Minimax Flow Tree Problems

Hui Chen[1], Ann Campbell[1], Barrett Thomas[1], and Arie Tamir[2]

[1] Dept. of Management Sciences, Tippie College of Business, The University of Iowa, USA

{hui-chen, ann-campbell, barrett-thomas}@uiowa.edu

[2] Dept. of Statistics and Operations Research, School of Mathematical Sciences, Tel Aviv University, Israel

atamir@post.tau.ac.il

July 5, 2007

Abstract

We examine a class of problems which seeks to find tree-structured networks which minimize the maximum cost among a subset of nodes in a graph. The cost metric is characterized by a series of parameters which can represent distance, flow volume, and delivery deadlines. Derived through variations in problem parameters, we present 17 different problems and discuss their worst-case complexity. Fourteen of the problems are new to the literature. We show that some of the problems are NP-complete and others are polynomially solvable.

Keywords: trees, minimax, flow, complexity, shortest path trees

1 Introduction

To be successful in the face of fierce competition and to meet the growing demand for time-definite services, delivery companies must design their delivery networks to reliably and efficiently meet their promised delivery times. In the near term, however, redesign of the network through the opening of new facilities and the closing of existing facilities is infeasible. Rather, the network can be modified by changing how freight flows through the network. In particular, companies can change which cities have direct connections to one another and which cities must flow freight and packages through intermediate cities in order to make a connection. Each city-to-city connection requires substantial investment, so delivery companies want to minimize the number of direct connections. Consequently, this study will focus on restricting the design of the delivery networks to tree structures, as they connect all of the nodes in the network with the minimum number of connections. In addition to cost minimization, Powell and Koskosidis (1992) note that tree-structured networks are easier networks to manage because there is only one path between each origin and destination pair.

Different tree structures offer different levels of service between customers. Delivery companies may have stricter service requirements or place more emphasis on service between certain sets or kinds of customers, and this should be reflected in their choice of tree-structured network. With this in mind, this paper considers different variants in assumptions and objective function and examines the complexity of each. All of the problems assume a simple, connected, and undirected graph $G = (V, E)$. For each problem, the objective is to minimize the maximum cost between a source node set $S \subseteq V$ and sink node set $U \subseteq V$ and involves two parameters: $f_{ij} \geq 0$ and $H_{ij} \geq 0$. A minimax objective optimizes service by minimizing the worst-case service between customers in the source and sink sets. The definition of the source and sink node set as well as the parameters define each variant in the class, allowing the objective to be expressed in the following general form:

$$\max_{v_i \in S} \max_{v_j \in U} \{f_{ij}(d_T(v_i, v_j) - H_{ij})\}. \tag{1}$$

The choice of source node set and sink node set reflects the combination of customers whose service level is considered in the objective function. For example, if $S = V$ and $U = V$, we want to consider the travel time from every customer to every other customer in the objective. If $S \neq V$ or $U \neq V$, we want to consider the travel time only between a subset of nodes in the objective function. The selection of S and U is thus a reflection of priorities among customers or cities in a delivery network. Next, the parameter f_{ij} reflects a weight placed on the service from customer v_i to customer v_j. This weight may reflect, for example, the flow of packages from city v_i to city v_j, so that cities with heavy flow between them are prioritized in the objective. We may consider simpler versions where f_{ij} is constant for all v_i, v_j pairs (f) or only dependent on v_i (f_i). We will also consider a special case of f_{ij} where it is based on values for f_i and f_j. Finally, the parameter H_{ij} represents a service time commitment from customer v_i to customer v_j. Many delivery providers offer varying levels of service between customers depending on their location and demand, so H_{ij} forces Equation 1 to minimize the maximum violation of these commitments. As with f_{ij}, we can consider simpler versions such as where H_{ij} is zero (0), a constant (H), or where H_{ij} is dependent only on v_i (H_i). Note that when $H_{ij} = 0$ for all v_i, v_j pairs this represents the case where there is no established commitment, and the objective is to simply minimize the maximum travel time between pairs.

Table 1 lists the different problem variants covered in this paper or in the literature and their worst-case complexity status. It should be noted that each of these problems is in $\mathcal{N}P$ as the objective function associated with a given tree can be evaluated in polynomial time. For those problems that can be solved in polynomial time, we set $n = |V|$, $m = |E|$, $k = |S|$, $l = |U|$ and $\bar{n} = |S| + |U| = k + l$. Without loss of generality, we assume that $k > 1$ and $l > 1$. Otherwise, if either S or U consists of a single node, then the optimal tree is a shortest path spanning tree rooted at that node.

In this paper, we examine the complexity of these minimax flow tree problems and find some to be polynomial and others to be $\mathcal{N}P$-Complete. In Section 2, we review the related literature. In Section 3, we define relevant notation for our study. Section 4 studies the algorithms for the tractable problems. In Section 5, the $\mathcal{N}P$-Completeness of the intractable problems is established. Finally, we discuss directions for future work in Section 6.

	Problem	S	U	f_{ij}	H_{ij}	Complexity	Reference
1	MDST	$S = V$	$U = V$	1	0	$O(mn + n^2 \log n)$	Handler and Mirchandani (1979) Kariv and Hakimi (1979) Hassin and Tamir (1995)
2	k-MEST	$S \subseteq V$	$U = V$	1	0	$O(mn + n^2 \log n)$	Wu (2004)
3	MEMT	$S \subseteq V$	$U \subseteq V$	1	0	$O(mn + n^2 \log n)$	Krumme and Fragopoulou (2001), this paper
4	UMVT	$S \subseteq V$	$U \subseteq V$	1	H	$O(mn + n^2 \log n)$	this paper
5	NMVT	$S \subseteq V$	$U \subseteq V$	1	H_i	$O(mn + n^2 \log n)$	this paper
6	PMVT	$S \subseteq V$	$U \subseteq V$	1	H_{ij}	$\mathcal{N}P$-Complete	this paper
7	NF-MDST	$S = V$	$U = V$	f_i	0	$O(mn \log n)$	this paper
8	NF-MEMT	$S \subseteq V$	$U \subseteq V$	f_i	0	$O(mn + n^2 \log n + m\bar{n} \log \bar{n})$	this paper
9	NF-UMVT	$S \subseteq V$	$U \subseteq V$	f_i	H	$O(mn + n^2 \log n + m\bar{n} \log \bar{n})$	this paper
10	NF-NMVT	$S \subseteq V$	$U \subseteq V$	f_i	H_i	$O(mn + n^2 \log n + m\bar{n} \log \bar{n})$	this paper
11	NF-PMVT	$S \subseteq V$	$U \subseteq V$	f_i	H_{ij}	$\mathcal{N}P$-Complete	this paper
12	SF-MDST	$S = V$	$U = V$	$\frac{f_i f_j}{f_i + f_j}$	0	$O(mn \log n)$	this paper
13	PF-MDST	$S = V$	$U = V$	f_{ij}	0	$\mathcal{N}P$-Complete	this paper
14	PF-MEMT	$S \subseteq V$	$U \subseteq V$	f_{ij}	0	$\mathcal{N}P$-Complete	this paper
15	PF-UMVT	$S \subseteq V$	$U \subseteq V$	f_{ij}	H	$\mathcal{N}P$-Complete	this paper
16	PF-NMVT	$S \subseteq V$	$U \subseteq V$	f_{ij}	H_i	$\mathcal{N}P$-Complete	this paper
17	PF-PMVT	$S \subseteq V$	$U \subseteq V$	f_{ij}	H_{ij}	$\mathcal{N}P$-Complete	this paper

Table 1: Problem Variants

2 Literature Review

The Minimum Diameter Spanning Tree Problem (MDST) is the problem of finding a spanning tree T of G with minimum diameter, where the *diameter* of T is defined as the longest path in T among the paths between all pairs of nodes in V. Ho et al. (1991) consider the case where the graph G is a complete Euclidean graph, induced by a set of n points in the Euclidean plane. Ho et al. call this special case the *geometric* MDST. Ho et al. develop an $O(n^3)$ algorithm to find a spanning tree of minimum diameter of a Euclidean graph, and they mention that these results extend to any graph whose edge lengths satisfy the triangle inequality. Handler and Mirchandani (1979) (see also Hassin and Tamir (1995)) consider the general case where the edge lengths in a general graph do not necessarily satisfy the triangle inequality. They observe that MDST is identical to the well studied Absolute 1-Center Problem introduced by Hakimi (1964), where the absolute center of a graph is a point in the graph whose maximum shortest distance from any node on the graph is minimal. Hassin and Tamir (1995) note that MDST can then be solved by the existing algorithms for the Absolute 1-Center Problem on a general graph in $O(mn+n^2 \log n)$ time (Kariv and Hakimi, 1979).

