On The Recoverable Robust Traveling Salesman Problem[*]

André Chassein and Marc Goerigk[†]

Fachbereich Mathematik, Technische Universität Kaiserslautern, Germany

Abstract

We consider an uncertain traveling salesman problem, where distances between nodes are not known exactly, but may stem from an uncertainty set of possible scenarios. This uncertainty set is given as intervals with an additional bound on the number of distances that may deviate from their expected, nominal value.

A recoverable robust model is proposed, that allows a tour to change a bounded number of edges once a scenario becomes known. As the model contains an exponential number of constraints and variables, an iterative algorithm is proposed, in which tours and scenarios are computed alternately.

While this approach is able to find a provably optimal solution to the robust model, it also needs to solve increasingly complex subproblems. Therefore, we also consider heuristic solution procedures based on local search moves using a heuristic estimate of the actual objective function. In computational experiments, these approaches are compared.

Finally, an alternative recovery model is discussed, where a second-stage recovery tour is not required to visit all nodes of the graph. We show that the previously NP-hard evaluation of a fixed solution now becomes solvable in polynomial time.

1 Introduction

The traveling salesman problem (TSP) is one of the most researched NP-complete optimization problems in the operations research community (see, e.g., [LLKS85]). Given a (directed, complete) graph with arc lengths, the problem is to determine the shortest circuit visiting all vertices exactly once.

However, depending on the application at hand it may happen that the input data (i.e., the arc lengths) are affected by uncertainty and not known in advance. As an example, there may be congestion within a road network,

[*]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.

[†]Corresponding author. Email: goerigk@mathematik.uni-kl.de

leading to unplanned, increased travel times. Robust optimization [BTGN09] is a paradigm to handle uncertain optimization problems, whose many recent successful applications kindle further research.

In [MBMG07], a robust TSP model is considered, based on the so-called regret or min-max-min approach (see [KY97, ABV09]). Here, for every edge there is an interval given that describes the possible values of lengths this edge may have (the *scenario*). The regret approach asks for a tour that minimizes the worst-case objective over all possible scenarios, where the objective is determined as the difference between the length of the tour, and the best possible tour in this scenario.

While the regret approach is a highly interesting method to handle uncertain optimization problems, it does not capture the possibility to modify a solution on-the-fly, when the scenario becomes known. In the example of a road network, a tour driver might decide to use a different route when the traffic radio informs him of congested sections.

Such an approach amounts to a two-stage problem formulation, and – with slight differences between the concepts – is known as adjustable robustness (see [BTGGN03]), or recoverable robustness (see [LLMS09]). See also [GS13] for a recent overview on the varieties of robustness concepts. Other examples where the concept of recoverable robustness was applied include, e.g., the knapsack problem [BKK11], the shortest path problem [Büs12], the timetabling problem [LLMS09], and the timetable information problem [GHMH$^+$13].

The remainder of this paper is structured as follows. In Section 2, we develop a recoverable robust model formulation for the TSP, with an uncertainty set in which the number of "bad" arc lengths are bounded. As the resulting problem contains an exponential number of scenarios (and thus an exponential number of variables and constraints), we consider an iterative solution approach in Section 3. As the resulting subproblems may still be computationally hard to solve, we introduce heuristic solution methods in Section 4. The presented solution methods are compared in an experimental study in Section 5. We consider an alternative recovery formulation in Section 6, before concluding the paper in Section 7.

2 Problem Formulation

2.1 Nominal Problem

We begin with recapturing the basic TSP. Let a complete graph of n vertices V and (possibly asymmetric) distances d_{ij} for all $i, j \in V$ be given. The TSP consists of finding a directed cycle containing all n vertices with minimal length. Its well-known subtour-elimination formulation is the following:

$$\min \sum_{i \in V} \sum_{j \in V} d_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \sum_{\substack{j \in V \\ j \neq i}} x_{ij} = 1 \qquad \forall i \in V \tag{2}$$

$$\sum_{\substack{j \in V \\ j \neq i}} x_{ji} = 1 \qquad \forall i \in V \tag{3}$$

$$\sum_{(i,j)\in C} x_{ij} \leq |C| - 1 \qquad \forall C \in \mathcal{C} \tag{4}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,j \in V \tag{5}$$

where x_{ij} is a binary variable that indicates if edge (i,j) is used in the tour or not. Denoting the set of all cycles as $\mathcal{C}$, Constraint (4) ensures that there are no subtours. As there are exponentially many such constraints, one typically generates them during the solution procedure, using a separation algorithm. In the following, we denote by $\mathcal{T}$ the set of all feasible traveling salesman tours, i.e., all vectors x fulfilling Constraints (2–5).

2.2 Finite Uncertainty

We now introduce a first robust problem variant, in which a simple, finite uncertainty set is considered. Let a TSP instance be given. Additionally, let $\mathcal{U} = \{d^1, \dots, d^N\}$ be an *uncertainty set* of N distance matrices, also referred to as *scenarios*. Each scenario denotes one possible outcome that we consider. However, it is not known in advance which of these scenarios will actually occur. Thus, we need to find a tour that performs well under all of these scenarios; however, as we apply a recoverable robust approach, we are allowed to modify our solution by at most L edges in each scenario. We denote this problem as TSP($\mathcal{U}$).

Let $\mathcal{N} = \{1, \dots, N\}$. More formally, we can model the recoverable robust TSP with finite uncertainty as the following mixed-integer program:

$$\min \max_{k\in\mathcal{N}} \sum_{i\in V}\sum_{j\in V} d^k_{ij} x^k_{ij} \tag{6}$$

$$\text{s.t.} \quad -y^k_{ij} \leq x_{ij} - x^k_{ij} \leq y^k_{ij} \qquad \forall i,j \in V, k \in \mathcal{N} \tag{7}$$

$$\sum_{i\in V}\sum_{j\in V} y^k_{ij} \leq L \qquad \forall k \in \mathcal{N} \tag{8}$$

$$x \in \mathcal{T} \tag{9}$$

$$x^k \in \mathcal{T} \qquad \forall k \in \mathcal{N} \tag{10}$$

$$y^k_{ij} \geq 0 \qquad \forall i,j \in V, k \in \mathcal{N} \tag{11}$$

We use variables x to denote the nominal tour (i.e., the first-stage solution) and x^k to denote a tour for each scenario k. Then, the objective (6) denotes the worst-case performance over all scenarios, under the additional constraints that solutions may not differ too much. To this end, we use variables y^k_{ij} to denote if x_{ij} and x^k_{ij} differ in Constraint (7), and bound the number of such differences for every scenario in Constraint (8). Note that $L \geq 6$ is necessary to perform any recovery action at all.

