The Complexity of Minimizing Certain Cost Metrics for k-Source Spanning Trees

Harold S. Connamacher
University of Oregon

Andrzej Proskurowski*
University of Oregon

May 9, 2001

Abstract

We investigate multi-source spanning tree problems where, given a graph with edge weights and a subset of the nodes defined as sources, the object is to find a spanning tree of the graph that minimizes some distance related cost metric. This problem can be used to model multicasting in a network where messages are sent from a fixed collection of senders and communication takes place along the edges of a single spanning tree. For a limited set of possible cost metrics of such a spanning tree, we either prove the problem is NP-hard or we demonstrate the existence of an efficient algorithm to find an optimal tree.

1 Introduction

The motivation for this paper is a message dissemination process called multicasting in which a message is broadcast to multiple receivers across a network. One possible paradigm of multicasting has several sources from a fixed set of vertices transmit the data with every vertex in the network as a receiver. Multicast protocols often use a single routing tree which is shared by all transmissions. The goal of the tree construction may be to minimize the time it takes to complete a message dissemination, and this paper examines the feasibility of constructing optimal routing trees for such a protocol. The optimality of a tree is determined by minimizing some given cost function. Multiple cost metrics are considered because different applications may call for different requirements and because some of the metrics turn out to define intractable optimization problems.

If there is only one source, an algorithm to find the single source shortest paths spanning tree will produce an optimal tree for each of the cost metrics considered. Therefore, this investigation

*Research supported in part by a National Science Foundation grant NSF-ANI-9977524.

	Problem	Cost Metric	Complexity	Reference
1	k-SPST	$cost_1(T) = \sum_{s\in S}\sum_{v\in V} d_T(s,v)$	$\mathcal{NP}$-complete	[2]
2	k-MVST	$cost_2(T) = \max_{v\in V}\sum_{s\in S} d_T(s,v)$	$\mathcal{NP}$-complete	this paper
3	k-MSST	$cost_3(T) = \max_{s\in S}\sum_{v\in V} d_T(s,v)$	$\mathcal{NP}$-complete	this paper
4	k-SVET	$cost_4(T) = \sum_{v\in V}\max_{s\in S} d_T(s,v)$	$\mathcal{P}$	this paper
5	k-SSET	$cost_5(T) = \sum_{s\in S}\max_{v\in V} d_T(s,v)$	$\mathcal{P}$	this paper
6	k-MEST	$cost_6(T) = \max_{s\in S, v\in V} d_T(s,v)$	$\mathcal{P}$	[6, 7]

Table 1: Multi-source Spanning Tree Problems and Their Complexity Status

considers only instances with more than one source. The problem is non-trivial because a shortest paths tree from one of the sources would not yield good results when used in conjunction with the other sources.

Each of the problems investigated in this paper has a specific cost metric parameterized by the number of sources, a positive integer k. All the metrics are combinations of distances between sources and vertices in the tree, and the operations combining the distances are max and sum.

k-Source Spanning Tree Problems

Instance: A graph $G = (V, E)$ with a length function, $l : E \rightarrow \Re$, k sources $S = \{s_1, \ldots, s_k\} \subseteq V$, a positive integer K.

Question: Is there a spanning tree T of G such that $cost(T) \leq K$?

Table 1 lists the different problems and their complexity status. It should be noted that each of these problems is in $\mathcal{NP}$ as one can simply guess a spanning tree and, in polynomial time, calculate the appropriate cost metric.

The first of these problems, k-Source Shortest Paths Spanning Tree (k-SPST), is an instance of the more general Optimum Communication Spanning Tree (cf. [ND7] in [3]) as defined in [4]. Also, if every vertex is a source, this problem becomes the Shortest Total Path Length Spanning Tree (cf. [ND3] in [3]). Both these problems are $\mathcal{NP}$-hard ([5]), and the k-SPST problem is $\mathcal{NP}$-complete even with two sources and uniform edge weights ([2]). An efficient solution exists for the last problem in Table 1, k-Source Maximum Eccentricity Spanning Tree

(k-MEST), and [7] presents an $\mathcal{O}(|V|^3+|E||V|\log|V|)$ algorithm while [6] presents an $\mathcal{O}(|V|^3)$ algorithm. The remaining four metrics were introduced as open problems in [7], and this paper completely characterizes the complexity status of each of the four remaining problems: we prove $\mathcal{NP}$-completeness of two other problems, k-MVST and k-MSST, in Section 2 and tractability of the remaining two problems, k-SVET and k-SSET, in Section 3.

Before proceeding, we give some basic definitions. A graph is a pair $G = (V, E)$ where V is the set of vertices and E is the set of edges. There is a length function defined on the edges, $l : E \to \Re$. This paper will also make use of points where a point may be either a vertex of G or a location along an edge of G. The sources of a graph are a nonempty subset of the vertices. A spanning tree T of G is a connected acyclic graph which connects all vertices of G using a subset of the edges of G. The distance function, $d : V \times V \to \Re$, on nodes u and v is the sum of the length of each edge on a path from u to v, minimum over all such paths. Depending on the set of edges considered, we distinguish between the tree distance $d_T(u, v)$ in which the uv-path is unique and the graph distance $d_G(u, v)$ which is defined over all possible paths from u to v in G. Finally, the source eccentricity of a graph is the maximum distance between a source vertex and any other vertex.

