17 of Algorithm 3.6. Secondly, consider the linear segment $g_f^{b_f^*,e^*,\delta^*}$, which is computed in Line 18 of Algorithm 3.6. Its value in the point $b_l = b - b_f$ is the leader's objective function value of the follower's solution that selects $Y_{b_f^*,e^*,\delta^*}$ and a fraction of $b_f - \lfloor b_f \rfloor$ of the item e^*, i.e., $c(Y_{b_f^*,e^*,\delta^*}) + c(e^*) \cdot (b_f - \lfloor b_f \rfloor)$. The same value is also considered in Line 15 of Algorithm 3.5. Hence, the sets $\overline{\mathcal{E}}$ are the same in both algorithms and the value of the function $g_f^{b_f^*}$, which is computed in Line 20 of Algorithm 3.6, in the point $b_l = b - b_f$ is the same as the minimum value that is considered in Line 15 of Algorithm 3.5. The same holds for the function g_f, which is determined in Line 21 of Algorithm 3.6. This finishes the proof of the correctness of Algorithm 3.6.

It remains to analyze the running time of Algorithm 3.6. According to Section 3.3.2, the function g_l has at most n linear segments and can be computed in time $\mathcal{O}(n \log(n))$ in Line 6. Analogously to Algorithm 3.5 (see Theorem 3.51), one iteration of the loop in Lines 9 to 20, except for Line 20, can be performed in time $\mathcal{O}(nu)$, where $u = \sum_{e \in \mathcal{E}_f} |\mathcal{U}_e|$. This would lead to a total running time of $\mathcal{O}(n^2 u)$ when considering the whole loop in Lines 9 to 20, except for Line 20. However, this time can be reduced by precomputing the sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$ and the sorting of $\mathcal{E}^0_{e^*,\delta^*}$ required for the Selection Problem in Line 16, for all u many possible choices of (e^*, δ^*). This is similar to the precomputations in several of the previous algorithms; see, e.g., Remark 3.19, Theorem 3.25, and Corollary 3.28. This can be done in time $\mathcal{O}(un \log(n))$ and enables to perform each iteration of the inner loop in constant time afterwards. The function $g_f^{b_f^*}$, which is defined as the pointwise

minimum of at most u linear segments in Line 20, can be computed in time $\mathcal{O}(u \log(u))$, using [Her89]; see also the proof of Theorem 3.44. This leads to a total running time of $\mathcal{O}(nu \log(nu))$ for the loop in Lines 9 to 20. Finally, again similarly to the proof of Theorem 3.44, the sum of the piecewise linear functions g_l and g_f and the maximum of the resulting piecewise linear function can be computed in time $\mathcal{O}(nu)$, as the total number of linear segments is $\mathcal{O}(nu)$ as well. Together, this results in a running time of $\mathcal{O}(nu \log(nu))$ for the whole algorithm. □

We emphasize that the term $u = \sum_{e \in \mathcal{E}_f} |\mathcal{U}_e|$ and therefore the running times stated in Theorems 3.51 and 3.52 are polynomial in the input size, while the total size $|\mathcal{U}| = \prod_{e \in \mathcal{E}_f} |\mathcal{U}_e|$ of the discrete uncorrelated uncertainty set is exponential in the input size. Moreover, if the uncertainty set is given by sets $\mathcal{U}_e$ of constant size, in particular if each $\mathcal{U}_e$ consists of only two values $d^-(e)$ and $d^+(e)$, then the running time of Algorithm 3.5 is $\mathcal{O}(n^2)$, and the running time of Algorithm 3.6 is $\mathcal{O}(n^2 \log n)$, like in case of interval uncertainty; see Theorem 3.34 and Corollary 3.35, and Theorems 3.47 and 3.48.

Remark 3.53. Observe that $g_f^{b_f^*}$ is defined for all $b_f^* \in \{b_f^-, \dots, b_f^+ - 1\}$ and that the function g_f is continuous, although both is not directly obvious from Algorithm 3.6: The alternative way of describing the function g_f, as a pointwise minimum of continuous piecewise linear functions for all (exponentially many) scenarios in $\mathcal{U}$ (see Remark 3.42 and also Remark 3.49), demonstrates that this is true. In fact, similarly to Algorithm 3.4, Algorithm 3.6 provides a way to determine g_f by computing only a polynomial number of linear segments that might contribute to the pointwise minimum.

Remark 3.54. The approach we developed for discrete uncorrelated uncertainty is similar to the one for interval uncertainty, but there are some differences we need to point out. As mentioned above, the heads $\bar{e}$ enumerated in Algorithms 3.3 and 3.4 are not necessarily the fractional items in the resulting follower's solutions; they only provide useful reference points in order to find all relevant fractional prefixes. Accordingly, we solve a Continuous Selection Problem in a subroutine there and determine the fractional item during that. In contrast, we fix the fractional item e^* in Algorithms 3.5 and 3.6, and solve a binary Selection Problem in a subroutine, in order to determine the remaining items of the fractional prefix. In fact, we could use the latter approach also for the case of interval uncertainty in order to enumerate the possible fractional items, together with a reasonable finite set of possible follower's item values the adversary may assign to them (depending on the endpoints of the other intervals), in polynomial time. However, this would not give any better results than Theorems 3.47 and 3.48.

As we will discuss in more detail in Remark 4.33, in the corresponding versions of the Robust Bilevel Continuous Knapsack Problem, the difference between the two approaches is more significant. In fact, the approach that is only applicable in case of interval uncertainty still runs in polynomial time, while the second one only has a pseudopolynomial running time. For this reason, both approaches are worth to be studied.

Remark 3.55. In Sections 3.4.1.1.2 and 3.4.1.2.2, for interval uncertainty, we first assumed that there are no one-point intersections between the intervals in the uncertainty set and then argued that, if this is not the case, technical adjustments can be made to the algorithms such that they correctly handle the optimistic and the pessimistic case; see Remark 3.36 for more details. For discrete uncorrelated uncertainty, one could assume

that the sets $\mathcal{U}_e$ are pairwise disjoint, in order to ensure that distinguishing the optimistic and the pessimistic case is not necessary. However, analogously to interval uncertainty, Algorithms 3.5 and 3.6, as they are stated above, are also correct for the pessimistic setting without this assumption, while an adjusted definition of the item sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$ is required for the optimistic setting, similarly to Remark 3.36. More precisely, the two sets can be defined as follows:

$$\begin{aligned}\mathcal{E}^-_{e^*,\delta^*} &:= \{e \in \mathcal{E}_f \mid d^-(e) > \delta^* \text{ or } [d^-(e) = \delta^* \text{ and } c(e) > c(e^*)]\}\\ \mathcal{E}^0_{e^*,\delta^*} &:= \{e \in \mathcal{E}_f \mid d^-(e) < \delta^* < d^+(e) \text{ or } [d^-(e) < d^+(e) = \delta^* \text{ and } c(e) > c(e^*)]\\ &\qquad\qquad \text{or } [\delta^* = d^-(e) < d^+(e) \text{ and } c(e) < c(e^*)]\} \setminus \{e^*\}\end{aligned}$$

The items in $\mathcal{E}^-_{e^*,\delta^*}$ are again the ones that will appear before e^* in the follower's greedy order, assuming that the adversary sets $d(e^*) = \delta^*$ and no matter how he sets $d(e)$ for the other items $e \in \mathcal{E}_f \setminus \{e^*\}$. Similarly, the items in $\mathcal{E}^0_{e^*,\delta^*}$ are the ones that might appear either before or after e^*, again assuming $d(e^*) = \delta^*$ and depending on the adversary's decision for the other items.

3.4.2 The Robust Bilevel Selection Problem Under Assumption (sub)

In the setting of Assumption (sub), the Robust Bilevel Selection Problem becomes much more involved than under Assumption (disj). Indeed, when the leader's and the follower's item sets are not disjoint, it is not clear anymore that a greedy decision on the leader's items is optimum. This can make the leader's problem significantly harder than the adversary's problem, in contrast to, e.g., Theorem 3.25 under Assumptions (disj) and (bb), which showed that the leader's problem can be polynomially reduced to the adversary's problem in this case. In fact, we will prove in Theorem 3.58 that the leader's problem is strongly $\mathcal{NP}$-hard under Assumptions (sub) and (bb), for discrete uncertainty, while the corresponding adversary's problem is solvable in polynomial time by Theorem 3.27.

From the adversary's perspective, under Assumption (bb), there is no relevant difference between Assumptions (sub) and (disj), and we have already proved that the adversary's problem can be solved in polynomial time for all types of uncertainty considered so far; see Theorems 3.27 and 3.34 and Corollary 3.38. In case of Assumption (cc), we have to be more careful when it comes to distinguishing Assumptions (sub) and (disj), even for the adversary's problem. Indeed, in case of Assumptions (sub) and (cc), the follower's problem is a Continuous Knapsack Problem rather than a Continuous Selection Problem; see the discussion before Lemma 3.14 as well as Section 3.4.2.2 for more details.

The leader's problem under Assumption (sub) cannot be expected to be easily solvable in most settings, as indicated by the complications explained above. However, in the setting of Assumptions (sub) and (bb), we will suggest some approaches to deal with the Robust Bilevel Selection Problem: A 2-approximation algorithm as well as two exact algorithms that run in superpolynomial time will be presented in Section 3.4.2.1, together with the $\mathcal{NP}$-hardness result mentioned above.

In Section 3.4.2.2, we will then discuss a few insights about the setting of Assumptions (sub) and (cc), which seems to be the case that is most complicated to handle.

3.4.2.1 The Robust Bilevel Selection Problem Under Assumptions (sub) and (bb)

In the setting of Assumptions (disj) and (bb), we have seen that the leader's problem can be solved in polynomial time for all considered types of uncertainty sets, by reducing it to the adversary's problem; see Theorem 3.25. In this algorithm, the leader enumerates how many items she selects herself and selects them greedily for every fixed number. We will now see that this algorithm does not yield an optimum solution anymore in the setting of Assumption (sub). This is in contrast to the problem without uncertainty, where the same idea could be used for the cases of both Assumptions (disj) and (sub); see Section 3.3.1. However, the difference is not surprising here in view of the result of Theorem 3.58 that the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty, is strongly $\mathcal{NP}$-hard. But the algorithm that is exact in case of Assumption (disj) now turns out to be a 2-approximation algorithm, at least for the minimization variant of the Robust Bilevel Selection Problem, as we will show in Theorem 3.56. In fact, we again prove this for any type of uncertainty set for which the adversary's problem can be solved in polynomial time, like in Theorem 3.25.

Besides approximation algorithms, another way to deal with $\mathcal{NP}$-hard problems is to solve them in superpolynomial time. We will analyze two such algorithms for the Robust Bilevel Selection Problem under Assumptions (sub) and (bb). The first approach, which relies on enumerating all possible leader's solutions, again works for any uncertainty set for which the adversary's problem can be solved in reasonable time; see Theorem 3.56. The second approach is specific to discrete uncertainty and will be presented in Theorem 3.63. The latter result shows that the problem can be solved in polynomial time given a constant number of scenarios.

We now start with the presentation of the approximation result and the first exact algorithm, which can both be formulated without specifying the type of uncertainty set. After that, we will again distinguish discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty in Sections 3.4.2.1.1, 3.4.2.1.2, and 3.4.2.1.3.

In order to make reasonable statements about approximating objective function values, we have to assume that they are nonnegative; see also Section 2.1.2. More precisely, we assume, for the following statements, that the function $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}_{\geq 0}$ of leader's item values only attains nonnegative values. Recall from Remark 3.4 that all leader's item values can be shifted arbitrarily by a constant, without changing the optimal solutions. However, this cannot be done without loss of generality here, as the resulting constant offset in the objective values has an impact on approximation factors.

Under the assumption of nonnegative leader's item values, minimization and maximization cannot be simply exchanged, as it is the case when we allow for arbitrary rational item values, like in the rest of this chapter; see also Remark 3.5. The following result is originally due to [Har21] (while its proof has been revised and simplified by the author of this thesis) where Hartmann considers the variant of the Robust Bilevel Selection Problem in which the leader and the follower solve minimization problems. Therefore, the following statement is formulated for this problem variant as well. It remains open whether the same assertion holds also for the maximization variant of the Robust Bilevel Selection Problem as we defined it in this chapter; see also Section 6.3.1.

Theorem 3.56. *Consider the minimization version of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with any type of uncertainty set. Suppose that the corresponding adversary's problem can be solved in time at most $A(I)$, given an*

instance I of the minimization version of the Robust Bilevel Selection Problem *together with any leader's solution. Then the problem of 2-approximating the minimization version of the* Robust Bilevel Selection Problem*, given an instance I with n items, can be solved in time $\mathcal{O}(n \log n + nA(I))$.*

Proof. The leader enumerates all values $b_l \in \{0, \dots, \min\{b, n_l\}\}$ and solves a single-level Selection Problem on $\mathcal{E}_l$ with capacity b_l in each iteration. Moreover, she determines an optimum solution of the adversary's problem corresponding to this leader's solution. The best solution achieved throughout these iterations is returned. This is the same algorithm as in Theorem 3.25, and also its running time is the same.

We will now prove that the algorithm always returns a solution with leader's costs that are at most twice the costs of an optimum solution. For this, denote the leader's solution in iteration b_l of the algorithm by X_{b_l}, the corresponding follower's response (assuming that this is unique due to a fixed follower's greedy order; see Remark 3.2) in any scenario $d \in \mathcal{U}$ by $Y^d_{b_l}$, and the follower's response in some worst-case scenario by Y_{b_l}, i.e., choose $Y_{b_l} \in \{Y^d_{b_l} \mid d \in \mathcal{U}\}$ with $c(Y_{b_l}) = \max\{c(Y^d_{b_l}) \mid d \in \mathcal{U}\}$. Moreover, denote an optimum leader's solution by X^*, the corresponding follower's response in any scenario $d \in \mathcal{U}$ by Y^d, and the follower's response in some worst-case scenario by Y^*.

For the rest of the proof, choose $b_l \in \{|X^*|, \dots, \min\{b, n_l\}\}$ to be minimum such that $|X_{b_l} \cap Y^d| \leq b_l - |X^*|$ for all $d \in \mathcal{U}$. This is always possible because $|X^*| \leq \min\{b, n_l\}$ and the inequality is always satisfied for $b_l = \min\{b, n_l\}$. Indeed, since X^* and Y^d are disjoint for all $d \in \mathcal{U}$ as corresponding leader's and follower's solutions, we then have either $|X_b \cap Y^d| \leq |Y^d| = b - |X^*|$ or $|X_{n_l} \cap Y^d| = |\mathcal{E}_l \cap Y^d| \leq |\mathcal{E}_l \setminus X^*| = n_l - |X^*|$ for all $d \in \mathcal{U}$.

If $b_l = 0$, we must have $X^* = X_{b_l} = \emptyset$, which means that the algorithm finds an optimum solution. Therefore, we assume $b_l \geq 1$ in the following.

Due to the minimality of b_l, we have $b_l = |X^*|$ or there is some $d \in \mathcal{U}$ with $|X_{b_l-1} \cap Y^d| > b_l - 1 - |X^*|$, which is equivalent to

$$|X_{b_l-1} \cap Y^d| \geq b_l - |X^*|. \tag{3.1}$$

In fact, $b_l = |X^*|$ also implies (3.1), even for all $d \in \mathcal{U}$. Hence, there is always some $d \in \mathcal{U}$ for which (3.1) is satisfied. Fix such a scenario d in the following.

In order to prove the required approximation guarantee, we will show that the leader's cost of both the leader's and the follower's solution in iteration b_l can be bounded from above by the costs of an optimum solution, i.e., $c(X_{b_l}) \leq c(X^*) + c(Y^*)$ and $c(Y_{b_l}) \leq c(X^*) + c(Y^*)$. Together, this proves that the total costs of the solution that the algorithm returns is at most $c(X_{b_l}) + c(Y_{b_l}) \leq 2(c(X^*) + c(Y^*))$.

We first focus on bounding $c(X_{b_l})$. For this, we partition the set X_{b_l} into the two disjoint sets $B \coloneqq X_{b_l-1} \cap Y^d$ and $A \coloneqq X_{b_l} \setminus B$. Note that $B \subseteq X_{b_l}$ because X_{b_l} consists of X_{b_l-1} and one additional item by the construction of the leader's solutions in the algorithm. Clearly, $c(B) \leq c(Y^d) \leq c(Y^*)$ holds, so it remains to show $c(A) \leq c(X^*)$.

Since X_{b_l} is selected greedily from $\mathcal{E}_l$ according to the costs c, we have $c(e_1) \leq c(e_2)$ for all $e_1 \in X_{b_l}$ and all $e_2 \in \mathcal{E}_l \setminus X_{b_l}$, and in particular for all $e_1 \in A \setminus X^* \subseteq X_{b_l}$ and all $e_2 \in X^* \setminus A \subseteq \mathcal{E}_l \setminus X_{b_l}$. The last inclusion holds because $X^* \cap Y^d = \emptyset$ for the follower's response Y^d to the leader's solution X^*, which implies $X^* \cap B = \emptyset$ and hence $X^* \cap X_{b_l} \subseteq A$.

Moreover, we can use (3.1) to derive $|A| = |X_{b_l}| - |B| = b_l - |X_{b_l-1} \cap Y^d| \leq |X^*|$ and hence $|A \setminus X^*| \leq |X^* \setminus A|$. Together with the previous paragraph, this yields

$c(A) = c(A \setminus X^*) + c(A \cap X^*) \leq c(X^* \setminus A) + c(A \cap X^*) = c(X^*)$. This concludes the first part of the proof with $c(X_{b_l}) = c(A) + c(B) \leq c(X^*) + c(Y^*)$.

For the second part, we partition also the set Y_{b_l} into two disjoint sets $A' \coloneqq Y_{b_l} \cap X^*$ and $B' \coloneqq Y_{b_l} \setminus X^*$. We directly observe $c(A') \leq c(X^*)$, so it remains to prove $c(B') \leq c(Y^*)$.

Let $d' \in \mathcal{U}$ be the worst-case scenario in the considered solution of the algorithm, i.e., such that $Y_{b_l}^{d'} = Y_{b_l}$. We derive $|B'| = |Y_{b_l} \setminus X^*| \leq |Y_{b_l}| = b - b_l$ and, by the definition of b_l, $|Y^{d'} \setminus X_{b_l}| = |Y^{d'}| - |Y^{d'} \cap X_{b_l}| \geq |Y^{d'}| - (b_l - |X^*|) = b - b_l$. Thus, $|B'| \leq |Y^{d'} \setminus X_{b_l}|$.

We know that $Y_{b_l}^{d'}$ is selected greedily from $\mathcal{E} \setminus X_{b_l}$ by the follower, according to the costs d'. (Recall that we assume this to be based on a fixed order of the follower's items; see Remark 3.2.) This means that $Y_{b_l}^{d'}$ consists of the first $|Y_{b_l}^{d'}| = b - b_l$ items of the corresponding order of the set $\mathcal{E} \setminus X_{b_l}$. By removing X^* from the sorted set, we obtain that $B' = Y_{b_l}^{d'} \setminus X^*$ consists of the first $|B'|$ items of the same order of the set $\mathcal{E} \setminus (X_{b_l} \cup X^*)$.

Analogously, $Y^{d'}$ is selected greedily from $\mathcal{E} \setminus X^*$ by the follower, i.e., again according to d'. We may assume that the same order as above is used. We now remove X_{b_l} and derive that $Y^{d'} \setminus X_{b_l}$ consists of the first $|Y^{d'} \setminus X_{b_l}|$ items of the same order of the set $\mathcal{E} \setminus (X_{b_l} \cup X^*)$.

Together with $|B'| \leq |Y^{d'} \setminus X_{b_l}|$, this implies that $B' \subseteq Y^{d'} \setminus X_{b_l}$ and thus $c(B') \leq c(Y^{d'}) \leq c(Y^*)$, which concludes the second part of the proof. □

In the following sections, we will again investigate several types of uncertainty sets. In order to obtain more specific approximation algorithms for the leader's problem, Theorem 3.56 can then be combined with the algorithmic results for the adversary's problem of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb) presented in Theorems 3.27 and 3.34 and Corollary 3.38. This leads to Corollaries 3.60, 3.65, and 3.68.

Note that, analogously to Remark 3.26, the running time $\mathcal{O}(n \log n + nA(I))$ stated in Theorem 3.56 could also be written as $\mathcal{O}(nA(I))$, but will be adapted anyway when specifying the algorithm to fixed types of uncertainty sets.

We now turn towards exact algorithms for the Robust Bilevel Selection Problem under Assumptions (sub) and (bb) that have exponential running time and can be interesting from the perspective of parameterized complexity (see Section 2.1.3).

Clearly, an exact algorithm can be achieved by enumerating potential leader's solutions; see Theorem 3.57. Similarly to Theorem 3.56, it uses an algorithm for the adversary's problem as a black box and can be specified for uncertainty sets for which we have such an algorithm.

Theorem 3.57. *Consider the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with any type of uncertainty set. Suppose that the corresponding adversary's problem can be solved in time at most $A(I)$, given an instance I of the Robust Bilevel Selection Problem together with any leader's solution. Then the Robust Bilevel Selection Problem, given an instance I with n_l leader's items and capacity b, can be solved in time $\mathcal{O}(\min\{2^{n_l}, n_l^b\}A(I))$.*

Proof. The Robust Bilevel Selection Problem can be solved by enumerating all feasible leader's solutions $X \subseteq \mathcal{E}_l$ and solving the adversary's problem for each of them. Recall that, under Assumption (sub), every $X \subseteq \mathcal{E}_l$ with $|X| \leq b$ is feasible. The adversary's problem is solved by an algorithm specific for the given type of uncertainty set. This algorithm can be assumed to return a corresponding follower's solution $Y \subseteq \mathcal{E}_f$ as well. (If not, we can solve the follower's problem in time $\mathcal{O}(n)$ for the given leader's solution and

adversary's choice.) By comparing the leader's costs of all resulting solutions $X \cup Y$, an optimum leader's solution can be determined.

For the running time, note that the number of feasible leader's solutions $X \subseteq \mathcal{E}_l$ is bounded by 2^{n_l} and, if $b \leq n_l$, by $\sum_{b_l=0}^{b} \binom{n_l}{b_l} \in \mathcal{O}(n_l^b)$ because the leader can select at most b items in a feasible solution. □

This implies that the Robust Bilevel Selection Problem under Assumptions (sub) and (bb) is fixed-parameter tractable ($\mathcal{FPT}$) in the parameter n_l as well as slice-wise polynomial ($\mathcal{XP}$) with respect to the parameter b, whenever the adversary's problem can be solved fast enough (e.g., in polynomial time). For discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty, we state more specific versions of this result in Corollaries 3.62, 3.67, and 3.69, respectively. For discrete uncertainty, we will also present a more advanced algorithm in Theorem 3.63, showing that the problem is also slice-wise polynomial with respect to the parameter $|\mathcal{U}|$.

3.4.2.1.1 Discrete Uncertainty The Robust Bilevel Selection Problem under Assumptions (sub) and (bb) is $\mathcal{NP}$-hard, as we prove in Theorem 3.58 for discrete uncertainty. For this, we will need the following optimization problem, which is well-known to be $\mathcal{NP}$-hard [GJ79]. For an introduction to the basic notions from graph theory we refer to Section 5.1.

Vertex Cover Problem
Instance: An undirected graph $G = (V, E)$
Task: Find a vertex cover, i.e., a vertex set $X \subseteq V$ such that, for every edge $e \in E$, at least one endpoint of e is contained in X, of minimum cardinality.

Theorem 3.58. *The Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty, is strongly $\mathcal{NP}$-hard.*

Proof. We prove the result by a reduction from the $\mathcal{NP}$-hard Vertex Cover Problem. Consider an instance of the Vertex Cover Problem, consisting of an undirected graph $G = (V, E)$ with $n \in \mathbb{N}$ vertices $V = \{v_1, \ldots, v_n\}$ and $m \in \mathbb{N}$ edges $E = \{e_1, \ldots, e_m\}$. Without loss of generality, we may assume that the minimum cardinality of a vertex cover in this instance is at least 2 and at most $n - 1$. We build an instance of the Robust Bilevel Selection Problem as follows: $\mathcal{E}_l = V$, $\mathcal{E}_f = V \cup \{h_1, \ldots, h_{n+3}\}$, $b = n + 1$, and

$$c(h) = \begin{cases} n-1 & \text{for } h \in V \cup \{h_{n+1}, h_{n+2}\}, \\ 0 & \text{for } h = h_{n+3}, \\ n & \text{for } h \in \{h_1, \ldots, h_n\}. \end{cases}$$

The uncertainty set $\mathcal{U} = \{d_1, \ldots, d_{m+1}\}$ consists of a scenario d_i for every edge $e_i \in E$ of G and one additional scenario d_{m+1}. Instead of defining the scenarios, i.e., follower's objective functions, d_i explicitly, it suffices to give orders of the items that are used in the follower's greedy algorithm in the respective scenarios. It is always possible to define a function d_i for which this order is the unique optimal one for the follower and, in particular, to compute such a function d_i with polynomial-size values in polynomial time; see also Remark 3.2. Note that the follower never selects more than $b = n + 1$ items, so that we can neglect the items after the first $n + 1$ ones in the follower's greedy order. For any $i \in \{1, \ldots, m\}$, the scenario d_i, corresponding to the edge $e_i = \{v_j, v_k\} \in E$ with $j < k$, is defined by the order

$h_{n+1}, h_{n+2}, v_1, \ldots, v_{j-1}, v_{j+1}, \ldots, v_{k-1}, v_{k+1}, \ldots, v_n, h_{n+3}$. The scenario d_{m+1} is defined by the order $h_{n+3}, h_1, \ldots, h_n$.

We will prove that an optimum leader's solution $X \subseteq \mathcal{E}_l = V$ is a minimum vertex cover in G. First, let $X \subseteq \mathcal{E}_l$ not be a vertex cover, i.e., there is an edge $e_i = \{v_j, v_k\} \in E$ with $j < k$ such that $v_j \notin X$ and $v_k \notin X$. By definition of the corresponding scenario d_i, all items selected by the leader in X are among the first $n+1$ items in the follower's greedy order. Hence, when the follower completes the leader's solution X greedily to $b = n+1$ items in total, this leads to a follower's solution Y with $X \cup Y$ consisting of exactly the first $n+1$ items in the follower's greedy order, i.e., $X \cup Y = \{h_{n+1}, h_{n+2}, v_1, \ldots, v_{j-1}, v_{j+1}, \ldots, v_{k-1}, v_{k+1}, \ldots, v_n, h_{n+3}\}$, which has a leader's objective function value of $c(X \cup Y) = n(n-1) = n^2 - n$. Thus, in the robust setting, every leader's solution X that is not a vertex cover leads to an objective function value of $n(n-1)$, which is the worst possible value from the leader's perspective.

Now we turn to leader's solutions $X \subseteq \mathcal{E}_l$ that are vertex covers. By definition, for every edge $e_i \in E$, at least one of its endpoints is in X. Accordingly, for every scenario d_i, at least one item that is not among the first $n+1$ items of the follower's greedy order in this scenario is selected by the leader. When the follower now greedily completes the leader's solution X to $n+1$ items in total, he will never select the item h_{n+3}, which is the $(n+1)$-th item in his greedy order. In fact, we get $X \cup Y \subseteq V \cup \{h_{n+1}, h_{n+2}\}$. Thus, in each of the scenarios $d_1, \ldots, d_m$, the leader's objective value is at least $(n+1)(n-1) = n^2 - 1$. In scenario d_{m+1}, the follower always selects item h_{n+3} and possibly some items with leader's objective value n. For the leader's solution X and the corresponding follower's solution Y, the leader's objective value is thus $c(X \cup Y) = |X|(n-1) + (n - |X|)n = n^2 - |X|$. This implies that the adversary can be assumed to always select scenario d_{m+1} when the leader has selected a vertex cover X, and that, as a leader's solution, any vertex cover is at least as good as any set that is not a vertex cover. Finally, optimizing the leader's objective value among the solutions that are vertex covers is equivalent to finding a vertex cover $X \subseteq V$ of minimum size $|X|$.

Note that we have indeed shown strong $\mathcal{NP}$-hardness, since the Vertex Cover Problem does not have any numerical parameters and all values of the constructed functions c and $d_i \in \mathcal{U}$ have polynomial size. □

While Theorem 3.58 shows $\mathcal{NP}$-hardness, observe that the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty, is also $\mathcal{NP}$-easy and therefore $\mathcal{NP}$-equivalent. Indeed, the corresponding adversary's problem is solvable in polynomial time (see Theorem 3.27) and the leader's problem cannot be more than one level harder than the adversary's problem, i.e., the evaluation of the leader's objective function; see also Theorem 2.39.

Remark 3.59. The proof of Theorem 3.58 shows the hardness of the maximization version of the Robust Bilevel Selection Problem with nonnegative leader's item values. In view of the approximation results in Theorem 3.56 and Corollary 3.60, the question arises whether the same hardness result can be obtained for the minimization version of the Robust Bilevel Selection Problem with nonnegative leader's item costs, or, equivalently, the maximization version of the Robust Bilevel Selection Problem with nonpositive leader's item values. In fact, the proof above is still valid when any polynomial-size constant $C \in \mathbb{Z}$ is added to each of the leader's item values $c(h)$, for all

$h \in \mathcal{E}_f$. The reason for this is that the value of every feasible solution changes by exactly bC because every feasible solution consists of exactly b items by definition; see also Remark 3.4. In particular, subtracting n from each of the leader's item values defined above leads to an instance with nonpositive values. Such a version of the proof is also presented in [Hen25, Theorem 15].

Because of Theorem 3.58, we cannot hope for a polynomial-time exact algorithm (unless $\mathcal{P} = \mathcal{NP}$) for the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty. However, we get the following approximation result:

Corollary 3.60. *The minimization version of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty, can be 2-approximated in $\mathcal{O}(|\mathcal{U}|n \log n)$ time.*

Proof. A polynomial-time 2-approximation algorithm directly follows from Theorems 3.27 and 3.56. Analogously to Corollary 3.28, its running time can be improved from $\mathcal{O}(|\mathcal{U}|n^2)$ to $\mathcal{O}(|\mathcal{U}|n \log n)$ by precomputing the bijections for all scenarios. Note that, in every iteration, we might have to remove the item that the leader adds to her solution from the items the follower chooses from, for each scenario, like in the general version of Algorithm 3.1; see also the proof of Theorem 3.20. □

The following example shows that the approximation factor of 2 is actually tight for the given algorithm in the case of discrete uncertainty. However, it remains open whether a better approximation can be achieved by another algorithm. Note that the reduction in the proof of Theorem 3.58 does not imply any statement about the approximability of the problem.

Example 3.61. Consider the following family of instances of the minimization version of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty: Let $n \in \mathbb{N}$ with $n \geq 4$ and $\varepsilon \in \mathbb{Q}_{>0}$, and set $b = n-2$. Define the item sets as $\mathcal{E}_l = \{e_1, e_2\}$ and $\mathcal{E}_f = \{e_1, e_2, e_3, \dots, e_n\}$, with leader's item costs $c(e_1) = 1 - \varepsilon$, $c(e_2) = 1$, $c(e_3) = 3$, and $c(e) = 0$ for all other items $e \in \{e_4, \dots, e_n\}$, and follower's item costs according to the two scenarios $\mathcal{U} = \{d_1, d_2\}$ such that the follower's greedy order in scenario d_1 is $e_4, \dots, e_n, e_3, e_1, e_2$, and in scenario d_2 it is $e_2, e_4, \dots, e_n, e_1, e_3$. Recall that it suffices to define the scenarios via the follower's greedy orders because the actual costs in the follower's objective are not important for the leader; see Remark 3.2.

The algorithm from Corollary 3.60 considers the three leader's selections $X_0 = \emptyset$, $X_1 = \{e_1\}$, and $X_2 = \{e_1, e_2\}$. They lead to worst-case follower's responses $Y_0 = \{e_4, \dots, e_n, e_3\}$ (in scenario d_1), $Y_1 = \{e_2, e_4, \dots, e_{n-1}\}$ (in scenario d_2), and $Y_2 = \{e_4, \dots, e_{n-1}\}$ (in both scenarios), respectively. The resulting leader's objective function values are 3, $2 - \varepsilon$, and $2 - \varepsilon$. Hence, the leader will choose either X_1 or X_2 and achieve a cost value of $2 - \varepsilon$. The optimum leader's solution $X^* = \{e_2\}$, however, leads to the worst-case follower's response $Y^* = \{e_4, \dots, e_{n-1}\}$ (in both scenarios) and therefore to a leader's objective value of only 1. This shows that, for any number of items, the solution returned by the algorithm might achieve an objective value of almost twice the optimum value. Depending on the choice the algorithm makes in case the greedy order of the leader's items is not unique, the same behavior might even occur for $\varepsilon = 0$, which would deliver a factor of exactly 2.

Next, we look at exact algorithms for the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty. Combining Theorems 3.27 and 3.57 directly implies:

Corollary 3.62. *The Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with a discrete uncertainty set $\mathcal{U}$, can be solved in time $\mathcal{O}(\min\{2^{n_l}, n_l^b\}|\mathcal{U}|n)$.*

We now derive another, more complex, exact algorithm which proves that the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with a discrete uncertainty set $\mathcal{U}$, is slice-wise polynomial ($\mathcal{XP}$) in the parameter $|\mathcal{U}|$, i.e., for every constant number of scenarios, the problem is solvable in polynomial time; see Theorem 3.63. We develop the idea of the algorithm in the following. A detailed description can also be found in [Har21].

To illustrate the idea, we first make some observations about the Bilevel Selection Problem under Assumption (sub), without uncertainty. Recall that, from the follower's perspective, a greedy order of his items $\mathcal{E}_f$ can be fixed, independently of the leader's choice, and we may assume that the follower always selects a prefix of this order, possibly after removing the items that have already been selected by the leader. We utilized this also in Algorithm 3.1. Hence, every overall solution $X \cup Y$ can be assumed to consist of a prefix of the follower's greedy order (some part of which might have been selected by the leader and the rest by the follower) and possibly some single leader's items that appear later in the follower's greedy order. Therefore, we could also find a feasible overall solution $X \cup Y$ that is best possible for the leader as follows: Enumerate all prefixes $\overline{Y} \subseteq \mathcal{E}_f$ of the follower's greedy order and, for each of them, check whether the leader can select some solution $X \subseteq \mathcal{E}_l$ such that, together with the follower's response Y to X, the overall solution $X \cup Y$ consists of the prefix $\overline{Y}$ and $b - |\overline{Y}|$ additional leader's items. If this is possible, find the best such leader's choice X according to the leader's objective c. The best solution obtained in this enumeration process must be an optimum leader's solution.

The subproblem for a fixed prefix $\overline{Y} \subseteq \mathcal{E}_f$ is easy to solve: Clearly, the leader must select $b - |\overline{Y}|$ items from $\mathcal{E}_l \setminus \overline{Y}$. If this set does not contain sufficiently many items, then there is no feasible solution of the desired structure corresponding to the prefix $\overline{Y}$. In addition, the leader may select any number of items from $\mathcal{E}_l \cap \overline{Y}$, but this has no influence on the overall solution $X \cup Y$ because the follower's optimum response is always $Y = \overline{Y} \setminus X$. The selection of $b - |\overline{Y}|$ items from $\mathcal{E}_l \setminus \overline{Y}$ can be made in a greedy way according to the leader's objective c because there is no difference in the follower's reaction among different such selections. Therefore, each iteration in the above algorithm boils down to a single-level Selection Problem.

In fact, this algorithm is very similar to Algorithm 3.1 because it also enumerates greedy choices of different sizes, for the leader and for the follower. Due to the different perspective of starting with the follower's instead of the leader's prefixes, only the items that could be selected by either player are handled differently: Observe that the leader's greedy solutions that are enumerated in Algorithm 3.1 correspond to the solutions of the single-level Selection Problems here, together with a specific additional choice from $\mathcal{E}_l \cap \overline{Y}$, which does not influence the overall solution.

We now turn to the robust setting again. The different scenarios now correspond to different follower's greedy orders, and the adversary may choose between them. However, in any given scenario, the above structural observations still apply, and a generalization of the enumeration procedure described above now solves the Robust Bilevel Selection

Problem. Instead of one prefix $\overline{Y} \subseteq \mathcal{E}_f$ of a single follower's greedy order, we now consider in each iteration a combination of prefixes of the greedy orders in all scenarios, i.e., some collection $(\overline{Y_d})_{d\in\mathcal{U}}$ such that $\overline{Y_d} \subseteq \mathcal{E}_f$ is a prefix of the follower's greedy order in scenario d, for all $d \in \mathcal{U}$. The problem that has to be solved in a given iteration is then to find the best leader's selection $X \subseteq \mathcal{E}_l$ (or decide that there is no such X) such that, for every scenario $d \in \mathcal{U}$, the overall solution $X \cup Y_d$ in this scenario (i.e., Y_d is the follower's optimum solution in scenario d) consists of the prefix $\overline{Y_d}$ of the follower's greedy order corresponding to this scenario and $b - |\overline{Y_d}|$ additional leader's items.

This subproblem can now be solved by another enumeration and several single-level Selection Problems on appropriate subsets of $\mathcal{E}_l$. We illustrate this for the setting of two scenarios $\mathcal{U} = \{d_1, d_2\}$: Given prefixes $\overline{Y_{d_1}}, \overline{Y_{d_2}} \subseteq \mathcal{E}_f$, in analogy to the situation without uncertainty above, the leader has to select $b - |\overline{Y_{d_1}}|$ items from $\mathcal{E}_l \setminus \overline{Y_{d_1}}$ and $b - |\overline{Y_{d_2}}|$ items from $\mathcal{E}_l \setminus \overline{Y_{d_2}}$. Since these two selections are not independent in general, we reformulate them as follows. If the leader selects b_1 items from $\mathcal{E}_l \setminus (\overline{Y_{d_1}} \cup \overline{Y_{d_2}})$, b_2 items from $(\mathcal{E}_l \cap \overline{Y_{d_1}}) \setminus \overline{Y_{d_2}}$, b_3 items from $(\mathcal{E}_l \cap \overline{Y_{d_2}}) \setminus \overline{Y_{d_1}}$, and b_4 items from $\mathcal{E}_l \cap \overline{Y_{d_1}} \cap \overline{Y_{d_1}}$, then this leads to the desired situation if and only if $b_1 + b_3 = b - |\overline{Y_{d_1}}|$ and $b_1 + b_2 = b - |\overline{Y_{d_2}}|$. Note that it does not matter how many items the leader selects that are contained in all considered prefixes, i.e., what the number b_4 is. The subproblem can now be solved by enumerating all combinations of values for b_1, b_2, and b_3 that satisfy these conditions and solving the three corresponding single-level Selection Problems.

It remains to analyze the running time of the algorithm. In the beginning, the follower's greedy orders in all scenarios are computed in time $\mathcal{O}(|\mathcal{U}|n \log n)$. In the outer enumeration, we consider $\mathcal{O}(b)$ possible prefixes for each scenario, which gives $\mathcal{O}(b^{|\mathcal{U}|})$ combinations $(\overline{Y_d})_{d\in\mathcal{U}}$ of prefixes in total. In the subproblem, the enumeration considers $\mathcal{O}(2^{|\mathcal{U}|})$ many subsets of $\mathcal{E}_l$, one for each subset $\overline{\mathcal{U}} \subseteq \mathcal{U}$ of the scenarios, namely $(\mathcal{E}_l \cap \bigcap_{d\in\mathcal{U}\setminus\overline{\mathcal{U}}} \overline{Y_d}) \setminus \bigcup_{d\in\overline{\mathcal{U}}} \overline{Y_d}$. Corresponding to each such set, we consider the number b_i of items the leader selects from this set, and enumerate possible values for this number. Each b_i can attain $\mathcal{O}(b)$ many different values, resulting in $\mathcal{O}(b^{2^{|\mathcal{U}|}})$ cases that have to be checked. For fixed values b_i, we check $\mathcal{O}(|\mathcal{U}|)$ conditions on sums of these values and, if they are satisfied, we solve $\mathcal{O}(2^{|\mathcal{U}|})$ many single-level Selection Problems, each of them in time $\mathcal{O}(n)$. In order to evaluate the leader's objective function value of such a solution, we can compute its leader's costs in all scenarios in $\mathcal{O}(|\mathcal{U}|n)$.

In summary, the described algorithm gives the following result:

Theorem 3.63. *The Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with a discrete uncertainty set $\mathcal{U}$ of constant size $|\mathcal{U}|$, can be solved in time polynomial in n.*

This result demonstrates another difference between typical complexity results for robust combinatorial optimization problems and our setting of robust bilevel problems: On the one hand, it is typical in robust optimization that a general discrete uncertainty set leads to a strongly $\mathcal{NP}$-hard problem, which is in line with Theorem 3.58. On the other hand, a discrete uncertainty set with a constant number of scenarios often leads to a weakly $\mathcal{NP}$-hard problem that is solvable in pseudopolynomial time; see also Section 2.3.2.1. For the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), the latter situation turns out to be easier than that because, according to Theorem 3.63, a constant number of scenarios even renders the problem polynomial-time solvable.

Remark 3.64. Note that the approach leading to Theorem 3.63 is related to the idea of the Response Problem for the Bilevel Minimum Spanning Tree Problem; see Sections 5.5, 5.6, and 6.3.4. The questions that these types of problems, which can be defined for any bilevel optimization problem, ask are: Can the leader enforce a given follower's response? If so, what is an optimal way to do so in terms of the leader's objective? If, for some bilevel optimization problem, these questions can be answered relatively easily and, in addition, the number of potential follower's responses is relatively small, a promising solution approach for the bilevel problem could be to enumerate the potential follower's responses and solve a response problem for each of them.

Here, we enumerate follower's prefixes instead of precise follower's solutions, where parts of these prefixes might also be selected by the leader. This has the advantage that the number of follower's prefixes is much smaller than the number of potential follower's solutions. In fact, the number of potential follower's solutions can be exponential, in every single scenario, independently of the number of scenarios. Hence, the algorithm enumerating follower's solutions would not give a polynomial running time in case of constant $|\mathcal{U}|$.

Moreover, the approach developed for Theorem 3.63 extends the general idea of solving response problems to the robust setting. The robust version of a response problem asks: Can the leader enforce a given combination of follower's responses for all scenarios? This question (and its optimization version) can be answered in polynomial time in case of a constant number of scenarios here, by solving several Selection Problems.

3.4.2.1.2 Interval Uncertainty In case of interval uncertainty, it is still an open question whether the Robust Bilevel Selection Problem under Assumptions (sub) and (bb) is $\mathcal{NP}$-hard. We conjecture that this is the case, as in the setting of discrete uncertainty; see Theorem 3.58. In view of the algorithmic results in Section 3.4.1.1, interval uncertainty does not seem to be easier than discrete uncertainty for the Robust Bilevel Selection Problem. Also the more general complexity results in Section 2.4.2 indicate that interval uncertainty is in some sense harder than discrete uncertainty in our robust bilevel setting.

Like for discrete uncertainty, we derive the following approximation result:

Corollary 3.65. *The minimization version of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with interval uncertainty, can be 2-approximated in time* $\mathcal{O}(n^2 \log n)$.

Proof. The existence of a polynomial-time 2-approximation algorithm directly follows from Theorems 3.34 and 3.56. Analogously to Corollary 3.35, it can be implemented in a faster running time of $\mathcal{O}(n^2 \log n)$ by making use of a preprocessing. Similarly to the proof of Corollary 3.60, the precomputed sets the follower's solutions are derived from might be affected by the different leader's solutions here and therefore need to be updated in each iteration. □

Moreover, like in the setting of discrete uncertainty, we give an example that shows that the approximation factor of 2 is tight for the given algorithm, but it remains open whether a better approximation (or even an exact polynomial-time algorithm) is possible.

Example 3.66. Consider the family of instances from Example 3.61 again, but instead of the discrete uncertainty set $\mathcal{U} = \{d_1, d_2\}$, we now work with an interval uncertainty set $\mathcal{U}$ according to which the following three follower's greedy orders are possible:

$e_4, \dots, e_{n-1}, e_2, e_n, e_3, e_1$, or $e_4, \dots, e_{n-1}, e_n, e_2, e_3, e_1$, or $e_4, \dots, e_{n-1}, e_n, e_3, e_2, e_1$. This can be achieved by fixing the follower's item costs of all items except for e_2 (i.e., using intervals of length 0) and defining an appropriate interval for the cost of e_2. Analogously to Example 3.66, it can be easily checked that the algorithm returns either $X_1 = \{e_1\}$ or $X_2 = \{e_1, e_2\}$ as a leader's solution, each achieving a leader's objective function value of $2 - \varepsilon$, while the optimum leader's solution $X^* = \{e_2\}$ leads to a cost value of only 1.

Like for discrete uncertainty, we suggest an algorithm with exponential running time for the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with interval uncertainty. From Theorems 3.34 and 3.57, we get:

Corollary 3.67. *The Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with interval uncertainty, can be solved in time* $\mathcal{O}(\min\{2^{n_l}, n_l^b\}n^2)$.

After stating these straightforward results for interval uncertainty, we will see in the following section that the analogous ones can also be derived for discrete uncorrelated uncertainty.

3.4.2.1.3 Discrete Uncorrelated Uncertainty

In Theorem 3.37, we have seen that, under Assumption (bb), the Robust Bilevel Selection Problem with discrete uncorrelated uncertainty can be linearly reduced to the one with interval uncertainty. As this is independent of the Assumptions (sub) and (disj), it holds also here and implies that the algorithmic results of Section 3.4.2.1.2 also apply to the case of discrete uncorrelated uncertainty. In particular, the following two results follow from Corollaries 3.65 and 3.67:

Corollary 3.68. *The minimization version of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncorrelated uncertainty, can be 2-approximated in time* $\mathcal{O}(n^2 \log n)$.

Corollary 3.69. *The Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncorrelated uncertainty, can be solved in time* $\mathcal{O}(\min\{2^{n_l}, n_l^b\}n^2)$.

This finishes our considerations of the case of Assumptions (sub) and (bb). It only remains to discuss the case of Assumptions (sub) and (cc), which is the purpose of the following section.

3.4.2.2 The Robust Bilevel Selection Problem Under Assumptions (sub) and (cc)

The case of Assumptions (sub) and (cc) appears to be the most complicated one among the four main cases considered in this chapter. It is not a focus of this thesis, and only a few observations and ideas about this setting will be discussed in the following.

The follower's problem of the Bilevel Selection Problem under Assumptions (sub) and (cc) has been briefly discussed before Lemma 3.14 already. It is equivalent to a Continuous Knapsack Problem, in contrast to other variants of the Bilevel Selection Problem, where the follower's problem is a Selection Problem or a Continuous Selection Problem. We will now give more details about this. The key situation that is only possible under Assumptions (sub) and (cc), is that the leader might have selected fractions of some items that are also controlled by the follower, i.e., the leader might set x_e, for some $e \in \mathcal{E}_l \cap \mathcal{E}_f$, to some fractional value $x_e \in (0, 1)$. Then the follower's choice of this

item is restricted to the range $y'_e \in [0, 1 - x_e]$; see the constraint $x_e + y'_e \leq 1 \; \forall e \in \mathcal{E}_l \cap \mathcal{E}_f$ in (BSP2). Therefore, given a (feasible) leader's solution with possibly distinct values of x_e for all $e \in \mathcal{E}_l \cap \mathcal{E}_f$, the follower's decision cannot be modeled as a Continuous Selection Problem anymore. However, it can be reformulated as a Continuous Knapsack Problem by scaling the variable ranges appropriately. More precisely, the follower's problem

$$\begin{aligned} \max_{y'} \quad & \sum_{e \in \mathcal{E}_f} d(e) y'_e \\ \text{s.t.} \quad & x_e + y'_e \leq 1 && \forall e \in \mathcal{E}_l \cap \mathcal{E}_f \\ & \sum_{e \in \mathcal{E}_l} x_e + \sum_{e \in \mathcal{E}_f} y'_e = b \\ & y' \in [0,1]^{\mathcal{E}_f} \end{aligned}$$

can be equivalently reformulated, using new variables $y'' \in [0,1]^{\mathcal{E}_f}$ whose intended meaning is that $y'_e = y''_e$ for all $e \in \mathcal{E}_f \setminus \mathcal{E}_l$ and $y'_e = (1 - x_e) y''_e$ for all $e \in \mathcal{E}_f \cap \mathcal{E}_l$ hold:

$$\begin{aligned} \max_{y''} \quad & \sum_{e \in \mathcal{E}_f \setminus \mathcal{E}_l} d(e) y''_e + \sum_{e \in \mathcal{E}_f \cap \mathcal{E}_l} d(e)(1 - x_e) y''_e \\ \text{s.t.} \quad & \sum_{e \in \mathcal{E}_f \setminus \mathcal{E}_l} y''_e + \sum_{e \in \mathcal{E}_f \cap \mathcal{E}_l} (1 - x_e) y''_e = b - \sum_{e \in \mathcal{E}_l} x_e \\ & y'' \in [0,1]^{\mathcal{E}_f} \end{aligned}$$

This problem has exactly the form of a Continuous Knapsack Problem and is therefore still solvable in polynomial time by a greedy algorithm; see Section 4.1. However, the slightly more complicated structure of the follower's problem might suffice to induce an increased complexity (in particular of the adversary's problem) in case of Assumptions (sub) and (cc), compared to the problem variant of Assumptions (disj) and (cc) where the follower's problem is a Continuous Selection Problem. This will be discussed in the following.

Since the follower's problem of the Bilevel Selection Problem (and hence of the Robust Bilevel Selection Problem) under Assumptions (sub) and (cc) is the same as the one of the Robust Bilevel Continuous Knapsack Problem, also the corresponding adversary's problems (where the given leader's solution is again part of the instance) can be seen as equivalent. In both cases, the adversary can influence, based on the uncertainty set, the follower's objective and therefore the follower's greedy order that he uses when solving a Continuous Knapsack Problem. Thus, the (algorithmic) results that will be presented in Chapter 4 for the adversary's problem of the Robust Bilevel Continuous Knapsack Problem can be transferred to the adversary's problem of the Robust Bilevel Selection Problem under Assumptions (sub) and (cc). In particular, this concerns the polynomial-time algorithms for discrete uncertainty and interval uncertainty (see Theorems 4.12 and 4.19) and the pseudopolynomial-time algorithm for discrete uncorrelated uncertainty (see Theorem 4.30). Note that, in the latter case, we have to scale the problem in order to obtain integral item sizes. Therefore, the running time of the resulting algorithm for the adversary's problem of the Bilevel Selection Problem under Assumptions (sub) and (cc), and with discrete uncorrelated uncertainty, depends on the fractions that appear in the given leader's solution. The $\mathcal{NP}$-hardness result for discrete uncorrelated uncertainty (see Corollary 4.29) might be transferable to the adversary's problem of the Robust Bilevel Selection Problem under Assumptions (sub) and (cc)

as well, using a similar reduction as in the proofs of Theorems 4.22 and 4.28, with scaled problem parameters and an appropriately defined given leader's solution. This indicates that the setting of Assumptions (sub) and (cc) might be harder than the one of Assumptions (disj) and (cc), where a polynomial-time algorithm for the adversary's problem in case of discrete uncorrelated uncertainty has been presented in Algorithm 3.6 and Theorem 3.52.

Turning to the leader's problem of the Bilevel Selection Problem under Assumptions (sub) and (cc), the situation becomes much more complicated because each of the two assumptions brings its own difficulties, compared to Assumptions (disj) and (bb), as we discuss in the following.

Assumption (sub), in contrast to Assumption (disj), complicates the interaction of leader and follower, for which not only the number of selected leader's items is relevant, but also the leader's specific decision on the items controlled by both players. For the setting of Assumptions (sub) and (bb), we have seen in Section 3.4.2.1 that greedy leader's selections are not optimum anymore in case of robustness (see, e.g., Example 3.61), and that the problem with discrete uncertainty even becomes $\mathcal{NP}$-hard (Theorem 3.58). Example 3.70 below will demonstrate that greedy leader's selections are not optimum for the setting of Assumptions (sub) and (cc) either.

The complications of Assumption (cc) include that combinatorial considerations and enumeration ideas are not valid, in contrast to the setting of Assumption (bb). For example, an infinite number of possible ways to split the capacity into leader's and follower's capacity has to be considered, even for Assumption (disj). In case of Assumptions (disj) and (cc), we could deal with that by considering piecewise linear functions in a single variable, which describes this splitting of the total capacity, being the only interaction between leader and follower (see Section 3.4.1.2). This is possible because an optimum leader's solution is completely determined once the capacity b_l to be used by the leader is fixed. Indeed, the best way to use a given capacity is a greedy choice, i.e., selecting a fractional prefix, of the corresponding cardinality, of the leader's greedy order of her items (see Section 3.3.2 and Definition 3.45). A fractional prefix contains at most one fractional item. In contrast to that, Example 3.70 demonstrates that the leader might need to select several items fractionally in the setting of Assumptions (sub) and (cc). Therefore, optimum leader's solutions seem to have a more complicated structure here, which might not be describable by a single parameter. In particular, having several fractional items in the (unique) optimum leader's solution implies that it cannot be described as a fractional prefix of any order of the leader's items.

Example 3.70. Consider the following instance of the Robust Bilevel Selection Problem under Assumptions (sub) and (cc), and with discrete uncertainty: Let $\mathcal{E}_l = \{e_1, e_2\}$, $\mathcal{E}_f = \{e_1, e_2, e_3, e_4, e_5\}$, $b = 2$, and let the leader's item values be defined as $c(e_1) = c(e_2) = 2$, $c(e_3) = 4$, $c(e_4) = 3$, and $c(e_5) = 0$. The uncertainty set $\mathcal{U} = \{d_1, d_2, d_3\}$ is given by the following follower's greedy orders in the three scenarios (recall Remark 3.2 for the fact that it suffices to define the scenarios like this): In scenario d_1, the follower orders his items as e_1, e_4, e_2, e_3, e_5, in scenario d_2, the order is e_2, e_4, e_1, e_3, e_5, and in scenario d_3, it is e_3, e_5, e_1, e_2, e_4.

We claim that, in this instance, the unique optimum leader's solution selects a proper fraction of e_1 and a proper fraction of e_2, i.e., $x_1 \in (0, 1)$ and $x_2 \in (0, 1)$.

In order to determine the optimum solution, we first consider the case that the leader

selects some fraction $x_1 \in [0,1]$ of item e_1 and nothing of item e_2, i.e., $x_2 = 0$. Then, in scenario d_1, the joint solution consists of e_1 and e_4, hence the total leader's item value is 5. In scenario d_2, the follower sets $y_2 = 1$ and $y_4 = 1 - x_1$ (and all other y variables to 0), leading to a total leader's item value of $2x_1 + 2 + 3(1 - x_1) = 5 - x_1$. In scenario d_3, the follower sets $y_3 = 1$ and $y_5 = 1 - x_1$, giving a total leader's item value of $4 + 2x_1$. It can be easily checked that the minimum of these three values is maximum for $x_1 = 1/3$, where $5 - x_1 = 4 + 2x_1 = 4\,2/3$. Thus, the best leader's solution with $x_2 = 0$ is the one with $x_1 = 1/3$ and value $4\,2/3$.

Finally, we investigate the general case where $x_1 \in [0,1]$ and $x_2 \in [0,1]$. Similarly to above, scenario d_2 gives a total leader's item value of $5 - x_1$, and, analogously, scenario d_1 gives a value of $5 - x_2$. Note that, if $x_1 + x_2 > 1$, then $\max\{x_1, x_2\} > 1/2$, hence the worst of these two cases, with a value of less than $4\,1/2$, is already worse than the value of $4\,2/3$ achieved above. Therefore, we may assume $x_1 + x_2 \leq 1$ from now on. For scenario d_3, this means that the follower sets $y_3 = 1$ and $y_5 = 1 - x_1 - x_2$, and this gives a total leader's item value of $4 + 2x_1 + 2x_2$. Again, it can be easily checked for which x_1 and x_2 the minimum of the three values from the three scenarios is maximum: This happens for $x_1 = x_2 = 1/5$, where $4 + 2x_1 + 2x_2 = 5 - x_1 = 5 - x_2 = 4\,4/5$, which is the best value that the leader can achieve.

Due to the above observations, we cannot expect that the Robust Bilevel Selection Problem under Assumptions (sub) and (cc) can be solved in a similar fashion as the Robust Bilevel Selection Problem under Assumptions (disj) and (cc). It might still be possible to represent the leader's objective function value, for any given leader's solution $x \in [0,1]^{\mathcal{E}_l}$, as a (higher-dimensional) piecewise linear function. However, such a function cannot be expected to be computable in polynomial time. If a polynomial-time algorithm for the Robust Bilevel Selection Problem under Assumptions (sub) and (cc) can be achieved, it will require new insights into the structure of some set of potentially optimum leader's solutions. The complexity of the Robust Bilevel Selection Problem under Assumptions (sub) and (cc) remains an open question, for all types of uncertainty sets. Note that it is not even clear that the leader's problem is at least as hard as the adversary's problem, i.e., an $\mathcal{NP}$-hard adversary's problem would not necessarily imply that the leader's problem is also $\mathcal{NP}$-hard because it could still be possible to restrict the structure of potential optimum leader's solutions enough to allow for a polynomial-time algorithm. On the other hand, observe that the leader's problem is $\mathcal{NP}$-easy if the adversary's problem can be solved in polynomial time, in particular in the cases of discrete and interval uncertainty, since the leader's problem can be at most one level harder than the adversary's problem; see also Section 2.4.2 and, in particular, Theorem 2.39.

