Performance Analysis in Robust Optimization*

André Chassein and Marc Goerigk

University of Kaiserslautern, Germany
{chassein,goerigk}@mathematik.uni-kl.de

Abstract. We discuss the problem of evaluating a robust solution. To this end, we first give a short primer on how to apply robustification approaches to uncertain optimization problems using the assignment problem and the knapsack problem as illustrative examples. As it is not immediately clear in practice which such robustness approach is suitable for the problem at hand, we present current approaches for evaluating and comparing robustness from the literature, and introduce the new concept of a scenario curve. Using the methods presented in this paper, an easy guide is given to the decision maker to find, solve and compare the best robust optimization method for his purposes.

1 Introduction

Assume you have to solve a real-world optimization problem, which can be modeled, e.g., by an integer linear program. However, an optimal solution to this model might perform quite poorly in practice, as this first modeling approach neglects uncertainty in the problem parameters. Thus, some optimization tool that includes uncertain data is required. You have quite a range of methods to choose from: Stochastic optimization [12], fuzzy programming [14], interval programming [19], or robust optimization.

Assume you have chosen the last option. Again, there is a wide range of robust optimization concepts you may select: classic (strict) robustness [7], absolute or relative regret [23], adjustable robustness [6], recoverable robustness [25], light robustness [16], soft robustness [3], lexicographic α-robustness [21], recovery-to-optimality [17], or similarity-based robustness [13], to name some.

How to decide which of these approaches is appropriate for the problem at hand? Furthermore, most approaches are connected with some kind of "robust objective function" that decides on the degree of robustness. That is, every approach uses a different measure to decide which solution should be called robust, and which not. Also, many robust optimization approaches have parameters that control their "degree of robustness", but it is unclear in advance how to set them. To decide which one should actually be put into practice, you need some possibility to compare the robust solutions of each of these parameters and approaches.

* Effort sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA8655-13-1-3066. The U.S Government is authorized to reproduce and distribute reprints for Governmental purpose notwithstanding any copyright notation thereon.

In this paper, we aim at shedding light on this problem of evaluating a robust solution from a bird's eye, or meta-perspective. We define a range of frameworks that allow to compare the performance of different robust solutions.

This work is not intended as a survey on robust optimization, for which we refer to [18, 4, 1] and [9]. Instead, having introduced some necessary notation in Section 2, we present a walk-through on the application of some of the most popular robust optimization approaches using an uncertain assignment problem as an example in Section 3. We discuss ways to evaluate a robust solution in Section 4, using an additional uncertain knapsack problem as an example. These evaluation frameworks are then illustrated using experimental data in Section 5.

2 Notations and Definitions

We first introduce the notation we use in this paper to denote a general optimization problem. Additionally, we present two academic test problems we use to apply and to compare different robustness concepts. It is a common approach to use well studied academic test problems to compare new algorithms or concepts. On the one hand, these problems have an easy structure and are, therefore, easy to understand. On the other hand, it turns out that many real world problems are closely related variants of these problems.

2.1 General Notation

Consider the general minimization problem

$$\begin{aligned} \min\ & f(x) \\ \text{s.t. } & g(x) \leq 0 \\ & x \in \mathcal{X}. \end{aligned}$$

In robust optimization it is assumed that the parameters ξ that describe the objective function or the constraints of the problem are not known exactly; instead, one assumes to know only a set $\mathcal{U}$ to which the parameters must belong to. These sets are called *uncertainty sets*.

We informally write the uncertain problem as

$$\begin{aligned} \text{“min”}\ & f(x, \xi) \\ \text{s.t. } & \text{“}g(x, \xi) \leq 0\text{”} \\ & x \in \mathcal{X}. \end{aligned}$$

As there exists no unique interpretation of the uncertain objective function and the uncertain constraints, it is not obvious how the robust problem should be solved. Different interpretations have been made, resulting in different robustness concepts. Some of these concepts are presented in Section 3.

The uncertainty set $\mathcal{U}$ can either be a finite list of different scenarios, which is denoted by *finite uncertainty* set, or a continuous sets. Continuous uncertainty

sets are typically hyper-boxes, polytopes, or ellipsoids. A hyper-box uncertainty set is also called *interval uncertainty* set as it defines intervals for the different parameter values. We focus in this paper mainly on finite and interval uncertainty sets. For the discussion of more advanced uncertainty sets we refer to [18].

We use the following notation for the uncertainty sets. Finite uncertainty sets are given by a list of scenarios $\mathcal{U}_F = \{\xi^1, \xi^2, \ldots, \xi^N\}$ and interval uncertainty sets are stated as $\mathcal{U}_I = \left\{\xi : \xi_j \in [\underline{\xi}_j, \overline{\xi}_j]\right\}$.

2.2 The Uncertain Assignment Problem

The assignment problem is defined by a complete bipartite graph with node sets V and W, $|V| = |W| = n$, and edge costs c_{ij} for all $i, j \in [n] := \{1, \ldots, n\}$. A feasible assignment is a subset of edges such that every node from V is connected to exactly one node from W (and vice versa). The problem is to find a feasible assignment that minimizes the sum of edge costs.

Written as an integer linear program (IP), the assignment problem can be stated as:

$$\text{(P)} \quad \min \sum_{i\in[n]} \sum_{j\in[n]} c_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \sum_{i\in[n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{2}$$

$$\sum_{j\in[n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{3}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in [n] \tag{4}$$

Variable x_{ij} equals to 1 if and only if edge (i, j) is part of the assignment. Constraints (2) and (3) ensure that the assignment is feasible. That means that every element from V must be mapped to exactly one element of W, and vice versa. As the constraint matrix is totally unimodular, the integrality constraint (4) is equivalent to its relaxed version $x_{ij} \in [0, 1]$ $\forall i, j \in [n]$. The resulting problem is a linear program (LP). Thus, problem (P) can be solved in polynomial time.

In the uncertain optimization problem the exact knowledge of all edge costs c_{ij} is not given. Instead, we assume that these values are the result of some uncertain process. The set of all possible outcomes of c define the uncertainty set $\mathcal{U}$. As in Section 2.1, we use the following notation for finite and interval uncertainty sets: $\mathcal{U}_F = \{c^1, \ldots, c^N\}$, and $\mathcal{U}_I = \times_{i,j\in[n]}[\underline{c}_{ij}, \overline{c}_{ij}]$. The midpoint of $\mathcal{U}_I$ is denoted by $\hat{c} = 0.5(\underline{c} + \overline{c})$. We write $\mathrm{P}(c)$ to denote the assignment problem with respect to the costs $c \in \mathcal{U}$.

Note that in this problem only the objective function is affected by uncertainty. If an assignment is feasible, it is feasible for all possible scenarios that might occur. This does not hold for the uncertain knapsack problem, which we explain in the following.

2.3 The Uncertain Knapsack problem

The knapsack problem is defined by a set I of n items and a fixed budget B. Each item $i \in I$ is described by a positive weight w_i and a profit p_i. A packing is a subset of all items. A packing is feasible if the sum of weights of all items contained in this packing does not exceed the budget B. The problem is to find a feasible packing that maximizes the sum of all the profits, and can be stated as an integer program (IP):

$$\max \sum_{i\in[n]} p_i x_i \tag{5}$$

$$\text{s.t.} \sum_{i\in[n]} w_i x_i \leq B \tag{6}$$

$$x_i \in \{0,1\} \qquad \forall i \in [n] \tag{7}$$

Variable x_i equals to 1 if and only if item i is part of the packing. Constraint (6) ensures that the budget capacity B is not exceeded. Being NP-complete, the computational complexity of this problem is harder than for the assignment problem (for a general survey on the knapsack problem, see [22]).