Given the source set $S \subseteq V$ such that $|S| = k$ and the sink set $U = V$ in G, the k-Source Minimum Max-Eccentricity Spanning Tree Problem (k-MEST) is defined as finding a spanning tree to minimize the maximal source eccentricities among k sources, where the *source eccentricity* in a spanning tree T is the longest distance from a source to all sink nodes. Farley et al. (2000) explore the variant with uniform edge lengths. First, they prove that there exists either a vertex x or an edge (y, z), such that the shortest path tree rooted from either x or the midpoint of (y, z) minimizes the maximal source eccentricity. They introduce an exact polynomial algorithm for k-MEST with uniform edge length based on solving many shortest path problems. McMahan and Proskurowski (2004) generalize the result for graphs with general edge lengths and show there exists a minimum max-eccentricity spanning tree which is a shortest path tree rooted at either a vertex or a created vertex lying on an edge. McMahan and Proskurowski (2004) then establish an exact polynomial algorithm with running time $O(n^3 + mn \log n)$ for a general graph. Krumme and Fragopoulou (2001) study the Minimum Eccentricity Multicast Tree (MEMT), which is a generalized version of k-MEST with the sink set $U \subseteq V$. By identifying the appropriate edge that can be cut to create a new vertex from which to construct an optimal shortest path spanning tree, Krumme and Fragopoulou offer a polynomial algorithm with running time $O(mn + n^3)$ for MEMT, which computes the all pair shortest path distances in $O(n^3)$ time. We note that, if we apply Fibonacci heaps to compute the all pair shortest path distances (Fredman and Tarjan, 1987), the complexity of the algorithm in (Krumme and Fragopoulou, 2001) improves to $O(mn + n^2 \log n)$ time. Based on a similar idea as Krumme and Fragopoulou (2001), Wu (2004) independently describes another polynomial algorithm with $O(mn + n^2 \log n)$ time for the k-MEST.

Another problem related to source eccentricity is the k-Source Minimum Sum-Eccentricity Spanning Tree Problem (k-SSET). This problem seeks to minimize the sum of the eccentricities of the k source nodes instead of the maximum eccentricity of the k source nodes. Connamacher and Proskurowski (2003) demonstrate that k-SSET is polynomially solvable.

Fragopoulou et al. (2005) design an $O(mn \log n)$ time algorithm to solve the k-SSET.

The Optimal Communication Spanning Tree Problem (OCST) is the only literature known to the authors which considers flow in conjunction with tree structures. In this problem, given a set of communication requirements r_{ij} between v_i and v_j in graph $G = (V, E)$, the cost measure of a spanning tree is defined as follows. For a pair of nodes v_i and v_j, there is a unique path in the spanning tree between them. The cost of communication for the pair of nodes v_i and v_j is r_{ij} multiplied by the distance of the path, $d_T(v_i, v_j)$. Summing over all $\binom{n}{2}$ pairs of nodes, we have the cost of the spanning tree. Hu (1974) introduces the OCST, and Johnson et al. (1978) show that OCST is $\mathcal{NP}$-hard. Ahuja and Murty (1987) provide an exact algorithm for small problems. Peleg (1997) and Peleg and Reshef (1998) establish polynomial-time approximation algorithms for OCST. Hu (1974) also studies the special OCST, which is labeled as ORST and assumes the length of every arc is one, and demonstrates a polynomial algorithm to solve the ORST in a complete graph. Hu (1974) introduces another special OCST, which is labeled as ODST and assumes the required flows are one for all pairs of nodes. Wu (2002) proves that the k-source ODST (k-ODST), where there are k source nodes, is $\mathcal{NP}$-hard even if $k = 2$ for a metric graph. Additional ODST research focuses on finding approximation algorithms (Wu et al., 1999, 2000a,b).

Also related to the research in this paper is the Hop-Constrained Minimum Spanning Tree Problem (HC-MST). Referring to each arc as a hop, hop constraints limit the number of hops between nodes and can be viewed as a distance constraint. In the literature, the objective of HC-MST is to find the minimum spanning tree T such that the number of the hops (arcs) in the unique path from a *single* root node to any other node is no greater than a constant number H. Dahl (1998) proves that the 2-hop constrained minimum spanning tree problem is $\mathcal{NP}$-Hard. Manyem and Stallmann (1996) show that HC-MST is not APX such that it is not possible to find a polynomial time heuristic which guarantees a constant approximation bound. Many different IP formulations and related solution approaches have been studied by Gouveia (1995), Gouveia (1996), Gouveia (1998), and Gouveia and Requejo (2001). Voss (1999) uses tabu search to improve a feasible initial solution. Althaus et al. (2005) build an algorithm with an $O(\log n)$-approximation in running time $O(n^5 k)$ for the k-hop constrained minimum spanning tree problem.

3 Notation

In this section, we describe the notation used throughout our paper. Let $G = (V, E)$ be a simple, connected, and undirected graph, where V is the set of nodes v_i and E is the set of edges e_{ij}. Without loss of generality, we assume $i < j$ for every edge e_{ij}. Each edge $e_{ij} \in E$ is associated with a positive length l_{ij} and is assumed to be rectifiable. We note that the literature often equates arc length with distance in the network. Because of our focus on service-time commitments, we consider these lengths to be travel times. However, the two measures of distance are equivalent for purposes of the results presented in this paper. Thus, our service-time commitment can also be thought of as a distance restriction.

A node which is incident to only one edge is a *leaf node*, otherwise it is an *internal node*. An edge incident to a leaf node is a *leaf edge*. A path from node v_i to node v_j is denoted as the v_i-v_j *path*. A *spanning tree* of G is a connected subgraph $T = (V, E_T \subseteq E)$ without cycles. The distance between node v_i and node v_j in T, $d_T(v_i, v_j)$, is the sum of the lengths of the edges which are in the unique v_i-v_j path. For two nodes v_i and v_j, the shortest path distance between them in a graph G is denoted by $d_G(v_i, v_j)$.

We define nodes $v_i \in V$ and $v_j \in V$ as *adjacent* in the tree T if $e_{ij} \in E_T$. The *endpoints* of a path are the two nodes at which the path begins and ends. In a path, all the other nodes except the two endpoints are *internal nodes of the path*. Let v_{i_1}-v_{i_2}-...-v_{i_a} be path A and v_{j_1}-v_{j_2}-...-v_{j_b} be path B in the graph. We say that path A and path B are *disjoint* if the sets $\{v_{i_1}, ..., v_{i_a}\}$ and $\{v_{j_1}, ..., v_{j_b}\}$ are disjoint. If the above two sets are not disjoint, we say that path A and path B *intersect*, and each node in their intersection is called an *intersection node*.

We define a *point* as a location in the graph which could be either a node or a location on an edge. We will refer to a point on a rectifiable edge by its distances from the two nodes of the edge. Let $I(G)$ denote the continuum set of points on the edges of G. The edge lengths of G induce a distance function on $I(G)$. For any two points $x, y \in I(G)$, $d_G(x, y)$ will denote the length of a shortest path in $I(G)$ connecting x and y. We refer to $I(G)$ as the *network* *metric space* induced by G and its edge lengths.

Recall, we let $S \subseteq V$ and $U \subseteq V$ denote the source node set and sink node set, respectively. Further, we let $k = |S|$, $l = |U|$ and $\bar{n} = |S| + |U| = k + l$. For a spanning tree T, we define the *longest intra-sink path* of T, D_T, to be a longest path in T among all the paths connecting a pair of sink nodes in U. The length of D_T is denoted by δ_T. The *center point* of a tree is the middle point of a longest intra-sink path. For a source node $v_i \in S$, define $h_i^T = \max_{v_j \in U}\{d_T(v_i, v_j)\}$.

4 The Tractable Problems

This section presents polynomial-time algorithms for the tractable problems for which no algorithm exists in the literature. For any problem with some $H_{ij} \neq 0$, we will refer to the problem as a Minimax Violation Tree Problem (MVT).

4.1 Uniform Commitment Minimax Violation Tree (UMVT)

The service-time commitment in the UMVT is the same for every pair of source and sink nodes ($H_{ij} = H$). Here, the maximum violation of a source is achieved by the longest travel-time or distance from the source to all sinks in the tree, which is the eccentricity of the source in the tree. As MEMT seeks to minimize the maximal source eccentricities, MEMT and UMVT are closely related to each other. Theorem 1 states the relationship between MEMT and UMVT.

Theorem 1 *Any optimal solution to MEMT is an optimal solution to UMVT.*

Proof: For UMVT, the objective is to minimize

$$\max_{v_i \in S} \max_{v_j \in U}\{d_T(v_i, v_j) - H_{ij}\} \quad = \quad \max_{v_i \in S} \max_{v_j \in U}\{d_T(v_i, v_j)\} - H.$$

For MEMT where $H_{ij} = 0$, the objective is to minimize

$$\max_{v_i \in S} \max_{v_j \in U} \{d_T(v_i, v_j) - H_{ij}\} \quad = \quad \max_{v_i \in S} \max_{v_j \in U} \{d_T(v_i, v_j)\}.$$

Thus, if a tree minimizes the objective of MEMT, it also minimizes the objective for UMVT. ■

Given this result, UMVT can be solved by the algorithms for MEMT. Thus, UMVT can be solved in $O(mn + n^2 \log n)$ time (Krumme and Fragopoulou, 2001). In addition, as the UMVT with $S = U = V$ is the MDST, the special case with $S = U = V$ also takes $O(mn + n^2 \log n)$ time to find an optimal solution.

4.2 Node Commitment Minimax Violation Tree (NMVT)

For NMVT, where $H_{ij} = H_i$, we shall show that it can be transformed into a special MEMT and solved by an algorithm for MEMT.

For an instance of NMVT, we can transform it to a special case of MEMT as follows. First, we build the input graph G' for MEMT from G. Let $H = 1 + \max_{v_i \in S}\{H_i\}$. For each source node $v_i \in S$, we create a new node v'_i and connect v'_i and v_i by a new edge $e_{v_i v'_i}$ with the edge length $l_{v_i v'_i} = H - H_i > 0$. Let G' be the graph with node set V' and edge set E' such that $V' = V \cup \{v'_i : v_i \in S\}$ and $E' = E \cup \{e_{v_i v'_i} : v_i \in S\}$. Now, $|V'| = n + k$ and $|E'| = m + k$. Next, we set the source and sink node sets for MEMT such that $S' = \{v'_i : v_i \in S\}$ and $U' = U$, respectively. Clearly, all source nodes in S' are leaf nodes in G'. Figure 1 shows an example of NMVT and the transformed MEMT.