Example 3.70 shows that, in order to find an optimum leader's solution, it does not suffice to consider greedy leader's choices for different capacities, in contrast to the settings of Assumption (disj), together with either of the Assumptions (bb) and (cc); see Theorem 3.25 and Section 3.4.1.2. In case of Assumptions (sub) and (bb), such an algorithm turned out to be suboptimal as well, but could be shown to be a 2-approximation algorithm, for the minimization version of the Robust Bilevel Selection Problem; see Theorem 3.56. One could try to prove a similar result for the setting of Assumptions (sub) and (cc). This would consider an algorithm that computes the sum of the piecewise linear function corresponding to greedy leader's choices and the function representing the worst-case leader's item values of the resulting follower's reactions, as described in the beginning

of Section 3.4.1.2. First, it would be necessary to investigate the structure of the latter function, depending on the type of uncertainty set, in order to compute it in polynomial time. This might be similar to the algorithms presented in Sections 3.4.1.2 and 4.4, but one has to take care of follower's items that have been selected by the leader already, depending on the current greedy leader's solution. Second, one could study whether any approximation guarantee for this type of algorithm can be proved, e.g., by adapting the proof of Theorem 3.56.

3.5 Conclusion

In this chapter, we have studied the computational complexity of robust bilevel optimization at the example of the polynomial-time solvable Bilevel Selection Problem, for several problem variants. The perspective is mainly a combinatorial one, using the fundamental Selection Problem as a basis, which is often considered when investigating the complexity of robust optimization concepts. For the Robust Bilevel Selection Problem under Assumptions (disj) and (bb), i.e., with leader's and follower's item sets being disjoint, and with binary decision variables, we have established polynomial-time algorithms for the cases of discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty; see Section 3.4.1.1.

Besides this purely combinatorial setting, we have also studied the Robust Bilevel Selection Problem under Assumption (cc), where the decision variables are continuous. The results and algorithmic ideas for the setting of Assumptions (disj) and (cc) are similar to the ones used for Assumptions (disj) and (bb), but additional effort was required in some aspects; see Section 3.4.1.2. In particular, piecewise linear functions have been utilized instead of simply enumerating possible splittings of the common capacity into a leader's and a follower's capacity, and an adapted approach was necessary for discrete uncorrelated uncertainty, which is not equivalent to interval uncertainty in this setting, in contrast to the one of Assumption (bb).

In Chapter 4, we will consider a closely related robust bilevel optimization problem, the Robust Bilevel Continuous Knapsack Problem. Several of the insights from this chapter, in particular from Section 3.4.1.2 about the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), will be generalized to this problem. However, we will also see some differences. In particular, in case of discrete uncorrelated uncertainty, the leader's as well as the adversary's problem will be shown to be $\mathcal{NP}$-hard in case of the Robust Bilevel Continuous Knapsack Problem (see Section 4.4.3.1). The polynomial-time algorithm for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) will be generalized to a pseudopolynomial-time algorithm for the Robust Bilevel Continuous Knapsack Problem (see Section 4.4.3.2), which shows that the problem is in fact only weakly $\mathcal{NP}$-hard, but not strongly $\mathcal{NP}$-hard.

Also the Bilevel Minimum Spanning Tree Problem studied in Chapter 5 is related to the Bilevel Selection Problem (under Assumption (bb)); see Section 6.1 for more details.

For the more general Robust Bilevel Selection Problem under Assumption (sub), in which the leader's and the follower's item sets are not necessarily disjoint, we have shown that the Robust Bilevel Selection Problem is strongly $\mathcal{NP}$-hard in general, but can be 2-approximated in polynomial time. Moreover, we investigated exponential-time

solution approaches for this problem, including an algorithm that runs in polynomial time if the number of scenarios in the uncertainty set is a constant. These results have been presented in Section 3.4.2.1.

Another interesting observation is that the Robust Bilevel Selection Problem under Assumption (cb), i.e., with continuous leader's variables and binary follower's variables, yields a bilevel optimization problem whose optimum value may not be attained (see Lemma 3.12). This demonstrates that, in general, bilevel optimization problems containing integer and continuous variables should be handled with care in this respect.

Open questions arising from this chapter will be discussed in Section 6.3.1.

4 The Bilevel Continuous Knapsack Problem

In this chapter, we will generalize some of the results obtained in Chapter 3 for variants of the Robust Bilevel Selection Problem to a knapsack version of the problem. In contrast to a Selection Problem, in a Knapsack Problem, the items can have different sizes and, instead of the total number of items we select, their total size has to fit a given capacity. In Chapter 3, we distinguished several different variants of the Bilevel Selection Problem. In particular, we considered the Assumptions (bb) and (cc) regarding binary or continuous decision variables. We will see that the corresponding binary variant of the Bilevel Knapsack Problem is already known to be Σ_2^{p}-hard, even without uncertainty; see Section 4.2.3 for more details. On the other hand, the Bilevel Continuous Knapsack Problem is solvable in polynomial time; see Section 4.3. Following the idea of Chapter 3 to investigate the additional complexity that robustness adds to (easy) bilevel optimization problems, we will then focus on the Robust Bilevel Continuous Knapsack Problem for different types of uncertainty sets in Section 4.4. The other main distinction of problem variants that we made in Chapter 3 concerns the item sets controlled by the leader and by the follower. We have seen that the Bilevel Selection Problem is easier to understand, and in some aspects also easier regarding the complexity, under Assumption (disj), i.e., when the two item sets are disjoint, compared to the more general Assumption (sub), where the follower controls all items and the leader controls a subset of them. In particular, our understanding of the Robust Bilevel Selection Problem under Assumptions (sub) and (cc) is still limited; see Section 3.4.2.2. Therefore, we will focus on Assumption (disj) for the more general Robust Bilevel Continuous Knapsack Problem in this chapter. For a more detailed discussion of the different assumptions for the Bilevel Knapsack Problem and the relations to the Bilevel Selection Problem, we refer to Section 4.2.2.

After introducing the single-level Knapsack Problem in Section 4.1 and bilevel variants of it in Section 4.2, we will show in Section 4.3 how the Bilevel Continuous Knapsack Problem can be solved in polynomial time. Section 4.4 then deals with the Robust Bilevel Continuous Knapsack Problem for different types of uncertainty sets. For discrete uncertainty and interval uncertainty, we generalize the results obtained in Sections 3.4.1.2.1 and 3.4.1.2.2 for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc); see Sections 4.4.1 and 4.4.2. In case of discrete uncorrelated uncertainty, we obtain an $\mathcal{NP}$-hardness result in Section 4.4.3.1, in contrast to the Robust Bilevel Selection Problem, where the corresponding case was solvable in polynomial time (see Section 3.4.1.2.3). However, we extend the polynomial-time algorithm for the Robust Bilevel Selection Problem to obtain a pseudopolynomial-time algorithm for the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty; see Section 4.4.3.2. Also for simplicial uncertainty and norm uncertainty, we prove that the Robust Bilevel Continuous Knapsack Problem is $\mathcal{NP}$-hard; see Sections 4.4.4 and 4.4.5. Finally, we conclude the chapter in Section 4.5.

As mentioned in Chapter 1, most of the results presented in this chapter have already appeared in [BH22] in a slightly restricted form, and a variant of the pseudopolynomial-time algorithms for the setting of discrete uncorrelated uncertainty (in Section 4.4.3.2) is contained in [Hen25]. The differences between the results in this chapter and the ones in [BH22; Hen25] will be discussed in more detail in Sections 4.2.2 and 4.4.

4.1 The Knapsack Problem

We first introduce the (single-level) Knapsack Problem, a very basic combinatorial optimization problem in the literature. This section is mostly based on [KV18, Chapter 17], and we also refer to [KPP04].

Knapsack Problem
Instance: A finite set $\mathcal{E}$ of items, item sizes $a\colon \mathcal{E} \to \mathbb{Q}_{>0}$, a number $b \in \mathbb{Q} \cap [0, a(\mathcal{E})]$, and item values $c\colon \mathcal{E} \to \mathbb{Q}$
Task: Find a subset $X \subseteq \mathcal{E}$ such that $a(X) \leq b$ and $c(X)$ is maximum.

The Knapsack Problem is weakly $\mathcal{NP}$-hard [Kar72], but solvable in pseudopolynomial time, using the idea of dynamic programming. More precisely, there are algorithms for it running in $\mathcal{O}(nb)$ or $\mathcal{O}(n\,c(\mathcal{E}))$ time, respectively, assuming that all item sizes a and item values c as well as the capacity b are integers. For more details, we refer to, e.g., [KV18, Chapter 17.2].

Observe that the special case of the Knapsack Problem in which $a(e) = 1$ holds for all $e \in \mathcal{E}$ is exactly the variant of the Selection Problem in which the cardinality constraint is given by the inequality $|X| \leq b$. As discussed in Section 3.1, this variant of the Selection Problem is very similar to our original variant of the Selection Problem with the constraint $|X| = b$. In particular, even if it is not required to fill the capacity completely, it will be filled completely in an optimum solution unless there are not enough items with positive value. In case of the Knapsack Problem, the distinction between the classical variant with the constraint $a(X) \leq b$ and the alternative variant $a(X) = b$ is more significant. In fact, in the latter case, even deciding if there is a feasible solution is an $\mathcal{NP}$-complete problem, namely the Subset Sum Problem (see below), while the trivial solution $X = \emptyset$ is always feasible in the classical variant. However, also the alternative problem variant with $a(X) = b$ is solvable in pseudopolynomial time using a slightly modified dynamic programming approach. In this chapter, we will usually consider the problem version with an equality constraint. In fact, we will mostly deal with the Continuous Knapsack Problem (see below) instead of the Knapsack Problem, where there is again not a significant difference between the two versions of the capacity constraint, and the equality version is sometimes easier to handle in our arguments.

We now define a decision problem that is closely related to the Knapsack Problem:

Subset Sum Problem
Instance: Numbers $m \in \mathbb{N}$ and $w_1, \dots, w_m, W \in \mathbb{N}$
Task: Decide if there is a subset $S \subseteq \{1, \dots, m\}$ with $\sum_{i \in S} w_i = W$.

The Subset Sum Problem is weakly $\mathcal{NP}$-complete [Kar72] and we will use it for several hardness proofs in this chapter; see Theorems 4.22, 4.34, and 4.42.

Like for the Selection Problem in Section 3.1, we can also look at the continuous

version of the Knapsack Problem. This version forms the foundation of the bilevel optimization problem that we will investigate throughout this chapter. As mentioned above, we will later concentrate on problems with an equality constraint. Therefore, we define the Continuous Knapsack Problem as follows:

Continuous Knapsack Problem

Instance: A finite set $\mathcal{E}$ of items, item sizes $a\colon \mathcal{E} \to \mathbb{Q}_{>0}$, a number $b \in \mathbb{Q} \cap [0, a(\mathcal{E})]$, and item values $c\colon \mathcal{E} \to \mathbb{Q}$

Task: Find a vector $x \in [0,1]^{\mathcal{E}}$ such that $\sum_{e\in\mathcal{E}} a(e)x_e = b$ and $\sum_{e\in\mathcal{E}} c(e)x_e$ is maximum.

In contrast to the binary problem version, the Continuous Knapsack Problem is clearly solvable in polynomial time as a linear program. As an easier approach, the greedy idea presented for the Selection Problem and the Continuous Selection Problem in Section 3.1 also works for the Continuous Knapsack Problem. The algorithm here is due to Dantzig and first appeared in [Dan57]. We sort the items by their *profits*, which are given as the ratios $c(e)/a(e)$, for $e \in \mathcal{E}$. Let $e_1, \dots, e_n$ denote the items in $\mathcal{E}$ in this order, i.e., such that $\frac{c(e_i)}{a(e_i)} \geq \frac{c(e_{i+1})}{a(e_{i+1})}$ for all $i \in \{1, \dots, n-1\}$. Again, the idea is to pack the items into the knapsack in this order until it is full. More precisely, if $b = a(\mathcal{E})$, all items have to be selected, so the optimum solution is given by $x_e = 1$ for all $e \in \mathcal{E}$. Otherwise, we consider the critical item $e_k \in \mathcal{E}$ with $k = \min\{j \in \{1, \dots, n\}\colon \sum_{i=1}^{j} a(e_i) > b\}$. An optimum solution is then given by

$$x_{e_j} = \begin{cases} 1 & \text{if } j \in \{1, \dots, k-1\}, \\ \frac{1}{a(e_k)} \left(b - \sum_{i=1}^{k-1} a(e_i)\right) & \text{if } j = k, \\ 0 & \text{if } j \in \{k+1, \dots, n\}. \end{cases}$$

As in Section 3.1, this algorithm runs in time $\mathcal{O}(n \log n)$, where $n = |\mathcal{E}|$, and can be improved to run in time $\mathcal{O}(n)$ by using the weighted median approach; see also [KV18, Chapter 17.1]. However, we will again work with explicitly sorting the items in this chapter because we will often solve the Continuous Knapsack Problem on the same set of items but according to different capacities; see, e.g., Algorithm 4.1.

Since the Continuous Knapsack Problem is very similar to the Continuous Selection Problem with its greedy structure, it can be easily seen that the observations about the two problem variants with an equality or an inequality constraint that we made in Section 3.1 for the Continuous Selection Problem can be transferred to the Continuous Knapsack Problem. In particular, we can analogously reduce the Continuous Knapsack Problem with an inequality constraint to the one with an equality constraint by restricting the item set to $\mathcal{E}_{\geq 0} := \{e \in \mathcal{E} \mid c(e) \geq 0\}$ and possibly decreasing the capacity to $\min\{b, a(\mathcal{E}_{\geq 0})\}$, or, alternatively, by adding a dummy item e with an item value $c(e) = 0$ and an item size $a(e) = b - a(\mathcal{E}_{\geq 0})$ to $\mathcal{E}$ in case $a(\mathcal{E}_{\geq 0}) < b$. For the reduction in the other direction, we now add $Ma(e)$ to the item value $c(e)$, for all $e \in \mathcal{E}$ and some large enough $M \in \mathbb{Q}_{\geq 0}$ such that all item values become nonnegative. Then, all profits $c(e)/a(e)$ change by the same value M and therefore the greedy order does not change. Note that similar arguments will be applied in the bilevel setting in Lemma 4.6.

Note that we define the Knapsack Problem and the Continuous Knapsack Problem (as well as their bilevel variants below) with arbitrary rational item values c, while nonnegative (and possibly integer) item values might be more usual. We focus on this

most general definition here in accordance with the variants of the Selection Problem that we discussed in Chapter 3. Moreover, it will sometimes be more convenient to allow arbitrary rational item values; see, e.g., Remark 4.5 and Algorithm 4.2. Observe that, by the construction described in the previous paragraph, nonnegative item values can be assumed without loss of generality for the Continuous Knapsack Problem. In addition, if all item values are nonnegative, the capacity b can always be filled completely in an optimum solution, even in case of an inequality constraint.

Remark 4.1. Like for the Selection Problem and the Continuous Selection Problem (see Remark 3.1), the greedy approach suggests that optimum solutions might not be unique if the order of the items that is computed in the algorithm, and in particular the critical item, is not unique, also in case of the Continuous Knapsack Problem. This will be relevant again when we discuss bilevel optimization problems in which the follower's problem is a Continuous Knapsack Problem and we need to adopt the optimistic or the pessimistic approach; see Section 4.2.1. Then, like for the Continuous Selection Problem, we will always assume also for the Continuous Knapsack Problem that it is solved using Dantzig's algorithm with some fixed order of the item profits. In particular, this implies that all returned solutions contain at most one nonbinary variable; see also Remark 3.1.

4.2 Definition and Variants of the Bilevel Knapsack Problem

As generalizations of the Bilevel Selection Problem under Assumptions (bb) and (cc) (see Section 3.2), and as bilevel variants of the Knapsack Problem and the Continuous Knapsack Problem (see Section 4.1), we will now introduce the Bilevel Knapsack Problem and the Bilevel Continuous Knapsack Problem. Moreover, we will discuss basic properties of the problems (see Section 4.2.1) as well as different problem variants (see Section 4.2.2). The latter cover the equivalents of the assumptions that we discussed for the Bilevel Selection Problem in Sections 3.2.2, 3.2.3, 3.2.4, and 3.2.5. In Section 4.2.3, we give some context by reviewing literature related to variants of the Bilevel Knapsack Problem, focusing on complexity results.

The basic problem variant of our Bilevel Knapsack Problem, which is a direct generalization of the basic Bilevel Selection Problem (see Section 3.2.1), is defined in the following. In fact, if $a(e) = 1$ holds for all $e \in \mathcal{E}_l \cup \mathcal{E}_f$ and b is an integer in the Bilevel Knapsack Problem, we obtain the Bilevel Selection Problem.

Bilevel Knapsack Problem

Instance: Finite leader's and follower's item sets $\mathcal{E}_l$ and $\mathcal{E}_f$, respectively, item sizes $a\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}_{>0}$, a number $b \in \mathbb{Q} \cap [0, a(\mathcal{E}_l \cup \mathcal{E}_f)]$, and leader's and follower's item values $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$ and $d\colon \mathcal{E}_f \to \mathbb{Q}$, respectively

Task: Find a subset $X \subseteq \mathcal{E}_l$ for the leader such that $c(X \cup Y)$ is maximum, where $Y \subseteq \mathcal{E}_f \setminus X$ is a follower's subset that satisfies $a(X \cup Y) = b$ and maximizes $d(Y)$.

Notation. As in Chapter 3, we denote the cardinalities of the item sets by $n_l = |\mathcal{E}_l|$, $n_f = |\mathcal{E}_f|$, and $n = |\mathcal{E}_l \cup \mathcal{E}_f|$ in this chapter.

Again, we can write the Bilevel Knapsack Problem as a bilevel optimization problem of type (L) (see Section 2.2.1), analogously to (BSP1) in Section 3.2.1:

$$\begin{aligned} &\max_{X} \; c(X \cup Y) \\ &\text{s.t. } X \subseteq \mathcal{E}_l \\ &\quad Y \in \operatorname*{argmax}_{Y'} \; d(Y') \\ &\qquad \text{s.t. } Y' \subseteq \mathcal{E}_f \setminus X \\ &\qquad\quad a(X \cup Y') = b \end{aligned} \tag{BKP}$$

Recall that, in Sections 3.2.2, 3.2.3, and 3.2.4, we introduced several assumptions one can make about the Bilevel Selection Problem: the Assumptions (arb), (sub), (disj), and (uni) about the item sets, the Assumptions (bb), (cc), (bc), and (cb) about binary and continuous decisions, and the Assumptions (=) and (≤) about the capacity constraint. All these assumptions can be equivalently applied also to the Bilevel Knapsack Problem, with only the following two adjustments needed.

First, the capacity constraints in the definitions of Assumptions (=) and (≤) now need to include the item sizes a, leading to the following modified assumptions:
(=′) $\sum_{e\in\mathcal{E}_l} a(e)x_e + \sum_{e\in\mathcal{E}_f} a(e)y'_e = b$
(≤′) $\sum_{e\in\mathcal{E}_l} a(e)x_e + \sum_{e\in\mathcal{E}_f} a(e)y'_e \leq b$

Second, we change the definition of Assumption (uni). Recall that, for the Bilevel Selection Problem, Assumption (uni) is a special case of Assumption (disj) (where the leader's and the follower's item sets are assumed to be disjoint), which additionally requires the item values of all leader's items to be the same. The main motivation to introduce this assumption was its equivalence to the alternative bilevel variant (BSP-b_f) of the Selection Problem; see Section 3.2.2. A similar equivalence for the setting of the Bilevel Continuous Knapsack Problem will be presented in Section 4.2.2. In order to follow the same idea as for the Bilevel Selection Problem, we now have to require all leader's items to have the same leader's value γ *per size unit* instead of the same absolute leader's value. This gives the following modified assumption:
(uni′) $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$ and $c(e) = \gamma a(e)$ for all $e \in \mathcal{E}_l$, where $\gamma \in \mathbb{Q}$
Note that, for the Bilevel Selection Problem, as the special case of $a(e) = 1$ for all $e \in \mathcal{E}_l \cup \mathcal{E}_f$, the definition of Assumption (uni′) is equivalent to the original one of Assumption (uni). However, in case of the general Bilevel Knapsack Problem, there is a difference and Assumption (uni′) is the one that models all leader's items $e \in \mathcal{E}_l$ being equally valuable in terms of capacity usage, i.e., having the same profit $c(e)/a(e) = \gamma$. This is the desired property in view of the equivalent formulation (BCKP-b_f) below; see Section 4.2.2.

In this chapter, we will not investigate all problem variants arising from the different assumptions in such detail as in Chapter 3. However, we will briefly explain the most important issues related to them for the Bilevel Knapsack Problem in Sections 4.2.2 and 4.2.3, before we focus on the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (=′) for the remainder of the chapter. We denote this problem variant as the Bilevel Continuous Knapsack Problem. We will see in Section 4.3 that it is solvable in polynomial time (while other problem variants are $\mathcal{NP}$- or even Σ_2^{p}-hard; see Section 4.2.3), which makes it a good starting point for investigating the complexity of robust versions of it, like in the case of the Bilevel Selection Problem in Chapter 3.

Analogously to the Bilevel Selection Problem under Assumptions (cc) and (=), which corresponds to the problem formulation (BSP2) in Section 3.2.1 with continuous

instead of binary variables, we can write the Bilevel Continuous Knapsack Problem (i.e., the variant of the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (=′)) as a bilevel linear program as follows.

$$\begin{aligned} \max_{x} \ & \sum_{e\in\mathcal{E}_l} c(e)x_e + \sum_{e\in\mathcal{E}_f} c(e)y_e \\ \text{s.t. } & x \in [0,1]^{\mathcal{E}_l} \\ & y \in \operatorname*{argmax}_{y'} \sum_{e\in\mathcal{E}_f} d(e)y'_e \\ & \qquad \text{s.t. } \sum_{e\in\mathcal{E}_l} a(e)x_e + \sum_{e\in\mathcal{E}_f} a(e)y'_e = b \\ & \qquad\quad y' \in [0,1]^{\mathcal{E}_f} \end{aligned} \tag{BCKP}$$

Note that the follower's constraint

$$x_e + y'_e \leq 1 \quad \forall e \in \mathcal{E}_l \cap \mathcal{E}_f$$

that appeared in (BSP2) is not required in case of Assumption (disj), where $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$.

4.2.1 Basic Properties of the Bilevel Continuous Knapsack Problem

The goal of this section is to give a better understanding of the Bilevel Continuous Knapsack Problem, i.e., the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (=′), which can be written as (BCKP) as above.

As usual in bilevel optimization (see Remark 2.22), the leader is required to ensure that the follower's problem is feasible; otherwise a leader's solution is considered infeasible. Here, this means that a feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$ has to satisfy $b - \sum_{e\in\mathcal{E}_l} a(e)x_e \in [0, a(\mathcal{E}_f)]$. More specifically, the leader may fill a capacity of at least $b_l^- = \max\{0, b - a(\mathcal{E}_f)\}$ and at most $b_l^+ = \min\{a(\mathcal{E}_l), b\}$, leading to a remaining capacity of at least $b_f^- = b - b_l^+ = \max\{0, b - a(\mathcal{E}_l)\}$ and at most $b_f^+ = b - b_l^- = \min\{a(\mathcal{E}_f), b\}$ for the follower. The assumption $b \in \mathbb{Q} \cap [0, a(\mathcal{E}_l \cup \mathcal{E}_f)]$ that we make for every instance in the problem definition ensures that there is always such a feasible leader's solution.

Given a fixed feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$, the follower's task is to solve a single-level Continuous Knapsack Problem on the item set $\mathcal{E}_f$ with respect to the item sizes given by a, the capacity $b - \sum_{e\in\mathcal{E}_l} a(e)x_e$, and the item values given by d. In particular, only the total size $\sum_{e\in\mathcal{E}_l} a(e)x_e$ of the leader's solution x, but not the solution itself is relevant for the follower, like in case of the Bilevel Selection Problem under Assumption (disj) (see, e.g., Theorem 3.18). As discussed in Section 4.1, this Continuous Knapsack Problem can be solved greedily by sorting the items according to their profits $d(e)/a(e)$, for $e \in \mathcal{E}_f$. This sorting and, accordingly, the follower's optimum solution might not be unique; see Remark 4.1. In this case, we distinguish the optimistic and the pessimistic setting, as introduced in Section 2.2.1. This here means that the leader's objective c, or its opposite, is taken into account as a secondary objective if there are several optimum solutions with respect to the follower's objective d. Similarly to the Bilevel Selection Problem (see Remark 3.2), this can be incorporated into the follower's greedy approach by using the leader's profits $c(e)/a(e)$, for $e \in \mathcal{E}_f$, as a secondary criterion when sorting the follower's items, i.e., an item $e_1 \in \mathcal{E}_f$ precedes an item $e_2 \in \mathcal{E}_f$ if either $\frac{d(e_1)}{a(e_1)} > \frac{d(e_2)}{a(e_2)}$

or $\frac{d(e_1)}{a(e_1)} = \frac{d(e_2)}{a(e_2)}$ and $\frac{c(e_1)}{a(e_1)} > \frac{c(e_2)}{a(e_2)}$ $\left[\frac{c(e_1)}{a(e_1)} < \frac{c(e_2)}{a(e_2)}\right]$ in the optimistic [pessimistic] setting. Note that, more generally, any fixed follower's greedy order leads to a well-defined bilevel problem. Therefore, we will usually assume that a fixed greedy order is given or defined; see also Remark 3.2 for more details.

The leader's task in the Bilevel Continuous Knapsack Problem is to select fractions $x \in [0,1]^{\mathcal{E}_l}$ of her items, while anticipating the follower's response $y \in [0,1]^{\mathcal{E}_f}$ and maximizing the total value c of all selected items. We will see in Algorithm 4.1 and Theorem 4.9 that this problem can be solved in polynomial time, generalizing the approach presented in Algorithm 3.1 and Theorem 3.18 for the Bilevel Selection Problem.

Remark 4.2. Note that, in contrast to the Bilevel Selection Problem (see Remark 3.9), we do not require the capacity b to be an integer in the definition of the Bilevel Knapsack Problem. This is in line with the item sizes a being arbitrary rational numbers as well. We could also assume a and b to be given as integers. In particular, this is reasonable in the context of pseudopolynomial-time algorithms, like for the Knapsack Problem (see Section 4.1). In fact, we present pseudopolynomial-time algorithms in Section 4.4.3.2 and assume the item sizes a to be integral there. However, we formulate the other algorithms and results in this chapter in terms of arbitrary rational inputs as we often have to deal with fractions of items and therefore rational numbers anyway.

4.2.2 Discussion of the Variants of the Bilevel Knapsack Problem

In this section, we demonstrate the relations between the variants of the Bilevel Knapsack Problem according to the Assumptions (bb), (cc), (bc), (cb), (arb), (sub), (disj), (uni′), (=′), and (≤′), similarly to Section 3.2.5 for the Bilevel Selection Problem. In particular, recall Figure 3.1 for an overview of the relations discussed in Section 3.2.5. However, we will not investigate as many technical details here as in Section 3.2.5 because we mainly aim for giving context to the variant that we will focus on in this chapter, namely the Bilevel Continuous Knapsack Problem, corresponding to the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (=′).

We start by discussing the Assumptions (bb), (cc), (bc), and (cb) about binary and continuous decisions. First observe that the follower's problem with binary variables corresponds to a single-level Knapsack Problem and is thus $\mathcal{NP}$-hard in general. The leader's problem is even Σ_2^{p}-hard in case of Assumptions (bb), (disj), and (≤′); see Section 4.2.3 below. On the other hand, the follower's problem is solvable in polynomial time as a Continuous Knapsack Problem in case of continuous follower's variables; see Section 4.2.1. We will see in Section 4.3 that also the leader's problem of the Bilevel Continuous Knapsack Problem can be solved in polynomial time. Accordingly, we do not obtain an equivalence between Assumptions (bb) and (cc) for the Bilevel Knapsack Problem, in contrast to the Bilevel Selection Problem (see Lemma 3.14). Moreover, it remains a task for future work to investigate the variants of the Bilevel Knapsack Problem corresponding to the Assumptions (bc) and (cb) and their complexity further. In particular, the arguments used to prove Lemmas 3.10 and 3.11 for the Bilevel Selection Problem do not carry over to the Bilevel Knapsack Problem. However, the issue that the Bilevel Selection Problem under Assumption (cb) does not necessarily admit an optimum solution (see Lemma 3.12) also applies to the Bilevel Knapsack Problem under Assumption (cb), the latter being a generalization of the former.

We next focus on the Assumptions (arb), (sub), (disj), and (uni′) about the leader's and the follower's item sets. In the Bilevel Continuous Knapsack Problem, we always make Assumption (disj), i.e., $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$. For some results in this chapter, also Assumption (uni′) will play a role, which corresponds to the special case of Assumption (disj) where all leader's items have the same profit (or we even have only one leader's item).

Recall that, in Section 3.2.2, we introduced the alternative bilevel variant (BSP-b_f) of the Selection Problem in which the leader does not control any items herself, but instead selects the capacity for the follower's Selection Problem. We have shown in Lemma 3.6 that this problem is equivalent to the Bilevel Selection Problem under Assumption (uni). We can now define a similar bilevel variant of the Knapsack Problem that is equivalent to the Bilevel Knapsack Problem under Assumption (uni′). We will demonstrate this for the Bilevel Continuous Knapsack Problem, but the same is possible for any variant of the Bilevel Knapsack Problem, like in case of the Bilevel Selection Problem (see Remark 3.7).

Formally, we define the alternative bilevel variant of the Continuous Knapsack Problem as follows. Like in the definition of the Bilevel Knapsack Problem above, we are given a finite set $\mathcal{E}_f$ of follower's items, item sizes $a\colon \mathcal{E}_f \to \mathbb{Q}_{>0}$, as well as leader's and follower's item values $c, d\colon \mathcal{E}_f \to \mathbb{Q}$, respectively. Instead of leader's items, we are now given bounds $b_f^-, b_f^+ \in \mathbb{Q} \cap [0, a(\mathcal{E}_f)]$ on the possible follower's capacity and a number $\gamma \in \mathbb{Q}$, which stands for the cost that the leader pays for every unit of capacity she provides to the follower. Given these data, we can state the problem as follows.

$$\begin{aligned} \max_{b_f} \ & -\gamma b_f + \sum_{e \in \mathcal{E}_f} c(e) y_e \\ \text{s.t. } & b_f \in [b_f^-, b_f^+] \\ & y \in \operatorname*{argmax}_{y'} \sum_{e \in \mathcal{E}_f} d(e) y'_e \\ & \qquad \text{s.t. } \sum_{e \in \mathcal{E}_f} a(e) y'_e = b_f \\ & \qquad\quad y' \in [0,1]^{\mathcal{E}_f} \end{aligned} \tag{BCKP-b_f}$$

Very similarly to Lemma 3.6, we now show that the problem (BCKP-b_f) is indeed equivalent to the Bilevel Continuous Knapsack Problem under Assumption (uni′).

Lemma 4.3. *The bilevel optimization problem stated in (BCKP-b_f) and the Bilevel Continuous Knapsack Problem under Assumption (uni′) can be polynomially reduced to each other.*

Proof. In the Bilevel Continuous Knapsack Problem under Assumption (uni′), the leader's only relevant decision amounts to the capacity $b_l \in [0, a(\mathcal{E}_l)]$ that she fills with the items she selects from $\mathcal{E}_l$. Her selection contributes a value of γb_l to her objective function and determines the capacity $b_f = b - b_l$ to be filled by the follower. A straightforward reduction of this problem to the problem (BCKP-b_f) is given by disregarding the leader's items $\mathcal{E}_l$ and setting $b_f^- = \max\{0, b - a(\mathcal{E}_l)\}$ and $b_f^+ = \min\{a(\mathcal{E}_f), b\}$. Again, the feasible solutions of the two problems directly correspond to each other and yield the same leader's objective values in both problems, up to the constant γb.

For the reduction in the opposite direction, we may assume that we are in the nontrivial case $b_f^- < b_f^+$. We introduce a single leader's item such that the leader can fix the follower's

capacity at a linear cost by selecting any fraction of it. Formally, we set $b = b_f^+$ and $\mathcal{E}_l = \{e\}$ with $a(e) = b - b_f^-$ and $c(e) = \gamma a(e)$. The same observation as above holds for the leader's objective values of the solutions. □

The problem (BCKP-b_f) and a polynomial-time algorithm for solving it have already been presented in [DKPK15, Chapter 1.3], where it is used as an introductory example of a bilevel optimization problem. In our articles [BH22; BHI22], we discussed robust and stochastic versions of the problem (BCKP-b_f), respectively. In Lemma 4.10, we will see that Lemma 4.3 holds also in the robust setting. Therefore, the algorithmic results for the (Robust) Bilevel Continuous Knapsack Problem that will be presented in Sections 4.3, 4.4.1, and 4.4.2 can be seen as generalizations of the ones in [DKPK15, Chapter 1.3] and in [BH22], i.e., from the (Robust) Bilevel Continuous Knapsack Problem under Assumption (uni′) to the one under Assumption (disj). Similarly, the pseudopolynomial-time algorithms in Section 4.4.3 are generalizations of the ones presented in [Hen25]. On the other hand, the hardness results in this chapter (see Sections 4.4.3, 4.4.4, and 4.4.5) are directly taken from [BH22] and formulated for robust versions of the problem (BCKP-b_f). Thus, using Lemma 4.10, they hold even for the special case of the Robust Bilevel Continuous Knapsack Problem corresponding to Assumption (uni′).

Remark 4.4. In the proof of Lemma 4.3, we introduced a single leader's item of size $b - b_f^-$ for the reduction from the problem (BCKP-b_f) to the Bilevel Continuous Knapsack Problem under Assumption (uni′), while we defined $b - b_f^-$ leader's items in the corresponding reduction for the Bilevel Selection Problem in the proof of Lemma 3.6. Clearly, the former is not possible in case of the Bilevel Selection Problem because the items cannot have different sizes there. On the other hand, several leader's items, e.g., each having an item size of 1, with a total size of $b - b_f^-$, would yield an equivalent instance of the Bilevel Knapsack Problem as well. However, the reduction might not be polynomial (but only pseudopolynomial) in this case because the number of introduced items is not polynomial in the size of the original instance in general.

Remark 4.5. Analogously to Remark 3.8, we may assume, without loss of generality, that $\gamma = 0$ holds in the problem (BCKP-b_f) and, accordingly, in the Bilevel Continuous Knapsack Problem under Assumption (uni′). For this, include the costs γ into the leader's values of the follower's items by setting $c'(e) = c(e) - \gamma a(e)$ for all $e \in \mathcal{E}_f$. Recall that we do not restrict the item values to be nonnegative, both in the Bilevel Continuous Knapsack Problem and in (BCKP-b_f). Otherwise, this transformation would not be possible in general. In [BH22], we focus on the variant of (BCKP-b_f) with $\gamma = 0$ (and arbitrary item values c) as well, after arguing that this setting is equivalent to the general one.

In Chapter 3, we also investigated the more general Assumptions (arb) and (sub) about the item sets, in which there is either no restriction on the relation between $\mathcal{E}_l$ and $\mathcal{E}_f$ or we assume the follower to control all items, i.e., $\mathcal{E}_l \subseteq \mathcal{E}_f$. They appear to be more complex than Assumption (disj); see Section 3.4.2. Since this setting, in particular in combination with Assumption (cc) (see Section 3.4.2.2), is not fully understood even for the Robust Bilevel Selection Problem, we focus on the simpler case of Assumption (disj) for the Robust Bilevel Continuous Knapsack Problem in this chapter.

We finally turn to the Assumptions (=′) and (≤′) about the capacity constraint. In the Bilevel Continuous Knapsack Problem, we make Assumption (=′), i.e., the follower is required to fill the capacity b completely. In contrast to that, under Assumption (≤′), the follower is allowed to leave some part of the capacity empty if it does not benefit him to pack further items. We have seen in Lemma 3.16 for the Bilevel Selection Problem that the two assumptions are equivalent. For the Bilevel Continuous Knapsack Problem, i.e., under Assumptions (cc) and (disj), this remains true:

Lemma 4.6. *The Bilevel Knapsack Problem under Assumptions (cc), (disj), and (=′) and the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (≤′) can be polynomially reduced to each other.*

Proof. We follow the ideas of the proof of Lemma 3.16. Let first an instance of the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (≤′) be given. Define $\mathcal{E}_f^{\geq 0} = \{e \in \mathcal{E} \mid d(e) > 0 \text{ or } d(e) = 0 \text{ and } c(e) \geq 0\}$ in the optimistic setting, replacing $c(e) \geq 0$ by $c(e) \leq 0$ in the pessimistic setting. Then the items in $\mathcal{E}_f \setminus \mathcal{E}_f^{\geq 0}$ will never occur in an optimum follower's solution under Assumption (≤′). We define an instance of the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (=′) as follows. If $b \leq a(\mathcal{E}_f^{\geq 0})$, use the same instance as the given one. Otherwise, add one new item e to $\mathcal{E}_f$ and set $a(e) = b - a(\mathcal{E}_f^{\geq 0})$ and $d(e) = c(e) = 0$ for it. Analogously to the proof of Lemma 3.16, this dummy item ensures that no item from $\mathcal{E}_f \setminus \mathcal{E}_f^{\geq 0}$ will be selected by the follower, even in the setting of Assumption (=′), and that the sets of feasible leader's choices of the two instances, as well as the corresponding objective function values, are the same.

For the other direction, consider an instance of the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (=′). We construct an instance of the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (≤′) as follows. For all $e \in \mathcal{E}_f$, we add $Ma(e)$ to the follower's item value $d(e)$, where $M \in \mathbb{Q}_{\geq 0}$ is large enough such that all values $d(e)$ become positive. This changes every follower's profit $d(e)/a(e)$ by the constant M. Hence, the follower's greedy order does not change. As all item values are positive, the follower now fills the capacity completely if there are enough items in $\mathcal{E}_f$ to do so. In order to discourage the leader from choosing a solution that leaves more capacity to the follower than he is able to fill using his regular items, in case $a(\mathcal{E}_f) < b$, we add an artificial follower's item e to $\mathcal{E}_f$, for which we set $a(e) = b - a(\mathcal{E}_f)$, $d(e) = \varepsilon$, and $c(e) = \alpha$. Here, $\varepsilon \in \mathbb{Q}_{>0}$ is chosen such that $\varepsilon/a(e)$ is smaller than the follower's profits of all other items, and $\alpha \in \mathbb{Q}$ such that $\alpha/a(e)$ is smaller than the leader's profits of all other items. Again, analogously to the proof of Lemma 3.16, this construction ensures that the sets of optimum leader's solutions of both instances are the same. □

For other variants of the Bilevel Knapsack Problem, one has to be more careful regarding the Assumptions (=′) and (≤′). In particular, for the Bilevel Knapsack Problem under Assumption (bb), like for the single-level Knapsack Problem (see Section 4.1), there is a significant difference between them. In fact, the problem version corresponding to Assumption (=′) might not even have a feasible solution, while this is always the case under Assumption (≤′). Moreover, even checking feasibility of the follower's problem (given some fixed leader's choice) is $\mathcal{NP}$-hard, as it corresponds to a Subset Sum Problem.

Remark 4.7. Additional natural variants of the Bilevel Knapsack Problem could be defined by restricting the leader's and/or follower's item values to be nonnegative. Similarly to Remarks 3.4 and 3.17, one can show such problem variants to be equivalent to the problem variants with arbitrary rational item values. Indeed, given arbitrary item values c and d, a corresponding instance with nonnegative item values can be defined by $c'(e) = c(e) + Ca(e)$ and $d'(e) = d(e) + Da(e)$ for all items e, for appropriate constants $C, D \in \mathbb{Q}_{>0}$. In case of Assumption $(=')$, this changes the objective function value of any given solution only by a constant. In case of Assumption $(\leq')$ (and Assumptions (cc) and (disj)), the argument can be combined with Lemma 4.6; see also Remark 3.17 for more details.

In this chapter, we will work with arbitrary rational item values. This allows to assume $\gamma = 0$ in Assumption (uni′) and (BCKP-b_f); see Remark 4.5. Moreover, it facilitates switching between maximization and minimization problem variants, e.g., in the subroutine for the Continuous Knapsack Problem used in Algorithm 4.2.

4.2.3 Complexity of the Bilevel Knapsack Problem Under Assumptions (bb) and $(\leq')$ and Related Literature

The Bilevel Knapsack Problem under Assumptions (bb), (disj), and $(\leq')$ as well as other bilevel variants of the Knapsack Problem having a different structure than our Bilevel Knapsack Problem have been studied in several previous works. Regarding the computational complexity of such problems, an interesting article is [CCLW14]. Its authors, Caprara, Carvalho, Lodi, and Woeginger, derive complexity results for three different bilevel knapsack problems that had appeared in the literature before. The first one ("DeRi") is due to Dempe and Richter [DR00] and corresponds to a bilevel optimization problem in which the leader determines the capacity for the follower's Knapsack Problem, like in the problem (BCKP-b_f), but under Assumptions (bb) and $(\leq')$. In [CCLW14], a restricted version is discussed in which the item sizes and the follower's item values coincide. The second bilevel knapsack problem studied in [CCLW14] ("MACH") was first defined by Mansi, Alves, Carvalho, and Hanafi in [MACH12] and is precisely our Bilevel Knapsack Problem under Assumptions (bb), (disj), and $(\leq')$. The third variant is the bilevel knapsack problem that was introduced by DeNegre in [DeN11] ("DNeg"). This is an interdiction problem, i.e., the follower solves a Knapsack Problem and the leader can remove items from the ones the follower is allowed to choose from, while respecting her own budget and aiming to harm the follower. In other words, the leader and the follower have a common set of items and each item can be selected by at most one of them. Each of the two players fills their own knapsack, given by separate capacities and possibly different leader's and follower's sizes of all items. The objective functions of the two players are opposite to each other. Similarly as for the DeRi variant, the discussion in [CCLW14] is restricted to the case in which the follower's item sizes and the follower's item values coincide.

All three mentioned bilevel knapsack problems are shown to be Σ_2^{p}-hard in [CCLW14]. Moreover, pseudopolynomial-time algorithms are presented for the DeRi and the MACH variant, while the DNeg variant is shown to be strongly $\mathcal{NP}$-hard. The DeRi and the MACH variant are also proved to be not approximable within any constant factor in polynomial time unless $\mathcal{P} = \mathcal{NP}$. Finally, a polynomial-time approximation scheme is developed for the DNeg variant (with coinciding follower's item sizes and values). In [CWZ22], a polynomial-

time approximation scheme for the general DNeg variant as well as approximation results for a more general class of interdiction problems have been presented.

For the DeRi variant, already Dempe and Richter developed a dynamic programming approach in [DR00] that solves the problem in pseudopolynomial time and that was later improved in [BHM09]. A pseudopolynomial-time dynamic programming algorithm for the MACH variant was first presented in [MACH12], together with several preprocessing steps that might reduce the size of the dynamic program. A more general version of the MACH variant, with general linear constraints in the leader's problem, has been considered in [BHM13], where a single-level reformulation is obtained in pseudopolynomial time using dynamic programming.

In summary, for the Bilevel Knapsack Problem we defined in this chapter, we know from [MACH12; CCLW14] that it is (weakly) Σ_2^{p}-hard and solvable in pseudopolynomial time in case of Assumptions (bb), (disj), and ($\leq'$).

Polynomial-time algorithms for the continuous version of the problem in [DeN11] can be found in [CLM18; FW20]. Similarly to the algorithmic results in this chapter, they rely on structural insights about the follower's Continuous Knapsack Problem given by Dantzig's algorithm and identifying the critical item. A stochastic version of the bilevel knapsack problem of [DR00] is considered in [ÖPS10]: It is assumed that the capacity in the follower's problem depends on the leader's decision as well as an uncertain value, which is drawn from a discrete probability distribution. Further bilevel variants of the Knapsack Problem are investigated, e.g., in [CZ13; QK15; PNP19; PNPS21]. Important studies of practical solution approaches for bilevel knapsack problems, in particular the problem in [DeN11], include [CCLW16; FLMS19; DS20; WF25a]. For a more thorough literature review of bilevel knapsack problems, including details on most of the mentioned results, we refer to [Car23].

In view of the hardness results mentioned above and our general motivation to investigate the additional complexity that uncertainty adds to easy bilevel optimization problems, we do not study the Bilevel Knapsack Problem under Assumption (bb) further in this thesis. Instead, we will focus on the Bilevel Continuous Knapsack Problem, i.e., the Bilevel Knapsack Problem under Assumptions (cc), (disj), and ($='$), for the remainder of this chapter.

A practical motivation to investigate the Bilevel Continuous Knapsack Problem under uncertainty is the possible application in revenue management that is also described in [BHM09; MACH12] for the respective variants of bilevel knapsack problems: The leader, who might be an investor or a company, hands a part of her savings over to the follower, an intermediary, who invests it, maximizing his own profit and giving back part of the returns to the leader. It is very natural to consider continuous variables in this application. Moreover, it is reasonable to consider the follower's objective uncertain, due to the inherent risk of investments and hence the investor's uncertainty about the intermediary's assessment of the expected returns.

In the next section, we will investigate the structure of and a polynomial-time algorithm for the Bilevel Continuous Knapsack Problem without uncertainty, before turning to the Robust Bilevel Continuous Knapsack Problem in Section 4.4.

4.3 The Bilevel Continuous Knapsack Problem Without Uncertainty

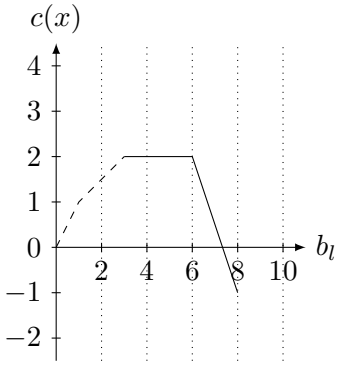
In this section, we show how to solve the Bilevel Continuous Knapsack Problem, i.e., the Bilevel Knapsack Problem under Assumptions (cc), (disj), and (=′), in polynomial time. We have already discussed in Section 4.2.1 that the follower can solve his problem as a single-level Continuous Knapsack Problem in a greedy way. We will now investigate the leader's point of view.

We follow the ideas of Section 3.3.2, where the structure of the leader's objective function of the Bilevel Selection Problem under Assumptions (cc) and (disj) was investigated, and generalize them to the Bilevel Continuous Knapsack Problem. Again, the only interaction between the leader and the follower lies in the capacity each of the two players uses. Given a fixed splitting of the capacity b into leader's and follower's capacities b_l and b_f, respectively, each player can be assumed to solve a single-level Continuous Knapsack Problem on their own set of items. The leader's objective values resulting from all possible such splittings of the capacity can be described as a continuous piecewise linear function in the parameter b_l, as detailed in the following.

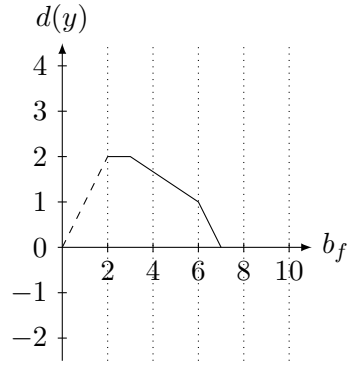
First recall that the range of leader's capacities corresponding to feasible solutions is given by $[b_l^-, b_l^+]$ with $b_l^- = \max\{0, b - a(\mathcal{E}_f)\}$ and $b_l^+ = \min\{b, a(\mathcal{E}_l)\}$. The order of the items in $\mathcal{E}_l$ in the leader's greedy algorithm is independent of the choice of b_l and given by sorting them nonincreasingly according to their profits $c(e)/a(e)$; see Section 4.1. Let $e_1, \dots, e_{n_l}$ denote the items of $\mathcal{E}_l$ in this order. For any b_l, the leader selects a fractional prefix (see also Definition 3.45) of this order, i.e., a solution $x \in [0,1]^{\mathcal{E}_l}$ packing the first items of the order and a fraction of the next (critical) item such that $\sum_{e \in \mathcal{E}_l} a(e)x_e = b_l$. For the function describing the leader's objective value, this results in a linear segment for each item j being in the role of the critical item, from $b_l = \sum_{i=1}^{j-1} a(e_i)$ with a value of $\sum_{i=1}^{j-1} c(e_i)$ to $b_l = \sum_{i=1}^{j} a(e_i)$ with a value of $\sum_{i=1}^{j} c(e_i)$, assuming this is within the range $[b_l^-, b_l^+]$. Hence, the width of the linear segment is given by $a(e_j)$ and its slope corresponds to the profit $c(e_j)/a(e_j)$. This defines a continuous piecewise linear function $\mathrm{PLF}'(\mathcal{E}_l, a \restriction \mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$ on the range $[b_l^-, b_l^+]$ that describes the leader's objective value resulting from her own choice of items. Note that this function is concave because, due to the greedy order of the items, the slopes of the linear segments are sorted nonincreasingly. See Figure 4.1a for an example. Such a function can be computed in $\mathcal{O}(n_l \log n_l)$ time and can be represented by a list of its $\mathcal{O}(n_l)$ many vertices.

The follower's objective value resulting from his choice on the items in $\mathcal{E}_f$ can be described analogously as a continuous piecewise linear function $\mathrm{PLF}'(\mathcal{E}_f, a \restriction \mathcal{E}_f, (d, \pm c \restriction \mathcal{E}_f), b_f^-, b_f^+)$ in the parameter $b_f \in [b_f^-, b_f^+]$ with $b_f^- = b - b_l^+ = \max\{0, b - a(\mathcal{E}_l)\}$ and $b_f^+ = b - b_l^- = \min\{b, a(\mathcal{E}_f)\}$. An example can be found in Figure 4.1b. The term $(d, \pm c \restriction \mathcal{E}_f)$ indicates the optimistic or the pessimistic setting, where the leader's item values c are respected as a secondary criterion. More specifically, the leader's profits $c(e)/a(e)$ are used as a secondary criterion when sorting the items according to the follower's profits $d(e)/a(e)$ in case several items have the same follower's profit; see Section 4.2.1.

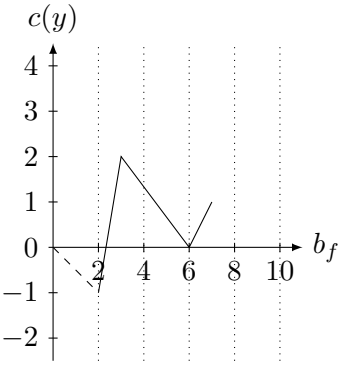
In order to describe the impact of the follower's choice on the leader's objective, we define a piecewise linear function that still depends on the follower's greedy order given by the follower's profits $d(e)/a(e)$ (and possibly the leader's profits $c(e)/a(e)$ as a secondary criterion), but describes the leader's objective value, given by her item values c,



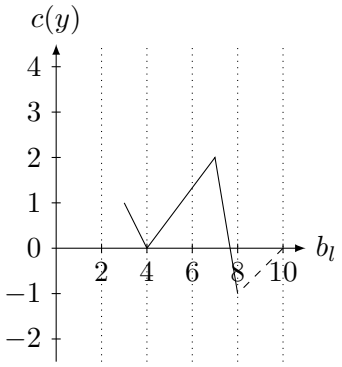
(a) Leader's objective of her own items, $\mathrm{PLF}'(\mathcal{E}_l, a \restriction \mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$



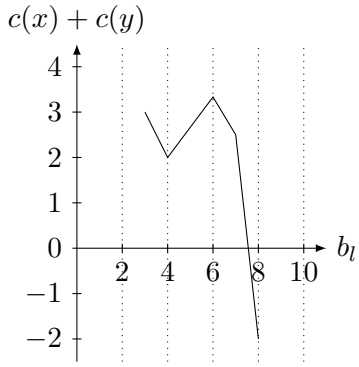
(b) Follower's objective of his own items, $\mathrm{PLF}'(\mathcal{E}_f, a \restriction \mathcal{E}_f, d, b_f^-, b_f^+)$



(c) Leader's objective of the follower's items, depending on b_f



(d) Leader's objective of the follower's items, depending on b_l, $\overline{\mathrm{PLF}}'(\mathcal{E}_f, a \restriction \mathcal{E}_f, d, c \restriction \mathcal{E}_f, b_l^-, b_l^+, b)$



(e) Leader's total objective (sum of the functions in Figures 4.1a and 4.1d)

Figure 4.1. Example of leader's and follower's objective functions of the Bilevel Continuous Knapsack Problem, depending on the splitting of the total capacity b into leader's and follower's capacities b_l and b_f, respectively. Here, $\mathcal{E}_l = \{e_1, e_2, e_3, e_4\}$, $\mathcal{E}_f = \{e_5, e_6, e_7, e_8\}$, $b = 10$, $a(e_1) = 1$, $a(e_2) = 2$, $a(e_3) = 3$, $a(e_4) = 2$, $a(e_5) = 2$, $a(e_6) = 1$, $a(e_7) = 3$, $a(e_8) = 1$, $c(e_1) = 1$, $c(e_2) = 1$, $c(e_3) = 0$, $c(e_4) = -3$, $c(e_5) = -1$, $c(e_6) = 3$, $c(e_7) = -2$, $c(e_8) = 1$, $d(e_5) = 2$, $d(e_6) = 0$, $d(e_7) = -1$, and $d(e_8) = -1$.
The terms $c(x)$, $c(y)$, and $d(y)$ are used as shortcuts for the objective function values $\sum_{e\in\mathcal{E}_l} c(e)x_e$, $\sum_{e\in\mathcal{E}_f} c(e)y_e$, and $\sum_{e\in\mathcal{E}_f} d(e)y_e$, where $x \in [0,1]^{\mathcal{E}_l}$ and $y \in [0,1]^{\mathcal{E}_f}$ are optimum leader's and follower's solutions for given values of b_l and b_f, respectively.
The dashed linear segments are parts of the functions that correspond to infeasible overall solutions due to the bounds $b_l^- = 3$, $b_l^+ = 8$, $b_f^- = 2$, and $b_f^+ = 7$.

of the resulting follower's solution. An example is depicted in Figure 4.1c. In addition, we need this function to depend on the parameter b_l instead of b_f in order to combine it with the function $\mathrm{PLF}'(\mathcal{E}_l, a \restriction \mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$ described above. Given the total capacity b, this can be done by a simple transformation, resulting in the function $\overline{\mathrm{PLF}}'(\mathcal{E}_f, a \restriction \mathcal{E}_f, (d, \pm c \restriction \mathcal{E}_f), c \restriction \mathcal{E}_f, b_l^-, b_l^+, b)$; see also Figure 4.1d. This piecewise linear function can be computed in $\mathcal{O}(n_f \log n_f)$ time and contains $\mathcal{O}(n_f)$ many vertices.

The total leader's objective function, as a function of b_l, is now the sum of the two piecewise linear functions described above. See Figure 4.1e for this function in the example. It is again a continuous piecewise linear function, has $\mathcal{O}(n)$ many vertices, and can be computed in an overall time of $\mathcal{O}(n \log n)$. Finally, an optimum solution from the leader's perspective corresponds to a maximum of this function and can be found by iterating through its vertices.

Remark 4.8. Consider the Bilevel Continuous Knapsack Problem under Assumption (uni′), which is the special case of Assumption (disj) where the leader's profits of all leader's items are the same. Analogously to Remark 3.21, the above observations then simplify such that the part of the leader's objective function that depends on her own items (see Figure 4.1a) is just a linear function with slope γ.

In Lemma 4.3, we have seen that the Bilevel Continuous Knapsack Problem under Assumption (uni′) is equivalent to the problem (BCKP-b_f), where the leader directly controls the follower's capacity b_f as a variable. In this problem variant, the leader's objective function is the sum of the piecewise linear function that is shown in Figure 4.1c and a linear function with slope $-\gamma$.

If $\gamma = 0$, which we usually assume in this chapter and also in [BH22] (see Remark 4.5), then the linear function vanishes and the leader's objective function is simply a piecewise linear function representing the follower's choice of his items.