In the uncertain version of this problem we assume that both item weights and item profits are affected by uncertainty. The uncertainty set $\mathcal{U}$ contains all possible combinations of weights and profits (p, w). We use the following notation for finite and interval scenarios sets: $\mathcal{U}_F = \{(p^1, w^1), (p^2, w^2), \dots (p^N, w^N)\}$ and $\mathcal{U}_I = (\bigtimes_i [\underline{p}_i, \overline{p}_i]) \times (\bigtimes_i [\underline{w}_i, \overline{w}_i])$.

Note that in this problem not only the objective function but also the constraints are affected by uncertainty. Hence, it is possible that packings are only feasible for some but not for all scenarios.

3 Approaches to Robust Optimization

In this section we present different robustness concepts that are compared in Section 4. The concrete solution of a robustness concepts needs the solution of a *robust counterpart*. The structure and the complexity of the robust counterpart depend greatly on the underlying uncertainty set that is used to describe the uncertainty. We use the assignment problem to illustrate the different concepts and the corresponding counterparts.

3.1 Strict Robustness

Also called *min-max* robustness or *classical* robustness, this is the most conservative way to solve an uncertain optimization problem (see [4]). This concepts asks for a solution that is feasible under all possible scenarios and gives the best performance guarantee, i.e. it optimizes the performance of the worst scenario

for the chosen solution. This yields the following interpretation of the general robust optimization problem

$$\begin{aligned} &\min \max_{\xi \in \mathcal{U}} f(x, \xi) \\ &\text{s.t. } g(x, \xi) \leq 0 && \forall \xi \in \mathcal{U} \\ &\quad x \in \mathcal{X}. \end{aligned}$$

Finite Uncertainty. In the case of a finite uncertainty set $\mathcal{U}_F$, the general optimization problem attains the form

$$\begin{aligned} &\min \max_{k \in [N]} f(x, \xi^k) \\ &\text{s.t. } g(x, \xi^k) \leq 0 && \forall k \in [N] \\ &\quad x \in \mathcal{X}. \end{aligned}$$

It turns out that this uncertainty can lead to very difficult robust counterparts. Several negative complexity results are shown even if $\mathcal{U}$ consists of only two scenarios and the underlying problems are very basic [1]. Nevertheless, the robust counterpart can be formulated in most cases as a mixed integer programming (MIP) problem. We showcase this in the following for the assignment problem.

$$\min z \tag{8}$$

$$\text{s.t. } \sum_{i \in [n]} \sum_{j \in [n]} c_{ij}^k x_{ij} \leq z \qquad \forall k \in [N] \tag{9}$$

$$\sum_{i \in [n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{10}$$

$$\sum_{j \in [n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{11}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in [n] \tag{12}$$

$$z \geq 0 \tag{13}$$

Constraint (9) ensures that variable z is equal to the worst performance of solution x for all possible scenarios $c^1, \ldots, c^N$ in an optimal solution. This additional constraints destroys the total unimodularity of the constraint matrix. Hence, one cannot relax the integrality constraints anymore; in fact, one can show that this problem is NP-complete already for two scenarios [23].

Interval Uncertainty. In the case of interval uncertainty $\mathcal{U}_I$ the robust counterpart can be stated as a semi-infinite program, which is an optimization prob-

lem with finitely many variables and infinitely many constraints.

$$\begin{aligned} \min\ & z & \\ \text{s.t. } & f(x,\xi) \leq z & \forall \xi \in \mathcal{U}_I \\ & g(x,\xi) \leq 0 & \forall \xi \in \mathcal{U}_I \\ & x \in \mathcal{X}. & \end{aligned}$$

In special cases this problem can be greatly simplified. If x is always positive and the objective function has the form $f(x,\xi) = \xi^t x$, the infinitely many constraints describing the objective function can be replaced by one. It suffices to consider $f(x,\bar{\xi}) \leq z$, as the worst scenario that might happen for any solution is scenario $\bar{\xi}$.

The presented assignment problem fulfills these properties. In the worst case scenario the edge costs are given by $\bar{c}$. Hence, the robust counterpart reduces to the problem $\mathrm{P}(\bar{c})$. As this problem has the structure of the original, certain problem, it can be solved with the same algorithms in polynomial time.

The following two approaches rely on the idea to reduce the size of the uncertainty sets. An illustration of both approaches is given in Figure 1.

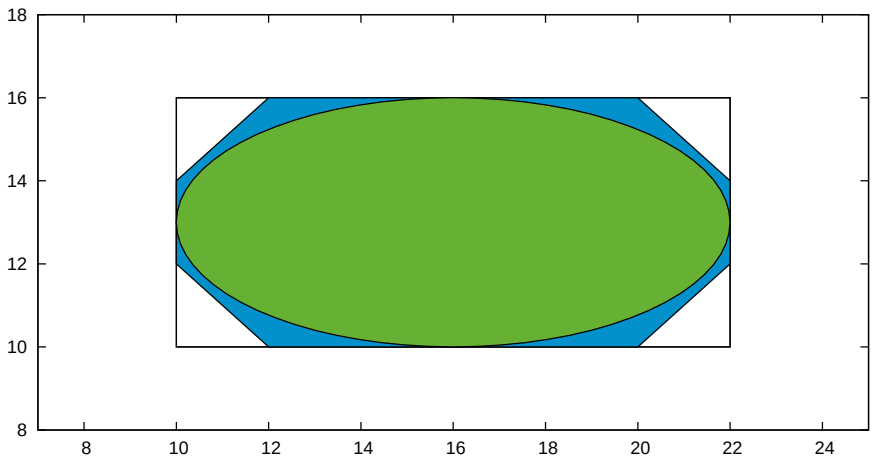

Fig. 1. Cutting unlikely corners. The rectangle represents the complete interval uncertainty set. The bounded uncertainty (see Section 3.2) set is shown as the blue polytope, the ellipsoidal uncertainty set (see Section 3.3) is represented by the green ellipsoid.

3.2 Bounded Uncertainty

This approach was introduced by Bertsimas and Sim [11]. They motivate their approach with the observation that the strict robustness concept with interval uncertainty is very pessimistic, as it assumes that all parameters attain their worst possible value at the same time. As this seems to be an unrealistic assumption for many real-world situations, they suggest to introduce another uncertainty set that bounds the deviation of the parameters.

We present this idea using the the assignment problem. For integral values of Γ, the resulting uncertainty set for the assignment problem has the following form:

$$\mathcal{U}_I(\Gamma) = \{c \in \mathcal{U}_I : |\{(i,j) : c_{ij} > \hat{c}_{ij}\}| \leq \Gamma\}$$

i.e., the number of coefficients that are larger than in the midpoint scenario is bounded by the parameter Γ. The concept can also be generalized to non-integral Γ values.

Using this uncertainty set the robust counterpart of the assignment problem is given by the following mixed integer program (MIP).

$$\min \sum_{i\in[n]} \sum_{j\in[n]} \hat{c}_{ij} x_{ij} + \Gamma\pi + \sum_{i\in[n]} \rho_{ij} \tag{14}$$

$$\text{s.t.} \sum_{i\in[n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{15}$$

$$\sum_{j\in[n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{16}$$

$$\pi + \rho_{ij} \geq (\overline{c}_{ij} - \hat{c}_{ij}) x_{ij} \qquad \forall i,j \in [n] \tag{17}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in [n] \tag{18}$$

$$\rho_i \geq 0 \qquad \forall i \in [n] \tag{19}$$

$$\pi \geq 0 \tag{20}$$

Additional to x, variables π and ρ are introduced. If Γ is chosen as large as the number of x-variables (i.e., $\Gamma = n^2$), then π is equal to 0 in an optimal solution. In this case ρ_{ij} is equal to $(\overline{c}_{ij} - \hat{c}_{ij})x_{ij}$, which is guaranteed by Inequality (17). Replacing ρ_{ij} accordingly in the objective function, only $\overline{c}^t x$ remains, i.e. the worst case objective function. This was expected as for such a large value of Γ the bounded uncertainty set is equal to the original uncertainty set, hence, the model reduces to the strict robust counterpart.

