Introduction

Combinatorial optimization is an important tool for solving optimization problems from industry like vehicle routing, network design or production scheduling. To define such an optimization problem, data concerning the cost, the constraints on the solutions or the topology of the networks are assumed to be known. However, these data can often only be estimated based on imprecise measuring methods or predictions of future events (development of the stock markets, change of weather conditions, variations in traffic volume). In several applications, average values from historical data adjusted by some anticipated changes are used to determine the problem setting.

An attractive approach for dealing with these variations in data is to include different data sets into the optimization process. Many researchers have selected a scenario approach, where each scenario represents a reasonable data set. Depending on the considered setting and the available information, such a set of data sets is equipped with a probability distribution to reflect the likelihoods of the scenarios.

There are two major trends in dealing with uncertainty given by a scenario set: stochastic programming and robust optimization. The goal in *stochastic programming* is to find some solution that is feasible in almost all considered scenarios and minimizes some stochastic function like the expected cost. This approach is only applicable if a probability distribution of the scenario set is known or can be estimated. Minimizing the expected cost is reasonable if the process is repeated several times and the same solution is chosen in each iteration. Furthermore, the approach assumes some flexibility in the realization process, since a solution may turn infeasible, and it assumes reasonable cost in the unlikely worst-case. These conditions are not satisfied when dealing with high risk situations or with basic services like planning water and power supply networks. In these settings a robust approach is more appropriate.

Robust optimization provides a high level of security but represents a rather risk-averse attitude. A solution is called *robust* if it remains feasible under all considered scenarios. The task in robust optimization is to find a robust solution that minimizes its worst-case cost. The difficulty in robustness is that feasibility in all realizations is quite demanding and may not be achieved by any solution. But even if a robust solution exists, it may generate high cost in many scenarios which is not representative for other solutions. These drawbacks have already be discussed when robustness was first applied to linear programming by Soyster [92] in 1973.

To address these concerns, Ben-Tal and Nemirovski [12], El Ghaoui et al. [40, 41], Bertsimas and Sim [15] and Bienstock [16] introduced for linear programs new types of scenario sets. They showed in particular that linear programs under their scenario sets remain tractable. Kouvelis and Yu [78] started to consider robust combinatorial optimization problems with discrete scenario sets and proved several hardness results. A famous theorem by Bertsimas and Sim [14] states that for every polynomially solvable 0-1 discrete optimization problem its robust version with so-called Γ-scenarios can also be solved in polynomial time. Note that all settings for combinatorial optimization problems assume that the values of the objective function are subject to variations, but that the set of feasible solutions remains unchanged.

Despite the progress in defining scenario sets, several researchers felt the need to relax the concept of robustness. Inspired by the idea of a recourse in two-stage stochastic programming, they defined a two-stage procedure that allows a solution chosen under uncertainty in a first stage to be modified by previously fixed means as soon as all data are known. As in the robust setting, the worst-case cost of such a solution is minimized. This idea was introduced by several groups, e.g., Ben-Tal et al. [11], Dhamdhere et al. [37] or Liebchen et al. [81], under the names of adaptive robustness, demand robustness or two-stage robustness, and recoverable robustness, respectively. In this thesis we will call this concept *recoverable robustness*.

Recoverable robust linear programs have been studied in [11, 81, 95]. One drawback of this setting is the increase in complexity, since most of these problems are strongly **NP**-hard. From a practical point of view, recoverable robustness turns out to be an important tool for modeling real world problems gaining valuable insights into the scope of applicability of optimization. For example, its application to several railway optimization problems is analyzed in [25, 26, 30, 81]. In terms of combinatorial optimization, the main focus of research activities focused on scenario sets modeling failures of the underlying topology or uncertainties on the side constraints like the demands (e.g., [37, 44, 55, 73]). The objective function in these cases is assumed to be fixed.