2 The Intractable Problems

2.1 k-Source Maximum Vertex Shortest Paths Spanning Tree (k-MVST)

Instance: A graph $G = (V, E)$ with a length function, $l : E \to \Re$, k sources $S = \{s_1, \ldots, s_k\} \subseteq V$, a positive integer K.

Question: Is there a spanning tree T of G such that $cost_2(T) = \max_{v \in V} \sum_{s \in S} d_T(s, v) \leq K$?

This metric minimizes the sum of the distance to each source from the farthest vertex in the tree. We show that this problem is $\mathcal{NP}$-hard. The proof is by a reduction from 3-SAT and closely follows the proof of $\mathcal{NP}$-completeness of k-SPST in [2].

Theorem 2.1. 2-MVST is $\mathcal{NP}$-complete even for graphs with unit edge lengths.

Proof. Given an instance of 3-SAT with m clauses $C_1, \ldots, C_m$ and n variables $x_1, \ldots, x_n$, we construct a graph G. For each variable, x_i, we create a 4-cycle gadget with vertices labeled, in order, $x_i^{'}$, x_i^T, $x_i^{''}$, x_i^F. We will connect these 4-cycles in a chain such that $x_i^{''} = x_{i+1}^{'}$ for $i = 1, \ldots, n-1$. For each clause, C_j, create a vertex labeled c_j, and for each of the three literals in the clause, connect the clause vertex to the associated variable gadget by a path with n intermediate nodes so that if clause C_j contains the literal x_i, the path will connect vertex c_j to vertex x_i^T and if clause C_j contains the literal $\neg x_i$, the path will connect vertex c_j to vertex x_i^F. Finally, let $S = \{s_1, s_2\}$ with $s_1 = x_1^{'}$ and $s_2 = x_n^{''}$, and let $K = 4n + 2$. See Figure 1 for an example of a clause and variable gadgets.

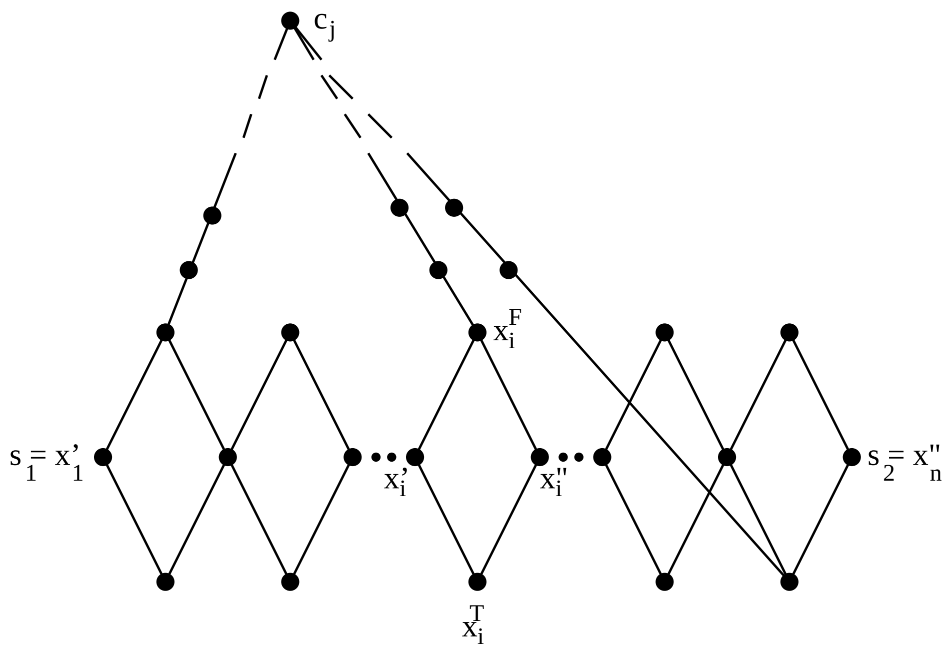


Figure 1: The construction for 2-MVST with clause $C_j = \neg x_1 \vee \neg x_i \vee x_n$.

This graph can be constructed in polynomial time because the chain of variable gadgets has $3n+1$ vertices and $4n$ edges and each clause vertex is connected to the chain by three paths of n vertices and $n+1$ edges. So there is a total of $3n+1+m(3n+1)$ vertices and $4n+3m(n+1)$ edges in G.

The instance of 3-SAT is satisfiable if and only if G has a spanning tree T with $cost_2(T) \leq K$. To prove this, we observe that an assignment satisfying the given instance of 3-SAT determines such a tree T. In this tree, the path between s_1 and s_2 traverses the variable gadget chain according to the variable truth assignment. If x_i is assigned true, x_i^T will be on the path, and likewise if x_i is assigned false, x_i^F will be on the path. The tree is completed by choosing a literal critical to satisfying each clause C_j and, of the three edges incident with vertex c_j, including in the tree only the edge which leads to that literal. Finally, one additional edge from each variable gadget is included in the tree. Define the weight of a vertex to be the sum of the distances to both sources and note that the weight of a vertex is equal to twice the distance of the vertex from the s_1s_2-path plus the length of the s_1s_2-path (equal to $2n$). Each variable gadget vertex not on the intra-source path is at distance one from it, each vertex in a path between a clause vertex and the variable gadget chain is at distance at most $n+1$ from the intra-source path, and each clause vertex is at distance $n+1$ from the intra-source path. Therefore, $cost_2(T) = 4n+2 = K$.