Instead of explicitly handling the piecewise linear functions, we can also directly work with a set of possible values for b_l, i.e., a set of (potential) vertices of the final piecewise linear function. In a similar fashion as Algorithm 3.1 for the Bilevel Selection Problem, this results in Algorithm 4.1. Its correctness can be derived from the construction of the piecewise linear functions explained above:

Theorem 4.9. *Algorithm 4.1 solves the Bilevel Continuous Knapsack Problem in time $\mathcal{O}(n \log n)$.*

Proof. For the follower, only the total capacity $b_l = \sum_{e \in \mathcal{E}_l} a(e)x_e$ used by the leader is relevant, but not the precise leader's solution x. Therefore, if b_l is fixed, it is optimum for the leader to fill this capacity greedily according to an optimum solution of the corresponding Continuous Knapsack Problem, as in Line 7. The follower fills the remaining capacity $b - b_l$ greedily according to his objective (and possibly the leader's objective as a secondary criterion) as well; see Line 8. By choosing b_l such that the resulting leader's and follower's solutions x and y lead to the maximum total leader's value in terms of c, the overall problem can be solved. This is done in Line 9. It remains to show that the set B defined in Line 5 actually contains an optimum value of b_l.

From the perspective of the piecewise linear functions described above, it is clear that an optimum b_l corresponds to a vertex in one of the two piecewise linear functions $\mathrm{PLF}'(\mathcal{E}_l, a \restriction \mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$ and $\overline{\mathrm{PLF}}'(\mathcal{E}_f, a \restriction \mathcal{E}_f, (d, \pm c \restriction \mathcal{E}_f), c \restriction \mathcal{E}_f, b_l^-, b_l^+, b)$. The set B

Algorithm 4.1. Algorithm for the Bilevel Continuous Knapsack Problem

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $n_l = |\mathcal{E}_l|$, $n_f = |\mathcal{E}_f|$, and $n = |\mathcal{E}_l \cup \mathcal{E}_f|$, $a\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}_{>0}$, $b \in \mathbb{Q} \cap [0, a(\mathcal{E}_l \cup \mathcal{E}_f)]$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, $d\colon \mathcal{E}_f \to \mathbb{Q}$

Output : an optimum solution $(x, y) \in [0,1]^{\mathcal{E}_l} \times [0,1]^{\mathcal{E}_f}$ of the Bilevel Continuous Knapsack Problem

1 compute a bijection $p_l\colon \{1, \dots, n_l\} \to \mathcal{E}_l$ such that $\frac{c(p_l(1))}{a(p_l(1))} \geq \dots \geq \frac{c(p_l(n_l))}{a(p_l(n_l))}$

2 compute a bijection $p_f\colon \{1, \dots, n_f\} \to \mathcal{E}_f$ such that $\frac{d(p_f(1))}{a(p_f(1))} \geq \dots \geq \frac{d(p_f(n_f))}{a(p_f(n_f))}$ and $\frac{c(p_f(i))}{a(p_f(i))} \geq \frac{c(p_f(i+1))}{a(p_f(i+1))}$ [or $\frac{c(p_f(i))}{a(p_f(i))} \leq \frac{c(p_f(i+1))}{a(p_f(i+1))}$] for all $i \in \{1, \dots, n_f - 1\}$ with $\frac{d(p_f(i))}{a(p_f(i))} = \frac{d(p_f(i+1))}{a(p_f(i+1))}$ in the optimistic [pessimistic] setting

3 $b_l^- := \max\{0, b - a(\mathcal{E}_f)\}$

4 $b_l^+ := \min\{b, a(\mathcal{E}_l)\}$

5 $B := [b_l^-, b_l^+] \cap \left(\left\{\sum_{i=1}^{j} a(p_l(i)) \mid j \in \{0, \dots, n_l\}\right\} \cup \left\{b - \sum_{i=1}^{j} a(p_f(i)) \mid j \in \{0, \dots, n_f\}\right\}\right)$

6 **for** $b_l \in B$ **do**

7 $\quad x^{b_l} :=$ Continuous Knapsack Problem$(\mathcal{E}_l, a \restriction \mathcal{E}_l, b_l, c \restriction \mathcal{E}_l)$

8 $\quad y^{b_l} :=$ Continuous Knapsack Problem$(\mathcal{E}_f, a \restriction \mathcal{E}_f, b - b_l, (d, \pm c \restriction \mathcal{E}_f))$

9 **return** $\operatorname{argmax}\left\{\sum_{e \in \mathcal{E}_l} c(e) x_e + \sum_{e \in \mathcal{E}_f} c(e) y_e \mid (x, y) \in \{(x^{b_l}, y^{b_l}) \mid b_l \in B\}\right\}$

is precisely the union of the vertex sets of these two functions. Therefore, the algorithm indeed considers all potentially optimum values of b_l.

Regarding the running time, first note that sorting the leader's and the follower's items in Lines 1 and 2 can be done in $\mathcal{O}(n \log n)$ time. Based on these orders, the set B can be computed in $\mathcal{O}(n)$ time and stored as a sorted list in Line 5. By iterating through this sorted list, the loop in Lines 6 to 8 can be implemented to run in $\mathcal{O}(n)$ time as well. Indeed, each iteration can be performed in constant time by adjusting the solutions from the previous iteration based on the current critical item given by the item orders from Lines 1 and 2. □

Finally, we highlight some differences between the Bilevel Selection Problem under Assumption (disj) (see Section 3.3) and the Bilevel Continuous Knapsack Problem that can be observed through the piecewise linear functions and the solution algorithms.

For the Bilevel Selection Problem, we have seen in Lemma 3.14 and Section 3.3.2 that the Assumptions (bb) and (cc) are equivalent, i.e., that even under Assumption (cc), there is always an optimum solution where both players take binary decisions. This is different for the Bilevel Continuous Knapsack Problem, as can be seen in the example depicted in Figure 4.1. Here, the optimum solution corresponds to the maximum of the function in Figure 4.1e, i.e., $b_l = 6$, and is given by the leader's decision $x_{e_1} = x_{e_2} = x_{e_3} = 1$, $x_{e_4} = 0$ and the follower's decision $y_{e_5} = y_{e_6} = 1$, $y_{e_7} = 1/3$, $y_{e_8} = 0$. Recall that also the known complexity results indicate that Assumptions (bb) and (cc) are not equivalent here, since the Bilevel Knapsack Problem under Assumption (bb) is Σ_2^{p}-hard (see Section 4.2.3), while Theorem 4.9 shows that the Bilevel Continuous Knapsack Problem can be solved in polynomial time. However, in an optimum solution of the Bilevel Continuous Knapsack Problem, at least one of the two players must be in a

situation where they (can be assumed to) select a binary solution. Indeed, every vertex in the piecewise linear function that the leader maximizes (see the set B defined in Line 5 of Algorithm 4.1) corresponds to a binary choice of the leader (in case it belongs to the set $\{\sum_{i=1}^{j} a(p_l(i)) \mid j \in \{0, \dots, n_l\}\}$) or to a binary choice of the follower (in case it belongs to the set $\{b - \sum_{i=1}^{j} a(p_f(i)) \mid j \in \{0, \dots, n_f\}\}$), but not necessarily both.

Compared to the set B for the Bilevel Continuous Knapsack Problem, the set of potential vertices of the piecewise linear function is very simple in case of the Bilevel Selection Problem under Assumption (cc), as it consists of all integers in the range $[b_l^-, b_l^+]$; see Algorithm 3.1 and Section 3.3.2. In the Bilevel Continuous Knapsack Problem, the widths of the linear segments are given by the item sizes a, which are all 1 in the special case corresponding to the Bilevel Selection Problem.

For the Bilevel Continuous Knapsack Problem, if all item sizes a and the total capacity b are integers, it would also be correct to simply enumerate all integer values $b_l \in \{b_l^-, \dots, b_l^+\}$, like in Algorithm 3.1 for the Bilevel Selection Problem, because they form a superset of the set B. However, this would not lead to a polynomial-time algorithm here, but only to a pseudopolynomial-time algorithm, because the range of possible capacities depends on the number b, which is not necessarily polynomial in the input size in case of the Bilevel Continuous Knapsack Problem.

On the other hand, a pseudopolynomial-time algorithm like this, enumerating all $b_l \in \{b_l^-, \dots, b_l^+\}$, can even be used in case of the Bilevel Knapsack Problem under Assumptions (bb) and (disj). For this, Knapsack Problems are solved in pseudopolynomial time as a subroutine instead of the Continuous Knapsack Problems in Lines 7 and 8 of Algorithm 4.1. This idea results in the pseudopolynomial-time algorithm for the Bilevel Knapsack Problem under Assumptions (bb), (disj), and ($\leq'$) presented in [MACH12; CCLW14]; see also Section 4.2.3. Note that restricting the set of potential values of b_l, like with the set B above, is not easily possible in case of Assumption (bb) because the leader's and follower's choices depend on b_l in a much more complicated fashion as they do not follow a greedy structure.

Similar structural complications of the Bilevel Continuous Knapsack Problem, in comparison to the Bilevel Selection Problem under Assumptions (disj) and (cc), as described above will also be reflected in the following section when we discuss robust variants of the Bilevel Continuous Knapsack Problem.

4.4 The Robust Bilevel Continuous Knapsack Problem

We have seen in Section 4.3 that the Bilevel Continuous Knapsack Problem can be solved in polynomial time by a combinatorial algorithm. Therefore, it is still a relatively easy bilevel optimization problem, although it is more general than the Bilevel Selection Problem under Assumptions (disj) and (cc) discussed in Section 3.4.1.2. On this basis, we will now study the Robust Bilevel Continuous Knapsack Problem, with different types of uncertainty sets, in order to further investigate the additional complexity that is added to bilevel optimization problems when including robustness in the sense of Section 2.4.1.

Recall from Section 2.4.1 that, in the robust bilevel problems we consider in this thesis, we assume the follower's objective function to be uncertain from the leader's perspective. Hence, given an uncertainty set $\mathcal{U} \subseteq \mathbb{R}^{\mathcal{E}_f}$ of possible follower's objective functions

$d\colon \mathcal{E}_f \to \mathbb{R}$, the Robust Bilevel Continuous Knapsack Problem can be written as follows, extending (BCKP):

$$\begin{aligned} \max_{x\in[0,1]^{\mathcal{E}_l}} \min_{d\in\mathcal{U}} & \sum_{e\in\mathcal{E}_l} c(e)x_e + \sum_{e\in\mathcal{E}_f} c(e)y_e \\ \text{s.t. } y \in \operatorname*{argmax}_{y'} & \sum_{e\in\mathcal{E}_f} d(e)y'_e \\ \text{s.t. } & \sum_{e\in\mathcal{E}_l} a(e)x_e + \sum_{e\in\mathcal{E}_f} a(e)y'_e = b \\ & y' \in [0,1]^{\mathcal{E}_f} \end{aligned} \tag{RBCKP}$$

Note that (RBCKP) is a special case of the general formulation (R-L) of a robust bilevel optimization problem (see Section 2.4.1) as well as a generalization of the variant of (RBSP) that corresponds to Assumptions (disj), (cc), and (=) (see Section 3.4).

Like in Sections 2.4.1 and 3.4, the structure of the Robust Bilevel Continuous Knapsack Problem can be illustrated with three decision makers: First, the leader chooses a solution $x \in [0,1]^{\mathcal{E}_l}$, i.e., she fractionally selects some items from $\mathcal{E}_l$, while she anticipates how the adversary and the follower will react. Second, the adversary draws a follower's objective function $d \in \mathcal{U}$ from the uncertainty set such that the outcome, i.e., the resulting follower's decision $y \in [0,1]^{\mathcal{E}_f}$, in terms of its leader's value $\sum_{e\in\mathcal{E}_f} c(e)y_e$, is worst possible. Finally, given the fixed leader's and adversary's decisions x and d, the follower solves the inner single-level Continuous Knapsack Problem on the item set $\mathcal{E}_f$. From the follower's perspective, nothing changes compared to the Bilevel Continuous Knapsack Problem without uncertainty (see Sections 4.2.1 and 4.3). But for the leader's problem, the uncertainty introduces a significant difficulty. In this section, we will study the computational complexity of both the adversary's problem (which is a bilevel optimization problem itself) and the leader's problem, for different types of uncertainty sets.

Analogously to Remark 3.22 for the Robust Bilevel Selection Problem, note that the adversary's decision determines the greedy order that the follower uses when solving his Continuous Knapsack Problem. Again, only this order is relevant for the leader's and the adversary's objectives, but not the precise follower's item values the adversary selects. Therefore, for any type of uncertainty set, there is only a finite number of different possible outcomes for the leader. Hence, in the following, it will be again crucial to study which follower's greedy orders are possible depending on the type of uncertainty set.

In addition to (RBCKP), we will also deal with the following robust variant of the problem (BCKP-b_f):

$$\begin{aligned} \max_{b_f\in[b_f^-,b_f^+]} \min_{d\in\mathcal{U}} & \sum_{e\in\mathcal{E}_f} c(e)y_e \\ \text{s.t. } y \in \operatorname*{argmax}_{y'} & \sum_{e\in\mathcal{E}_f} d(e)y'_e \\ \text{s.t. } & \sum_{e\in\mathcal{E}_f} a(e)y'_e = b_f \\ & y' \in [0,1]^{\mathcal{E}_f} \end{aligned} \tag{RBCKP-b_f}$$

Note that, while the leader's objective in (BCKP-b_f) contains an additional term depending on the parameter γ, we have set $\gamma = 0$ in the formulation (RBCKP-b_f). This is

not a restriction according to Remark 4.5, which is still valid in the robust setting.

We have seen in Lemma 4.3 that the problem (BCKP-b_f) is equivalent to the Bilevel Continuous Knapsack Problem under Assumption (uni′). Using the same arguments as in the proof of Lemma 4.3, one can show that this is also true for the corresponding robust problems:

Lemma 4.10. *The problem stated in (RBCKP-b_f) and the Robust Bilevel Continuous Knapsack Problem under Assumption (uni′) can be polynomially reduced to each other.*

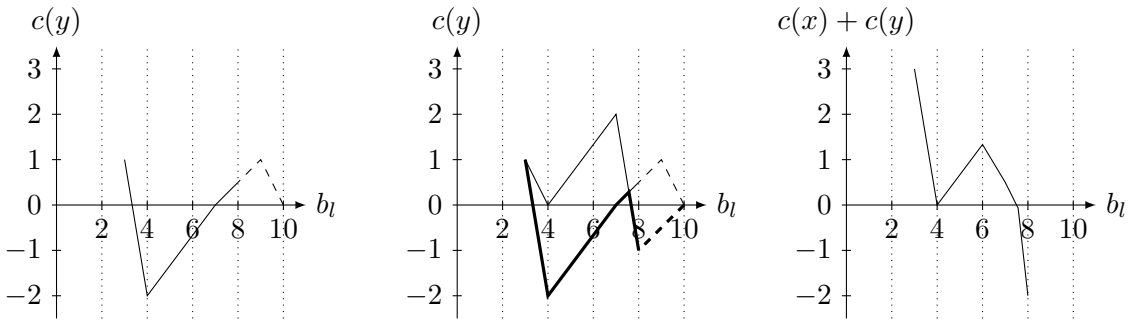
The problem (RBCKP-b_f) corresponds to the variant of a robust bilevel continuous knapsack problem that is studied in [BH22]. There are only minor differences in the two problem definitions, concerning Assumptions ($='$) and ($\leq'$) and nonnegativity assumptions for the item values. These differences are negligible due to Remark 4.5, Lemma 4.6, and Remark 4.7, which can be easily transferred to the robust setting, similarly to Lemma 3.23.

As already mentioned in Section 4.2.2, all $\mathcal{NP}$-hardness results in this chapter (see Sections 4.4.3.1, 4.4.4, and 4.4.5) will be proved for the problem (RBCKP-b_f), like in [BH22]. By Lemma 4.10, they also hold for the Robust Bilevel Continuous Knapsack Problem under Assumption (uni′) and therefore for the more general Robust Bilevel Continuous Knapsack Problem (under Assumption (disj)). On the other hand, the algorithms in Sections 4.4.1, 4.4.2, and 4.4.3.2 will be stated for the general Robust Bilevel Continuous Knapsack Problem and therefore generalize the ones in [BH22; Hen25].

Before we start investigating the different types of uncertainty sets in the following subsections, we describe some structural observations about the Robust Bilevel Continuous Knapsack Problem. They are very similar to the Robust Bilevel Selection Problem under Assumptions (disj) and (cc); see the beginnings of Sections 3.4.1 and 3.4.1.2.

First note again that, due to Assumption (disj), only the capacity b_l filled by the leader (which defines the capacity $b_f = b - b_l$ to be filled by the follower) is relevant for the adversary and the follower, but not the precise leader's solution x. Hence, like for the Bilevel Continuous Knapsack Problem without uncertainty, the leader solves a single-level Continuous Knapsack Problem on her items when the choice of b_l is fixed. However, determining an optimum choice of b_l is more challenging in the robust setting because the corresponding leader's objective values involve solving the adversary's problem, which corresponds to the worst case of several versions of the follower's problem. When developing algorithmic results for discrete uncertainty, interval uncertainty, and discrete uncorrelated uncertainty (see Sections 4.4.1, 4.4.2, and 4.4.3.2), we will first study the adversary's problem, given a fixed feasible leader's solution (or, equivalently, a fixed feasible choice of b_l and b_f). The ideas used for the algorithms that solve the adversary's problems will then be extended to algorithms for solving the leader's problem. This will be similar to the results in Section 3.4.1.2 and again based on viewing the leader's objective function as a piecewise linear function. Also in the proofs of the $\mathcal{NP}$-hardness results in Sections 4.4.3, 4.4.4, and 4.4.5, we will argue based on the problem description in terms of piecewise linear functions.

We will now describe how to extend the piecewise linear function representation presented in Section 4.3 and Figure 4.1 to the robust setting. Without uncertainty, the leader's total objective function, viewed as a function in the parameter b_l (see Figure 4.1e), was the sum of the function describing the leader's value of the items selected by herself ("PLF′"; see



(a) Leader's objective of the follower's items, in scenario d_2, $\overline{\text{PLF}}'(\mathcal{E}_f, a \restriction \mathcal{E}_f, d_2, c \restriction \mathcal{E}_f, b_l^-, b_l^+, b)$

(b) Leader's objective of the follower's items, in the worst case (pointwise minimum of the functions in Figures 4.1d and 4.2a)

(c) Leader's total objective (sum of the functions in Figures 4.1a and 4.2b)

Figure 4.2. Example of the leader's objective function of the Robust Bilevel Continuous Knapsack Problem with discrete uncertainty, depending on the leader's capacity b_l. Here, $\mathcal{E}_l = \{e_1, e_2, e_3, e_4\}$, $\mathcal{E}_f = \{e_5, e_6, e_7, e_8\}$, $b = 10$, $a(e_1) = 1$, $a(e_2) = 2$, $a(e_3) = 3$, $a(e_4) = 2$, $a(e_5) = 2$, $a(e_6) = 1$, $a(e_7) = 3$, $a(e_8) = 1$, $c(e_1) = 1$, $c(e_2) = 1$, $c(e_3) = 0$, $c(e_4) = -3$, $c(e_5) = -1$, $c(e_6) = 3$, $c(e_7) = -2$, $c(e_8) = 1$, and $\mathcal{U} = \{d_1, d_2\}$ with $d_1(e_5) = 2$, $d_1(e_6) = 0$, $d_1(e_7) = -1$, $d_1(e_8) = -1$, $d_2(e_5) = 1$, $d_2(e_6) = -3$, $d_2(e_7) = 0$, and $d_2(e_8) = 3$. This is the same instance as in Figure 4.1 without uncertainty, but now with an additional second scenario d_2.

Figure 4.1a) and the function describing the leader's value of the items selected by the follower ("$\overline{\text{PLF}}'$"; see Figure 4.1d). Again, while the former function does not change in the robust setting, we now get different versions of the latter function for different scenarios from the uncertainty set, i.e., different follower's item values. Each of these functions is a continuous piecewise linear function again and its linear segments correspond to the follower's items $e \in \mathcal{E}_f$, with the width given by the item size $a(e)$ and the slope given by the leader's profit $c(e)/a(e)$. Just the order of the linear segments, corresponding to the greedy order defined by the follower's profits $d(e)/a(e)$, is affected by the uncertainty. Extending the example shown in Figure 4.1, a different scenario than the one displayed in Figure 4.1d is shown in Figure 4.2a.

The adversary's decision amounts to selecting a worst-case scenario, after the leader has fixed her solution and therefore her capacity b_l. Hence, the adversary's decision (including the implied follower's decision) can be represented by taking the pointwise minimum of the piecewise linear functions corresponding to the scenarios; see Figure 4.2b. The overall leader's objective function is again the sum of this function and the one corresponding to her choice on her own items; see Figure 4.2c.

Like in case of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), Figures 4.2b and 4.2c show that the pointwise minimum can introduce additional vertices in the final piecewise linear function, possibly at non-integral values of b_l. For the Robust Bilevel Selection Problem, this implies that Assumptions (bb) and (cc) are not equivalent, in contrast to the Bilevel Selection Problem without uncertainty; see Example 3.41. Here, even for the Bilevel Continuous Knapsack Problem without uncertainty, Assumptions (bb) and (cc) are not equivalent; see the discussion in Section 4.3.

In Section 4.3, we have also observed that, given an optimum value of b_l for the Bilevel Continuous Knapsack Problem, at least one of the two players can select an optimum solution that is binary. This is not the case for the Robust Bilevel Continuous Knapsack Problem because the vertices introduced by the pointwise minimum can correspond to values of b_l where both players have to select some item fractionally. For an example, we refer to Example 3.41 for the Robust Bilevel Selection Problem under Assumptions (uni), (cc), and (=), which can also be interpreted as an example for the Robust Bilevel Continuous Knapsack Problem (under Assumption (uni′)).

Remark 4.11. Analogously to Remark 3.42, observe that the leader's objective function of the Robust Bilevel Continuous Knapsack Problem, for any type of uncertainty set, can be described as the pointwise minimum of a finite number of piecewise linear functions. If an algorithm is available that enumerates all follower's greedy orders that can arise from the given uncertainty set, then the leader's objective function can be determined explicitly by computing the corresponding piecewise linear functions. In case of discrete uncertainty, this is again a straightforward polynomial-time algorithm; see Theorem 4.13. For other types of uncertainty sets, the number of possible greedy orders and therefore the running time of this algorithm is not polynomial in general.

On a related note, we can investigate the complexity of the final piecewise linear function. If it consists of only a polynomial number of linear segments, it might still be possible to compute the function explicitly using a polynomial-time algorithm, even if the number of follower's greedy orders is exponential. We have demonstrated this for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with interval uncertainty or discrete uncorrelated uncertainty; see also Remarks 3.49 and 3.53. In case of the Robust Bilevel Continuous Knapsack Problem with interval uncertainty, we will present an analogous result in Section 4.4.2; see Remark 4.21. For the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty however, we will present an $\mathcal{NP}$-hardness result and a pseudopolynomial-time algorithm in Section 4.4.3. The latter implies that the number of linear segments is pseudopolynomial as well in this case; see Remark 4.32.

In the following subsections, we will now investigate the complexity of the Robust Bilevel Continuous Knapsack Problem for different types of uncertainty sets. While the Bilevel Continuous Knapsack Problem without uncertainty is easily solvable according to Theorem 4.9, it becomes much more involved when adding uncertainty in the follower's objective. However, we show that the Robust Bilevel Continuous Knapsack Problem can still be solved in polynomial time in case of discrete or interval uncertainty; see Sections 4.4.1 and 4.4.2. This will be shown by extending the ideas used for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) in Sections 3.4.1.2.1 and 3.4.1.2.2. For other types of uncertainty sets, the Robust Bilevel Continuous Knapsack Problem turns out to be $\mathcal{NP}$-hard, and the same is true even for the problem of evaluating the leader's objective function, i.e., the adversary's optimization problem. We show this for all remaining types of uncertainty sets introduced in Section 2.3.2. First, we investigate the setting of discrete uncorrelated uncertainty in Section 4.4.3. The hardness result for this setting (see Section 4.4.3.1) demonstrates a difference to the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), where discrete uncorrelated uncertainty leads to a problem that is still solvable in polynomial time; see Section 3.4.1.2.3. However, we also show that the algorithm

presented in Section 3.4.1.2.3 can be extended to yield a pseudopolynomial-time algorithm for the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty; see Section 4.4.3.2. Afterwards, we obtain $\mathcal{NP}$-hardness results also for simplicial uncertainty sets (see Section 4.4.4), thus settling the more general cases of polytopal and budgeted uncertainty, as well as for norm uncertainty sets (see Section 4.4.5), which includes the case of ellipsoidal uncertainty.

4.4.1 Discrete Uncertainty

For discrete uncertainty, i.e., a finite set $\mathcal{U}$, we have seen in Section 3.4.1.2.1 that the adversary's problem and the leader's problem of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) can be solved in polynomial time. Very similarly, we can show the same also for its generalization, the Robust Bilevel Continuous Knapsack Problem.

Theorem 4.12. *For any fixed feasible leader's solution of the Robust Bilevel Continuous Knapsack Problem with a discrete uncertainty set $\mathcal{U}$, the adversary's problem can be solved in time $\mathcal{O}(|\mathcal{U}|n)$.*

Proof. Let $x \in [0,1]^{\mathcal{E}_l}$ be the given leader's solution, $b_l = \sum_{e\in\mathcal{E}_l} a(e)x_e$ the capacity it uses, and $b_f = b - b_l$ the capacity left to the follower. Since we are in the setting of Assumption (disj), the follower's task is to solve a single-level Continuous Knapsack Problem on $\mathcal{E}_f$ with capacity b_f, and the adversary decides on the item values in this problem. Hence, the adversary can enumerate all scenarios in $\mathcal{U}$, solve the follower's problem for each of them in $\mathcal{O}(n)$, and choose a scenario that results in the worst objective value for the leader. This solves the adversary's problem in time $\mathcal{O}(|\mathcal{U}|n)$. □

As already mentioned in Remark 4.11, the simple approach of enumerating all scenarios that is used in the proof of Theorem 4.12 can be combined with the idea to view the leader's objective function as a piecewise linear function in the parameter b_l, leading to a polynomial-time algorithm for the leader's problem:

Theorem 4.13. *The Robust Bilevel Continuous Knapsack Problem with a discrete uncertainty set $\mathcal{U}$ can be solved in time $\mathcal{O}(|\mathcal{U}|n\log(|\mathcal{U}|n))$.*

Proof. We follow the approach described above and compute the piecewise linear functions in the parameter b_l that correspond to the leader's choice and to the follower's choices in all scenarios. This can be done in total time $\mathcal{O}(|\mathcal{U}|n\log n)$ by sorting the items according to the leader's and each scenario's follower's item values. Then the pointwise minimum of the $|\mathcal{U}|$ piecewise linear functions corresponding to the scenarios has to be computed. Each of these functions contains $\mathcal{O}(n)$ vertices. Hence, their pointwise minimum is the lower envelope of $\mathcal{O}(|\mathcal{U}|n)$ many linear segments and can be computed in time $\mathcal{O}(|\mathcal{U}|n\log(|\mathcal{U}|n))$ using [Her89]. Finally, the sum of this piecewise linear function with the one corresponding to the leader's items has to be computed and a maximum of the resulting function has to be found. Since the total number of vertices is bounded by $\mathcal{O}(|\mathcal{U}|n\log(|\mathcal{U}|n))$, this is also possible within the total running time of $\mathcal{O}(|\mathcal{U}|n\log(|\mathcal{U}|n))$. □

Remark 4.14. The running times stated in Theorem 3.44 for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) and in Theorem 4.13 for the Robust Bilevel Continuous Knapsack Problem are the same, while the computation

of the pointwise minimum is more challenging here. In the proof of Theorem 3.44, we could consider each section between two subsequent integer values of b_l separately, computing the pointwise minimum of $|\mathcal{U}|$ many linear segments for each of them. This was possible because all items have the same size in case of the Continuous Selection Problem, leading to the vertices of all functions being at $\{b_l^-, \dots, b_l^+\}$ (or a subset thereof), which has cardinality $\mathcal{O}(n)$. In case of the Continuous Knapsack Problem, the items can have different sizes. Therefore, the structure of the set of vertices is more complicated and enumerating all integers in $\{b_l^-, \dots, b_l^+\}$ would not be polynomial in general; see also Section 4.3. In particular, the sets of values of b_l that define the vertices of the piecewise linear functions coming from the different scenarios in $\mathcal{U}$ might all be disjoint. Therefore, we consider all $\mathcal{O}(|\mathcal{U}|n)$ many linear segments simultaneously and determine the lower envelope of them using the general algorithm described in [Her89]. On the other hand, the proof of Theorem 3.44 only requires to solve a special case of the problem of determining the lower envelope of linear segments, namely the one where all linear segments are defined on the same range (here between two subsequent integers).

In summary, in the proof of Theorem 3.44, the pointwise minimum is computed in $\mathcal{O}(|\mathcal{U}|n \log(|\mathcal{U}|))$ time, while in the proof of Theorem 4.13, the pointwise minimum is computed in $\mathcal{O}(|\mathcal{U}|n \log(|\mathcal{U}|n))$ time. In fact, the latter running time can be reduced to $\mathcal{O}(|\mathcal{U}|n \log(|\mathcal{U}|))$ as well, using [Bla+24, Theorem 7]. However, both total running times stated in Theorems 3.44 and 4.13 still contain the factor $\log(n)$ in general, due to the sorting of the items in the beginning of the algorithm.

Moreover, note that the number of vertices of the lower envelope of k many linear segments is not only bounded by $\mathcal{O}(k \log(k))$ (which we used in the proof of Theorem 4.13), but even by $\mathcal{O}(k\,\alpha(k))$, where α is the very slowly growing inverse of Ackermann's function [HS86]. However, using this result would not improve the total running time.

4.4.2 Interval Uncertainty

We now consider interval uncertainty sets, i.e., sets of the form $\mathcal{U} = \prod_{e \in \mathcal{E}_f} [d^-(e), d^+(e)]$ with given functions $d^-, d^+ \in \mathbb{Q}^{\mathcal{E}_f}$ that satisfy $d^-(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$. Also for interval uncertainty, the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) could be solved in polynomial time; see Section 3.4.1.2.2. Again, these results can be generalized to the Robust Bilevel Continuous Knapsack Problem. In the following, we will first generalize the technical notions around interval orders and then present the polynomial-time algorithms for the adversary's problem (Algorithm 4.2) and for the leader's problem (Algorithm 4.3).

Recall Definitions 3.29 and 3.45 about partial orders, prefixes, and fractional prefixes, which we discussed in the context of the Robust Bilevel Selection Problem under Assumption (disj), and with interval uncertainty. We will work with these notions again in the following. Instead of the cardinality of a fractional prefix (see Definition 3.45), we will now need the size of a fractional prefix:

Definition 4.15. Let $(\mathcal{E}, \prec)$ be a partially ordered set and let $a\colon \mathcal{E} \to \mathbb{Q}_{>0}$ be given item sizes. The *size* of a fractional prefix $F \in \mathcal{F}_\prec$ is then defined as $a(F) := \sum_{e \in \mathcal{E}} a(e)(f_\prec(F))_e$.

A special type of partial orders, namely interval orders (see Definition 3.30), has been useful for describing structural insights about interval uncertainty. For the Robust Bilevel Selection Problem, we have shown in Lemmas 3.33 and 3.46 that the follower's greedy orders that can arise from the scenarios in an interval uncertainty set can be described in

terms of an interval order, and that the sets of follower's optimum solutions can be described in terms of (fractional) prefixes of this interval order. We will show a corresponding result for the Robust Bilevel Continuous Knapsack Problem in Lemma 4.18 below. For this, we first need to introduce a few more adjusted notions.

In Sections 3.4.1.1.2 and 3.4.1.2.2 about the Robust Bilevel Selection Problem, we have considered the interval order $\prec_{\mathcal{U}}$ corresponding to an interval uncertainty set $\mathcal{U} = \prod_{e\in\mathcal{E}_f}[d^-(e), d^+(e)]$; see Definition 3.31 and Lemma 3.32. In the Robust Bilevel Continuous Knapsack Problem, the follower's greedy orders depend not only on the follower's item values and therefore the uncertainty set $\mathcal{U}$, but also on the item sizes a. Indeed, the follower, solving his Continuous Knapsack Problem, sorts the items according to their profits $d(e)/a(e)$, for $e \in \mathcal{E}_f$; see Section 4.2.1. This leads to Definition 4.16 of $\prec_{\mathcal{U},a}$ below.

Similarly to Sections 3.4.1.1.2 and 3.4.1.2.2, we make an assumption in the following that simplifies the presentation. We assume that the intervals $\left[\frac{d^-(e)}{a(e)}, \frac{d^+(e)}{a(e)}\right]$ of possible follower's profits, for $e \in \mathcal{E}_f$, have no one-point intersections, i.e., that $\frac{d^-(e_1)}{a(e_1)} \neq \frac{d^+(e_2)}{a(e_2)}$ holds for all $e_1, e_2 \in \mathcal{E}_f$. If there are one-point intersections, then some technical complications related to the optimistic and the pessimistic setting arise. These were discussed in detail for the Robust Bilevel Selection Problem in Remark 3.36, and they are very similar for the Robust Bilevel Continuous Knapsack Problem. In particular, if one-point intersections are allowed, the algorithms as presented below are still correct for the pessimistic setting, while slight modifications are necessary in the optimistic setting, analogously to the ones described in Remark 3.36.

Definition 4.16. Given an interval uncertainty set $\mathcal{U} = \prod_{e\in\mathcal{E}_f}[d^-(e), d^+(e)]$ and item sizes $a\colon \mathcal{E}_f \to \mathbb{Q}_{>0}$, we define a binary relation $\prec_{\mathcal{U},a}$ on $\mathcal{E}_f$ such that, for $e_1, e_2 \in \mathcal{E}_f$, we have $e_1 \prec_{\mathcal{U},a} e_2$ if and only if $\frac{d^-(e_1)}{a(e_1)} > \frac{d^+(e_2)}{a(e_2)}$.

Analogously to Lemma 3.32, we then have:

Lemma 4.17. *The relation $\prec_{\mathcal{U},a}$ in Definition 4.16 is an interval order.*

Very similarly to Lemmas 3.33 and 3.46 for the Robust Bilevel Selection Problem under Assumption (disj), and with interval uncertainty, we can now show the following formal statement for the Robust Bilevel Continuous Knapsack Problem with interval uncertainty. Note that the main difference to the setting of the Robust Bilevel Selection Problem is that the follower's greedy orders do not depend on the intervals $[d^-(e), d^+(e)]$ of follower's item values directly, as they are given by the uncertainty set $\mathcal{U}$, but instead on the intervals $\left[\frac{d^-(e)}{a(e)}, \frac{d^+(e)}{a(e)}\right]$ of follower's item profits, which are scaled by the item sizes a.

Lemma 4.18. *In the Robust Bilevel Continuous Knapsack Problem with an interval uncertainty set $\mathcal{U}$, the set of greedy orders of the follower's items that arise from the scenarios in $\mathcal{U}$ is exactly the set of linear extensions of the interval order $\prec_{\mathcal{U},a}$ on $\mathcal{E}_f$. Moreover, the set of follower's optimum solutions, given a fixed feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$ and any scenario in $\mathcal{U}$, is exactly the set $\{f_{\prec_{\mathcal{U},a}}(F) \mid F \in \mathcal{F}_{\prec_{\mathcal{U},a}}, a(F) = b - \sum_{e\in\mathcal{E}_l} a(e)x_e\}$.*

Proof. Given the item sizes a and fixed leader's and follower's item values c and d, the follower solves a Continuous Knapsack Problem. To this end, he sorts his items

greedily by their profits, as in Line 2 of Algorithm 4.1. Now let an interval uncertainty set $\mathcal{U} = \prod_{e\in\mathcal{E}_f}[d^-(e), d^+(e)]$ be given. For every follower's item $e \in \mathcal{E}_f$, this defines an interval $\left[\frac{d^-(e)}{a(e)}, \frac{d^+(e)}{a(e)}\right]$ of possible follower's profits. If two such intervals are disjoint, then the order of the corresponding two items in the greedy order is the same for every scenario $d \in \mathcal{U}$. Considering Definition 4.16 of $\prec_{\mathcal{U},a}$, this shows that every follower's greedy order is a linear extension of $\prec_{\mathcal{U},a}$.

For the other direction, we argue that the adversary can achieve every linear extension $\prec_L$ of $\prec_{\mathcal{U},a}$ as the unique follower's greedy order by an appropriate choice of $d \in \mathcal{U}$. We define $D = \min_{e_1,e_2\in\mathcal{E}_f}\left|\frac{d^+(e_1)}{a(e_1)} - \frac{d^-(e_2)}{a(e_2)}\right|$ and $\varepsilon = D/n_f$. Note that D and ε are positive because we assume that the intervals have no one-point intersections. We iterate over the items in $\mathcal{E}_f$ in the order given by $\prec_L$. For the first item $e \in \mathcal{E}_f$, we set $d(e) = d^+(e)$. Now consider another iteration that deals with an item $e \in \mathcal{E}_f$, and let $e' \in \mathcal{E}_f$ be the item from the previous iteration. If $\frac{d^+(e)}{a(e)} < \frac{d(e')}{a(e')}$, then set $d(e) = d^+(e)$. Otherwise, set $d(e) = (d(e')/a(e') - \varepsilon) \cdot a(e)$. Note that $d(e) \geq d^-(e)$ holds due to the definition of ε. Hence, this defines follower's item values $d \in \mathcal{U}$ such that the corresponding profits $d(e)/a(e)$ are sorted decreasingly in the order of $\prec_L$. This finishes the proof of the first statement.

For the second statement, let a fixed feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$ be given. For any scenario $d \in \mathcal{U}$, the follower sorts his items according to a linear extension $\prec_{d,a}$ of $\prec_{\mathcal{U},a}$ that corresponds to d, and then (fractionally) selects the first items of this order until their total size $\sum_{e\in\mathcal{E}_f} a(e)y'_e$ has reached the capacity $b_f = b - \sum_{e\in\mathcal{E}_l} a(e)x_e$ that remains after the leader's decision. In other words, the follower's optimum solution y' is given by $f_{\prec_{d,a}}(F)$ where F is the unique fractional prefix $F \in \mathcal{F}_{\prec_{d,a}}$ with $a(F) = b - \sum_{e\in\mathcal{E}_l} a(e)x_e$. Finally, recall that the fractional prefixes of the partial order $\prec_{\mathcal{U},a}$ are exactly the fractional prefixes of all possible linear extensions $\prec_{d,a}$ of $\prec_{\mathcal{U},a}$, which concludes the proof. □

We can now generalize Algorithm 3.3, which solves the adversary's problem of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with interval uncertainty, to the setting of the Robust Bilevel Continuous Knapsack Problem. This results in Algorithm 4.2 and Theorem 4.19. Note that the proof of Theorem 4.19 is almost exactly the same as the one of Theorem 3.47 and only requires to include the item sizes in a straightforward manner, given the adjusted definitions above. For the sake of completeness, we still include the entire proof below.

Recall that the idea of the algorithms for the Robust Bilevel Selection Problem with interval uncertainty was based on the algorithm of [Woe03] for the precedence constraint knapsack problem; see Section 3.4.1.1.2 for more details. In Algorithms 3.2 and 3.3, we used a subroutine for solving a Selection Problem and a Continuous Selection Problem, respectively. In Algorithm 4.2, a subroutine for solving a Continuous Knapsack Problem is required instead. Its input consists of a finite item set, the items' sizes, a capacity, and the items' values; see also Section 4.1. Note that the subroutine is called (in Line 7) with the negated item values $-c$, i.e., it actually solves the minimization variant of the Continuous Knapsack Problem with the original item values c. This aligns with the adversary's goal of minimizing the leader's objective function value. Further note that the conditions in Line 6 ensure that the instance passed to the subroutine in Line 7 is feasible in the sense that the capacity can be filled completely. We may assume that the subroutine packs greedily and, in particular, that it always returns a solution with at most one nonbinary value; see also Remark 4.1 and Section 4.2.1. This assumption ensures that the computation of d in Line 12 is well-defined.

Algorithm 4.2. Algorithm for the adversary's problem of the Robust Bilevel Continuous Knapsack Problem with interval uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_f \neq \emptyset$ and $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $a\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}_{>0}$, $b \in \mathbb{Q} \cap [0, a(\mathcal{E}_l \cup \mathcal{E}_f)]$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, an interval uncertainty set $\mathcal{U}$ given by $d^-, d^+\colon \mathcal{E}_f \to \mathbb{Q}$ with $d^-(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$, a feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$

Output : an optimum adversary's solution $d \in \mathcal{U}$ (i.e., $d\colon \mathcal{E}_f \to \mathbb{R}$ with $d^-(e) \leq d(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$) of the Robust Bilevel Continuous Knapsack Problem

1 $b_f := b - \sum_{e \in \mathcal{E}_l} a(e)x_e$

2 $\overline{\mathcal{E}} := \emptyset$

3 **for** $\overline{e} \in \mathcal{E}_f$ **do**

4 $\quad \mathcal{E}_{\overline{e}}^- := \left\{ e \in \mathcal{E}_f \mid \frac{d^-(e)}{a(e)} > \frac{d^+(\overline{e})}{a(\overline{e})} \right\}$

5 $\quad \mathcal{E}_{\overline{e}}^0 := \left\{ e \in \mathcal{E}_f \mid \frac{d^-(e)}{a(e)} \leq \frac{d^+(\overline{e})}{a(\overline{e})} \leq \frac{d^+(e)}{a(e)} \right\}$

6 $\quad$ **if** $a(\mathcal{E}_{\overline{e}}^-) \leq b_f$ *and* $a(\mathcal{E}_{\overline{e}}^0) \geq b_f - a(\mathcal{E}_{\overline{e}}^-)$ **then**

7 $\quad\quad y^{\overline{e},0} :=$ Continuous Knapsack Problem$(\mathcal{E}_{\overline{e}}^0, a \restriction \mathcal{E}_{\overline{e}}^0, b_f - a(\mathcal{E}_{\overline{e}}^-), -c \restriction \mathcal{E}_{\overline{e}}^0)$

8 $\quad\quad$ define $y^{\overline{e}} \in [0,1]^{\mathcal{E}_f}$ by $y_e^{\overline{e}} := 1$ for all $e \in \mathcal{E}_{\overline{e}}^-$, $y_e^{\overline{e}} := y_e^{\overline{e},0}$ for all $e \in \mathcal{E}_{\overline{e}}^0$, and $y_e^{\overline{e}} := 0$ for all $e \in \mathcal{E}_f \setminus (\mathcal{E}_{\overline{e}}^- \cup \mathcal{E}_{\overline{e}}^0)$

9 $\quad\quad \overline{\mathcal{E}} := \overline{\mathcal{E}} \cup \{\overline{e}\}$

10 select $\overline{e} \in \operatorname{argmin} \left\{ \sum_{e' \in \mathcal{E}_f} c(e') y_{e'}^e \mid e \in \overline{\mathcal{E}} \right\}$ arbitrarily

11 $\varepsilon := \frac{1}{2} \min_{e \in \mathcal{E}_{\overline{e}}^0} \left(\frac{d^+(\overline{e})}{a(\overline{e})} - \frac{d^-(e)}{a(e)} \right)$

12 **return** $d\colon \mathcal{E}_f \to \mathbb{Q}$ *with* $d(e) := d^+(e)$ *for all* $e \in \{e' \in \mathcal{E}_f \mid y_{e'}^{\overline{e}} = 1\}$, $d(e) := d^-(e)$ *for all* $e \in \{e' \in \mathcal{E}_f \mid y_{e'}^{\overline{e}} = 0\}$, *and* $d(e) := (\frac{d^+(\overline{e})}{a(\overline{e})} - \varepsilon) \cdot a(e)$ *for the unique (if it exists)* $e \in \mathcal{E}_f$ *with* $y_e^{\overline{e}} \in (0,1)$

Theorem 4.19. *For any fixed feasible leader's solution of the Robust Bilevel Continuous Knapsack Problem with interval uncertainty, Algorithm 4.2 solves the adversary's problem in time* $\mathcal{O}(n^2)$.

Proof. First note that we can make the assumption $\mathcal{E}_f \neq \emptyset$ for the input without loss of generality because the adversary and the follower have nothing to decide otherwise. We first assume that $\overline{\mathcal{E}} \neq \emptyset$ holds in the end of the algorithm, and see that this is indeed true in the end of this proof. Let $\overline{e}$ and d be defined as in Lines 10 and 12, respectively.

We claim that $y^{\overline{e}}$ is the optimum follower's solution when the leader's solution x and the adversary's choice d are fixed. Similarly to Algorithm 3.3 and the proof of Theorem 3.47, the sets $\mathcal{E}_{\overline{e}}^-$ and $\mathcal{E}_{\overline{e}}^0$ defined in Lines 4 and 5, respectively, are the set of predecessors of $\overline{e}$ and an appropriate set of items that are incomparable to $\overline{e}$ and to each other, with respect to the interval order $\prec_{\mathcal{U},a}$.

By the definition of $y^{\overline{e}}$ in Line 10 and by the correctness of the subroutine for the Continuous Knapsack Problem we call in Line 7, we may assume that $y^{\overline{e}}$ corresponds (in the sense of $f_{\prec_{\mathcal{U},a}}$; see Definition 3.45) to a fractional prefix of $\prec_{\mathcal{U},a}$ of size $\sum_{e \in \mathcal{E}_f} a(e) y_e^{\overline{e}} = b_f$. By Lemma 4.18, the adversary can enforce the follower's solution $y^{\overline{e}}$ by choosing an appropriate scenario from $\mathcal{U}$. However, similarly to Algorithm 3.3, this can be done in an

easier way than the one described in Lemma 4.18 because the adversary does not need to enforce a fixed linear extension, but only some linear extension having the given fractional prefix. In particular, this is achieved by the choice $d \in \mathcal{U}$ that is made in Line 12 because it ensures that both $\{e \in \mathcal{E}_f \mid y_e^{\overline{e}} = 1\}$ and $\{e \in \mathcal{E}_f \mid y_e^{\overline{e}} > 0\}$ are prefixes of the corresponding linear extension of $\prec_{\mathcal{U},a}$.

We now prove that the returned adversary's solution d is optimum. First observe that, in case $b_f \in \{0, a(\mathcal{E}_f)\}$, any adversary's solution is optimum because there is only a single feasible follower's solution, namely $y_e = 0$ for all $e \in \mathcal{E}_f$ or $y_e = 1$ for all $e \in \mathcal{E}_f$, respectively. For the rest of the proof, we assume that we are in none of these cases. Let $y \in [0,1]^{\mathcal{E}_f}$ be an optimum follower's solution corresponding to an optimum adversary's choice and let $\overline{e} \in \{e \in \mathcal{E}_f \mid y_e > 0\}$ such that $d^+(\overline{e})/a(\overline{e})$ is minimum. By Lemma 4.18, y is the image of a fractional prefix $(\mathcal{E}_y, e_y, \lambda_y)$ of $\prec_{\mathcal{U},a}$ under $f_{\prec_{\mathcal{U},a}}$. By the properties of a fractional prefix, we have $\mathcal{E}_{\overline{e}}^- \subseteq \mathcal{E}_y$, and by the choice of $\overline{e}$, we have $\mathcal{E}_y \setminus \mathcal{E}_{\overline{e}}^- \subseteq \mathcal{E}_{\overline{e}}^0$. As y is also a feasible follower's solution, we have $a(\mathcal{E}_y, e_y, \lambda_y) = b_f$, and hence, $f_{\prec_{\mathcal{U},a} \restriction \mathcal{E}_{\overline{e}}^0 \times \mathcal{E}_{\overline{e}}^0}(\mathcal{E}_y \setminus \mathcal{E}_{\overline{e}}^-, e_y, \lambda_y)$ is a feasible solution for the Continuous Knapsack Problem in Line 7. It must be optimum as well because a better adversary's solution could be constructed from it otherwise, analogously to the proof of Theorem 3.34. Thus, y is the follower's solution that is enforced in iteration $\overline{e}$ or has the same leader's objective function value as it. This proves the optimality of the returned solution and also shows that $\overline{\mathcal{E}} \neq \emptyset$ is satisfied in the end of the algorithm.

Analogously to Theorem 3.34, the running time is given by $|\mathcal{E}_f| \in \mathcal{O}(n)$ iterations each of which can be implemented to run in linear time. □

We now turn to the leader's problem. Again, we can generalize Algorithm 3.4, which solves the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with interval uncertainty, to the setting of the Robust Bilevel Continuous Knapsack Problem, resulting in Algorithm 4.3. Recall the description of the leader's objective function, as a function in the parameter b_l, from Section 4.3 and the beginning of Section 4.4. Algorithm 4.3 gives a polynomial-time approach for determining this function; see also Remark 4.21 below. In Lines 3 and 11, we compute piecewise linear functions of the types PLF′ and $\overline{\text{PLF}}'$, as introduced in Section 4.3; see also Figure 4.1.

The correctness of Algorithm 4.3 can be shown as follows. Again, the proof is very close to the one of Theorem 3.48, but still included for the sake of completeness.

Theorem 4.20. *Algorithm 4.3 solves the Robust Bilevel Continuous Knapsack Problem with interval uncertainty in time* $\mathcal{O}(n^2 \log n)$.

Proof. We can again make the assumption $\mathcal{E}_f \neq \emptyset$ for the input without loss of generality: Otherwise, assuming a feasible instance, we have $b_l^- = b_l^+ = b$, so the returned value Continuous Knapsack Problem$(\mathcal{E}_l, a \restriction \mathcal{E}_l, b, c \restriction \mathcal{E}_l)$ is clearly the leader's optimum solution.

First note that the function g_l computed in Line 3 is well-defined because $0 \leq b_l^- \leq b_l^+ \leq a(\mathcal{E}_l)$, and defined on the range $[b_l^-, b_l^+]$. This is the range of possible capacities b_l that the leader can use in a feasible solution. The function g_l describes the part of the leader's objective function that represents the optimum value she can achieve by a selection of her own items when using a capacity of b_l. We refer to Section 4.3 and the beginning of Section 4.4 for more details. For the optimum value $b_l^* \in [b_l^-, b_l^+]$, an optimum leader's solution is given by Continuous Knapsack Problem$(\mathcal{E}_l, a \restriction \mathcal{E}_l, b_l^*, c \restriction \mathcal{E}_l)$, as returned

Algorithm 4.3. Algorithm for the leader's problem of the Robust Bilevel Continuous Knapsack Problem with interval uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_f \neq \emptyset$ and $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $a\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}_{>0}$, $b \in \mathbb{Q} \cap [0, \dots, a(\mathcal{E}_l \cup \mathcal{E}_f)]$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, an interval uncertainty set $\mathcal{U}$ given by $d^-, d^+\colon \mathcal{E}_f \to \mathbb{Q}$ with $d^-(e) \leq d^+(e)$ for all $e \in \mathcal{E}_f$

Output : an optimum leader's solution $x \in [0,1]^{\mathcal{E}_l}$ of the Robust Bilevel Continuous Knapsack Problem

1 $b_l^- := \max\{0, b - a(\mathcal{E}_f)\}$, $b_f^+ := b - b_l^-$

2 $b_l^+ := \min\{b, a(\mathcal{E}_l)\}$, $b_f^- := b - b_l^+$

3 compute $g_l := \mathrm{PLF}'(\mathcal{E}_l, a \restriction \mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$

4 $\overline{\mathcal{E}} := \emptyset$

5 **for** $\overline{e} \in \mathcal{E}_f$ **do**

6 $\quad \mathcal{E}_{\overline{e}}^- := \left\{ e \in \mathcal{E}_f \mid \frac{d^-(e)}{a(e)} > \frac{d^+(\overline{e})}{a(\overline{e})} \right\}$

7 $\quad \mathcal{E}_{\overline{e}}^0 := \left\{ e \in \mathcal{E}_f \mid \frac{d^-(e)}{a(e)} \leq \frac{d^+(\overline{e})}{a(\overline{e})} \leq \frac{d^+(e)}{a(e)} \right\}$

8 $\quad$ **if** $a(\mathcal{E}_{\overline{e}}^-) \leq b_f^+$ *and* $a(\mathcal{E}_{\overline{e}}^0) \geq b_f^- - a(\mathcal{E}_{\overline{e}}^-)$ **then**

9 $\quad\quad \overline{b_f^-} := \max\{0, b_f^- - a(\mathcal{E}_{\overline{e}}^-)\}$

10 $\quad\quad \overline{b_f^+} := \min\{b_f^+ - a(\mathcal{E}_{\overline{e}}^-), a(\mathcal{E}_{\overline{e}}^0)\}$

11 $\quad\quad$ compute $g_f^{\overline{e}} := \overline{\mathrm{PLF}}'(\mathcal{E}_{\overline{e}}^0, a \restriction \mathcal{E}_{\overline{e}}^0, -c \restriction \mathcal{E}_{\overline{e}}^0, c \restriction \mathcal{E}_{\overline{e}}^0, b - \overline{b_f^+}, b - \overline{b_f^-}, b)$

12 $\quad\quad$ compute $g_f^{\overline{e}} := g_f^{\overline{e}} + (-a(\mathcal{E}_{\overline{e}}^-), c(\mathcal{E}_{\overline{e}}^-))$ (i.e., shift the function $g_f^{\overline{e}}$ by $-a(\mathcal{E}_{\overline{e}}^-)$ in x direction and by $c(\mathcal{E}_{\overline{e}}^-)$ in y direction)

13 $\quad\quad \overline{\mathcal{E}} := \overline{\mathcal{E}} \cup \{\overline{e}\}$

14 compute $g_f := \min\{g_f^{\overline{e}} \mid \overline{e} \in \overline{\mathcal{E}}\}$ (i.e., the pointwise minimum)

15 compute $g := g_l + g_f$

16 compute some $b_l^* \in \operatorname{argmax}\{g(b_l) \mid b_l \in [b_l^-, b_l^+]\}$

17 **return** Continuous Knapsack Problem$(\mathcal{E}_l, a \restriction \mathcal{E}_l, b_l^*, c \restriction \mathcal{E}_l)$

by the algorithm in Line 17. It remains to show that the value b_l^* derived by the algorithm is indeed optimum.

We claim that, for every fixed value $b_l \in [b_l^-, b_l^+]$, the function g computed in Line 15 correctly describes the best leader's objective function value that is possible if the leader uses a capacity of b_l and the follower uses a capacity of $b_f = b - b_l$. By the considerations about g_l above, this is equivalent to the function g_f defined in Line 14 correctly describing the best objective function values the adversary can achieve for fixed values of b_l. Hence, we claim that, for every fixed value $b_l \in [b_l^-, b_l^+]$, the value of the adversary's solution returned by Algorithm 4.2, with any given feasible leader's solution x using capacity b_l, is precisely $g_f(b_l)$. Recall that, from the adversary's and the follower's point of view, only the capacity b_l occupied by the leader's solution is relevant, but not the precise leader's solution x. The correctness of Algorithm 4.3 then follows from the correctness of Algorithm 4.2, which we proved in Theorem 4.19.

Let $b_l \in [b_l^-, b_l^+]$ be a fixed leader's capacity and $b_f = b - b_l \in [b_f^-, b_f^+]$ the corresponding follower's capacity. Fix some item $\overline{e} \in \mathcal{E}_f$ and consider the corresponding iterations of the loop in Line 3 of Algorithm 4.2 and of the loop in Line 5 of Algorithm 4.3. The sets

$\mathcal{E}_{\overline{e}}^-$ and $\mathcal{E}_{\overline{e}}^0$ are defined in the same way in both algorithms. It can be checked that the conditions in Line 6 of Algorithm 4.2 are satisfied for b_f if and only if the conditions in Line 8 of Algorithm 4.3 are satisfied and the function $g_f^{\overline{e}}$ is defined in $b - b_f$, in the end of the iteration. For the latter, note that the range of the function $g_f^{\overline{e}}$ according to Lines 11 and 12, using the definitions of $\overline{b_f^-}$ and $\overline{b_f^+}$ in Lines 9 and 10, is

$$\begin{aligned}&\left[b - \overline{b_f^+} - a(\mathcal{E}_{\overline{e}}^-), b - \overline{b_f^-} - a(\mathcal{E}_{\overline{e}}^-)\right]\\ &= \left[\max\{b - b_f^+, b - a(\mathcal{E}_{\overline{e}}^0) - a(\mathcal{E}_{\overline{e}}^-), \min\{b - a(\mathcal{E}_{\overline{e}}^-), b - b_f^-\}\right],\end{aligned}$$

and that this range contains $b - b_f$ for some given $b_f \in [b_f^-, b_f^+]$ if and only if $b_f \in [a(\mathcal{E}_{\overline{e}}^-), a(\mathcal{E}_{\overline{e}}^0) + a(\mathcal{E}_{\overline{e}}^-)]$ is satisfied, which is exactly the condition in Line 6 of Algorithm 4.2. Hence, either in both or in none of the two algorithms, the case of b_f and $\overline{e}$ contributes to the computations after the loop. We now demonstrate that, if the above conditions are satisfied, then the values contributed by this case are also the same in both algorithms. The corresponding follower's solution $y^{\overline{e}}$ defined in Line 8 of Algorithm 4.2 is given by selecting all items in $\mathcal{E}_{\overline{e}}^-$ and a part of the items in $\mathcal{E}_{\overline{e}}^0$ according to the Continuous Knapsack Problem with the values $-c$ and the remaining capacity $b_f - a(\mathcal{E}_{\overline{e}}^-)$ (see Line 7). The total value $\sum_{e \in \mathcal{E}_f} c(e) y_e^{\overline{e}}$ of this solution is then considered in the minimum in Line 10. The function $g_f^{\overline{e}}$ in Algorithm 4.3 describes exactly the same: The piecewise linear function defined in Line 11 describes the total value c of the follower's solutions that are solutions of a Continuous Knapsack Problem on $\mathcal{E}_{\overline{e}}^0$ with values $-c$, depending on the capacity $b_l = b - b_f$ that the leader uses (see also Section 4.3), and the shift in Line 12 adjusts this function corresponding to the additional selection of all items in $\mathcal{E}_{\overline{e}}^-$. Therefore, the value $g_f^{\overline{e}}(b_l) = g_f^{\overline{e}}(b - b_f)$ considered in the pointwise minimum in Line 14 is exactly the same as the value of the solution described above that contributes to the minimum in Line 10 of Algorithm 4.2. This implies that, for every $b_l \in [b_l^-, b_l^+]$, the value $g_f(b_l)$ is exactly the leader's value of an optimum follower's solution corresponding to an optimum adversary's decision, as returned by Algorithm 4.2. This concludes the proof of the correctness of Algorithm 4.3.