For MEMT, if we denote $C(T)$ as the maximal source eccentricity in a tree T, then $C(T) = \max_{v_i \in S}\{\max_{v_j \in U}\{d_T(v_i, v_j)\}\}$. For NMVT, if we let $F(T)$ denote the maximal violation of a tree T, then $F(T) = \max_{v_i \in S}\{\max_{v_j \in U}\{d_T(v_i, v_j) - H_i\}\}$, where H_i is the service-time commitment for the source $v_i \in S$. Let T' be any solution tree for the transformed MEMT of G'. As all the source nodes in S' are leaf nodes in G', they are still leaf nodes in T'. If we delete all the source nodes in S' and all their adjacent leaf edges from T', we can obtain

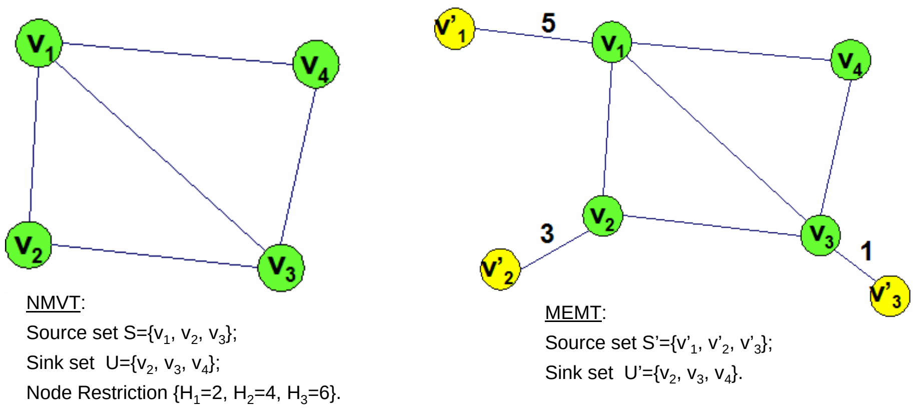

Figure 1: An example of NMVT and the transformed special MEMT

a new tree T of G. Conversely, given any T of G, we can also construct a tree T' of G' by adding all the source nodes of S' and all their adjacent leaf edges in G' to T. We have the following observation for $C(T')$ and $F(T)$.

Lemma 1 *$C(T') = F(T) + H$, where $H = 1 + \max_{v_i \in S}\{H_i\}$.*

Proof: Clearly, $d_T(v_i, v_j) = d_{T'}(v_i, v_j)$ for any $v_i, v_j \in V$. Because $v_i \in S$ and $v'_i \in S'$ are one to one corresponding and $U' = U$,

$$\begin{aligned} C(T') &= \max_{v'_i \in S'} \max_{v_j \in U'} \{d_{T'}(v'_i, v_j)\} \\ &= \max_{v_i \in S} \max_{v_j \in U} \{d_{T'}(v_i, v_j) + l_{v_i v'_i}\} \\ &= \max_{v_i \in S} \max_{v_j \in U} \{d_T(v_i, v_j) + H - H_i\} \\ &= F(T) + H. \end{aligned}$$

■

We present the following theorem that follows directly from Lemma 1.

Theorem 2 *A tree T is an optimal solution to the NMVT in G if and only if the corresponding T' is optimal for the transformed MEMT in G'.*

Given this result, the algorithm for MEMT can be applied to NMVT. As an optimal solution for MEMT can be found in $O(mn + n^2 \log n)$ time (Krumme and Fragopoulou, 2001) and $|V'| = n + k \leq 2n$ and $|E'| = m + k \leq m + n$, NMVT can be solved in $O(mn + n^2 \log n)$ computation time as well.

4.3 Node Flow Minimax Violation Tree (NF-MVT)

Now we consider the problem variants with node flow, where $f_{ij} = f_i$. We shall study four polynomially solvable variants in this class. These variants are NF-MDST, NF-MEMT, NF-UMVT, and NF-NMVT. From Table 1, we can see that NF-NMVT is the most general of these variants. Therefore, we shall first focus on establishing a polynomial algorithm for NF-NMVT and later adjust it for the other three variants.

4.3.1 Preliminaries

To begin, we present the following lemma from Connamacher and Proskurowski (2003), which characterizes the longest travel time or distance from a source node to all the sink nodes in a tree. A similar result is also described by Handler (1973), for the case where $U = V$.

Lemma 2 *Given a spanning tree T, let sink nodes v_1 and v_2 be the two endpoints of D_T. For any source node $v_i \in S$, h_i^T is obtained by either the v_i-v_1 path or the v_i-v_2 path.*

Handler (1973) further clarifies from a node to which other node it obtains the longest travel time or distance in a tree. We extend this result in the next lemma, which specifies which endpoint of D_T defines h_i^T for any source $v_i \in S$ and the value of h_i^T. The proof follows directly from Handler (1973) and is omitted.

Lemma 3 *Given a spanning tree T, let sink nodes v_1 and v_2 be the two endpoints of D_T. Let o be the center point of T. Let $v_i \in S$ be any source node.*

When v_i is not on D_T, let node v_s be the node on D_T where the path connecting v_i with D_T intersects D_T. If v_s lies in the v_1-o path, then $h_i^T = d_T(v_i, v_2)$. If v_s lies in the v_2-o path, then $h_i^T = d_T(v_i, v_1)$. Similarly, when v_i is on D_T, if v_i lies on the v_1-o path, then $h_i^T = d_T(v_i, v_2)$. If v_i lies on the v_2-o path, then $h_i^T = d_T(v_i, v_1)$.

In addition, $h_i^T = d_T(v_i, o) + \frac{1}{2}\delta_T$.

An important result for the shortest path tree rooted at the center point of an optimal tree for k-MEST is proved by Farley et al. (2000) and restated by Krumme and Fragopoulou (2001) and McMahan and Proskurowski (2004). We now extend the result to a more general form and revisit its proof as it leads to the development of our algorithm.

Lemma 4 *Given a spanning tree T in graph G and its center point o, let tree T_o be the shortest path tree rooted at o in G. For any source node $v_i \in S$, $h_i^{T_o} \leq h_i^T$.*

Proof: For any source $v_i \in S$ and any sink node $v_j \in U$, because of the triangle inequality, the shortest paths from o in T_o, the definition of longest intra-sink path, and Lemma 3, we obtain

$$\begin{aligned} d_{T_o}(v_i, v_j) &\leq d_{T_o}(v_i, o) + d_{T_o}(o, v_j) \\ &\leq d_T(v_i, o) + d_T(o, v_j) \\ &\leq d_T(v_i, o) + \frac{1}{2}\delta_T \\ &= h_i^T. \end{aligned}$$

Therefore, $\max_{v_j \in U}\{d_{T_o}(v_i, v_j)\} = h_i^{T_o} \leq h_i^T$. ■

For the same spanning trees T and T_o, we can also characterize the relationship between the length of the longest intra-source paths in T and T_o.

Lemma 5 *Given a spanning tree T in graph G and its center point o, let tree T_o be the shortest path tree rooted at o in G. Then, $\delta_{T_o} \leq \delta_T$.*

Proof: Let $v_{j_1}, v_{j_2} \in U$ be any two sinks. By the triangle inequality, shortest paths from o in T_o, and the longest intra-sink path,

$$\begin{aligned} d_{T_o}(v_{j_1}, v_{j_2}) &\leq d_{T_o}(v_{j_1}, o) + d_{T_o}(o, v_{j_2}) \\ &\leq d_T(v_{j_1}, o) + d_T(o, v_{j_2}) \\ &\leq \frac{1}{2}\delta_T + \frac{1}{2}\delta_T \\ &= \delta_T. \end{aligned}$$

This result implies

$$\delta_{T_o} = \max_{v_{j_1} \in U} \max_{v_{j_2} \in U} \{d_{T_o}(v_{j_1}, v_{j_2})\} \leq \delta_T.$$

■

For NF-NMVT, we now further describe the relationship of h_i^T for any source $v_i \in S$ in a tree T and the maximal violation for v_i in T.

Lemma 6 *For NF-NMVT, the maximal violation for any source $v_i \in S$ in a spanning tree T is equal to $f_i h_i^T - f_i H_i$.*

Proof: Let H_i and f_i be the node restriction and node flow for v_i, respectively. The maximal violation of v_i in T equals

$$\begin{aligned} \max_{v_j \in U}\{f_i(d_T(v_i, v_j) - H_i)\} &= f_i \max_{v_j \in U}\{d_T(v_i, v_j)\} - f_i H_i \\ &= f_i h_i^T - f_i H_i. \end{aligned}$$

■

This result implies that, given the node restriction H_i and the node flow f_i, the maximal violation for any source $v_i \in S$ in T depends on only h_i^T.

Based on Lemma 4 and Lemma 6, we can show that for any optimal solution to NF-NMVT, there exists a shortest path spanning tree which is also optimal.

Theorem 3 *For NF-NMVT, given an optimal tree T^*, let o be the center point of T^*. The shortest path spanning tree rooted at o, T_o, is also an optimal tree.*

Proof: The result follows directly from Lemmas 4 and 6. ■

In the light of Theorem 3, given an optimal tree T^* and its center point o, we can build another optimal tree which is the shortest path tree T_o rooted at o. Because the longest intra-sink path of T_o may move, however, the center point of T_o may no longer be located at o. In fact, o is not even guaranteed to be on a longest intra-sink path. Nevertheless, we can prove that there does exist an optimal shortest path tree whose root coincides with its center point.