This thesis focuses on two aspects of recoverable robustness. The first one is motivated by a theoretical interest:

- How to define useful recoverable robust combinatorial optimization problems for scenario sets modeling uncertainties in the objective function?
- What kind of changes in respect to complexity can be observed compared to robust models?
- Which combinatorial structures or properties can be detected within the recoverable robust model?

The second aspect is a practical one and emphasizes on how a recoverable robust approach can be adapted to practical problems, e.g., what kind of scenario sets to consider and how to define the recovery actions.

Chapter 1 contains a short introduction to the concept of robustness, important results in this area and their extension to recoverable robustness. The following three Chapters 2, 3 and 4 are dedicated to theoretical issues and Chapters 5 and 6 to more practical applications. In the Appendices A and B, we investigate two combinatorial subproblems of the recoverable robust models introduced in Chapters 2, 3 and 4, in order to give a deeper understanding of the results obtained there. Apart from the definition of different scenario types in Chapter 1, all chapters are self-contained and thus can be read independently of each other. In the remainder of this introduction, we give a short overview of each chapter.

Chapter 1: Dealing with Uncertainties in a Robust Way This chapter contains the history of robustness and its extension to recoverable robustness with the main results in linear programming and combinatorial optimization. In particular we give the definition of three different types of scenario sets, which we will investigate throughout this thesis: the so-called discrete scenario sets, the interval scenario sets and the Γ-scenario sets. Note that we mainly consider uncertainties occurring in the cost function of a given optimization problem.

The considered types of scenario sets differ in the way they are modeled in the input, i.e., whether they are defined explicitly or implicitly, and by further restrictions on the values of the cost functions. In a *discrete scenario set* each scenario and its integer cost function is

explicitly given. *Interval scenario sets* consist of all scenarios that determine a cost function whose values lie in a given cost interval defined by lower and upper cost bounds. For some integer Γ, Γ-*scenario sets* are modifications of interval scenario sets. In contrast to interval scenarios, a Γ-scenario may change only Γ cost values from the lower bound, in the worst-case to the corresponding upper bound.

Chapter 2: k-Distance Recoverable Robustness This chapter covers a recoverable robust approach to combinatorial optimization which is probably the most natural: the *k-distance recoverable robustness* approach. Let us consider a linear combinatorial minimization (LCMin) problem such as the shortest path problem, and a set of scenarios, each defining a cost function, e.g., in the shortest path setting different traveling times. In the k-distance recoverable robust model, a solution is chosen in the first stage, e.g., a simple path. As soon as the scenario is revealed, i.e., once the exact data for the cost function is known, we are allowed to choose a solution that differs just a "little" from the first one. In the shortest path setting one can think of a path using small detours compared to the originally chosen path. In general the difference is measured by the number of new elements contained in the second solution. This number is bounded by some integer k, also called *recovery parameter*. The goal in the k-distance recoverable robust setting is to find a solution in the first stage with minimum total cost. This cost is composed of the so-called first-stage cost and the maximum cost occurring in a scenario for the chosen second path.

We investigate the k-distance recoverable robust version of several well-known combinatorial optimization problems, e.g., the shortest path problem, the minimum spanning tree problem and the minimum perfect matching problem, in combination with the three basic scenario sets defined above, namely discrete scenario sets, interval scenario sets and Γ-scenario sets. For discrete scenarios we start by analyzing the k-Dist-RR version of a quite simple LCMin problem, the weighted disjoint hitting set (WDHS) problem. This problem represents a special case of the deterministic shortest path problem, the minimum (s,t)-cut problem, the minimum perfect matching problem and the minimum spanning tree problem. For two scenarios we show that the k-distance recoverable robust (k-Dist-RR) version of this WDHS problem is weakly **NP**-hard. We also present a pseudo-polynomial algorithm with a run-time depending on a constant number of scenarios and the values of the scenario cost functions. If the number of scenarios is not constant, the problem turns out to be strongly **NP**-hard. In a special case, a lower bound of 1.5 (later improved independently to a bound of 2 by Kasperski and Zieliński [70]) on the best possible approximation factor is achieved, unless $\mathbf{P} = \mathbf{NP}$. The hardness results can easily be transferred to all the problems of which the WDHS problem is a subproblem.