For the running time, observe that each of the piecewise linear functions g_l (Line 3) and $g_f^{\overline{e}}$ (Line 11) can be computed in time $\mathcal{O}(n \log n)$; see Section 4.3. The pointwise minimum of the at most n piecewise linear functions $g_f^{\overline{e}}$ (Line 14), each consisting of at most n linear segments, can be computed in time $\mathcal{O}(n^2 \log n)$ using [Her89], similarly to several of the previous algorithms in this thesis; see, e.g., Theorem 4.13 and Remark 4.14. Again, like, e.g., in the proof of Theorem 4.13, the sum with the piecewise linear function g_l with at most n linear segments and the maximum of the resulting piecewise linear function (Lines 15 and 16) can be computed in time $\mathcal{O}(n^2 \log n)$. □

Remark 4.21. Analogously to Remark 3.49, note that the function g computed by Algorithm 4.3 is the piecewise linear function that is described in the beginning of Section 4.4, consisting of the best leader's objective function values that can be achieved for any fixed $b_l \in [b_l^-, b_l^+]$. Recalling Remark 4.11, it could also be determined by enumerating all possible follower's greedy orders (which here correspond to the linear extensions of the interval order $\prec_{\mathcal{U},a}$ by Lemma 4.18), but these could be exponentially many. The polynomial running time of Algorithm 4.3 also implies that the function consists of only polynomially many linear segments. On the other hand, the continuity of the function g clearly follows from the description of the same function in the beginning of Section 4.4.

4.4.3 Discrete Uncorrelated Uncertainty

We now consider the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty, i.e., with an uncertainty set of the form $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$ with finite sets $\mathcal{U}_e \subseteq \mathbb{Q}$ for all $e \in \mathcal{E}_f$. Hence, for every item, there is a finite number of possible follower's item values the adversary can choose from, and these are independent of each other. While the corresponding Robust Bilevel Selection Problem under Assumptions (disj) and (cc) was shown to be solvable in polynomial time in Section 3.4.1.2.3, it turns out that the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty is $\mathcal{NP}$-hard in general, even if we consider the special case where $|\mathcal{U}_e| = 2$ holds for all $e \in \mathcal{E}_f$. We will prove this in Section 4.4.3.1, both for the leader's and the adversary's problem; see Corollaries 4.23 and 4.29. In Section 4.4.3.2, we will show that we can still transfer the algorithmic results that we presented in Section 3.4.1.2.3 for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) to the Robust Bilevel Continuous Knapsack Problem. However, the running times of the resulting algorithms are not polynomial, but only pseudopolynomial; see Theorems 4.30 and 4.31. Hence, the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty is weakly $\mathcal{NP}$-hard.

4.4.3.1 $\mathcal{NP}$-Hardness

We show the $\mathcal{NP}$-hardness result for the problem (RBCKP-b_f); see Theorem 4.22. Recall that this problem is equivalent to the Robust Bilevel Continuous Knapsack Problem under Assumption (uni′), a special case of the Robust Bilevel Continuous Knapsack Problem, according to Lemma 4.10. Hence, the result stated in Corollary 4.23 directly follows from Theorem 4.22.

Theorem 4.22. *The problem stated in (RBCKP-b_f), with a discrete uncorrelated uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$, is $\mathcal{NP}$-hard, even if $|\mathcal{U}_e| = 2$ for all $e \in \mathcal{E}_f$.*

Corollary 4.23. *The Robust Bilevel Continuous Knapsack Problem with a discrete uncorrelated uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$ is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′) and $|\mathcal{U}_e| = 2$ for all $e \in \mathcal{E}_f$.*

Our goal in the following is to prove Theorem 4.22. We do this by a reduction from the Subset Sum Problem, which is well-known to be $\mathcal{NP}$-complete; see Section 4.1.

Let an instance of the Subset Sum Problem be given, consisting of the numbers $m \in \mathbb{N}$ and $w_1, \dots, w_m, W \in \mathbb{N}$. Without loss of generality, we may assume $1 \leq W \leq \sum_{i=1}^{m} w_i - 1$ because the problem is trivial to solve otherwise. We show that we can decide in polynomial time whether there is a subset $S \subseteq \{1, \dots, m\}$ with $\sum_{i \in S} w_i = W$ if the following instance of (RBCKP-b_f) can be solved in polynomial time: Define $\varepsilon := 1/4$ and $M := \sum_{i=1}^{m} w_i + \varepsilon$. Let $\mathcal{E}_f = \{e_0, \dots, e_{m+1}\}$, set the item sizes as $a(e_0) = \varepsilon$, $a(e_{m+1}) = M$, and $a(e_i) = w_i$ for $i \in \{1, \dots, m\}$, the leader's item values as $c(e_0) = -M$, $c(e_{m+1}) = \varepsilon$, and $c(e_i) = -w_i$ for $i \in \{1, \dots, m\}$, and the capacity bounds as $b_f^- = W$ and $b_f^+ = W + 2\varepsilon$. For the uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$, define $d^-(e_i) = (i+1) \cdot a(e_i)$, $d^+(e_i) = (m+i+3) \cdot a(e_i)$, and $\mathcal{U}_{e_i} = \{d^-(e_i), d^+(e_i)\}$ for all $i \in \{0, \dots, m+1\}$. This leads to

$$\frac{d^-(e_0)}{a(e_0)} < \dots < \frac{d^-(e_{m+1})}{a(e_{m+1})} < \frac{d^+(e_0)}{a(e_0)} < \dots < \frac{d^+(e_{m+1})}{a(e_{m+1})}.$$

These values are the possible follower's profits and their order is important because the follower, when solving his Continuous Knapsack Problem, sorts the items by the profits that are determined by the adversary's choice $d \in \mathcal{U}$. Note that, since all possible profits are distinct, the optimistic and the pessimistic problem variants do not have to be distinguished for this instance.

In the following lemma, we investigate structural properties of optimum adversary's and follower's solutions in the constructed instance of (RBCKP-b_f). For this, we ignore the actual capacity bounds b_f^- and b_f^+ defined above and instead allow leader's decisions in a larger range, namely $b_f \in [\varepsilon, M)$. Note that $[b_f^-, b_f^+] \subsetneq [\varepsilon, M)$ and that extending the range of possible capacities does not change the behavior on the original range. However, studying the larger range will later be useful for understanding the two different behaviors that can occur in the actual range $[{b_f}^-, {b_f}^+]$, depending on whether the initial instance of the Subset Sum Problem is a yes-instance or a no-instance.

Lemma 4.24. *Let any leader's decision $b_f \in [\varepsilon, M)$ be given, let $d^* \in \mathcal{U}$ be an optimum adversary's solution given b_f, and let $y^* \in [0,1]^{\mathcal{E}_f}$ be an optimum follower's solution given b_f and d^*. Then the following hold:*

(a) *$d^*(e_0) = d^+(e_0)$ and $y^*_{e_0} > 0$,*
(b) *$d^*(e_{m+1}) = d^-(e_{m+1})$ and $y^*_{e_{m+1}} < 1$,*
(c) *for every $i \in \{1, \dots, m\}$, either $d^*(e_i) = d^+(e_i)$ and $y^*_{e_i} = 1$ or $d^*(e_i) = d^-(e_i)$ and $y^*_{e_i} = 0$.*

Proof. We first argue that the adversary can always enforce a follower's solution with a leader's value of less than $-M + \varepsilon$. Indeed, he can set $d(e_0) = d^+(e_0)$ and $d(e_i) = d^-(e_i)$ for all $i \in \{1, \dots, m+1\}$. Then the item e_0 appears first in the follower's greedy order and is therefore selected completely by the follower, i.e., the follower sets $y'_{e_0} = 1$, because $a(e_0) = \varepsilon \leq b_f$ by assumption. The item e_{m+1}, which is the second item in the follower's greedy order, cannot be selected completely by the follower, i.e., we have $y'_{e_{m+1}} < 1$, because $a(e_{m+1}) = M > b_f$. Hence, the capacity is completely filled after packing a fraction of e_{m+1}, and the follower sets $y'_{e_i} = 0$ for all $i \in \{1, \dots, m\}$. Thus, the optimum follower's solution y' corresponding to this adversary's choice d has a leader's value of $\sum_{i=0}^{m+1} c(e_i) y'_{e_i} < c(e_0) + c(e_{m+1}) = -M + \varepsilon$.

We can now show the statement in (a). Observe that, if the adversary sets $d(e_0) = d^-(e_0)$, then the item e_0 appears last in the follower's greedy order. Since $\sum_{i=1}^{m+1} a(e_i) > M > b_f$, this means that the follower does not select e_0, i.e., that the follower sets $y'_{e_0} = 0$. A follower's solution y' with $y'_{e_0} = 0$ has a leader's value of $\sum_{i=0}^{m+1} c(e_i) y'_{e_i} \geq \sum_{i=1}^{m} c(e_i) = -\sum_{i=1}^{m} w_i = -M + \varepsilon$. The first paragraph of this proof implies that this is not optimum from the adversary's perspective. Thus, if the adversary makes an optimum decision, then he sets $d^*(e_0) = d^+(e_0)$ and the resulting follower's optimum solution y^* satisfies $y^*_{e_0} > 0$.

Similarly, we can show the statement in (b). Note that $y'_{e_{m+1}} < 1$ holds for any feasible follower's solution y' because $a(e_{m+1}) = M > b_f$. If the adversary sets $d(e_{m+1}) = d^+(e_{m+1})$, then the item e_{m+1} is the first one in the follower's greedy order. As its size already exceeds the capacity, no other item is selected by the follower, i.e., the follower sets $y'_{e_i} = 0$ for all $i \in \{0, \dots, m\}$. The leader's objective value of such a follower's solution y' is $\sum_{i=0}^{m+1} c(e_i) y'_{e_i} = \varepsilon y'_{e_0} > 0$. Again, due to the first paragraph of this proof, this is clearly not optimum from the adversary's perspective, which shows that the adversary sets $d^*(e_{m+1}) = d^-(e_{m+1})$ in every optimum solution d^*.

Now we turn to (c) and let $i \in \{1, \dots, m\}$. If the adversary chooses $d^*(e_i) = d^+(e_i)$, then the item e_i precedes the item e_0 in the follower's greedy order. As we know from (a) that $y^*_{e_0} > 0$ holds in the optimum setting, this implies that the item e_i must be selected completely in the follower's greedy solution, i.e., that $y^*_{e_i} = 1$ holds. If the adversary chooses $d^*(e_i) = d^-(e_i)$, then the item e_i appears after the item e_{m+1} in the follower's greedy order. Since, by (b), $y^*_{e_{m+1}} < 1$ holds in the optimum setting, this implies that the item e_i cannot be selected in the follower's greedy solution, i.e., that $y^*_{e_i} = 0$ holds. □

The next lemma deals with the leader's objective function, which we denote by f for the remainder of the $\mathcal{NP}$-hardness proof. As in Lemma 4.24, we consider a large range $[\varepsilon, M)$ of possible leader's solutions. For each $b_f \in [\varepsilon, M)$, the value $f(b_f)$ corresponds to the minimum leader's value that the adversary can achieve, given the capacity b_f, and is described by Lemma 4.25.

Lemma 4.25. *Let $V_1, V_2 \in \mathbb{N}_0$ be two values with $V_1 < V_2$ that can arise as sums of subsets of $\{w_1, \dots, w_m\}$, such that there is no other such value V with $V_1 < V < V_2$. Then*

$$f(b_f) = \begin{cases} -M - V_1 + \frac{\varepsilon}{M}(b_f - V_1 - \varepsilon) & \text{if } b_f \in [V_1 + \varepsilon, V_2), \\ \min\{-M - V_1 + \frac{\varepsilon}{M}(b_f - V_1 - \varepsilon), -V_2 - \frac{M}{\varepsilon}(b_f - V_2)\} & \text{if } b_f \in [V_2, V_2 + \varepsilon). \end{cases}$$

Proof. Let $T \subseteq \{1, \dots, m\}$ be a subset with $\sum_{i \in T} w_i = V_1$. Using Lemma 4.24, we can describe the structure of the optimum adversary's and follower's solutions for all $b_f \in [V_1 + \varepsilon, V_2)$ as follows. The unique best choice of the adversary is to set $d(e_0) = d^+(e_0)$, $d(e_{m+1}) = d^-(e_{m+1})$, $d(e_i) = d^+(e_i)$ for all $i \in T$, and $d(e_i) = d^-(e_i)$ for all $i \in \{1, \dots, m\} \setminus T$, which leads to the follower selecting the items corresponding to T, item e_0, and a fraction of item e_{m+1}. Indeed, as $b_f < V_2$, no better packing of the items $e_1, \dots, e_m$ than T is possible, and since $b_f \geq V_1 + \varepsilon$, the most profitable item e_0 (from the adversary's perspective) can be added entirely without making the packing T infeasible. This leads to

$$f(b_f) = -M - V_1 + \frac{\varepsilon}{M}(b_f - V_1 - \varepsilon)$$

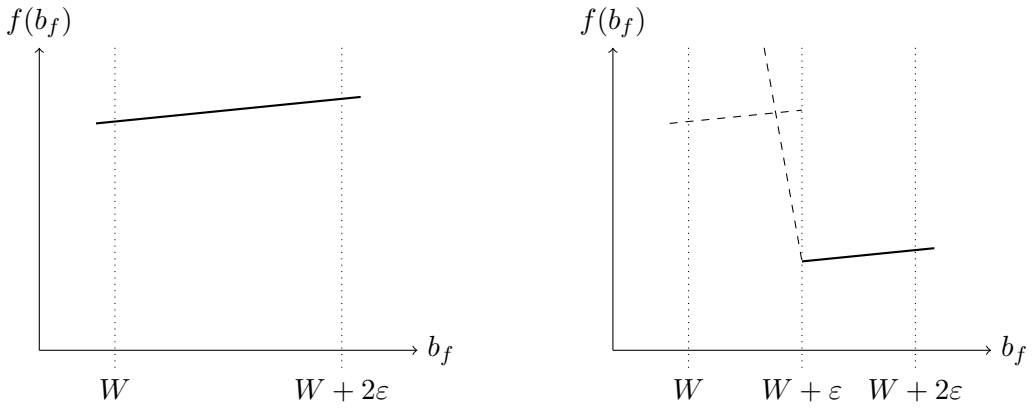
for $b_f \in [V_1 + \varepsilon, V_2)$.

If $b_f \in [V_2, V_2 + \varepsilon)$, the same adversary's solution d as above still leads to the same follower's solution, only with a slightly larger fraction of item e_{m+1}. However, since now a better packing $S \subseteq \{1, \dots, m\}$ with $\sum_{i \in S} w_i = V_2$ is available, we also have to consider the adversary's solution corresponding to S as a candidate for the optimum adversary's solution. More precisely, the adversary can set $d(e_0) = d^+(e_0)$, $d(e_{m+1}) = d^-(e_{m+1})$, $d(e_i) = d^+(e_i)$ for all $i \in S$, and $d(e_i) = d^-(e_i)$ for all $i \in \{1, \dots, m\} \setminus S$. Then the follower's optimum solution selects the items corresponding to S and a fraction of item e_0. This solution has a leader's value of

$$-V_2 - \frac{M}{\varepsilon}(b_f - V_2).$$

Among these two solutions, the adversary chooses the one with the smaller leader's value, which finishes the proof of the second case. □

Let $b^*_f \in [b^-_f, b^+_f] = [W, W + 2\varepsilon]$ be an optimum leader's solution in the original range, i.e., $b^*_f \in \operatorname{argmax}\{f(b_f) \mid b_f \in [W, W + 2\varepsilon]\}$. We conclude the proof of Theorem 4.22 by



(a) The case of a no-instance (Lemma 4.26) **(b)** The case of a yes-instance (Lemma 4.27)

Figure 4.3. The leader's objective function f in the instance of (RBCKP-b_f) constructed in the proof of Theorem 4.22

showing that the given instance of the Subset Sum Problem is a yes-instance if and only if $b_f^* \neq b_f^+$, which follows from the following two lemmas. The two situations arising in Lemmas 4.26 and 4.27 are illustrated in Figure 4.3.

Lemma 4.26. *If the given instance $(w_1, \dots, w_m, W)$ of the Subset Sum Problem is a no-instance, then $b_f^* = b_f^+$.*

Proof. In case of a no-instance, W cannot be written as a sum of a subset of $\{w_1, \dots, w_m\}$. Let $V_1, V_2 \in \mathbb{N}_0$ be the largest subset sum with $V_1 < W$ and the smallest subset sum with $V_2 > W$, respectively. Note that both of these numbers exist because we assume $1 \leq W \leq \sum_{i=1}^m w_i - 1$. Then we have $[b_f^-, b_f^+] = [W, W + 2\varepsilon] \subseteq [V_1 + \varepsilon, V_2)$. Hence, by Lemma 4.25, the function f is linear on $[b_f^-, b_f^+]$ with slope $\varepsilon/M > 0$. Thus, its maximum and therefore the optimum leader's solution is attained in $b_f^* = b_f^+$. □

Lemma 4.27. *If the given instance $(w_1, \dots, w_m, W)$ of the Subset Sum Problem is a yes-instance, then $b_f^* \neq b_f^+$.*

Proof. In case of a yes-instance, W can be written as a sum of a subset of $\{w_1, \dots, w_m\}$. Let $V \in \mathbb{N}_0$ be the largest subset sum with $V < W$. By applying Lemma 4.25 to $V_1 = W$, we obtain $f(b_f^+) = f(W + 2\varepsilon) = -M - W + \frac{\varepsilon^2}{M}$. By applying Lemma 4.25 to $V_1 = V$ and $V_2 = W$, the two possible values of f in $b_f^- = W$ are $-M - V + \frac{\varepsilon}{M}(W - V - \varepsilon)$ and $-W$. Both can be easily seen to be strictly larger than $f(b_f^+)$, which implies $f(b_f^-) > f(b_f^+)$. Thus, the maximum of f in the range $[b_f^-, b_f^+]$ is not attained in b_f^+. □

This finishes the proof of Theorem 4.22. In fact, our arguments not only show that the leader's problem is $\mathcal{NP}$-hard, but also that the adversary's problem is $\mathcal{NP}$-hard:

Theorem 4.28. *The adversary's problem of the problem stated in (RBCKP-b_f), with a discrete uncorrelated uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$, and given some feasible leader's solution, is $\mathcal{NP}$-hard, even if $|\mathcal{U}_e| = 2$ for all $e \in \mathcal{E}_f$.*

Proof. We give a reduction from the Subset Sum Problem that is closely related to the one we used in the proof of Theorem 4.22. Consider the same instance of (RBCKP-b_f) as constructed above, and fix the leader's solution to $b_f = b_f^+ = W + 2\varepsilon$. Then the adversary's problem consists in determining the value $f(b_f)$ (and the corresponding optimum solution d).

We have to show how to distinguish whether the given instance of the Subset Sum Problem is a yes-instance or a no-instance, using a solution of the constructed instance of the adversary's problem of (RBCKP-b_f). Considering Lemmas 4.26 and 4.27 and their proofs again, observe that the two cases can also be distinguished by computing the value $f(b_f)$ for $b_f = W + 2\varepsilon$. Indeed, using Lemma 4.25 like in Lemmas 4.26 and 4.27, it is

$$f(b_f) = -M - W + \frac{\varepsilon}{M}(b_f - W - \varepsilon)$$

in case of a yes-instance and

$$f(b_f) = -M - V + \frac{\varepsilon}{M}(b_f - V - \varepsilon)$$

in case of a no-instance, where $V \in \mathbb{N}_0$ is again the largest subset sum with $V < W$. Clearly, the second value is greater than the first one. Thus, by computing $f(b_f)$, we can solve the Subset Sum Problem, which completes the reduction. □

As above, the $\mathcal{NP}$-hardness result can be transferred to the Robust Bilevel Continuous Knapsack Problem under Assumption (uni′), which is equivalent to (RBCKP-b_f) by Lemma 4.10; see Corollary 4.29. Note that the correspondence of the two problems shown in Lemmas 4.3 and 4.10 includes a straightforward correspondence between leader's solutions of both problems and also holds for the adversary's problems when fixing such corresponding leader's solutions.

Corollary 4.29. *The adversary's problem of the Robust Bilevel Continuous Knapsack Problem with a discrete uncorrelated uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$, and given some feasible leader's solution, is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′) and $|\mathcal{U}_e| = 2$ for all $e \in \mathcal{E}_f$.*

We have shown that both the adversary's and the leader's problem of the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty are $\mathcal{NP}$-hard; see Corollaries 4.29 and 4.23. Given a robust bilevel optimization problem with an $\mathcal{NP}$-hard adversary's problem, the question can be raised whether the leader's problem is $\mathcal{NP}$-equivalent or even Σ_2^{p}-hard, like, e.g., in Theorem 2.33. This question, for the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty, remains a task for future research; see also Section 6.3.2. On the other hand, it is clear that the leader's problem can be at most one level harder than the adversary's problem; see also Section 2.4.2.

The $\mathcal{NP}$-hardness of the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty marks a difference to the Robust Bilevel Continuous Knapsack Problem with interval uncertainty, for which we have shown in Section 4.4.2 that it can be solved efficiently. In particular, this implies that replacing an uncertainty set by its convex hull may change the problem significantly, in contrast to the situation in single-level robust optimization; see also Section 2.4.2.3. Also for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and therefore for a special case

of the Robust Bilevel Continuous Knapsack Problem, we have seen that a discrete uncorrelated uncertainty set cannot simply be replaced by the interval uncertainty set that is its convex hull; see Example 3.50. However, in case of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) with discrete uncorrelated uncertainty, we have still seen a polynomial-time algorithm (see Algorithm 3.6 and Theorem 3.52), while, for the Robust Bilevel Continuous Knapsack Problem, interval uncertainty and discrete uncorrelated uncertainty even differ in their complexity unless $\mathcal{P} = \mathcal{NP}$.

4.4.3.2 Pseudopolynomial-Time Algorithms

On the positive side, we will now show that both polynomial-time algorithms for the adversary's and the leader's problem of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with discrete uncorrelated uncertainty, which have been presented in Section 3.4.1.2.3, can be adapted to yield pseudopolynomial-time algorithms for the adversary's and the leader's problem of the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty. The generalized version of Algorithm 3.5 and Theorem 3.51 will be presented in Algorithm 4.4 and Theorem 4.30, and the generalized version of Algorithm 3.6 and Theorem 3.52 will be presented in Algorithm 4.5 and Theorem 4.31. This proves that the $\mathcal{NP}$-hardness results in Corollaries 4.29 and 4.23 are only in the weak sense.

In contrast to the other main results in this chapter, the pseudopolynomial-time algorithms are not contained in [BH22]. In fact, it has been stated as an open question there whether the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty is strongly $\mathcal{NP}$-hard or can be solved in pseudopolynomial time. The pseudopolynomial-time algorithms have then been published in [Hen25], together with the corresponding results for the Robust Bilevel Selection Problem. Moreover, recall that [BH22; Hen25] deal with the problem (RBCKP-b_f) rather than with the Robust Bilevel Continuous Knapsack Problem as we defined it in this chapter. Therefore, similarly to the previous algorithmic results in this chapter, the following algorithms are slightly more general than the ones in [Hen25].

In view of the pseudopolynomial-time algorithms that we will discuss, we assume for the rest of the section that the item sizes $a\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{N}$ are all integers (which can be achieved by scaling the instance if necessary). The total size $a(\mathcal{E}_f) = \sum_{e\in\mathcal{E}_f} a(e)$ of all follower's items will then appear in the running times of the algorithms.

The item sizes are the main complication of the Robust Bilevel Continuous Knapsack Problem compared to the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and we need to take care of them in several ways when generalizing Algorithms 3.5 and 3.6:

Firstly, like in the previous algorithms in this chapter, we have to consider that the follower's greedy algorithm works with the order of the item profits $d(e)/a(e)$ instead of only the item values $d(e)$. This is reflected in the definitions of the sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$ in Algorithms 4.4 and 4.5, similarly to Algorithms 4.2 and 4.3 for the adversary's problem and the leader's problem of the Robust Bilevel Continuous Knapsack Problem with interval uncertainty.

Secondly, when fixing the fractional item of the follower's Continuous Knapsack Problem, the fraction with which it is selected is not uniquely determined by the capacity b_f, in contrast to the case where the follower solves a Continuous Selection

Problem. If b_f is not an integer, in the latter case, the fraction is always given by $b_f - \lfloor b_f \rfloor$, and we know that $\lfloor b_f \rfloor$ items are selected before the fractional item. We made use of this property in Algorithm 3.5. In the more general case of a Continuous Knapsack Problem, the items selected before the fractional item e^* are known to have an integer total size as well (as we assume all item sizes to be integers), but this can be any integer number b_f^* from $\lfloor b_f - a(e^*) \rfloor + 1$ to $\lfloor b_f \rfloor$ now, corresponding to fractions $y_{e^*} = \frac{1}{a(e)}(i + (b_f - \lfloor b_f \rfloor))$ for $i \in \{0, \dots, a(e) - 1\}$ (in reverse order). Therefore, we enumerate all these possibilities for b_f^* in addition to the possible fractional items e^* and their follower's item values δ^* in Algorithm 4.4. Note that Line 13 of Algorithm 4.4 combines this with the capacity condition that is used also in previous versions of the algorithm; see, e.g., Line 11 of Algorithm 3.5. When solving the leader's problem, we have to enumerate all reasonable integer capacities b_f^* anyway (see Algorithm 3.6) and do this in a similar way also in Algorithm 4.5.

For fixed e^*, δ^*, and b_f^*, the remaining task is now to solve a binary Knapsack Problem instead of a binary Selection Problem as in Algorithms 3.5 and 3.6. More precisely, the subroutine corresponds to a binary Knapsack Problem with an equality constraint because the items in the fractional prefix before the fractional item need to have a total size of exactly b_f^* according to the above considerations. If this problem is infeasible, then we skip the current value of b_f^*.

Formally, the problem that the subroutine solves can be stated as follows: Given a finite set $\mathcal{E}'$ of items, item sizes $a' \colon \mathcal{E}' \to \mathbb{N}$, a capacity $b' \in \{0, \dots, a'(\mathcal{E}')\}$, and item values $c' \colon \mathcal{E}' \to \mathbb{Q}$, select a subset $Y \subseteq \mathcal{E}'$ such that $a'(Y) = b'$ and $c'(Y)$ is maximum, or decide that there is no such Y. We write Knapsack Problem$(\mathcal{E}', a', b', c')$ when we call this subroutine and assume that it returns an optimum solution Y if there is one, or the string "infeasible" otherwise. The subroutine can be implemented to run in time $\mathcal{O}(|\mathcal{E}'| \cdot b')$ using dynamic programming; see, e.g., [KPP04; MACH12]. Recall also the discussion of the binary Knapsack Problem and variants of it in Section 4.1.

Like in Section 3.4.1.2.3, we neglect some technical details related to the optimistic and the pessimistic setting in the following. For the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), and with discrete uncorrelated uncertainty, we discussed them in detail in Remark 3.55. Analogously, Algorithms 4.4 and 4.5 are valid for the pessimistic setting as they are stated below, but can also be used for the optimistic setting after slight modifications.

We first adapt Algorithm 3.5 to the adversary's problem of the Robust Bilevel Continuous Knapsack Problem. Respecting all aspects explained above, we obtain Algorithm 4.4. Similarly to Theorem 3.51, we can show:

Theorem 4.30. *For any fixed feasible leader's solution of the Robust Bilevel Continuous Knapsack Problem with a discrete uncorrelated uncertainty set $\mathcal{U} = \prod_{e \in \mathcal{E}_f} \mathcal{U}_e$, and with integral item sizes a, Algorithm 4.4 solves the adversary's problem in time $\mathcal{O}(nuA)$, where $u = \sum_{e \in \mathcal{E}_f} |\mathcal{U}_e|$ and $A = a(\mathcal{E}_f)$.*

Proof. If the algorithm terminates from Line 3, any choice of $d \in \mathcal{U}$ is optimum for the adversary because the follower has only one trivial feasible solution anyway, namely either selecting no items or selecting all items from $\mathcal{E}_f$.

We first assume that $\overline{\mathcal{E}} \neq \emptyset$ holds in the end of the algorithm, and see that this is indeed true in the end of this proof. Let b_f^*, e^*, δ^*, and d be defined as in Lines 18 and 19, respectively.

Algorithm 4.4. Algorithm for the adversary's problem of the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $a\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{N}$, $b \in \mathbb{Q} \cap [0, a(\mathcal{E}_l \cup \mathcal{E}_f)]$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, a discrete uncorrelated uncertainty set $\mathcal{U}$ given by a finite set $\mathcal{U}_e \subseteq \mathbb{Q}$ for every $e \in \mathcal{E}_f$, a feasible leader's solution $x \in [0,1]^{\mathcal{E}_l}$

Output : an optimum adversary's solution $d \in \mathcal{U}$ (i.e., a value $d(e) \in \mathcal{U}_e$ for every $e \in \mathcal{E}_f$) of the Robust Bilevel Continuous Knapsack Problem

$b_f \coloneqq b - \sum_{e \in \mathcal{E}_l} a(e) x_e$
if $b_f = 0$ *or* $b_f = a(\mathcal{E}_f)$ **then**
  return *an arbitrary* $d \in \mathcal{U}$
for $e \in \mathcal{E}_f$ **do**
  $d^-(e) \coloneqq \min \mathcal{U}_e$, $d^+(e) \coloneqq \max \mathcal{U}_e$
$\overline{\mathcal{E}} \coloneqq \emptyset$
for $e^* \in \mathcal{E}_f$ **do**
  for $\delta^* \in \mathcal{U}_{e^*}$ **do**
    $\mathcal{E}^-_{e^*,\delta^*} \coloneqq \left\{ e \in \mathcal{E}_f \mid \frac{d^-(e)}{a(e)} > \frac{\delta^*}{a(e^*)} \right\}$
    $\mathcal{E}^0_{e^*,\delta^*} \coloneqq \left\{ e \in \mathcal{E}_f \mid \frac{d^-(e)}{a(e)} \le \frac{\delta^*}{a(e^*)} \le \frac{d^+(e)}{a(e)} \right\} \setminus \{e^*\}$
    $b_f^{*-} \coloneqq \max\{a(\mathcal{E}^-_{e^*,\delta^*}), \lfloor b_f - a(e^*) \rfloor + 1\}$
    $b_f^{*+} \coloneqq \min\{a(\mathcal{E}^-_{e^*,\delta^*} \cup \mathcal{E}^0_{e^*,\delta^*}), \lfloor b_f \rfloor\}$
    for $b_f^* \in \{b_f^{*-}, \dots, b_f^{*+}\}$ **do**
      $Y^0_{b_f^*,e^*,\delta^*} \coloneqq$ Knapsack Problem$(\mathcal{E}^0_{e^*,\delta^*}, a \restriction \mathcal{E}^0_{e^*,\delta^*}, b_f^* - a(\mathcal{E}^-_{e^*,\delta^*}), -c \restriction \mathcal{E}^0_{e^*,\delta^*})$
      if $Y^0_{b_f^*,e^*,\delta^*} \neq$ *"infeasible"* **then**
        $Y_{b_f^*,e^*,\delta^*} \coloneqq \mathcal{E}^-_{e^*,\delta^*} \cup Y^0_{b_f^*,e^*,\delta^*}$
        $\overline{\mathcal{E}} \coloneqq \overline{\mathcal{E}} \cup \{(b_f^*, e^*, \delta^*)\}$
select $(b_f^*, e^*, \delta^*) \in \operatorname{argmin}\{c(Y_{\overline{b_f},e,\delta}) + \frac{c(e)}{a(e)}(b_f - \overline{b_f}) \mid (\overline{b_f}, e, \delta) \in \overline{\mathcal{E}}\}$ arbitrarily
return $d\colon \mathcal{E}_f \to \mathbb{Q}$ *with* $d(e^*) \coloneqq \delta^*$, $d(e) \coloneqq d^+(e)$ *for all* $e \in Y_{b_f^*,e^*,\delta^*}$ *and* $d(e) \coloneqq d^-(e)$ *for all* $e \in \mathcal{E}_f \setminus (Y_{b_f^*,e^*,\delta^*} \cup \{e^*\})$

We claim that the follower's solution that selects the items in $Y_{b_f^*,e^*,\delta^*}$ and a fraction of $\frac{1}{a(e^*)}(b_f - b_f^*)$ of the item e^* is optimum for the follower when the leader's solution x and the adversary's choice d are fixed. Denote this follower's solution by the vector $y^{b_f^*,e^*,\delta^*} \in [0,1]^{\mathcal{E}_f}$, i.e., $y_e^{b_f^*,e^*,\delta^*} = 1$ for all $e \in Y_{b_f^*,e^*,\delta^*}$, $y_{e^*}^{b_f^*,e^*,\delta^*} = \frac{1}{a(e^*)}(b_f - b_f^*)$, and $y_e^{b_f^*,e^*,\delta^*} = 0$ for all $e \in \mathcal{E}_f \setminus (Y_{b_f^*,e^*,\delta^*} \cup \{e^*\})$. The follower's solution $y^{b_f^*,e^*,\delta^*}$ is clearly feasible, as we have $\sum_{e\in\mathcal{E}_f} a(e) y_e^{b_f^*,e^*,\delta^*} = a(\mathcal{E}^-_{e^*,\delta^*}) + a(Y^0_{b_f^*,e^*,\delta^*}) + (b_f - b_f^*) = b_f$, using the correctness of the subroutine for the Knapsack Problem we call in Line 14. Similarly to Algorithm 4.2 and the proof of Theorem 4.19, the sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$ defined in Lines 9 and 10, respectively, are the set of items that the follower always sorts before item e^* in his greedy order, given any $d' \in \mathcal{U}$ with $d'(e^*) = \delta^*$, and the set of items that the follower might sort before or after item e^* in such a greedy order, depending on the adversary's choice of their follower's item values. Accordingly, given the item values d returned by the algorithm, $y^{b_f^*,e^*,\delta^*}$ is optimum for the follower, since $\frac{d(e)}{a(e)} \geq \frac{\delta^*}{a(e^*)} = \frac{d(e^*)}{a(e^*)}$ for all $e \in \mathcal{E}_f$ with $y_e^{b_f^*,e^*,\delta^*} = 1$ and $\frac{d(e)}{a(e)} \leq \frac{\delta^*}{a(e^*)} = \frac{d(e^*)}{a(e^*)}$ for all $e \in \mathcal{E}_f$ with $y_e^{b_f^*,e^*,\delta^*} = 0$.

We now prove that the returned adversary's solution d is optimum. Let $y \in [0,1]^{\mathcal{E}_f}$ be an optimum follower's solution corresponding to an optimum adversary's choice $\overline{d}$ and let $\overline{e}$ be the fractional item, or, if the solution y is binary, let $\overline{e} \in \{e \in \mathcal{E}_f \mid y_e = 0\}$ such that $\overline{d}(e)/a(e)$ is maximum. This is valid because $b_f < a(\mathcal{E}_f)$ holds after Line 3 and because we may assume that y has been selected greedily according to an order given by $\overline{d}$ and a, and therefore has at most one fractional item; see Remark 4.1. Let $\overline{b_f} = a(\{e \in \mathcal{E}_f \mid y_e = 1\})$. Accordingly, we have $y_{\overline{e}} = \frac{1}{a(\overline{e})}(b_f - \overline{b_f})$ because we know that y is a feasible follower's solution, i.e., it satisfies $\sum_{e\in\mathcal{E}_f} a(e) y_e = b_f$. Let $\overline{\delta} = \overline{d}(\overline{e})$. The follower's greedy choice implies that $\mathcal{E}^-_{\overline{e},\overline{\delta}} \subseteq \{e \in \mathcal{E}_f \mid y_e = 1\}$ and $\{e \in \mathcal{E}_f \mid y_e = 1\} \setminus \mathcal{E}^-_{\overline{e},\overline{\delta}} \subseteq \mathcal{E}^0_{\overline{e},\overline{\delta}}$. Hence, $\overline{b_f} = a(\{e \in \mathcal{E}_f \mid y_e = 1\}) \in \{a(\mathcal{E}^-_{e^*,\delta^*}), \dots, a(\mathcal{E}^-_{e^*,\delta^*} \cup \mathcal{E}^0_{e^*,\delta^*})\}$. Moreover, since $y_{\overline{e}} = \frac{1}{a(\overline{e})}(b_f - \overline{b_f}) \in [0,1)$, we also have $\overline{b_f} \in \{\lfloor b_f - a(\overline{e}) \rfloor + 1, \dots, \lfloor b_f \rfloor\}$. Together, this implies that $\overline{b_f}$ is one of the values considered in Line 13 in the iteration corresponding to $\overline{e}$ and $\overline{\delta}$, and that the items in $\{e \in \mathcal{E}_f \mid y_e = 1\} \setminus \mathcal{E}^-_{\overline{e},\overline{\delta}}$ form a feasible solution for the Knapsack Problem in Line 14. It must be optimum as well because a better adversary's solution could be constructed from an optimum solution of the Knapsack Problem otherwise, analogously to the proof of Theorem 3.34. Thus, y is a follower's solution that is enforced in the iteration corresponding to $\overline{e}$, $\overline{\delta}$, and $\overline{b_f}$ or has the same leader's objective function value as it. This proves the optimality of the returned solution and also shows that $\overline{\mathcal{E}} \neq \emptyset$ is satisfied in the end of the algorithm.

The running time is given by $u = \sum_{e\in\mathcal{E}_f} |\mathcal{U}_e|$ iterations each of which can be implemented to run in time $\mathcal{O}(nA)$. Note that all Knapsack Problems in the loop in Lines 13 to 17 can be solved using a single dynamic program because all of them deal with the same item set and only different capacities. □

We now turn to the leader's algorithm and adapt Algorithms 3.6 and 4.4 to the leader's problem of the Robust Bilevel Continuous Knapsack Problem. This results in Algorithm 4.5.

In Algorithm 4.5, analogously to Algorithm 4.4, we enumerate all possible fractional items $e^* \in \mathcal{E}_f$, the values $\delta^* \in \mathcal{U}_{e^*}$ assigned to them by the adversary, and all possible total

Algorithm 4.5. Algorithm for the leader's problem of the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty

Input : finite sets $\mathcal{E}_l$ and $\mathcal{E}_f$ with $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$, $a\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{N}$, $b \in \mathbb{Q} \cap [0, a(\mathcal{E}_l \cup \mathcal{E}_f)]$, $c\colon \mathcal{E}_l \cup \mathcal{E}_f \to \mathbb{Q}$, a discrete uncorrelated uncertainty set $\mathcal{U}$ given by a finite set $\mathcal{U}_e \subseteq \mathbb{Q}$ for every $e \in \mathcal{E}_f$

Output : an optimum leader's solution $x \in [0,1]^{\mathcal{E}_l}$ of the Robust Bilevel Continuous Knapsack Problem

$b_l^- \coloneqq \max\{0, b - a(\mathcal{E}_f)\}$, $b_f^+ \coloneqq b - b_l^-$
$b_l^+ \coloneqq \min\{b, a(\mathcal{E}_l)\}$, $b_f^- \coloneqq b - b_l^+$
if $b_l^- = b_l^+$ **then**
  $b_l^* \coloneqq b_l^-$
else
  compute $g_l \coloneqq \mathrm{PLF}'(\mathcal{E}_l, a \restriction \mathcal{E}_l, c \restriction \mathcal{E}_l, b_l^-, b_l^+)$
  for $e \in \mathcal{E}_f$ **do**
    $d^-(e) \coloneqq \min \mathcal{U}_e$, $d^+(e) \coloneqq \max \mathcal{U}_e$
  $\overline{\mathcal{E}} \coloneqq \emptyset$
  for $e^* \in \mathcal{E}_f$ **do**
    for $\delta^* \in \mathcal{U}_{e^*}$ **do**
      $\mathcal{E}^-_{e^*,\delta^*} \coloneqq \left\{e \in \mathcal{E}_f \mid \frac{d^-(e)}{a(e)} > \frac{\delta^*}{a(e^*)}\right\}$
      $\mathcal{E}^0_{e^*,\delta^*} \coloneqq \left\{e \in \mathcal{E}_f \mid \frac{d^-(e)}{a(e)} \le \frac{\delta^*}{a(e^*)} \le \frac{d^+(e)}{a(e)}\right\} \setminus \{e^*\}$
      $b_f^{*-} \coloneqq \max\{a(\mathcal{E}^-_{e^*,\delta^*}), \lfloor b_f^- - a(e^*) \rfloor + 1\}$
      $b_f^{*+} \coloneqq \min\{a(\mathcal{E}^-_{e^*,\delta^*} \cup \mathcal{E}^0_{e^*,\delta^*}), \lfloor b_f^+ \rfloor\}$
      for $b_f^* \in \{b_f^{*-}, \dots, b_f^{*+}\}$ **do**
        $Y^0_{b_f^*,e^*,\delta^*} \coloneqq$ Knapsack Problem$(\mathcal{E}^0_{e^*,\delta^*}, a \restriction \mathcal{E}^0_{e^*,\delta^*}, b_f^* - a(\mathcal{E}^-_{e^*,\delta^*}), -c \restriction \mathcal{E}^0_{e^*,\delta^*})$
        if $Y^0_{b_f^*,e^*,\delta^*} \neq$ *"infeasible"* **then**
          $Y_{b_f^*,e^*,\delta^*} \coloneqq \mathcal{E}^-_{e^*,\delta^*} \cup Y^0_{b_f^*,e^*,\delta^*}$
          $g_f^{b_f^*,e^*,\delta^*} \coloneqq$ linear segment from $(b - b_f^* - a(e^*), c(Y_{b_f^*,e^*,\delta^*}) + c(e^*))$ to $(b - b_f^*, c(Y_{b_f^*,e^*,\delta^*}))$
          $\overline{\mathcal{E}} \coloneqq \overline{\mathcal{E}} \cup \{(b_f^*, e^*, \delta^*)\}$
  compute $g_f \coloneqq \min\{g_f^{b_f^*,e^*,\delta^*} \mid (b_f^*, e^*, \delta^*) \in \overline{\mathcal{E}}\}$ (i.e., the pointwise minimum)
  compute $g \coloneqq g_l + g_f$
  compute some $b_l^* \in \operatorname{argmax}\{g(b_l) \mid b_l \in [b_l^-, b_l^+]\}$
return Continuous Knapsack Problem$(\mathcal{E}_l, a \restriction \mathcal{E}_l, b_l^*, c \restriction \mathcal{E}_l)$

sizes b_f^* of the items the follower selects before the fractional item e^*. The feasible values for b_f^* are related to the total sizes of the sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$ (see Line 16), like in Line 13 of Algorithm 4.4, but we consider all possible capacities $b_f \in [b_f^-, b_f^+]$ simultaneously now, as the leader's decision is not fixed yet. If the subroutine in Line 17 returns a feasible solution, we construct a linear segment corresponding to the leader's values of the follower's solutions consisting of the binary prefix and any fraction $y_{e^*} \in [0,1]$ of item e^* (see Line 20), analogously to Line 18 of Algorithm 3.6. Note that the width of the linear segment is now given by the size $a(e^*)$, in contrast to the setting of Algorithm 3.6 where all items have size 1. Accordingly, we cannot compute the pointwise minimum for every b_f^* separately like in Algorithm 3.6 (see Lines 20 and 21 there), but we collect all linear segments and compute the pointwise minimum globally in Line 22 of Algorithm 4.5. A similar situation occurred for discrete uncertainty as well; see Remark 4.14. Moreover, also the leader solves a Continuous Knapsack Problem instead of a Continuous Selection Problem now, for every fixed b_l, which is reflected in the function g_l computed in Line 6 and in the final computation of the leader's solution in Line 25.

Similarly to Theorem 3.52, we can prove:

Theorem 4.31. *Algorithm 4.5 solves the Robust Bilevel Continuous Knapsack Problem with a discrete uncorrelated uncertainty set $\mathcal{U} = \prod_{e\in\mathcal{E}_f} \mathcal{U}_e$, and with integral item sizes a, in time $\mathcal{O}(nuA\log(nuA))$, where $u = \sum_{e\in\mathcal{E}_f}|\mathcal{U}_e|$ and $A = a(\mathcal{E}_f)$.*

Proof. First observe that, if $b_l^- = b_l^+$, then there is only one way to distribute the total capacity of b among the two players, such that Lines 3, 4, and 25 lead to an optimum leader's solution in this case. From now on, we assume $b_l^- < b_l^+$.

Analogously to the proof of Theorem 4.20, the function g_l computed in Line 6, describing the part of the leader's objective function that represents the optimum value she can achieve by a selection of her own items when using a capacity of b_l, is well-defined and has the range $[b_l^-, b_l^+]$. Again, for the optimum value $b_l^* \in [b_l^-, b_l^+]$, an optimum leader's solution is given by Continuous Knapsack Problem$(\mathcal{E}_l, a \restriction \mathcal{E}_l, b_l^*, c \restriction \mathcal{E}_l)$, as returned by the algorithm in Line 25. Therefore, it remains to show that the value b_l^* derived by the algorithm is indeed optimum.

Again as in the proof of Theorem 4.20, we will show that, for every fixed value $b_l \in [b_l^-, b_l^+]$, the function g computed in Line 23 correctly describes the best leader's objective function value that is possible if the leader uses a capacity of b_l and the follower uses a capacity of $b_f = b - b_l$. Hence, it remains to prove that, for every fixed value $b_l \in [b_l^-, b_l^+]$, the value of the adversary's solution returned by Algorithm 4.4, with any given feasible leader's solution x using capacity b_l, is precisely $g_f(b_l)$. The correctness of Algorithm 4.5 then follows from Theorem 4.30.

Let $b_l \in [b_l^-, b_l^+]$ be some fixed leader's capacity and $b_f = b - b_l \in [b_f^-, b_f^+]$. Fix some $e^* \in \mathcal{E}_f$ and $\delta^* \in \mathcal{U}_{e^*}$. The sets $\mathcal{E}^-_{e^*,\delta^*}$ and $\mathcal{E}^0_{e^*,\delta^*}$ are defined in the same way in both algorithms. Let b_f^* be one of the values considered in Line 13 of Algorithm 4.4 in the iteration corresponding to e^* and δ^*. Note that b_f^* is also one of the values considered in Line 16 of Algorithm 4.5 then. Consider the iterations of the loops in Lines 7, 8, and 13 of Algorithm 4.4 and of the loops in Lines 10, 11, and 16 of Algorithm 4.5 that correspond to e^*, δ^*, and b_f^*. The Knapsack Problem that is solved in Line 14 of Algorithm 4.4 is exactly the same as the one that is solved in Line 17 of Algorithm 4.5. If it is feasible, then the sets $Y_{b_f^*,e^*,\delta^*}$ can be assumed to be the same in both algorithms. Now consider the linear

segment $g_f^{b_f^*,e^*,\delta^*}$, which is computed in Line 20 of Algorithm 4.5. Observe that it is defined in the point $b_l = b - b_f$ and that its value in this point is the leader's objective function value of the follower's solution that selects $Y_{b_f^*,e^*,\delta^*}$ and a fraction of $\frac{1}{a(e^*)}(b_f - b_f^*)$ of the item e^*, i.e., $c(Y_{b_f^*,e^*,\delta^*}) + \frac{c(e^*)}{a(e^*)}(b_f - b_f^*)$. The same value is also considered in Line 18 of Algorithm 4.4. On the other hand, a linear segment that is defined in an iteration $b_f^{*\prime}$ of the loop in Line 16 of Algorithm 4.5 such that $b_f^{*\prime}$ is not considered in Line 13 of Algorithm 4.4, is not defined in $b_l = b - b_f$ for our fixed capacities b_l and b_f. Hence, the set $\overline{\mathcal{E}}$ in the end of Algorithm 4.4 coincides with the entries of the set $\overline{\mathcal{E}}$ in the end of Algorithm 4.4 that correspond to linear segments that are defined in b_l. Thus, the value of the function g_f, which is computed in Line 22 of Algorithm 4.5, in the point b_l is the same as the minimum value that is considered in Line 18 of Algorithm 4.4. This finishes the proof of the correctness of Algorithm 4.5.

It remains to analyze the running time of Algorithm 4.5. According to Section 4.3, the function g_l has at most n linear segments and can be computed in time $\mathcal{O}(n \log(n))$ in Line 6. Analogously to Algorithm 4.4 (see Theorem 4.30), the whole loop in Lines 16 to 21 can be implemented to run in time $\mathcal{O}(nA)$ and the whole loop in Lines 10 to 21 in time $\mathcal{O}(nuA)$. The function g_f, which is defined as the pointwise minimum of $\mathcal{O}(uA)$ many linear segments in Line 22, can be computed in time $\mathcal{O}(uA \log(uA))$ using [Her89]. Finally, the sum of the piecewise linear functions g_l and g_f and the maximum of the resulting piecewise linear function can be computed in time $\mathcal{O}(n + uA \log(uA))$, as the total number of linear segments is $\mathcal{O}(n + uA \log(uA))$ as well. Together, this results in a running time of $\mathcal{O}(nuA \log(nuA))$ for the whole algorithm. □

Remark 4.32. Analogously to Remarks 3.53 and 4.21, note that the function g computed by Algorithm 4.5 is the piecewise linear function that is described in the beginning of Section 4.4, consisting of the best leader's objective function values that can be achieved for any fixed $b_l \in [b_l^-, b_l^+]$; see also Remark 4.11. This also implies again that g is continuous, which is not obvious from its definition in Algorithm 4.5. Moreover, Theorem 4.31 implies that the number of linear segments of the piecewise linear function g is pseudopolynomial.

Remark 4.33. The algorithmic approaches used for the Robust Bilevel Continuous Knapsack Problem with interval uncertainty (see Section 4.4.2) and the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty (in this section) are based on similar ideas, but contain some significant differences as well. The same applies to the corresponding algorithms for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) presented in Sections 3.4.1.2.2 and 3.4.1.2.3, which can be seen as special cases of the algorithms here. We refer to Remark 3.54 for a more detailed comparison of the two approaches. While both algorithms run in polynomial time in case of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), we have presented a polynomial-time algorithm for interval uncertainty and a pseudopolynomial-time algorithm for discrete uncorrelated uncertainty in case of the Robust Bilevel Continuous Knapsack Problem. The differences between the four variants of the algorithm and their complexity are most prominently demonstrated by looking at the subroutines that are called in them: For the Robust Bilevel Selection Problem under Assumptions (disj) and (cc), the subroutine solves a Continuous Selection Problem and a Selection Problem, respectively, in case of interval uncertainty and discrete uncorrelated uncertainty. For the Robust Bilevel Continuous Knapsack Problem,

the subroutine solves a Continuous Knapsack Problem and a Knapsack Problem, respectively, in case of interval uncertainty and discrete uncorrelated uncertainty. While the binary Knapsack Problem is weakly $\mathcal{NP}$-hard and solvable in pseudopolynomial time, all other subroutines can be implemented to run in polynomial time using simple greedy algorithms. Note that, in Algorithms 4.4 and 4.5, not only the subroutine, but also the enumeration of values b_f^* leads to the running time being only pseudopolynomial but not polynomial. The $\mathcal{NP}$-hardness of the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty (see Corollary 4.23) also suggests that we cannot expect the more efficient variant of the approach that works for interval uncertainty to be applicable to the setting of discrete uncorrelated uncertainty.

4.4.4 Simplicial Uncertainty

In this section, we consider uncertainty sets that are simplices and again show that the Robust Bilevel Continuous Knapsack Problem as well as its adversary's problem are $\mathcal{NP}$-hard in this case; see Corollaries 4.36 and 4.38. This type of uncertainty set can be considered to be a special case of a continuous budgeted uncertainty set (see Section 2.3.2.4) and we define it as follows, for our setting of the Robust Bilevel Continuous Knapsack Problem: Given some $\hat{d} \in \mathbb{Q}^{\mathcal{E}_f}$ and $\Gamma \in \mathbb{Q}_{\geq 0}$, the corresponding simplicial uncertainty set is given as

$$\mathcal{U}_{\hat{d},\Gamma} \coloneqq \left\{ d \in \mathbb{R}^{\mathcal{E}_f} \mid d(e) \geq \hat{d}(e) \text{ for all } e \in \mathcal{E}_f, \sum_{e \in \mathcal{E}_f} (d(e) - \hat{d}(e)) \leq \Gamma \right\}.$$

Hence, we will always assume that, for a robust problem with simplicial uncertainty, the parameters $\hat{d}$ and Γ are part of the input.

A simplicial uncertainty set is a special case of a polytopal uncertainty set (see Section 2.3.2.5). Therefore, our $\mathcal{NP}$-hardness results also hold for this type of uncertainty. This is true both in case the polytope is given by an outer description (as above) and in case it is given by an inner description (since the set of vertices of a simplicial uncertainty set can be determined efficiently); see Corollaries 4.39 and 4.40.

Like in Section 4.4.3.1, we prove the $\mathcal{NP}$-hardness results for the problem (RBCKP-b_f); see Theorems 4.34 and 4.37. By Lemma 4.10, this problem is equivalent to the special case of Robust Bilevel Continuous Knapsack Problem where Assumption (uni′) is satisfied, which implies the $\mathcal{NP}$-hardness results stated in Corollaries 4.36 and 4.38.

Theorem 4.34. *The problem stated in (RBCKP-b_f), with simplicial uncertainty, is $\mathcal{NP}$-hard.*

Proof. We present a reduction from the $\mathcal{NP}$-complete Subset Sum Problem (see Section 4.1). Let $m \in \mathbb{N}$ and $w_1, \dots, w_m, W \in \mathbb{N}$ be an instance of the Subset Sum Problem. We show that we can decide in polynomial time whether there is a subset $S \subseteq \{1, \dots, m\}$ with $\sum_{i \in S} w_i = W$ if the following instance of (RBCKP-b_f) can be solved in polynomial time: Define $M \coloneqq \sum_{i=1}^{m} w_i + 1$ and let $\mathcal{E}_f = \{e_1, \dots, e_{m+1}\}$. Set the item sizes as $a(e_i) = w_i$ for $i \in \{1, \dots, m\}$ and $a(e_{m+1}) = M$, the leader's item values as $c(e_i) = -w_i$ for $i \in \{1, \dots, m\}$ and $c(e_{m+1}) = M$, and the capacity bounds as $b_f^- = 0$ and $b_f^+ = a(\mathcal{E}_f) = \sum_{i=1}^{m} w_i + M$. The simplicial uncertainty set $\mathcal{U}_{\hat{d},\Gamma}$ is given by $\hat{d}(e_i) = (2M - 1)w_i$ for $i \in \{1, \dots, m\}$, $\hat{d}(e_{m+1}) = 2M^2$, and $\Gamma = W$.

