ELSEVIER

Operations Research Letters 29 (2001) 31–40

operations research letters

www.elsevier.com/locate/dsw

The robust spanning tree problem with interval data

Hande Yaman[1], Oya Ekin Karaşan, Mustafa Ç. Pınar *

Faculty of Engineering, Department of Industrial Engineering, Bilkent University, TR-06533 Bilkent, Ankara, Turkey

Received 1 July 1999; received in revised form 1 May 2001; accepted 17 May 2001

Abstract

Motivated by telecommunications applications we investigate the minimum spanning tree problem where edge costs are interval numbers. Since minimum spanning trees depend on the realization of the edge costs, we define the robust spanning tree problem to hedge against the worst case contingency, and present a mixed integer programming formulation of the problem. We also define some useful optimality concepts, and present characterizations for these entities leading to polynomial time recognition algorithms. These entities are then used to preprocess a given graph with interval data prior to the solution of the robust spanning tree problem. Computational results show that these preprocessing procedures are quite effective in reducing the time to compute a robust spanning tree. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Uncertainty; Spanning tree; Robust optimization; Interval data

1. Introduction

The purpose of this paper is to introduce the robust version of the minimum spanning tree problem where edge costs (lengths) are specified as interval numbers. Each edge cost can take any value in its interval, independent of the other edge costs. Under the above specification of the data, we propose to compute a *robust* spanning tree, i.e., a spanning tree whose total cost minimizes the maximum deviation from the optimal spanning tree over all realizations of the edge costs. Our study is motivated by two applications in the telecommunications industry. Consider, for instance, the design of a communication network where routing delays on links are not known with certainty due to the time varying nature of the traffic load on the network. In this application, it is desirable to develop a network configuration that hedges against the worst possible contingency in terms of routing delays [4]. A second application arises when a supervisor node in a data network wants to send a control message to all other nodes in the network where transmission lines are subject to uncertain delays [3]. Then, the supervisor node may want to use a robust spanning tree to broadcast the message to all nodes while hedging against the worst possible delay. The combination of interval uncertainty with robustness is attractive in three respects: (1) we do not have to specify a distribution for the data, nor its moments, which is not always easy, (2) although the complexity status of the problem is open we can formulate the robust spanning tree problem as

* Corresponding author. Tel.: +90-312-290-1514; fax: +90-312-266-4126.

E-mail addresses: hyaman@smg.ulb.ac.be (H. Yaman), karasan@bilkent.edu.tr (O.E. Karaşan), mustafap@bilkent.edu.tr (M.Ç. Pınar).

[1] Also at: SMG, Free University of Brussels, CP 210/01 1050 Brussels, Belgium.

0167-6377/01/$ - see front matter © 2001 Elsevier Science B.V. All rights reserved.
PII: S0167-6377(01)00078-5

a mixed integer linear program that can be solved by off-the-shelf software as demonstrated in Section 3.5, and (3) the interval uncertainty allows us to derive properties of the robust spanning tree that we use to our advantage as a preprocessor to reduce significantly the solution time of the mixed integer program.

Our study is not the first to consider a robust version of the minimum spanning tree problem. Kozina and Perepelista [5] have studied the minimum spanning tree problem with interval edge costs. They have defined a relation order on the set of feasible solutions and generated a Pareto set. Kouvelis and Yu [4] have studied the robust spanning tree problem for problems where edge costs assume values in a certain scenario set. They prove that the problem is NP-hard for bounded number of scenarios, and strongly NP-hard for unbounded number of scenarios.

On the other hand, the concept of robustness was studied by Mulvey et al. [7], and Ben-Tal and Nemirovski [2]. Mulvey et al. use a scenario based approach for modeling uncertainty. They use penalty functions to develop robust models to hedge against the worst possible scenario. Under a minimax penalty function, our approach would be similar to the Mulvey et al. approach with the important difference that we refrain from the use of scenarios which we find hard to specify. The approach of Ben-Tal and Nemirovski is based on specifying the uncertainty in a certain ellipsoidal set, and defining a robust counterpart problem. This approach would lead to the specification of a nonlinear 0–1 program in the case of the spanning tree problem, and hence, would be computationally much less practical. A similar remark holds in our case for the use of stochastic programming [8]. Here, we would be in need of specifying a probability distribution, and converting the problem to optimization of expected value (or quantiles) of the objective. This approach would again lead to a possibly nonlinear 0–1 program, which we avoid in our present approach.