Furthermore, note that there are $N + 1$ TSP problems that need to be solved simultaneously. While, e.g., generated cuts can be used for all variables x and x^k, an increasing number of scenarios still results in a highly increased computational complexity. Containing the classic TSP, the recoverable robust TSP is also NP-hard.

2.3 Bounded Uncertainty

We now extend the recoverable robust TSP with finite scenario sets to include more realistic scenario descriptions. Specifically, we assume we are given an interval $[\underline{d}_{ij}, \overline{d}_{ij}]$ that describes the possible outcomes for each distance. Similar to the Bertsimas and Sim idea of bounded uncertainty (see [BS04]), we note that such an uncertainty set also contains unrealistically bad scenarios (e.g., all distances have simultaneously their worst-case length $\overline{d}$), and thus a robust solution would be too conservative. We therefore restrict the number of edges which do not have their best-case length $\underline{d}_{ij}$ to be less or equal to some constant K.

In other words, we consider the following bounded uncertainty set:

$$\mathcal{U}(K) := \left\{ d \in \mathbb{R}^{|V|\times|V|} \mid \underline{d}_{ij} \leq d_{ij} \leq \overline{d}_{ij}, \left|\left\{(i,j) : d_{ij} > \underline{d}_{ij}\right\}\right| \leq K \right\}$$

As this set of scenarios has infinitely many elements (being continuous), the resulting problem TSP($\mathcal{U}$) also has infinitely many constraints and variables. However, note that we can equivalently consider the finite set

$$\mathcal{U}'(K) := \left\{ d \in \mathbb{R}^{|V|\times|V|} \mid d_{ij} = \underline{d}_{ij} + (\overline{d}_{ij} - \underline{d}_{ij})w_{ij}, \sum_{i,j \in V} w_{ij} \leq K, w_{ij} \in \{0,1\} \right\}$$

containing only the extreme scenarios of $\mathcal{U}(K)$. This is because for any fixed tour x, the worst-case scenario from $\mathcal{U}(K)$ is also contained in $\mathcal{U}'(K)$. Therefore, we will sometimes identify a scenario $d \in \mathcal{U}'(K)$ with its defining boolean matrix $w \in \{0,1\}^{n\times n}$.

Hence, the recoverable robust TSP with bounded scenario sets can also be considered as a recoverable robust TSP with exponentially many scenarios.

In the following, we also consider a variant of $\mathcal{U}'(K)$, where we relax the integrality condition of w_{ij}. The resulting uncertainty is denoted as

$$\overline{\mathcal{U}}(K) := \left\{ d \in \mathbb{R}^{|V|\times|V|} \mid d_{ij} = \underline{d}_{ij} + (\overline{d}_{ij} - \underline{d}_{ij})w_{ij}, \sum_{i,j \in V} w_{ij} \leq K, 0 \leq w_{ij} \leq 1 \right\}$$

Note that $\overline{\mathcal{U}}(K) \supseteq \mathcal{U}(K)$. Here, we also bound the total amount of delay that may occur along the tour; however, these delays can be more evenly distributed, thus rendering a potential recovery action less effective.

3 Exact Solution Method

We begin with uncertainties $\mathcal{U} = \mathcal{U}'(K)$. As TSP($\mathcal{U}$) contains a number of scenarios that grows exponentially in K, we now introduce a Benders-decomposition-style solution method which iteratively generates the worst-case for the current first-stage solution.

To this end, we rewrite the problem in the following way:

$$\min\ f(x) \tag{12}$$

$$\text{s.t. } x \in \mathcal{T} \tag{13}$$

where $f(x)$ is the worst-case second-stage performance, i.e.,

$$\begin{aligned} f(x) := \max\ & g(x,w) && && (14)\\ \text{s.t.}\ & \sum_{i\in V}\sum_{j\in V} w_{ij} \le K && && (15)\\ & w_{ij} \in \{0,1\} && \forall i,j \in V && (16) \end{aligned}$$

and $g(x,w)$ is the best recovery option, i.e.,

$$\begin{aligned} g(x,w) := \min\ & \sum_{i\in V}\sum_{j\in V} \left(\underline{d}_{ij} + \left(\overline{d}_{ij} - \underline{d}_{ij}\right) w_{ij}\right) x'_{ij}\\ & x' \in \mathcal{T}\\ & \Delta(x,x') \le L \end{aligned}$$

and $\Delta(x,x')$ denotes the Hamming distance between x and x'.

Note that for arbitrary K and L, computing the evaluation function $f(x)$ of a fixed tour is already NP-hard, as it contains the computation of a new tour as a subproblem. However, if we assume K and L as being fixed, there are only polynomially many scenarios and alternative tours to check.

We use this problem formulation to motivate the following auxiliary problem. Let a set of tours $\mathcal{X} = \{x^1, \dots, x^M\}$ be given. Define $f(\mathcal{X})$ as the optimal objective value of the following program:

$$\begin{aligned} \max\ & \min_{k=1,\dots,M} \sum_{i\in V}\sum_{j\in V} \left(\underline{d}_{ij} + \left(\overline{d}_{ij} - \underline{d}_{ij}\right) w_{ij}\right) x^k && && (17)\\ & \sum_{i\in V}\sum_{j\in V} w_{ij} \le K && && (18)\\ & w_{ij} \in \{0,1\} && \forall i,j \in V && (19) \end{aligned}$$

i.e., $f(\mathcal{X})$ is the worst-case objective value for a fixed set of possible second-stage solutions.