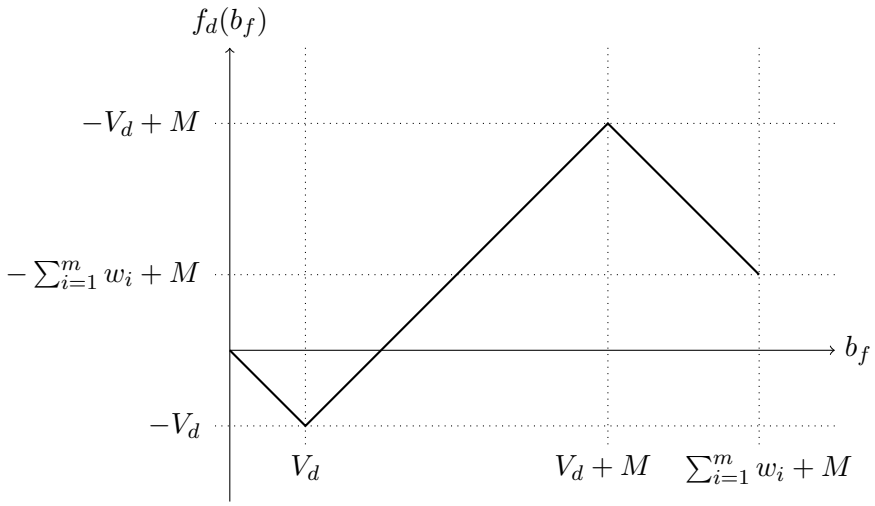


Figure 4.4. The function f_d, for some $d \in \mathcal{U}_{\hat{d},\Gamma}$, in the proof of Theorem 4.34

Recall the representation of the leader's objective function in terms of piecewise linear functions described in Section 4.3 and in the beginning of Section 4.4. Similarly to that, we can describe the leader's objective function of the constructed instance of (RBCKP-b_f), which we again denote by $f\colon [b_f^-, b_f^+] \to \mathbb{R}$ in the following. Indeed, for every fixed $d \in \mathcal{U}_{\hat{d},\Gamma}$, the leader's values of the optimum follower's solutions corresponding to different capacities b_f form a piecewise linear function $f_d\colon [b_f^-, b_f^+] \to \mathbb{R}$ of the type depicted in Figure 4.1c; see the corresponding explanation in Section 4.3 and also Remark 4.8. Since the piecewise linear function only depends on the follower's greedy order given by the profits $d(e)/a(e)$, but not on the precise follower's item values d, there is only a finite number of distinct functions f_d; see also Remark 4.11. Analogously to the discussion in the beginning of Section 4.4, the leader's objective function f is the pointwise minimum of all these piecewise linear functions f_d, for $d \in \mathcal{U}_{\hat{d},\Gamma}$.

Since $c(e_i)/a(e_i) = -1$ for all $i \in \{1, \dots, m\}$ and $c(e_{m+1})/a(e_{m+1}) = 1$, only the slopes -1 and 1 occur in the piecewise linear functions f_d. For a more precise description, define, for all $d \in \mathcal{U}_{\hat{d},\Gamma}$,

$$I_d := \left\{ i \in \{1, \dots, m\} \colon \frac{d(e_i)}{a(e_i)} \geq \frac{d(e_{m+1})}{a(e_{m+1})} \right\}$$

and $V_d := \sum_{i \in I_d} a(e_i) = \sum_{i \in I_d} w_i$. Then we may assume that, in the follower's greedy order corresponding to the follower's item values d, the items corresponding to I_d occur first, then the item e_{m+1}, and then all remaining items. Note that we consider the pessimistic setting here; see Remark 4.35 below for more details. Hence, the function f_d consists of a linear segment with slope -1 for $b_f \in [0, V_d]$, a linear segment with slope 1 for $b_f \in [V_d, V_d + M]$, and another linear segment with slope -1 for $b_f \in [V_d + M, \sum_{i=1}^{m} w_i + M]$; see Figure 4.4.

Due to the special structure of the functions f_d, the leader's objective function f, which is the pointwise minimum of all f_d, agrees with f_{d^*} for $d^* \in \operatorname{argmax}\{V_d \mid d \in \mathcal{U}_{\hat{d},\Gamma}\}$.

The leader's optimum solution corresponds to the maximum of the function $f = f_{d^*}$ in the range $[b_f^-, b_f^+] = [0, \sum_{i=1}^{m} w_i + M]$. Since, by the choice of M, we have $f(V_{d^*} + M) =$

$-V_{d^*} + M > 0 = f(0)$, this maximum is attained in $b_f^* = V_{d^*} + M$ with a leader's value of $f(b_f^*) = -V_{d^*} + M$.

It remains to show that, by computing b_f^*, and therefore V_{d^*}, we can decide whether the given instance of the Subset Sum Problem is a yes-instance or a no-instance. Since

$$\frac{\hat{d}(e_i)}{a(e_i)} = 2M - 1 < 2M = \frac{\hat{d}(e_{m+1})}{a(e_{m+1})}$$

for all $i \in \{1, \dots, m\}$, the adversary has two options for every item e_i:

1. Either he decides to include i in the set I_d, which can be achieved by increasing the value $\hat{d}(e_i)/a(e_i)$ by 1, up to $\hat{d}(e_{m+1})/a(e_{m+1})$. For this, the adversary has to set $d(e_i) = \hat{d}(e_i) + a(e_i) = \hat{d}(e_i) + w_i$ and thus pay $d(e_i) - \hat{d}(e_i) = w_i$ in the constraint $\sum_{i=1}^{m+1}(d(e_i) - \hat{d}(e_i)) \leq \Gamma$ on the uncertainty set. Note that there is no incentive to choose $d(e_i)$ greater than that, as it would not change the leader's objective function value.
2. Or he decides not to include i in the set I_d, for which it is most efficient in terms of the Γ-constraint to set $d(e_i) = \hat{d}(e_i)$.

Note that there is no incentive to choose $d(e_{m+1}) > \hat{d}(e_{m+1})$. Therefore, we may assume that $d(e_i) \in \{\hat{d}(e_i), \hat{d}(e_i)+w_i\}$ for all $i \in \{1, \dots, m\}$ and $d(e_{m+1}) = \hat{d}(e_{m+1})$ hold for an optimum adversary's solution d. Moreover, d has to satisfy $V_d = \sum_{i \in I_d} w_i = \sum_{i=1}^{m+1}(d(e_i) - \hat{d}(e_i)) \leq \Gamma = W$. Hence, by maximizing V_d, i.e., by computing V_{d^*}, we can compute the largest subset sum V of $\{w_1, \dots, w_m\}$ with $V \leq W$. Thus, the given instance of the Subset Sum Problem is a yes-instance if and only if $V_{d^*} = W$. □

Remark 4.35. In the proof of Theorem 4.34, we used the assumption of the pessimistic setting because we included the index $i \in \{1, \dots, m\}$ in the set I_d even if $\frac{d(e_i)}{a(e_i)} = \frac{d(e_{m+1})}{a(e_{m+1})}$, thus assuming that the follower sorts item e_i before item e_{m+1} in this case, which is the worse option from the leader's perspective. In the optimistic setting, item e_i may be packed before item e_{m+1} only if $\frac{d(e_i)}{a(e_i)} > \frac{d(e_{m+1})}{a(e_{m+1})}$. Without changing the effect of the constraint $\sum_{i=1}^{m}(d(e_i) - \hat{d}(e_i)) \leq \Gamma$, this could be modeled by choosing $\hat{d}(e_{m+1})$ slightly smaller than $2M^2$ in the definition of the instance.

As mentioned above, Lemma 4.10 and Theorem 4.34 imply:

Corollary 4.36. *The Robust Bilevel Continuous Knapsack Problem with simplicial uncertainty is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′).*

Similarly to Section 4.4.3.1, the arguments used in the proof of Theorem 4.34 also imply that the adversary's problem is $\mathcal{NP}$-hard:

Theorem 4.37. *The adversary's problem of the problem stated in (RBCKP-b_f), with simplicial uncertainty, and given some feasible leader's solution, is $\mathcal{NP}$-hard.*

Proof. We give a very similar reduction from the Subset Sum Problem as in the proof of Theorem 4.34. Define the same instance of (RBCKP-b_f) as there and fix the leader's decision as $b_f = W$. The proof of Theorem 4.34 implies that the given instance of the Subset Sum Problem is a yes-instance if and only if $V_{d^*} = W$, which is equivalent to $f(W) = f_{d^*}(W) = -W$ and thus to the adversary being able to achieve a leader's objective value of $-W$, while only some larger value can be achieved in case of a no-instance. □

Again, by Lemma 4.10, we get:

Corollary 4.38. *The adversary's problem of the* Robust Bilevel Continuous Knapsack Problem *with simplicial uncertainty, and given some feasible leader's solution, is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′).*

The simplicial uncertainty set $U_{\hat{d},\Gamma}$ is a polytope defined explicitly by $n_f + 1$ linear inequalities. Therefore, all $\mathcal{NP}$-hardness results also hold for polytopal uncertainty sets given by an outer description:

Corollary 4.39. *The* Robust Bilevel Continuous Knapsack Problem *with an uncertainty set being a polytope, given by a set of linear inequalities, is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′). The same holds for the problem stated in (RBCKP-b$_f$) and for the adversary's problems of both robust bilevel problems.*

The simplicial uncertainty set can also be written as a convex hull of $n_f + 1$ vectors, namely as

$$\mathcal{U}_{\hat{d},\Gamma} = \operatorname{conv}(\{\hat{d}\} \cup \{d^e \mid e \in \mathcal{E}_f\}),$$

where, for all $e \in \mathcal{E}_f$, the vector $d^e \in \mathbb{Q}^{\mathcal{E}_f}$ is defined by $d^e(e) = \hat{d}(e) + \Gamma$ and $d^e(e') = \hat{d}(e')$ for all $e' \in \mathcal{E}_f \setminus \{e\}$. This corresponds to the inner description of the polytope $\mathcal{U}_{\hat{d},\Gamma}$. Hence, the $\mathcal{NP}$-hardness results also hold if the uncertainty set is given in this way:

Corollary 4.40. *The* Robust Bilevel Continuous Knapsack Problem *with an uncertainty set being the convex hull of a finite set of vectors, which are explicitly given as part of the input, is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′). The same holds for the problem stated in (RBCKP-b$_f$) and for the adversary's problems of both robust bilevel problems.*

Recall that the Robust Bilevel Continuous Knapsack Problem is solvable in polynomial time if the uncertainty set $\mathcal{U}$ is finite and given explicitly as part of the input; see Theorem 4.13. Together with Corollary 4.40, this again shows that replacing the uncertainty set by its convex hull may not only change the optimum solution, but even the complexity of the problem significantly; see also Section 6.2.

Remark 4.41. In contrast to the result of Corollary 4.40, the Robust Bilevel Continuous Knapsack Problem can be solved in polynomial time if the uncertainty set is given as the convex hull of a constant number k of vectors. In fact, the number of possible follower's greedy orders can be bounded by $\mathcal{O}(n_f^{2k-2})$ in this case. We give an outline of a proof for this below. Then, for constant k, all such orders can be enumerated explicitly in polynomial time, which leads to a polynomial-time algorithm for the Robust Bilevel Continuous Knapsack Problem in this case; see Remark 4.11. In other words, the Robust Bilevel Continuous Knapsack Problem with an uncertainty set being the convex hull of a finite set of vectors, which are explicitly given as part of the input, is slice-wise polynomial ($\mathcal{XP}$) in the number of given vectors.

For an illustration of the case $k = 2$, consider Figure 4.5. The ordered follower's profits of all follower's items in the two given scenarios d_1 and d_2 are shown in the left and in the right column, respectively. Each of the lines represents one item and connects its follower's profit in scenario d_1 on the left with its follower's profit in scenario d_2 on the right. Hence, for each $\lambda \in [0, 1]$, the profits given by the follower's item values $d = (1 - \lambda)d_1 + \lambda d_2 \in \mathcal{U}$

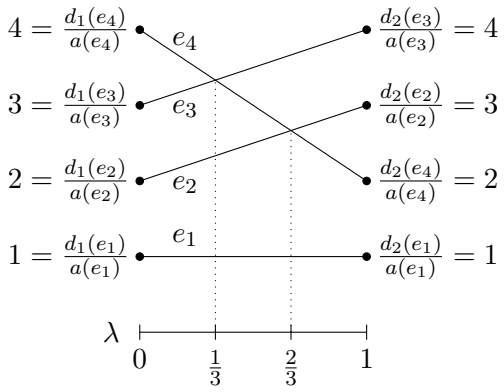


Figure 4.5. Illustration of the possible follower's greedy orders given by an uncertainty set that is the convex hull of a constant number k of vectors, as described in Remark 4.41, in the case $k = 2$. In the example, we have $\mathcal{E}_f = \{e_1, e_2, e_3, e_4\}$, the item sizes are given by $a(e) = 1$ for all $e \in \mathcal{E}_f$, and the uncertainty set is $\mathcal{U} = \operatorname{conv}\{d_1, d_2\}$ with $d_1(e_1) = 1$, $d_1(e_2) = 2$, $d_1(e_3) = 3$, $d_1(e_4) = 4$, $d_2(e_1) = 1$, $d_2(e_2) = 3$, $d_2(e_3) = 4$, and $d_2(e_4) = 2$. The possible follower's item values $d \in \mathcal{U}$ can be written in the form $d = (1 - \lambda)d_1 + \lambda d_2$ for $\lambda \in [0, 1]$. For $\lambda \in [0, 1/3)$, the unique follower's greedy order is e_4, e_3, e_2, e_1. For $\lambda \in (1/3, 2/3)$, the unique follower's greedy order is e_3, e_4, e_2, e_1. For $\lambda \in (2/3, 1]$, the unique follower's greedy order is e_3, e_2, e_4, e_1. For $\lambda = 1/3$ and $\lambda = 2/3$, the follower's greedy order is not unique.

correspond to the positions of the items' lines above λ in the illustration. Accordingly, the follower's greedy order corresponding to λ can be determined from the order of the lines above it. Whenever two lines intersect, the corresponding two items swap their positions in the follower's greedy order (and the greedy order is not unique in this point). In order to determine the number of possible orders that can arise from the uncertainty set, i.e., from different values of $\lambda \in [0, 1]$, we project each intersection point to the line segment that represents the interval $[0, 1]$ and count the regions into which the interval is divided by the projected points. Since there are n_f lines, the number of intersections is at most $\binom{n_f}{2} = \mathcal{O}(n_f^2)$. The number of regions into which their projections divide the interval is one more than the number of intersections and therefore $\mathcal{O}(n_f^2)$ as well.

We now explain how to apply the same idea to general k. Let the uncertainty set be given as $\mathcal{U} = \operatorname{conv}\{d_1, \dots, d_k\}$. Instead of the interval $[0, 1]$ of λ values, we can now use a $(k-1)$-dimensional simplex to parameterize the uncertainty set. This simplex has one vertex for each of the k given scenarios and can be imagined as a subset of $\mathbb{R}^{k-1}$. (In particular, it is a line segment if $k = 2$ and a triangle if $k = 3$.) We now introduce an additional dimension for representing the follower's profits again. More precisely, consider the space $\mathbb{R}^k$, where the $(k-1)$-dimensional simplex is embedded into the subspace $\mathbb{R}^{k-1} \times \{0\}$. We write the positions of the k vertices of the simplex as $(z^i, 0) \in \mathbb{R}^{k-1} \times \{0\}$ for $i \in \{1, \dots, k\}$. The follower's profits of all items in scenario d_i are now represented by the points $(z^i, d_i(e)/a(e))$ for $e \in \mathcal{E}_f$, and each follower's item $e \in \mathcal{E}_f$ is represented by the $(k-1)$-dimensional hyperplane that contains the points $(z^i, d_i(e)/a(e))$ for all $i \in \{1, \dots, k\}$. Again, any intersection of two of these hyperplanes marks two items swapping their positions in the follower's greedy

order and there are at most $\binom{n_f}{2} = \mathcal{O}(n_f^2)$ such intersections. Each such intersection is $(k-2)$-dimensional and we consider its projection onto the subspace $\mathbb{R}^{k-1} \times \{0\}$ and hence onto the simplex that parameterizes the uncertainty set. The number of possible follower's greedy orders is therefore given by the number of regions into which the $\mathcal{O}(n_f^2)$ many $(k-2)$-dimensional hyperplanes divide the $(k-1)$-dimensional simplex. Using [Buc43], this number can be bounded by $\sum_{i=0}^{k-1} \binom{n_f^2}{i} \leq (n_f^2)^{k-1} + 1 = \mathcal{O}(n_f^{2k-2})$.

In order to enumerate all possible follower's greedy orders, we need to find a point in each of the regions of the $(k-1)$-dimensional simplex and determine the corresponding scenario $d \in \mathcal{U}$. This can be done, for example, by explicitly computing the hyperplanes and from them a polyhedral description of each region, and by making use of the ellipsoid method.

4.4.5 Norm Uncertainty

This section deals with the Robust Bilevel Continuous Knapsack Problem with norm uncertainty; see Section 2.3.2.7. Given some $p \in [1, \infty]$, $\hat{d} \in \mathbb{Q}^{\mathcal{E}_f}$, and $\Gamma \in \mathbb{Q}_{\geq 0}$, we define the corresponding norm uncertainty set as

$$\mathcal{U}^p_{\hat{d},\Gamma} := \{d \in \mathbb{Q}^{\mathcal{E}_f} \mid \|d - \hat{d}\|_p \leq \Gamma\},$$

where the p-norm $\|\cdot\|_p$ of some $d' \in \mathbb{Q}^{\mathcal{E}_f}$ is given by $\|d'\|_p = \left(\sum_{e \in \mathcal{E}_f} |d'(e)|^p\right)^{1/p}$ in case $p \in [1, \infty)$, and by $\|d'\|_\infty = \max_{e \in \mathcal{E}_f} |d'(e)|$ for $p = \infty$.

Norm uncertainty can be seen as a generalization of several other types of uncertainty. First, the case $p = 1$ is closely related to the simplicial uncertainty set $\mathcal{U}_{\hat{d},\Gamma}$ defined in Section 4.4.4. Note that the only difference is that the 1-norm uncertainty set $\mathcal{U}^1_{\hat{d},\Gamma}$ allows to set $d(e) < \hat{d}(e)$, while the simplicial uncertainty set requires $d(e) \geq \hat{d}(e)$, for all $e \in \mathcal{E}_f$. However, for many problems, only deviations from $\hat{d}$ in one direction are reasonable anyway, such that this does not mark a significant difference. For our problem, we will also show this in the proof of Theorem 4.42. If $p = \infty$, we obtain a special case of interval uncertainty, where all intervals have the same length 2Γ.

For simplicial uncertainty, we have proved $\mathcal{NP}$-hardness in Section 4.4.4, while for interval uncertainty, the Robust Bilevel Continuous Knapsack Problem turned out to be solvable in polynomial time; see Section 4.4.2. In this section, we will present a generalization of the $\mathcal{NP}$-hardness results for simplicial uncertainty to the setting of general p-norm uncertainty, for any fixed $p \in [1, \infty)$; see Corollaries 4.45 and 4.47. This also includes the case $p = 2$, which corresponds to uncorrelated ellipsoidal uncertainty (see Section 2.3.2.6). Therefore, the Robust Bilevel Continuous Knapsack Problem is also $\mathcal{NP}$-hard in this setting; see Corollary 4.48.

Like in Sections 4.4.3 and 4.4.4, we show the $\mathcal{NP}$-hardness results for the problem (RBCKP-b_f) in Theorems 4.42 and 4.46, and then conclude, by Lemma 4.10, that they also hold for the Robust Bilevel Continuous Knapsack Problem under Assumption (uni′); see Corollaries 4.45 and 4.47.

Theorem 4.42. *Let $p \in [1, \infty)$. Then the problem stated in (RBCKP-b_f), with p-norm uncertainty, is $\mathcal{NP}$-hard.*

Proof. We present a reduction from the Subset Sum Problem that is very similar to the one in the proof of Theorem 4.34. Define the same instance of (RBCKP-b_f) as there, except for the uncertainty set. The p-norm uncertainty set $\mathcal{U}^p_{\hat{d},\Gamma}$ is now defined by $\hat{d}(e_i) = 2Mw_i - w_i^{1/p}$ for $i \in \{1,\dots,m\}$, $\hat{d}(e_{m+1}) = 2M^2$, and $\Gamma = W^{1/p}$. By the same reasoning as in the proof of Theorem 4.34, we may assume that an optimum adversary's solution satisfies $d(e_i) \in \{\hat{d}(e_i), \hat{d}(e_i) + w_i^{1/p}\}$ for all $i \in \{1,\dots,m\}$ and $d(e_{m+1}) = \hat{d}(e_{m+1})$. For this, note that, although it is now allowed to deviate from $\hat{d}$ in any direction, the adversary has no incentive to choose $d(e_i) < \hat{d}(e_i)$ for any $i \in \{1,\dots,m+1\}$ in the given instance. In particular, it is cheaper in terms of the constraint $\sum_{i=1}^{m+1} |d_i - \hat{d}_i|^p \leq \Gamma^p$ to increase all values $\frac{d(e_1)}{a(e_1)},\dots,\frac{d(e_m)}{a(e_m)}$ by some $\varepsilon > 0$ than to decrease $d(e_{m+1})/a(e_{m+1})$ by ε, i.e.,

$$\sum_{i=1}^{m} (\varepsilon a(e_i))^p = \varepsilon^p \sum_{i=1}^{m} w_i^p \leq \varepsilon^p \left(\sum_{i=1}^{m} w_i\right)^p < \varepsilon^p M^p = (\varepsilon a(e_{m+1}))^p.$$

The rest of the proof is analogous to the proof of Theorem 4.34. □

Remark 4.43. In the proof of Theorem 4.42, we implicitly assumed that we can compute p-th roots in polynomial time when defining $\hat{d}$ and Γ. In general, these values cannot be computed or even represented exactly in a polynomial-time algorithm. However, they only influence the set of possible follower's greedy orders and therefore have an indirect impact on the leader, but do not appear in the leader's solutions or her objective function values. For modeling the correct set of possible orders, a sufficiently good approximation of the p-th roots, which can be computed in polynomial time, leads to the same result.

Remark 4.44. In the situation of Theorem 4.42, Remark 4.35 still applies: While the proof is formulated for the pessimistic setting, the variation proposed in Remark 4.35 allows to show $\mathcal{NP}$-hardness also in the optimistic setting.

Theorem 4.42, together with Lemma 4.10, implies:

Corollary 4.45. *Let $p \in [1,\infty)$. The Robust Bilevel Continuous Knapsack Problem with p-norm uncertainty is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′).*

Analogously to Theorem 4.37, the arguments used to prove Theorem 4.42 also imply that the adversary's problem is $\mathcal{NP}$-hard in case of norm uncertainty:

Theorem 4.46. *Let $p \in [1,\infty)$. Then the adversary's problem of the problem stated in (RBCKP-b_f), with p-norm uncertainty, and given some feasible leader's solution, is $\mathcal{NP}$-hard.*

Again, using Lemma 4.10, we can conclude:

Corollary 4.47. *Let $p \in [1,\infty)$. Then the adversary's problem of the Robust Bilevel Continuous Knapsack Problem with p-norm uncertainty, and given some feasible leader's solution, is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′).*

Since the 2-norm uncertainty set $\mathcal{U}^2_{\hat{d},\Gamma}$ is an axis-parallel ellipsoid (see Section 2.3.2.6), we obtain the following result as a special case of the $\mathcal{NP}$-hardness results above:

Corollary 4.48. *The Robust Bilevel Continuous Knapsack Problem with (uncorrelated) ellipsoidal uncertainty is $\mathcal{NP}$-hard, even if the instance satisfies Assumption (uni′). The same holds for the problem stated in (RBCKP-b_f) and for the adversary's problems of both robust bilevel problems.*

This finishes our investigation of the Robust Bilevel Continuous Knapsack Problem with all types of uncertainty sets introduced in Section 2.3.2.

4.5 Conclusion and Related Problems

In this chapter, we have extended the algorithmic results presented for the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) in Section 3.4.1.2 to the more general Robust Bilevel Continuous Knapsack Problem. This concerns polynomial-time algorithms for the problem without uncertainty and for the robust problems with discrete uncertainty and interval uncertainty, as well as pseudopolynomial-time algorithms for the setting of discrete uncorrelated uncertainty. The generalizations of the algorithms were relatively straightforward and required to include the item sizes appropriately. We used that key properties of the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) remain true for the Robust Bilevel Continuous Knapsack Problem, namely that the underlying single-level problem and therefore the follower's problem can be solved by a simple greedy algorithm and, related to this, that the leader's objective function can be represented by a piecewise linear function.

However, we could also demonstrate that the Robust Bilevel Continuous Knapsack Problem can be more complex than its special case. In particular, for discrete uncorrelated uncertainty, the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) is solvable in polynomial time and we showed that the Robust Bilevel Continuous Knapsack Problem is weakly $\mathcal{NP}$-hard. Besides this hardness result, we have also proved $\mathcal{NP}$-hardness for the adversary's and the leader's problems in the settings of simplicial uncertainty and p-norm uncertainty, thus covering all of the common types of uncertainty sets introduced in Section 2.3.2.

The complexity results presented in this chapter add to the insights about robust bilevel optimization obtained in Section 2.4.2 and Chapter 3. We will discuss and summarize them in Section 6.2. Open questions arising from this chapter will be discussed in Section 6.3.2.

While our research was focused on uncertain follower's item values, one may also consider the variant of the Robust Bilevel Continuous Knapsack Problem in which the (follower's) item sizes are uncertain. This was investigated in [Hüg20], considering different types of uncertainty sets similarly to here, but also only for the problem variant (RBCKP-b_f), as in [BH22]. In case of discrete uncertainty, the algorithm presented in Section 4.4.1 could be transferred directly. Regarding interval uncertainty, different types of leader's objective functions were distinguished and one has to be very careful about the optimistic setting because, with uncertain item sizes, the optimum value might not be attained in the adversary's problem. Many structural results were established and an algorithm similar to the one presented in Section 4.4.2 was shown to solve the problem under mild additional assumptions. Similarly to the results in Sections 4.4.3.1 and 4.4.4, $\mathcal{NP}$-hardness was also proved for uncertain item sizes lying in a discrete uncorrelated or a polytopal uncertainty set, assuming the pessimistic setting in the latter case.

Besides the Robust Bilevel Continuous Knapsack Problem, one can also investigate a stochastic version of the Bilevel Continuous Knapsack Problem. This has been done in [BHI22], while, like in [BH22], the restricted problem variant (BCKP-b_f) is used as the underlying bilevel optimization problem. We assumed that the follower's item values are again uncertain and now given by a probability distribution, while the leader aims at optimizing the expected value of her objective function. In case the possible scenarios, together with their probabilities, are given explicitly as part of the input, the problem can be solved in polynomial time. This setting can be seen as a stochastic equivalent of discrete uncertainty and therefore matches the result obtained in Section 4.4.1. If the follower's item values are independently and uniformly distributed, we showed that the problem is $\#\mathcal{P}$-hard. This holds for both the discrete and the continuous setting. These can be seen as stochastic analogs of discrete uncorrelated uncertainty and interval uncertainty, respectively. This suggests that the stochastic setting is in some sense harder than the robust one, where interval uncertainty still leads to a problem that is solvable in polynomial time (see Section 4.4.2). Moreover, $\#\mathcal{P}$-hard problems and therefore stochastic bilevel optimization problems can be considered to be at least as hard as all problems in the polynomial-time hierarchy [Tod91], while robust bilevel optimization problems naturally fall into classes of the polynomial-time hierarchy; see also Section 2.4.2. On the other hand, like the $\mathcal{NP}$-hardness result presented in Section 4.4.3 for discrete uncorrelated uncertainty, also the $\#\mathcal{P}$-hardness results in [BHI22] are only in the weak sense and complemented by pseudopolynomial-time algorithms.

5 The Bilevel Minimum Spanning Tree Problem

In this chapter, we will investigate the Bilevel Minimum Spanning Tree Problem, a bilevel combinatorial optimization problem that has a similar structure as the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem in Chapters 3 and 4 (all three problems are partitioned-items bilevel optimization problems), but is not a direct generalization of them. More precisely, both the Selection Problem and the Minimum Spanning Tree Problem can be seen as matroid optimization problems, corresponding to the uniform matroid and the graphic matroid, respectively. Accordingly, the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem are both special cases of the Bilevel Minimum Matroid Basis Problem. It turns out that some basic results from Chapter 3 hold for general matroids and can therefore be applied also to the Minimum Spanning Tree Problem and the Bilevel Minimum Spanning Tree Problem. For more details on the common features of the problems, we also refer to Section 6.1.

In contrast to Chapters 3 and 4, we will focus on the Bilevel Minimum Spanning Tree Problem without uncertainty, although one could define robust problem versions in the same way as before. However, the Bilevel Minimum Spanning Tree Problem turns out to be $\mathcal{NP}$-hard already without uncertainty (see Theorem 5.23). Hence, robust problem versions can be expected to be at least $\mathcal{NP}$-hard as well and could even be Σ_2^{p}-hard.

After a short introduction of the basic notions on graphs and the Minimum Spanning Tree Problem in Section 5.1, we will define the Bilevel Minimum Spanning Tree Problem and discuss problem variants, basic properties, and related work in Section 5.2. In Sections 5.3 and 5.4, we then consider different types of restrictions on the set of allowed instances and investigate their relations. In Section 5.5, we present an approximation-preserving reduction from the Steiner Forest Problem to the Bilevel Minimum Spanning Tree Problem and derive our main complexity results, in particular $\mathcal{NP}$-hardness of our problem. On the positive side, some results concerning fixed-parameter tractability are presented in Section 5.6, and in Section 5.7, we devise an approximation algorithm for the Bilevel Minimum Spanning Tree Problem.

In the previous chapters and in this chapter up to Section 5.7, we concentrate on the classical setting of sum objective functions, as it is common in combinatorial optimzation. However, also the alternative setting of bottleneck objectives can be interesting to investigate. Here, the cost function to minimize is not the sum of the costs of all selected items or edges but the maximum, i.e., one pays only for the most expensive item or edge. To this end, we review the case of one or both decision makers in the Bilevel Minimum Spanning Tree Problem having a bottleneck objective function in Section 5.8. Section 5.9 concludes. This chapter is based on [BHH22].

5.1 Graph Notions and the Minimum Spanning Tree Problem

We start by introducing basic notions on graphs that will be used throughout this chapter. They follow the ones in [KV18].

Definition 5.1. A *(simple, undirected) graph* $G = (V, E)$ consists of a finite set $V(G) \coloneqq V$ of *vertices* and a finite set $E(G) \coloneqq E \subseteq \{X \subseteq V(G) \mid |X| = 2\}$ of *edges*. A graph that is not *simple* may contain *parallel* edges, i.e., several copies of an edge $\{v, w\}$.

Technically, the notation for simple graphs does not allow for parallel edges, but, for the sake of simplicity, we will use it anyway and speak of the set $E(G)$ and the edge $\{v, w\}$, although there might be several parallel edges connecting the vertices v and w. Note that Definition 5.1 does not allow for *loops*, i.e., edges both of whose endpoints are the same.

Definition 5.2. Let G be a graph. Given an edge $e = \{v, w\} \in E(G)$, the vertices v and w are *adjacent* to each other, and the edge e is *incident* to both of its *endpoints* v and w. The *degree* of a vertex $v \in V(G)$ is the number of edges incident to v. A *subgraph* of G is a graph H with $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. We also say that G *contains* H. If $V(H) = V(G)$, the subgraph H is *spanning*.

Definition 5.3. Let G be a graph and $X \subseteq V(G)$. *Contracting* X is the procedure of building a new graph from G by deleting the vertices of X and their incident edges, and adding a new vertex x and an edge $\{v, x\}$ for every edge $\{v, y\} \in E(G)$ with $v \notin X$ and $y \in X$.

Definition 5.4. A *path* is a graph $P = (\{v_1, \dots, v_{k+1}\}, \{\{v_1, v_2\}, \{v_2, v_3\}, \dots, \{v_k, v_{k+1}\}\})$ such that $k \in \mathbb{N}_0$ and $v_i \neq v_j$ for all $i, j \in \{1, \dots, k+1\}$ with $i \neq j$. We also say that P is a v_1-v_{k+1}*-path* or a path *from* v_1 *to* v_{k+1}. The vertices $v_2, \dots, v_k$ are called *interior vertices* of the path P. A *cycle* is a graph $C = (\{v_1, \dots, v_k\}, \{\{v_1, v_2\}, \{v_2, v_3\}, \dots, \{v_{k-1}, v_k\}, \{v_k, v_1\}\})$ such that $k \in \mathbb{N}$ with $k \geq 2$, and $v_i \neq v_j$ for all $i, j \in \{1, \dots, k\}$ with $i \neq j$. The *length* of a path or a cycle is the number of its edges.

Definition 5.5. A graph G is *connected* if it contains a v-w-path for all $v, w \in V(G)$. We call a subgraph H of a graph G *minimal* (or *maximal*) with respect to some property if H has the property, but no proper subgraph of H (or no subgraph of G of which H is a proper subgraph) has the property. The *connected components* of a graph are its maximal subgraphs that are connected. A *forest* is a graph that does not contain a cycle. A *tree* is a connected forest. A *star* is a tree in which at most one vertex has a degree greater than one. A *spanning tree* in a graph G is a spanning subgraph of G that is a tree. A *matching* is a graph in which every vertex has degree at most one.

Sometimes, in order to simplify notation, we will identify a path, cycle, or tree (in particular a spanning tree in a given graph) with its edge set when the vertex set is just the set of all endpoints of edges in the edge set.

There are several equivalent characterizations of trees; see [KV18, Theorem 2.4]. For instance, trees are connected graphs G with exactly $|V(G)| - 1$ edges. Moreover, a graph G contains a spanning tree if and only if G is connected, and its spanning trees are then its minimal connected spanning subgraphs.

We are now ready to define the following fundamental combinatorial optimization problem:

Minimum Spanning Tree Problem
Instance: A connected graph G and edge weights $c\colon E(G) \to \mathbb{Q}$
Task: Find a spanning tree T in G such that $c(E(T))$ is minimum.

The Minimum Spanning Tree Problem can be solved in polynomial time in a greedy fashion, as the Selection Problem (see Section 3.1) and the general Minimum Matroid Basis Problem (see Section 6.1). Here, we can use, e.g., Kruskal's algorithm [Kru56]: Given a connected graph $G = (V, E)$ with edge weights $c\colon E \to \mathbb{Q}$, iterate through E in nondecreasing order of the edges' weights, and select every edge that does not form a cycle together with previously selected edges. This algorithm can be implemented to run in $\mathcal{O}(|E| \log(|V|))$ time; see [KV18, Theorem 6.5]. For more details and literature on the Minimum Spanning Tree Problem, we refer to [KV18, Chapter 6.1].

5.2 Introduction of the Bilevel Minimum Spanning Tree Problem

In this section, we first define the Bilevel Minimum Spanning Tree Problem and give some basic insights about it in Section 5.2.1. In Sections 5.2.2 and 5.2.3, we then describe possible applications of and other literature on the Bilevel Minimum Spanning Tree Problem, together with its relation to our results.

5.2.1 Problem Definition and Basic Properties

We now introduce the following bilevel variant of the Minimum Spanning Tree Problem, in which two players build a spanning tree together, according to different objective functions. It has the same structure as the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem, in the sense that they are all partitioned-items bilevel problems; see Section 2.2.2.2. In fact, both the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem are special cases of the Bilevel Minimum Matroid Basis Problem; see Section 6.1.

Bilevel Minimum Spanning Tree Problem
Instance: A connected graph G, leader's and follower's edge sets $\mathcal{E}_l \subseteq E(G)$ and $\mathcal{E}_f \subseteq E(G)$, respectively, with $\mathcal{E}_l \cup \mathcal{E}_f = E(G)$, and leader's and follower's edge weights $c\colon E(G) \to \mathbb{Q}_{\geq 0}$ and $d\colon \mathcal{E}_f \to \mathbb{Q}_{\geq 0}$, respectively
Task: Find a subset $X \subseteq \mathcal{E}_l$ for the leader such that $c(X \cup Y)$ is minimal, where $Y \subseteq \mathcal{E}_f \setminus X$ is a follower's subset that makes $(V(G), X \cup Y)$ a spanning tree in G and minimizes $d(Y)$.

We can write the Bilevel Minimum Spanning Tree Problem as a bilevel mathematical program as follows:

$$
\begin{aligned}
&\min_{X} \; c(X \cup Y) \\
&\text{s.t. } X \subseteq \mathcal{E}_l \\
&\qquad Y \in \operatorname*{argmin}_{Y'} \; d(Y') \\
&\qquad\qquad \text{s.t. } Y' \subseteq \mathcal{E}_f \setminus X \\
&\qquad\qquad\qquad X \cup Y' \text{ is a spanning tree in } G
\end{aligned}
\tag{BMSTP}
$$

Note that we assume all edge weights to be nonnegative in the problem definition. For many of our considerations, this assumption is not relevant. However, as we will also talk about approximating the Bilevel Minimum Spanning Tree Problem, we sometimes need to ensure that the objective values are nonnegative. For convenience, we assume it to be always the case. When one is only interested in exact solutions but not in approximation, this assumption is even without loss of generality. Indeed, it does not make a difference for the leader's optimization problem to add some constant to each leader's edge weight because every spanning tree $X \cup Y$ has the same cardinality. Similarly, every follower's edge weight can be shifted by a constant because, given a fixed leader's solution X, every feasible follower's response has the same cardinality. However, the resulting reduction is not approximation-preserving.

As usual, one has to distinguish the optimistic and the pessimistic version of this bilevel optimization problem, while the way in which the problem definition and (BMSTP) are stated is imprecise about this; see Section 2.2.1 for more details. However, we will see below that there are no relevant differences between the optimistic and the pessimistic version of the Bilevel Minimum Spanning Tree Problem, when it comes to the complexity of or algorithmic approaches for the problem. In contrast to that, we will have to be more careful when dealing with bottleneck versions of the Bilevel Minimum Spanning Tree Problem; see Section 5.8. In one of the bottleneck cases, we will even see that the pessimistic problem version is solvable in polynomial time, while the optimistic version is $\mathcal{NP}$-hard.

If the leader selects some edge set X rendering the follower's problem infeasible, then, by definition, this choice is not valid for her. In particular, the leader must choose X to be a forest and such that the graph $(V(G), X \cup \mathcal{E}_f)$ is connected, and the follower will augment X to a spanning tree at minimum cost according to his own objective function d. The objective function minimized by the leader is the total cost of the resulting spanning tree with respect to c.

Checking the conditions that make some set $X \subseteq \mathcal{E}_l$ a feasible leader's solution as well as solving the resulting follower's problem can be done in a straightforward way:

Lemma 5.6. *Given an instance of the Bilevel Minimum Spanning Tree Problem and a set X of leader's edges, one can determine whether X is a feasible leader's solution and, if this is the case, compute an optimum solution of the follower's problem, all in polynomial time.*

Proof. Let $(G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ be an instance of the Bilevel Minimum Spanning Tree Problem and $X \subseteq \mathcal{E}_l$. We first check whether X contains a cycle, as this would render it infeasible for the leader. Otherwise, starting from the graph G, we contract all edges in X (or, equivalently, the vertex set of each connected component of $(V(G), X)$). In the resulting graph, we restrict ourselves to the edges in $\mathcal{E}_f$. The follower's task is now equivalent to solving a Minimum Spanning Tree Problem in this graph, which can be done in polynomial time, e.g., using Kruskal's algorithm [Kru56] (see also Section 5.1). If it turns out that the graph is not connected, X is again infeasible for the leader because the follower cannot complete the solution to a spanning tree. Otherwise, X is a feasible leader's solution and the contraction procedure clearly guarantees that the resulting solution $X \cup Y$ connects all vertices in $V(G)$ and does not contain a cycle.

Alternatively, the follower can perform Kruskal's algorithm on the edges in $\mathcal{E}_f$ without contracting the edges in X before, but starting with the edge set X instead of the empty set in the initialization phase of the algorithm. Clearly, both approaches are equivalent.

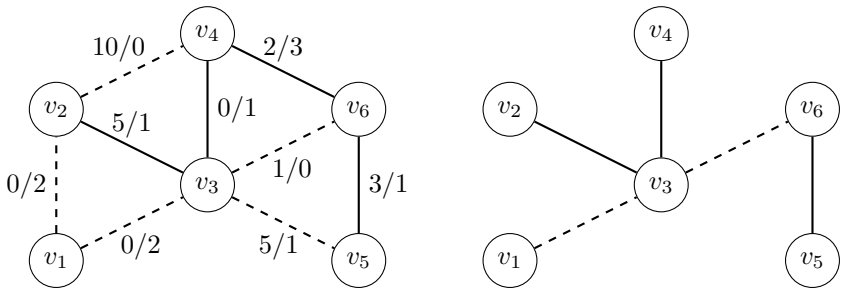


Figure 5.1. Example of an instance of the Bilevel Minimum Spanning Tree Problem and its optimum solution together with the corresponding response of the follower. The edge sets $\mathcal{E}_l$ and $\mathcal{E}_f$ are represented as solid and dashed edges, respectively. The labels show the leader's and the follower's cost of an edge $e \in E(G)$ in the form $c(e)/d(e)$.

In case the follower's optimum solution is not unique, he has to act according to the optimistic or pessimistic setting. Both options can easily be integrated in the procedure described above; see Remark 5.7. □

Remark 5.7. In order to make the Bilevel Minimum Spanning Tree Problem well-defined in case the follower's optimum solution is not unique, we will always assume that the follower chooses his solution greedily according to some given order of preference that is consistent with his cost function d. It is easy to verify that both the optimistic and the pessimistic version of the Bilevel Minimum Spanning Tree Problem can be modeled in this way, namely by using the leader's cost function c as a secondary criterion in case of several edges having the same follower's costs. This is very similar as for the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem, as all three problems have in common that the follower's problem can be solved in a greedy way and that the follower's solution is paid for in the leader's objective as well. Note that the follower's feasible set is uniquely determined by the connected components of the graph $(V(G), X)$, and therefore also his response Y when assuming any deterministic strategy to resolve non-uniqueness.

As in the case of more general bilevel optimization problems with a tractable follower's problem, it follows that the leader's problem is $\mathcal{NP}$-easy; see also Lemma 2.32. Hence, the $\mathcal{NP}$-hardness results that we will prove imply that the Bilevel Minimum Spanning Tree Problem is $\mathcal{NP}$-equivalent.

Corollary 5.8. *The Bilevel Minimum Spanning Tree Problem is $\mathcal{NP}$-easy.*

The following example illustrates the definition of the Bilevel Minimum Spanning Tree Problem and the behavior of the two decision makers.

Example 5.9. In Figure 5.1, we give an example of an instance of the Bilevel Minimum Spanning Tree Problem and its optimum solution. Note that, as we defined the problem above, leader's edges do not need to have any follower's weights because they do not appear in his objective function. However, this can be different in the setting of bottleneck objectives; see Remark 5.11. Therefore, all edges have leader's and follower's costs in the example.

The cost of the leader's optimum solution in Figure 5.1 is 9. In contrast to the follower, it is not optimum for the leader to select edges in a greedy way since taking the edge $\{v_4, v_6\}$ into her solution would result in overall costs of at least 10. It is cheaper for her to let the follower connect the components $\{v_2, v_3, v_4\}$ and $\{v_5, v_6\}$ with each other. However, this strategy relies on the fact that the edge $\{v_3, v_6\}$ is cheaper than $\{v_3, v_5\}$ also for the follower.

Remark 5.10. The definition of the Bilevel Minimum Spanning Tree Problem above does not require the leader's and follower's edge sets to be disjoint. However, in this chapter, we will always assume them to be disjoint, and we argue that this is not a restriction. In fact, the Bilevel Minimum Spanning Tree Problem can be reduced to the Bilevel Minimum Spanning Tree Problem on instances with $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$ as follows. Given an instance $(G, \mathcal{E}_l, \mathcal{E}_f, c, d)$, we replace every edge $e \in \mathcal{E}_l \cap \mathcal{E}_f$ by a pair of parallel edges e' and e'' between the same vertices in G, replace e in $\mathcal{E}_l$ by e' and in $\mathcal{E}_f$ by e'', and set $c(e') := c(e'') := c(e)$ and $d(e'') := d(e)$. The two instances are equivalent because a spanning tree cannot contain any parallel edges. By this construction, any algorithmic results that we will prove for the Bilevel Minimum Spanning Tree Problem with $\mathcal{E}_l \cap \mathcal{E}_f = \emptyset$ also apply to the general problem.

Remark 5.11. Besides the problem with sum objective functions, in Section 5.8, we will also consider bottleneck versions of the Bilevel Minimum Spanning Tree Problem, meaning that the leader and/or the follower only pay for the most expensive edge instead of the sum over the costs of all selected edges. In case the follower has a bottleneck objective function, one has to distinguish between two possible models: Either the follower pays for the most expensive edge he selects himself, i.e., his objective function is to minimize $\max_{e \in Y'} d(e)$, or he pays for the most expensive edge selected by any of the two actors, i.e., he minimizes $\max_{e \in X \cup Y'} d(e)$; the latter case is the only situation in which the follower's cost d of edges in $\mathcal{E}_l$ is relevant. These two models are not equivalent, in contrast to the sum objective case, where $d(X \cup Y') = d(X) + d(Y')$ in any optimum solution and hence the two objectives only differ by $d(X)$, which is constant from the follower's perspective. This is the same situation as in the Bilevel Selection Problem; see Remark 3.3. For more details on the case of bottleneck objective functions, we refer to Section 5.8.

5.2.2 Applications

In a possible application of the Bilevel Minimum Spanning Tree Problem (see [SZP19]), the two decision makers can be imagined as a central and a local government whose common task is to design a transportation network connecting a given set of facilities. Hence, together they have to construct a spanning tree, where each of the decision makers may control a different set of potential links, e.g., federal highways are controlled by the central government in contrast to local roads, or building grounds are owned by different actors. First, the central government constructs some of the connections, and then the local government decides how to complete the network. The central government, as the leader, pays for the construction of all connections, while the follower optimizes a different objective function, e.g., respecting requests of the citizens. Another possible application is described in [Gas02] and deals with a communication network between cities. A state is modeled as the leader and assumed to own some communication connections, while a

private company, the follower, is permitted to build its own connections in specific places. The state subsidizes the connections built by the company, i.e., the actual building costs are shared between the two actors. In turn, the company is required to ensure that each city can communicate with each other city using some activated state-owned and the new private connections, i.e., that the result is a spanning tree. The decision process now has a hierarchical structure: First, the state decides which of its own connections are to be activated and thus paid for. Second, the company decides which new connections to build such that all cities are connected and the costs incurred for the company are minimized. However, the company's decision influences the state's costs as well, because of the subsidies. Hence, the state has to anticipate what the company will do already when making the first decision.

5.2.3 Related Work and Overview of Results

Under the assumption that the leader's and the follower's edge sets are not disjoint, but that the follower controls all edges, i.e., that $\mathcal{E}_l \subseteq \mathcal{E}_f = E(G)$, the Bilevel Minimum Spanning Tree Problem has recently been considered in [SZP19; Shi21]. The authors showed that the problem is tractable in case the leader or the follower (or both) optimize a bottleneck instead of the sum objective function, where the follower is assumed to minimize $\max_{e \in X \cup Y'} d(e)$ in the bottleneck case. Related results have also been obtained in [Gas02]. Gassner considered the problem version in which $\mathcal{E}_l$ and $\mathcal{E}_f$ are disjoint and the follower's objective is $\max_{e \in Y'} d(e)$ in the bottleneck case; see also Remark 5.11. Polynomial-time algorithms are presented for the cases where the leader has a bottleneck objective and the follower either has a sum or a bottleneck objective, while restricting to the pessimistic problem version in the latter case. In [SPR23], (single-level) mixed integer linear programming formulations for some variants of the Bilevel Minimum Spanning Tree Problem are derived.

Other variants of bilevel optimization problems dealing with minimum spanning trees are considered in the literature, but in contrast to the problem addressed here, they usually assume the leader to choose the prices of some edges, while the follower solves a Minimum Spanning Tree Problem on all edges according to these costs; see, e.g., [LPP21] and the references therein. These problems fall into the category of price-setting problems, a type of combinatorial bilevel optimization problems that is also frequently investigated, but whose structure is different from the one of our problems;

As already mentioned in Section 2.2.2.2, a bilevel assignment problem of the same structure as the bilevel optimization problems investigated in this work, was studied in [GK09; FMW22]. Also in these works, not only sum but also bottleneck objective functions are considered.

In [SZP19], the authors conjecture that the version of the Bilevel Minimum Spanning Tree Problem in which both leader and follower have a sum objective is $\mathcal{NP}$-hard. Our main result is a proof of this conjecture. More specifically, we show that the Bilevel Minimum Spanning Tree Problem is at least as hard as the Steiner Forest Problem, hence it is not approximable to within a factor of $^{96}/_{95}$ unless $\mathcal{P} = \mathcal{NP}$ [CC08]. We can show the same result for the special case where the follower only controls a matching, and give some evidence that the problem might remain intractable even when the follower controls only a fixed number of edges. These hardness results can be found in Section 5.5. We also show in Section 5.4 that certain assumptions on the structure of the problem can

be made without loss of generality, e.g., that the follower controls a tree or that the leader controls a connected graph.

We emphasize that we consider a more general variant of the Bilevel Minimum Spanning Tree Problem than Shi, Zeng, and Prokopyev in [SZP19], in terms of the edges controlled by the follower. However, we will show that our hardness results also hold for the problem version in [SZP19]; see Corollaries 5.20 and 5.25.

In view of the negative complexity results mentioned above, one can expect only very limited positive results. We are able to devise a $(|V(G)| - 1)$-approximation algorithm for the Bilevel Minimum Spanning Tree Problem (see Section 5.7) and show that 2-approximating the optimum solution is fixed-parameter tractable in the number of edges controlled by the follower. For the same parameter, the decision whether a given follower's response can be enforced by the leader is fixed-parameter tractable, which implies that the variant of the Bilevel Minimum Spanning Tree Problem with uniform costs $c(e)$ for all $e \in \mathcal{E}_l$ is fixed-parameter tractable as well. The results regarding fixed-parameter tractability can be found in Section 5.6. In Section 5.8, we focus on the bottleneck cases. We show that the problem is tractable in case the leader has a bottleneck objective and the follower has a sum objective, while it is $\mathcal{NP}$-hard when the leader has a sum objective and the follower has a bottleneck objective. If both have a bottleneck objective, the problem turns out to be polynomial-time solvable in the pessimistic setting, while it is hard to solve in the optimistic case. An overview of our results for different objective functions, together with the related results from [Gas02; SZP19], can be found in Table 5.1 in Section 5.8 below.

5.3 Restricted Problem Variants

In this section, we introduce sets of instances of the Bilevel Minimum Spanning Tree Problem with certain structural properties, some of them similar to the assumptions we discussed for the Bilevel Selection Problem; see Section 3.2.2. We will then show in Section 5.4 that, without loss of generality, we may restrict ourselves to some of them. Our aim is to simplify some of the proofs later on, but also to clarify the connections between different settings corresponding to reasonable restricted problem variants, which sometimes lead to different complexity results.

Let $\mathcal{I}$ be the set of all instances $I = (G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ of the Bilevel Minimum Spanning Tree Problem as described in Section 5.2.1. As already mentioned in Remark 5.10, we assume throughout that $\mathcal{E}_l$ and $\mathcal{E}_f$ are disjoint sets. We now define the following subsets of $\mathcal{I}$, all corresponding to certain restrictions on the edge sets controlled by leader and follower:

- $\mathcal{I}_{\mathcal{E}_l\,\text{conn}}$, the set of instances for which the leader's graph $(V(G), \mathcal{E}_l)$ is connected,
- $\mathcal{I}_{\mathcal{E}_l\,\text{forest}}$, the set of instances for which the leader's graph $(V(G), \mathcal{E}_l)$ is a forest,
- $\mathcal{I}_{\mathcal{E}_f\,\text{conn}}$, the set of instances for which the follower's graph $(V(G), \mathcal{E}_f)$ is connected,
- $\mathcal{I}_{\mathcal{E}_f\,\text{forest}}$, the set of instances for which the follower's graph $(V(G), \mathcal{E}_f)$ is a forest,
- $\mathcal{I}_{\mathcal{E}_f\,\text{matching}}$, the set of instances for which the follower's graph $(V(G), \mathcal{E}_f)$ is a matching, and
- $\mathcal{I}_{\mathcal{E}_f\,\text{all}}$, the set of instances such that, for each leader's edge in $\mathcal{E}_l$, there exists a parallel follower's edge in $\mathcal{E}_f$ with the same leader's cost.

The instances in $\mathcal{I}_{\mathcal{E}_f\,\text{all}}$ exactly correspond to those considered in [SZP19], after our construction that ensures disjointness of $\mathcal{E}_l$ and $\mathcal{E}_f$; see Remark 5.10. Since G is always

assumed to be connected, we have $\mathcal{I}_{\mathcal{E}_f \text{ all}} \subseteq \mathcal{I}_{\mathcal{E}_f \text{ conn}}$, and $\mathcal{I}_{\mathcal{E}_f \text{ conn}}$ is precisely the set of instances where any forest selected by the leader is feasible, i.e., for any forest $X \subseteq \mathcal{E}_l$, there is at least one response $Y' \subseteq \mathcal{E}_f$ of the follower such that $X \cup Y'$ is a spanning tree in G. Moreover, we have $\mathcal{I}_{\mathcal{E}_f \text{ matching}} \subseteq \mathcal{I}_{\mathcal{E}_f \text{ forest}}$, as every matching is a forest. We will also be interested in the case of instances from $\mathcal{I}_{\mathcal{E}_f \text{ conn}} \cap \mathcal{I}_{\mathcal{E}_f \text{ forest}}$, in which the follower's graph $(V(G), \mathcal{E}_f)$ is a tree.

The distinction we make here about the edge sets controlled by the leader and the follower, respectively, are comparable to the Assumptions (arb), (sub), and (disj) we investigated in case of the Bilevel Selection Problem in Chapter 3. However, Remark 5.10 shows that it is not a restriction here to always assume disjointness of $\mathcal{E}_l$ and $\mathcal{E}_f$, while, for the Bilevel Selection Problem, Assumption (disj) renders the problem significantly easier than the general case of Assumption (arb), at least in the robust setting; see, e.g., Theorems 3.25 and 3.58. The intuitive reason for this difference between the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem is that the required spanning tree structure of a solution for the Bilevel Minimum Spanning Tree Problem causes more complex interactions between the leader's solution and possible follower's responses, even if the edge sets are disjoint. In particular, while the only interaction in case of the Bilevel Selection Problem under Assumption (disj) concerns the number of items to select by each decision maker, in the Bilevel Minimum Spanning Tree Problem, the leader can exclude certain follower's edges from his feasible solutions by her choice of certain edges. In the easiest case, the excluded follower's edge is parallel to the one selected by the leader. Therefore, disjointness is not a significant simplification here.

In case of the Bilevel Selection Problem, we mentioned also Assumption (uni), as a special case of Assumption (disj), in which the leader's costs are uniform on her own edges. Also here, the corresponding special case is interesting, as we will prove that the Bilevel Minimum Spanning Tree Problem is fixed-parameter tractable in the number of follower's edges in this case; see Corollary 5.32.

On the other hand, the case in which the follower controls everything corresponds to $\mathcal{I}_{\mathcal{E}_f \text{ all}}$ and to the setting of [SZP19] for the Bilevel Minimum Spanning Tree Problem and to Assumption (sub) for the Bilevel Selection Problem. In the latter case, we have seen a polynomial reduction from the general setting of Assumption (arb) to the one of Assumption (sub) in Lemma 3.15. Using similar arguments, we will obtain the corresponding result for the Bilevel Minimum Spanning Tree Problem as well; see Corollary 5.20.

Moreover, we focus on the combinatorial version of the Bilevel Minimum Spanning Tree Problem, without continuous relaxations, i.e., corresponding to Assumption (bb) for the Bilevel Selection Problem. However, it might also be interesting to study a problem variant of the Bilevel Minimum Spanning Tree Problem corresponding to Assumption (cc) for the Bilevel Selection Problem; see also Sections 6.1 and 6.3.3.

Finally, we always require that a spanning tree and not only a spanning forest is found, i.e., a basis and not only an independent set of the graphic matroid, corresponding to Assumption (=) for the Bilevel Selection Problem; see also Section 6.1.

5.4 Equivalence Between Problem Variants

We will now establish connections between the variants of the Bilevel Minimum Spanning Tree Problem defined by the restricted sets of instances introduced in Section 5.3 by presenting appropriate reductions. All reductions are polynomial and approximation-preserving, i.e., they can be used to transform an approximation algorithm for one problem into an approximation algorithm with the same guarantee for the other problem; see also Section 2.1.2.

Our first reduction shows that we may assume that the edges controlled by the follower connect all vertices of G.

Lemma 5.12. *The Bilevel Minimum Spanning Tree Problem on $\mathcal{I}$ can be polynomially reduced to the Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_f \text{ conn}}$. The reduction preserves $\mathcal{I}_{\mathcal{E}_l \text{ conn}}$, $\mathcal{I}_{\mathcal{E}_l \text{ forest}}$, and $\mathcal{I}_{\mathcal{E}_f \text{ forest}}$, meaning that if we start with an instance in one of these sets, the reduction again results in an instance in this set.*

Proof. Let $I = (G, \mathcal{E}_l, \mathcal{E}_f, c, d) \in \mathcal{I}$. We construct an instance $I' \in \mathcal{I}_{\mathcal{E}_f \text{ conn}}$ from I by adding arbitrary edges controlled by the follower in order to make $(V(G), \mathcal{E}_f)$ connected. The new edges e' have cost $c(e') := d(e') := M$ for some large enough number M, e.g., one can set $M := \sum_{e \in E(G)} \max\{c(e), d(e)\} + 1$.