Next, we establish the notation used in the sequel. Let $G=(V,E)$ be an undirected graph with n nodes and m edges. Each edge $e=\{i,j\}$ has cost $c_e \in [\underline{c}_e, \bar{c}_e]$. No probability distribution is assumed for edge costs. We use c_e^s to denote the cost of edge e in scenario s. We denote by c_e an arbitrary cost for edge e in $[\underline{c}_e, \bar{c}_e]$. A spanning tree is a set $T \subseteq E$ such that for all $i \in V$ there exists $j \in V$ with $\{i,j\} \in T$ and such that the subgraph (V,T) is acyclic. Let Γ denote the set of all spanning trees. We denote the cost of spanning tree T under scenario s by $c_T^s = \sum_{e \in T} c_e^s$. We use $\underline{c}_T$ to denote the cost of spanning tree T when the costs of all edges on this tree are at lower bounds and $\bar{c}_T$ denotes the cost of spanning tree T when the costs of all edges on this tree are at upper bounds.

The rest of the paper is organized as follows. In Section 2, we define the concepts of weak edge and strong edge used to preprocess the graph efficiently prior to solution of the robust spanning tree problem by mixed-integer programming. Weak and strong edges are characterized in such a way that they can be easily (polynomially) identified. In Section 3, we define the robust spanning tree problem, discuss its properties and relation to strong and weak edges. More specifically, it is shown that robust spanning trees must consist entirely of weak edges, and that there exists a robust spanning tree which uses every strong edge in the graph. A mixed-integer programming formulation for the solution of the robust spanning tree problem is given. We preprocess the mixed integer program by removing from the graph edges which are not weak and by forcing strong edges into the solution. We also report our computational experience in this section. Concluding remarks are given in Section 4.

2. Weak and strong edges

In this section we analyze the problems of deciding whether a given edge is always on a minimum spanning tree (strong edge), or whether a given edge is never on a minimum spanning tree (non-weak edge) and give characterizations to solve both problems in polynomial time.

2.1. Weak edges and trees

We begin our analysis by a characterization of weak trees, i.e., spanning trees that have minimum costs for some realization of edge costs. Similar concepts are proposed in [9] for location of problems.

Definition 2.1. A spanning tree is a *weak tree* if it is a minimum spanning tree for some realization of edge costs.

The following theorem characterizes weak trees.

Theorem 2.1. *A spanning tree is a weak tree if and only if it is a minimum spanning tree when the costs of all edges on this tree are at their lower bounds and the costs of the other edges are at their upper bounds.*

Proof. If a spanning tree is minimum for the stated realization of edge costs, it is a weak tree by definition.

If a spanning tree T is a weak tree then there exists a scenario s for which $c^s_T \leqslant c^s_{T'}$ for all $T' \in \Gamma$. Let c^0_T be the cost realization

$$c^0_e = \begin{cases} \underline{c}_e, & e \in T, \\ \bar{c}_e, & \text{otherwise} \end{cases}$$

for all $e \in E$ and T' be an arbitrary spanning tree in Γ. Then

$$\begin{aligned}
& c^s_T = \sum_{e\in T} c^s_e \leqslant \sum_{e\in T'} c^s_e = c^s_{T'} \\
& \Rightarrow \sum_{e\in T\setminus T'} c^s_e + \sum_{e\in T\cap T'} c^s_e \leqslant \sum_{e\in T'\setminus T} c^s_e + \sum_{e\in T\cap T'} c^s_e \\
& \Rightarrow \sum_{e\in T\setminus T'} c^s_e \leqslant \sum_{e\in T'\setminus T} c^s_e \\
& \Rightarrow \sum_{e\in T\setminus T'} \underline{c}_e \leqslant \sum_{e\in T'\setminus T} \bar{c}_e \\
& \Rightarrow \sum_{e\in T\setminus T'} \underline{c}_e + \sum_{e\in T\cap T'} \underline{c}_e \leqslant \sum_{e\in T'\setminus T} \bar{c}_e + \sum_{e\in T\cap T'} \underline{c}_e \\
& \Rightarrow \sum_{e\in T} c^0_e \leqslant \sum_{e\in T'} c^0_e \\
& \Rightarrow c^0_T \leqslant c^0_{T'}.
\end{aligned}$$

Therefore, T is also a minimum spanning tree under the scenario corresponding to the costs c^0, as required. □

Definition 2.2. An edge is a *weak edge* if it lies on some weak tree.

The following theorem gives a characterization of weak edges.

Theorem 2.2. *Edge e is a weak edge if and only if there exists a minimum spanning tree using edge e when the cost of edge e is at its lower bound and the costs of the remaining edges are at upper bounds.*

Proof. If there exists a minimum spanning tree that uses edge e for the above scenario, then edge e is weak by definition.