Summarizing, we consider the following three problem steps:

- Problem (P1): Given a finite set of scenarios $\mathcal{U}'$, solve TSP($\mathcal{U}'$).
- Problem (P2): Given a finite set of tours $\mathcal{X}$, solve $f(\mathcal{X})$.
- Problem (P3): Given a tour x and a scenario w, solve $g(x,w)$.

Problem (P1) results in a first-stage solution, while problems (P2) and (P3) are used to evaluate this solution. We repeatedly solve (P2) and (P3), until both objective values are equal. The scenario generated this way is fed back to (P1), until the perceived objective value of (P1) equals the evaluated objective value from (P2) and (P3). Algorithm 1 summarizes this procedure.

An example run of Algorithm 1 is visualized in Figure 1.

In the left subfigure, we show the iterations between solving problem (P1) and the evaluation problem (P2)+(P3). As in every iteration, one scenario is added to the set of currently considered scenarios, the objective value computed in (P1) is monotonically increasing. As each subproblem (P1) is a relaxation of TSP($\mathcal{U}$), every such solution is a lower bound on the optimal objective value.

Algorithm 1 (Exact Approach for Bounded Uncertainty Sets)

```
Require: An instance of TSP($\mathcal{U}$) with a bounded uncertainty set $\mathcal{U}$.
 1: Set $\mathcal{U}' := \emptyset$.
 2: Solve (P1) with respect to $\mathcal{U}'$. Let $x$ be the resulting solution, and $\alpha$ the
    resulting objective value.
 3: Set $\mathcal{X} := \{x\}$.
 4: Solve (P2) with respect to $\mathcal{X}$. Let $w$ be the resulting scenario, and $\beta$ the
    resulting objective value.
 5: Solve (P3) with respect to $x$ and $w$. Let $x'$ be the resulting solution, and $\gamma$
    the resulting objective value.
 6: if $\beta \neq \gamma$ then
 7:     Set $\mathcal{X} := \mathcal{X} \cup \{x'\}$.
 8:     Goto 4.
 9: else
10:     if $\alpha \neq \beta$ then
11:         Set $\mathcal{U}' := \mathcal{U}' \cup \{w\}$
12:         Goto 2.
13:     else
14:         return Optimal solution $x$ with objective value $\alpha = \beta = \gamma$.
15:     end if
16: end if
```

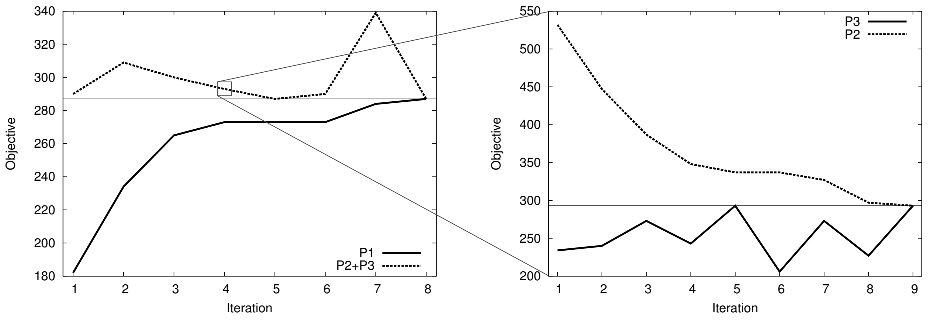


Figure 1: Example run of Algorithm 1.

The evaluated objective values on the other hand, are upper bounds on the optimal objective value and not necessarily monotonically decreasing. When upper and lower bound are equal, an optimal solution is found.

Each evaluation step includes another iterative procedure, shown in the right subfigure for the fourth iteration of the left subfigure. Here, we consider a relaxation for the computation of $f(x)$, as only subsets of possible recovery solutions $\mathcal{X}$ are considered. So, every solution of (P2) gives an upper bound on the objective value of $f(x)$ (being a maximization problem), while every solution of (P3) gives an accompanying lower bound. Again, when both values are equal, $f(x)$ is computed correctly.

As both $g(x,w)$ and $f(x)$ are computed exactly, we find that Algorithm 1 indeed results in an optimal solution to TSP($\mathcal{U}$). Note that the algorithm can also be used as a heuristic solution procedure, by stopping after a finite number of iterations, and using the best solution observed so far (which is not necessarily the last solution).

Note that uncertainties $\mathcal{U} = \overline{\mathcal{U}}(K)$ can be modeled and solved analogously, relaxing constraints (16) and (19) to $0 \leq w_{ij} \leq 1$. As the recovery problem is still NP-hard, evaluating a solution is also still NP-hard.

4 Heuristic Solution Methods

In the following, we present heuristics for $\mathcal{U} = \mathcal{U}'(K)$. However, these concepts can analogously be applied to $\overline{\mathcal{U}}(K)$. As the evaluation of a solution (via the iterative computation of $f(x)$) is in itself already NP-hard, we consider in the following several heuristic algorithms with non-polynomial runtime. The motivation is that while we cannot avoid NP-hardness, the computationally more demanding steps in the exact algorithm is the computation of TSP($\mathcal{U}'$) with increasingly large finite sets $\mathcal{U}'$. Therefore, the number of solution evaluations a heuristic performs will have a large impact on the computation time.

4.1 Starting Solutions

We first consider several ways to construct feasible heuristic solutions. The easiest ways to do so are to solve the nominal problem (i.e., the TSP with respect to lengths $\underline{d}$), or the worst-case problem (i.e., the TSP with respect to lengths $\overline{d}$). We consider two more elaborate alternatives.