Contrary, if Γ is set to 0, π can be made arbitrary large in an optimal solution. Inequality (17) is superfluous and ρ_{ij} can be set to 0. The objective function reduces to $\hat{c}^t x$. Also this was expected, as for Γ equal to 0, only $\hat{c}^t x$ is contained in the bounded uncertainty set. In [10, 11] it is explained in detail how this formulation can be derived. There, it is also shown that the resulting robust problem is solvable in polynomial time. Note that this concept is also applicable if constraints are affected by uncertainty.

3.3 Ellipsoidal Uncertainty

The use of ellipsoidal uncertainty sets can be motivated from two different reasons. The first one is that many uncertainty sets are already ellipsoidal in practice, e.g., when stemming from a normal data distribution. The second one follows the idea of bounded uncertainty. Even if you have given an interval uncertainty set it can be a good idea to use an ellipsoidal uncertainty set to cut off unlikely corners.

For more information about ellipsoidal uncertainty sets we refer to the papers of Ben-Tal and Nemirovski [5, 8]. We use again the assignment problem to present the resulting robust counter part if an ellipsoidal uncertainty set is used to cut off unlikely corners of the interval uncertainty set $\mathcal{U}_I$. As before, the midpoint of $\mathcal{U}_I$ is denoted by $\hat{c}$.

$$\min \sum_{i\in[n]}\sum_{j\in[n]} \overline{c}_{ij}x_{ij} - \sum_{i,j\in[n]} (\overline{c}_{ij} - \hat{c}_{ij})p_{ij} + \Omega q \tag{21}$$

$$\text{s.t.} \sum_{i,j\in[n]} (\overline{c}_{ij} - \hat{c}_{ij})^2 p_{ij}^2 \leq q^2 \tag{22}$$

$$0 \leq p_{ij} \leq x_{ij} \qquad \forall i,j \in [n] \tag{23}$$

$$0 \leq q \tag{24}$$

$$\sum_{i\in[n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{25}$$

$$\sum_{j\in[n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{26}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in [n] \tag{27}$$

Ben-Tal and Nemirovski explain in [8] how to derive this problem formulation. The parameter Ω controls the size of the ellipsoid that is used to approximate $\mathcal{U}_I$. If Ω is set to 0 the ellipsoid consists of the single point $\hat{c}$. In this case the robust counterpart reduces to $P(\hat{c})$. On the other hand, if Ω is large enough, the problem becomes the strict robust problem $P(\overline{c})$. Note that the problem formulation contains a quadratic constraint. Therefore, it can not be solved anymore using mixed integer programming. Nevertheless, if the integer constraints are relaxed one obtains a convex program that can be solved efficiently.

3.4 Regret Robustness

To apply the regret robustness concept it is assumed that only the objective function is affected by uncertainty and the constraints are certain. In strict robustness the evaluation of a solution depends solely on the performance under one special scenario. It is neglected that this special scenario might also be bad for all other possible solutions. Hence, it could be meaningful to take into account the best possible performance that could be achieved for this special scenario. This idea is used in regret robustness. For a fixed scenario, the regret

of a solution is computed using both the objective function of the solution and the best possible objective value. There exist different methods to compute the regret of a solution. We present three in the following.

Absolute Regret. In absolute regret robustness, one adds some normalization to the robust objective value, so that taking the maximum over all scenarios becomes "more fair". Specifically, we consider the robust objective function

$$reg(x) = \max_{\xi \in \mathcal{U}} \; f(x, \xi) - opt(\xi)$$

where $opt(\xi)$ denotes the best possible objective value for the problem that is described by parameter ξ.

This objective function yields to a different interpretation of the general robust optimization problem.

$$\begin{aligned} &\min \max_{\xi \in \mathcal{U}} f(x, \xi) - opt(\xi) \\ &\text{s.t. } g(x) \leq 0 \\ &\quad x \in \mathcal{X}. \end{aligned}$$

Finite and interval uncertainty sets lead again to different robust counterparts. We present this with the assignment problem.

For a finite set of scenarios $\mathcal{U}_F$ of polynomial size, the optimal objective values can be precomputed in polynomial time. The resulting robust counterpart is very similar to the strict robust counterpart:

$$\min \; z \tag{28}$$

$$\text{s.t.} \sum_{i \in [n]} \sum_{j \in [n]} c_{ij}^k x_{ij} - opt(c^k) \leq z \qquad \forall k \in [N] \tag{29}$$

$$\sum_{i \in [n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{30}$$

$$\sum_{j \in [n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{31}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in [n] \tag{32}$$

$$z \geq 0 \tag{33}$$

Again, variable z is introduced to monitor the robust objective function. Constraint (29) ensures that z equals to the maximum regret in an optimal solution. Specialized algorithms such as branch and bound algorithms can be applied to this problem formulation. However, this is quite different for interval-uncertainty sets, where it is not possible to compute all values $opt(c)$ in advance. Neverthe-

less, it is possible to formulate the resulting robust counterpart as:

$$\min \sum_{i\in[n]}\sum_{j\in[n]} \overline{c}_{ij}x_{ij} - \sum_{i\in[n]}(\alpha_i+\beta_i) \tag{34}$$

$$\text{s.t. } \alpha_i + \beta_j \leq \underline{c}_{ij} + (\overline{c}_{ij} - \underline{c}_{ij})x_{ij} \qquad \forall i,j \in [n] \tag{35}$$

$$\sum_{i\in[n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{36}$$

$$\sum_{j\in[n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{37}$$

$$\alpha_i, \beta_i \gtrless 0 \qquad \forall i \in [n] \tag{38}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in [n] \tag{39}$$

where x, α, and β are variables. To derive this problem formulation it is used that the scenario that maximizes the regret of a solution is described by the following rule: All elements that are chosen by the solution x are as expensive as possible, and all other elements as cheap as possible. The recipe that was used to derive this MIP formulation can be applied to any combinatorial optimization problem with uncertainty in the costs, and for which the nominal problem (P) can be solved by using its linear relaxation. For more information about the derivation of the robust counterpart that arises from the absolute regret concept we refer to [27].

Relative Regret. The previous absolute regret approach aims at a normalization of objective values by using the difference to the best possible objective value in any scenario. However, this normalization may not be appropriate for some applications. The relative regret normalizes the absolute regret by dividing it with the best possible objective function under a scenario. The relative regret objective function has the following form for a general optimization problem.

$$rreg(x) = \max_{\xi\in\mathcal{U}} \frac{f(x,\xi) - opt(\xi)}{opt(\xi)}$$

The relative regret concept breaks down to the absolute regret concept for finite uncertainty sets with a different scaling of scenarios. But for interval uncertainty sets this does not hold. For this case, we present a possible formulation of the

relative regret robust counterpart for the assignment problem.

$$\min \mu \tag{40}$$

$$\text{s.t.} \sum_{i\in[n]} \sum_{j\in[n]} \overline{c}_{ij} x_{ij} \leq \sum_{i\in[n]} (\alpha_i + \beta_i) \tag{41}$$

$$\alpha_i + \beta_j \leq \mu \underline{c}_{ij} + (\overline{c}_{ij} - \underline{c}_{ij}) x_{ij} \qquad \forall i,j \in [n] \tag{42}$$

$$\sum_{i\in[n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{43}$$

$$\sum_{j\in[n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{44}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in [n] \tag{45}$$

$$\alpha, \beta, \mu \geqslant 0 \tag{46}$$

Deriving this formulation of the problem is more involved than in the case of absolute regret, but solving it seems to be almost of the same computational complexity. An additional variable μ is introduced that represents the ratio of objective function and optimal objective value. In [2] one can find the detailed derivation of this problem formulation.