We prove the converse by showing that if there does not exist a minimum spanning tree using edge e for the stated scenario, then edge e cannot be weak. Consider Kruskal's algorithm [1] which sorts all edges in non-decreasing order of their costs, and defines $\mathscr{L}$, the set of edges chosen to form a minimum spanning tree. Kruskal's algorithm proceeds as follows. Initially, the set $\mathscr{L}$ is empty. The algorithm examines each edge in the sorted order in turn and checks whether adding the current edge to the set $\mathscr{L}$ creates a cycle with the edges already in $\mathscr{L}$. If it does not, the current edge is added to $\mathscr{L}$, otherwise it is discarded. The algorithm stops when there are $n-1$ edges in $\mathscr{L}$. In case of ties in the sorted order, an edge may be chosen arbitrarily from amongst those with least cost: we modify Kruskal's algorithm slightly by asking that in case of ties, the algorithm favors edge e over other edges to add to $\mathscr{L}$. With this modification, it can be shown that if e is not on the minimum spanning tree found by the algorithm for a particular scenario, then it is not on any minimum spanning tree for that scenario. Let s be the scenario with cost on edge e at its lower bound and all other costs at their upper bounds. We now show that if e is not on any minimum spanning tree for costs c^s, then it cannot be on any minimum spanning tree under any scenario, and so it must be that edge e is not weak. Let $\mathscr{L}^{s'}$ denote the minimum spanning tree returned by the algorithm applied to the graph under scenario s', i.e. with edge costs $c^{s'}$. If e is not on any minimum spanning tree for costs c^s, then it is not on the minimum spanning tree found by the algorithm for costs c^s, so either $|\mathscr{L}^s|$ reaches $n-1$ before e is encountered in the sorted order, or adding edge e to $\mathscr{L}^s$ at the point it was encountered would have introduced a cycle. In either case adding e to $\mathscr{L}^s$ at the point it is encountered in the sorted order would introduce a cycle. Let C denote the edges in such a cycle. Now suppose there is a scenario s' such that $e \in \mathscr{L}^{s'}$. Let D denote the set of edges $e' \in C \setminus e$ such that $e' \notin \mathscr{L}^{s'}$. Clearly $D \neq \emptyset$ and $C \setminus D \subseteq \mathscr{L}^s$. For each edge $e' \in D$, since it was not added to $\mathscr{L}^{s'}$, it must be that e' forms a cycle with edges already in

$\mathscr{L}^{s'}$: let $C_{e'}$ denote the edges in such a cycle. Now it is not hard to see that $(C \setminus D) \cup (\bigcup_{e' \in D} C_{e'} \setminus e')$ induces a cycle with all edges in the set $\mathscr{L}^{s'}$. This is a contradiction, since $\mathscr{L}^{s'}$ forms a tree, so e cannot lie on a minimum spanning tree for scenario s' found by the algorithm, for any scenario s'. Furthermore, by our modification to Kruskal's algorithm, we have that e cannot lie on any minimum spanning tree under any scenario s', and hence e cannot be a weak edge. □

As a corollary of this theorem, we can decide whether a given edge e is weak in $O(m \log m)$ time. All we have to do is to set the cost of edge e to its lower bound, all other edge costs to their upper bounds and apply Kruskal's algorithm in a fashion which will favor edge e as stated in the previous proof. If the minimum spanning tree contains edge e then it must be weak, otherwise it cannot be weak.

2.2. Strong edges

Definition 2.3. An edge is a *strong edge* if it lies on a minimum spanning tree for all realizations of edge costs.

Below, we give a characterization for strong edges. The proof is similar to that of Theorem 2.2, and, hence is omitted.

Theorem 2.3. *Edge e is a strong edge if and only if there exists a minimum spanning tree using edge e when the cost of edge e is at its upper bound and the costs of the remaining edges are at lower bounds.*

As in the case of weak edges we can recognize very efficiently strong edges in a graph using an algorithm similar to the one mentioned at the end of Section 2.1.

3. Robust trees

The purpose of this section is threefold. First, we define the concept of a "robust spanning tree". We define two robustness measures, absolute robustness and relative robustness for the minimum spanning tree problem with interval edge costs, and characterize the worst case scenarios for a given spanning tree for both measures. Second, we propose a mixed integer programming formulation to compute a robust spanning tree. Finally, in Section 3.4 we relate the robust spanning tree to weak and strong edges to help preprocess the graph prior to solution by a mixed integer programming solver. Let S denote the set of all possible scenarios.

3.1. Absolute robust trees

Definition 3.1. Given a spanning tree T, an *absolute worst case scenario* s_T^a is a scenario in which the cost of this spanning tree is the maximum. That is, $s_T^a \in \arg\max_{s \in S} c_T^s$.

It follows from this definition that in an absolute worst case scenario for a given spanning tree the costs of all edges of the spanning tree are fixed at their upper bounds and the costs of the remaining edges can assume any value in their intervals.

Definition 3.2. A spanning tree whose absolute worst case scenario cost is minimum is called an *absolute robust spanning tree*. So an absolute robust spanning tree is given by $T^a \in \arg\min_{T \in \Gamma} \max_{s \in S} c_T^s$.