Bounded uncertainty without recovery. We consider the strict robust solution with respect to the bounded uncertainty set $\mathcal{U}$ (i.e., a solution in the spirit of [BS04]). The corresponding optimization problem is given as

$$\min \max_{\substack{S \subseteq V \times V \\ |S| \leq K}} \left(\sum_{(i,j) \in S} \overline{d}_{ij} x_{ij} + \sum_{(i,j) \in (V \times V) \setminus S} \underline{d}_{ij} x_{ij} \right) \tag{20}$$

$$\text{s.t. } x \in \mathcal{T} \tag{21}$$

We rewrite the inner maximization problem as

$$\begin{aligned} \max \; & \sum_{i \in V} \sum_{j \in V} \left(\underline{d}_{ij} + \left(\overline{d}_{ij} - \underline{d}_{ij} \right) z_{ij} \right) x_{ij} \\ & \sum_{i \in V} \sum_{j \in V} z_{ij} \leq K \\ & z_{ij} \in \{0,1\} & \forall i,j \in V \end{aligned}$$

Here, the variables z_{ij} is used to determine if edge (i,j) should be worsened or not, while the values x_{ij} are assumed to be fixed. As an optimal solution to this problem is found by simply sorting the objective coefficients of z, and taking the K largest, we may as well assume that the variables z are relaxed (i.e., from $[0,1]$). Dualizing the problem yields

$$\begin{aligned} \min \; & K\alpha + \sum_{i \in V} \sum_{j \in V} \beta_{ij} + \sum_{i \in V} \sum_{j \in V} \underline{d}_{ij} x_{ij} \\ \text{s.t. } & \alpha + \beta_{ij} \geq \left(\overline{d}_{ij} - \underline{d}_{ij} \right) x_{ij} & \forall i,j \in V \end{aligned}$$

$$\begin{aligned} &\alpha \geq 0 \\ &\beta_{ij} \geq 0 && \forall i,j \in V \end{aligned}$$

Using strong duality, we can reformulate model (20–21) to

$$\begin{aligned} \min\ & K\alpha + \sum_{i\in V}\sum_{j\in V}\beta_{ij} + \sum_{i\in V}\sum_{j\in V}\underline{d}_{ij}x_{ij} \\ \text{s.t. }\ & x \in \mathcal{T} \\ & \alpha + \beta_{ij} \geq \left(\overline{d}_{ij} - \underline{d}_{ij}\right)x_{ij} && \forall i,j \in V \\ & \alpha \geq 0 \\ & \beta_{ij} \geq 0 && \forall i,j \in V \end{aligned}$$

A solution of the above problem can be interpreted as an optimal solution to the recoverable robust TSP with respect to any K and to $L = 0$, and may thus perform well as a heuristic for small $L > 0$.

Recovery to optimality. For the second heuristic, we also take the recovery action into account. We follow the idea of RecOpt [GS14] and interpret the computation of a robust solution as a location problem in the solution space. For a finite set of scenarios, we compute an optimal TSP solution each. Then, a robust solution is a tour that minimizes the distance (given as recovery costs) to these solutions.

In other words, we consider the following solution procedure. We first sample randomly a finite set of scenarios $\mathcal{U}' \subseteq \mathcal{U}$. Let $\mathcal{U}' = \{d^1, \dots, d^N\}$. Then, we solve the corresponding TSP for each such scenario. Let x^k, $k \in \mathcal{N} = \{1, \dots, N\}$, be the solution computed this way. We solve

$$\begin{aligned} \min\ & \sum_{k\in\mathcal{N}}\sum_{i\in V}\sum_{j\in V} y^k_{ij} \\ \text{s.t. }\ & x \in \mathcal{T} \\ & -y^k_{ij} \leq x_{ij} - x^k_{ij} \leq y^k_{ij} && \forall k \in \mathcal{N}, i,j \in V \\ & y^k_{ij} \geq 0 && \forall k \in \mathcal{N}, i,j \in V \end{aligned}$$

which is equivalent to

$$\begin{aligned} \min\ & \sum_{i\in V}\sum_{j\in V}\left(\sum_{k\in\mathcal{N}}\chi^k_{ij}\right)x_{ij} \\ \text{s.t. }\ & x \in \mathcal{T} \end{aligned}$$

where $\chi^k_{ij} = -1$, if $x^k_{ij} = 1$, and $\chi^k_{ij} = 1$ otherwise.

4.2 Local Search

Given a feasible starting tour, we now consider methods to improve their objective value. We may reuse well-known neighborhoods from the literature (such as k-opt, see, e.g., [Hel00]), while search guiding mechanisms based on the original TSP objective function are not directly applicable.

As computing the robust objective function $f(x)$ is considerably more time consuming than the computation of the classic TSP objective function, we are limited in the size of search space we evaluate.

Therefore, we consider a heuristic evaluation procedure to speed-up the local search and to make a larger search space possible. To this end, we relax the computation of $g(x, w)$ by considering continuous decision variables, and ignoring subtour elimination constraints. We get the following program:

$$\min \sum_{i\in V}\sum_{j\in V}(\underline{d}_{ij} + (\overline{d}_{ij} - \underline{d}_{ij})w_{ij})x'_{ij} \tag{22}$$

$$\text{s.t.} \sum_{j\neq i} x'_{ij} = 1 \qquad \forall i \in V \tag{23}$$

$$\sum_{i\neq j} x'_{ij} = 1 \qquad \forall j \in V \tag{24}$$

$$\sum_{i\in V}\sum_{j\in V} \chi_{ij} x'_{ij} \le L - n \tag{25}$$

$$x'_{ij} \ge 0 \qquad \forall i,j \in V \tag{26}$$

where $\chi_{ij} = -1$, if $x_{ij} = 1$, and 1 otherwise. The constraints $x'_{ij} \le 1$ for all $i, j \in V$ are redundant due to the assignment constraints. Constraint (25) incorporates the previously used y^k_{ij} variables to model the recovery distance. Dualizing this problem and inserting it into the computation of $f(x)$ yields

$$f'(x) := \max \sum_{i\in V}\lambda_i + \sum_{j\in V}\mu_j + (n-L)\pi \tag{27}$$

$$\text{s.t. } \lambda_i + \mu_j + \nu_{ij} - \chi_{ij}\pi \le \underline{d}_{ij} + (\overline{d}_{ij} - \underline{d}_{ij})w_{ij} \qquad \forall i,j \in V \tag{28}$$

$$\sum_{i\in V}\sum_{j\in V} w_{ij} \le K \tag{29}$$

$$w_{ij} \in \{0,1\} \qquad \forall i,j \in V \tag{30}$$

$$\lambda_i, \mu_j \gtrless 0 \qquad \forall i,j \in V \tag{31}$$

$$\pi \ge 0 \qquad \forall i,j \in V \tag{32}$$

We can use $f'(x)$ as an estimate on the quality of a solution. Its accuracy is considered as part of the following experimental study.