Every solution of I is also a solution of I' of the same cost for both leader and follower. The follower's solution is still optimum because, by the choice of M, selecting one of the new edges can only make the solution worse for him. Conversely, given a solution of I', it is also a solution of I of the same cost if it does not contain any new edges. Otherwise, the leader's solution of I' is not a feasible choice in I, as the follower will only select a new edge in I' if he cannot produce all necessary connections using only the original edges. In this case, any feasible solution of I is cheaper than the one of I', due to the choice of M.

Since $\mathcal{E}_l$ is not changed, the reduction preserves all structural properties of $\mathcal{E}_l$, in particular $(V(G), \mathcal{E}_l)$ being connected or a forest. By adding only a minimum number of edges necessary to make $(V(G), \mathcal{E}_f)$ connected, we may also assume that $(V(G), \mathcal{E}_f)$ being a forest is preserved. In this case, the resulting follower's graph is a tree. □

Using a similar construction, one can show the same result for the graph controlled by the leader:

Lemma 5.13. *The Bilevel Minimum Spanning Tree Problem on $\mathcal{I}$ can be polynomially reduced to the Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_l \text{ conn}}$. The reduction preserves $\mathcal{I}_{\mathcal{E}_l \text{ forest}}$, $\mathcal{I}_{\mathcal{E}_f \text{ conn}}$, $\mathcal{I}_{\mathcal{E}_f \text{ forest}}$, and $\mathcal{I}_{\mathcal{E}_f \text{ matching}}$.*

We next show an important structural result about the Bilevel Minimum Spanning Tree Problem, from which we can conclude that we may assume without loss of generality that the follower controls a forest, but which will also be useful on its own. The result is not only true for the Bilevel Minimum Spanning Tree Problem, but also for the corresponding setting of general matroids; see Section 6.1.

As stated in Remark 5.7, we assume a fixed ordering of the edges in $\mathcal{E}_f$ that the follower will always, i.e., for any choice of X, use in his greedy algorithm, and that can be defined to reflect optimism or pessimism appropriately. This fixed order is important for the following proof. Moreover, we do not require X_1 or X_2 to be feasible leader's solutions in the following statement, i.e., it might not be possible for the follower to complete them

to a spanning tree. However, we assume that the follower applies his greedy algorithm anyway and adds as many edges as possible, leading to forests $X_1 \cup Y_1$ and $X_2 \cup Y_2$.

Lemma 5.14. *Given an instance $(G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ of the Bilevel Minimum Spanning Tree Problem and two forests $X_1 \subseteq X_2 \subseteq \mathcal{E}_l$, let $Y_1, Y_2 \subseteq \mathcal{E}_f$ be the corresponding follower's responses. Then $Y_1 \supseteq Y_2$.*

Proof. Let $\mathcal{E}_f = \{e_1, \dots, e_m\}$, where $e_1, \dots, e_m$ is the follower's order of preference (see Remark 5.7), and let $Y_1^{(i)}$ and $Y_2^{(i)}$ be the partial solutions of the follower after considering edge e_i in his application of Kruskal's algorithm (see Section 5.1 and Lemma 5.6), starting from the leader's choice X_1 or X_2, respectively. It then suffices to prove the following claim: For all $i \in \{0, \dots, m\}$, each pair of vertices in $V(G)$ that is connected by a path contained in $X_1 \cup Y_1^{(i)}$ is also connected by a path in $X_2 \cup Y_2^{(i)}$. This implies that, if e_{i+1} is added to $Y_2^{(i)}$ in the next step, it is also added to $Y_1^{(i)}$. Thus, the full follower's response $Y_2^{(m)} = Y_2$ to X_2 is contained in $Y_1^{(m)} = Y_1$. We show the claim by induction over i.

Since $Y_1^{(0)} = Y_2^{(0)} = \emptyset$ and $X_1 \subseteq X_2$, there is nothing to show for the case $i = 0$. For $i \in \{1, \dots, m\}$, consider two vertices $v, w \in V(G)$ that are connected by a path in $X_1 \cup Y_1^{(i)}$. If v and w are already connected by $X_1 \cup Y_1^{(i-1)}$, they are connected by $X_2 \cup Y_2^{(i-1)}$ as well, by the induction hypothesis, and thus also by $X_2 \cup Y_2^{(i)} \supseteq X_2 \cup Y_2^{(i-1)}$. Otherwise, the connection has been established by adding $e_i = \{v_i, w_i\}$, implying that $X_1 \cup Y_1^{(i-1)}$ connects v to v_i and w to w_i (or vice versa). Again by the induction hypothesis, we derive that also $X_2 \cup Y_2^{(i-1)}$ connects v to v_i and w to w_i. Hence, either v and w are already connected by $X_2 \cup Y_2^{(i-1)}$, in which case we are done, or v_i and w_i are not connected by $X_2 \cup Y_2^{(i-1)}$. In the latter case, edge e_i will be contained in $Y_2^{(i)}$, so that v and w are connected by $X_2 \cup Y_2^{(i)}$ also in this case. □

Based on Lemma 5.14, we now prove that, without loss of generality, we can eliminate follower's edges from the graph that he would never use because they form cycles with edges that are better for him.

Corollary 5.15. *The Bilevel Minimum Spanning Tree Problem on $\mathcal{I}$ can be polynomially reduced to the Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_f \text{ forest}}$. The reduction preserves $\mathcal{I}_{\mathcal{E}_l \text{ conn}}$, $\mathcal{I}_{\mathcal{E}_l \text{ forest}}$, and $\mathcal{I}_{\mathcal{E}_f \text{ conn}}$.*

Proof. Let $I = (G, \mathcal{E}_l, \mathcal{E}_f, c, d) \in \mathcal{I}$ be an instance of the Bilevel Minimum Spanning Tree Problem and let $Y^* \subseteq \mathcal{E}_f$ be the result of Kruskal's algorithm applied to the graph $(V(G), \mathcal{E}_f)$, using the fixed order of edges defined by the follower's preferences. Note that Y^* is a forest in G, but not necessarily a spanning tree, since we do not require $(V(G), \mathcal{E}_f)$ to be connected. If the follower's graph is connected, then $X^* = \emptyset$ is a feasible leader's solution and Y^* is the follower's response to it.

Let I' be the instance that arises from I by removing the edges in $\mathcal{E}_f \setminus Y^*$ from $\mathcal{E}_f$ and from G. Then $I' \in \mathcal{I}_{\mathcal{E}_f \text{ forest}}$. For any feasible leader's solution X in I, by applying Lemma 5.14 for $X_1 := \emptyset$ and $X_2 := X$, it then follows that the follower's response Y lies in Y^*. Hence, X has the same objective value in I as in I'. As the leader's feasible set is not changed by the above transformation, we obtain the desired reduction result.

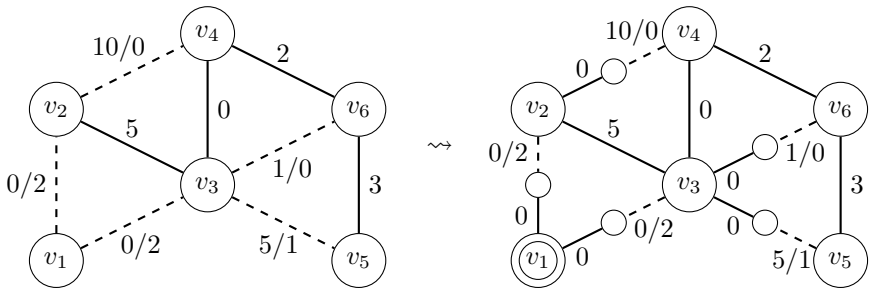


Figure 5.2. Illustration of the construction in the proof of Lemma 5.16 applied to the example instance of Figure 5.1, with vertex v_1 chosen as root.

Since $\mathcal{E}_l$ is not changed, the reduction preserves any specific structure of $\mathcal{E}_l$, in particular $(V(G), \mathcal{E}_l)$ being a forest or connected. Connectedness of $(V(G), \mathcal{E}_f)$ is obviously preserved by the construction as well. □

It is worth mentioning that, even if the follower's edge set $\mathcal{E}_f$ is a forest, the follower might have several feasible or even several optimum responses to some leader's solution X. Indeed, after the contraction of X, the follower's edges might form cycles again. For an example, consider the instance illustrated in Figure 5.1, in which the follower's edges form a tree. When the leader takes the edge $\{v_2, v_3\}$ into her solution, the vertices v_2 and v_3 can be thought of as being merged into a single vertex from the follower's perspective. This leads to the follower's edges $\{v_1, v_2\}$ and $\{v_1, v_3\}$ becoming parallel edges, of which the follower must select one. In this example, the two edges even have the same leader's and follower's cost such that the follower will select any of the two edges, depending on his preferences.

If we are not interested in the connectedness of the follower's edge set $\mathcal{E}_f$, but rather in a simple combinatorial structure of the latter, we can even further restrict $\mathcal{E}_f$ to form a matching:

Lemma 5.16. *The* Bilevel Minimum Spanning Tree Problem *on* $\mathcal{I}_{\mathcal{E}_f \text{ forest}}$ *can be polynomially reduced to the* Bilevel Minimum Spanning Tree Problem *on* $\mathcal{I}_{\mathcal{E}_f \text{ matching}}$. *The reduction preserves* $\mathcal{I}_{\mathcal{E}_l \text{ conn}}$ *and* $\mathcal{I}_{\mathcal{E}_l \text{ forest}}$.

Proof. Let $I = (G, \mathcal{E}_l, \mathcal{E}_f, c, d) \in \mathcal{I}_{\mathcal{E}_f \text{ forest}}$. From I, construct an instance $I' \in \mathcal{I}_{\mathcal{E}_f \text{ matching}}$ by applying the following transformation to every connected component of the graph $(V(G), \mathcal{E}_f)$ containing more than one edge: Define an arbitrary vertex in the connected component as its root. Replace every edge $e \in \mathcal{E}_f$ in the connected component by a path of length two, with a new vertex in the middle. The new edge e' that is closer to the root is added to $\mathcal{E}_l$ and assigned $c(e') := 0$, while the other new edge e'' replaces e in $\mathcal{E}_f$ and is assigned $d(e'') := d(e)$ and $c(e'') := c(e)$; moreover, edge e'' takes the position of e in the follower's order of preference. This construction ensures that the follower's edge set in I' forms a matching because every new vertex has only one incident follower's edge, and for every vertex v that was already present in I, only the follower's edge e'' arising from the edge e which is contained in the unique path from v to the corresponding root is incident to v. See Figure 5.2 for an illustration.

A solution for I can be transformed to a solution for I' of the same cost by adding all newly introduced leader's edges to her solution, which does not change the cost. Indeed,

the follower solves exactly the same problem after the leader's solution is contracted. For the opposite transformation, consider an optimum leader's solution X' for I'. Observe that we may assume all newly introduced edges to be in X' because otherwise, adding them would lead to the follower removing some of his edges from his response by Lemma 5.14, which cannot increase the leader's objective value. Now, remove all new edges from X' in order to get a solution X for I. Again, the follower has exactly the same choices responding to X and X', respectively. Thus, the objective value of X in I is at most the objective value of X' in I'.

Since the edges added to $\mathcal{E}_l$ connect every new vertex by exactly one edge, the reduction preserves $\mathcal{I}_{\mathcal{E}_l\,\text{conn}}$ and $\mathcal{I}_{\mathcal{E}_l\,\text{forest}}$. □

Combining the reductions from Lemma 5.12, Lemma 5.13, and Corollary 5.15, we obtain that we may assume the leader's edge set to be connected and the follower's edge set to be a tree:

Corollary 5.17. *The Bilevel Minimum Spanning Tree Problem on $\mathcal{I}$ can be polynomially reduced to the Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_l\,conn} \cap \mathcal{I}_{\mathcal{E}_f\,conn} \cap \mathcal{I}_{\mathcal{E}_f\,forest}$. The reduction preserves $\mathcal{I}_{\mathcal{E}_l\,forest}$.*

Dropping the connectedness of $\mathcal{E}_f$, we can apply Lemma 5.16 to obtain:

Corollary 5.18. *The Bilevel Minimum Spanning Tree Problem on $\mathcal{I}$ can be polynomially reduced to the Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_l\,conn} \cap \mathcal{I}_{\mathcal{E}_f\,matching}$. The reduction preserves $\mathcal{I}_{\mathcal{E}_l\,forest}$.*

As pointed out in Sections 5.2.3 and 5.3, Shi, Zeng, and Prokopyev in [SZP19] only consider instances from $\mathcal{I}_{\mathcal{E}_f\,\text{all}}$, i.e., the follower controlling many edges. This could be seen as an opposite assumption to instances being chosen from $\mathcal{I}_{\mathcal{E}_f\,\text{forest}}$ or even $\mathcal{I}_{\mathcal{E}_f\,\text{matching}}$. To show that our main complexity results still hold in the setting of [SZP19], we use the following result:

Lemma 5.19. *The Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_f\,conn}$ can be polynomially reduced to the Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_f\,all}$. The reduction preserves $\mathcal{I}_{\mathcal{E}_l\,conn}$ and $\mathcal{I}_{\mathcal{E}_l\,forest}$.*

Proof. Let $I = (G, \mathcal{E}_l, \mathcal{E}_f, c, d) \in \mathcal{I}_{\mathcal{E}_f\,\text{conn}}$. Construct an instance $I' \in \mathcal{I}_{\mathcal{E}_f\,\text{all}}$ from I by creating a copy e' of each edge $e \in \mathcal{E}_l$ that does not have a parallel follower's edge of the same leader's cost, adding e' to $\mathcal{E}_f$ and setting $c(e') \coloneqq c(e)$ and $d(e') \coloneqq M$, for some large M, e.g., $M \coloneqq \sum_{e \in E(G)} d(e) + 1$. The construction is illustrated in Figure 5.3.

All forests $X \subseteq \mathcal{E}_l$ are feasible leader's solutions for both I and I' because we assume $(V(G), \mathcal{E}_f)$ to be connected. By construction, any feasible leader's solution X leads to the same follower's response in I and I', since the additional edges are the most expensive ones for the follower and will thus never be selected because he can establish any desired connection using only the original edges.

As the reduction does not change the set $\mathcal{E}_l$, it clearly preserves all its structural properties, in particular $(V(G), \mathcal{E}_l)$ being a forest or connected. □

From Lemma 5.12 and Lemma 5.19 we derive:

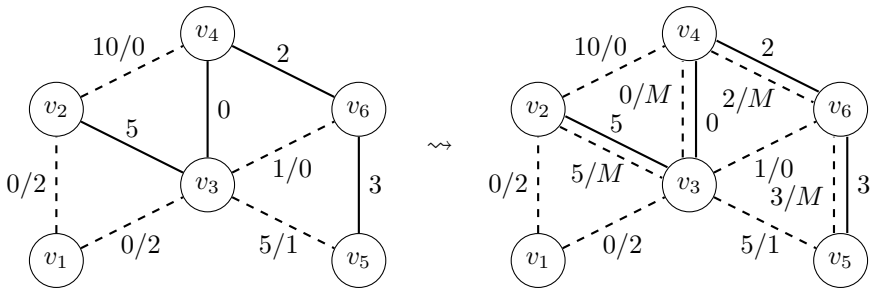


Figure 5.3. Illustration of the construction in the proof of Lemma 5.19 applied to the example instance of Figure 5.1.

Corollary 5.20. *The* Bilevel Minimum Spanning Tree Problem *on* $\mathcal{I}$ *can be polynomially reduced to the* Bilevel Minimum Spanning Tree Problem *on* $\mathcal{I}_{\mathcal{E}_f \text{ all}}$. *The reduction preserves* $\mathcal{I}_{\mathcal{E}_l \text{ conn}}$ *and* $\mathcal{I}_{\mathcal{E}_l \text{ forest}}$.

In the following sections, we will also consider the case of uniform leader's costs on the leader's edges $\mathcal{E}_l$; see also the corresponding setting of Assumption (uni) for the Bilevel Selection Problem. It seems to be easier than the general setting and we will prove that the Bilevel Minimum Spanning Tree Problem is fixed-parameter tractable in the number of follower's edges in this case; see Corollary 5.32. For the general Bilevel Minimum Spanning Tree Problem, it is an open question whether this is the case. However, the following result demonstrates that, in some sense, the Bilevel Minimum Spanning Tree Problem with uniform leader's costs is not easier than the one of more general polynomially-bounded integer costs. But we will see that it cannot be used to generalize Corollary 5.32 because the construction significantly increases the number of follower's edges.

Lemma 5.21. *The* Bilevel Minimum Spanning Tree Problem *on* $\mathcal{I}_{\mathcal{E}_f \text{ conn}}$ *with polynomially-bounded integer costs* c *on* $\mathcal{E}_l$ *can be polynomially reduced to the* Bilevel Minimum Spanning Tree Problem *with* $c(e) = 1$ *for all* $e \in \mathcal{E}_l$. *The reduction preserves* $\mathcal{I}_{\mathcal{E}_l \text{ conn}}$, $\mathcal{I}_{\mathcal{E}_l \text{ forest}}$, $\mathcal{I}_{\mathcal{E}_f \text{ conn}}$, *and* $\mathcal{I}_{\mathcal{E}_f \text{ forest}}$.

Proof. Let $I = (G, \mathcal{E}_l, \mathcal{E}_f, c, d) \in \mathcal{I}_{\mathcal{E}_f \text{ conn}}$ with polynomially-bounded integer costs c on $\mathcal{E}_l$. Construct an instance I' of the Bilevel Minimum Spanning Tree Problem with uniform costs c on $\mathcal{E}_l$ as follows: Contract $\mathcal{E}_l^0 := \{e \in \mathcal{E}_l \mid c(e) = 0\}$ in G and remove $\mathcal{E}_l^0$ from $\mathcal{E}_l$. Each edge $e = \{v, w\} \in \mathcal{E}_l \setminus \mathcal{E}_l^0$ is replaced by a path P_e of length $c(e)$, consisting of leader's edges again. In I', each of these edges has a leader's weight of 1. In addition, each interior vertex u of P_e is connected to v by a new edge e' added to $\mathcal{E}_f$ with $c(e') := 0$ and $d(e') := M$ for some large enough constant M, such as $M := \sum_{e \in E(G)} d(e) + 1$. Note that nothing changes for edges $e \in \mathcal{E}_l$ with $c(e) = 1$.

We claim that the instances I and I' have the same optimum value. Given an optimum solution X to I, we first may assume that X contains a maximal forest in $\mathcal{E}_l^0$ because otherwise, we could add an edge from $\mathcal{E}_l^0$ to X, replacing some edge e with $c(e) \geq 0$ in the resulting spanning tree. Indeed, either some edge $e \in X$ can be removed if the new edge closes a cycle in X, which does not change the follower's response, or we apply Lemma 5.14 otherwise. A feasible solution X' to I' having the same objective value as X can then be defined by setting $X' := \bigcup_{e \in X} E(P_e)$. This is true since the follower has to connect all

interior vertices of paths P_e with $e \notin X$ using the newly introduced follower's edges in order to ensure that the resulting solution is a tree. These edges have cost 0 for the leader. After adding these edges, the follower has exactly the same choices as in the instance I after the leader has selected X.

Conversely, given an optimum solution X' to I', we may assume that, for each edge $e \in \mathcal{E}_l \setminus \mathcal{E}_l^0$, either all edges in P_e belong to X' or none: Assume this is not true and consider some solution X' to I' that contradicts this property. Let Y' be the follower's response to X'. We construct a solution $X'' \subsetneq X'$ to I' with follower's response Y'' with $c(X'') < c(X')$ and $c(Y') = c(Y'')$ as follows: Let

$$X'' \coloneqq \bigcup \{P_e \mid e \in \mathcal{E}_l \setminus \mathcal{E}_l^0, P_e \subseteq X'\}$$

consist of all the paths that are entirely contained in X', i.e., we simply remove all edges of paths that were only selected partially in X'. As we assume $I \in \mathcal{I}_{\mathcal{E}_f \text{ conn}}$, which implies also $I' \in \mathcal{I}_{\mathcal{E}_f \text{ conn}}$, the leader's solution X'' is clearly feasible because the follower can complete any solution to a spanning tree. Moreover, observe that, since the edges connecting the inner vertices of the paths P_e have very high cost for the follower, they are only selected if absolutely necessary. Therefore, the response Y'' to X'' is the same as the response Y' to X' with some additional edges that connect the inner vertices of the paths P_e that are connected by X' but not by X''. This shows $c(Y') = c(Y'')$, since these additional edges have cost 0 for the leader; hence, X'' is the desired solution. Thus, we can assume that, for each $e \in \mathcal{E}_l \setminus \mathcal{E}_l^0$, either all edges in P_e belong to X' or none. Setting $X \coloneqq \{e \in \mathcal{E}_l \setminus \mathcal{E}_l^0 \mid P_e \subseteq X'\} \cup F$, where F is a maximal forest in $\mathcal{E}_l^0$, then yields a feasible solution to I with the same objective value as X' in I' because the follower's responses to X and X' have the same cost, by the same arguments as in the first part of the proof.

If one of the graphs $(V(G), \mathcal{E}_l)$ and $(V(G), \mathcal{E}_f)$ is a forest or connected, these properties are preserved because the construction ensures that the newly introduced vertices are all connected to the old vertices in an acyclic manner in both $\mathcal{E}_l$ and $\mathcal{E}_f$. □

The reduction described in the proof of Lemma 5.12 only introduces follower's edges. We can thus combine it with Lemma 5.21 to obtain the following result.

Corollary 5.22. *The* Bilevel Minimum Spanning Tree Problem *(on $\mathcal{I}$) with polynomially-bounded integer costs c on $\mathcal{E}_l$ can be polynomially reduced to the* Bilevel Minimum Spanning Tree Problem *with $c(e) = 1$ for all $e \in \mathcal{E}_l$. The reduction preserves $\mathcal{I}_{\mathcal{E}_l \text{ conn}}$, $\mathcal{I}_{\mathcal{E}_l \text{ forest}}$, $\mathcal{I}_{\mathcal{E}_f \text{ conn}}$, and $\mathcal{I}_{\mathcal{E}_f \text{ forest}}$.*

5.5 Main Hardness Results

In this section, we establish a first hardness result for the Bilevel Minimum Spanning Tree Problem, using a reduction from the

Steiner Forest Problem

Instance: A connected graph G, edge weights $w\colon E(G) \to \mathbb{Q}_{\geq 0}$, and $k \in \mathbb{N}$ disjoint terminal sets $S_1, \ldots, S_k \subseteq V(G)$

Task: Find a forest F in G such that, for each $i \in \{1, \ldots, k\}$, all vertices in S_i are in the same connected component of F, and such that $c(E(F))$ is minimum.

The best factor of an approximation algorithm that is known for the Steiner Forest Problem is 2 [Jai01] and the problem is $\mathcal{NP}$-hard to approximate within a factor of $^{96}/_{95}$ [CC08]. Note that the well-known Steiner Tree Problem is the special case of the Steiner Forest Problem where $k = 1$. We will reduce the Steiner Forest Problem to the Bilevel Minimum Spanning Tree Problem in order to obtain the following result:

Theorem 5.23. *The Bilevel Minimum Spanning Tree Problem cannot be approximated to within a factor of $^{96}/_{95}$ in polynomial time, unless $\mathcal{P} = \mathcal{NP}$, even if $\mathcal{E}_f$ is a tree.*

Proof. Let I be an instance of the Steiner Forest Problem, consisting of a graph G with edge weights $w\colon E(G) \to \mathbb{Q}_{\geq 0}$ and disjoint terminal sets $S_1, \dots, S_k \subseteq V(G)$. We construct an instance I' of the Bilevel Minimum Spanning Tree Problem as follows. The graph in I' is $G' \coloneqq (V(G), \mathcal{E}_l \cup \mathcal{E}_f)$, where $\mathcal{E}_l \coloneqq E(G)$ and $\mathcal{E}_f$ is defined as follows: First introduce arbitrary new edges forming a forest with connected components having vertex sets $S_1, \dots, S_k$ and call this edge set $\mathcal{E}_f^0$. Then, add any further edges turning $\mathcal{E}_f^0$ into a spanning tree on $V(G)$. All new edges together form the set $\mathcal{E}_f$. The cost function for the leader is

$$c(e) \coloneqq \begin{cases} w(e), & \text{if } e \in \mathcal{E}_l, \\ M, & \text{if } e \in \mathcal{E}_f^0, \\ 0, & \text{if } e \in \mathcal{E}_f \setminus \mathcal{E}_f^0, \end{cases}$$

where M is some large constant such as $\sum_{e \in E(G)} w(e) + 1$. The cost function for the follower is given by

$$d(e) \coloneqq \begin{cases} 0, & \text{if } e \in \mathcal{E}_f^0, \\ 1, & \text{if } e \in \mathcal{E}_f \setminus \mathcal{E}_f^0. \end{cases}$$

This finishes the construction of I'. We now show that any optimum solution X to I corresponds to a feasible leader's solution X' to I' of the same cost, and vice versa. So let $X \subseteq E$ be any solution to I. Then $X' \coloneqq X$ is a feasible leader's solution since X forms a forest and $\mathcal{E}_f$ connects all vertices, so that the follower can complete any forest selected by the leader to a tree. Since X connects each terminal set, the follower's response to X' does not contain any edges from $\mathcal{E}_f^0$ as they would form a cycle together with X'. Hence, the follower's response only consists of edges having cost 0 for the leader. Therefore, the overall cost for the leader is simply $c(X') = w(X)$.

It remains to show that any optimum solution X' to I' corresponds to a feasible solution X to the Steiner Forest Problem of the same cost. Clearly, there exists a leader's solution to I' of cost at most $M - 1$, e.g., one could select any spanning tree in $G = (V(G), \mathcal{E}_l)$. By optimality of X', this implies that the follower's response to X' does not contain any of the edges in $\mathcal{E}_f^0$. However, since the follower's cost for the edges in $\mathcal{E}_f^0$ is cheaper than the cost of the edges in $\mathcal{E}_f \setminus \mathcal{E}_f^0$, this implies that the leader's solution X' connects each terminal set. As X' is also a forest, it is a solution to I having cost $w(X')$. □

Remark 5.24. The definition of $\mathcal{E}_f$ in the proof of Theorem 5.23 leaves a lot of freedom concerning the structure of the follower's tree. For example, it can always be chosen to form a path. Moreover, the reduction can be performed analogously from the Steiner Tree Problem instead of the Steiner Forest Problem, i.e., where only one terminal

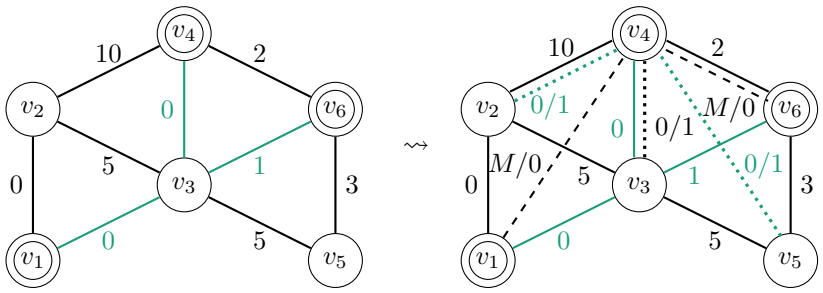


Figure 5.4. Illustration of the proof of Theorem 5.23 for $k = 1$ with $\mathcal{E}_f$ being a star. The marked vertices v_1, v_4, v_6 are the given terminals. Edges in $\mathcal{E}_f^0$ are represented as dashed edges, the remaining edges in $\mathcal{E}_f$ are drawn as dotted egdes. Any vertex of S_1 can be chosen as the center of the star, here it is v_4. Green edges mark the optimum Steiner tree in the input graph and the optimum leader's solution and corresponding follower's response in the instance of the Bilevel Minimum Spanning Tree Problem.

set S_1 is given. Then the structure of the follower's tree is even less restricted, for example, the set $\mathcal{E}_f$ can be chosen to form a star; see Figure 5.4 for an illustration. Note that the hardness result from [CC08] also holds for the Steiner Tree Problem. Thus, the hardness of Theorem 5.23 still holds for restrictions of the follower's tree's structure such as $\mathcal{E}_f$ being a path or a star.

Theorem 5.23 and Corollary 5.20 together prove the conjecture stated by Shi, Zeng, and Prokopyev in [SZP19] that the Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_f \text{ all}}$ is $\mathcal{NP}$-hard:

Corollary 5.25. *The Bilevel Minimum Spanning Tree Problem on $\mathcal{I}_{\mathcal{E}_f \text{ all}}$ cannot be approximated to within a factor of $^{96}/_{95}$ in polynomial time, unless $\mathcal{P} = \mathcal{NP}$.*

Remark 5.26. If we allow negative costs in the Bilevel Minimum Spanning Tree Problem, the proof of Theorem 5.23 works in the same way if we define $d(e) := -c(e)$ for all $e \in \mathcal{E}_l \cup \mathcal{E}_f$ instead. This shows that the special case of the Bilevel Minimum Spanning Tree Problem in which the follower is adversarial to the leader, having the opposite objective function, is hard as well. This is in contrast to [SZP19] where this special case (called MMST there) is shown to be polynomial-time solvable, for both sum and bottleneck objective. However, this is not a contradiction because the authors of [SZP19] only work with instances from $\mathcal{I}_{\mathcal{E}_f \text{ all}}$. In fact, Corollary 5.25 does not carry over to the special case of MMST since the property of opposite objective functions is lost in the construction in the proof of Lemma 5.19.

Together with Corollary 5.18, we can conclude that the Bilevel Minimum Spanning Tree Problem remains hard even if the follower controls a matching and hence a very simple combinatorial structure.

Corollary 5.27. *The Bilevel Minimum Spanning Tree Problem cannot be approximated to within a factor of $^{96}/_{95}$ in polynomial time, unless $\mathcal{P} = \mathcal{NP}$, even if $\mathcal{E}_f$ is a matching.*

From Corollary 5.22, we conclude that the hardness of the Bilevel Minimum Spanning Tree Problem is preserved even in the case of uniform leader's costs on her own edges.

We emphasize that Theorem 5.23 still holds for polynomially-bounded integer leader's costs. Indeed, $\mathcal{NP}$-hardness (and even $\mathcal{MAXSNP}$- and therefore $\mathcal{APX}$-hardness) of this case follows from the fact that the Steiner Forest Problem with edge weights 1 and 2 is proved to be $\mathcal{MAXSNP}$-hard in [BP89]. Moreover, the result that the Steiner Forest Problem is $\mathcal{NP}$-hard to approximate within a factor of $^{96}/_{95}$ also holds in case of polynomially-bounded integer weights by [CC08, Remark 16].

Corollary 5.28. *The Bilevel Minimum Spanning Tree Problem cannot be approximated to within a factor of $^{96}/_{95}$ in polynomial time, unless $\mathcal{P} = \mathcal{NP}$, even if $\mathcal{E}_f$ is a tree and $c(e) = 1$ holds for all $e \in \mathcal{E}_l$.*

To conclude this section, we consider a related question which could be asked in any bilevel optimization problem: Can the leader enforce a given follower's response? More formally, we consider the following decision problem:

Response Problem

Instance: An instance $(G, \mathcal{E}_l, \mathcal{E}_f, d)$ of the Bilevel Minimum Spanning Tree Problem (omitting the leader's objective) and a set $\overline{Y} \subseteq \mathcal{E}_f$

Task: Decide whether there is some leader's solution $X \subseteq \mathcal{E}_l$ such that $\overline{Y}$ is the follower's response to X.

Note that the leader's objective function is not relevant for this problem because it only deals with the existence of a leader's solution but not with its costs.

For the Response Problem to be well-defined, as for the Bilevel Minimum Spanning Tree Problem itself, it is essential to assume that the follower has a consistent strategy to select a follower's response in case his optimum solution is not unique. As discussed in Remark 5.7, we ensure such a consistent strategy by assuming that the follower chooses edges greedily according to some deterministic order.

Apart from being an interesting structural question in its own right, we will see in Section 5.6 that the Response Problem – or more precisely, the optimization version in which the cheapest solution X enforcing $\overline{Y}$ is desired – is related to the fixed-parameter tractability of the Bilevel Minimum Spanning Tree Problem in terms of $|\mathcal{E}_f|$. However, we will prove that the Response Problem, even in the decision version, is $\mathcal{NP}$-complete. For this, we use the following problem:

Vertex-Disjoint Steiner Trees Problem

Instance: A connected graph G and $k \in \mathbb{N}$ disjoint terminal sets $S_1, \dots, S_k \subseteq V(G)$

Task: Decide whether there are vertex-disjoint trees $T_1, \dots, T_k$ in G such that $S_i \subseteq V(T_i)$ for all $i \in \{1, \dots, k\}$.

This problem is similar to the Steiner Forest Problem defined previously, but not the same. The important difference is that, in the Steiner Forest Problem, no disjointness of the trees in the solution is required, i.e., it is feasible to have several sets S_i lying in the same connected component of the solution. Moreover, we are considering the decision version of the Vertex-Disjoint Steiner Trees Problem here, without any edge costs. Such a decision version of the Steiner Forest Problem would not be interesting because it is always feasible to select a spanning tree.

The Vertex-Disjoint Steiner Trees Problem is known to be $\mathcal{NP}$-complete even for $k = 2$ in so-called two-layer routing graphs [KPS90]. We use this fact to prove the following result:

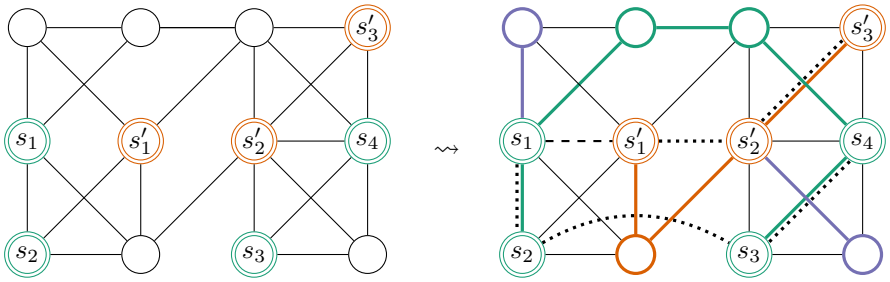


Figure 5.5. Illustration of the proof of Theorem 5.29. The two terminal sets in the instance of the Vertex-Disjoint Steiner Trees Problem are marked by green and orange vertices, respectively. Dotted lines represent follower's edges of cost 0, whereas the dashed line represents the follower's edge in $\overline{Y}$ having cost 1. One feasible solution to the Vertex-Disjoint Steiner Trees Problem is marked by green and orange edges; a corresponding leader's solution consists of the green, orange, and purple edges. The purple edges are added to the original solution of the Vertex-Disjoint Steiner Trees Problem in order to cover all vertices.

Theorem 5.29. *The Response Problem is $\mathcal{NP}$-complete, even if $|\overline{Y}| = 1$ and $\mathcal{E}_f$ forms a path on a subset of the vertex set.*

Proof. The Response Problem clearly belongs to $\mathcal{NP}$. To show completeness, we reduce the Vertex-Disjoint Steiner Trees Problem for $k = 2$ to the Response Problem. Given an instance of the Vertex-Disjoint Steiner Trees Problem consisting of a connected graph G and disjoint sets $S = \{s_1, \dots, s_r\} \subseteq V(G)$ and $S' = \{s'_1, \dots, s'_{r'}\} \subseteq V(G)$, we construct an instance of the Response Problem as follows: Define a graph G' by $V(G') \coloneqq V(G)$ and $E(G') \coloneqq \mathcal{E}_l \cup \mathcal{E}_f$, where $\mathcal{E}_l \coloneqq E(G)$ and

$$\mathcal{E}_f \coloneqq \{\{s_i, s_{i+1}\} \mid i \in \{1, \dots, r-1\}\} \cup \{\{s'_i, s'_{i+1}\} \mid i \in \{1, \dots, r'-1\}\} \cup \{\{s_1, s'_1\}\}.$$

Set $d(\{s_1, s'_1\}) \coloneqq 1$ and $d(e) \coloneqq 0$ for all $e \in \mathcal{E}_f \setminus \{\{s_1, s'_1\}\}$. Let $\overline{Y} \coloneqq \{\{s_1, s'_1\}\}$. An illustration of this construction is given in Figure 5.5. We now show that the answer to this instance of the Response Problem is "yes" if and only if the answer to the given instance of the Vertex-Disjoint Steiner Trees Problem is "yes".

Assume that T and T' are vertex-disjoint trees in G such that $S \subseteq V(T)$ and $S' \subseteq V(T')$. Since G is connected, we may assume that $E(T) \cup E(T')$ covers all vertices of G, by connecting all non-covered vertices to either T or T' arbitrarily; see the purple edges in Figure 5.5. We claim that the leader's choice $X \coloneqq E(T) \cup E(T')$ forces the follower to respond with $\overline{Y}$. Indeed, the follower's preferred edges e with $d(e) = 0$ would all produce cycles, while $\{s_1, s'_1\}$ needs to be added to turn X into a spanning tree.

Now assume that there exists a leader's solution X forcing the follower to respond with exactly the set $\overline{Y}$. Since the latter prefers edges from $\mathcal{E}_f \setminus \overline{Y}$, the leader must prevent him from adding any of those, i.e., all vertices in S are connected by X and the same is true for the vertices in S'. On the other hand, since the follower selects $\{s_1, s'_1\}$, the sets S and S' cannot be connected by X. Hence, X contains two vertex-disjoint trees covering S and S', respectively. □

Note that, similar to the proof of Theorem 5.23, there is some freedom in the construction of the follower's edge set $\mathcal{E}_f$ in this proof; see Remark 5.24. Instead of the paths given

by the edge sets $\{\{s_i, s_{i+1}\} \mid i \in \{1, \dots, r-1\}\}$ and $\{\{s'_i, s'_{i+1}\} \mid i \in \{1, \dots, r'-1\}\}$, one could choose any other graph structure spanning the vertices in S and S', respectively. Therefore, Theorem 5.29 does not only hold for sets $\mathcal{E}_f$ forming a path, but also for many other topologies.

Using the same construction as in Lemma 5.19, one can show that the result of Theorem 5.29 holds for instances in $\mathcal{I}_{\mathcal{E}_f \text{ all}}$ as well. Moreover, since the leader's costs are not relevant in the Response Problem, Theorem 5.29 trivially remains true for any specific choice of leader's costs, in particular in the case of uniform leader's costs.

5.6 Fixed-Parameter Tractability

In view of the hardness results presented in Section 5.5, the question arises how the Bilevel Minimum Spanning Tree Problem can still be tackled. When one cannot hope for polynomial-time exact algorithms, two obvious directions to consider are approximation and parameterized complexity; see also Sections 2.1.2 and 2.1.3. Some results of these types will be shown for the Bilevel Minimum Spanning Tree Problem in this and the next section.

We first focus on parameterized complexity. It turns out that the number of leader's edges is a parameter that easily leads to a fixed-parameter tractability result (see Theorem 5.30), while the situation is much more complex in terms of the number of follower's edges. The question whether the Bilevel Minimum Spanning Tree Problem is fixed-parameter tractable in the number of follower's edges remains open, but we will prove several related results. In particular, the Bilevel Minimum Spanning Tree Problem with uniform leader's costs on her own edges is fixed-parameter tractable; see Corollary 5.32. We also show that 2-approximating the Bilevel Minimum Spanning Tree Problem is fixed-parameter tractable; see Theorem 5.36. On the other hand, a reduction from the Shortest Vertex-Disjoint Steiner Trees Problem in Theorem 5.33 indicates that finding a polynomial-time algorithm for the Bilevel Minimum Spanning Tree Problem with a very small fixed number of follower's edges would already be a very strong result.

It is easy to see that the Bilevel Minimum Spanning Tree Problem is tractable when the number of edges controlled by the leader is bounded. In fact, we have

Theorem 5.30. *The Bilevel Minimum Spanning Tree Problem is fixed-parameter tractable in the number of edges controlled by the leader.*

Proof. If $k = |\mathcal{E}_l|$, the leader can choose between at most 2^k different solutions. We enumerate all subsets of $\mathcal{E}_l$ and return the best resulting solution. Checking if a given subset of $\mathcal{E}_l$ is a feasible leader's solution and, if this is the case, computing the follower's response and the corresponding objective function value is all possible in polynomial time by Lemma 5.6. □

Note that this result and the described algorithm are closely related to the tractability of the follower's problem implying that the leader's problem is $\mathcal{NP}$-easy; see Corollary 5.8. Moreover, recall that a very similar result was shown in Theorem 3.57 for the Robust Bilevel Selection Problem under Assumptions (sub) and (bb).

We now turn to the much more involved question whether the Bilevel Minimum Spanning Tree Problem is fixed-parameter tractable in the number of edges controlled

by the follower. We start by considering the Response Problem introduced in Section 5.5. In the proof of Theorem 5.29, a connection between the Response Problem and the Vertex-Disjoint Steiner Trees Problem was established in order to prove $\mathcal{NP}$-completeness. It turns out that this relation is also useful for translating positive results from the Vertex-Disjoint Steiner Trees Problem to the Response Problem. More precisely, the fact that the Vertex-Disjoint Steiner Trees Problem is fixed-parameter tractable in the total number $\sum_{i=1}^{k}|S_i|$ of terminals [RS90; RS95] can be used to prove the fixed-parameter tractability of the Response Problem in terms of $|\mathcal{E}_f|$.

Theorem 5.31. *The Response Problem is fixed-parameter tractable in the number of edges controlled by the follower.*

Proof. Consider an instance $(G, \mathcal{E}_l, \mathcal{E}_f, d, \overline{Y})$ of the Response Problem. Let $V_f \subseteq V(G)$ be the set of all endpoints of edges in $\mathcal{E}_f$. The algorithm proceeds as follows: All partitions of V_f into nonempty subsets are enumerated. For a given partition $S_1, \dots, S_k$ with $k \in \mathbb{N}$, the Vertex-Disjoint Steiner Trees Problem on $(V(G), \mathcal{E}_l)$ is solved by the algorithm given in [RS95]. Note that the graph $(V(G), \mathcal{E}_l)$ does not have to be connected, but the definition of the Vertex-Disjoint Steiner Trees Problem and the algorithm can be used anyway. If the result is negative, the partition is discarded. Otherwise, let $T_1, \dots, T_k$ be a corresponding solution of the Vertex-Disjoint Steiner Trees Problem and extend the trees T_i such that $X \coloneqq \bigcup_{i=1}^{k} E(T_i)$ covers all vertices, while the T_i must remain vertex-disjoint. This is possible since we assume that G is connected. Next, compute the follower's response Y to X. If it agrees with $\overline{Y}$, stop and return "yes" and, if desired, the set X. If the end of the enumeration is reached, return "no".

The correctness of the algorithm immediately follows from the fact that the follower's response only depends on whether two vertices in V_f are connected by the leader or not, and all possible situations are enumerated. For the running time, note that the number and size of the enumerated partitions only depend on $|V_f| \leq 2|\mathcal{E}_f|$, but not on the size of the overall graph. □

The algorithm proposed in the proof of Theorem 5.31 can actually be used for enumerating all possible follower's responses, along with one inducing leader's solution for each response. Unfortunately, Theorem 5.31 does not imply that the problem of computing the *best* leader's solution enforcing a given response $\overline{Y}$ is fixed-parameter tractable; see the discussion below. Therefore, we cannot derive that the Bilevel Minimum Spanning Tree Problem itself is fixed-parameter tractable in the number of edges controlled by the follower. However, all leader's solutions enforcing a given follower's response consist of the same number of edges because every spanning tree in the overall graph has the same number of edges. Hence, if the leader has uniform costs on the edges in $\mathcal{E}_l$, this algorithm can be used to solve the Bilevel Minimum Spanning Tree Problem, leading to the following result:

Corollary 5.32. *The Bilevel Minimum Spanning Tree Problem with $c(e) = \gamma$ for all $e \in \mathcal{E}_l$, for some constant $\gamma \in \mathbb{Q}_{\geq 0}$, is fixed-parameter tractable in the number of edges controlled by the follower.*

However, different costs on the edges in $\mathcal{E}_l$ cannot be handled easily. In particular, we cannot use the reduction in Lemma 5.21 to make the costs uniform, since it increases the size of $\mathcal{E}_f$ by $\sum_{e \in \mathcal{E}_l}(c(e) - 1)$. The result only carries over to instances where the

latter sum is bounded by some function in the original number of follower's edges. Unfortunately, we are not able to answer the question whether Corollary 5.32 also holds for arbitrary weights, but we conjecture that this is not the case. In fact, there is some evidence that the Bilevel Minimum Spanning Tree Problem is not easy to solve even for a *fixed* number of edges controlled by the follower. To justify this conjecture, we will establish a relation between the Bilevel Minimum Spanning Tree Problem and the optimization version of the Vertex-Disjoint Steiner Trees Problem:

Shortest Vertex-Disjoint Steiner Trees Problem
Instance: A connected graph G, edge weights $w\colon E(G) \to \mathbb{Q}_{\geq 0}$, and $k \in \mathbb{N}$ disjoint terminal sets $S_1, \dots, S_k \subseteq V(G)$
Task: Find vertex-disjoint trees $T_1, \dots, T_k$ in G such that $S_i \subseteq V(T_i)$ for all $i \in \{1, \dots, k\}$, and such that $\sum_{i=1}^{k} w(E(T_i))$ is minimum, or decide that such trees do not exist.

Given that already the decision version, the Vertex-Disjoint Steiner Trees Problem, is a very difficult problem, it can be expected that the Shortest Vertex-Disjoint Steiner Trees Problem is very hard as well. In fact, even for the special case in which each set S_i consists of only two vertices, which is called the Shortest Vertex-Disjoint Paths Problem, there are a lot of open complexity questions. Considerable research has been devoted to the Shortest Vertex-Disjoint Paths Problem for $k = 2$ and uniform edge weights w. Recently, a randomized polynomial-time algorithm for this case has been developed [BH19a]. To the best of our knowledge, no deterministic polynomial-time algorithm for $k = 2$ nor the complexity of the Shortest Vertex-Disjoint Paths Problem for any fixed $k \geq 3$ is known, even in case of uniform weights w. According to our next result, presenting an efficient algorithm for the Bilevel Minimum Spanning Tree Problem with a fixed number $|\mathcal{E}_f| = 2k$ of edges controlled by the follower would settle these open questions for k, and even related questions about the more general Shortest Vertex-Disjoint Steiner Trees Problem. In particular, an efficient algorithm for the Bilevel Minimum Spanning Tree Problem with $|\mathcal{E}_f| = 4$ would lead to an efficient algorithm for the Shortest Vertex-Disjoint Paths Problem with $k = 2$.

Theorem 5.33. *The Shortest Vertex-Disjoint Steiner Trees Problem with a fixed number $K := \sum_{i=1}^{k} |S_i|$ of terminals can be polynomially reduced to the Bilevel Minimum Spanning Tree Problem with K edges controlled by the follower.*

Proof. Given an instance $(G, w, S_1, \dots, S_k)$ of the Shortest Vertex-Disjoint Steiner Trees Problem with $K = \sum_{i=1}^{k} |S_i|$ terminals, we construct an instance of the Bilevel Minimum Spanning Tree Problem as follows. We extend G by one vertex s_0, i.e., we set $G' := (V', \mathcal{E}_l \cup \mathcal{E}_f)$ with $V' := V(G) \cup \{s_0\}$. The edges controlled by the leader are given by $\mathcal{E}_l := E(G) \cup \mathcal{E}_l^0$, where

$$\mathcal{E}_l^0 := \Big\{ \{s_0, v\} \mid v \in V(G) \setminus \bigcup_{i=1}^{k} S_i \Big\}.$$

For the follower's edges, we introduce an arbitrary spanning tree on each vertex set S_i and call the set of these edges $\mathcal{E}_f^0$. Moreover, for each $i \in \{1, \dots, k\}$, we select a vertex $s_i \in S_i$ arbitrarily and introduce a follower's edge $\{s_{i-1}, s_i\}$. Together with $\mathcal{E}_f^0$, these edges form

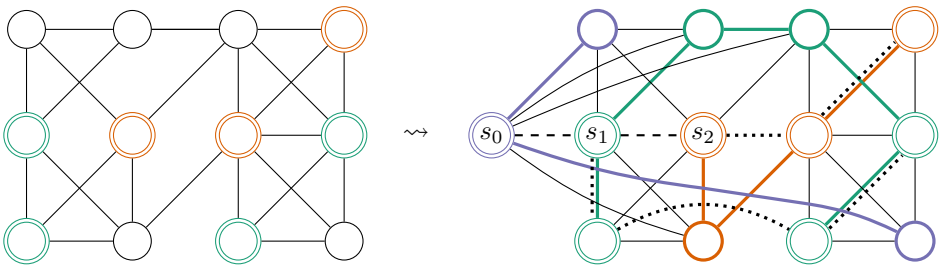


Figure 5.6. Illustration of the proof of Theorem 5.33 for $k = 2$ and $K = 7$. The two terminal sets S_1 and S_2 in the instance of the Shortest Vertex-Disjoint Steiner Trees Problem are marked by green and orange vertices, respectively. Dotted lines represent follower's edges of cost 0, whereas the dashed lines represent the follower's edges having follower's cost 1. An optimum leader's solution is given by the colored edges, where green and orange edges correspond to an optimum solution of the original instance of the Shortest Vertex-Disjoint Steiner Trees Problem and purple edges connect all vertices not covered by the latter with the auxiliary vertex s_0.

the set $\mathcal{E}_f$. The cost function for the leader is defined as

$$c(e) := \begin{cases} w(e) + M, & \text{if } e \in E(G), \\ M, & \text{if } e \in \mathcal{E}_l^0, \\ M|V(G)|, & \text{if } e \in \mathcal{E}_f^0, \\ 0, & \text{if } e \in \mathcal{E}_f \setminus \mathcal{E}_f^0, \end{cases}$$

where $M := \sum_{e \in E(G)} w(e) + 1$. The cost function for the follower is given by

$$d(e) := \begin{cases} 0, & \text{if } e \in \mathcal{E}_f^0, \\ 1, & \text{if } e \in \mathcal{E}_f \setminus \mathcal{E}_f^0. \end{cases}$$

Clearly, this construction is polynomial, with $|\mathcal{E}_f| = k + \sum_{i=1}^{k} (|S_i| - 1) = K$; an illustration is given in Figure 5.6. We claim that the given instance of the Shortest Vertex-Disjoint Steiner Trees Problem is feasible if and only if the optimum value of the constructed instance of the Bilevel Minimum Spanning Tree Problem is smaller than $M(|V(G)| - k + 1)$, and that in this case the optimum values differ by exactly $M(|V(G)| - k)$.

So first assume that vertex-disjoint trees T_i with $S_i \subseteq V(T_i)$ for $i \in \{1, \dots, k\}$ exist. Then consider the leader's solution X consisting of all edges contained in any of the trees T_i and, for each vertex $v \in V(G)$ not belonging to any tree, the edge $\{s_0, v\}$. We have $|X| = |V(G)| - k$ because X forms a forest with $k + 1$ connected components on $|V(G)| + 1$ vertices. The follower's response to X is $Y := \{\{s_{i-1}, s_i\} \mid i \in \{1, \dots, k\}\}$ with $c(Y) = 0$. In summary, the objective value of X is

$$c(X) + c(Y) = \sum_{i=1}^{k} w(E(T_i)) + M(|V(G)| - k) + 0 < M(|V(G)| - k + 1).$$

For the other direction, consider any feasible leader's choice X in the constructed instance of the Bilevel Minimum Spanning Tree Problem and assume that it has

an objective value less than $M(|V(G)| - k + 1)$. Then, for all $i \in \{1, \dots, k\}$, all vertices in S_i must be connected in X, as otherwise the follower would select an edge with leader's cost $M|V(G)| \geq M(|V| - k + 1)$. Moreover, since each leader's edge costs at least M and the final tree must have $|V(G)|$ edges, the only way to achieve costs less than $M(|V(G)| - k + 1)$ is to take exactly $|V(G)| - k$ edges and make the follower select all edges $\{s_{i-1}, s_i\}$ for $i \in \{1, \dots, k\}$. It follows that X has $k + 1$ components containing exactly one of the vertices $s_0, \dots, s_k$ each. Thus, X contains disjoint trees T_i with $S_i \subseteq V(T_i)$, and their total cost is

$$\sum_{i=1}^{k} w(E(T_i)) = \sum_{e \in X} (c(e) - M) = \sum_{e \in X} c(e) - M(|V(G)| - k).$$

This concludes the proof. □

As in Theorems 5.23 and 5.29, also other topologies of the follower's edges are possible; see also Remark 5.24. We emphasize that Theorem 5.33 gives a second proof for the $\mathcal{NP}$-hardness of the Bilevel Minimum Spanning Tree Problem if we do not bound the number of edges controlled by the follower. In particular, it shows that the Bilevel Minimum Spanning Tree Problem is at least as hard as the Shortest Vertex-Disjoint Steiner Trees Problem. However, the reduction used in the proof of Theorem 5.33 is not approximation-preserving, so that the negative result of Theorem 5.23 concerning approximability does not follow from Theorem 5.33.

Remark 5.34. As argued above, Theorem 5.33 shows that finding a polynomial-time algorithm for the Bilevel Minimum Spanning Tree Problem with only $|\mathcal{E}_f| = 4$ is already a very interesting and challenging task. This raises the same question also for the special cases of the Bilevel Minimum Spanning Tree Problem with $|\mathcal{E}_f| \leq 3$, although these cases do not have any interesting implications in terms of Theorem 5.33.

The case $|\mathcal{E}_f| = 0$ is trivial, as it amounts to solving a usual Minimum Spanning Tree Problem for the leader. Also the case $|\mathcal{E}_f| = 1$ is easy to solve; we can distinguish two cases: Either the leader selects a whole spanning tree herself or she contracts the follower's edge and solves a Minimum Spanning Tree Problem in the resulting graph, which forces the follower to select his single edge in his response. The better of the two solutions is optimum for the leader.

However, already for the cases $|\mathcal{E}_f| = 2$ and $|\mathcal{E}_f| = 3$, we cannot answer the question whether the Bilevel Minimum Spanning Tree Problem is solvable in polynomial time. While there are still only very few possible follower's solutions that one can easily enumerate, the task to enforce a given follower's solution, as in the Response Problem, but now including the optimization of the leader's objective, quickly becomes more challenging. In particular, for $|\mathcal{E}_f| = 2$, three of the four cases corresponding to fixed follower's solutions are easy to solve, similarly to the situation for $|\mathcal{E}_f| = 1$ above. However, the fourth case seems to be more involved. Here, the desired follower's solution is $\overline{Y} = \{e_2\}$, where $\mathcal{E}_f = \{e_1, e_2\}$ with e_1, e_2 being the follower's order of preference. The leader's task now is to find a forest $X \subseteq \mathcal{E}_l$ on $V(G)$ with two connected components such that the endpoints of e_1 both lie in the same component, but the endpoints of e_2 lie in different components, and such that the leader's cost $c(X)$ is minimum. Solving this problem as well as similar ones that arise from fixed follower's solutions in case of small values $|\mathcal{E}_f| \geq 3$ remain very interesting open questions.

While Theorem 5.33 makes it unlikely that the Bilevel Minimum Spanning Tree Problem is fixed-parameter tractable in the number of follower's edges, we will show next that at least approximating the Bilevel Minimum Spanning Tree Problem within a factor of 2 is fixed-parameter tractable in the same parameter. As a first step, we show that a similar result holds for a variant of the Steiner Forest Problem defined as follows:

Spanning Steiner Forest Problem

Instance: A connected graph G, edge weights $w\colon E(G) \to \mathbb{Q}_{\geq 0}$, and $k \in \mathbb{N}$ disjoint terminal sets $S_1, \ldots, S_k \subseteq V(G)$

Task: Find a forest F in G such that, for each $i \in \{1, \ldots, k\}$, all vertices in S_i are in the same connected component of F, every vertex in $V(G) \setminus \bigcup_{i=1}^{k} S_i$ is in the same connected component as one of the sets S_i, and such that $c(E(F))$ is minimum.

The difference from the usual Steiner Forest Problem is that, in addition to connecting each terminal set S_i, each non-terminals needs to be connected to one of the terminal sets.

Theorem 5.35. *The problem of approximating the Spanning Steiner Forest Problem within a factor of 2 is fixed-parameter tractable in the total number of terminals.*

Proof. We use the fact that the Steiner Forest Problem is fixed-parameter tractable, which can be seen as follows: For the classical Steiner Tree Problem in a graph G, an exact algorithm with running time $\mathcal{O}(3^{|S|}|V(G)|)$, where $S \subseteq V(G)$ is the set of terminals, is well-known [DW71]. This algorithm can be extended to the Steiner Forest Problem in the following way: Enumerate all partitions of the set $\{S_1, \ldots, S_k\}$ of terminal sets into nonempty subsets. Every such partition, by building the union of the terminal sets in each of the subsets, results in a coarser partition $S'_1, \ldots, S'_r$ of the set $\bigcup_{i=1}^{k} S_i$ of all terminals. Now solve the Steiner Tree Problem for each terminal set S'_i and merge the resulting r edge sets in order to obtain a feasible solution of the Steiner Forest Problem. Obviously, the best solution obtained in this way is optimum.