Consider the scenario in which all edge costs are at their upper bounds. The set of minimum spanning trees under this scenario is exactly the set of absolute robust spanning trees. In particular, every absolute robust spanning tree is a weak tree. Kouvelis and Yu [4] have studied the absolute robust spanning tree problem, where the scenario set is finite, and they have shown that the absolute robust spanning tree problem is NP-complete for bounded scenario set and strongly NP-hard when the scenario set is unbounded. However, in view of the remarks and definitions made above, the absolute robust spanning tree problem with interval edge costs can be solved in polynomial time by finding a minimum spanning tree when all edge costs are at upper bounds.

3.2. Relative robust trees

Definition 3.3. Given a spanning tree T, a *relative worst case scenario* s_T is a scenario in which the difference between the cost of the spanning tree T and the cost of a minimum spanning tree is maximum. That is, $s_T \in \arg\max_{s \in S} \{c_T^s - c_{T^*(s)}^s\}$, where $T^*(s)$ is

a minimum spanning tree for scenario s. We call the difference $d_T = c_T^{s_T} - c_{T^*(s_T)}^{s_T}$ the *robust deviation* for spanning tree T.

Definition 3.4. A spanning tree whose robust deviation is minimum is called a *relative robust spanning tree*. In other words, a relative robust spanning tree $T^r \in \arg\min_{T\in\Gamma} d_T$.

The following proposition gives a relative worst case scenario for a given spanning tree.

Proposition 3.1. *The scenario in which the costs of all edges on T are at upper bounds and the costs of all other edges are at lower bounds is a relative worst case scenario for spanning tree T.*

Proof. Let d_T be the robust deviation for spanning tree T. Then

$$d_T = c_T^{s_T} - c_{T^*(s_T)}^{s_T} = \sum_{e\in T\setminus T^*(s_T)} c_e^{s_T} - \sum_{e\in T^*(s_T)\setminus T} c_e^{s_T}.$$

Let s be the scenario in which the costs of all edges on T are at their upper bounds and the costs of the remaining edges are at their lower bounds. Now,

$$d_T \leqslant \sum_{e\in T\setminus T^*(s_T)} c_e^s - \sum_{e\in T^*(s_T)\setminus T} c_e^s = c_T^s - c_{T^*(s_T)}^s.$$

Since $c_{T^*(s)}^s \leqslant c_{T^*(s_T)}^s$, we get

$$d_T \leqslant c_T^s - c_{T^*(s)}^s.$$

As $d_T = \max_{s'\in S} c_T^{s'} - c_{T^*(s')}^{s'}$, we obtain $d_T = c_T^s - c_{T^*(s)}^s$. Therefore, s is a relative worst case scenario for spanning tree T. □

Kouvelis and Yu [4] also proved that the relative robust spanning tree problem is NP-complete for bounded number of scenarios and is strongly NP-hard with unbounded number of scenarios. They conjecture that the problem with interval edge costs is also NP-complete.

3.3. A mixed integer programming formulation

From this section onwards, we will refer to the relative robust spanning tree problem as the robust spanning tree problem for short. In [6], the minimum spanning tree problem is considered as a special version of a network design problem: we wish to send flow between the nodes of the network and view the edge variable x_e as indicating whether or not we install the edge $e \in E$ to be available to carry any flow. One such flow model as stated in [6] is the single commodity model. In this model, one of the nodes, say node **1** serves as a source node. One unit of flow must be sent from this node to every other node. Define the arc set $A = \{(i,j) \in V \times V : \{i,j\} \in E\}$. Let f_{ij} denote the flow on arc (i,j). The model is

$$\begin{aligned}
\min\quad & \sum_{e\in E} c_e x_e \\
\text{s.t.}\quad & \sum_{(i,j)\in A} f_{ij} - \sum_{(j,i)\in A} f_{ji} = \begin{cases} n-1 & \text{if } i=1, \\ -1 & \forall i \in V\setminus\{1\}, \end{cases} \\
& f_{ij} \leqslant (n-1)x_{ij} \quad \forall \{i,j\} \in E, \\
& f_{ji} \leqslant (n-1)x_{ij} \quad \forall \{i,j\} \in E, \qquad (P_1) \\
& \sum_{e\in E} x_e = n-1, \quad f \geqslant 0, \\
& x_e \in \{0,1\} \quad \forall e \in E.
\end{aligned}$$

Magnanti and Wolsey [6] point out that if we select any node, say node **1**, as the root node for any spanning tree, then we can direct the edges of the tree so that the path from the root node to any other node is directed from the root to that node. To develop a model for this directed version of the problem, the digraph $D = (V, A)$ is formed.