Theorem 4 *For NF-NMVT, there exists an optimal shortest path spanning tree whose root is also its center point.*

Proof: By Theorem 3, for any optimal tree T^* with center point r, we can generate a new optimal tree T_r, which is a shortest path spanning tree rooted at r. If r is the center point of T_r, we are done. If r is not the center point of T_r, we must construct another optimal shortest path tree whose root coincides with its center point.

Let o be the center point of T_r. Let the v_1-v_2 path be a longest intra-sink path D_{T_r} of T_r with length δ_{T_r}. Then, o divides T_r into two subtrees, T_r^1 and T_r^2, containing v_1 and v_2, respectively. Because r is either in T_r^1 or in T_r^2, without loss of generality, let r be in T_r^1. If o is a node rather than a point, we let o belong to T_r^2. As v_2 is in T_r^2, o is on the r-v_2 path. Thus, by the principle of optimality, a shortest path spanning tree T_o rooted at o contains T_r^2 such that $T_r^2 \subset T_o$. Let S_1 and S_2 be two source node subsets, such that S_1 and S_2 contain all the sources whose violations are equal to the maximal violation among all the sources in T_r^1 and T_r^2, respectively. The maximal violation of T_r is obtained by either the sources in S_1 or those in S_2. For example, let v_{i_1} and v_{i_2} be two sources such that $v_{i_1} \in S_1$ and $v_{i_2} \in S_2$. By Lemma 3, the maximal violation for v_{i_1} and v_{i_2} are obtained by the v_{i_1}-v_2 path and the v_{i_2}-v_1 path, respectively. This situation is shown in Figure 2.

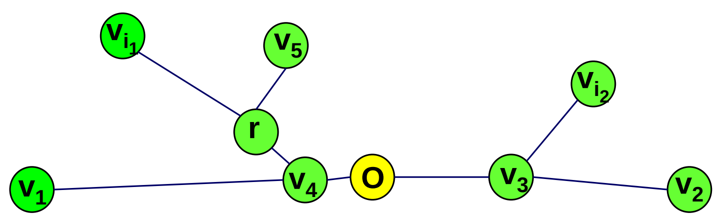

Figure 2: Two subtrees of T_r separated by its center point o

Given the above characterization of T_r, we construct a shortest path spanning tree T_o rooted at o, which must also be optimal by Theorem 3. Without loss of generality, we assume that, in the construction of T_o, if for a node $v \in V$, the path from o to v in T_r is a shortest path, then the path is maintained in the new spanning tree T_o. We now demonstrate the existence of an optimal shortest path tree rooted at its center point.

Continuing the proof, we next demonstrate that the maximum violation in T_o must be obtained by a source belonging to either the set S_1 or S_2. Let F^* denote the optimal maximum violation. By construction of S_1 and S_2, $\max_{v_i \notin S_1 \cup S_2}\{f_i h_i^{T_r} - f_i H_i\} < \max_{v_i \in S_1 \cup S_2}\{f_i h_i^{T_r} -$

$f_iH_i\} = F^*$ for all sources $v_i \in S$. Therefore, because $h_i^{T_o} \leq h_i^{T_r}$ for all sources $v_i \in S$ by Lemma 4, we have

$$\max_{v_i \notin S_1 \cup S_2} \{f_i h_i^{T_o} - f_i H_i\} \leq \max_{v_i \notin S_1 \cup S_2} \{f_i h_i^{T_r} - f_i H_i\} < F^*.$$

Thus, the maximal violation in T_o must be still obtained by a source either in S_1 or in S_2. In addition, by the optimality of T_o, there exists such a source $v_i \in S_1 \cup S_2$ that $f_i h_i^{T_o} - f_i H_i = F^*$ and thus $h_i^{T_o} = h_i^{T_r}$.

Suppose the maximal violation in T_o is obtained by a source $v_{i_2} \in S_2$. We will show that in this case o is also the center point of T_o. Because, by Lemma 2, $h_{i_2}^{T_r}$ is obtained by the v_{i_2}-v_1 path in T_r and T_r^2 is contained in T_o, $h_{i_2}^{T_o}$ must be obtained by the path from v_{i_2} to a sink v_j belonging to T_r^1 in T_r. Otherwise, the v_{i_2}-v_1 path in T_r would not have led to the maximal violation for source v_{i_2}.

We now show that the v_j-o path in T_o contains no nodes belonging to T_r^2 except o. Suppose the v_j-o path in T_o contains some nodes belonging to T_r^2. By our assumption that T_o maintains shortest paths from o that also exist in T_r, we know that $d_{T_o}(o, v_j) < d_{T_r}(o, v_j)$. Thus, because $d_{T_r}(o, v_j) \leq \frac{1}{2}\delta_{T_r}$, we also have that $d_{T_o}(o, v_j) < \frac{1}{2}\delta_{T_r}$. Then,

$$\begin{aligned} h_{i_2}^{T_o} &= d_{T_o}(v_{i_2}, v_j) \\ &\leq d_{T_o}(o, v_j) + d_{T_o}(o, v_{i_2}) \\ &< \frac{1}{2}\delta_{T_r} + d_{T_o}(o, v_{i_2}) \\ &= h_{i_2}^{T_r}, \end{aligned}$$

where the first inequality follows from the triangle inequality and the last equality from Lemma 3. This result creates a contradiction. Therefore, the v_j-o path in T_o must consist entirely of nodes originally belonging to T_r^1.

As the v_j-o path in T_o contains no nodes belonging to T_r^2 except o and the v_{i_2}-o path in T_o contains nodes belonging to only T_r^2, $h_{i_2}^{T_o} = d_{T_o}(v_j, v_{i_2}) = d_{T_o}(o, v_{i_2}) + d_{T_o}(o, v_j)$. Moreover, because $h_{i_2}^{T_r} = d_{T_r}(o, v_{i_2}) + \frac{1}{2}\delta_{T_r} = d_{T_o}(o, v_{i_2}) + \frac{1}{2}\delta_{T_r}$ and $h_{i_2}^{T_r} = h_{i_2}^{T_o}$, we have that $d_{T_o}(o, v_j) = \frac{1}{2}\delta_{T_r}$. Therefore, as the v_2-o path contains nodes belonging to only T_r^2 in T_o, $d_{T_o}(v_j, v_2) =$

$d_{T_o}(v_j, o) + d_{T_o}(o, v_2) = \frac{1}{2}\delta_{T_r} + \frac{1}{2}\delta_{T_r} = \delta_{T_r}$. By Lemma 5, the v_2-v_j path is a longest intra-sink path of T_o and consequently o must be center point of D_{T_o}. The result then follows.

We are left to consider the case that a source $v_{i_1} \in S_1$ achieves the optimal violation in T_o. We will show that in this case the center point o_1 of T_o is on the o-v_2 path in T_r^2 and $d_{T_r}(o_1, v_2) \leq d_{T_r}(o, v_2)$. We first show that $d_{T_o}(v_{i_1}, o) = d_{T_r}(v_{i_1}, o)$. Because T_o is a shortest path tree rooted at o, $d_{T_o}(v_{i_1}, o) \leq d_{T_r}(v_{i_1}, o)$. Suppose $d_{T_o}(v_{i_1}, o) < d_{T_r}(v_{i_1}, o)$. For any sink $v_j \in U$,

$$
\begin{aligned}
d_{T_o}(v_{i_1}, v_j) &\leq d_{T_o}(v_{i_1}, o) + d_{T_o}(o, v_j) \\
&< d_{T_r}(v_{i_1}, o) + d_{T_r}(o, v_j) \\
&\leq d_{T_r}(v_{i_1}, o) + \frac{1}{2}\delta_{T_r} \\
&= h_{i_1}^{T_r}.
\end{aligned}
$$

The derivation implies that $h_{i_1}^{T_o} = \max_{v_j \in U}\{d_{T_o}(v_{i_1}, v_j)\} < h_{i_1}^{T_r}$, which is a contradiction. Therefore, $d_{T_o}(v_{i_1}, o) = d_{T_r}(v_{i_1}, o)$. This result implies that the v_{i_1}-o path and consequently the v_{i_1}-v_2 path in T_r are kept in T_o, such that $d_{T_o}(v_{i_1}, v_2) = d_{T_r}(v_{i_1}, v_2) = h_{i_1}^{T_r} = h_{i_1}^{T_o}$. That is, the v_{i_1}-v_2 path is a longest path from v_{i_1} to all sinks in T_o.

By Lemma 3, the longest path from source v_{i_1} to all sinks in T_o must pass the center point of T_o and end at one endpoint of D_{T_o}. Then, v_2 is one endpoint of D_{T_o} and o_1 lies in the v_{i_1}-v_2 path. In addition, because $\delta_{T_o} \leq \delta_{T_r}$ by Lemma 5, we have

$$
d_{T_o}(o_1, v_2) = \frac{1}{2}\delta_{T_o} \leq \frac{1}{2}\delta_{T_r} = d_{T_r}(o, v_2)
$$

Therefore, o_1 lies in the o-v_2 path in T_o.

By the above, we can iteratively build optimal shortest path trees rooted at the center point of former optimal shortest path tree. If the center point of the new tree does not coincide with its root, then it is closer to v_2 on the v_{i_1}-v_2 path. Although in each such construction, the center point $o_{\theta+1}$ in the new tree T_θ moves a positive distance towards v_2 , $o_{\theta+1}$ can never reach v_2 because $d_{T_\theta}(o_{\theta+1}, v_2) = \frac{1}{2}\delta_{T_\theta}$ must be greater than 0. Therefore, due to the fact that there is a finite number of tree center points (Kariv and Hakimi, 1979), after a finite number of such iterations, the center point cannot move any further towards v_2. In other words, the center point will eventually coincide with the root of an optimal shortest path tree before v_2 is reached. ■

Theorem 4 characterizes a special property of the structure of an optimal tree for NF-NMVT. We now provide the notation for the special tree set containing all the trees with this property. Let $\mathcal{T}^{SP}$ be the set of all the shortest path trees rooted at a point in $I(G)$. Then, $\mathcal{T}_o^{SP}$ denotes the subset of $\mathcal{T}^{SP}$ containing all the trees whose roots coincide with their center point. Based on Theorem 4, we can identify the optimal tree that minimizes the maximal violation for NF-NMVT in Corollary 1.