Now we compute a solution to the Spanning Steiner Forest Problem in the following way: First, compute an optimum solution F of the corresponding Steiner Forest Problem, for example using the algorithm described above. Second, contract the set $\bigcup_{i=1}^{k} S_i$ of all terminals, together with all non-terminals that are connected to a terminal by edges in $E(F)$, into a single new vertex. Now compute a minimum spanning tree T in the resulting graph and return the forest $(V(G), E(F) \cup E(T))$ as a solution to the given instance of the Spanning Steiner Forest Problem.

Clearly, the solution is feasible for the Spanning Steiner Forest Problem and the running time is the same as the one of the applied algorithm for the Steiner Forest Problem because the running time for solving the Minimum Spanning Tree Problem is negligible. It remains to show that it is a 2-approximation. For this, observe that both $E(F)$ and $E(T)$ have at most the cost of an optimum solution to the Spanning Steiner Forest Problem, since every such solution must contain a Steiner forest having at least the cost of $E(F)$, as well as a spanning tree in the graph in which T is a minimum spanning tree. □

Theorem 5.35 now allows us to show the desired result about the Bilevel Minimum Spanning Tree Problem:

Theorem 5.36. *The problem of approximating the* Bilevel Minimum Spanning Tree Problem *within a factor of* 2 *is fixed-parameter tractable in the number of edges controlled by the follower.*

Proof. Consider an instance $(G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ of the Bilevel Minimum Spanning Tree Problem. We may assume that $(V(G), \mathcal{E}_l)$ is connected, since the construction according to Lemma 5.13 does not increase the number of follower's edges. The idea of the algorithm is to enumerate all possible follower's solutions and apply Theorem 5.35 to each of them. More precisely, as in the proof of Theorem 5.31, we consider the set $V_f \subseteq V(G)$ of vertices incident to a follower's edge and enumerate all possible partitions of V_f into nonempty disjoint sets. For a fixed partition $S_1, \dots, S_k$, we solve the Spanning Steiner Forest Problem on the leader's graph $(V(G), \mathcal{E}_l)$ with edge weights defined by the leader's cost function c, using the algorithm given in Theorem 5.35, and obtain some forest $X \subseteq \mathcal{E}_l$. Next, we compute the follower's response to X and, if there is a feasible response Y, store it together with X as a candidate for our final solution. Finally, we return the candidate solution minimizing the total weight $c(X) + c(Y)$.

Clearly, the running time of this algorithm is as desired. Moreover, it computes a feasible solution to the Bilevel Minimum Spanning Tree Problem if there is one. It remains to prove that in this case the algorithm always computes a 2-approximate solution. For this, let X^* be an optimum leader's solution of the given instance of the Bilevel Minimum Spanning Tree Problem, together with the follower's response Y^*, and let $S_1^*, \dots, S_k^*$ be the partition of V_f corresponding to the connected components of $(V(G), X^*)$. Then X^* is a (not necessarily optimum) solution for the Spanning Steiner Forest Problem corresponding to this partition. Let X be the solution for the Spanning Steiner Forest Problem computed by the algorithm presented above when considering this partition. Since we use a 2-approximation algorithm for the Spanning Steiner Forest Problem, we have $c(X) \leq 2c(X^*)$. Moreover, the partition of V_f induced by X is either $S_1^*, \dots, S_k^*$ or a coarser one, which implies that the follower's response Y to X is a subset of Y^* by Lemma 5.14, based on the deterministic behavior of the follower. Altogether, we now obtain

$$c(X) + c(Y) \leq 2c(X^*) + c(Y^*) \leq 2(c(X^*) + c(Y^*)),$$

which shows the desired result. □

5.7 An Approximation Algorithm

In the previous section, we have investigated questions regarding the fixed-parameter tractability of the Bilevel Minimum Spanning Tree Problem and showed in Theorem 5.36 that the problem can be 2-approximated in polynomial time if the number of edges controlled by the follower is a fixed constant. Now the question arises whether there are also approximation algorithms whose running time is polynomial, independently of a fixed parameter. In this section, we present a polynomial-time $(|V(G)| - 1)$-approximation algorithm for the Bilevel Minimum Spanning Tree Problem.

Theorem 5.37. *The* Bilevel Minimum Spanning Tree Problem *admits a polynomial-time* $(|V(G)| - 1)$*-approximation algorithm.*

Proof. Let $(G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ be an instance of the Bilevel Minimum Spanning Tree Problem. The algorithm starts with an empty leader's solution $X := \emptyset$ and iteratively

adds leader's edges to X. At the same time, the graph G, initially given as part of the considered instance, is modified in each iteration of the algorithm. More specifically, in each iteration, we first apply Corollary 5.15 in order to turn $\mathcal{E}_f$ into a forest. Then, in the current graph G (with $E(G) = \mathcal{E}_l \cup \mathcal{E}_f$), we compute a minimum spanning tree T according to the leader's cost function c. Let $E_l^T := E(T) \cap \mathcal{E}_l$ be the set of edges in the spanning tree that are controlled by the leader, and add the edges in E_l^T to X. If $E_l^T = E(T)$ or $E_l^T = \emptyset$, we stop and output X as the leader's solution. Otherwise, we contract the edges in E_l^T and start the next iteration.

The algorithm clearly runs in polynomial time, since we perform at most $|V(G)| - 1$ iterations, and in each iteration we apply the polynomial reduction of Corollary 5.15 and solve a Minimum Spanning Tree Problem. It is also not hard to see that the algorithm computes a feasible solution: If it stops with $E_l^T = E(T)$, the leader's solution X already forms a spanning tree in the original graph. Otherwise, it stops with $E_l^T = \emptyset$. In this case, the follower is able to complete X to a spanning tree, for example, by using the edges in $E(T)$. It remains to show that the objective value of X is at most $|V(G)|-1$ times the optimum value.

We prove this by induction on the number $|V(G)|$ of vertices. If $|V(G)| = 2$, the statement is clearly true since we may assume that we only have two edges, one leader's and one follower's edge. So let us assume that, for some arbitrary but fixed $n \in \mathbb{N}$, the statement is true for all graphs that have at most n vertices. Let $I = (G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ be an instance, where $(V(G), \mathcal{E}_f)$ is a forest and $|V(G)| = n + 1$. Let T be the spanning tree that is computed in the first iteration of the algorithm. If it stops after the first iteration, i.e., if $E_l^T = E(T)$ or $E_l^T = \emptyset$, either the leader or the follower selects the whole tree $E(T)$, while the other player selects $\emptyset$; note that, for $E_l^T = \emptyset$, the follower can only select $E(T)$ as response, as $(V(G), \mathcal{E}_f)$ is a forest. In both cases, the leader's objective value is $c(E(T))$, which is clearly optimum. Otherwise, we have $c(E_l^T) \leq c(E(T)) \leq \mathrm{OPT}(I)$, where $\mathrm{OPT}(I)$ denotes the value of an optimum solution to instance I. Let $\hat{I}$ with the graph $\hat{G}$ be the instance that is considered in the second iteration, i.e., after contracting the edges in E_l^T. Observe that, by Lemma 5.14, we have $\mathrm{OPT}(\hat{I}) \leq \mathrm{OPT}(I)$, since $\hat{I}$ arises from I by contracting certain edges of the graph. Furthermore, a solution $\hat{X}$ to $\hat{I}$ with follower's response $\hat{Y}$ can be augmented to a solution to I by simply adding E_l^T, such that $\hat{Y}$ remains the follower's response. Finally, observe that $|V(\hat{G})| \leq n$ and hence the induction hypothesis holds, i.e., the solution $\hat{X}$ to $\hat{I}$ produced by the algorithm is a $(|V(\hat{G})| - 1)$-approximation, where $|V(\hat{G})| - 1 \leq |V(G)| - 2$. Putting things together, we derive that

$$\begin{aligned} c(X) + c(\hat{Y}) &= c(E_l^T) + c(\hat{X}) + c(\hat{Y}) \\ &\leq \mathrm{OPT}(I) + (|V(\hat{G})| - 1)\,\mathrm{OPT}(\hat{I}) \\ &\leq (|V(G)| - 1)\,\mathrm{OPT}(I) \end{aligned}$$

holds for the objective value of the leader's solution $X = E_l^T \cup \hat{X}$ returned by the algorithm. □

Our results regarding the approximability of the Bilevel Minimum Spanning Tree Problem still leave a huge gap: On the one hand, we have seen that it is $\mathcal{NP}$-hard to approximate the problem within a factor of $96/95$ (Theorem 5.23), and on the other hand, we provided a $(|V(G)| - 1)$-approximation algorithm (Theorem 5.37). It is a task for future research to make this gap smaller and, e.g., investigate whether a polynomial-time algorithm with a constant approximation factor is achievable.

5.8 Problem Variants with Bottleneck Objectives

As already mentioned in Remark 5.11 and Section 5.2.3, the two decision makers in the Bilevel Minimum Spanning Tree Problem may also optimize bottleneck instead of sum objective functions; they then pay only for the most expensive edge in their solution instead of the sum of all edge costs. Since we can choose between the two types of objectives for leader and follower independently, this results in four problem variants in total, one of which is the case of two sum objectives investigated in the previous sections. This section will deal with the other three problem variants, in which one or both of the two decision makers have a bottleneck objective function.

In Section 5.8.1, we discuss the definitions and basic properties of the bottleneck variants of the Bilevel Minimum Spanning Tree Problem and give an overview of previously known as well as our new results about them. Algorithms for the cases that are solvable in polynomial time will then be presented in Section 5.8.2, while hardness results for the remaining cases are shown in Section 5.8.3.

5.8.1 Introduction of the Bottleneck Problem Variants

If the leader has a bottleneck objective, in contrast to the problem definition in Section 5.2.1, her objective is to minimize $\max_{e \in X \cup Y'} c(e)$ instead of $c(X \cup Y') = \sum_{e \in X \cup Y'} c(e)$. Recall that, when the follower has a bottleneck objective, we again have to distinguish two variants of this objective, namely minimizing either $\max_{e \in Y'} d(e)$ or $\max_{e \in X \cup Y'} d(e)$, i.e., the follower either takes only his own edges into account or both the edges selected by the leader and by himself. As already explained in Remark 5.11, these variants are not equivalent, in contrast to the corresponding variants in the sum objective case. The problem version in which the follower considers only his own edges can be seen as a special case of the one in which he considers all edges by setting $d(e) \coloneqq 0$ for all $e \in \mathcal{E}_l$.

While the follower's costs of leader's edges were not relevant in case both decision makers have a sum objective (and even not provided by an instance in the original definition of the Bilevel Minimum Spanning Tree Problem in Section 5.2.1), this is different now if the follower's objective is to minimize $\max_{e \in X \cup Y'} d(e)$. Hence, from now on, we assume the follower's weight function $d\colon E(G) \to \mathbb{Q}_{\geq 0}$ to be defined for all edges, as it is the case for the leader's weight function c. Note also that, in the construction of Remark 5.10, we have to require the parallel edges to have not only the same leader's but also the same follower's cost now. The same applies to the definition of $\mathcal{I}_{\mathcal{E}_f \text{ all}}$ (see Section 5.3), in order to achieve a setting that is equivalent to the one in [SZP19].

Moreover, the distinction between the optimistic and the pessimistic setting is more important now than in the previous sections, and we will even see a significant difference in the computational complexity in one case. In order to understand the difference to the previous setting, we first have to clarify how the follower solves his problem in case he has a bottleneck objective. Recall that the follower's algorithm in case of a sum objective consists of contracting all edges of the given leader's solution X and solving a Minimum Spanning Tree Problem in the resulting graph, restricted to the edges in $\mathcal{E}_f$; see Lemma 5.6. Instead of a Minimum Spanning Tree Problem, the follower now has to solve a bottleneck variant of the Minimum Spanning Tree Problem, i.e., find a spanning tree that minimizes the weight of its most expensive edge. It can be easily seen that every optimum solution of the Minimum Spanning Tree Problem is also optimum for the

corresponding bottleneck variant, while the other direction is not true in general; see, e.g., [Gas02, Chapter 7]. Hence, algorithms for the Minimum Spanning Tree Problem, such as Kruskal's algorithm, can also be applied to the bottleneck variant. However, we have to be careful in the bilevel setting since the follower also has to respect the leader's objective, according to the optimistic or pessimistic setting, in case his optimum solution is not unique. For sum objectives, we always assumed that the follower acts greedily, using a fixed order of his edges that reflects the desired setting; see Remark 5.7. This assumption cannot be made in the bottleneck case, as we will see in Example 5.38 below. The intuitive reason for this is that the secondary criterion can come into play more frequently now, as there can be more follower's solutions that have the same objective value for himself.

Example 5.38. Consider the example depicted in Figure 5.1 and assume that the leader still has a sum objective, but the follower has a bottleneck objective (either $\max_{e\in Y'} d(e)$ or $\max_{e\in X\cup Y'} d(e)$). In his response to the leader's solution shown in Figure 5.1, the follower could now also select the edge $\{v_3, v_5\}$ instead of the edge $\{v_3, v_6\}$. Both options are optimum from the follower's perspective here because his objective value is 2 anyway. Under the optimistic assumption, the follower would still select $\{v_3, v_6\}$ because it is better for the leader. But under the pessimistic assumption, the follower would select $\{v_3, v_5\}$ instead, increasing the leader's objective value by 4 compared to the optimistic case and the case in which the follower has a sum objective.

A slightly modified version of the example also shows that we cannot assume the follower to work with a greedy algorithm and a fixed order of preference in case he has a bottleneck objective: Consider the example in Figure 5.1 again, assume the follower to minimize $\max_{e\in Y'} d(e)$, and suppose there is an additional leader's edge $\{v_1, v_4\}$. If the leader selects this new edge in addition to the depicted solution, then the follower's response will consist of the edge $\{v_3, v_6\}$ in both the optimistic and the pessimistic setting because this is cheaper than $\{v_3, v_5\}$ for the follower. However, if the leader chooses the solution drawn in Figure 5.1, without the new edge, the response in the pessimistic setting contains the edge $\{v_3, v_5\}$ instead, as described in the previous paragraph. This shows that there is no order of the two edges $\{v_3, v_5\}$ and $\{v_3, v_6\}$ that can always be applied, and also that Lemma 5.14 is not true if the follower has a bottleneck objective.

Although this implies that the follower cannot solve his problem directly with Kruskal's algorithm for a fixed order of his edges anymore, he can still solve it in polynomial time as follows.

Lemma 5.39. *Given an instance of any variant of the Bilevel Minimum Spanning Tree Problem in which one or both of the two decision makers have a bottleneck objective, and a set X of leader's edges, one can determine whether X is a feasible leader's solution and, if this is the case, compute an optimum solution of the follower's problem, all in polynomial time.*

Proof. Let $(G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ be an instance of the Bilevel Minimum Spanning Tree Problem and let $X \subseteq \mathcal{E}_l$. Clearly, feasibility of X as a leader's solution can be checked in a straightforward way as in Lemma 5.6. We now assume X to be a feasible leader's solution that is not a spanning tree in G yet.

If the leader has a bottleneck objective and the follower has a sum objective, the follower can clearly apply the same greedy algorithm as in Lemma 5.6. The leader's objective can be used as a secondary criterion in the same way as explained in Remark 5.7 as well.

We now turn to the case in which the follower has a bottleneck objective. As before, the follower contracts the edges in X in the graph G. Then he enumerates all possible values of his objective, which are (at most) the values in either $D \coloneqq \{d(e) \mid e \in \mathcal{E}_f\}$ or $D \coloneqq \{\delta\} \cup \{d(e) \mid e \in \mathcal{E}_f, d(e) > \delta\}$, where $\delta \coloneqq \max_{e \in X} d(e)$, depending on the type of his objective. For every potential bottleneck value $\delta' \in D$, he restricts himself to the edge set $E_{\delta'} \coloneqq \{e \in \mathcal{E}_f \mid d(e) \leq \delta'\}$ and, if the resulting graph is connected, solves a Minimum Spanning Tree Problem with respect to either the leader's costs or the negative of the leader's costs, depending on the optimistic or pessimistic setting. Note that this is independent of whether the leader has a sum or a bottleneck objective. The smallest bottleneck value δ' that leads to a connected graph and hence to a feasible solution then gives an optimum follower's solution. □

Again, we assume a deterministic behavior of the follower when resolving any non-uniqueness. Also note that the follower's solution again depends only on the connected components of the graph $(V(G), X)$, and possibly also on the value $\delta \coloneqq \max_{e \in X} d(e)$ now, but not on the exact structure of the leader's solution X.

As in Corollary 5.8, the tractability of the follower's problem (Lemma 5.39) implies that the leader's problem is $\mathcal{NP}$-easy, also in case of bottleneck objectives.

Corollary 5.40. *All variants of the Bilevel Minimum Spanning Tree Problem in which one or both of the two decision makers have a bottleneck objective are $\mathcal{NP}$-easy.*

Shi, Zeng, and Prokopyev showed in [SZP19] that the Bilevel Minimum Spanning Tree Problem is tractable as soon as the leader or the follower (or both) optimize a bottleneck objective. However, the general assumption in [SZP19] is that the follower's and the leader's edge sets are not disjoint, but that the follower controls all edges, or, equivalently in our notation, that instances belong to $\mathcal{I}_{\mathcal{E}_f \text{ all}}$; see Sections 5.2.3 and 5.3. Without this assumption, the tractability results do not hold anymore in general. In fact, we will see that most cases are $\mathcal{NP}$-hard then. In [Gas02], Gassner developed two polynomial-time algorithms without the assumption that the follower controls all edges, namely for the cases in which the leader has a bottleneck objective and the follower either has a sum objective or a bottleneck objective, restricting to the pessimistic problem version in the latter case. In this case, however, she always assumes the follower to minimize $\max_{e \in Y'} d(e)$. We generalize this result to the case of the follower's objective being $\max_{e \in X \cup Y'} d(e)$ and slightly simplify her other algorithm. Moreover, our hardness results show that these are the only two cases which are polynomial-time solvable in general, unless $\mathcal{P} = \mathcal{NP}$. An overview of the different cases and results is given in Table 5.1.

5.8.2 Tractability Results

We now present polynomial-time algorithms for the tractable cases, namely the problem variant with the leader having a bottleneck objective and the follower having a sum objective (Theorem 5.41), and the pessimistic variant with both decision makers having a bottleneck objective (Theorem 5.42).

Theorem 5.41. *The variant of the Bilevel Minimum Spanning Tree Problem in which the leader has a bottleneck objective and the follower has a sum objective can be solved in polynomial time.*

Leader	Follower	Assumption	Results
sum	sum	opt/pess	$\mathcal{NP}$-hard (even $^{96}/_{95}$-approx.; Theorem 5.23), $(\lvert V(G)\rvert - 1)$-approx. (Theorem 5.37)
sum	bottleneck	pess	$\mathcal{P}$ for $\mathcal{I}_{\mathcal{E}_f\,\text{all}}$ and $\max_{e\in X\cup Y'} d(e)$ [SZP19], $\mathcal{NP}$-hard (even $^{96}/_{95}$-approx.; Theorem 5.43)
sum	bottleneck	opt	$\mathcal{P}$ for $\mathcal{I}_{\mathcal{E}_f\,\text{all}}$ and $\max_{e\in X\cup Y'} d(e)$ [SZP19], $\mathcal{NP}$-hard (Theorem 5.44)
bottleneck	sum	opt/pess	$\mathcal{P}$ for $\mathcal{I}_{\mathcal{E}_f\,\text{all}}$ [SZP19], $\mathcal{P}$ ([Gas02] and Theorem 5.41)
bottleneck	bottleneck	pess	$\mathcal{P}$ for $\mathcal{I}_{\mathcal{E}_f\,\text{all}}$ and $\max_{e\in X\cup Y'} d(e)$ [SZP19], $\mathcal{P}$ for $\max_{e\in Y'} d(e)$ [Gas02], $\mathcal{P}$ (Theorem 5.42)
bottleneck	bottleneck	opt	$\mathcal{P}$ for $\mathcal{I}_{\mathcal{E}_f\,\text{all}}$ and $\max_{e\in X\cup Y'} d(e)$ [SZP19], $\mathcal{NP}$-hard (even any approx.; Theorem 5.45)

Table 5.1. Tractability, hardness, and approximation results for all variants of the Bilevel Minimum Spanning Tree Problem with sum or bottleneck objective functions, assuming an optimistic or pessimistic setting.

Proof. Let $(G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ be an instance of the Bilevel Minimum Spanning Tree Problem. We first present the algorithm: For each $\gamma \in C := \{c(e) \mid e \in \mathcal{E}_l\} \cup \{0\}$, the leader considers the set $E_\gamma := \{e \in \mathcal{E}_l \mid c(e) \leq \gamma\}$ and selects an arbitrary edge set $X_\gamma \subseteq E_\gamma$ that consists of a spanning tree in each connected component of $G_\gamma := (V(G), E_\gamma)$. Let Y_γ be the corresponding response of the follower and c_γ the resulting leader's objective value, where $c_\gamma := \infty$ in case X_γ is infeasible, i.e., the follower cannot extend X_γ to a spanning tree. Finally, she chooses $\gamma^* \in \operatorname{argmin}_{\gamma\in C} c_\gamma$ and returns X_{γ^*} as optimum solution.

The algorithm clearly runs in polynomial time, so it remains to show that X_{γ^*} is indeed an optimum solution. For this, it suffices to show that, for any γ, choosing a solution $X \subseteq E_\gamma$ with the same bottleneck cost cannot yield a smaller objective function value than c_γ. Since the objective function of the follower is a sum, Lemma 5.14 applies; thus, his response Y to X is a superset of Y_γ. Now the cost of $X \cup Y$ (in the leader's bottleneck objective) is at least the cost of $X_\gamma \cup Y_\gamma$. □

We now turn to the case in which both leader and follower have a bottleneck objective. Then, the above algorithm does not work in general because Lemma 5.14 is not true in case the follower has a bottleneck objective; see Example 5.38. However, Gassner showed in [Gas02] that the same algorithm solves the problem version in which both leader and follower have a bottleneck objective, the pessimistic setting is assumed, and the follower's objective is to minimize $\max_{e\in Y'} d(e)$. We next prove that a generalized form of the algorithm can be used to solve the problem version with follower's objective $\max_{e\in X\cup Y'} d(e)$, completing the investigation of all polynomial-time solvable cases.

Theorem 5.42. *The variant of the Bilevel Minimum Spanning Tree Problem in which both leader and follower have a bottleneck objective and the pessimistic setting is assumed can be solved in polynomial time.*

Proof. Let $(G, \mathcal{E}_l, \mathcal{E}_f, c, d)$ be an instance of the Bilevel Minimum Spanning Tree Problem. The algorithm works as follows: For all $e_c, e_d \in \mathcal{E}_l$ with $c(e_c) \geq c(e_d)$ and $d(e_c) \leq d(e_d)$, and such that either $e_c = e_d$ or e_c and e_d are not parallel, consider the set

$$E_{e_c,e_d} := \{e \in \mathcal{E}_l \mid c(e) \leq c(e_c) \text{ and } d(e) \leq d(e_d)\}.$$

The leader selects any edge set $X_{e_c,e_d} \subseteq E_{e_c,e_d}$ with $e_c, e_d \in X_{e_c,e_d}$ that consists of a spanning tree in each connected component of $G_{e_c,e_d} := (V(G), E_{e_c,e_d})$. Let Y_{e_c,e_d} be the corresponding response of the follower and c_{e_c,e_d} the resulting objective value, where $c_{e_c,e_d} := \infty$ in case X_{e_c,e_d} is infeasible, i.e., the follower cannot extend X_{e_c,e_d} to a spanning tree. If the case $c(e_c) = d(e_d) = 0$ does not occur, consider $X_0 := \emptyset$ as an additional candidate. Finally, choose (e_c^*, e_d^*) minimizing c_{e_c,e_d} and return $X_{e_c^*,e_d^*}$.

The algorithm clearly runs in polynomial time, so it remains to show that $X_{e_c^*,e_d^*}$ is indeed an optimum solution. For this, let $e_c, e_d \in \mathcal{E}_l$ be such that $c(e_c)$ and $d(e_d)$ are the maximum leader's and follower's edge costs, respectively, among a leader's optimum solution $X \subseteq \mathcal{E}_l$, assuming $X \neq \emptyset$. We show that X cannot have a smaller objective function value than c_{e_c,e_d}.

If X is a maximal forest in G_{e_c,e_d}, it leads to the same follower's response and hence the same objective function value as X_{e_c,e_d}. Otherwise, we may assume that $X \subsetneq X_{e_c,e_d}$. The follower cannot achieve a better objective value when responding to X than to X_{e_c,e_d}. Hence, by the pessimistic assumption, the maximum leader's edge cost among the follower's response cannot be smaller in the former than in the latter case. Since the maximum leader's and follower's edge costs among X and X_{e_c,e_d}, respectively, are the same, it follows that X cannot lead to a smaller objective function value than X_{e_c,e_d}. □

5.8.3 Hardness Results

We will now prove that the remaining bottleneck variants of the Bilevel Minimum Spanning Tree Problem are $\mathcal{NP}$-hard, namely the variant in which the leader has a sum objective and the follower has a bottleneck objective (Theorems 5.43 and 5.44), and the optimistic variant in which both decision makers have a bottleneck objective (Theorem 5.45). We will reuse several ideas from Sections 5.5 and 5.6 that can be applied directly or need to be changed slightly for the bottleneck cases.

First note that, in Theorem 5.23, the follower's sum objective can be easily replaced by a bottleneck objective, assuming the pessimistic setting:

Theorem 5.43. *The variant of the Bilevel Minimum Spanning Tree Problem in which the leader has a sum objective, the follower has a bottleneck objective, and the pessimistic setting is assumed, cannot be approximated to within a factor of* $^{96}/_{95}$ *in polynomial time, unless* $\mathcal{P} = \mathcal{NP}$*, even if* $\mathcal{E}_f$ *is a tree.*

Proof. We can use the same reduction from the Steiner Forest Problem as in the proof of Theorem 5.23. For the sake of simplicity, the follower's cost function can now be defined as $d(e) := 0$ for all edges $e \in E(G) \cup \mathcal{E}_f$. Then, the follower's objective value is always 0 and his decision is only guided by the pessimism. For this definition of d, both variants of the follower's bottleneck objective function, $\max_{e \in Y'} d(e)$ and $\max_{e \in X \cup Y'} d(e)$, are equivalent, hence this proof clearly holds for both of them. The pessimistic assumption about the follower's behavior here is equivalent to the behavior of a follower having a cost

function of $d(e) \coloneqq -c(e)$ for all $e \in \mathcal{E}_f$ and a sum objective, which is equivalent to the problem variant which is reduced to in Theorem 5.23; see also Remark 5.26. □

Theorem 5.43 cannot be easily adapted to the optimistic assumption. However, the hardness of this case can be concluded using Theorem 5.29:

Theorem 5.44. *All variants of the* Bilevel Minimum Spanning Tree Problem *where the leader has a sum objective and the follower has a bottleneck objective are $\mathcal{NP}$-hard, even if $c(e) = 1$ for all $e \in \mathcal{E}_l$.*

Proof. First note that the proof of Theorem 5.29 works without modification if the follower has a bottleneck objective, for both the optimistic and the pessimistic setting, no matter if the follower is taking only his own or all edges into account; for the latter case, define $d(e) \coloneqq 0$ for all $e \in \mathcal{E}_l$. Hence, all corresponding modifications of the Response Problem are $\mathcal{NP}$-complete as well, even if $|\overline{Y}| = 1$ and $\mathcal{E}_f$ forms a path on a subset of the vertex set.

We now show that the modified Response Problem can be reduced to the Bilevel Minimum Spanning Tree Problem in all these variants, assuming a sum objective for the leader, which proves the desired result. Consider an instance $(G, \mathcal{E}_l, \mathcal{E}_f, d, \overline{Y})$ of the Response Problem. Define a leader's cost function $c\colon E(G) \to \mathbb{Q}_{\geq 0}$ by setting

$$c(e) \coloneqq \begin{cases} 1, & \text{if } e \in \mathcal{E}_l, \\ 0, & \text{if } e \in \overline{Y}, \\ |V(G)|, & \text{if } e \in \mathcal{E}_f \setminus \overline{Y}. \end{cases}$$

We claim that the answer to the given instance of the Response Problem is "yes" if and only if the leader's optimum solution value in this instance of the Bilevel Minimum Spanning Tree Problem is at most $|V(G)| - |\overline{Y}| - 1$.

Assume that $X \subseteq \mathcal{E}_l$ is a leader's solution such that $\overline{Y}$ is the follower's response to X. Selecting X then yields a leader's objective value of $c(X) + c(\overline{Y}) = |V(G)| - |\overline{Y}| - 1$, since $X \cup \overline{Y}$ is a tree on $V(G)$ and hence consists of $|V(G)| - 1$ edges in total. Conversely, assume that the leader can achieve an objective value of at most $|V(G)| - |\overline{Y}| - 1$. By construction, this is only possible if the follower's response is exactly $\overline{Y}$ and the leader thus selects $|V(G)| - |\overline{Y}| - 1$ of her edges. Hence, the follower's response $\overline{Y}$ can be enforced. □

The proof of Theorem 5.44 does not carry over to cases in which the leader has a bottleneck objective because the reduction from the Response Problem to the Bilevel Minimum Spanning Tree Problem does not work there. However, the case in which both leader and follower have a bottleneck objective, assuming the optimistic setting, is $\mathcal{NP}$-hard as well, which can be shown using similar ideas as in the proof of Theorem 5.44.

Theorem 5.45. *The variant of the* Bilevel Minimum Spanning Tree Problem *in which both the leader and the follower have a bottleneck objective and the optimistic setting is assumed cannot be approximated to within any factor in polynomial time, unless $\mathcal{P} = \mathcal{NP}$.*

Proof. We show the result by a reduction from the Vertex-Disjoint Steiner Trees Problem, restricted to $k = 2$. Given an instance of the Vertex-Disjoint Steiner Trees Problem consisting of a connected graph G and disjoint sets $S = \{s_1, \dots, s_r\} \subseteq V(G)$ and $S' = \{s'_1, \dots, s'_{r'}\} \subseteq V(G)$, We define an instance of the Bilevel Minimum Spanning

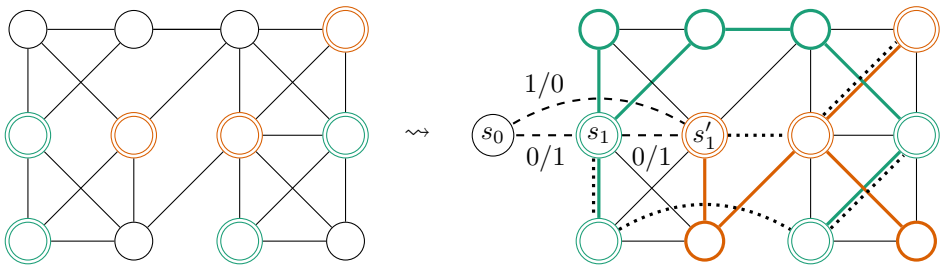


Figure 5.7. Illustration of the proof of Theorem 5.45. The two terminal sets in the instance of the Vertex-Disjoint Steiner Trees Problem are marked by green and orange vertices, respectively. Dotted lines represent edges in $\mathcal{E}_f^0$ having leader's cost 1 and follower's cost 0, dashed lines have costs as specified. Solid edges are controlled by the leader and have leader's and follower's cost 0. One feasible solution of the Vertex-Disjoint Steiner Trees Problem that covers all vertices is marked by green and orange edges.

Tree Problem as follows: define a graph G' by $V(G') \coloneqq V(G) \cup \{s_0\}$ for a new vertex s_0 and $E(G') \coloneqq \mathcal{E}_l \cup \mathcal{E}_f$, where $\mathcal{E}_l \coloneqq E(G)$,

$$\mathcal{E}_f^0 \coloneqq \{\{s_i, s_{i+1}\} \mid i \in \{1, \dots, r-1\}\} \cup \{\{s'_i, s'_{i+1}\} \mid i \in \{1, \dots, r'-1\}\},$$

and $\mathcal{E}_f \coloneqq \mathcal{E}_f^0 \cup \{\{s_1, s'_1\}, \{s_0, s_1\}, \{s_0, s'_1\}\}$. Let the leader's and follower's costs be defined as follows:

$$c(e) \coloneqq \begin{cases} 0, & \text{if } e \in \mathcal{E}_l \cup \{\{s_1, s'_1\}, \{s_0, s_1\}\}, \\ 1, & \text{if } e \in \mathcal{E}_f^0 \cup \{\{s_0, s'_1\}\}, \end{cases}$$

$$d(e) \coloneqq \begin{cases} 0, & \text{if } e \in \mathcal{E}_l \cup \mathcal{E}_f^0 \cup \{\{s_0, s'_1\}\}, \\ 1, & \text{if } e \in \{\{s_1, s'_1\}, \{s_0, s_1\}\}. \end{cases}$$

This construction is illustrated in Figure 5.7.

We now show that the answer to the given instance of the Vertex-Disjoint Steiner Trees Problem is "yes" if and only if the leader's optimum value in the constructed instance of the Bilevel Minimum Spanning Tree Problem is 0. Since $d(e) = 0$ for all $e \in \mathcal{E}_l$, the following arguments hold for both types of follower's objective functions, $\max_{e \in Y'} d(e)$ as well as $\max_{e \in X \cup Y'} d(e)$.

Assume that T and T' are vertex-disjoint trees in G such that $S \subseteq V(T)$ and $S' \subseteq V(T')$. Since G is connected, we may assume that $E(T) \cup E(T')$ covers all vertices of G, by connecting all non-covered vertices to either T or T' arbitrarily. If the leader selects $X \coloneqq E(T) \cup E(T')$ as her solution, the follower must take any two of the three edges $\{s_1, s'_1\}$, $\{s_0, s_1\}$, and $\{s_0, s'_1\}$ in order to complete X to a spanning tree. As the follower's objective value is 1 for any of these choices and we assume the optimistic setting, his response is $Y \coloneqq \{\{s_1, s'_1\}, \{s_0, s_1\}\}$, resulting in a leader's objective value of 0.

For the other direction, assume that the leader can achieve an objective value of 0. This means that the follower uses the edge $\{s_0, s_1\}$ in order to connect the vertex s_0 to the original graph. Since this edge is more expensive than $\{s_0, s'_1\}$ for the follower, he will only do that if he is also forced to connect the vertices s_1 and s'_1 because, otherwise, he can always achieve an objective value of 0. Hence, the leader must not connect s_1 and s'_1. Moreover, all vertices in S have to be connected by the leader, as well as all vertices in S',

in order to prevent the follower from selecting any edge from $\mathcal{E}_f^0$. Thus, the leader's solution contains two vertex-disjoint trees each of which covers one of the vertex sets S and S'.

The proof not only implies $\mathcal{NP}$-hardness, but even that computing any approximate solution is $\mathcal{NP}$-hard because the reduction only relies on distinguishing whether the optimum value is 0 or 1; see also Section 2.1.2. □

5.9 Conclusion

In this chapter, we investigated the computational complexity of the Bilevel Minimum Spanning Tree Problem. After giving some structural insights, we proved that the Bilevel Minimum Spanning Tree Problem is $\mathcal{NP}$-hard, thus answering a conjecture stated in [SZP19]. Furthermore, we considered the parameterized complexity of the problem in the number of edges controlled by the follower and showed that the problem is at least as hard as the Shortest Vertex-Disjoint Steiner Trees Problem, parameterized by the number of terminal vertices, giving some evidence that the Bilevel Minimum Spanning Tree Problem might be intractable even for a fixed number of follower's edges. Moreover, we devised a polynomial-time $(|V(G)| - 1)$-approximation algorithm for the Bilevel Minimum Spanning Tree Problem, and showed that 2-approximating the problem is fixed-parameter tractable in the number of follower's edges. Finally, we considered several variants of the Bilevel Minimum Spanning Tree Problem in which at least one of the two decision makers has a bottleneck objective function and gave a complete complexity classification of all these variants.

Compared with the bilevel optimization problems considered in Chapters 3 and 4, the Bilevel Minimum Spanning Tree Problem is more involved, as it is $\mathcal{NP}$-hard already without uncertainty. Like the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem, it has the structure of a partitioned-items bilevel problem, and it is also based on a very easy single-level combinatorial optimization problem so that, in particular, the follower's problem can be solved efficiently in all cases. However, the interaction between the two decision makers is much more complex now, due to the involved general graph structure. See Section 6.1 for a discussion of the connections between the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem, which can be both seen as special cases of the Bilevel Minimum Matroid Basis Problem.

Open questions arising from this chapter will be discussed in Section 6.3.3.

6 Conclusions, Generalizations, and Outlook

In this thesis, we have provided an extensive study of the computational complexity of several specific bilevel optimization problems with a relatively easy structure, as well as of robust versions of some of them and of a more general class of robust bilevel optimization problems.

Our focus was on the Bilevel Selection Problem (Chapter 3), the Bilevel Continuous Knapsack Problem (Chapter 4), and the Bilevel Minimum Spanning Tree Problem (Chapter 5), and we considered each of them in several variants. The Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem (without uncertainty) have been shown to be solvable in polynomial time, while the Bilevel Minimum Spanning Tree Problem turned out to be $\mathcal{NP}$-hard. All three problems fall into the category of *partitioned-items bilevel problems*, where the leader and the follower each control a subset of a common ground set of items and together build a feasible solution to the underlying single-level optimization problem, while having different objective functions. Together with [CCLW14; FMW22; HW25] (see also Sections 2.2.2.2 and 4.2.3), the computational complexity of the partitioned-items bilevel versions of most classical polynomial-time solvable combinatorial optimization problems and of the Knapsack Problem is settled now. However, there are still open questions, other problem variants and other underlying single-level problems that can be investigated; see Section 6.3.

The relatively simple structure of the problems studied in this thesis is related to the Selection Problem and the Minimum Spanning Tree Problem being special cases of the Minimum Matroid Basis Problem. We will discuss some common structural features of the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem and their possible generalization to the Bilevel Minimum Matroid Basis Problem in Section 6.1. The Bilevel Continuous Knapsack Problem does not fit into this framework, but can also be seen as a straightforward generalization of the Bilevel Selection Problem (under Assumptions (disj) and (cc)). We refer to Sections 4.5 and 6.2 for a summary of similarities and differences of these two problems, with and without uncertainty.

Our motivation was not only to study the complexity of variants of partitioned-items bilevel problems, but also to study the additional complexity that robustness introduces in bilevel optimization problems, where the follower's objective function is uncertain from the leader's perspective. In fact, without uncertainty, the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem turned out to be solvable by relatively simple polynomial-time algorithms. This makes them particularly suited for studying the additional complexity introduced by the uncertainty. We analyzed the complexity of the Robust Bilevel Selection Problem and the Robust Bilevel Continuous Knapsack Problem with different types of uncertainty sets, which revealed some interesting structural insights and several differences to single-level robust optimization. These will be summarized in Section 6.2. Besides the studies of specific robust bilevel problems, a similar analysis was also performed for a more general class of robust bilevel

problems in Section 2.4.2. With these insights, we add to the still relatively limited amount of literature on bilevel optimization under uncertainty, in particular regarding its computational complexity.

In Section 6.1, we will present the Bilevel Minimum Matroid Basis Problem as a common generalization of the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem and discuss some open questions and ideas related to this problem. In Section 6.2, we will then summarize the most important insights about the complexity of robust bilevel optimization problems that can be derived from the results in this thesis. Finally, we will list open question that arise from the studies in this thesis in Section 6.3.

6.1 The Bilevel Minimum Matroid Basis Problem

Besides both being partitioned-items bilevel problems, the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem also have a more specific structure in common: Both can be seen as special cases of the Bilevel Minimum Matroid Basis Problem, where the leader and the follower build a basis of a matroid together. In this section, we discuss some common features of the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem and their generalization to the Bilevel Minimum Matroid Basis Problem.

Other types of bilevel optimization problems based on general matroids have been studied in, e.g., [Fan11; BS20; WF25b].

The following basic notions and properties of matroids are taken from [KV18, Chapter 13] and we refer to there for more information about matroids.

Definition 6.1. A *matroid* consists of a finite ground set $\mathcal{E}$ and a family $\mathcal{F} \subseteq \mathcal{P}(\mathcal{E})$ of subsets of $\mathcal{E}$ such that the following three properties are satisfied:

- $\emptyset \in \mathcal{F}$
- If $Y \in \mathcal{F}$ and $X \subseteq Y$, then $X \in \mathcal{F}$.
- If $X, Y \in \mathcal{F}$ with $|X| > |Y|$, then there is some $x \in X \setminus Y$ such that $Y \cup \{x\} \in \mathcal{F}$.

The elements of $\mathcal{F}$ are called *independent sets* and the maximal independent sets are called *bases*. All bases of a matroid have the same cardinality, the *rank* of the matroid.

From the perspective of combinatorial optimization, a useful property of matroids is that a linear objective function can be optimized over the bases by using a greedy algorithm. More precisely, the Minimum Matroid Basis Problem can be defined as follows. Note that, for algorithmic considerations, a matroid is usually assumed to be given by an *independence oracle*, i.e., an oracle which, given a set $X \subseteq \mathcal{E}$, decides if $X \in \mathcal{F}$ or $X \notin \mathcal{F}$.

Minimum Matroid Basis Problem
Instance: A matroid $(\mathcal{E}, \mathcal{F})$ and a weight function $c\colon \mathcal{E} \to \mathbb{Q}$
Task: Find a basis $X \in \mathcal{F}$ such that $c(X)$ is minimum.

The greedy algorithm that solves this problem sorts the items in $\mathcal{E}$ by their weights and iterates through them in nondecreasing order, adding each item to the solution X that does not cause X to become dependent.

Two simple examples of matroids correspond to the underlying combinatorial optimization problems we studied in Chapters 3 and 5:

In a *uniform matroid*, the independent sets are given by $\mathcal{F} = \{X \subseteq \mathcal{E} \mid |X| \leq b\}$ for some given number $b \in \mathbb{N}_0$. The bases of this matroid are all sets $X \subseteq \mathcal{E}$ of cardinality $|X| = b$ and therefore exactly the feasible sets in a Selection Problem, as we defined it in Section 3.1. Hence, the Minimum Matroid Basis Problem corresponds to the Selection Problem in case of a uniform matroid. Recall that we defined the Selection Problem as a maximization problem, but the direction of optimization can be easily changed if we allow for arbitrary item weights; see Section 3.1 and Remark 3.5 for more details. The greedy algorithm described in Section 3.1 is the corresponding special case of the general greedy algorithm mentioned above.

In a *graphic matroid*, an undirected connected graph $G = (V, \mathcal{E})$ is given and the independent sets are $\mathcal{F} = \{X \subseteq \mathcal{E} \mid (V, X) \text{ is a forest}\}$. The bases of this matroid are the spanning trees of the graph G. Hence, for a graphic matroid, the Minimum Matroid Basis Problem corresponds to the Minimum Spanning Tree Problem (see Section 5.1). The greedy algorithm for this problem is Kruskal's algorithm, as described in Section 5.1.

We often made use of the matroid properties of the underlying single-level problems of our bilevel problems when assuming that the follower decides according to a fixed greedy order given by his item values. This raises the question if some of the insights obtained in Chapters 3 and 5 can be generalized to the partitioned-items bilevel version of the Minimum Matroid Basis Problem:

Bilevel Minimum Matroid Basis Problem

Instance: A matroid $(\mathcal{E}, \mathcal{F})$, leader's and follower's item sets $\mathcal{E}_l \subseteq \mathcal{E}$ and $\mathcal{E}_f \subseteq \mathcal{E}$, respectively, with $\mathcal{E}_l \cup \mathcal{E}_f = \mathcal{E}$, and leader's and follower's item weights $c\colon \mathcal{E} \to \mathbb{Q}$ and $d\colon \mathcal{E}_f \to \mathbb{Q}$, respectively

Task: Find a subset $X \subseteq \mathcal{E}_l$ for the leader such that $c(X \cup Y)$ is minimum, where $Y \subseteq \mathcal{E}_f \setminus X$ is a follower's subset that makes $X \cup Y$ a basis of $(\mathcal{E}, \mathcal{F})$ and minimizes $d(Y)$.

Analogously to (BSP1) (see Section 3.2.1) and (BMSTP) (see Section 5.2.1), the Bilevel Minimum Matroid Basis Problem can be written as a bilevel mathematical program as follows:

$$\begin{aligned} &\min_{X} \ c(X \cup Y) \\ &\text{s.t. } X \subseteq \mathcal{E}_l \\ &\quad Y \in \operatorname*{argmin}_{Y'} \ d(Y') \\ &\qquad \text{s.t. } Y' \subseteq \mathcal{E}_f \setminus X \\ &\qquad\quad X \cup Y' \text{ is a basis of } (\mathcal{E}, \mathcal{F}) \end{aligned} \tag{BMMBP}$$

Like in case of the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem, also the follower's problem of the Bilevel Minimum Matroid Basis Problem, given a fixed feasible leader's solution $X \subseteq \mathcal{E}_l$, can be seen as an instance of a single-level Minimum Matroid Basis Problem and therefore solved by a greedy algorithm. To this end, extend the follower's item weight function $d\colon \mathcal{E}_f \to \mathbb{Q}$ to a function $d_{\mathcal{E}}\colon \mathcal{E} \to \mathbb{Q}$ on all items $\mathcal{E}$, e.g., by setting $d_{\mathcal{E}}(e) := -M$ for $e \in X$, $d_{\mathcal{E}}(e) := M$ for $e \in \mathcal{E}_l \setminus \mathcal{E}_f$, and $d_{\mathcal{E}}(e) := d(e)$ for $e \in \mathcal{E}_f \setminus X$, where $M \in \mathbb{Q}_{\geq 0}$ is a number that is greater than the absolute values of all item weights given by d. Note that, analogously to Remarks 3.2 and 5.7, the optimistic or pessimistic setting can be modeled by incorporating the leader's

objective into the follower's sorting as a secondary criterion.

For the Bilevel Selection Problem, we distinguished many problem variants based on the assumptions defined in Sections 3.2.2, 3.2.3, and 3.2.4. Similar variants can be considered for the Bilevel Minimum Matroid Basis Problem. The Assumptions (arb), (sub), (disj), and (uni) about the relation between the leader's and the follower's item sets can be equivalently applied here. Recall that there is a significant difference between Assumptions (sub) and (disj) in case of the Bilevel Selection Problem (in particular when adding robustness), while Assumption (disj) could be made without loss of generality in case of the Bilevel Minimum Spanning Tree Problem and hence does not simplify the problem compared to the most general problem variant; see Remark 5.10 and Section 5.3 for more details. Thus, the impact of the assumptions about the two decision makers' item sets seems to depend on the specific matroid in the Bilevel Minimum Matroid Basis Problem.

We introduced the Bilevel Minimum Matroid Basis Problem as a purely combinatorial problem, like the Bilevel Selection Problem under Assumption (bb) and the Bilevel Minimum Spanning Tree Problem. However, one could also assume one or both of the two decision makers to make continuous decisions in the sense that they may select fractions of items and have to obtain a point in the matroid basis polytope (i.e., the convex hull of the incidence vectors of all bases). In case of the Bilevel Selection Problem, a formulation for this could be easily achieved by replacing the binary variables by continuous variables; see (BSP2) with the Assumptions (bc), (cb), and (cc).

Finally, consider the Assumptions (=) and (≤) about the capacity constraint in the Bilevel Selection Problem. In case of Assumption (=), one optimizes over the bases, like in the Bilevel Minimum Matroid Basis Problem as defined above, while Assumption (≤) corresponds to the setting where all independent sets are feasible. The latter could also be studied as a variant of the Bilevel Minimum Matroid Basis Problem.

For the Bilevel Minimum Spanning Tree Problem, we introduced similar and additional restrictions in Section 5.3, most of which can also be formulated for general matroids. In particular, for the Bilevel Minimum Matroid Basis Problem, one might study the assumptions that $\mathcal{E}_l$ and/or $\mathcal{E}_f$ contains a basis of the matroid $(\mathcal{E}, \mathcal{F})$ (which means that the corresponding subgraph is connected in case of the Bilevel Minimum Spanning Tree Problem) or is an independent set (which means that the corresponding subgraph is a forest in case of the Bilevel Minimum Spanning Tree Problem). Analogously to Lemmas 5.12 and 5.13, the assumptions that $\mathcal{E}_l$ and $\mathcal{E}_f$ each contain a basis can be made without loss of generality. The same is true for the assumption that $\mathcal{E}_f$ is independent, which will be discussed in the following.

First recall Lemma 5.14, where we proved a fundamental structural property of the Bilevel Minimum Spanning Tree Problem, namely that the follower's responses $Y_1, Y_2 \in \mathcal{E}_f$ corresponding to leader's solutions $X_1, X_2 \in \mathcal{E}_l$ with $X_1 \subseteq X_2$ satisfy $Y_1 \supseteq Y_2$. The proof heavily relies on the greedy property of the follower's problem and can be easily generalized to the case of general matroids. For this, note that the claim that is shown by induction in the proof of Lemma 5.14 can be formulated as follows for a general matroid $(\mathcal{E}, \mathcal{F})$: For all $i \in \{0, \dots, m\}$, we have $A(X_1 \cup Y_1^{(i)}) \subseteq A(X_2 \cup Y_2^{(i)})$, where, given an independent set $Z \in \mathcal{F}$, we denote the set of items that would make Z dependent when

being added by $A(Z) \coloneqq \{e \in \mathcal{E} \mid Z \cup \{e\} \notin \mathcal{F}\}$.

Analogously to Corollary 5.15, the generalization of Lemma 5.14 implies that, in the Bilevel Minimum Matroid Basis Problem, one can assume without loss of generality that the set $\mathcal{E}_f$ is independent. Informally speaking, this is because edges that form a circuit with edges of better follower's weight will never be selected by the follower and can therefore be eliminated.

Note that we made use of the analogs of Lemma 5.14 and Corollary 5.15 also for the Bilevel Selection Problem, where they are very easy to see. For instance, we refer to Algorithm 3.1 and to the proof of Theorem 3.58, where it was sufficient to define the follower's greedy orders only for the first b items.

We have now argued that the Bilevel Minimum Matroid Basis Problem admits some general structural properties, which are related to the follower's problem being solvable by a greedy algorithm. The leader's problem, however, is much more complicated. In fact, our $\mathcal{NP}$-hardness result for the Bilevel Minimum Spanning Tree Problem (see Theorem 5.23) implies that the Bilevel Minimum Matroid Basis Problem is $\mathcal{NP}$-hard in general. In contrast, the Bilevel Selection Problem has been shown to be solvable in polynomial time for all considered problem variants; see Section 3.3.1. However, this was possible due to the very simple structure of the uniform matroid and cannot be expected to be extensible to many other matroids.

Regarding other results presented in Chapters 3 and 5, one would need to check carefully whether anything can be generalized to the Bilevel Minimum Matroid Basis Problem. For example, the approximation algorithm presented for the Bilevel Minimum Spanning Tree Problem in Theorem 5.37 might be a good candidate, as it is built on the structural insights of Lemma 5.14 and Corollary 5.15. For the Bilevel Minimum Spanning Tree Problem, we have also considered bottleneck variants of the leader's and the follower's objective functions; see Section 5.8. Such variants can be analogously defined for the general Bilevel Minimum Matroid Basis Problem, and we expect that many of the results in Section 5.8 can be generalized as well.

Also robust variants of the Bilevel Minimum Matroid Basis Problem could be studied, like the Robust Bilevel Selection Problem in Section 3.4. These can be expected to be very hard for general matroids given that the Bilevel Minimum Matroid Basis Problem is already $\mathcal{NP}$-hard without uncertainty. However, note that the robust problem with discrete uncertainty is $\mathcal{NP}$-easy because the adversary can enumerate all scenarios, like in Theorem 2.39; see also Section 6.2.

Besides extending results to general matroids, it could also be interesting to study (the computational complexity of) variants of the Bilevel Minimum Matroid Basis Problem, and possibly their robust versions, for other specific matroids than the two discussed in this thesis. For example, we expect that many of the results for the Bilevel Selection Problem and the Robust Bilevel Selection Problem can be extended to the (robust) Bilevel Minimum Matroid Basis Problem for the *partition matroid*, which is a direct sum of uniform matroids and corresponds to the *multi-representative selection problem*; see, e.g., [GH24, Chapter 5.4].

6.2 Complexity of Robust Bilevel Optimization

In Section 2.4.2.3, we have identified several differences regarding the complexity that robustness adds to bilevel optimization problems compared to single-level problems. The results of Section 2.4.2 have been confirmed and refined by our studies of the Robust Bilevel Selection Problem and the Robust Bilevel Continuous Knapsack Problem in Chapters 3 and 4, as we discuss in the following.

Recall that both the Bilevel Selection Problem and the Bilevel Continuous Knapsack Problem without uncertainty are solvable in polynomial time. Not surprisingly, a robust bilevel optimization problem is usually more complicated than its underlying bilevel optimization problem. Sometimes this reflects in an additional level of complexity, e.g., in case of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty (see Theorem 3.58), and in case of the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty (see Corollary 4.23). Other variants of the Robust Bilevel Selection Problem and the Robust Bilevel Continuous Knapsack Problem were still solvable in polynomial time, but required a more clever algorithm than the corresponding problems without uncertainty; see, e.g., Algorithms 3.4 and 4.3 for interval uncertainty.

Regarding specific types of uncertainty sets, we studied discrete uncertainty and interval uncertainty in Section 2.4.2 as well as in Sections 3.4 and 4.4. For discrete uncertainty, the situation is very similar in all cases: The adversary's problem is still solvable in polynomial time because he can enumerate all scenarios in the uncertainty set and solve the follower's problem for each of them; see Theorems 2.39, 3.27, 3.43, and 4.12. Therefore, the leader's problem is $\mathcal{NP}$-easy in general (see Theorem 2.39). However, while the leader's problem is still polynomial-time solvable in case of the Robust Bilevel Selection Problem under Assumption (disj) (see Corollary 3.28 and Theorem 3.44) and in case of the Robust Bilevel Continuous Knapsack Problem (see Theorem 4.13), we proved that it is $\mathcal{NP}$-hard in case of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb) (see Theorem 3.58).

The setting of the Robust Bilevel Selection Problem under Assumptions (sub) and (bb) is also interesting because it demonstrates a difference to the typical situation in single-level robust optimization. In the latter setting, many problems are weakly $\mathcal{NP}$-hard in case of a constant number of scenarios and strongly $\mathcal{NP}$-hard in case of an unbounded number of scenarios (see Section 2.3.2.1). For the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with discrete uncertainty, we have shown in Theorem 3.58 that it is strongly $\mathcal{NP}$-hard if the number of scenarios is part of the input, but Theorem 3.63 implies that it is solvable in polynomial time in case of a constant number of scenarios.

Interval uncertainty can lead to an additional level of complexity in the adversary's and in the leader's problem (see Theorem 2.33), while the Robust Bilevel Selection Problem under Assumption (disj) and the Robust Bilevel Continuous Knapsack Problem are still solvable in polynomial time also in this case (see Corollary 3.35 and Theorems 3.48 and 4.20). However, the algorithms we developed for interval uncertainty are more involved than the ones for discrete uncertainty. Moreover, the fact that the leader's problem is on the same level of the polynomial-time hierarchy as the adversary's problem in case of the Robust Bilevel Selection Problem under Assumption (disj) and the Robust Bilevel Continuous Knapsack Problem can be attributed to the relatively simple

nature of the leader's decision, which allows her, in case of Assumption (bb), to enumerate a single parameter (see Theorem 3.25), and in case of Assumption (cc), to represent her objective function as a piecewise linear function in a single parameter (see Section 3.4.1.2). We expect this to be different for the Robust Bilevel Selection Problem under Assumption (sub); see Section 3.4.2.1.2 and Section 3.4.2.2. As mentioned in Section 2.4.2.3, we can conclude that, in robust bilevel optimization, interval uncertainty is not trivial, like it is often the case in robust single-level optimization, and can even be significantly harder than discrete uncertainty.

For some of our problems, we have studied additional types of uncertainty sets. Discrete uncorrelated uncertainty leads to a polynomial-time solvable Robust Bilevel Selection Problem under Assumption (disj) (see Corollary 3.39 and Theorem 3.52), while, for the Robust Bilevel Continuous Knapsack Problem, both the adversary's and the leader's problem turned out to be $\mathcal{NP}$-hard (see Corollaries 4.29 and 4.23) and solvable in pseudopolynomial time (see Theorems 4.30 and 4.31).