Alpha Regret. For discrete uncertainty sets there exist another approach to interpret how the regret of a solution should be calculated. The alpha regret concept is similar to the concept of absolute regret, but extends it by the notion of anonymization. The idea is to compare the realized solution not with the optimal solution that could be realized in the same scenario. Instead, the vector of solution values $V(x) = \big(f(x,\xi^1), f(x,\xi^2), \ldots, f(x,\xi^N)\big)$ and the vector of optimal solution values $V^* = \big(opt(\xi^1), opt(\xi^2), \ldots, opt(\xi^N)\big)$ are both sorted and then compared in each component. The maximum difference is called the alpha regret. By comparing the solution of the k^{th} best scenario with the k^{th} best optimal solution, the scenarios are made anonymous. This is plausible if it is not known in advance, which scenario will happen or is more likely to happen. Formally, the alpha regret of a solution can be computed as

$$\alpha reg(x) = \min_{\pi\in\sigma(N)} \max_{i\in[N]} f(x,\xi^i) - opt(\xi^{\pi(i)}),$$

where $\sigma(N)$ denotes the set of all permutations of the set $[N]$. Hence, the resulting formulation for the general optimization problem looks as follows

$$\begin{aligned}
&\min\ z \\
&\text{s.t. } f(x,\xi^i) - f^*(\xi^{\pi(i)}) \leq z && \forall i \in [N] \\
&\quad g(x) \leq 0 \\
&\quad x \in \mathcal{X} \\
&\quad \pi \in \sigma(\mathcal{U})
\end{aligned}$$

To give a concrete example we use again the assignment problem.

$$\min\ z \tag{47}$$

$$\text{s.t.} \sum_{i\in[n]}\sum_{j\in[n]} c_{ij}^{\ell} x_{ij} - \sum_{k=1}^{N} p_{k\ell} opt(c^k) \leq z \qquad \forall \ell \in [N] \tag{48}$$

$$\sum_{k\in[N]} p_{k\ell} = 1 \qquad \forall \ell \in [N] \tag{49}$$

$$\sum_{\ell\in[N]} p_{k\ell} = 1 \qquad \forall k \in [N] \tag{50}$$

$$\sum_{i\in[n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{51}$$

$$\sum_{j\in[n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{52}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in [n] \tag{53}$$

$$p_{k\ell} \in \{0,1\} \qquad \forall k,\ell \in [N] \tag{54}$$

The variables $p_{k\ell}$ are used to represent the possible permutations of the scenarios. Variable $p_{k\ell}$ is set to 1 if the k^{th} scenario is sorted to position ℓ, i.e. if $\pi(k) = \ell$. The alpha regret concept is introduced in [21].

3.5 Recoverable Robustness

In the previous approaches, we are interested in finding a single solution, that is supposed to perform well under all possible scenario outcomes. It is not possible to modify this solution, once the actual scenario becomes known. In two-stage approaches to robust optimization (see also the approach of adjustable robustness), this possibility is included in the model. Once the scenario is revealed, we can do some modifications to our solutions. Naturally, if we could change the complete solution, we could simply recover to an optimal solution in any scenario. Thus, the amount of modifications that we can perform is usually bounded. Typically, one considers the min-max objective over all scenarios in this setting; however, any other objective function such as absolute or relative regret would also be conceivable. For the general uncertain optimization problem this yields an infinite program with infinitely many variables and constraints.

$$\begin{aligned}
&\min\ z \\
&\text{s.t. } f(x_\xi, \xi) \leq z && \forall \xi \in \mathcal{U} \\
&\quad g(x_\xi, \xi) \leq 0 && \forall \xi \in \mathcal{U} \\
&\quad dist(x, x_\xi) \leq D && \forall \xi \in \mathcal{U} \\
&\quad x_\xi \in \mathcal{X} \\
&\quad x \in \mathcal{X}
\end{aligned}$$

The solution to the recovery robust problem is given by x. For each possible scenario ξ a variable x_ξ is introduced. Each solution x_ξ must itself be feasible and close to the solution x. The function $dist$ is used to measure how close two solutions are. The maximum allowed distance is given by parameter D.

For the assignment problem with finite uncertainty sets the problem can be written as a MIP. We assume that we are allowed to modify up to $2K$ variables x_{ij} once the scenario is known, i.e. we can remove K choices, and add K new choices to our solution. The resulting problem is given as

$$\min\ z \tag{55}$$

$$\text{s.t.} \sum_{i\in[n]}\sum_{j\in[n]} c^k_{ij}x^k_{ij} \le z \qquad \forall k \in [N] \tag{56}$$

$$\sum_{i\in[n]} x_{ij} = 1 \qquad \forall j \in [n] \tag{57}$$

$$\sum_{j\in[n]} x_{ij} = 1 \qquad \forall i \in [n] \tag{58}$$

$$\sum_{i\in[n]} x^k_{ij} = 1 \qquad \forall j \in [n], k \in [N] \tag{59}$$

$$\sum_{j\in[n]} x^k_{ij} = 1 \qquad \forall i \in [n], k \in [N] \tag{60}$$

$$-y^k_{ij} \le x_{ij} - x^k_{ij} \le y^k_{ij} \qquad \forall i,j \in [n], k \in [N] \tag{61}$$

$$\sum_{i\in[n]}\sum_{j\in[n]} y^k_{ij} \le 2K \qquad \forall k \in [N] \tag{62}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in [n] \tag{63}$$

$$x^k_{ij} \in \{0,1\} \qquad \forall i,j \in [n], k \in [N] \tag{64}$$

$$y^k_{ij} \in \{0,1\} \qquad \forall i,j \in [n], k \in [N] \tag{65}$$

We use variables x to model the first-stage solution, and variables x^k for every scenario $k \in [N]$, to model the second-stage (adapted) solutions. The auxiliary variables y^k are used to measure the difference between x and x^k. In Constraints (57) and (58), we ensure that our first-stage solution x is a feasible assignment, while Constraints (59) and (60) ensure the same for each scenario. Constraints (61) and (62) bound the difference between first- and second-stage solutions. More about recovery robust optimization can be found in [25].

3.6 Summary

We discussed numerous different concepts for robust optimization – still, the presented list of concepts is not exhaustive. Other interesting concepts can be found for example in [18], and in Section 1 of this paper.

In this section we provide a short overview about all presented concepts. In Table 1 we highlight under which uncertainty contexts the different concepts are applicable.

	Cons & Obj		Obj	
	$\mathcal{U}_F$	$\mathcal{U}_I$	$\mathcal{U}_F$	$\mathcal{U}_I$
Strict Robustness	✓	✓	✓	✓
Bounded Uncertainty	–	✓	–	✓
Ellipsoid Uncertainty	a)	✓	a)	✓
Regret Robustness				
Absolute Regret	–	–	✓	✓
Relative Regret	–	–	*b*)	✓
Alpha Regret	–	–	✓	–
Recoverable Robustness	✓	✓	✓	✓

Table 1. This table shows under which uncertainty context the different robustness concepts are applicable. The columns with the label Cons & Obj refer to the case where both the constraints and the objective function are affected by uncertainty. The columns with the label Obj refer to the case where only the objective function is affected by uncertainty. a) An ellipsoid can be computed that contains all point from the discrete set. This will guarantee a safe approximation of the original problem. b) The concept of relative regret reduces to the concept of absolute regret for finite uncertainty sets.

4 Frameworks to Evaluate Robust Solutions

Every robustness concept is motivated from a different perspective and has its own benefits and drawbacks. Therefore, it is unclear how these different concepts perform in comparison to each other. To make two concepts comparable one has to define a framework in which the quality of the solutions that are produced by the different concepts can be measured. In this section we give a short introduction to some of these frameworks. We discuss them in more detail in the experimental Section 5.

Two robustness concepts can only be compared if the used uncertainty context is applicable for both (see Table 1). Hence we define different kind of frameworks for different uncertainty contexts.

For some frameworks we assume the knowledge of an average case scenario, also called *nominal* scenario. The performance of the solution under the nominal scenario is an important indication for the overall quality of the solution.

If we want to speak about feasibility probability we have to make assumptions on the underlying probability structure of the problem.