Considering interval scenarios, where the cost function of each scenario is bounded by box-constraints, the complexity status varies between not approximable and solvable in polynomial time. On one hand, the k-Dist-RR shortest path problem is strongly **NP**-hard and cannot be approximated, unless $\mathbf{P} = \mathbf{NP}$. This is in stark contrast to the case where a not necessarily simple (s,t)-path needs to be chosen in a first stage, and the recovery parameter k is a constant, since then the problem is tractable. Also the k-Dist-RR version of the minimum weight basis problem for matroids is solvable in polynomial time for constant k. For the WDHS problem, it can be solved efficiently even if k is not constant but part of the input.

Finally, we analyze Γ-scenarios. By a reduction from the closely related max-scenario problem considered in Appendix A, we show that the k-Dist-RR shortest path problem and the k-Dist-RR minimum (s,t)-cut problem become strongly **NP**-hard.

Chapter 3: Rent Recoverable Robustness Rent recoverable robustness focuses on situations where a choice of an element in the first stage lowers the cost to actually purchase this element in the second stage. The idea is similar to option dealing and the right of first refusal. Let U be a set of elements and $\mathcal{F}$ be a set of feasible solutions, where the goal is to possess a cheap feasible solution in the second stage. In the first stage some elements can be rented. We assume that the rental cost is a fraction of the cost for the elements revealed in the second stage. If we rent an element $u \in U$, in scenario S we need to pay the rent cost of $\alpha \cdot c^S(u)$, where $c^S : U \to \mathbb{N}$ is a cost function defined by S and $0 < \alpha \leq 1$ is a previously fixed *rental factor*. To purchase the element in the second stage we just need to pay the remaining cost $(1-\alpha)c^S(u)$. On the other hand, if we did not rent the element before, we have to pay the original cost and additional inflation cost, i.e., $(1+\beta)c^S(u)$ for some *inflation factor* $\beta \geq 0$. To sum up, an element u may produce cost 0 or $\alpha c^S(u)$ or $c^S(u)$ or $(1+\beta)c^S(u)$. The task in a rent recoverable robust problem is to determine a set of elements for rent, such that the maximum cost for purchasing a solution over all scenarios is minimized. In contrast to the k-Dist-recoverable robust model, a first-stage solution does not limit the set of solutions we can choose from in the second stage.

We show that the rent recoverable robust version of all combinatorial optimization problems that contain the WDHS problem, is weakly **NP**-hard for two scenarios. If the number of scenarios is not constant, the same problems are not approximable with a factor better than 2, unless $\mathbf{P} = \mathbf{NP}$. Considering interval scenarios, a rent recoverable robust problem is solvable in polynomial time if the underlying combinatorial optimization problem is in **P**. In the case of Γ-scenarios the complexity of the rent recoverable robust problem depends on the complexity of the max-scenario problem, i.e., if the max-scenario problem is strongly **NP**-hard and some technical details are fulfilled then the rent recoverable robust version is also strongly **NP**-hard.

In the last section we provide an approximation algorithm, which depends on a robust solution. If the robust solution is a γ-approximation of the corresponding robust problem, we obtain a $\min\{\gamma + 1 + \beta, \frac{\gamma}{\alpha}\}$-approximation for the rent recoverable robust version with a given rental factor α and an inflation factor β. In general, such relations between robust and recoverable robust solutions cannot be achieved, since a robust solution may produce arbitrarily high total cost like it is the case for k-Dist-recoverable robust problems.