Likewise, if we find an optimal tree for G, we can construct a satisfying 3-SAT assignment by noting which literals lie along the $s_1 s_2$-path and setting each variable's truth value accordingly. This is possible because for G to have a spanning tree T with $cost_2(T) \leq K$, the path between s_1 and s_2 must not contain any of the clause vertices, and the nodes on the path must correspond to a satisfying assignment for the C_js. Considering cases, if we allow the path to contain two or more clause vertices, then the length of the $s_1 s_2$-path will be at least $4n+6$, and if we allow the intra-source path to contain exactly one clause vertex, then the length of the path will be at least $2n+4$. In this case, the weight for some other clause vertex, and thus the cost of the tree, will be at least $4n+6$ because that vertex is at distance $n+1$ from the intra-source path. Thus, no clause vertex can be on the path from s_1 to s_2. Now, assume the tree does not correspond to a satisfying assignment for the C_js. Then, by the way G was constructed, some clause vertex must be at a distance $n+2$ from the intra-source path, and thus, $cost_2(T) = 4n+4 > K$. Therefore, G has an optimal tree if and only if there is a satisfying assignment to the x_is. $\square$

Corollary 2.2. k-MVST is $\mathcal{NP}$-complete.

2.2 k-Source Maximum Source Shortest Paths Spanning Tree (k-MSST)

Instance: A graph $G = (V, E)$ with a length function, $l : E \rightarrow \Re$, k sources $S = \{s_1, \ldots, s_k\} \subseteq V$, a positive integer K.

Question: Is there a spanning tree T of G such that $cost_3(T) = \max_{s \in S} \sum_{v \in V} d_T(s, v) \leq K$?

This problem is also $\mathcal{NP}$-hard. We prove this using a similar technique to the proof of Theorem 2.1, but in this case the reduction is from Exact Cover By 3-Sets (X3C) ([SP2] in [3]). In X3C we are given a set X with $|X| = 3m$ and a collection C of three-element subsets of X, and we are asked whether there exists $C' \subseteq C$ such that every member of X occurs in exactly one member of C'. In the proof we will make use of the following observations:

Observation 2.3. $cost_3(T) \geq \frac{1}{|S|} \cdot cost_1(T)$ where $cost_1(T) = \sum_{s \in S} \sum_{v \in V} d_T(s, v)$.

The second observation is from [2, Observation 1].

Observation 2.4. The cost of a 2-SPST spanning tree T of a graph G with N vertices is equal to $N \cdot d(p) + 2 \sum_{v \in V} d(v, p)$ where p is the $s_1 s_2$-path, $d(p)$ is the length of p, and $d(v, p)$ is the shortest distance from v to a vertex of p.

Theorem 2.5. 2-MSST is $\mathcal{NP}$-complete even for graphs with unit edge lengths.

Proof. Given an instance of X3C with set X, $|X| = 3m$, and a collection C, $|C| = n$, of three-element subsets of X, construct the graph G as follows. First, without loss of generality,

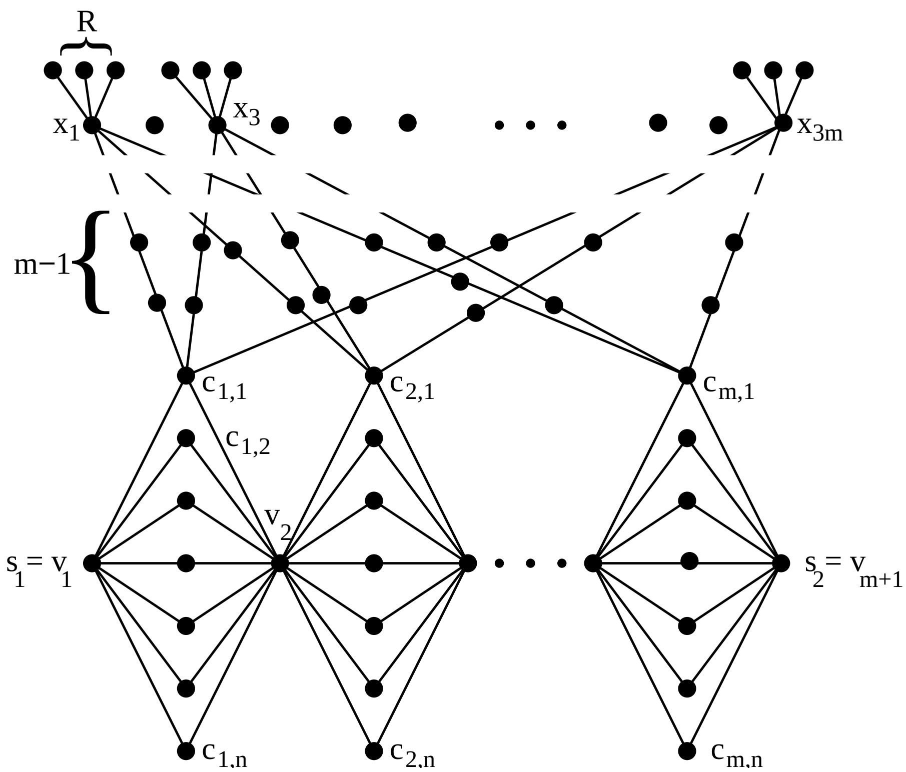


Figure 2: The construction for 2-MSST with triple $t_1 = \{x_1, x_3, x_{3m}\}$.