5 Computational Experiments

In the following, we describe the setup and results of three experiments. In the first one, we use small instances that can be solved to optimality, and compare the results to the heuristic solutions from Section 4.1. In the second experiment, we compare the estimated objective function $f'(x)$ from Section 4.2 with the exact objective $f(x)$. The third experiment concerns larger instances, which cannot be solved to optimality within the considered time limit; we compare the best solutions found by the local search algorithm and the iterative algorithm.

5.1 Environment and Instances

All experiments were conducted on a computer with a 16-core Intel Xeon E5-2670 processor, running at 2.60 GHz with 20MB cache, and Ubuntu 12.04. Processes were pinned to one core. Mixed-integer programs were solved using Gurobi v. 5.5 using C++ programs compiled with gcc v. 4.5.4. and flag -O3.

To create an instance of size n, we generated two $n \times n$ matrices $\underline{d}$ and $\overline{d}$ with $\underline{d}_{ij} \in [0, 100]$ and $\overline{d}_{ij} \in [0, 100] + 100$ uniformly at random. Note that such uncorrelated uncertain instances are usually harder to solve than correlated ones for robust optimization problems (see, e.g. [Goe14]).

5.2 Experiment 1: Starting Solutions

We first compare the quality of the "one-shot" solutions from Section 4.1. To this end, we consider 10 instances of each size $n = 5, 6, 7, 8$. Furthermore, we use two sets of parameters: $(K = 1, L = 6)$ and $(K = 3, L = 10)$. The sample size to compute solutions of type RecOpt is 10 scenarios.

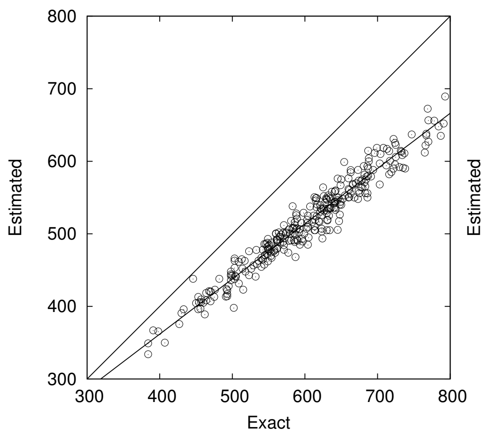
n	K	L	NOM	WC	B	REC	Nr. It.
5	1	6	11.77	22.54	13.59	11.77	2.9
6			6.30	31.88	6.94	6.30	3.0
7			7.71	62.22	9.64	7.71	3.6
8			4.65	84.42	6.64	4.65	3.6
5	3	10	0.00	0.00	0.00	0.00	2.0
6			0.69	3.25	1.51	0.69	3.3
7			4.61	15.43	7.00	4.61	10.6
8			4.80	32.10	7.61	5.61	15.7

Table 1: Results for experiment 1.

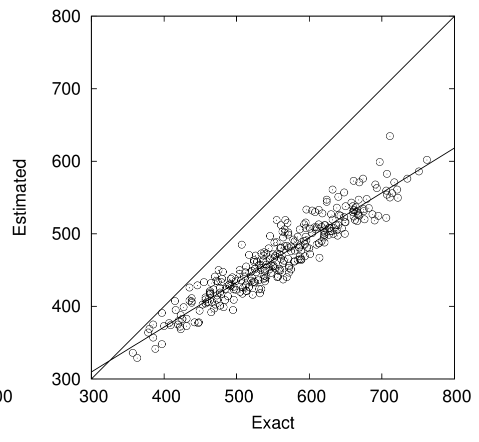
The results are summarized in Table 1. The columns "NOM" refer to the nominal solution, "WC" to the worst-case solution, "B" to the bounded uncertainty without recovery solution, and "REC" to the RecOpt solution. Values are in percent, averaged over 10 instances per instance size n, and normalized with respect to the optimal objective value (e.g., the value 11.77 in the first column and row means that on average, the nominal solution has a robust objective value which is 11.77% larger than the optimal robust objective value). The last column "Nr. It." shows the number of iterations for the exact solution algorithm, i.e., the number of generated worst-case scenarios.

We find that the worst-case solution shows considerably worse performance than all other approaches. This is because the danger of an edge becoming worse than in the nominal case is much overrated, as both the number of worst-cases K and the recovery action L are ignored. Accordingly, solutions of type B perform better, as they include the parameter K. Nominal and RecOpt solutions are identical for most instances, and perform best for the considered parameter sets. Due to the recovery action, ignoring the possible worst-case of an edge is not expensive.

Comparing the parameter sets $K = 1$, $L = 6$ and $K = 3$, $L = 10$, we find an interesting behavior: While the relative objective value of algorithms NOM, B and REC decreases for increasing n and $K = 1$, it increases of increasing n and $K = 3$. The relative objective value of algorithm WC increases in both



(a) $K = 1$, $L = 6$



(b) $K = 3$, $L = 10$

Figure 2: Exact objective values against estimated objective values.

cases. Generally, the decreasing quality of a heuristic one-shot solution can be expected for an increasing instance size; in this case, the uncertainty was small enough that one single edge on the worst-case becomes decreasingly important when the instance size grows. Note that in the case $n = 5$, $L = 10$, any feasible tour is optimal.