Corollary 1 *The tree with the minimax violation among all the trees in $\mathcal{T}_o^{SP}$ is the optimal tree for NF-NMVT.*

However, the set $\mathcal{T}_o^{SP}$ is potentially infinite in size. Fortunately, Kariv and Hakimi (1979) provide an efficient way for us to further examine the set $\mathcal{T}_o^{SP}$ by studying the points on an edge which could be the root of a shortest path tree in $\mathcal{T}_o^{SP}$. Before introducing these results, the following definitions are necessary. In a connected graph G, for an arbitrary edge e_{pq}, let x be a point on e_{pq}. Let $d_p(x)$ and $d_q(x)$ denote the distances along the edge from v_p and v_q to x, respectively, such that $d_p(x) + d_q(x) = l_{pq}$. For any node $v \in V$, the shortest path distance to x in G is $d_G(v, x) = \min\{d_G(v, v_p) + d_p(x), d_G(v, v_q) + d_q(x)\}$. If we denote $I_{pq} \subseteq I(G)$ as all the points on the line segment of e_{pq}, then for each node $v \in V$, the distance function $d_G(v, x)$, such that $x \in I_{pq}$, is a roof function with slopes 1 or -1. We define a *bottleneck point* γ_v on e_{pq} for each node $v \in V$ as the unique maximum point of the distance function $d_G(v, x)$. There are at most $O(n)$ different bottleneck points on e_{pq} and the location of γ_v on e_{pq} can be decided by $d_p(\gamma_v) = \frac{1}{2}(d_G(v_q, v) - d_G(v_p, v) + l_{pq})$. Further, given any point $x \in I_{pq}$, we can define the maximum distance function from all sinks $v_j \in U$ to x as

$$D(x) : I_{pq} \to R \quad s.t. \quad D(x) = \max_{v_j \in U}\{d_G(v_j, x)\}.$$

For example, the solid line in Figure 3 represents the maximum distance function for an edge e_{pq}. Kariv and Hakimi (1979) provide the following properties for $D(x)$:

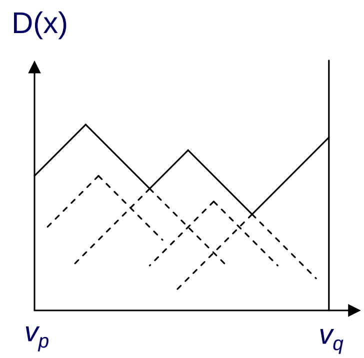

Figure 3: An example of the maximum distance function $D(x)$ for e_{pq}

1. $D(x)$ is a piecewise linear function with slopes -1 or 1.
2. $D(x)$ has at most l local maximum points, each of which is a bottleneck point for a sink.
3. $D(x)$ has at most $l+1$ local minimum points.
4. Given the shortest distance matrix, the set of local maxima and local minima can be constructed in $O(|U|) = O(l)$ time.

We now establish the relationship between certain points $x \in I_{pq}$ and the trees in $\mathcal{T}_o^{SP}$.

Theorem 5 *An interior point x of the edge (p,q) is the root of a shortest path tree $T_x \in \mathcal{T}_o^{SP}$ if and only if there are at least two sinks in T_x, say v_{j_1} and v_{j_2}, such that $D(x) = d_G(v_{j_1}, x) = d_G(v_{j_2}, x)$.*

Proof: First, if $T_x \in \mathcal{T}_o^{SP}$, then x is the center point of T_x. Thus, if we let the v_{j_1}-v_{j_2} path be a longest intra-sink path of T_x, then the v_{j_1}-v_{j_2} path contains x and $d_G(v_{j_1}, x) = d_G(v_{j_2}, x) = D(x) = \frac{1}{2}\delta_{T_x}$. Clearly, the v_{j_1}-v_{j_2} path contains e_{pq} as well, such that the shortest paths from v_{j_1} to x and from v_{j_2} to x in T_x approach x from different directions, through v_p and v_q, respectively.

Second, suppose that there are two sinks v_{j_1} and v_{j_2} such that $D(x) = d_G(v_{j_1}, x) = d_G(v_{j_2}, x)$ and the shortest paths from v_{j_1} to x and from v_{j_2} to x in T_x approach x from different directions, through v_p and through v_q, respectively. In order to show $T_x \in \mathcal{T}_o^{SP}$, we need to show that x is the center point of T_x. We claim the v_{j_1}-v_{j_2} path is a longest intra-sink path of T_x. First of all, the v_{j_1}-v_{j_2} path consists of the v_{j_1}-v_p path, edge e_{pq}, and the v_q-v_{j_2} path because the shortest paths from v_{j_1} to x and from v_{j_2} to x in T_x approach x from different directions, through v_p and v_q, respectively. In addition, if we let the v_a-v_b path be any other simple intra-sink path in T_x, then by the triangle inequality

$$\begin{aligned} d_{T_x}(v_a, v_b) &\leq d_{T_x}(v_a, x) + d_{T_x}(v_b, x) \\ &\leq 2D(x) \\ &= d_G(v_{j_1}, x) + d_G(v_{j_2}, x) \\ &= d_{T_x}(v_{j_1}, v_{j_2}). \end{aligned}$$

Hence, the v_{j_1}-v_{j_2} path is a longest intra-sink path such that x is the center point of T_x. ■

Based on the above, it follows that each interior local minimum point of $D(x)$, which excludes the endpoints v_p and v_q of e_{pq}, is a potential root for an optimal tree. In addition, any other point x on e_{pq} is a potential root for an optimal tree, if there are two sinks v_{j_1}, v_{j_2} with $D(x) = d_G(v_{j_1}, x) = d_G(v_{j_2}, x)$ and x is a bottleneck point for v_{j_1} or v_{j_2}. To conclude, each potential root for an optimal tree is either an interior local minimum point of $D(x)$ or a bottleneck point for a sink on $D(x)$, and there are at most $2|U| + 1 = 2l + 1$ such points.

4.3.2 Polynomial-Time Algorithm for NF-NMVT

In the algorithm, we shall check all those points described above to find the one offering an optimal tree. Before proceeding, we present definitions for calculating the objective function. For any point $x \in I_{pq}$, let T_x be a shortest path tree rooted at x. Let $V_p(x)$ and $V_q(x)$ be the sets of nodes $v \in V$, such that $V_p(x) \cup V_q(x) = V$, and v_p, v_q are on the path connecting v to x in T_x, respectively. If x is not a bottleneck point, $V_p(x)$ and $V_q(x)$ are uniquely defined. Otherwise, x is a bottleneck point such that $x = \gamma_v$ for some $v \in V$, then v could be in either $V_p(x)$ or $V_q(x)$. In this case, we shall specify $v \in V_p(x)$ or $v \in V_q(x)$ later in the algorithm. We define the sets $S_p(x)$ and $S_q(x)$ of sources v_i such that $v_i \in V_p(x)$ and $v_i \in V_q(x)$, respectively. Similarly, the sets $U_p(x)$ and $U_q(x)$ of sinks v_j are defined such that $v_j \in V_p(x)$ and $v_j \in V_q(x)$, respectively. We define the longest distance from all sinks $v_j \in U_p(x)$ to v_p and from all sinks $v_j \in U_q(x)$ to v_q in T_x as $\alpha(x)$ and $\beta(x)$, respectively, such that

$$\alpha(x) = \max_{v_j \in U_p(x)} \{d_G(v_j, v_p)\} \quad \text{and} \quad \beta(x) = \max_{v_j \in U_q(x)} \{d_G(v_j, v_q)\}.$$

In addition, based on Lemma 6, we define the maximal violation among nodes in $S_p(x)$ and $S_q(x)$ in T_x as

$$F_p(x) = \max_{v_i \in S_p(x)} \{f_i(h_i^{T_x} - H_i)\} \quad \text{and} \quad F_q(x) = \max_{v_i \in S_q(x)} \{f_i(h_i^{T_x} - H_i)\},$$

where $h_i^{T_x}$ is the longest distance from the source v_i to all sinks in T_x. The objective function valued at x, which is the maximal violation of T_x, is

$$F(x) = \max\{F_p(x), F_q(x)\}.$$

Based on Theorem 5, if the point x is the root of a shortest path tree $T_x \in \mathcal{T}_o^{SP}$, then

$$\alpha(x) \;=\; D(x) - d_p(x) \tag{2}$$

and

$$\beta(x) \;=\; D(x) - d_q(x). \tag{3}$$

Therefore, given the function $D(x)$ and the location of x represented by $d_p(x)$ or $d_q(x)$, it takes constant time to obtain $\alpha(x)$ and $\beta(x)$. In addition, in this case,

$$F_p(x) \;=\; \max_{v_i \in S_p(x)} \{f_i(d_G(v_i, v_p) + l_{pq} + \beta(x) - H_i)\} \tag{4}$$

and

$$F_q(x) \;=\; \max_{v_i \in S_q(x)} \{f_i(d_G(v_i, v_q) + l_{pq} + \alpha(x) - H_i)\}. \tag{5}$$

Therefore, given the node flow f_i and node restriction H_i for $v_i \in S$ and given the shortest path distance matrix in G, for any point $x \in I_{pq}$, the objective value $F(x)$ depends on only $S_p(x)$, $S_q(x)$, $\alpha(x)$, and $\beta(x)$.