assume n is odd since we can always supplement C by a duplicate member. G will contain m "triples" gadgets, each consisting of $n+2$ vertices $v_i, c_{i,1}, \ldots, c_{i,n}, v_{i+1}$ and edges between every $c_{i,j}$ vertex and v_i, v_{i+1}, for $i = 1, \ldots, m$. For each triple $t_j \in C$, there are m vertices $c_{i,j}$ in G. Linking the gadgets through the vertices v_i forms a chain in G. Also, for each of the $3m$ elements $x_k \in X$ there is a vertex x_k in G, and for every $t_j \in C$ and for each $x_k \in t_j$, we connect $c_{i,j}$ and x_k in G by a path with $m-1$ internal vertices, for $i = 1, \ldots, m$. Finally, connected to each vertex x_k there will be $R = \frac{9}{2}m^2 n + \frac{5}{2}m$ additional vertices of degree one. The two sources are $s_1 = v_1$ and $s_2 = v_{m+1}$. See Figure 2 for an illustration of a "triples" gadget.

Now, let

$$K = 4m^2 + 3m + \frac{9}{2}m^3 n - \frac{1}{2}m^2 n - 2mn + 6m^2 R + 3mR.$$

Note that

$$\begin{aligned}|V(G)| &= 3m(R+1)+mn(1+3(m-1))+m+1\\ |E(G)| &= 2mn+3mn(m-1)+3mR.\end{aligned}$$

Thus, the reduction can be done in polynomial time. To complete the proof, we will show that X has an exact cover if and only if G has a 2-MSST tree of $cost_3 \leq K$.

For G to have a tree T^* with $cost_3(T^*) \leq K$, the s_1s_2-path in this optimal tree must pass along all the "triples" gadgets and not include any element vertex x_k. Also, the path from an element vertex x_k to a source vertex will not include any vertex $c_{i,j}$ which is not on the s_1s_2-path. If the s_1s_2-path does not include any x_k, there will be exactly m vertices $c_{i,j}$ on the path, and since each $c_{i,j}$ vertex has a direct path to only three of the x_k vertices, T^* exists if and only if X has an exact cover.

The "only if" direction is easier and we will start the proof with it. A spanning tree T^* corresponding to a cover C' has an s_1s_2-path which includes exactly vertices $c_{i,j}$ from "triples" gadgets, for all triples $t_j \in C'$ (in any order of gadgets). The cost of T^* is equal to the sum of vertex distances to any of the sources (they are the same) and consists of (i) the total cost of vertices on the s_1s_2-path, (ii) the total cost of vertices on the paths from the chosen $c_{i,j}$s to all element gadgets, and (iii) the total cost of vertices on all the other "truncated" paths, leading from the not chosen vertices $c_{i,j}$ to element gadgets.

(i) The total cost of vertices on the s_1s_2-path is

$$\sum_{i=1}^{2m} i = 2m^2 + m.$$

(ii) The total cost of vertices on the paths between each chosen $c_{i,j}$ (on the s_1s_2-path) and the element vertices it covers is

$$\begin{aligned}3\left[\sum_{i=1}^{m}(i+1)+\sum_{i=1}^{m}(i+3)+\ldots+\sum_{i=1}^{m}(i+2m-1)\right]&\\ &= 3\left[m\sum_{i=1}^{m} i + m\sum_{i=1}^{m}(2i-1)\right]\\ &= \frac{9}{2}m^3+\frac{3}{2}m^2.\end{aligned}$$

The total cost of the degree one nodes attached to each x_k is

$$\begin{aligned}
3R\left[(m+1+1)+(m+1+3)+\ldots+(m+1+2m-1)\right] \\
&= 3R\left[m(m+1)+\sum_{i=1}^{m}(2i-1)\right] \\
&= 3R[m^2+m+m^2] \\
&= 6Rm^2+3Rm.
\end{aligned}$$

(iii) The remaining $c_{i,j}$ nodes which are not on the s_1s_2-path will be adjacent to either vertex v_j or v_{j+1}, and to each of these $c_{i,j}$ nodes will be attached three paths of $m-1$ vertices. T^* will be balanced so if a node hangs from v_j, then another node will hang from v_{m-j+2}. Since n is odd, there will be an even number of these extra $c_{i,j}$s to distribute so it will be possible to balance the tree. The total cost of “garbage collecting” these extra nodes is

$$\begin{aligned}
m(n-1)\left[(m+1)+3\sum_{i=1}^{m-1}(i+m+1)\right] \\
&= (mn-m)\left[\frac{9m^2-m}{2}-2\right] \\
&= \frac{9}{2}m^3n-\frac{1}{2}m^2n-2mn-\frac{9}{2}m^3+\frac{1}{2}m^2+2m.
\end{aligned}$$

Thus, the cost of T^* is

$$cost_3(T^*) = 4m^2+3m+\frac{9}{2}m^3n-\frac{1}{2}m^2n-2mn+6m^2R+3mR = K,$$

and thus T^* is optimal for k-MSST.