What is assumed to be known for the different frameworks is given in Table 2.

4.1 The Price of Robustness

This framework is applicable if constraints are affected by uncertainty. Stemming from the seminal paper carrying the same name [11], the price of robustness (PoR) is defined as "the tradeoff between the probability of violation and the effect to the objective function of the nominal problem". This idea can be used to measure the quality of a solution. First, all solutions that have to be compared

Framework	Nominal scenario	Probability distribution
Price of Robustness	✓	✓
AC-WC Curve	✓	–
Scenario Curve	–	–
Sampled Scenario Curve	–	✓
Scenario Curve with Recovery	✓	–

Table 2. Assumptions for the different frameworks.

are evaluated with respect to their nominal performance. Next, simulation can be used to compute the probability of violation. The solutions are compared by drawing them into a two dimensional coordinate system. The $y-$axis defines the nominal performance of a solution and the $x-$axis gives the violation probability.

We note that the term "price of robustness" is also used differently, see, e.g., [26].

4.2 The AC-WC Curve

This framework is only applicable if the the objective function is affected by uncertainty. If constraints are not affected by uncertainty, solutions are feasible for all scenarios; hence, it is not meaningful any more to speak about probability of violation. Instead, one can use the performance in the worst case to compare solutions. Solutions that need to be compared are drawn into a two dimensional coordinate system. The $x-$axis gives the nominal performance guarantee of a solution and the $y-$axis gives the performance in the worst case scenario.

We call a solution non-dominated if there exist no other solution that has both a better average and worst case performance. The set of all non-dominated solutions is defined as the *AC-WC curve*. The AC-WC curve can be computed effectively if the feasibility set of the problem is convex and some further technical assumptions are fulfilled. For more information about the AC-WC curve we refer to [15].

4.3 The Scenario Curve

For this framework it is assumed that the objective function is affected by uncertainty and the uncertainty set is finite. As the uncertainty set is finite the performance of a solution can be evaluated for each possible scenario to obtain the vector $F(x) = \left(f(x, \xi^1), \ldots, f(x, \xi^N)\right)$. Next we use the idea of anonymization: Similar to the concept of alpha regret, the vector $F(x)$ is sorted from good to bad performance. The sorted version of $F(x)$ is denoted by $F_s(x)$. To compare two different solutions x and x' the vectors $F_s(x)$ and $F_s(x')$ are drawn into a two dimensional coordinate system. The k^{th} component of vector $F_s(x)$ is represented by the point $(k, (F_s(x))_k)$. The leftmost point corresponds to the performance under the best scenario and the rightmost point corresponds to

the performance under the worst scenario. The sorting of the different solutions leads to a better visualization of the solution quality.

Additionally, the vector $F^* = \big(opt(\xi^1), \ldots, opt(\xi^N)\big)$ can be computed and the sorted version of this vector is drawn into the same plot. This creates an optimal benchmark curve, that can be used for comparison.

4.4 The Sampled Scenario Curve

Using a sampling procedure the concept of the scenario curve can be transferred to arbitrary uncertainty sets. One needs to be able to sample a set $S = \left\{\tilde{\xi}^1, \ldots, \tilde{\xi}^K\right\}$ of possible parameter realizations. The sampled scenarios are then used to draw the sampled scenario curve.

If a solution is infeasible for a certain scenario, a "bad" value is assigned to this solution. This value must be considerably worse than the worst value of all feasible solutions (e.g., a profit of 0 for the knapsack problem).

4.5 The Scenario Curve with Recovery

We now discuss a second extension of the scenario curve approach from Section 4.3 to optimization problems with uncertainty in the constraints. To this end, we use the uncertain knapsack problem as an illustrative example again.

In these circumstances, it may happen that the robust solution we would like to evaluate is not feasible for some scenarios. We therefore assume that a recovery action is available: By changing up to K many items, we can manipulate our solution for every scenario. For every such recovery distance K, we can calculate a scenario curve as before, which results in a 2-dimensional scenario curve overall.

More precisely, we suggest the following approach to evaluate a solution x. We calculate optimal objective values for every scenario in the uncertainty set and sort these values. For every possible recovery distance $K = 1, \ldots, K_{max}$ we do the following: We calculate the best possible objective value of x for every scenario after the recovery action. Next we sort these values and normalize them using the sorted vector of optimal solutions. In this way we generate K_{max} scenario curves for solution x. We plot all these curves in one plot using a heat map.

On the horizontal axis is the recovery distance, and on the vertical axis are the sorted scenarios. Bright colors mean that the solution is close to the optimal solution after the recovery. A black field means that the solution could not be recovered to a feasible solution in this scenario for the given recovery budget.

5 Experiments

In this last section we use the uncertain assignment problem and the uncertain knapsack problem to illustrate the different frameworks. We use two different uncertainty setups for the assignment problem and four for the knapsack problem. We consider finite and interval uncertainty for the assignment problem and

for the knapsack problem. For the knapsack problem we consider either profit uncertainty or profit and weight uncertainty.

Please keep in mind that all figures show the performance of the different concepts only for specific instances. Therefore, we avoid to make general statements about the different concepts. Instead, we want to explain how the different frameworks can be used to chose the best solution for the specific instances. Beside the different robustness concepts, we also compute the naive solution of the nominal scenario for comparison. This solution is called the average case solution.

5.1 Assignment Problem

We use an assignment instance with 40 nodes, 20 nodes on each side. For the finite uncertainty set we sample 10 different costs for each edge. The costs of an edge are chosen uniformly at random from the interval $[50, 150]$ for each scenario. The nominal scenario is chosen to be the average of the 10 sampled scenarios. In the case of interval uncertainty the midpoint of each interval is chosen uniformly at random from the interval $[100, 150]$ and the length of the interval is also chosen randomly such that edges that are cheap on average tend to have longer intervals.

Finite Uncertainty. We apply the following robustness concepts for this setting: Strict robustness, absolute/relative/alpha regret, and recoverable robustness. The recovery budget for recoverable robustness was set to 2. We present the scenarios curve in Figure 2.

On the left side, the worst scenario for all solutions is compared, and on the right side, the best one (as this is a minimization problem, smaller values indicate better performance). As expected, strict robustness generates the solution that performs at best if the worst may happen. The other robustness concepts perform relatively similar in their worst scenario.

Let us compare the performance of the strict robust solution and the average case solution in more detail to highlight some aspects of robust optimization. The performance of the strict solution shows only little deviation among the different scenarios, whereas the average case solution has the largest performance deviation of all compared solutions. The average case solution is better in all but one scenario compared to the strict robust solution. If one is willing to accept the risk of bad performance in few scenarios, the average case solution may be an appropriate choice. But if this risk cannot be taken, one should rely on a robust solution.

For this instance, an interesting alternative to the strict robust solution is given by the alpha regret solution. It performs better in all but the worst scenario. Further, the performance in the worst scenario is still relatively close to the performance of the strict robust solution.

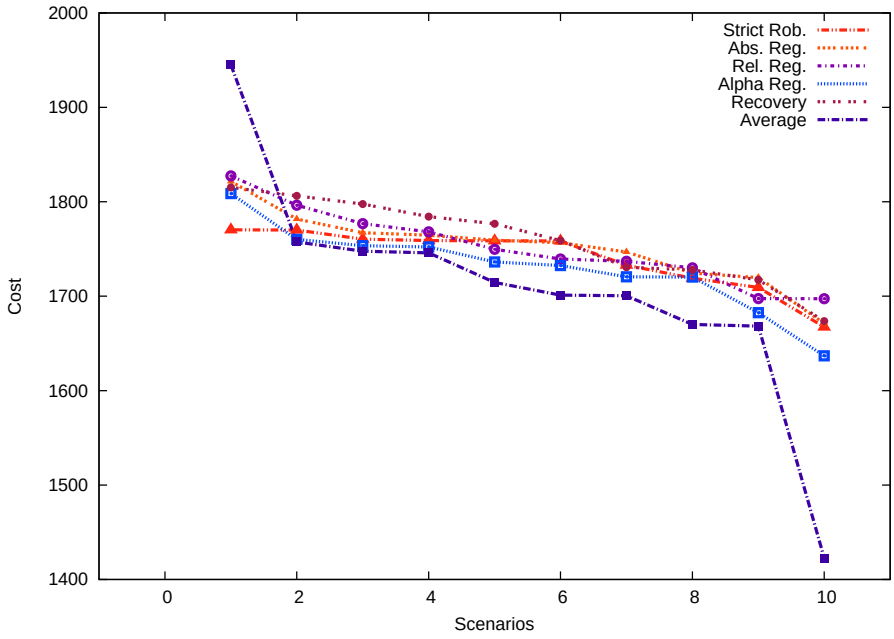

Fig. 2. Scenario curve of an assignment instance with 40 nodes an 10 cost scenarios.