Additional complexity has also been introduced in the Robust Bilevel Continuous Knapsack Problem, again both for the adversary's and the leader's problem, by simplicial uncertainty (see Corollary 4.39), which is a special case of budgeted uncertainty and of polytopal uncertainty, the latter both in case the polytope is given by an inner description and in case it is given by an outer description. An uncertainty set that is a polytope given by its inner description is also considered in Theorem 2.40, where it introduces additional complexity as well.

Finally, we have considered p-norm uncertainty, which includes (uncorrelated) ellipsoidal uncertainty, for the Robust Bilevel Continuous Knapsack Problem. Also here, we have obtained $\mathcal{NP}$-hardness results for both the adversary's and the leader's problem (see Corollaries 4.47 and 4.45). This shows another difference to the typical situation in single-level robust optimization, where uncorrelated ellipsoidal uncertainty often leads to a tractable robust counterpart; see Section 2.3.2.6. On the other hand, it is typical that results for ellipsoidal uncertainty hold for all p-norms; see Section 2.3.2.7.

Note that, given the $\mathcal{NP}$-hardness results that hold for both the adversary's and the leader's problem, it remains open whether the leader's problems are on the first or actually on the second level of the polynomial-time hierarchy; see also Section 6.3.2.

Another insight that appeared several times throughout this thesis is that uncertainty sets cannot be replaced by their convex hulls without changing the problem, in case of robust bilevel optimization. Recall that this is usually different in the single-level case; see Lemma 2.26. In Section 2.4.2, we have seen that both the adversary's problem and the leader's problem can become one level harder when replacing an uncertainty set by its convex hull. Similarly, Theorem 4.13 and Corollary 4.40 imply that the adversary's and the leader's problem of the Robust Bilevel Continuous Knapsack Problem become one level harder when replacing a discrete uncertainty set by its convex hull. But we have also presented an example of the opposite, as the Robust Bilevel Continuous Knapsack Problem with interval uncertainty is solvable in polynomial time, while it is $\mathcal{NP}$-hard in case of discrete uncorrelated uncertainty. Apart from the complexity, we have also shown that the problem structure and the optimum solutions may change when replacing an uncertainty set by its convex hull; see, e.g., Example 3.50.

6.3 Open Questions

In this section, we list specific questions that remain open in this thesis as well as some possible directions of future research. We start with questions related to Chapters 3, 4, and 5 in Sections 6.3.1, 6.3.2, and 6.3.3, and conclude with a few more general questions in Section 6.3.4.

6.3.1 Open Questions From Chapter 3

We have studied many variants of the Bilevel Selection Problem, and the structural properties and the complexity are settled for most of these problem variants without uncertainty (see Sections 3.2.5 and 3.3). Only the problem variant corresponding to Assumption (cb), where the leader takes a continuous decision and the follower takes a binary decision, turned out to be special, as the optimum value might not be attained in this case (see Lemma 3.12). It could be interesting to study this problem variant further, e.g., in view of computing the optimum value or computing ε-optimal solutions or feasible solutions that approach the optimum value.

For the majority of Chapter 3, we have studied robust versions of the Bilevel Selection Problem. Here, some cases are not well understood yet. For example, we have shown that the Robust Bilevel Selection Problem under Assumptions (sub) and (bb) is $\mathcal{NP}$-hard in case of discrete uncertainty (see Theorem 3.58), but we do not know its complexity for any other type of uncertainty set. In particular, it would be interesting to study interval uncertainty, since we already have a 2-approximation algorithm for this setting (see Corollary 3.65).

Also in case of Assumption (disj) (together with either of the Assumptions (bb) and (cc)), it could be interesting to study other types of uncertainty sets, e.g., budgeted, polytopal, or ellipsoidal uncertainty, like in Section 4.4. If the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) could be solved in polynomial time here, then it would again turn out to be easier than its generalization, the Robust Bilevel Continuous Knapsack Problem, like for discrete uncorrelated uncertainty (see Section 4.4.3). Moreover, polynomial-time algorithms for this setting could again be helpful to develop pseudopolynomial-time algorithms for the Robust Bilevel Continuous Knapsack Problem in the corresponding cases (see also Section 4.4.3.2).

Finally, in Section 3.4.2.2, we discussed the Robust Bilevel Selection Problem under Assumptions (sub) and (cc), but the understanding of this case is still very limited. The structural insights regarding piecewise linear functions, which we obtained for the case of Assumptions (disj) and (cc) (see Section 3.4.1.2), do not seem to be easily applicable to this setting, and we conjecture that it becomes significantly harder, similarly to the case of Assumptions (sub) and (bb) compared to the one of Assumption (disj). We refer to Section 3.4.2.2 for more details and ideas that could be investigated further.

We have defined the Selection Problem and the Bilevel Selection Problem as maximization problems in Chapter 3. Most of our results hold for both the minimization and the maximization variant (see also Remark 3.5), but the proof of the approximation result in Theorem 3.56 is only valid for the minimization version. This suggests the open question whether a similar result can also be obtained for the corresponding maximization problem. Concerning Theorem 3.56, one could also investigate if a better approximation factor can be achieved by another algorithm (while Examples 3.61 and 3.66 show that

the factor of 2 is tight for the given algorithm) or if any lower bound for a possible approximation guarantee can be obtained.

6.3.2 Open Questions From Chapter 4

The Robust Bilevel Continuous Knapsack Problem generalizes the Robust Bilevel Selection Problem under Assumptions (disj) and (cc) and we were able to settle its complexity for many types of uncertainty sets. The setting of Assumptions (disj) and (cc) seems to be the easiest one (also) in case of the Bilevel Knapsack Problem, which was the reason for focusing on it. However, also the other main variants of the Bilevel Knapsack Problem, corresponding to Assumption (sub) and/or Assumption (bb) could be interesting to investigate further.

In the setting of Assumptions (sub) and (cc), even the complexity of the Bilevel Knapsack Problem without uncertainty is open. While the Bilevel Selection Problem without uncertainty was solvable in polynomial time under Assumption (disj) as well as under Assumption (sub) by the same algorithm (see Algorithm 3.1), it is not clear if the polynomial-time algorithm for the Bilevel Continuous Knapsack Problem (see Algorithm 4.1) can be transferred to the setting of Assumption (sub), or if the problem might already be $\mathcal{NP}$-hard in this setting.

Before studying robust versions of the Bilevel Knapsack Problem under Assumptions (sub) and (cc) in more detail, it might also be useful to first understand the corresponding version of the Robust Bilevel Selection Problem better, where many questions are still open as well; see Section 3.4.2.2.

The Bilevel Knapsack Problem with Assumption (bb) is already Σ_2^{p}-hard without uncertainty, even in the presumably easier setting of Assumption (disj); see Section 4.2.3. Hence, robust versions of this problem can be expected to be at least on the second, possibly even on the third level of the polynomial-time hierarchy. The Bilevel Knapsack Problem under Assumption (bb) could be a good starting point to investigate the additional complexity that robustness introduces in a harder underlying bilevel optimization problem because its structure is still relatively easy to understand.

While the Bilevel Knapsack Problem under Assumptions (disj) and (bb) can be solved in pseudopolynomial time (see Section 4.2.3), it is open if the Bilevel Knapsack Problem with Assumptions (sub) and (bb) is also solvable in pseudopolynomial time or if it is strongly $\mathcal{NP}$-hard (or even strongly Σ_2^{p}-hard). In fact, this setting could be related to the bilevel knapsack problem of [DeN11] (see Section 4.2.3 for more details), which has been shown to be strongly $\mathcal{NP}$-hard in [CCLW14].

Also the settings of Assumptions (bc) and (cb), where one of the two decision makers takes a binary decision and the other one takes a continuous decision, could be studied further for the Bilevel Knapsack Problem; see Section 4.2.2 for more details.

For the Robust Bilevel Continuous Knapsack Problem with discrete uncorrelated uncertainty, simplicial uncertainty, and norm uncertainty, we proved $\mathcal{NP}$-hardness of both the adversary's and the leader's problems (see Sections 4.4.3, 4.4.4, and 4.4.5). This raises the question whether the leader's problem is $\mathcal{NP}$-equivalent or even Σ_2^{p}-hard in these cases. Exploring this could yield a better understanding of problems on the second level of the polynomial-time hierarchy and of robust bilevel optimization.

Moreover, all $\mathcal{NP}$-hardness results obtained in Chapter 4 are only in the weak sense. For the adversary's and the leader's problem of the Robust Bilevel Continuous Knapsack

Problem under discrete uncorrelated uncertainty, we presented pseudopolynomial-time algorithms in Section 4.4.3.2. But in the cases of simplicial uncertainty and norm uncertainty, it is still open if pseudopolynomial-time algorithms exist or if the problems are strongly $\mathcal{NP}$-hard.

As mentioned in Section 4.5, similar complexity results for a different robust version of the Bilevel Continuous Knapsack Problem (under Assumption (uni)), where the (follower's) item sizes instead of the follower's item values are uncertain, have been obtained in [Hüg20]. One could try to extend these results, e.g., to other problem versions, and also consider the variant where uncertainty appears both in the follower's objective and in the constraints.

6.3.3 Open Questions From Chapter 5

For the Bilevel Minimum Spanning Tree Problem, we have shown that it is strongly $\mathcal{NP}$-hard in Theorem 5.23. The reduction also implies that the problem cannot be approximated to within a factor of $^{96}/_{95}$ in polynomial time. Our approximation algorithm (see Theorem 5.37) achieves an approximation factor of $|V(G)|-1$. Given this large gap, it is an interesting question to study the approximability of the Bilevel Minimum Spanning Tree Problem further.

We have considered variants of the Bilevel Minimum Spanning Tree Problem with a small number of follower's edges in Section 5.6. It is still open whether the problem is fixed-parameter tractable in this parameter and even whether it can be solved in polynomial time for any fixed number of follower's edges greater than one. We have only shown fixed-parameter tractability for the special case of the Bilevel Minimum Spanning Tree Problem where all leader's edges have the same cost (see Corollary 5.32) and for the problem of 2-approximating the Bilevel Minimum Spanning Tree Problem (see Theorem 5.36). The relation between the Bilevel Minimum Spanning Tree Problem and the Shortest Vertex-Disjoint Steiner Trees Problem established in Theorem 5.33 makes the complexity of the Bilevel Minimum Spanning Tree Problem with a small number of follower's edges a particularly interesting question, since a polynomial-time algorithm for the case of four follower's edges would also give a polynomial-time algorithm for the Shortest Vertex-Disjoint Paths Problem; see Section 5.6 for more details. However, even the cases of two and three follower's edges are still open; see Remark 5.34.

As a common generalization of the Bilevel Selection Problem and the Bilevel Minimum Spanning Tree Problem, we discussed the Bilevel Minimum Matroid Basis Problem in Section 6.1. We mentioned many open questions regarding this generalization in Section 6.1. Note that some of them can also be applied to the Bilevel Minimum Spanning Tree Problem specifically. For example, one could think of a variant of the Bilevel Minimum Spanning Tree Problem where one or both decision makers take a continuous decision such that the combined solution must be contained in the spanning tree polytope, i.e., the convex hull of the incidence vectors of all spanning trees.

Also robust versions of the Bilevel Minimum Spanning Tree Problem could be interesting to investigate. Since the Bilevel Minimum Spanning Tree Problem without uncertainty is already $\mathcal{NP}$-hard, they can be expected to be on the first or on the second level of the polynomial-time hierarchy. The Bilevel Minimum Spanning Tree Problem might be a suitable first problem to investigate when studying the computational

complexity that robustness introduces in $\mathcal{NP}$-hard bilevel optimization problems, as it still has a relatively simple structure. Note again that the robust bilevel problem with discrete uncertainty is $\mathcal{NP}$-easy; see Section 6.2. Hence, the robust version of the Bilevel Minimum Spanning Tree Problem with discrete uncertainty is $\mathcal{NP}$-equivalent.

6.3.4 Other Open Questions

Besides the problem-specific open questions listed above, we finally mention a few more general open questions.

First, the complexity considerations in Section 2.4.2 for a general class of robust bilevel optimization problems, covering discrete uncertainty and interval uncertainty, could be extended. In particular, one could study other types of uncertainty sets, such as discrete uncorrelated uncertainty, budgeted uncertainty, and ellipsoidal uncertainty. Such results could then be considered in comparison to the problem-specific ones obtained for the Robust Bilevel Selection Problem and the Robust Bilevel Continuous Knapsack Problem, possibly adding to the insights summarized in Section 6.2.

In Section 5.5, we defined the Response Problem, which asks if the leader can enforce a given follower's response, for the Bilevel Minimum Spanning Tree Problem. An analogous Response Problem, as well as its optimization version where the leader tries to enforce the given follower's response as cheaply as possible, can actually be defined and investigated for any bilevel optimization problem. Solving the Response Problem can be helpful for solving the original bilevel problem if, in addition, the number of potential follower's responses is small, such that they can be enumerated. A similar idea was used for Theorem 3.63, where we presented an algorithm that solves the Robust Bilevel Selection Problem under Assumptions (sub) and (bb), and with a constant number of scenarios; see Remark 3.64 for more details. Also note that the structure of the Response Problem is similar to that of an inverse problem (see Section 2.2.2.2), where the leader tries to enforce a given (partial) follower's solution as well, but usually through adjusting the follower's objective function.

We have given several results for partitioned-items bilevel problems in the special case where either the number of leader's items or the number of follower's items is constant; see, e.g., Theorem 3.57, Section 5.6, and [HW25, Section 5]. This usually simplifies the problem since the leader (or the follower) has only a constant number of possible solutions. As a variant of this special case, it could also be interesting to limit the number of items that the leader (or the follower) can select by a constraint instead. Also then, the number of possible leader's (or follower's) solutions is reduced compared to the general setting, but not as drastically as in the case where the total number of leader's (or follower's) items is small. More precisely, if the given maximum number of items to be selected by the leader (or by the follower) is a constant, the number of possible leader's (or follower's) selections is polynomial in the total number of items, but not constant.

While our focus in this thesis was on partitioned-items bilevel problems whose underlying single-level problems are solvable in polynomial time, we consider it a very interesting next step to extend the complexity studies of partitioned-items bilevel problems to $\mathcal{NP}$-hard underlying problems (other than the Knapsack Problem, which has already been studied) whose bilevel versions are likely to be on the second level of the polynomial-time hierarchy.

Besides studying further specific problems, one could also try to obtain more general complexity results. Promising general techniques related to this question have been

presented in [GW25a; GW24; GW25b]. In these works, several problem structures that involve two or three stages of decisions, e.g., interdiction problems, two-stage adjustable robust problems, and recoverable robust problems, are considered. It is shown that they lead to Σ_2^{p}- or Σ_3^{p}-hard problems if the underlying single-level problem is from a class that contains many classical $\mathcal{NP}$-hard combinatorial optimization problems. It would be very interesting to investigate if similar general results can also be obtained for partitioned-items bilevel problems or other classes of bilevel optimization problems (or even robust bilevel optimization problems).

Moreover, also the complexity studies of robust bilevel optimization problems can next be extended to more complex underlying bilevel problems, such as the Bilevel Minimum Spanning Tree Problem and the Bilevel Knapsack Problem. Here, the robust bilevel problems could be on the second or third level of the polynomial-time hierarchy.

Bibliography

[ABV05] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten. "Pseudo-polynomial algorithms for min–max and min–max regret problems". In: *Proceedings of the 5th International Symposium on Operations Research and its Applications*. 2005, pages 171–178 (cited on page 29).

[ABV09] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten. "Min–max and min–max regret versions of combinatorial optimization problems: A survey". In: *European Journal of Operational Research* 197.2 (2009), pages 427–438. DOI: 10.1016/j.ejor.2008.09.012 (cited on pages 33, 34).

[AHJS97] Charles Audet, Pierre Hansen, Brigitte Jaumard, and Gilles Savard. "Links between linear bilevel and mixed 0–1 programming problems". In: *Journal of Optimization Theory and Applications* 93.2 (1997), pages 273–300. DOI: 10.1023/A:1022645805569 (cited on page 23).

[AO00] Ravindra K. Ahuja and James B. Orlin. "A faster algorithm for the inverse spanning tree problem". In: *Journal of Algorithms* 34.1 (2000), pages 177–193. DOI: 10.1006/jagm.1999.1052 (cited on page 26).

[AO01] Ravindra K. Ahuja and James B. Orlin. "Inverse optimization". In: *Operations Research* 49.5 (2001), pages 771–783. DOI: 10.1287/opre.49.5.771.10607 (cited on page 26).

[AS20] Didier Aussel and Anton Svensson. "A short state of the art on multi-leader-follower games". In: *Bilevel Optimization. Advances and Next Challenges*. Edited by Stephan Dempe and Alain Zemkoho. Springer Optimization and Its Applications 161. Cham: Springer, 2020, pages 53–76. DOI: 10.1007/978-3-030-52119-6_3 (cited on page 21).

[Ave01] Igor Averbakh. "On the complexity of a class of combinatorial optimization problems with uncertainty". In: *Mathematical Programming* 90.2 (2001), pages 263–272. DOI: 10.1007/PL00011424 (cited on page 34).

[Bar91] Jonathan F. Bard. "Some properties of the bilevel programming problem". In: *Journal of Optimization Theory and Applications* 68.2 (1991), pages 371–378. DOI: 10.1007/BF00941574 (cited on page 22).

[BB90] Omar Ben-Ayed and Charles E. Blair. "Computational difficulties of bilevel linear programming". In: *Operations Research* 38.3 (1990), pages 556–560. DOI: 10.1287/opre.38.3.556 (cited on page 22).

[BBHH23] Luce Brotcorne, Christoph Buchheim, Dick den Hertog, and Dorothee Henke. "Optimization at the Second Level (Dagstuhl Seminar 22441)". In: *Dagstuhl Reports* 12.10 (2023), pages 207–224. DOI: 10.4230/DagRep.12.10.207 (cited on page 2).

[BEN09] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization*. Princeton Series in Applied Mathematics 28. Princeton and Oxford: Princeton University Press, 2009. DOI: 10.1515/9781400831050 (cited on page 27).

[Ben89] Harold P. Benson. "On the structure and properties of a linear multilevel programming problem". In: *Journal of Optimization Theory and Applications* 60.3 (1989), pages 353–373. DOI: 10.1007/BF00940342 (cited on page 21).

[BFPRT73] Manuel Blum, Robert W. Floyd, Vaughan Pratt, Ronald L. Rivest, and Robert E. Tarjan. "Time bounds for selection". In: *Journal of Computer and System Sciences* 7.4 (1973), pages 448–461. DOI: 10.1016/S0022-0000(73)80033-9 (cited on page 46).

[BH19a] Andreas Björklund and Thore Husfeldt. "Shortest two disjoint paths in polynomial time". In: *SIAM Journal on Computing* 48.6 (2019), pages 1698–1710. DOI: 10.1137/18M1223034 (cited on page 182).

[BH19b] Christoph Buchheim and Dorothee Henke. "The robust bilevel continuous knapsack problem". In: *Proceedings of the 17th Cologne-Twente Workshop on Graphs and Combinatorial Optimization*. 2019, pages 17–20 (cited on page 3).

[BH22] Christoph Buchheim and Dorothee Henke. "The robust bilevel continuous knapsack problem with uncertain coefficients in the follower's objective". In: *Journal of Global Optimization* 83.4 (2022), pages 803–824. DOI: 10.1007/s10898-021-01117-9. arXiv: 1903.02810 [cs.DS] (cited on pages 3, 51, 112, 119, 125, 129, 145, 159, 160).

[BHH21] Christoph Buchheim, Dorothee Henke, and Felix Hommelsheim. "On the complexity of robust bilevel optimization with uncertain follower's objective". In: *Operations Research Letters* 49.5 (2021), pages 703–707. DOI: 10.1016/j.orl.2021.07.009. arXiv: 2105.08378 [math.OC] (cited on pages 2, 5, 36).

[BHH22] Christoph Buchheim, Dorothee Henke, and Felix Hommelsheim. "On the complexity of the bilevel minimum spanning tree problem". In: *Networks* 80.3 (2022), pages 338–355. DOI: 10.1002/net.22111. arXiv: 2012.12770 [cs.DS] (cited on pages 3, 161).

[BHI22] Christoph Buchheim, Dorothee Henke, and Jannik Irmai. "The stochastic bilevel continuous knapsack problem with uncertain follower's objective". In: *Journal of Optimization Theory and Applications* 194.2 (2022), pages 521–542. DOI: 10.1007/s10957-022-02037-8. arXiv: 2108.12303 [cs.DS] (cited on pages 3, 27, 36, 45, 119, 160).

[BHM09] Luce Brotcorne, Saïd Hanafi, and Raïd Mansi. "A dynamic programming algorithm for the bilevel knapsack problem". In: *Operations Research Letters* 37.3 (2009), pages 215–218. DOI: 10.1016/j.orl.2009.01.007 (cited on page 122).

[BHM13] Luce Brotcorne, Saïd Hanafi, and Raïd Mansi. "One-level reformulation of the bilevel knapsack problem using dynamic programming". In: *Discrete Optimization* 10.1 (2013), pages 1–10. DOI: 10.1016/j.disopt.2012.09.001 (cited on page 122).

[BK18] Christoph Buchheim and Jannis Kurtz. "Robust combinatorial optimization under convex and discrete cost uncertainty". In: *EURO Journal on Computational Optimization* 6.3 (2018), pages 211–238. DOI: 10.1007/s13675-018-0103-0 (cited on pages 27–31, 33, 43).

[BL11] John R. Birge and François Louveaux. *Introduction to Stochastic Programming*. 2nd edition. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2011. DOI: 10.1007/978-1-4614-0237-4 (cited on page 27).

[Bla+24] Jannis Blauth, Stephan Held, Dirk Müller, Niklas Schlomberg, Vera Traub, Thorben Tröbst, and Jens Vygen. "Vehicle routing with time-dependent travel times: Theory, practice, and benchmarks". In: *Discrete Optimization* 53, 100848 (2024). DOI: 10.1016/j.disopt.2024.100848 (cited on page 133).

[Bla92] Charles Blair. "The computational complexity of multi-level linear programs". In: *Annals of Operations Research* 34.1 (1992), pages 13–19. DOI: 10.1007/BF02098170 (cited on page 22).

[BLS23] Yasmine Beck, Ivana Ljubić, and Martin Schmidt. "A survey on bilevel optimization under uncertainty". In: *European Journal of Operational Research* 311.2 (2023), pages 401–426. DOI: 10.1016/j.ejor.2023.01.008 (cited on page 36).

[BM73] Jerome Bracken and James T. McGill. "Mathematical programs with optimization problems in the constraints". In: *Operations Research* 21.1 (1973), pages 37–44. DOI: 10.1287/opre.21.1.37 (cited on page 22).

[BP89] Marshall Bern and Paul Plassmann. "The Steiner problem with edge lengths 1 and 2". In: *Information Processing Letters* 32.4 (1989), pages 171–176. DOI: 10.1016/0020-0190(89)90039-2 (cited on page 178).

[BPP16] Behdad Beheshti, Oleg A. Prokopyev, and Eduardo L. Pasiliao. "Exact solution approaches for bilevel assignment problems". In: *Computational Optimization and Applications* 64.1 (2016), pages 215–242. DOI: 10.1007/s10589-015-9799-4 (cited on page 25).

[BS04] Dimitris Bertsimas and Melvyn Sim. "The price of robustness". In: *Operations Research* 52.1 (2004), pages 35–53. DOI: 10.1287/opre.1030.0065 (cited on page 30).

[BS20] Toni Böhnlein and Oliver Schaudt. "On the complexity of Stackelberg matroid pricing problems". In: *Combinatorial Algorithms. 31st International Workshop, IWOCA 2020, Bordeaux, France, June 8–10, 2020, Proceedings*. Edited by Leszek Gąsieniec, Ralf Klasing, and Tomasz Radzik. Lecture Notes in Computer Science 12126. 2020, pages 83–96. DOI: 10.1007/978-3-030-48966-3_7 (cited on page 198).

[BS21] Yasmine Beck and Martin Schmidt. *A Gentle and Incomplete Introduction to Bilevel Optimization*. Lecture notes. 2021. Optimization Online: 17182 (cited on pages 18, 21, 22).

[Buc23] Christoph Buchheim. “Bilevel linear optimization belongs to NP and admits polynomial-size KKT-based reformulations”. In: *Operations Research Letters* 51.6 (2023), pages 618–622. DOI: 10.1016/j.orl.2023.10.006 (cited on pages 23, 38).

[Buc43] Robert C. Buck. “Partition of space”. In: *The American Mathematical Monthly* 50.9 (1943), pages 541–544. DOI: 10.1080/00029890.1943.11991447 (cited on page 157).

[Car23] Margarida Carvalho. “Bilevel knapsack problems”. In: *Encyclopedia of Optimization*. Edited by Panos M. Pardalos and Oleg A. Prokopyev. Cham: Springer, 2023, pages 1–7. DOI: 10.1007/978-3-030-54621-2_739-1 (cited on page 122).

[CC08] Miroslav Chlebík and Janka Chlebíková. “The Steiner tree problem on graphs: Inapproximability results”. In: *Theoretical Computer Science* 406.3 (2008), pages 207–214. DOI: 10.1016/j.tcs.2008.06.046 (cited on pages 167, 176–178).

[CCLW14] Alberto Caprara, Margarida Carvalho, Andrea Lodi, and Gerhard J. Woeginger. “A study on the computational complexity of the bilevel knapsack problem”. In: *SIAM Journal on Optimization* 24.2 (2014), pages 823–838. DOI: 10.1137/130906593 (cited on pages 121, 122, 127, 197, 205).

[CCLW16] Alberto Caprara, Margarida Carvalho, Andrea Lodi, and Gerhard J. Woeginger. “Bilevel knapsack with interdiction constraints”. In: *INFORMS Journal on Computing* 28.2 (2016), pages 319–333. DOI: 10.1287/ijoc.2015.0676 (cited on page 122).

[CDYZ08] Mao-Cheng Cai, Cees W. Duin, Xiaoguang Yang, and Jianzhong Zhang. “The partial inverse minimum spanning tree problem when weight increase is forbidden”. In: *European Journal of Operational Research* 188.2 (2008), pages 348–353. DOI: 10.1016/j.ejor.2007.04.031 (cited on page 26).

[CLM18] Margarida Carvalho, Andrea Lodi, and Patrice Marcotte. “A polynomial algorithm for a continuous bilevel knapsack problem”. In: *Operations Research Letters* 46.2 (2018), pages 185–188. DOI: 10.1016/j.orl.2017.12.009 (cited on page 122).

[CMS07] Benoît Colson, Patrice Marcotte, and Gilles Savard. “An overview of bilevel optimization”. In: *Annals of Operations Research* 153.1 (2007), pages 235–256. DOI: 10.1007/s10479-007-0176-2 (cited on pages 18, 22).

[Coo71] Stephen A. Cook. “The complexity of theorem-proving procedures”. In: *Proceedings of the third Annual ACM Symposium on Theory of Computing*. 1971, pages 151–158. DOI: 10.1145/800157.805047 (cited on page 8).

[CS20] Matthias Claus and Maximilian Simmoteit. “A note on Σ_2^{p}-completeness of a robust binary linear program with binary uncertainty set”. In: *Operations Research Letters* 48.5 (2020), pages 594–598. DOI: 10.1016/j.orl.2020.07.006 (cited on page 33).

[CWZ22] Lin Chen, Xiaoyu Wu, and Guochuan Zhang. "Approximation algorithms for interdiction problem with packing constraints". In: *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*. 2022, 39:1–39:19. DOI: 10.4230/LIPIcs.ICALP.2022.39. arXiv: 2204.11106 [cs.DS] (cited on page 121).

[Cyg+15] Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. 1st edition. Cham: Springer, 2015. DOI: 10.1007/978-3-319-21275-3 (cited on page 14).

[CZ13] Lin Chen and Guochuan Zhang. "Approximation algorithms for a bi-level knapsack problem". In: *Theoretical Computer Science* 497 (2013), pages 1–12. DOI: 10.1016/j.tcs.2012.08.008 (cited on page 122).

[Dan57] George B. Dantzig. "Discrete-variable extremum problems". In: *Operations Research* 5.2 (1957), pages 266–277. DOI: 10.1287/opre.5.2.266 (cited on page 113).

[Dem20] Stephan Dempe. "Bilevel Optimization: Theory, Algorithms, Applications and a Bibliography". In: *Bilevel Optimization. Advances and Next Challenges*. Edited by Stephan Dempe and Alain Zemkoho. Springer Optimization and Its Applications 161. Cham: Springer, 2020, pages 581–672. DOI: 10.1007/978-3-030-52119-6_20 (cited on page 22).

[DeN11] Scott DeNegre. "Interdiction and discrete bilevel linear programming". PhD thesis. Lehigh University, Bethlehem, Pennsylvania, 2011 (cited on pages 26, 121, 122, 205).

[Den98] Xiaotie Deng. "Complexity Issues in Bilevel Linear Programming". In: *Multilevel Optimization: Algorithms and Applications*. Edited by Athanasios Migdalas, Panos M. Pardalos, and Peter Värbrand. Nonconvex Optimization and Its Applications 20. Boston: Springer, 1998, pages 149–164. DOI: 10.1007/978-1-4613-0307-7_6 (cited on page 22).

[DKPK15] Stephan Dempe, Vyacheslav Kalashnikov, Gerardo A. Pérez-Valdés, and Nataliya Kalashnykova. *Bilevel Programming Problems. Theory, Algorithms and Applications to Energy Networks*. 1st edition. Berlin, Heidelberg: Springer, 2015. DOI: 10.1007/978-3-662-45827-3 (cited on pages 18, 22, 119).

[DKW98] Tibor Dudás, Bettina Klinz, and Gerhard J. Woeginger. "The computational complexity of multi-level bottleneck programming problems". In: *Multilevel Optimization: Algorithms and Applications*. Edited by Athanasios Migdalas, Panos M. Pardalos, and Peter Värbrand. Nonconvex Optimization and Its Applications 20. Boston: Springer, 1998, pages 165–179. DOI: 10.1007/978-1-4613-0307-7_7 (cited on pages 22, 23).

[DM14] Marc Demange and Jérôme Monnot. "An introduction to inverse combinatorial problems". In: *Paradigms of Combinatorial Optimization: Problems and New Approaches*. Edited by Vangelis Th. Paschos. 2nd edition. John Wiley & Sons, Inc., 2014, pages 547–586. DOI: 10.1002/9781119005353.ch17 (cited on page 26).

[DMM03] Mauro Dell'Amico, Francesco Maffioli, and Federico Malucelli. "The base-matroid and inverse combinatorial optimization problems". In: *Discrete Applied Mathematics* 128.2–3 (2003), pages 337–353. DOI: 10.1016/S0166-218X(02)00498-5 (cited on page 26).

[DR00] Stephan Dempe and Katrin Richter. "Bilevel programming with knapsack constraints". In: *Central European Journal of Operations Research* 8.2 (2000), pages 93–107 (cited on pages 121, 122).

[DS20] Federico Della Croce and Rosario Scatamacchia. "An exact approach for the bilevel knapsack problem with interdiction constraints and extensions". In: *Mathematical Programming* 183.1–2 (2020), pages 249–281. DOI: 10.1007/s10107-020-01482-5 (cited on page 122).

[DW10] Vladimir G. Deineko and Gerhard J. Woeginger. "Pinpointing the complexity of the interval min–max regret knapsack problem". In: *Discrete Optimization* 7.4 (2010), pages 191–196. DOI: 10.1016/j.disopt.2010.03.008 (cited on page 33).

[DW71] Stuart E. Dreyfus and Robert A. Wagner. "The Steiner problem in graphs". In: *Networks* 1.3 (1971), pages 195–207. DOI: 10.1002/net.3230010302 (cited on page 185).

[Emd90] Peter van Emde Boas. "Machine Models and Simulation". In: *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*. Edited by Jan van Leeuwen. Amsterdam: Elsevier, 1990, pages 1–66 (cited on page 6).

[Fan11] Diana Fanghänel. "Optimality conditions for a bilevel matroid problem". In: *Journal of Combinatorial Optimization* 22.4 (2011), pages 594–608. DOI: 10.1007/s10878-010-9307-9 (cited on page 198).

[FLMS19] Matteo Fischetti, Ivana Ljubić, Michele Monaci, and Markus Sinnl. "Interdiction games and monotonicity, with application to knapsack problems". In: *INFORMS Journal on Computing* 31.2 (2019), pages 390–410. DOI: 10.1287/ijoc.2018.0831 (cited on page 122).

[FMW22] Dennis Fischer, Komal Muluk, and Gerhard J. Woeginger. "A note on the complexity of the bilevel bottleneck assignment problem". In: *4OR – A Quarterly Journal of Operations Research* 20.4 (2022), pages 713–718. DOI: 10.1007/s10288-021-00499-6 (cited on pages 25, 167, 197).

[FS99] Greg N. Frederickson and Roberto Solis-Oba. "Increasing the weight of minimum spanning trees". In: *Journal of Algorithms* 33.2 (1999), pages 244–266. DOI: 10.1006/jagm.1999.1026 (cited on page 26).

[FW20] Dennis Fischer and Gerhard J. Woeginger. "A faster algorithm for the continuous bilevel knapsack problem". In: *Operations Research Letters* 48.6 (2020), pages 784–786. DOI: 10.1016/j.orl.2020.09.007 (cited on page 122).

[Gas02] Elisabeth Gassner. "Maximal spannende Baumprobleme mit einer Hierarchie von zwei Entscheidungsträgern". Diploma thesis. Graz University of Technology, 2002 (cited on pages 166–168, 189–191).

[Gas09] Elisabeth Gassner. *The computational complexity of continuous-discrete bilevel network problems*. Technical report. Graz University of Technology, 2009. URL: https://www.math.tugraz.at/fosp/pdfs/tugraz_main_0150.pdf (cited on page 26).

[GH24] Marc Goerigk and Michael Hartisch. *An Introduction to Robust Combinatorial Optimization. Concepts, Models and Algorithms for Decision Making under Uncertainty*. International Series in Operation Research & Management Science 361. Cham: Springer, 2024. DOI: 10.1007/978-3-031-61261-9 (cited on pages 27, 30, 33, 201).

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. A Series of Books in the Mathematical Sciences. New York: W. H. Freeman and Company, 1979 (cited on pages 6, 9, 15, 99).

[GK09] Elisabeth Gassner and Bettina Klinz. "The computational complexity of bilevel assignment problems". In: *4OR – A Quarterly Journal of Operations Research* 7.4 (2009), pages 379–394. DOI: 10.1007/s10288-009-0098-8 (cited on pages 25, 167).

[GKST25] Marc Goerigk, Jannis Kurtz, Martin Schmidt, and Johannes Thürauf. "Connections between robust and bilevel optimization". In: *Open Journal of Mathematical Optimization* 6, 2 (2025). DOI: 10.5802/ojmo.38 (cited on page 33).

[GLS93] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. 2nd edition. Algorithms and Combinatorics 2. Berlin, Heidelberg: Springer, 1993. DOI: 10.1007/978-3-642-78240-4 (cited on pages 6, 11, 12).

[GLW23] Marc Goerigk, Stefan Lendl, and Lasse Wulf. "On the complexity of robust multi-stage problems with discrete recourse". In: *Discrete Applied Mathematics* 343 (2023), pages 355–370. DOI: 10.1016/j.dam.2023.10.018 (cited on page 33).

[GW24] Christoph Grüne and Lasse Wulf. *On the complexity of recoverable robust optimization in the polynomial hierarchy*. 2024. arXiv: 2411.18590 [cs.CC] (cited on pages 17, 208).

[GW25a] Christoph Grüne and Lasse Wulf. "Completeness in the polynomial hierarchy for many natural problems in bilevel and robust optimization". In: *Integer Programming and Combinatorial Optimization (IPCO 2025)*. Lecture Notes in Computer Science 15620. 2025, pages 256–269. DOI: 10.1007/978-3-031-93112-3_19. arXiv: 2311.10540v3 [cs.CC] (cited on pages 17, 26, 208).

[GW25b] Christoph Grüne and Lasse Wulf. *The complexity of blocking all solutions*. 2025. arXiv: 2502.05348 [cs.CC] (cited on pages 17, 26, 208).

[Har21] Kristina Hartmann. "The robust bilevel selection problem". Master's thesis. TU Dortmund University, 2021 (cited on pages 2, 50, 96, 102).

[Hen25] Dorothee Henke. "The robust bilevel selection problem". In: *Open Journal of Mathematical Optimization* 6, 4 (2025), pages 1–35. DOI: 10.5802/ojmo.40. arXiv: 2401.03951 [math.OC] (cited on pages 2, 3, 45, 50, 101, 112, 119, 129, 145).

[Her89] John Hershberger. "Finding the upper envelope of n line segments in $\mathcal{O}(n \log n)$ time". In: *Information Processing Letters* 33.4 (1989), pages 169–174. DOI: 10.1016/0020-0190(89)90136-1 (cited on pages 81, 87, 94, 132, 133, 139, 151).

[Heu04] Clemens Heuberger. "Inverse combinatorial optimization: A survey on problems, methods, and results". In: *Journal of Combinatorial Optimization* 8.3 (2004), pages 329–361. DOI: 10.1023/B:JOCO.0000038914.26975.9b (cited on page 26).

[HJS92] Pierre Hansen, Brigitte Jaumard, and Gilles Savard. "New branch-and-bound rules for linear bilevel programming". In: *SIAM Journal on Scientific and Statistical Computing* 13.5 (1992), pages 1194–1217. DOI: 10.1137/0913069 (cited on page 22).

[HLST25a] Dorothee Henke, Henri Lefebvre, Martin Schmidt, and Johannes Thürauf. "On coupling constraints in linear bilevel optimization". In: *Optimization Letters* 19.3 (2025), pages 689–697. DOI: 10.1007/s11590-024-02156-3. arXiv: 2402.12191 [math.OC] (cited on pages 3, 21, 23).

[HLST25b] Dorothee Henke, Henri Lefebvre, Martin Schmidt, and Johannes Thürauf. *On coupling constraints in pessimistic linear bilevel optimization*. 2025. arXiv: 2503.01563 [math.OC] (cited on pages 3, 21).

[Hoe08] Stan van Hoesel. "An overview of Stackelberg pricing in networks". In: *European Journal of Operational Research* 189.3 (2008), pages 1393–1402. DOI: 10.1016/j.ejor.2006.08.064 (cited on page 25).

[HS86] Sergiu Hart and Micha Sharir. "Nonlinearity of Davenport–Schinzel sequences and of generalized path compression schemes". In: *Combinatorica* 6.2 (1986), pages 151–177. DOI: 10.1007/BF02579170 (cited on page 133).

[Hüg20] Maja Hügging. "The bilevel continuous knapsack problem with uncertain weights". Master's thesis. TU Dortmund University, 2020 (cited on pages 159, 206).

[HW25] Dorothee Henke and Lasse Wulf. "On the complexity of the bilevel shortest path problem". In: *Networks* 86.4 (2025), pages 428–445. DOI: 10.1002/net.70002. arXiv: 2412.21134 [cs.DS] (cited on pages 3, 24, 197, 207).

[Irm21] Jannik Irmai. "The stochastic bilevel selection problem". Master's thesis. TU Dortmund University, 2021 (cited on page 45).

[Isr99] Eitan Israeli. "System interdiction and defense". PhD thesis. Naval Postgraduate School, Moneterey, California, 1999 (cited on page 26).

[Jai01] Kamal Jain. "A factor 2 approximation algorithm for the generalized Steiner network problem". In: *Combinatorica* 21.1 (2001), pages 39–60. DOI: 10.1007/s004930170004 (cited on page 176).

[Jer85] Robert G. Jeroslow. "The polynomial hierarchy and a simple model for competitive analysis". In: *Mathematical Programming* 32.2 (1985), pages 146–164. DOI: 10.1007/BF01586088 (cited on pages 22, 23, 39–41).

[Joh11] Berit Johannes. "New classes of complete problems for the second level of the polynomial hierarchy". PhD thesis. Technische Universität Berlin, 2011 (cited on pages 15, 23).

[Kar72] Richard M. Karp. "Reducibility among combinatorial problems". In: *Complexity of Computer Computations*. Edited by Raymond E. Miller, James W. Thatcher, and Jean D. Bohlinger. 1972, pages 85–103. DOI: 10.1007/978-1-4684-2001-2_9 (cited on page 112).

[KK12] Tamás Kis and András Kovács. "On bilevel machine scheduling problems". In: *OR Spectrum* 34.1 (2012), pages 43–68. DOI: 10.1007/s00291-010-0219-y (cited on page 26).

[Kle21] Thomas Kleinert. "Algorithms for mixed-integer bilevel problems with convex followers". PhD thesis. Friedrich-Alexander-Universität Erlangen-Nürnberg, 2021 (cited on page 21).

[KLLS21] Thomas Kleinert, Martine Labbé, Ivana Ljubić, and Martin Schmidt. "A survey on mixed-integer programming techniques in bilevel optimization". In: *EURO Journal on Computational Optimization* 9, 100007 (2021). DOI: 10.1016/j.ejco.2021.100007 (cited on page 22).

[KPP04] Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack Problems*. 1st edition. Berlin, Heidelberg: Springer, 2004. DOI: 10.1007/978-3-540-24777-7 (cited on pages 73, 112, 146).

[KPS90] Bernhard Korte, Hans Jürgen Prömel, and Angelika Steger. "Steiner trees in VLSI-layout". In: *Paths, flows, and VLSI-layout*. Edited by Bernhard Korte, Laszlo Lovasz, Hans Jürgen Prömel, and Alexander Schrijver. Algorithms and Combinatorics 9. Berlin, Heidelberg: Springer, 1990, pages 185–214 (cited on page 178).

[Kru56] Joseph B. Kruskal Jr. "On the shortest spanning subtree of a graph and the traveling salesman problem". In: *Proceedings of the American Mathematical Society* 7.1 (1956), pages 48–50. DOI: 10.2307/2033241 (cited on pages 163, 164).

[KV18] Bernhard Korte and Jens Vygen. *Combinatorial Optimization. Theory and Algorithms*. 6th edition. Algorithms and Combinatorics 21. Berlin, Heidelberg: Springer, 2018. DOI: 10.1007/978-3-662-56039-6 (cited on pages 6, 12–14, 32, 38, 46, 112, 113, 162, 163, 198).

[KW96] John K. Karlof and Wei Wang. "Bilevel programming applied to the flow shop scheduling problem". In: *Computers & Operations Research* 23.5 (1996), pages 443–451. DOI: 10.1016/0305-0548(95)00034-8 (cited on page 26).

[KY97] Panos Kouvelis and Gang Yu. *Robust Discrete Optimization and Its Applications*. Nonconvex Optimization and Its Applications 14. New York: Springer, 1997. DOI: 10.1007/978-1-4757-2620-6 (cited on pages 27, 33, 34).

[KZ09] Adam Kasperski and Paweł Zieliński. "A randomized algorithm for the min–max selecting items problem with uncertain weights". In: *Annals of Operations Research* 172.1 (2009), pages 221–230. DOI: 10.1007/s10479-009-0564-x (cited on page 34).

[KZ16] Adam Kasperski and Paweł Zieliński. "Robust discrete optimization under discrete and interval uncertainty: A survey". In: *Robustness Analysis in Decision Aiding, Optimization, and Analytics*. Edited by Michael Doumpos, Constantin Zopounidis, and Evangelos Grigoroudis. International Series in Operations Research & Management Science 241. Cham: Springer, 2016, pages 113–143. DOI: 10.1007/978-3-319-33121-8_6 (cited on page 33).

[LC93] Kao-Chêng Lin and Maw-Sheng Chern. "The most vital edges in the minimum spanning tree problem". In: *Information Processing Letters* 45.1 (1993), pages 25–31. DOI: 10.1016/0020-0190(93)90247-7 (cited on page 26).

[LMS98] Martine Labbé, Patrice Marcotte, and Gilles Savard. "A bilevel model of taxation and its application to optimal highway pricing". In: *Management Science* 44.12-part-1 (1998), pages 1608–1622. DOI: 10.1287/mnsc.44.12.1608 (cited on page 25).

[LPP21] Martine Labbé, Miguel A. Pozo, and Justo Puerto. "Computational comparisons of different formulations for the Stackelberg minimum spanning tree game". In: *International Transactions in Operational Research* 28.1 (2021), pages 48–69. DOI: 10.1111/itor.12680 (cited on pages 25, 167).

[LS95] Yi-Hsin Liu and Thomas H. Spencer. "Solving a bilevel linear program when the inner decision maker controls few variables". In: *European Journal of Operational Research* 81.3 (1995), pages 644–651. DOI: 10.1016/0377-2217(94)00005-W (cited on page 22).

[LSW23] Leo Liberti, Sebastian Sager, and Angelika Wiegele. "Mixed-integer Nonlinear Optimization: A Hatchery for Modern Mathematics". In: *Oberwolfach Reports* 20.3 (2023), pages 1953–2016. DOI: 10.4171/OWR/2023/35 (cited on page 2).

[LV13] Martine Labbé and Alessia Violin. "Bilevel programming and price setting problems". In: *4OR – A Quarterly Journal of Operations Research* 11.1 (2013), pages 1–30. DOI: 10.1007/s10479-015-2016-0 (cited on page 25).

[LZD18] Xianyue Li, Zhao Zhang, and Ding-Zhu Du. "Partial inverse maximum spanning tree in which weight can only be decreased under l_p-norm". In: *Journal of Global Optimization* 70.3 (2018), pages 677–685. DOI: 10.1007/s10898-017-0554-5 (cited on page 26).

[LZL16] Shuangshuang Li, Zhao Zhang, and Hong-Jian Lai. "Algorithm for constraint partial inverse matroid problem with weight increase forbidden". In: *Theoretical Computer Science* 640 (2016), pages 119–124. DOI: 10.1016/j.tcs.2016.06.002 (cited on page 26).

[MACH12] Raïd Mansi, Cláudio Alves, José Manuel Valério de Carvalho, and Saïd Hanafi. "An exact algorithm for bilevel 0-1 knapsack problems". In: *Mathematical Problems in Engineering*, 504713 (2012). DOI: 10.1155/2012/504713 (cited on pages 121, 122, 127, 146).

[MD06] Ayalew Getachew Mersha and Stephan Dempe. "Linear bilevel programming with upper level constraints depending on the lower level solution". In: *Applied Mathematics and Computation* 180.1 (2006), pages 247–254. DOI: 10.1016/j.amc.2005.11.134 (cited on page 21).

[Möh89] Rolf H. Möhring. "Computationally tractable classes of ordered sets". In: *Algorithms and Order*. Edited by Ivan Rival. NATO ASI Series C: Mathematical and Physical Sciences 255. Dordrecht: Kluwer Academic Publishers, 1989, pages 105–193. DOI: 10.1007/978-94-009-2639-4_4 (cited on page 71).

[MPS13] Michele Monaci, Ulrich Pferschy, and Paolo Serafini. "Exact solution of the robust knapsack problem". In: *Computers & Operations Research* 40.11 (2013), pages 2625–2631. DOI: 10.1016/j.cor.2013.05.005 (cited on page 34).

[MS05] Patrice Marcotte and Gilles Savard. "Bilevel programming: A combinatorial perspective". In: *Graph Theory and Combinatorial Optimization*. Edited by David Avis, Alain Hertz, and Odile Marcotte. 1st edition. Boston: Springer, 2005, pages 191–217. DOI: 10.1007/0-387-25592-3_7 (cited on pages 22, 23, 25).

[MS72] Albert R. Meyer and Larry J. Stockmeyer. "The equivalence problem for regular expressions with squaring requires exponential space". In: *13th Annual Symposium on Switching and Automata Theory (SWAT 1972)*. 1972, pages 125–129. DOI: 10.1109/SWAT.1972.29 (cited on pages 15, 16).

[Nik10] Evdokia Nikolova. *Approximation algorithms for offline risk-averse combinatorial optimization*. Technical report. Massachusetts Institute of Technology, 2010. URL: https://users.ece.utexas.edu/~nikolova/papers/journal-risk-preprint.pdf (cited on page 34).

[ÖPS10] Osman Y. Özaltın, Oleg A. Prokopyev, and Andrew J. Schaefer. "The bilevel knapsack problem with stochastic right-hand sides". In: *Operations Research Letters* 38.4 (2010), pages 328–333. DOI: 10.1016/j.orl.2010.04.005 (cited on page 122).

[PNP19] Ulrich Pferschy, Gaia Nicosia, and Andrea Pacifici. "A Stackelberg knapsack game with weight control". In: *Theoretical Computer Science* 799 (2019), pages 149–159. DOI: 10.1016/j.tcs.2019.10.007 (cited on page 122).

[PNPS21] Ulrich Pferschy, Gaia Nicosia, Andrea Pacifici, and Joachim Schauer. "On the Stackelberg knapsack game". In: *European Journal of Operational Research* 291.1 (2021), pages 18–31. DOI: 10.1016/j.ejor.2020.09.007 (cited on page 122).

[QK15] Xian Qiu and Walter Kern. "Improved approximation algorithms for a bilevel knapsack problem". In: *Theoretical Computer Science* 595 (2015), pages 120–129. DOI: 10.1016/j.tcs.2015.06.027 (cited on page 122).

[RS90] Neil Robertson and Paul D. Seymour. "An outline of a disjoint paths algorithm". In: *Paths, flows, and VLSI-layout*. Edited by Bernhard Korte, Laszlo Lovasz, Hans Jürgen Prömel, and Alexander Schrijver. Algorithms and Combinatorics 9. Berlin, Heidelberg: Springer, 1990, pages 267–292 (cited on page 181).

[RS95] Neil Robertson and Paul D. Seymour. "Graph minors. XIII. The disjoint paths problem". In: *Journal of Combinatorial Theory, Series B* 63.1 (1995), pages 65–110. DOI: 10.1006/jctb.1995.1006 (cited on page 181).

[SAO99] P. T. Sokkalingam, Ravindra K. Ahuja, and James B. Orlin. "Solving inverse spanning tree problems through network flow techniques". In: *Operations Research* 47.2 (1999), pages 291–298. DOI: 10.1287/opre.47.2.291 (cited on page 26).

[Sch03] Alexander Schrijver. *Combinatorial Optimization. Polyhedra and Efficiency.* 1st edition. Algorithms and Combinatorics 24. Berlin, Heidelberg: Springer, 2003 (cited on page 6).

[Shi21] Xueyu Shi. "Exploiting Local Optimality and Strong Inequalities for Solving Bilevel Combinatorial and Submodular Optimization Problems". PhD thesis. University of Pittsburgh, 2021 (cited on page 167).

[SM73] Larry J. Stockmeyer and Albert R. Meyer. "Word problems requiring exponential time: Preliminary report". In: (1973), pages 1–9. DOI: 10.1145/800125.804029 (cited on page 15).

[SMD18] Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. "A review on bilevel optimization: From classical to evolutionary approaches and applications". In: *IEEE Transactions on Evolutionary Computation* 22.2 (2018), pages 276–295. DOI: 10.1109/TEVC.2017.2712906 (cited on page 22).

[Soy73] Allen L. Soyster. "Technical Note – Convex Programming with Set-Inclusive Constraints and Applications to Inexact Linear Programming". In: *Operations Research* 21.5 (1973), pages 1154–1157. DOI: 10.1287/opre.21.5.1154 (cited on page 27).

[SPG13] J. Cole Smith, Mike Prince, and Joseph Geunes. "Modern network interdiction problems and algorithms". In: *Handbook of Combinatorial Optimization.* Edited by Panos M. Pardalos, Ding-Zhu Du, and Ronald L. Graham. New York: Springer, 2013, pages 1949–1987. DOI: 10.1007/978-1-4419-7997-1_61 (cited on page 26).

[SPR23] Xueyu Shi, Oleg A. Prokopyev, and Ted K. Ralphs. "Mixed integer bilevel optimization with k-optimal follower: a hierarchy of bounds". In: *Mathematical Programming Computation* 15.1 (2023), pages 1–51. DOI: 10.1007/s12532-022-00227-z (cited on page 167).

[SS83] Hanif D. Sherali and Allen L. Soyster. "Preemptive and nonpreemptive multi-objective programming: Relationships and counterexamples". In: *Journal of Optimization Theory and Applications* 39.2 (1983), pages 173–186. DOI: 10.1007/BF00934527 (cited on page 38).

[SST16] Alan Shuchat, Randy Shull, and Ann N. Trenk. "Unit interval orders of open and closed intervals". In: *Order* 33.1 (2016), pages 85–99. DOI: 10.1007/s11083-015-9354-z (cited on page 72).

[Sta34] Heinrich von Stackelberg. *Marktform und Gleichgewicht.* Wien: Springer, 1934 (cited on page 22). *Market Structure and Equilibrium.* Translated by Damien Bazin, Lynn Urch, and Rowland Hill. 1st edition. Berlin, Heidelberg: Springer, 2011. DOI: 10.1007/978-3-642-12586-7.

[Sto76] Larry J. Stockmeyer. "The polynomial-time hierarchy". In: *Theoretical Computer Science* 3.1 (1976), pages 1–22. DOI: 10.1016/0304-3975(76)90061-X (cited on page 15).

[SU02a] Marcus Schaefer and Christopher Umans. "Completeness in the polynomial-time hierarchy: A compendium". In: *ACM SIGACT News* 33.3 (2002), pages 32–49. DOI: 10.1145/582475.582484 (cited on page 17). *Updated version*. 2008. URL: http://ovid.cs.depaul.edu/documents/phcom.pdf.

[SU02b] Marcus Schaefer and Christopher Umans. "Completeness in the polynomial-time hierarchy: Part II". In: *ACM SIGACT News* 33.4 (2002), pages 22–36. DOI: 10.1145/601819.601829 (cited on page 17).

[SZP19] Xueyu Shi, Bo Zeng, and Oleg A. Prokopyev. "On bilevel minimum and bottleneck spanning tree problems". In: *Networks* 74.3 (2019), pages 251–273. DOI: 10.1002/net.21881 (cited on pages 166–169, 173, 177, 188, 190, 191, 195).

[Tod91] Seinosuke Toda. "PP is as hard as the polynomial-time hierarchy". In: *SIAM Journal on Computing* 20.5 (1991), pages 865–877. DOI: 10.1137/0220053 (cited on page 160).

[VSJ94] Luis Vicente, Gilles Savard, and Joaquim Júdice. "Descent approaches for quadratic bilevel programming". In: *Journal of Optimization Theory and Applications* 81.2 (1994), pages 379–399. DOI: 10.1007/BF02191670 (cited on page 22).

[VSJ96] Luis Vicente, Gilles Savard, and Joaquim Júdice. "Discrete linear bilevel programming problem". In: *Journal of Optimization Theory and Applications* 89.3 (1996), pages 597–614. DOI: 10.1007/BF02275351 (cited on pages 20, 23).

[WF25a] Noah Weninger and Ricardo Fukasawa. "A fast combinatorial algorithm for the bilevel knapsack problem with interdiction constraints". In: *Mathematical Programming* 210.1–2 (2025), pages 847–879. DOI: 10.1007/s10107-024-02133-9 (cited on page 122).

[WF25b] Noah Weninger and Ricardo Fukasawa. "Interdiction of minimum spanning trees and other matroid bases". In: *Mathematical Programming* (2025). DOI: 10.1007/s10107-025-02273-6 (cited on pages 26, 198).

[Woe03] Gerhard J. Woeginger. "On the approximability of average completion time scheduling under precedence constraints". In: *Discrete Applied Mathematics* 131.1 (2003), pages 237–252. DOI: 10.1016/S0166-218X(02)00427-4 (cited on pages 73, 81, 135).

[Woe21] Gerhard J. Woeginger. "The trouble with the second quantifier". In: *4OR – A Quarterly Journal of Operations Research* 19.2 (2021), pages 157–181. DOI: 10.1007/s10288-021-00477-y (cited on pages 15, 17, 23, 33).

[Wra76] Celia Wrathall. "Complete sets and the polynomial-time hierarchy". In: *Theoretical Computer Science* 3.1 (1976), pages 23–33. DOI: 10.1016/0304-3975(76)90062-1 (cited on pages 15–17).

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge: Cambridge University Press, 2011. DOI: 10.1017/CBO9780511921735 (cited on pages 13, 14).

[Yu96] Gang Yu. "On the max-min 0-1 knapsack problem with robust optimization applications". In: *Operations Research* 44.2 (1996), pages 407–415. DOI: 10.1287/opre.44.2.407 (cited on page 34).

[ZLLD16] Zhao Zhang, Shuangshuang Li, Hong-Jian Lai, and Ding-Zhu Du. "Algorithms for the partial inverse matroid problem in which weights can only be increased". In: *Journal of Global Optimization* 65.4 (2016), pages 801–811. DOI: 10.1007/s10898-016-0412-x (cited on page 26).

[ZXM97] Jianzhong Zhang, Shaoji Xu, and Zhongfan Ma. "An algorithm for inverse minimum spanning tree problem". In: *Optimization Methods and Software* 8.1 (1997), pages 69–84. DOI: 10.1080/10556789708805666 (cited on page 26).