Chapter 4: Exact Subset Recoverable Robustness In network design the words robustness and stability are often used interchangeably. In that context a network is called robust/stable if its task is not influenced by intentional or random attacks. For example, in telecommunication networks the demand should remain routable if certain links fail. In general, a network is more robust if it contains redundant links. On the other hand, maintaining such a network is rather costly. In order to reduce the cost, one is interested in abandoning as many links as possible without losing stability. Yet, this approach does not take into account the needs of the customers. A customer is generally not only interested that his requests are routed but also that they are routed as fast as possible.

Exact subset recoverable robustness concentrates on this second requirement. Assuming uncertainties given in the cost function, e.g., in the routing times, the task is to find a subnetwork with minimum size guaranteeing that in every scenario the considered request is routed in the subnetwork as fast as in the original network according to the realized cost function. This problem can be easily extended to all combinatorial optimization problems and interpreted as finding a small set of elements such that this set always contains an optimal solution w.r.t. the original instance.

Starting again with discrete scenario sets, we show strong **NP**-hardness for the exact subset recoverable robust version of the minimum spanning tree problem, the minimum shortest path problem and the minimum perfect matching problem. The key difficulty lies in the subproblem of choosing one element out of a given set with minimum cost. This problem is a special case of the weighted disjoint hitting set problem, which was investigated already in different chapters.

For interval scenario sets the problem becomes more interesting. We develop a general criterion to decide in which case a given element needs to be part of any feasible solution. If a considered linear combinatorial optimization problem is in **P**, we can use this criterion to show that its exact subset recoverable robust version is in **coNP**. Furthermore, we use the criterion in two directions: on the one hand, we prove that the exact subset recoverable robust version of the shortest path problem and the minimum (s,t)-cut problem are not approximable within a factor of $|A|^{(1-\varepsilon)}$ on a directed graph $G=(V,A)$ for any $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$. On the other hand, we derive an algorithm for solving the exact subset recoverable robust minimum weight basis problem on matroids.

The case of Γ-scenarios is again closely related to the max-scenario problem, and thus it follows that the exact subset recoverable robust versions of the shortest path problem and the minimum (s,t)-cut problem are strongly **NP**-hard. But even for matroids the problem turns out to be more difficult. The exact subset recoverable robust minimum spanning tree problem contains as a subproblem the k-connected minimum subgraph problem and can therefore not be solved in polynomial time, unless $\mathbf{P} = \mathbf{NP}$. Finally, we introduce an approximation scheme for interval and Γ-scenarios. This chapter is based on joint work with Rico Zenklusen. In [22] results on the exact subset recoverable robust shortest path problem are published.

Chapter 5: A Recoverable Robust Knapsack Problem An important task in telecommunication is to assign bandwidth to different customers, maximizing the profit for the company. In many cases the demand of the customers varies, e.g., the source, the destination and the traffic volume. Hence, not enough capacity may be available at the point of realization, although service was granted beforehand. To obtain a trade-off between the loss of benefit due to unused resources and the loss in reputation, we allow in our model violations of up to k service promises and new service offers for up to ℓ new requests while maximizing the profit. Applying this approach to a single link of a telecommunication network leads to a recoverable robust version of the knapsack problem. Uncertainties are given in the profit function and in the weight function. As recovery action, k items of the first-stage solution may be deleted and ℓ items may be added, as soon as a scenario reveals its profits and the weights.

Our main focus in this chapter is to obtain similar results as for the classical (robust) knapsack problem. We start with an investigation of the complexity status for discrete scenario sets. As in the robust and deterministic case, the (k,ℓ)-recoverable robust knapsack ((k,ℓ)-rrKP) problem is weakly **NP**-hard and can be solved in pseudo-polynomial time via a dynamic program if the number of scenarios is constant. If this is not the case, the problem is strongly **NP**-hard and in some cases even not approximable, unless $\mathbf{P} = \mathbf{NP}$. occur to

In addition to its complexity status, we are interested in obtaining strongly polyhedral descriptions for this problem. We thus generalize the well-known concept of covers to gain valid inequalities for the recoverable robust knapsack polytope. Besides the canonical extension of covers we introduce a second kind of extension exploiting the scenario-based problem structure and producing stronger valid inequalities. Furthermore, we present two computational studies to investigate the influence of parameters k and ℓ to the objective and evaluate the effectiveness of our new class of valid inequalities.