Interval Uncertainty. We apply the following robustness concepts for this setting: Strict robustness, bounded uncertainty, ellipsoidal uncertainty, absolute and relative regret. The parameter Γ describing the bounded uncertainty concept is chosen from the set $\{1, 2, \ldots, 10\}$. The parameter Ω defining the size of the ellipsoid used in the ellipsoidal uncertainty concept is chosen from the set $\{0.5, 1.0, 1.5, 2.0\}$. Figure 3 shows the AC-WC curve of the instance.

For this instance strict robustness, absolute and relative regret generate the same solution. Nevertheless, it is interesting to compare the solutions of bounded and ellipsoidal uncertainty for different parameter choices. Small values of Γ resp. Ω produce solutions that are closer to the average case solution and larger values lead to solutions close to the strict solution. The AC-WC curve enables us to visualize the exact trade-off for moving from average to worst case optimization. For this instance, the bounded uncertainty concept generates solutions that are often dominated by solutions from the ellipsoidal uncertainty concept.

5.2 Knapsack Problem

The capacity of the knapsack is set to 500 in all instances. The following setup is used for finite uncertainty. If both weights and profits are affected by uncertainty, we use an instance with 50 items. For each scenario, the profit of an item is chosen

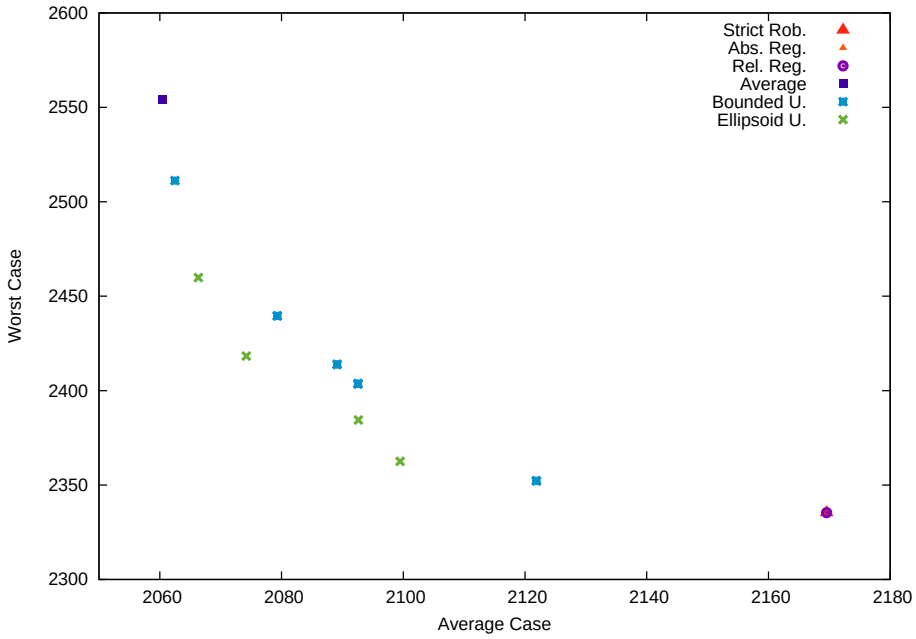

Fig. 3. The AC-WC curve of an assignment instance with 40 nodes and interval uncertainty. The solution of the Strict Robustness, Absolute and Relative regret concept coincide.

uniformly at random from the interval $[50, 150]$ and the weight from the interval $[15, 25]$. If only the profits are affected by uncertainty, we use an instance with 200 items, where the weight of each item is chosen uniformly at random from the interval $[15, 25]$. The profits are generated as before. In both cases, we sample 10 scenarios.

For interval uncertainty we use 500 items. In this case we sample the average profit of an item uniformly from the interval $[60, 140]$ and the average weight from the interval $[12, 28]$. The length of the intervals is chosen randomly proportional to the midpoint of the interval. If the weights are not affected by uncertainty, the interval length is set to 0.

Finite Profit Uncertainty. We apply the following robustness concepts for this setting: Strict robustness, absolute/relative/alpha regret, and recoverable robustness. The recovery budget is set to 2. The scenario curve is shown in Figure 4.

Again the performance in the worst scenario is shown on the left and the performance in the best on the right (as this is a maximization problem, larger values indicate better performance). As expected, the strict robust solution is

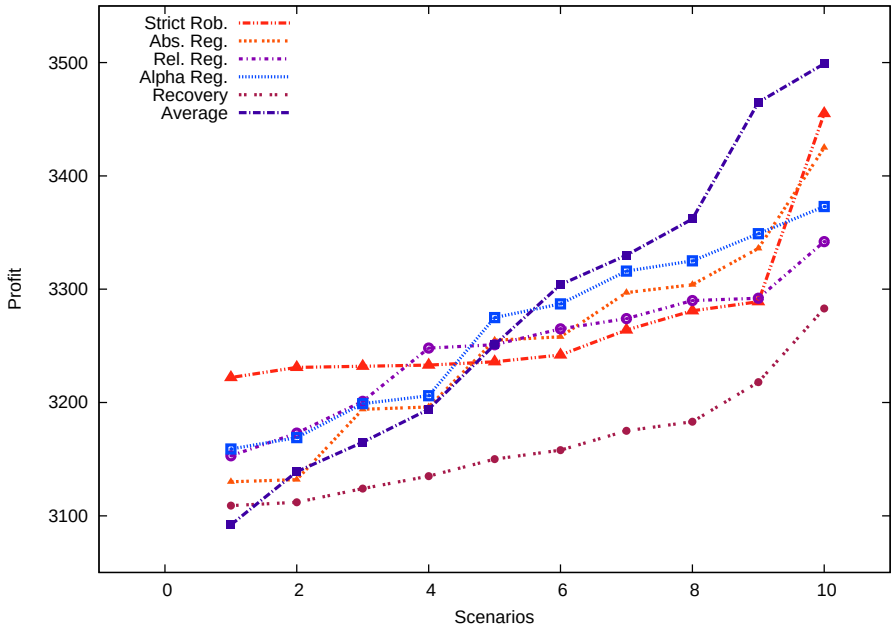

Fig. 4. The scenario curve of an knapsack problem with 200 items and 10 profit scenarios.

the best in the worst case. Interestingly, this is not only true for the worst but also for the three worst scenarios, for this instance. Again the average case solution shows the largest performance deviation of all solutions. If we compare the absolute and the relative regret solution, none of them is clearly preferable. The absolute regret solution is preferable for good scenarios and the relative regret solution is preferable for bad scenarios. It is interesting to note that the strict robust solution performs unexpectedly well for its best scenario.

Finite Uncertainty. We apply the following robustness concepts for this setting: Strict robustness and recoverable robustness. The recovery budget is set to 10. The scenario curves with recovery are shown in Figure 5 for both solutions. Values are normalized with respect to the optimal benchmark curve, where yellow indicates good performance, and darker colors indicate worse performance.