This confirms an increasing computational problem complexity for higher values of K and L, which is also mirrored in the increasing number of iterations necessary to find an optimal solution (see the last column).

5.3 Experiment 2: Objective Estimation

We now consider the usage of an estimated objective function $f'(x)$ as described in Section 4.2 in more detail. To this end, we sampled 300 random tours for instances of size $n = 12$, and evaluated both the exact and the estimated objective function. The results are visualized in Figures 2(a) and 2(b), respectively, where we also show the linear regression of the data.

We find both values highly correlated for both parameter sets, with an observed correlation of 0.98 for $K = 1$ and $L = 6$, and 0.94 for $K = 3$ and $L = 10$. Furthermore, estimated objective values tend to be too optimistic, as they relax the subproblem of finding a recovery tour.

5.4 Experiment 3: Local Search

In this experiment we evaluate the quality of the exact algorithm when being used as a heuristic with a timelimit for larger instances. We used 20 minutes of maximum computation time for instances of size $n = 10, \ldots, 18$, with 5 instances of each size. As a comparison, we use the local search from Section 4.2 with the nominal solution as starting solution. To overcome local optima, we used a tabu search implementation, where objective values are slightly randomly perturbed to avoid early cycling[1]. Four variants of this tabu search are used: We either use a 2-opt or a 3-opt neighborhood, and we either use the exact objective values $f(x)$ or the estimated objective values $f'(x)$. In the latter case, the estimated values are computed to assess the neighborhood, but the actual objective value is calculated after each performed move to determine if a new current best solution has been found.

The average objective values are presented in Table 2, where we normalized with respect to the best solution found over all five solution approaches. Column "EX" shows the objective value for Algorithm 1, columns TXF the tabu search using the (full) computation $f(x)$, columns TXE the tabu search using the (estimated) computation $f'(x)$, and the number in the middle stands for the 2-opt or 3-opt neighborhood, respectively.

n	K	L	EX	T2F	T3F	T2E	T3E
10	1	6	0.00	0.00	0.00	0.00	0.00
11			0.00	0.00	1.14	0.00	0.00
12			0.00	5.59	0.83	2.05	0.11
13			0.00	3.47	1.29	2.46	1.32
14			0.00	2.71	3.60	2.83	2.54
15			0.00	0.00	0.08	1.30	0.54
16			0.00	4.71	3.40	4.08	2.50
17			0.00	2.36	0.79	0.79	0.00
18			0.00	1.50	1.11	1.50	1.50
10	3	10	1.23	2.93	2.10	0.00	0.00
11			2.18	1.34	1.90	1.21	0.00
12			1.16	3.20	3.14	0.24	0.17
13			1.38	2.93	1.30	1.81	0.00
14			1.38	1.54	1.54	1.54	0.08
15			0.42	0.65	0.65	0.65	0.23
16			0.23	1.31	1.31	0.00	1.00
17			0.68	0.84	0.84	0.84	0.16
18			1.43	1.43	1.43	1.43	0.00

Table 2: Results for experiment 2.

The results show that while the exact algorithm performs best for instances with $K = 1$, $L = 6$, the increased problem complexity for $K = 3$, $L = 10$ makes the tabu search the better choice in most cases. Furthermore, algorithms which use the exact objective value are outperformed by those using the estimated objective value, resulting in algorithm T3E to be the best choice for the more difficult instances.

By looking into the numbers of iterations presented in Table 3, the improved performance is explained by the considerably larger number of iterations that are possible within the 20 minutes time limit. Even from $n = 12$ on, not all neighbors from the 3-opt neighborhood can be evaluated within the time limit for $K = 3$, $L = 10$.

[1]Naturally, many other meta-heuristic approaches are possible here. A simulated annealing algorithm has also been tested; however, results are not discussed, as tabu search showed a better performance.

n	K	L	EX	T2F	T3F	T2E	T3E
10			4.0	199.6	46.4	633.0	129.4
11			5.0	123.2	24.6	412.8	80.2
12			4.4	90.6	13.8	283.2	46.0
13			5.2	60.8	8.6	212.2	30.2
14	1	6	6.0	43.4	5.4	150.4	18.8
15			6.4	31.0	3.8	113.6	11.8
16			6.2	23.4	3.0	84.8	8.4
17			5.6	16.6	2.0	67.8	6.0
18			5.4	12.4	1.8	50.4	4.2
10			14.8	18.2	3.8	239.6	56.6
11			11.0	10.2	1.8	173.6	37.0
12			9.2	5.4	1.0	115.8	21.6
13			9.4	3.8	1.0	94.0	15.4
14	3	10	7.8	2.2	1.0	58.4	9.2
15			6.2	1.6	1.0	41.4	5.6
16			6.6	1.0	1.0	27.2	3.6
17			5.8	1.0	1.0	23.6	3.2
18			5.4	1.0	1.0	15.6	2.2

Table 3: Iterations in experiment 2.

6 Alternative Recovery Actions

6.1 Setting and Model

The recovery model described in Section 2 includes the possibility to change the current tour once a scenario becomes known, but under the condition that the recovery action results in a new traveling salesman tour. We now consider the case that it is allowed to skip a node along a tour instead, i.e., when the scenario becomes known, we are allowed to modify our tour by leaving out up to L cities.

If a tour contains the arcs (i, j) and (j, k), and node j is skipped, we have to use the arc (i, k) instead. Analogously, for a sequence of arcs

$$(i_1, i_2), (i_2, i_3), \ldots, (i_{k-1}, i_k),$$

skipping nodes $i_2, \ldots, i_{k-1}$ means that arc (i_1, i_k) must be used.