For the “if” direction, we show that an optimal spanning tree T^* of G must have an s_1s_2-path that includes m vertices $c_{i,j}$ and none of the element vertices x_k. Also, the path from an element vertex x_k to a source vertex will not include any vertex $c_{i,j}$ which is not on the s_1s_2-path. If the s_1s_2-path contains an element vertex, x_r, then the length of the intra-source path is at least $2m+2$. Also, for one of the two sources, at least half of the paths in the tree from that source to the element vertices must include x_r, and after taking into account the paths to the remaining element vertices, there are still $m(n-1)(3m-2)$ additional nodes in the graph to be counted.

From this, we can estimate the cost of such a tree T' to be

$$
\begin{aligned}
cost_3(T') \quad > \quad & \{s_1 s_2\text{-path}\} + \{\text{ paths to element vertices which include } x_r\} \\
& + \{\text{ paths to remaining element vertices }\} + \{\text{ additional nodes }\} \\
\geq \quad & \sum_{i=1}^{2m+2} i + (\frac{3m}{2} - 1)\left[\sum_{i=1}^{m}(i + 2m + 1) + (3m + 2)R\right] \\
& + (\frac{3m}{2} + 1)\left[\sum_{i=1}^{m}(i + 1) + (m + 2)R\right] + m(n - 1)(3m - 2) \\
= \quad & \frac{(2m + 2)(2m + 3)}{2} + (\frac{3m}{2} - 1)\left[\frac{m^2 + m}{2} + 2m^2 + m + 3mR + 2R\right] \\
& + (\frac{3m}{2} + 1)\left[\frac{m^2 + m}{2} + m + mR + 2R\right] + 3mn^2 - 2mn - 3m^2 + 2m \\
= \quad & 2m^2 + 5m + 3 + \frac{3m}{2}\left[m^2 + m + 2m^2 + 2m + 4mR + 4R\right] \\
& - 2m^2 - 2mR + 3mn^2 - 2mn - 3m^2 + 2m \\
= \quad & \frac{9}{2}m^3 + \frac{3}{2}m^2 + 7m + 3m^2n - 2mn + 6m^2R + 4mR \\
= \quad & \frac{69}{2}m^3 + \frac{23}{2}m^2 + 7m + 27m^4n + 18m^3n + 3m^2n - 2mn \\
> \quad & 30m^3 + \frac{23}{2}m^2 + 3m + 27m^4n + 18m^3n - \frac{1}{2}m^2n - 2mn \\
= \quad & 4m^2 + 3m + \frac{9}{2}m^3n - \frac{1}{2}m^2n - 2mn + 6m^2R + 3mR \\
= \quad & K.
\end{aligned}
$$

Therefore, the $s_1 s_2$-path in an optimal tree must not include any element vertex x_r.

We define $\mathcal{T} = \{T$ a spanning tree of $G | s_1 s_2$-path in T does not contain an element vertex $x_k\}$. By Observation 2.3, a spanning tree T^* is optimal for k-MSST if $cost_3(T^*) = \frac{1}{|S|} \cdot \min_{T \in \mathcal{T}}\{cost_1(T)\}$.

To calculate $\min_{T \in \mathcal{T}}\{cost_1(T)\}$, fix an arbitrary $s_1 s_2$-path along the "triples" gadgets and look at the minimum distance in G of each vertex v to the path. By Observation 2.4, we can use this distance to find the minimum cost. Note that there are $2m + 1$ vertices on the path, $3m + m(n - 1)$ vertices at distance 1 from the path, $3mn$ vertices each at distance from 2 to m

from the path, and $3mR$ vertices at distance $m+1$ from the path. Summing these up,

$$\begin{aligned}
\min_{T\in\mathcal{T}}\{cost_1(T)\} &= N\cdot d(p) + 2\sum_{v\in V} d(v,p) \\
&= [3mR + 4m + 3m^2n - 2mn + 1](2m) \\
&\quad + 2\left[(3m + m(n-1)) + 3mn\sum_{i=2}^{m} i + 3mR(m+1)\right] \\
&= 8m^2 + 6m + 9m^3n - m^2n - 4mn + 12m^2R + 6mR \\
&= 2cost_3(T^*) = 2K.
\end{aligned}$$

Thus, by Observation 2.3, T^* has the minimum $cost_3$. Note that in T^* paths from element vertices x_k to a source only include the subset of vertices $c_{i,j}$ that lie on the s_1s_2-path. If the path from x_k to a source included a vertex $c_{i,j}$ not on the s_1s_2-path, the distance from x_k to the s_1s_2-path would increase by one. By Observation 2.4, $cost_1$ of the tree would increase and, by Observation 2.3, so would $cost_3$.

Thus, a spanning tree T with $cost_3(T) \leq K$ can only exist if the s_1s_2-path does not include any element vertices and if the path from each element vertex to a source does not include any vertex $c_{i,j}$ not on the s_1s_2-path. As there are $3m$ element vertices, m vertices $c_{i,j}$ on the s_1s_2-path, and each vertex $c_{i,j}$ directly connects to exactly 3 element vertices, G will have a spanning tree T with $cost_3(T) \leq K$ if and only if X has an exact cover. $\square$

Corollary 2.6. k-MSST is $\mathcal{NP}$-complete.