The strict robust solution is ensured to be feasible for all scenarios. Hence, no black boxes can occur in the left figure. The recovery robust solution only guarantees feasibility within the recovery budget that was used for the computation. In this case we used a recovery budget of 10. This means that black boxes may occur in columns 0 to 9, which is indeed the case on the right figure. Note that the recovery robust solution is not feasible for a single scenario if recovery is not

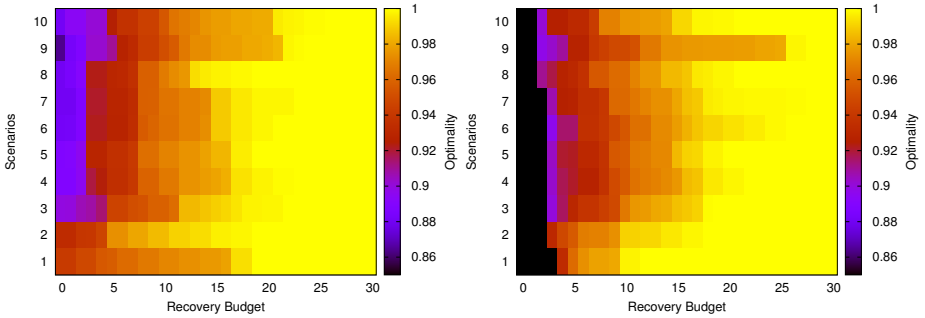

Fig. 5. The scenario curve with recovery of an knapsack problem with 50 items and 10 profit and weight scenarios. The left figure represents the strict robust solution, the right figure the solution generated by the recovery robust concept.

allowed. But, if a recovery budget of 10 is allowed, the recovery robust solution performs considerably better as the worst case solution in the worst case. This can be seen by comparing row 1 of both figures. If the allowed recovery budget is large enough, the originally chosen solution becomes irrelevant as recovery to the optimal solution is possible for each scenario. This explains the bright right side of both figures. Observe that the strict robust solution optimizes purely the worst case without recovery. Therefore this solution has the best value in the field corresponding to scenario 1 and a recovery budget of 0 compared to all other solutions. Whereas the recovery robust solution optimizes the worst case performance if a recovery budget of 10 is allowed. Hence, it has the best value in the field corresponding to scenario 1 and a recovery budget of 10 in comparison with all other solutions.

Interval Profit Uncertainty. We apply the following robustness concepts for this setting: Strict robustness, bounded uncertainty and ellipsoidal uncertainty. We chose $\Gamma = 15$ and $\Omega = 4$. We use this instance to present the sampled scenario curve. We sampled 1000 scenarios. Remember that the profits of the items are defined by intervals. In each scenario that we sample the profit of an item is equally likely one of the endpoints of the interval. The sampled scenario curve is shown in Figure 6.

We also show the "optimal curve" that can be used for comparison. We separately solve each of the 1000 sampled scenarios. The resulting vector of performances is sorted and plotted in the figure. It is clear that no solution can generate a point that lies above the optimal curve. The sampled scenario curve visualizes the conservatism of the strict robust solution for interval uncertainty. For all sampled scenarios the solutions from the bounded and the ellipsoidal uncertainty concept perform better. Only the average case solution performs worse for some scenarios. The sampled scenario curve shows clearly the benefit of the bounded and ellipsoidal uncertainty approach, as very unlikely scenarios

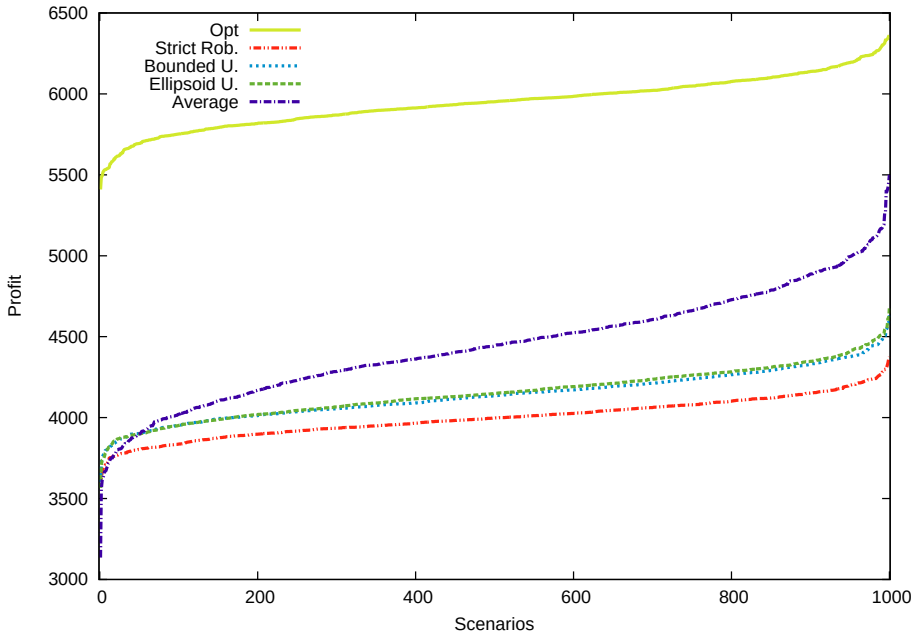

Fig. 6. The sampled scenario curve of an knapsack instance with 200 items and interval profit uncertainty. To generate the scenario curve, 1000 scenarios are sampled.

that will never happen in practice are ignored. It is interesting to note how similar the solutions generated by the bounded and ellipsoid uncertainty concept are, if the describing parameters are chosen accordingly.

Interval Uncertainty. We apply the following robustness concepts for this setting: Strict robustness, bounded uncertainty and ellipsoidal uncertainty. The parameter Γ describing the bounded uncertainty concept is chosen from the set $\{1, 2, \ldots, 10\}$. The parameter Ω defining the size of the ellipsoid used in the ellipsoidal uncertainty concept is chosen from the set $\{0.5, 1.0, 1.5, 2.0\}$. We present the price of robustness in Figure 7 and the sampled scenario curve in Figure 8.

We first consider Figure 7. The strict robust solution is calculated under the assumption that every item has it highest possible weight. Hence, this solution is feasible for all possible parameter realizations, i.e. with 100%. The average case solution, on the other hand, assumes that every item attains its average weight. Therefore, in almost 50% of all parameter realizations the average case solution is infeasible, as the budget constraint is violated. Again, similar to the AC-WC curve the bounded and ellipsoidal uncertainty concepts generate interesting compromise solutions between worst and average case. The indicated curve is

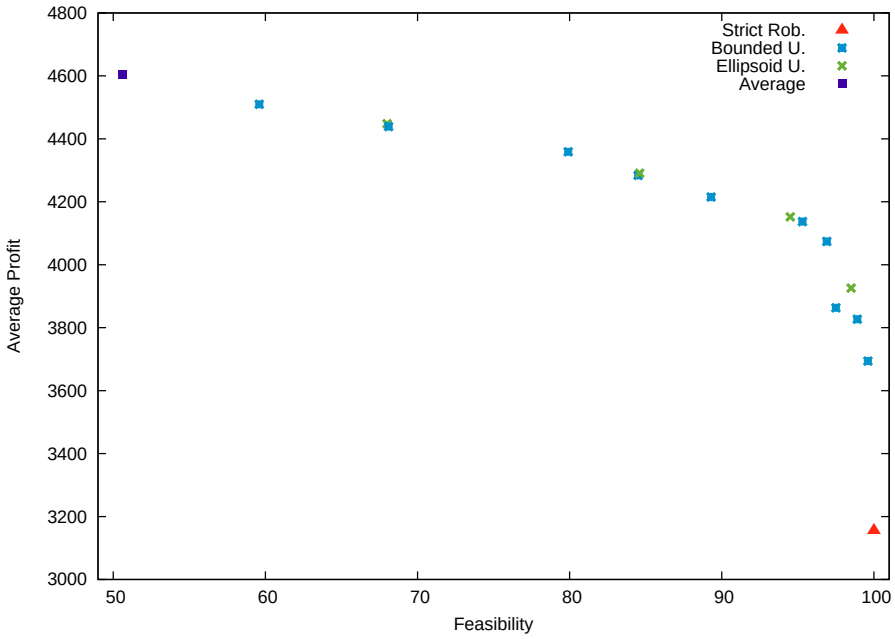

Fig. 7. Price of robustness for a knapsack problem with 200 items and interval uncertainty for profits and weights.

steep near the strict robust solution, which shows the fact that a small relaxation of the feasibility requirement can lead to a big improvement in the average case performance.