For each edge $e_{pq} \in E$, the algorithm identifies the set of points offering a tree $T \in \mathcal{T}_o^{SP}$ and finds the point in the set which offers the best shortest path tree among all of the points in the set. After finding the best points for all edges, we choose the best one among these points and construct the shortest path tree rooted at this point as the optimal tree for NF-NMVT. Now we describe the algorithm in detail.

Algorithm: Edge-Examination

Before we examine the edges, we first compute all pairs shortest path distances in G and obtain the sorted sequence of shortest path distances from each node $v \in V$. For each edge $e_{pq} \in E$, do

1. Preprocessing step:
   (a) Calculate all the bottleneck points γ_v for all $v \in S \cup U$ by $d_p(\gamma_v) = \frac{1}{2}(d_G(v_q, v) - d_G(v_p, v) + l_{pq})$ and sort them in non-decreasing order according to $d_p(\gamma_v)$.
   (b) Using the algorithm in (Kariv and Hakimi, 1979), calculate the piecewise linear maximum distance function $D(x)$ for $x \in I_{pq}$, and record its sorted sequence of break points including local minimum and local maximum points with their respective values of $D(x)$ and $d_p(x)$.
   (c) Merge the lists of bottleneck points in (1a) and the break points in (1b) into a combined and sorted list L_{pq}.
2. Initialization step:
   (a) Check whether $x = v_p$ is the center point of a shortest path tree rooted at x. If it is, evaluate the objective function $F(v_p) = \max_{v_i \in S}\{f_i(d_G(v_i, v_p) + D(v_p) - H_i)\}$.

(b) Define the set $S_p(x)$ and $S_q(x)$ such that x is sufficiently close to v_p. For example, let x be any point between v_p and the point closest to v_p in the merged list of the preprocessing step.

(c) Using the data structures described in (Hershberger and Suri, 1996), maintain the following two piecewise linear convex function for $y \in R$,

$$G_p(x(y)) = \max_{v_i \in S_p(x)} \{f_i(d_G(v_i, v_p) + l_{pq} + y - H_i)$$

and

$$G_q(x(y)) = \max_{v_i \in S_q(x)} \{f_i(d_G(v_i, v_p) + l_{pq} + y - H_i).$$

Note that for any x, $F_p(x) = G_p(x(\beta(x)))$ and $F_q(x) = G_q(x(\alpha(x)))$. The above functions are defined as upper envelopes of collections of linear functions. Therefore, these data structures can support an insertion or a deletion of a linear function from the respective collection, as well as the evaluation of $G_p(x(y))$ and $G_q(x(y))$ at any given value of y in $O(\log |S|) = O(\log k)$ time.

3. Updating step:

Consider the next point in the list L_{pq} generated in the preprocessing step which has not been visited. Let $\bar{x}$ denote this point. The following cases shown in Figure 4 are considered.

(a) Case 1: $\bar{x}$ is a local minimum point of $D(x)$. Compute $\alpha(\bar{x})$, $\beta(\bar{x})$, $F_p(\bar{x})$, and $F_q(\bar{x})$ by Equation 2, Equation 3, Equation 4, and Equation 5, respectively, and evaluate the objective value $F(\bar{x}) = \max\{F_p(\bar{x}), F_q(\bar{x})\}$. Additionally, if there exist some sources $v_i \in S$ such that $\gamma_{v_i} = \bar{x}$, then for each source v_i such that $\gamma_{v_i} = \bar{x}$, delete v_i from $S_p(\bar{x})$ and insert it into $S_q(\bar{x})$.

(b) Case 2: $\bar{x}$ is a local maximum point of $D(x)$ such that there is a sink $v_{j_0} \in U$ such that $d_G(v_{j_0}, \bar{x}) = D(\bar{x})$. Also, there is another sink $v_{j_1} \neq v_{j_0}$, such that $\gamma_{v_{j_1}} = \bar{x}$ and $d_G(v_{j_1}, \bar{x}) = D(\bar{x})$. Compute $\alpha(\bar{x})$, $\beta(\bar{x})$, $F_p(\bar{x})$, and $F_q(\bar{x})$ by Equations 2, 3, 4, and 5, respectively, and evaluate the objective value $F(\bar{x}) = \max\{F_p(\bar{x}), F_q(\bar{x})\}$. Additionally, if there exist some sources $v_i \in S$ such that

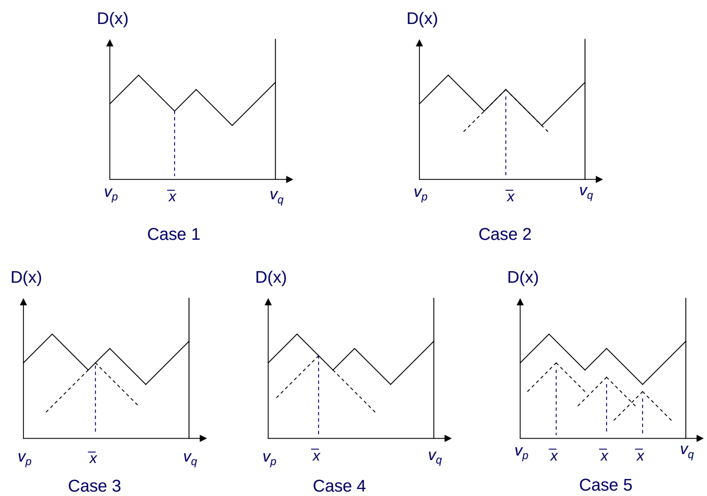

Figure 4: Examples for the cases in the algorithm

$\gamma_{v_i} = \bar{x}$, then for each source v_i such that $\gamma_{v_i} = \bar{x}$, delete v_i from $S_p(\bar{x})$ and insert it into $S_q(\bar{x})$.

(c) Case 3: $\bar{x}$ is such that $D(x)$ is strictly increasing in the interval $[\bar{x} - \varepsilon, \bar{x} + \varepsilon]$ for some $\varepsilon > 0$ sufficiently small. Also, there is a sink v_{j_0}, such that $\gamma_{v_{j_0}} = \bar{x}$ and $d_G(v_{j_0}, \bar{x}) = D(\bar{x})$. Compute $\alpha(\bar{x})$, $\beta(\bar{x})$, $F_p(\bar{x})$, and $F_q(\bar{x})$ by Equations 2, 3, 4, and 5, respectively, and evaluate the objective value $F(\bar{x}) = \max\{F_p(\bar{x}), F_q(\bar{x})\}$. Additionally, if there exist some sources $v_i \in S$ such that $\gamma_{v_i} = \bar{x}$, then for each source v_i such that $\gamma_{v_i} = \bar{x}$, delete v_i from $S_p(\bar{x})$ and insert it into $S_q(\bar{x})$.

(d) Case 4: $\bar{x}$ is such that the function $D(x)$ is strictly decreasing in the interval $[\bar{x} - \varepsilon, \bar{x} + \varepsilon]$ for some $\varepsilon > 0$ sufficiently small. Also, there is a sink v_{j_0}, such that $\gamma_{v_{j_0}} = \bar{x}$ and $d_G(v_{j_0}, \bar{x}) = D(\bar{x})$. Compute $\alpha(\bar{x})$, $\beta(\bar{x})$, $F_p(\bar{x})$, and $F_q(\bar{x})$ by Equations 2, 3, 4, and 5, respectively, and evaluate the objective value $F(\bar{x}) = \max\{F_p(\bar{x}), F_q(\bar{x})\}$. Additionally, if there exist some sources $v_i \in S$ such that $\gamma_{v_i} = \bar{x}$, then for each source v_i such that $\gamma_{v_i} = \bar{x}$, delete v_i from $S_p(\bar{x})$ and insert it into $S_q(\bar{x})$.

(e) Case 5: If none of the above cases apply at $\bar{x}$, if there exist some sources $v_i \in S$

such that $\gamma_{v_i} = \bar{x}$, then for each source v_i such that $\gamma_{v_i} = \bar{x}$, delete v_i from $S_p(\bar{x})$ and insert it into $S_q(\bar{x})$.

In the above algorithm, it takes $O(mn+n^2 \log n)$ to compute all pairs shortest path distances in G. For each of the m edges, step 1 calculates and sorts the list of bottleneck points for nodes $v \in S \cup U$ in $O(|S| + |U|) = O(\bar{n} \log \bar{n})$ time, generates the maximum distance function $D(x)$ and calculates the sorted list of its local minimum points and local maximum points in $O(|U|) = O(l)$ time (Kariv and Hakimi, 1979), and merges the two sorted lists of points to the list L_{pq} in $O(\bar{n})$ time. Step 3 scans the values in the merged list L_{pq} defined in step 1 to calculate $F(x)$ at all potential points on an edge. In step 3, for each such potential point, it takes constant time to compute $\alpha(x)$, $\beta(x)$. We will use the dynamic data structure in (Hershberger and Suri, 1996) for calculating $F_p(x)$ and $F_q(x)$ in step 3. Since there are $O(|S|) = O(k)$ insertions, deletions, updates, and $O(|U|) = O(l)$ evaluations of $F_p(x)$ and $F_q(x)$ and each such operation takes $O(\log |S|) = O(\log k)$ time with this data structure, the effort for finding the best solution on an edge is $O((|S| + |U|) \log |S|) = O(\bar{n} \log k)$. Thus, the total time for the algorithm to find an optimal tree for NF-NMVT is $O(mn + n^2 \log n + m\bar{n} \log \bar{n})$.