The (k, ℓ)-rrKP problem with interval scenarios is a special case of the setting with just one discrete scenario. For Γ-scenarios this is not the case. We start by investigating the so-called maximum weight set problem and introduce a combinatorial algorithm for computing for a given set of items the scenario that induces the maximum weight on this set after the recovery is applied. Using this result we introduce an IP formulation for the (k, ℓ)-rrKP problem with Γ-scenarios and no scenario profit whose size is polynomial in the size of the knapsack instance. Note that the set of Γ-scenarios contains an exponential number of different scenarios.

As for discrete scenarios, we adapt cover inequalities and strengthen previous formulations introduced for the robust knapsack polytope by Klopfenstein and Nace [76]. Furthermore, we give an optimal pseudo-polynomial algorithm for solving the corresponding separation problem. This chapter is joint work with Arie Koster and Manuel Kutschka [23].

Chapter 6: Recoverable Robust Train Classification Train classification is an important task in railway optimization, in which a set of given freight trains is sorted to form new trains. In general these trains arrive according to a previously known order at the so-called classification yard, where the cars are decoupled such that they can be sorted. The sorting is performed by moving the cars over a hump, collecting them at some receiving track and pulling them out again if necessary, until the desired train is formed. Since trains are often delayed, the expected order of cars changes and a previously determined sorting schedule becomes infeasible. Traditional classification methods deal with this problem by assuming a worst-case scenario, i.e., that all cars arrive in reversed order, with the drawback of using more sorting steps than necessary.

In our recoverable robust model, we assume that we can interfere the sorting process after an offset of p sorting steps to add k new sorting steps. But, we expect that a recoverable robust schedule sorts the incoming trains into the desired outgoing train if no delay occurs. In the case of disturbances, feasibility of the schedule is reobtained by using the described recovery means. The parameters p and k model the trade-off between robustness and rescheduling. For large p and small k almost no changes happen to a schedule fixed in the first stage, for small p and large k many changes are possible.

We start by introducing a generic algorithm for computing a recoverable robust train classification schedule. For the special scenario set in which each scenario delays up to j trains, this algorithm can be implemented in polynomial time. Yet, in general a special **NP**-hard subproblem needs to be solved, which also induces **NP**-hardness on the recoverable robust train classification problem for $k \geq 1$. In experiments on real-world traffic data we further explore the trade-off underlining that our algorithm saves sorting steps in comparison to traditional methods even for small recovery actions. The results in this chapter are joint work with Jens Maue and partly published in [24].

Appendix A: Max-Scenario Problems The max-scenario problem is an important subproblem of several recoverable robust settings. Given a combinatorial optimization problem and a set of scenarios, the task of this problem is to find a scenario that maximizes the minimum cost of a feasible solution. For discrete scenario sets and interval scenario sets the problem is easy to solve. However, for Γ-scenarios we show that the max-scenario problem for the shortest path problem and the minimum (s, t)-cut problem become strongly **NP**-hard. Using this result, we can show in several recoverable robust settings that recoverable robust versions of these two problems are at least strongly **NP**-hard.

Appendix B: Cardinality Constrained Minimum (s,t)-Cut Problem The upper bounded cardinality constrained minimum (s,t)-cut problem asks for a minimum (s,t)-cut in a graph with bounded cardinality such that its cost is minimized. To the best of our knowledge the complexity status of this problem was open, e.g., stated in [21]. We show that the problem is strongly **NP**-hard. As a consequence, the total cost for an (s,t)-cut cannot be computed in polynomial time in the k-distance recoverable robust model. This result is joint work with Rico Zenklusen.