We now consider Figure 8. For clarity we only show two solutions of the bounded and ellipsoidal uncertainty set, one for the smallest and one for the largest parameter used in the computation. The performance of an infeasible solution is set to 0. The first look at the average case solution reveals directly that the solution is feasible for about 50% of the parameter realizations. Allowing infeasibility for few scenarios yields a significant improvement for all most all scenarios if the bounded or ellipsoidal uncertainty concept is used. If the parameter values of these concepts are chosen small, solutions are found that perform similar as the average case solution, but are feasible for more scenarios.

References

1. Aissi, H., Bazgan, C., Vanderpooten, D.: Min–max and min–max regret versions of combinatorial optimization problems: A survey. European journal of operational research 197(2), 427–438 (2009)
2. Averbakh, I.: Computing and minimizing the relative regret in combinatorial optimization with interval data. Discrete Optimization 2(4), 273 – 287 (2005)

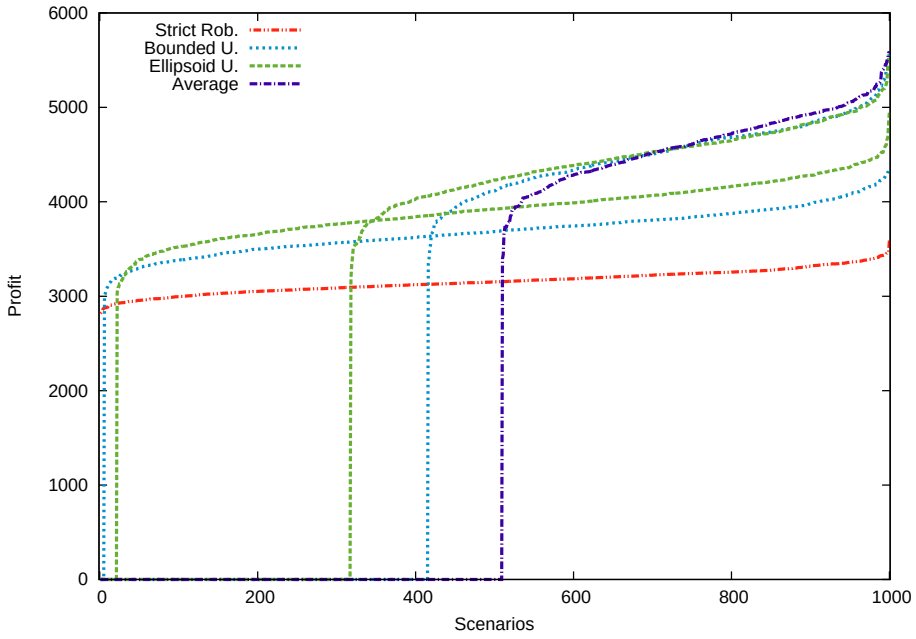

Fig. 8. The sampled scenario curve of a knapsack problem with 200 items and interval uncertainty for profits and weights. To generate the scenario curve, 1000 scenarios are sampled.

3. Ben-Tal, A., Bertsimas, D., Brown, D.B.: A soft robust model for optimization under ambiguity. Operations Research 58(4-Part-2), 1220–1234 (Jul 2010)
4. Ben-Tal, A., Ghaoui, L.E., Nemirovski, A.: Robust Optimization. Princeton University Press, Princeton and Oxford (2009)
5. Ben-tal, A., Goryashko, A., Guslitzer, E., Nemirovski, A.: Robust solutions of uncertain linear programs. Operations Research Letters 25, 1–13 (1999)
6. Ben-Tal, A., Goryashko, A., Guslitzer, E., Nemirovski, A.: Adjustable robust solutions of uncertain linear programs. Math. Programming A 99, 351–376 (2003)
7. Ben-Tal, A., Nemirovski, A.: Robust convex optimization. Mathematics of Operations Research 23(4), 769–805 (1998)
8. Ben-tal, A., Nemirovski, A.: Robust solutions of linear programming problems contaminated with uncertain data. Mathematical Programming 88, 411–424 (2000)
9. Bertsimas, D., Brown, D., Caramanis, C.: Theory and applications of robust optimization. SIAM Review 53(3), 464–501 (2011)
10. Bertsimas, D., Sim, M.: Robust discrete optimization and network flows. Mathematical Programming Series B 98, 2003 (2003)
11. Bertsimas, D., Sim, M.: The price of robustness. Operations Research 52(1), 35–53 (2004)
12. Birge, J.R., Louveaux, F.: Introduction to stochastic programming. Springer Science & Business Media (2011)

13. Buhmann, J.M., Mihalák, M., Srámek, R., Widmayer, P.: Robust optimization in the presence of uncertainty. In: Proceedings of the 4th conference on Innovations in Theoretical Computer Science. pp. 505–514. ACM (2013)
14. Carlsson, C., Fuller, R.: Fuzzy reasoning in decision making and optimization, vol. 82. Physica (2012)
15. Chassein, A., Goerigk, M.: A bicriteria approach to robust optimization. Computers & Operations Research 66, 181–189 (2015)
16. Fischetti, M., Monaci, M.: Light robustness. In: Ahuja, R.K., Möhring, R., Zaroliagis, C. (eds.) Robust and online large-scale optimization, Lecture Note on Computer Science, vol. 5868, pp. 61–84. Springer (2009)
17. Goerigk, M., Schöbel, A.: Recovery-to-optimality: A new two-stage approach to robustness with an application to aperiodic timetabling. Computers & Operations Research 52, Part A(0), 1 – 15 (2014)
18. Goerigk, M., Schöbel, A.: Algorithm engineering in robust optimization. LNCS State-of-the-Art Surveys Springer (2015), to appear.
19. Hladık, M.: Interval linear programming: A survey. Linear programming-new frontiers in theory and applications pp. 85–120 (2012)
20. Kalai, R., Lamboray, C., Vanderpooten, D.: Lexicographic α-robustness: An alternative to min–max criteria. European Journal of Operational Research 220(3), 722 – 728 (2012)
21. Kalaï, R., Lamboray, C., Vanderpooten, D.: Lexicographic α-robustness: An alternative to min–max criteria. European Journal of Operational Research 220(3), 722 – 728 (2012)
22. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems. Springer (2004)
23. Kouvelis, P., Yu, G.: Robust Discrete Optimization and Its Applications. Kluwer Academic Publishers (1997)
24. Liebchen, C., Lübbecke, M., Möhring, R.H., Stiller, S.: The concept of recoverable robustness, linear programming recovery, and railway applications. In: Ahuja, R.K., Möhring, R., Zaroliagis, C. (eds.) Robust and online large-scale optimization, Lecture Note on Computer Science, vol. 5868, pp. 1–27. Springer (2009)
25. Liebchen, C., Lübbecke, M., Möhring, R., Stiller, S.: The concept of recoverable robustness, linear programming recovery, and railway applications. In: Robust and Online Large-Scale Optimization, Lecture Notes in Computer Science, vol. 5868, pp. 1–27. Springer Berlin Heidelberg (2009)
26. Monaci, M., Pferschy, U.: On the robust knapsack problem. SIAM OPT 23(4), 1956–1982 (2013)
27. Yaman, H., Karaşan, O.E., Pınar, M..: The robust spanning tree problem with interval data. Operations Research Letters 29(1), 31 – 40 (2001)