4.3.3 NF-MDST, NF-MEMT, and NF-UMVT

Because each of the three variants is a special case of NF-NMVT, they can be solved by specifically modifying the above algorithm for NF-NMVT. We describe the modifications of the above algorithm for finding an optimal solution for NF-MDST, NF-MEMT, and NF-UMVT as follows.

For NF-MDST, set $S = V$ and $U = V$. For NF-MEMT, set $S \subseteq V$ and $U \subseteq V$. Further, for both NF-MDST and NF-MEMT, replace Equations 4 and 5 in the algorithm, respectively, by

$$F_p(x) \quad = \quad \max_{v_i \in S_p(x)} \{f_i(d_G(v_i, v_p) + l_{pq} + \beta(x))\} \tag{6}$$

and

$$F_q(x) = \max_{v_i \in S_q(x)} \{f_i(d_G(v_i, v_q) + l_{pq} + \alpha(x))\}. \tag{7}$$

For NF-UMVT, set $S \subseteq V$ and $U \subseteq V$ and replace Equations 4 and 5 in the algorithm, respectively, by

$$F_p(x) = \max_{v_i \in S_p(x)} \{f_i(d_G(v_i, v_p) + l_{pq} + \beta(x) - H)\} \tag{8}$$

and

$$F_q(x) = \max_{v_i \in S_q(x)} \{f_i(d_G(v_i, v_q) + l_{pq} + \alpha(x)) - H\}. \tag{9}$$

Therefore, the algorithm solves NF-MEMT and NF-UMVT in $O(mn + n^2 \log n + m\bar{n} \log \bar{n})$ time, which is the same as NF-NMVT problem. For NF-MDST, the algorithm finds the optimal solution in $O(mn + n^2 \log n + mn \log n) = O(mn \log n)$ time.

4.4 Special Flow Minimax Diameter Spanning Tree (SF-MDST)

In the next section we will prove that PF-MDST is generally $\mathcal{NP}$-hard. Nevertheless, there are some interesting special cases which are polynomially solvable. In this section, we study one solvable case, which is equivalent to the classical Weighted Absolute 1-Center problem (WAC). This problem is a variant of PF-MDST, where $S = U = V$, $H_{ij} = 0$, and $f_{ij} = \frac{f_i f_j}{f_i + f_j}$ for all pairs of nodes $v_i, v_j \in V$. We label this variant as SF-MDST.

We now show that SF-MDST is equivalent to WAC on a graph G. Recall that the goal of WAC is to find a point $x \in I(G)$ minimizing the objective

$$\max_{v_i \in V} \{f_i d_G(x, v_i)\}.$$

Assuming without loss of generality that there are at least two nodes with positive flow weights, WAC is equivalent to finding a minimum spanning tree T of G minimizing

$$r_T = \min_{x \in I(T)} \max_{v_i \in V} \{f_i d_T(x, v_i)\}.$$

Using the results in Dearing and Francis (1974) (see also Section 7.4 in (Francis et al., 1992)), we observe that

$$r_T = \max_{v_i, v_j \in V} \{\frac{f_i f_j}{f_i + f_j} d_T(v_i, v_j)\}.$$

Therefore, we conclude that WAC is equivalent to SF-MDST, whose objective function is the minimization of r_T. Moreover, a spanning tree T^* optimizing the SF-MDST can be obtained in $O(mn \log n)$ time as follows. Use the algorithm in Kariv and Hakimi (1979) to find x^*, an optimal weighted absolute 1-center. T^* is a shortest path spanning tree rooted at x^*.

5 The Intractable Problems

In this section, we shall show that all the problem variants with either pairwise service-time commitments or pairwise flow are $\mathcal{NP}$-Complete. All of the intractable problems involve transformations with the Tree t-Spanner problem.

5.1 Pairwise Commitment Minimax Violation Tree (PMVT)

We now establish the $\mathcal{NP}$-Completeness of all the problem variants with pairwise service-time commitment. This class includes PMVT, NF-PMVT, and PF-PMVT. We shall focus on PMVT and show it is $\mathcal{NP}$-Complete. The $\mathcal{NP}$-Completeness of NF-PMVT and PF-PMVT follows because PMVT is a special case of the other two problems. We first state the decision version of PMVT.

Instance: Graph $G = (V, E)$, source set $S \subseteq V$, sink set $U \subseteq V$, pairwise service-time commitment H_{ij} for any pairs of source $v_i \in S$ and sink $v_j \in U$, integer bound $K \in Z^+$.

Question: Is there a spanning tree T for G such that the maximum violation in T over all pairs of source $v_i \in S$ and sink $v_j \in U$, $\max_{v_i \in S} \max_{v_j \in U} \{d_T(v_i, v_j) - H_{ij}\}$, is no more than K?

In order to demonstrate the $\mathcal{NP}$-Completeness of PMVT, we need to first introduce the Tree t-Spanner problem. The following is the decision version of the Tree t-Spanner problem.

Instance: For a graph $G = (V, E)$, let $d_G(v_i, v_j)$ be the shortest path distance between any two nodes $v_i, v_j \in V$ in G and $d_T(v_i, v_j)$ be the length of the unique path connecting v_i and v_j in a spanning tree T. Given an integer $t \in Z^+$.

Question: Is there a spanning tree T for G such that for each pair of nodes $v_i, v_j \in V$, $d_T(v_i, v_j) \le t d_G(v_i, v_j)$?

Chew (1986) and Peleg and Ullman (1987) introduce the notation of Tree t-Spanner. For general nonnegative edge lengths, Cai and Corneil (1995) demonstrate the Tree t-Spanner problem is $\mathcal{NP}$-Complete even for $t = 2$, and it remains $\mathcal{NP}$-Complete for $t \ge 4$ for unit edge lengths. The $\mathcal{NP}$-Completeness of PMVT is presented in the following theorem.

Theorem 6 *PMVT is $\mathcal{NP}$-Complete.*

Proof: As stated in Section 1, PMVT is in $\mathcal{NP}$, so we need only to prove PMVT is $\mathcal{NP}$-hard. We shall next prove PMVT is $\mathcal{NP}$-hard by transforming the Tree t-Spanner problem into a special case of PMVT.

Given an instance of the Tree t-spanner problem, we can define a special case of PMVT by setting $S = V$, $U = V$, the service-time commitment between any source $v_i \in V$ and sink $v_j \in V$ as $H_{ij} = t d_G(v_i, v_j)$, and the integer bound $K = 0$. We claim that this special case of PMVT has a solution if and only if the Tree t-Spanner problem has a solution.

Suppose first that this special PMVT has a solution. If we let source $v_i \in V$ and sink $v_j \in V$ be any pair of source and sink nodes in the graph, there exists a tree T such that the violation between v_i and v_j is $F_T(v_i, v_j) = d_T(v_i, v_j) - H_{ij} = d_T(v_i, v_j) - t d_G(v_i, v_j) \le K = 0$. Then, $d_T(v_i, v_j) \le t d_G(v_i, v_j)$ for any nodes $v_i \in V$ and $v_j \in V$ in T. Hence, the tree T is also the solution of the Tree t-Spanner problem.

Next, suppose the Tree t-Spanner problem has a solution. Thus, there exists a tree T such that $d_T(v_i, v_j) \leq td_G(v_i, v_j)$ for any two nodes $v_i \in V$ and $v_j \in V$ in T. The violation between any source $v_i \in V$ and any sink $v_j \in V$ in the tree T is $F_T(v_i, v_j) = d_T(v_i, v_j) - H_{ij} = d_T(v_i, v_j) - td_G(v_i, v_j) \leq 0 = K$. Hence, the tree T is a solution of the special PMVT as well.

Therefore, because the Tree t-Spanner problem is $\mathcal{NP}$-hard, PMVT is $\mathcal{NP}$-hard as well. Since PMVT is in $\mathcal{NP}$, it is $\mathcal{NP}$-Complete. ■

Based on the above reduction, we present two additional observations which demonstrate that, given the pairwise service-time commitment, both the problem of minimizing a monotone function of violation among all pairs of sources and sinks in a tree and the problem of finding an approximation solution to PMVT within any constant factor are $\mathcal{NP}$-hard. We state these results but omit their proofs for the sake of brevity.

Corollary 2 *Suppose that $S = U = V$ for the pairwise service-time commitment model. For each spanning tree T and pair (v_i, v_j), $i < j$, define $H^T_{ij} = \max(0, d_T(v_i, v_j) - H_{ij})$. Let $g(x_{11}, x_{12}, ..., x_{1n}, x_{23}, ..., x_{2n}, ..., x_{n-1,n})$ be a monotone nondecreasing function of its $n(n-1)/2$ arguments. Suppose that $g(0, ..., 0) = 0$, and g is positive for any nonnegative nonzero vector. Then, the problem of finding a tree minimizing $g(H^T_{11}, H^T_{12}, ..., H^T_{1n}, H^T_{23}, ..., H^T_{2n}, ..., H^T_{n-1,n})$ is $\mathcal{NP}$-hard.*

Corollary 3 *Finding an approximation solution to PMVT within any constant factor is $\mathcal{NP}$-hard.*

5.2 Pairwise Flow Minimax Violation Tree (PF-MVT)

We shall study the problem variants with pairwise flow. This class includes PF-MDST, PF-MEMT, PF-UMVT, and PF-NMVT problems. From Table 1, we can see that PF-MDST is a special case of the other three problems. Therefore, we shall focus on establishing the

$\mathcal{NP}$-Completeness for PF-MDST. We will first offer the decision version of PF-MDST and demonstrate that it is $\mathcal{NP}$-Complete by the transformation from Tree t-Spanner problem.