3 The Tractable Problems

The key strategy used in this section for proving that a minimum spanning tree problem has an efficient solution is to prove that a single source shortest paths spanning tree (SPST) from some point ϕ is optimal for the given cost metric. By the argument presented next, if some SPST is optimal, we can find the tree in polynomial time.

3.1 A Sufficient Set of Shortest Paths Spanning Trees

Let Q be a set of spanning trees of G such that, for all points ϕ of G, Q contains a single source shortest paths spanning tree (SPST) from ϕ. An important result, for this paper, from [7] is that we can construct Q in polynomial time. The key idea is that although there is an infinite number of points on a graph, we only need to construct a shortest paths tree from a polynomially bounded subset of these points. Therefore, to prove a problem is in $\mathcal{P}$, it suffices to show that a SPST from some point ϕ is optimal for the problem. Although this fact does not directly lead to efficient algorithms, it does at least present us with a naïve polynomial time solution which is to generate a SPST from every necessary point and then choose the tree which has minimum cost. Because this result is crucial to the proofs given later, it is described in full here with two theorems of McMahan and Proskurowski [7].

For two points of an edge, α and β, let (α, β) denote the set of all intermediate points on the edge. (Thus, for two adjacent vertices u and v, (u, v) is the set of all points on the edge (u, v).) First, for each edge (p, q) in G, define a set of points $\Gamma_{(p,q)}$ as follows. For each vertex $v \in V$, let $\gamma_v \in \Gamma_{(p,q)}$ be a point on the edge (p, q) so that for any point $\alpha \in (p, \gamma_v)$ the shortest path from α to v is through the vertex p, and likewise for any point $\alpha \in (\gamma_v, q)$, the shortest path from α to v is through the vertex q. Let $d_p(\gamma_v)$ be the distance along the edge (p, q) from p to γ_v. Then,

$$d_p(\gamma_v) = \frac{1}{2}(d_G(q, v) - d_G(p, v) + l(p, q)).$$

Thus, each of these points can be located in polynomial time. For the next two theorems, consider a set $\Gamma_{(p,q)}$ and index the vertices of G so that $d_p(\gamma_{v_1}) \leq d_p(\gamma_{v_2}) \leq \ldots \leq d_p(\gamma_{v_n})$ and consider the $|V| - 1$ intervals $(\gamma_{v_i}, \gamma_{v_{i+1}})$ for $1 \leq i < n$ where $\gamma_{v_1} = v_n = p$ and $\gamma_{v_n} = v_1 = q$. Without loss of generality, we assume the absence of "long" edges (longer than the distance between their endpoints, ie., $(u, v) \in E$ such that $l(u, v) > d_G(u, v)$).

<u>Theorem</u> 3.1. For any two points α_1 and α_2 in the interval $(\gamma_{v_i}, \gamma_{v_{i+1}})$, the set of SPSTs rooted at α_1 is the same as the set of SPSTs rooted at α_2.

<u>Proof</u>. For both α_1 and α_2, the shortest path to a vertex v_j goes through p if $j \leq i$ and through q if $j > i$. Thus, a SPST from either α_1 or α_2 contains a shortest paths tree from p to the vertices $v_1, \ldots, v_i$ and a shortest paths tree from q to the vertices $v_{i+1}, \ldots, v_n$. Therefore, the sets of all SPSTs from α_1 is identical to the set of all SPSTs from α_2. □

<u>Theorem</u> 3.2. Any SPST for a point $\alpha \in (\gamma_{v_{i-1}}, \gamma_{v_{i+1}})$ is also a SPST for the point γ_{v_i}.

<u>Proof</u>. The proof follows from Theorem 3.1 and the definition of γ_v. □

Therefore, to create Q, pick an arbitrary point $\alpha \in (\gamma_{v_i}, \gamma_{v_{i+1}})$ for each interval $(\gamma_{v_i}, \gamma_{v_{i+1}})$ along an edge, find a SPST from α, and repeat the process for each edge in G. As there are at most $|V| - 1$ intervals per edge, Q can be constructed in polynomial time. More efficient methods for forming Q are possible, and both [7] and [8] give such procedures.

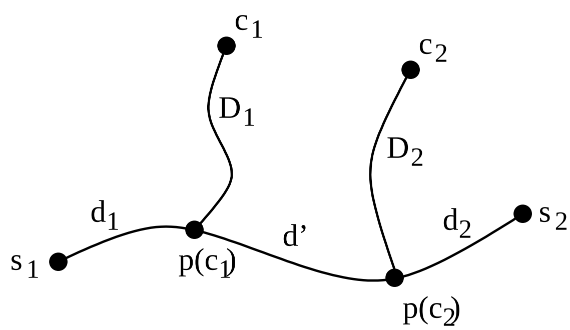


Figure 3: Diagram for the proof of Lemma 3.3

3.2 k-Source Sum of Vertex Eccentricities Spanning Tree (k-SVET)

Instance: A graph $G = (V, E)$ with a length function, $l : E \rightarrow \Re$, k sources $S = \{s_1, \ldots, s_k\} \subseteq V$, a positive integer K.