Using these concepts, the authors present another formulation of the minimum spanning tree problem, called the directed multicommodity flow model. In this model every node $k \neq \mathbf{1}$ defines a commodity: one unit of commodity k originates at the root node **1** and must be delivered to node k. Letting f_{ij}^k be the flow of commodity k on arc (i,j), they formulated this model as follows:

$$\begin{aligned}
\min\quad & \sum_{\{i,j\}\in E} c_{ij}(y_{ij} + y_{ji}) \\
\text{s.t.}\quad & \sum_{(j,1)\in A} f_{j,1}^k - \sum_{(1,j)\in A} f_{1,j}^k = -1 \quad \forall k \in V\setminus\{1\}, \\
& \sum_{(j,i)\in A} f_{j,i}^k - \sum_{(i,j)\in A} f_{i,j}^k = 0 \quad \forall i,k \in V\setminus\{1\} \\
& \qquad \text{and} \quad i \neq k, \qquad (P_2)
\end{aligned}$$

$$\sum_{(j,k)\in A} f^k_{j,k} - \sum_{(k,j)\in A} f^k_{k,j} = 1 \quad \forall k \in V \setminus \{1\},$$

$$f^k_{ij} \leqslant y_{ij} \quad \forall (i,j) \in A \quad \text{and} \quad \forall k \in V \setminus \{1\},$$

$$\sum_{(i,j)\in A} y_{ij} = n-1,$$

$$f \geqslant 0 \quad \text{and} \quad y \geqslant 0.$$

In this model, the variable y_{ij} defines a capacity for the flow of each commodity on arc (i,j) only if that arc is a member of the directed spanning tree defined by the vector y. Notice that we do not impose the constraints that y_{ij}'s are integer. This is due to the result of Magnanti and Wolsey [6] where it is shown that the feasible set of P_2 has integer extreme points. We shall use both formulations in our model to find a robust spanning tree. We shall use model (P_1) to characterize the edges on the robust spanning tree, and the dual version of model (P_2) to find the cost of the minimum spanning tree when the costs of all edges on the robust tree are at upper bounds and the costs of all remaining edges are at lower bounds. We replace the flow balance constraints by the equivalent inequality constraints. Then, the dual LP of (P_2) can be written as follows:

$$\max \quad \sum_{k\in V, k\neq 1} (\alpha^k_k - \alpha^k_1) + (n-1)\mu$$

$$\text{s.t.} \quad \sigma^k_{ij} \geqslant \alpha^k_j - \alpha^k_i \quad \forall (i,j) \in A \quad \text{and} \quad \forall k \in V \setminus \{1\},$$

$$\sum_{k\neq 1} \sigma^k_{ij} + \mu \leqslant c_{ij} \quad \forall \{i,j\} \in E, \qquad (D_2)$$

$$\sum_{k\neq 1} \sigma^k_{ji} + \mu \leqslant c_{ij} \quad \forall \{i,j\} \in E,$$

$$\sigma, \alpha \geqslant 0 \text{ and } \mu \text{ unrestricted},$$

where we have associated dual variables $\{\alpha^k_1 : k \in V \setminus \{1\}\}$, $\{\alpha^k_i : i,k \in V \setminus \{1\} \text{ and } i \neq k\}$, $\{\alpha^k_k : k \in V \setminus \{1\}\}$, $\{\sigma^k_{ij} : (i,j) \in A \text{ and } k \in V \setminus \{1\}\}$ and μ for each set of the primal constraints, respectively.

Now, we are ready to give our robust tree formulation:

$$\min \quad \sum_{e\in E} \bar{c}_e x_e - \sum_{k\in V, k\neq 1} (\alpha^k_k - \alpha^k_1) - (n-1)\mu$$

$$\text{s.t.} \quad \sigma^k_{ij} \geqslant \alpha^k_j - \alpha^k_i \quad \forall (i,j) \in A \quad \forall k \in V \setminus \{1\},$$

$$\sum_{k\neq 1} \sigma^k_{ij} + \mu \leqslant \underline{c}_{ij} + (\bar{c}_{ij} - \underline{c}_{ij}) x_{ij} \quad \forall \{i,j\} \in E,$$

$$\sum_{k\neq 1} \sigma^k_{ji} + \mu \leqslant \underline{c}_{ij} + (\bar{c}_{ij} - \underline{c}_{ij}) x_{ij} \quad \forall \{i,j\} \in E,$$

$$\sum_{(i,j)\in A} f_{ij} - \sum_{(j,i)\in A} f_{ji} = \begin{cases} n-1 & \text{if } i = 1, \\ -1 & \forall i \in V \setminus \{1\}, \end{cases}$$

$$f_{ij} \leqslant (n-1) x_{ij} \quad \forall \{i,j\} \in E, \qquad (R)$$

$$f_{ji} \leqslant (n-1) x_{ij} \quad \forall \{i,j\} \in E,$$

$$\sum_{e\in E} x_e = n-1,$$

$$f, \sigma, \alpha \geqslant 0 \text{ and } \mu \text{ unrestricted},$$

$$x_e \in \{0,1\} \quad \forall e \in E.$$

The binary variables x_e's index the edges in the potential robust tree, $\sum_{e\in E} \bar{c}_e x_e$ is the cost of this tree under a relative worst case scenario. For a given 0–1 vector x defining a spanning tree, the cost of edge e can be expressed as $\underline{c}_e + (\bar{c}_e - \underline{c}_e) x_e$. In particular, this model looks for the spanning tree whose robust deviation is the minimum.