The resulting recoverable robust TSP for finite scenarios $\mathcal{U} = \{d^1, \ldots, d^N\}$ can be modeled using the following integer program:

$$\min \max_{k\in\mathcal{N}} \sum_{i\in V} \sum_{j\in V} d^k_{ij} x^k_{ij} \tag{33}$$

$$\text{s.t. } x_{ij} \le x^k_{ij} + z^k_i \qquad \forall i, j \in V, k \in \mathcal{N} \tag{34}$$

$$x_{ji} \le x^k_{ji} + z^k_i \qquad \forall i, j \in V, k \in \mathcal{N} \tag{35}$$

$$\sum_{i\in V} z^k_i \le L \qquad \forall k \in \mathcal{N} \tag{36}$$

$$\sum_{j\in V} x^k_{ij} = 1 - z^k_i \qquad \forall i \in V, k \in \mathcal{N} \tag{37}$$

$$\sum_{j \in V} x^k_{ji} = 1 - z^k_i \qquad \forall i \in V, k \in \mathcal{N} \tag{38}$$

$$\sum_{(i,j) \in C} x^k_{ij} \leq |C| - 1 - \sum_{i \in V} z^k_i \qquad \forall k \in \mathcal{N}, \mathcal{C} \in \mathcal{C} \tag{39}$$

$$x \in \mathcal{T} \tag{40}$$

$$z^k_{i_0} = 0 \qquad \forall k \in \mathcal{N} \tag{41}$$

$$x^k_{ij} \in \{0,1\} \qquad \forall i,j \in V, k \in \mathcal{N} \tag{42}$$

$$z^k_i \in \{0,1\} \qquad \forall i \in V, k \in \mathcal{N} \tag{43}$$

We use binary variables z^k_i to determine if node i is skipped in scenario k. As before, we consider the worst-case performance in the objective (33). With the help of Constraints (34) and (35), we allow a recovery tour x^k to drop in- and outgoing edges of nodes which are not part of the tour. The total number of skipped nodes is bounded by (36), while Constraints (37) and (38) are modified versions of Constraints (2) and (3). Finally, subtours are eliminated with the help of Constraint (39). We use Constraint (41) to denote that the starting point of a tour cannot be skipped (assuming that $i_0 \in V$ is the start of the tour). The constraint is used for clarity in presentation only – to solve the above problem, one would not include the variables $z^k_{i_0}$ at all.

6.2 Recovery Complexity

In the following, we investigate the complexity of evaluating a given solution under this alternative recovery action. We begin with considering problem (P3), i.e., given a tour and a scenario, what is the best recovery action?

We can model this problem as a resource-constrained shortest path problem. More formally, we are given a distance matrix $\tilde{d}_{ij}$ (taking into account the current scenario) and a tour $P = (i_0, i_1), \ldots, (i_n, i_0)$. We set $r_{ij} = 0$ if $(i,j) \in P$, and for $(i,j) \notin P$, let r_{ij} be the number of nodes which are left out when heading from nodes i to j directly. The recovery problem is now to find a shortest path P' with respect to $\tilde{d}_{ij}$ from i_0 to i_0 using at least one arc, and respecting the resource constraint $\sum_{e \in P'} r_e \leq L$.

While resource-constrained shortest path problems are NP-hard in general (see, e.g., [ID05]), this special case can be solve in polynomial time. To this end, we consider a layered graph $G' = (V', E')$, which is constructed as follows: For every $i \in V$ and $\ell \in \{0, \ldots, L\}$, there exists a node $(i, \ell) \in V'$. We set $i_{n+1} := i_0$ and construct additional nodes $(i_{n+1}, 0), \ldots (i_{n+1}, L)$.

For $p, p' = 0, \ldots, L$, and $q = 0, \ldots, n$ we add an edge going from (i_q, p) to $(i_{q+p'+1}, p + p')$ (if $q + p' + 1 \leq n + 1$ and $p + p' \leq L$), with length $\tilde{d}_{i_q, i_{q+p'+1}}$. Figure 3 shows an example construction for $n = 5$ and $L = 2$.

Finally, we add arcs connecting (i_{n+1}, ℓ) with $(i_{n+1}, \ell+1)$ for $\ell = 0, \ldots, L-1$ with zero length. Setting $s := (i_0, 0)$ and $t := (i_{n+1}, L)$ the resource-constrained shortest path problem is now reformulated as a shortest path problem from s to t. As $L \leq n$, G' is of polynomial size; thus, (P3) can be solved in polynomial time.

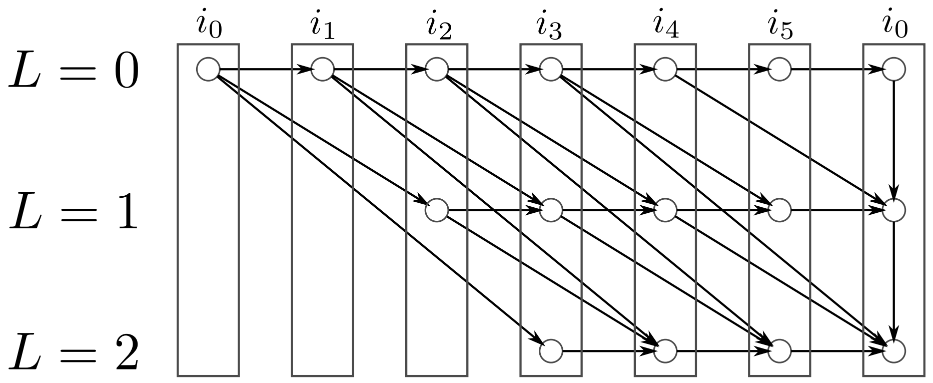


Figure 3: Recovery as a shortest path problem.

6.3 Evaluation Complexity

We now consider the combination of problems (P2) and (P3), i.e., the evaluation of a fixed traveling salesman tour. We distinguish between the discrete set $\mathcal{U}'(K)$ and the continuous set $\overline{\mathcal{U}}(K)$.