Question: Is there a spanning tree T of G such that $cost_4(T) = \sum_{v \in V} \max_{s \in S} d_T(s, v) \leq K$?

In a spanning tree T, define a vertex to be critical for a source if it is at the maximum distance from the source. Likewise, a source is critical for a vertex if it is the source at maximum distance from the vertex. (Note that these are not necessarily unique.) Let $p(v)$ be the projection of v on the $s_1 s_2$-path in the tree T. The first lemma shows that the paths between two sources s_1, s_2 and their critical vertices must intersect along the $s_1 s_2$-path.

Lemma 3.3. Given a tree T and sources $s_1, s_2 \in V(T)$, then for all critical vertices $c_1, c_2 \in V(T)$ such that $d_T(s_i, c_i) = \max_{v \in V} d_T(s_i, v)$, $i \in \{1, 2\}$, we have $d_T(s_1, p(c_1)) \geq d_T(s_1, p(c_2))$, and thus the $s_1 c_1$-path and the $s_2 c_2$-path intersect.

Proof. Let $d_1 = d_T(s_1, p(c_1))$, $D_1 = d_T(c_1, p(c_1))$, $d_2 = d_T(s_2, p(c_2))$, $D_2 = d_T(c_2, p(c_2))$, and let $d' = d_T(s_1, p(c_2)) - d_T(s_1, p(c_1))$. Assume that $d' > 0$. Figure 3 illustrates this situation. However,

$$
\begin{array}{rrcl}
 & d_1 + D_1 & \geq & d_1 + d' + D_2 \\
+ & d_2 + D_2 & \geq & d_2 + d' + D_1 \\
\hline
 & d_1 + d_2 + D_1 + D_2 & \geq & d_1 + d_2 + D_1 + D_2 + 2d' \\
 & 0 & \geq & d'
\end{array}
$$

which contradicts the assumption. $\square$

The next lemma shows that in a tree the path from each vertex to a source critical for it must include the midpoint of the path between two sources with maximum intrasource distance.

Lemma 3.4. Given a tree T, let s_1 and s_2 be two sources with maximum intrasource distance, and let χ be the midpoint of the s_1s_2-path in T. For all vertices $v \in V(T)$, the path in T from v to its critical source must include χ.

Proof. Given a vertex v assume, without loss of generality, $d_T(s_1, p(v)) \geq d_T(s_1, \chi)$. Otherwise, replace s_1 with s_2 in the following equations. Let s_j, $j \in \{1, \ldots, k\}$, be the source critical for v then, by definition, $d_T(s_j, v) \geq d_T(s_1, v)$ and $d_T(s_j, \chi) \leq d_T(s_1, \chi)$. Now, assume χ is not on the s_jv-path. Then

$$\begin{aligned} d_T(s_j, v) &< d_T(s_j, \chi) + d_T(\chi, v) \\ &\leq d_T(s_1, \chi) + d_T(\chi, v) \\ &= d_T(s_1, v). \end{aligned}$$

Thus the s_jv-path must include χ. □

The next lemma shows that we only need to consider two sources of a k-SVET instance.

Lemma 3.5. Let s_1 and s_2 be two sources with maximum intrasource distance in a tree T, and pick χ to be the midpoint on the s_1s_2-path in T. For any vertex v and any source s_i, $i = 3, \ldots, k$, either $d_T(v, s_1) \geq d_T(v, s_i)$ or $d_T(v, s_2) \geq d_T(v, s_i)$.

Proof. Assume, without loss of generality, that $d_T(s_1, p(v)) \geq d_T(s_1, \chi)$. Then, $d_T(v, s_1) = d_T(v, \chi) + d_T(\chi, s_1)$.

$$\begin{aligned} d_T(v, s_i) &\leq d_T(v, \chi) + d_T(\chi, s_i) \\ &\leq d_T(v, \chi) + d_T(\chi, s_1) \\ &= d_T(v, s_1). \end{aligned}$$

□

We are now ready to prove that there exists a SPST from some point in G which is optimal for k-SVET.

Theorem 3.6. Given a graph G with sources $s_1, \ldots, s_k \in V(G)$, there exists a point χ such that any SPST rooted at χ is an optimal tree for k-SVET.

Proof. Let T^* be an optimal tree for k-SVET and let s_1 and s_2 be two sources with maximum intrasource distance in T^*. Pick χ to be the midpoint on the s_1s_2-path in T^*. Let T_χ be a shortest paths spanning tree of G with the root χ and assume $cost_4(T^*) < cost_4(T_\chi)$. Thus, there exists some vertex v and a source s_j, $j \in \{1, \ldots, k\}$, of greatest distance from v in T_χ for which $d_{T^*}(v, s_i) < d_{T_\chi}(v, s_j)$ where s_i is the source of maximum distance from

v in T^*. By Lemma 3.5 and without loss of generality, assume $i = 1$. By Lemma 3.4, $d_{T^*}(v, s_1) = d_{T^*}(v, \chi) + d_{T^*}(\chi, s_1)$ and note that $d_{T_\chi}(v, s_j) \leq d_{T_\chi}(v, \chi) + d_{T_\chi}(\chi, s_j)$. Thus, $d_{T^*}(v, \chi) + d_{T^*}(\chi, s_1) < d_{T_\chi}(v, \chi) + d_{T_\chi}(\chi, s_j)$. From the definition of χ, we have $d_{T^*}(\chi, s_1) \geq d_{T^*}(\chi, s_j)$. This implies $d_{T^*}(v, \chi) + d_{T^*}(\chi, s_j) < d_{T_\chi}(v, \chi) + d_{T_\chi}(\chi, s_j)$ but contradicts the fact that T_χ is a shortest paths tree. Therefore, $cost_4(T^*) = cost_4(T_\chi)$ so a SPST rooted at χ is optimal for k-SVET.