3.4. Robust trees, weak edges and strong edges

As pointed out in Section 3.1, an absolute robust spanning tree is a weak tree. Proposition 3.2 below shows that a relative robust spanning tree is also a weak tree. This result is instrumental in preprocessing the graph before the search for the robust tree as it implies that we can discard non-weak edges from the graph.

Lemma 3.1. *If spanning tree T is not the unique minimum spanning tree for the scenario $\hat{s}$ with costs on edges in T at their lower bounds and costs on edges not in T at their upper bounds, then there exists a tree $T' \neq T$ such that $c^s_T \geqslant c^s_{T'}$ for all scenarios s. Furthermore, if T is not weak, $c^s_T > c^s_{T'}$ for all scenarios s.*

Proof. If T is not the unique minimum spanning tree for scenario $\hat{s}$, there exists a spanning tree $T' \neq T$ which is a minimum spanning tree for this scenario. For any scenario s,

$$c_T^s - c_{T'}^s = \sum_{e\in T\setminus T'} c_e^s - \sum_{e\in T'\setminus T} c_e^s \geqslant \sum_{e\in T\setminus T'} \underline{c}_e - \sum_{e\in T'\setminus T} \bar{c}_e = c_T^{\hat{s}} - c_{T'}^{\hat{s}}.$$

Now by the definition of T', $c_T^{\hat{s}} \geqslant c_{T'}^{\hat{s}}$ and thus $c_T^s \geqslant c_{T'}^s$ as required. Furthermore, if T is not weak, it must be that $c_T^{\hat{s}} > c_{T'}^{\hat{s}}$ and so $c_T^s > c_{T'}^s$ for all scenarios s. □

Corollary 3.1. *If T is the unique minimum spanning tree under some scenario, then T is the unique minimum spanning tree for the scenario $\hat{s}$ with costs on edges in T at their lower bounds and costs on edges not in T at their upper bounds.*

Proposition 3.2. *A relative robust spanning tree is a weak tree.*

Proof. Let T be a spanning tree which is not weak. By Lemma 3.1, there exists a spanning tree $T' \neq T$ such that $c_T^s > c_{T'}^s$ for all scenarios s. Consider scenario s' which a relative worst case scenario for spanning tree T'. We have

$$d_{T'} = c_{T'}^{s'} - c_{T^*(s')}^{s'} < c_T^{s'} - c_{T^*(s')}^{s'} \leqslant \max_{s\in S}\{c_T^s - c_{T^*(s)}^s\} = d_T.$$

Therefore T cannot be a robust spanning tree. □

In the remainder of this section our purpose is to establish in Corollary 3.3 that there exists a relative robust spanning tree that uses every strong edge in the graph. To arrive at this conclusion we prove several intermediate results which, among other things, contain a characterization of strong edges using the concept of unionwise permanent sets that we define below [9]. In the sequel we work with the following assumption that is instrumental in the proof of Lemma 3.2.

Assumption 3.1. *All edges* have *non-degenerate costs, that is $\underline{c}_e < \bar{c}_e$ for all $e \in E$.*

Lemma 3.2. *For any two distinct spanning trees T and T', there exists a scenario s' for which $c_T^{s'} \neq c_{T'}^{s'}$.*

Proof. Pick the scenario s corresponding to edge costs at their lower bounds. If $c_T^s = c_{T'}^s$ then pick an edge $e \in T \setminus T'$ and assign the corresponding cost to its upper bound. Let s' denote this scenario. Then $c_T^{s'} > c_T^s = c_{T'}^s = c_{T'}^{s'}$. □

Definition 3.5. A set of spanning trees is a *unionwise permanent set* if for each realization there exists a minimum spanning tree in this set.

Definition 3.6. A unionwise permanent set is a *minimal unionwise permanent set* if it is no longer a unionwise permanent set when a spanning tree is removed.

Lemma 3.3. *If a spanning tree T is never the unique minimum spanning tree, there exists a spanning tree T' such that $c_T \geqslant c_{T'}$ for all scenarios and T' is the unique minimum spanning tree for some scenario.*

Proof. If T is not the unique minimum spanning tree for any scenario, then by Lemma 3.1 there exists another spanning tree $T_1 \neq T$ such that $c_T \geqslant c_{T_1}$ for all scenarios. If T_1 is the unique minimum spanning tree for some scenario, we are done. Assume not. Then there exists another spanning tree $T_2 \neq T_1$ such that $c_{T_1} \geqslant c_{T_2}$ for all scenarios by Lemma 3.1. Besides $T_2 \neq T$ since the contrary would imply that $c_T = c_{T_1} = c_{T_2}$ for all scenarios, which contradicts Lemma 3.2. Repeating this argument we either stop with a spanning tree which is the unique minimum spanning tree for a scenario or we enumerate all the spanning trees in the graph. In the latter case, we will end up with a sequence of spanning trees which are not the unique minimum spanning tree for any scenario and which satisfy:

$$c_T \geqslant c_{T_1} \geqslant \cdots \geqslant c_{T_{k-1}} \geqslant c_{T_k}.$$