Instance: Graph $G = (V, E)$ and pairwise flow f_{ij} for any pair of two nodes $v_i \in V$ and $v_j \in V$, integer bound $K \in Z^+$.

Question: Is there a spanning tree T for G such that $f_{ij} d_T(v_i, v_j) \leq K$ for all pairs of nodes $v_i, v_j \in V$?

Theorem 7 *PF-MDST is $\mathcal{NP}$-Complete.*

Proof: As stated in Section 1, PF-MDST is in $\mathcal{NP}$, so we need only to prove it is $\mathcal{NP}$-hard. We shall prove this result by transforming the Tree t-Spanner problem to a special case of PF-MDST.

Given an instance of the Tree t-Spanner problem, we can define a special case of PF-MDST as follows. Let the pairwise flow $f_{ij} = \frac{1}{t d_G(v_i, v_j)}$ and the integer bound $K = 1$. We claim that this special PF-MDST has a solution if and only if the Tree t-Spanner problem has a solution.

For this special PF-MDST, the violation for a pair of node v_i and v_j in tree T is

$$\begin{aligned} f_{ij} d_T(v_i, v_j) &= \frac{1}{t d_G(v_i, v_j)} d_T(v_i, v_j) \\ &= \frac{d_T(v_i, v_j)}{t d_G(v_i, v_j)}. \end{aligned}$$

This special PF-MDST is to find a tree T such that $\frac{d_T(v_i, v_j)}{t d_G(v_i, v_j)} \leq K = 1$, that is, $\frac{d_T(v_i, v_j)}{d_G(v_i, v_j)} \leq t$ for all pairs of nodes $v_i, v_j \in V$.

Therefore, any solution tree for this special PF-MDST is also a solution tree for the Tree t-Spanner problem, and vice versa. Because the Tree t-Spanner problem is $\mathcal{NP}$-hard, PF-MDST is $\mathcal{NP}$-hard as well. ■

In addition, because we can specify S, U, and H_{ij} for each of PF-MEMT, PF-UMVT, and PF-NMVT to obtain PF-MDST, PF-MEMT, PF-UMVT, and PF-NMVT are $\mathcal{NP}$-Complete as well.

6 Future Work

There are several directions deserving further investigation. In this study, we treat the service-time commitment as a *soft constraint*. However, in reality it may be a *hard constraint* for certain customers. If a spanning tree cannot satisfy the distance constraints, it may become necessary to find a subgraph, other than a spanning tree, in order to eliminate the violation. Furthermore, because a delivery company has limited facilities in each city, it may only be able to support a limited number of connections in each city. Hence, it may be useful to consider the spanning tree with degree constraints. Likewise, arcs in the network are often capacity constrained due to the limitations on the number of trucks and truck capacities. Thus, another interesting variant would be a capacitated minimax flow tree problem.

In this paper, we have discussed minimax objectives. However, companies may instead prefer to measure the sum of violations in their network. This objective motivates an additional interesting problem, the minisum violation tree problem. Chen et al. (2007) explore the complexity and solution approaches for the minisum problem.

Acknowledgments

This work was partially supported by the National Science Foundation through grant number 0237726(Campbell).

References

R.K. Ahuja and V.V.S Murty. Exact and heuristic algorithms for the optimum communication spanning tree problem. *Transportation Science*, 21:163–170, 1987.

Ernst Althaus, Stefan Funke, Sariel Har-Peled, Jochen Könemann, Edgar A.Ramos, and Martin Skutella. Approximating k-hop minimum spanning trees. *Operations Research Letters*, 33: 115–120, 2005.

Leizhen Cai and Derek G. Corneil. Tree spanners. *SIAM Journal on Discrete Mathematics*, 8: 359–387, 1995.

Hui Chen, Ann M. Campbell, and Barrett Thomas. Minisum distance violation tree problems. Working Paper, 2007.

L. Paul Chew. There is a planar graph almost as good as the complete graph. In *Proceedings of the Second Annual Symposium on Computational Geometry*, pages 169–177, New York, NY, USA, 1986.

Harold S. Connamacher and Andrzej Proskurowski. The complexity of minimizing certain cost metrics for k-source spanning trees. *Discrete Applied Mathematics*, 131:113–127, 2003.

Geir Dahl. The 2-hop spanning tree problem. *Operations Research Letters*, 23:21–26, 1998.

P.M. Dearing and R.L. Francis. A minimax location problem on tree networks. *Transportation Science*, 8:333–343, 1974.

Arthur M. Farley, Paraskevi Fragopoulou, David Krumme, Andrzej Proskurowski, and Dana Richards. Multi-source spanning tree problem. *Journal of Interconnection Networks*, 1:61–71, 2000.

Paraskevi Fragopoulou, Stavros D. Nikolopoulos, and Leonidas Palios. Multi-source trees: Algorithms for minimizing eccentricity cost metrics. In *ISAAC*, volume 3827 of *Lecture Notes in Computer Science*, pages 1080–1089. Springer, 2005.

Richard L. Francis, Leon F. McGinnis Jr., and John A. White. *Facility Layout and Location: An Analytical Approach*. Prentice Hall, 1992.

Michael L. Fredman and Robert Endre Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the Association for Computing Machinery*, 34(3):596–615, 1987.

Luis Gouveia. Using the Miller-Tucker-Zemlin constraints to formulate a minimal spanning tree problem with hop constraints. *Computers & Operations Research*, 22:959–970, 1995.

Luis Gouveia. Multicommodity flow models for spanning trees with hop constraints. *European Journal of Operational Research*, 95:178–190, 1996.

Luis Gouveia. Using variable redefinition for computing lower bounds for minimum spanning and steiner trees with hop constraints. *Informs Journal on Computing*, 10:180–188, 1998.

Luis Gouveia and Cristina Requejo. A new Lagrangian relaxation approach for the hop-constrained minimum spanning tree problem. *European Journal of Operational Research*, 132:539–552, 2001.

S. L. Hakimi. Optimal locations of switching centers and medians of a graph. *Operations Research*, 12:450–459, 1964.

Gabriel Y. Handler. Minimax location of a facility in an undirected tree graph. *Transportation Science*, 7:287–293, 1973.

Gabriel Y. Handler and Pitu B. Mirchandani. *Location on Networks*. MIT Press, Cambridge, MA, 1979.

Refael Hassin and Arie Tamir. On the minimum diameter spanning tree problem. *Information Processing Letters*, 53:109–111, 1995.

John Hershberger and Subhash Suri. Off-line maintenance of planar configurations. *Journal of Algorithms*, 21:453–475, 1996.

Jan-Ming Ho, D. T. Lee, Chia-Hsiang Chang, and C. K. Wong. Minimum diameter spanning trees and related problems. *SIAM Journal on Computing*, 20(5):987–997, 1991.

T.C. Hu. Optimum communication spanning trees. *SIAM Journal on Computing*, 3:188–195, 1974.

D. S. Johnson, J. K. Lenstra, and A. H. G. Rinnooy Kan. The complexity of the network design problem. *Networks*, 8:279–285, 1978.

O. Kariv and S. L. Hakimi. An algorithmic approach to network location problems. I: The p-centers. *SIAM Journal on Applied Mathematics*, 37:513–538, 1979.

David W. Krumme and Paraskevi Fragopoulou. Minimum eccentricity multicast trees. *Discrete Mathematics and Theoretical Computer Science*, 4:157–172, 2001.

Prabhu Manyem and Matthias F. M. Stallmann. Some approximation results in multicasting. Working Paper, 1996.

H.Brendan McMahan and Andrzej Proskurowski. Multi-source spanning trees: Algorithm for minimizing source eccentricities. *Discrete Applied Mathematics*, 137:213–222, 2004.

David Peleg. Polylogarithmic approximation for minimum communication spanning trees. Technical report, Department of Applied Mathematics and Computer Science, The Weizmann Institute of Science, 1997. CS97-10.

David Peleg and Eilon Reshef. Deterministic polylog approximation for minimum communication spanning trees (extended abstract). *Lecture Notes in Computer Science*, 1443:670–681, 1998.

David Peleg and Jeffrey D. Ullman. An optimal synchronizer for the hypercube. *Proceedings of the 6th ACM Symposium on Principles of Distributed Computing*, pages 77–85, 1987.

W. B. Powell and I. A. Koskosidis. Shipment routing algorithms with tree constraints. *Transportation Science*, 26:230–245, 1992.

Stefan Voss. The steiner tree problem with hop constraints. *Annals of Operations Research*, 86: 321–345, 1999.

Bang Ye Wu. A polynomial-time approximation scheme for the two-source minimum routing cost spanning trees. *Journal of Algorithms*, 44:359–378, 2002.

Bang Ye Wu. An improved algorithm for the k-source maximum eccentricity spanning trees. *Discrete Applied Mathematics*, 143:342–350, 2004.

Bang Ye Wu, Kun-Mao Chao, and ChuanYi Tang. Approximation algorithm for some optimum communication spanning tree problems. *Discrete Applied Mathematics*, 102:245–266, 2000a.

Bang Ye Wu, Kun-Mao Chao, and ChuanYi Tang. Approximation algorithm for the shortest total path length spanning tree problem. *Discrete Applied Mathematics*, 105:273–289, 2000b.

Bang Ye Wu, Giuseppe Lancia, Vineet Bafna, Kun-Mao Chao, R.Ravi, and Chuan Yi Tang. A polynomial-time approximation scheme for minimum routing cost spanning trees. *SIAM Journal on Computing*, 29:761–778, 1999.