In the latter case, we can solve the evaluation in polynomial time using a linear program. Let $G' = (V', E')$ be the layered graph from the previous section. The evaluation problem we consider is given as

$$\begin{aligned} \max\ & g'(x,w) \\ \text{s.t. } & \sum_{i\in V}\sum_{j\in V} w_{ij} \le K \\ & 0 \le w_{ij} \le 1 && \forall i,j \in V \end{aligned}$$

where

$$\begin{aligned} g'(x,w) := \min & \sum_{((i,p),(j,q))\in E'} \left(\underline{d}_{ij} + \left(\overline{d}_{ij} - \underline{d}_{ij}\right) w_{ij}\right) x'_{ipjq} \\ & \sum_{((i,p),(j,q))\in E'} x'_{ipjq} - \sum_{((j,q),(i,p))\in E'} x'_{jqip} = \begin{cases} 1 & \text{if } (i,p) = s \\ -1 & \text{if } (i,p) = t \\ 0 & \text{else} \end{cases} \\ & && \forall (i,p) \in V' \\ & x'_{ipjq} \ge 0 && \forall ((i,p),(j,q)) \in E' \end{aligned}$$

Using strong duality, we find that $g'(x,w)$ is equal to the optimal objective value of the dual of the above problem. Thus, we can reformulate the evaluation to

$$\begin{aligned} \max\ & \pi_t - \pi_s \\ \text{s.t. } & \pi_{(j,q)} - \pi_{(i,p)} \le \underline{d}_{ij} + \left(\overline{d}_{ij} - \underline{d}_{ij}\right) w_{ij} && \forall ((i,p),(j,q)) \in E' \\ & \sum_{i\in V}\sum_{j\in V} w_{ij} \le K \\ & 0 \le w_{ij} \le 1 && \forall i,j \in V \\ & \pi_{(i,p)} \gtrless 0 && \forall (i,p) \in V' \end{aligned}$$

Concerning the case of $\mathcal{U}'(K)$, the above formulation can be adapted by demanding $w_{ij} \in \{0, 1\}$. However, the problem is not polynomially solvable anymore: A direct reduction can be made from the max-scenario-problem, see [Büs09].

7 Conclusion

We introduced a new variant of the traveling salesman problem, in which edge lengths are uncertain and a bounded number of them may become larger than in the nominal case, but we are allowed to use a recovery action once a scenario is realized, which allows to swap a bounded number of edges with alternatives.

As the resulting problem has an exponential number of variables and constraints, we developed an iterative solution algorithm. As even the computation of the robust objective value is NP-hard, this is also done using an inner iterative algorithm based on relaxations of increasing size. Several heuristic solution procedures are investigated. Due to the complex objective evaluation, we formulated a compact mixed-integer program to estimate the actual objective, which can be used within a local search algorithm.

Using experimental data, the high correlation between estimated and exact objective value can be confirmed, and the resulting heuristic algorithm seems to find solutions with better objective value than the iterative solution approach within the same time for the more complex instances.

Finally, we discussed an alternative recovery model, where a recovery action consists of skipping a bounded number of nodes along the tour. We showed that , using a continuous uncertainty set, evaluating the objective value of a fixed solution can be done in polynomial time, while it remains NP-hard for a discrete uncertainty set.

References

[ABV09] H. Aissi, C. Bazgan, and D. Vanderpooten. Minmax and minmax regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197(2):427 – 438, 2009.

[BKK11] C. Büsing, A. M. C. A. Koster, and M. Kutschka. Recoverable robust knapsacks: the discrete scenario case. *Optimization Letters*, 5(3):379–392, 2011.

[BS04] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[BTGGN03] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Math. Programming A*, 99:351–376, 2003.

[BTGN09] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization.* Princeton University Press, Princeton and Oxford, 2009.

[BTN00] A. Ben-Tal and A. Nemirovski. Robust solutions of linear programming problems contaminated with uncertain data. *Math. Programming A*, 88:411–424, 2000.

[Büs09] C. Büsing. The exact subgraph recoverable robust shortest path problem. In R. K. Ahuja, R. H. Möhring, and C. D. Zaroliagis, editors, *Robust and Online Large-Scale Optimization*, volume 5868 of *Lecture Notes in Computer Science*, pages 231–248. Springer Berlin Heidelberg, 2009.

[Büs12] C. Büsing. Recoverable robust shortest path problems. *Networks*, 59(1):181–189, 2012.

[GHMH$^+$13] M. Goerigk, S. Heße, M. Müller-Hannemann, M. Schmidt, and A. Schöbel. Recoverable Robust Timetable Information. In D. Frigioni and S. Stiller, editors, *Proc. of ATMOS 13*, volume 33 of *OASIcs*, pages 1–14, Dagstuhl, Germany, 2013. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Goe14] M. Goerigk. A note on upper bounds to the robust knapsack problem with discrete scenarios. *Annals of Operations Research*, pages 1–9, 2014.

[GS13] M. Goerigk and A. Schöbel. Algorithm engineering in robust optimization. Technical report, Preprint-Reihe, Institut für Numerische und Angewandte Mathematik, Universität Göttingen, 2013. Submitted.

[GS14] M. Goerigk and A. Schöbel. 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.

[Hel00] K. Helsgaun. An effective implementation of the Lin-Kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106 – 130, 2000.

[ID05] S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, pages 33–65. Springer US, 2005.

[KY97] P. Kouvelis and G. Yu. *Robust Discrete Optimization and Its Applications*. Kluwer Academic Publishers, 1997.

[LLKS85] E. L. Lawler, J. K. Lenstra, A. H. G. R. Kan, and D. B. Shmoys. *The traveling salesman problem: a guided tour of combinatorial optimization*, volume 3. Wiley New York, 1985.

[LLMS09] C. Liebchen, M. Lübbecke, R. H. Möhring, and S. Stiller. The concept of recoverable robustness, linear programming recovery, and railway applications. In R. K. Ahuja, R.H. Möhring, and C.D. Zaroliagis, editors, *Robust and online large-scale optimization*, volume 5868 of *Lecture Note on Computer Science*, pages 1–27. Springer, 2009.

[MBMG07] R. Montemanni, J. Barta, M. Mastrolilli, and L. M. Gambardella. The robust traveling salesman problem with interval data. *Transportation Science*, 41(3):366–381, 2007.