Note that in this sequence no spanning tree can be repeated since by Lemma 3.2 two distinct spanning trees cannot have the same cost under all scenarios. Finally, again by Lemma 3.2 there exists a scenario s where $c_{T_{k-1}}^s \neq c_{T_k}^s$. Together with the above inequality this implies that for scenario s, we have

$$c_T^s \geqslant c_{T_1}^s \geqslant \cdots \geqslant c_{T_{k-1}}^s > c_{T_k}^s.$$

For this scenario, tree T_k is the unique minimum spanning tree and so it is the desired spanning tree T'. □

Now we give a characterization for a minimal unionwise permanent set and show that it is unique.

Theorem 3.1. *Let Γ' be the set of spanning trees each of which is the unique minimum spanning tree when the costs of all edges on this spanning tree are at their lower bounds and the costs of the remaining edges are at their upper bounds. Then, Γ' is a minimal unionwise permanent set.*

Proof. Assume Γ' is not a unionwise permanent set. Then there exists a scenario s, for which no spanning tree in Γ' is minimum. Let T be a minimum spanning tree under scenario s, so T is a weak tree and $c_T^s < c_{\bar{T}}^s$ for all $\bar{T} \in \Gamma'$. Now since $T \notin \Gamma'$, it is not the unique minimum spanning tree under the scenario with costs on edges in T at their lower bounds and costs on other edges at their upper bounds. Thus, by Corollary 3.1, T is never a unique minimum spanning tree. So by Lemma 3.3 there exists a tree T' such that $c_T^{s'} \geqslant c_{T'}^{s'}$ for all scenarios s', and such that T' is the unique minimum spanning tree for some scenario. Thus, by Corollary 3.1, T' must be the unique minimum spanning tree when costs of edges in T' are at their lower bounds and costs of other edges are at their upper bounds, i.e., it must be that $T' \in \Gamma'$. But $c_T^{s'} \geqslant c_{T'}^{s'}$ for all scenarios s', in particular $c_T^s \geqslant c_{T'}^s$. This contradicts that cost of T is smaller than the costs of all spanning trees in Γ'. So Γ' is a unionwise permanent set. Γ' is minimal since any spanning tree in Γ' is the unique minimum spanning tree for some scenario. □

Corollary 3.2. *Minimal unionwise permanent set is unique.*

Proof. All of the trees in Γ' defined in the statement of Theorem 3.1 must be in any unionwise permanent set, since they are unique minimum spanning trees with respect to some scenario. Furthermore, since they form on their own a minimal unionwise permanent set, they must be the only such set. □

As the minimal unionwise permanent set is unique, when we refer to the minimal unionwise permanent set, we refer to the set Γ' which is defined in Theorem 3.1. Now we are in a position to characterize strong edges in terms of the minimal unionwise permanent set.

Proposition 3.3. *An edge is strong if and only if all spanning trees in the minimal unionwise permanent set share that edge.*

Proof. If an edge e is strong, it is on a minimum spanning tree for all scenarios. Since each spanning tree in the minimal unionwise permanent set is the unique minimum spanning tree for some scenario, e should lie on all of them.

If an edge e is shared by all spanning trees in the minimal unionwise permanent set, then it is on a minimum spanning tree for all scenarios, thus it is strong. □

Proposition 3.4. *There exists a relative robust spanning tree in the minimal unionwise permanent set Γ'.*

Proof. Assume none exists. Then there is a relative robust spanning tree $T \in \Gamma \setminus \Gamma'$. By Proposition 3.2, T is a weak tree, and since $T \notin \Gamma'$, it is not the unique minimum spanning tree for the scenario with costs of edges in T at their lower bounds and costs of other edges at their upper bounds. So by Corollary 3.1, T is not the unique minimum spanning tree for any scenario. Then by Lemma 3.3, there exists a spanning tree T' such that $c_T^s \geqslant c_{T'}^s$ for all scenarios s, and such that T' is the unique minimum spanning tree for some scenario. The latter implies that $T' \in \Gamma'$. Consider a relative worst case scenario $s_{T'}$ for spanning tree T'. We have

$$d_{T'} = c_{T'}^{s_{T'}} - c_{T^*(s_{T'})}^{s_{T'}} \leqslant c_T^{s_{T'}} - c_{T^*(s_{T'})}^{s_{T'}} \leqslant d_T.$$

Since spanning tree T is a relative robust spanning tree, we have $d_{T'} = d_T$ therefore T' is also a relative robust spanning tree. □

The following is now a corollary of Propositions 3.3 and 3.4.

Corollary 3.3. *There exists a relative robust spanning tree such that every strong edge in the graph lies on the tree.*

Theorems 2.2 and 2.3 show that all weak and strong edges in the graph can be identified in polynomial time. By Proposition 3.2, we know that every relative robust tree uses only weak edges, and by Corollary 3.3 we know that every strong edge in the graph must lie on some relative robust tree. We can use these results to preprocess the mixed integer programming formulation, as follows. For every $e \in E$ which is not weak, we may set $x_e = 0$, since edges which are not

weak cannot lie on a relative robust tree. For every edge $e \in E$ which is strong, we may set $x_e = 1$, since there exists a relative robust tree which includes all the strong edges in the graph.

3.5. Computational results

We used our MIP formulation (R) to compute the robust spanning tree in complete graphs with $n = 10, 15, 20, 25$. We conduct two experiments on a Pentium II PC with 450 MHz clock speed: (1) We solve the model using the CPLEX 6.5.1 MIP solver without any preprocessing, and (2) we preprocess the problem graph using the results of the previous sections before we feed to the CPLEX solver, i.e., remove the non-weak arcs and set the variables corresponding to strong edges equal to 1. For problems with $n = 10, 15$ and 20 we generated six sets of five problems each with varying interval specifications as follows:

For each edge e, $\underline{c}_e$ is uniformly distributed in the first interval, and $\bar{c}_e$ is uniformly distributed in the second interval, respectively, as listed below:

1. set: [0, 10] and ($\underline{c}_e$, 10]
2. set: [0, 15] and ($\underline{c}_e$, 15]
3. set: [0, 20] and ($\underline{c}_e$, 20]
4. set: [0, 10] and ($\underline{c}_e$, 20]
5. set: [0, 15] and ($\underline{c}_e$, 30]
6. set: [0, 20] and ($\underline{c}_e$, 40].

For the case $n = 25$ we solved only five test problems generated from the first set above as a result of increasing computational time of solving model (R) without any preprocessing. The results are reported in the table below where we give the minimum and maximum number of strong and weak edges, respectively along with average computational times in CPU seconds. The % gain is defined as the ratio of the difference in CPU times to the computation time without preprocessing (see Table 1). The preprocessing procedure which eliminates non-weak and strong edges results in the removal of approximately 50–70% of the edges of the graph as can be seen from the table above. These results show that, on higher dimensions the computational savings from preprocessing almost become a requirement in the solution of the robust tree problem.

4. Conclusion

In this paper, we investigated the robust version of the minimum spanning tree problem where edge costs are represented by intervals. We defined two robustness measures, showed that we can solve the absolute robust tree problem in polynomial time and proposed an MIP formulation for the relative robust tree problem. To preprocess a given graph for the relative robust tree problem, we analyzed edges to distinguish the ones that are on minimum spanning trees for all realizations and the ones that are on minimum spanning trees for some realizations. We presented characterizations for these edges which suggest polynomial time algorithms to decide whether a given edge is weak and strong. Our computational results show that knowing weak and strong edges helps shorten significantly the computation of the relative robust tree.

Acknowledgements

The authors are grateful to an anonymous referee for a very careful reading of the paper and for numerous specific suggestions that improved the presentation tremendously.

Table 1

n	No. of strong edges	No. of weak edges	Without preprocessing	After preprocessing	% gain
10	0–4	17–36	3.92	1.96	50
15	0–5	41–69	131.13	33.09	74.77
20	0–3	66–105	3437.2	693.88	79.81
25	0–5	91–103	27126	2027.6	92.53

References

[1] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[2] A. Ben-Tal, A. Nemirovski, Robust solutions of uncertain linear programs, Oper. Res. Lett. 25 (1999) 1–13.

[3] D.P. Bertsekas, R. Gallagher, Data Networks, Prentice-Hall, Englewood Cliffs, NJ, 1987.

[4] P. Kouvelis, G. Yu, Robust Discrete Optimization and its Applications, Kluwer Academic Publishers, The Netherlands, 1997.

[5] G.L. Kozina, V.A. Perepelista, Interval spanning trees problem: solvability and computational complexity, Interval Comput. 1 (1994) 42–50.

[6] T.L. Magnanti, L. Wolsey, Optimal trees, in: M.O. Ball, et al., (Eds.), Network Models, Handbook in Operations Research and Management Science, Vol. 7, North-Holland, Amsterdam, 1995, pp. 503–615.

[7] J.M. Mulvey, R.J. Vanderbei, S.A. Zenios, Robust optimization of large-scale systems, Oper. Res. 43 (1995) 264–281.

[8] A. Prekopa, Stochastic Programming, Kluwer Academic Publishers, Dordrecht, 1995.

[9] M.H. Demir, B.G. Tansel, G.F. Scheuenstuhl, The network 1-median location with interval data: a parameter space based approach, IIE Transactions, to appear, 2001.