□

The main result of this subsection follows directly from Theorems 3.6, 3.1, and 3.2.

Theorem 3.7. k-SVET $\in \mathcal{P}$.

3.3 k-Source Sum of Source Eccentricities Spanning Tree (k-SSET)

Instance: A graph $G = (V, E)$ with a length function, $l : E \rightarrow \Re$, k sources $S = \{s_1, \ldots, s_k\} \subseteq V$, a positive integer K.

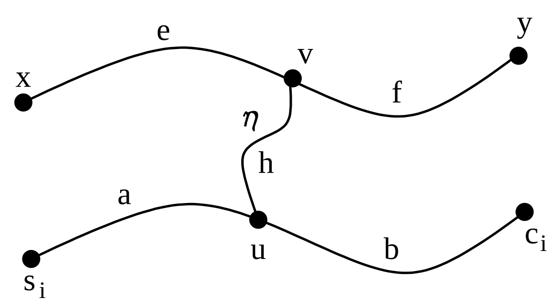
Question: Is there a spanning tree T of G such that $cost_5(T) = \sum_{s \in S} \max_{v \in V} d_T(s, v) \leq K$?

To prove that this problem is polynomially solvable we will again prove that there exists a point ϕ such that a shortest paths spanning tree rooted at ϕ has the optimal $cost_5$. The diameter of a graph is the maximum distance between any two vertices, and a shortest path of this length is called diametral. The next lemma proves that a path from a source to its critical vertex in a tree T must intersect a diametral path in T. Also, one of the endpoints of this diametral path will be critical to the source. From this lemma, we can prove that all paths between source nodes and their critical vertices will intersect at the midpoint of a diametral path in T.

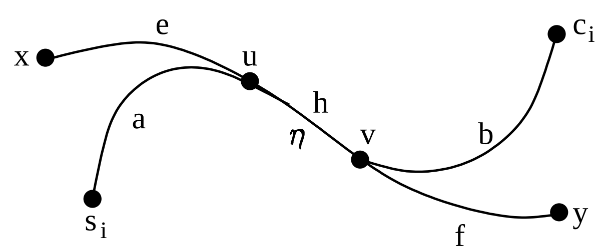
Lemma 3.8. In a tree T, let χ be the midpoint of a diametral path with x and y the endpoints of this path. For each source s_i and its critical vertex c_i, $(i = 1, \ldots, k)$, χ is on the $s_i c_i$-path in T. Moreover, if C_i is the set of critical vertices for s_i, then $C_i \cap \{x, y\} \neq \emptyset$.

Proof. First, show the xy-path intersects with the $s_i c_i$-path for some i. If the two paths do not intersect, they must be joined in T by some non-empty path η sharing a vertex u with the $s_i c_i$-path and a vertex v with the xy-path. See Figure 4a for an illustration of this situation. Note that $f \geq h + b$, otherwise the $x c_i$-path would be longer than the xy-path. Also note that $b \geq h + f$, otherwise the $s_i y$-path would be longer than the $s_i c_i$-path. Yet, these two facts imply $h \leq 0$, and thus the xy-path must intersect the $s_i c_i$-path.

The next step is to show $C_i \cap \{x, y\} \neq \emptyset$. Let η be the intersection of the xy-path and the $s_i c_i$-path, and let u and v be the (not necessarily different) endpoints of this intersection. Without loss of generality, assume the situation is as in Figure 4b. Let q be the diameter of T,



(a) Assume paths do not intersect.



(b) Assume paths do intersect.

Figure 4: Diagrams for the proof of Lemma 3.8

$q = d_T(x,y)$. As $c_i \in C_i$, $b \geq f$, and as the xy-path is a diametral path of T, $f \geq b$. Thus, $b = f$ and $d_T(s_i, y) = d_T(s_i, c_i)$. Therefore, $y \in C_i$.

Finally, let χ be the midpoint of the xy-path so $d_T(x,\chi) = d_T(y,\chi) = \frac{1}{2}q$. We show that χ lies in η and thus on the s_ic_i-path. If $\chi \notin \eta$, then either $e > \frac{1}{2}q$ or $f > \frac{1}{2}q$. If we let $e > \frac{1}{2}q$, then $h + f < \frac{1}{2}q$ hence $h + b < \frac{1}{2}q$ so $e > h + b$ and $a + e > a + h + b$, and a contradiction is reached. Also, if we let $f > \frac{1}{2}q$, then $b > \frac{1}{2}q$ which implies $b + f > q$, and a contradiction is again reached. Therefore, $e, f \leq \frac{1}{2}q$, so $\chi \in \eta$, and thus χ is on the s_ic_i-path. □

With this lemma, we can prove k-SSET $\in \mathcal{P}$.

Theorem 3.9. Given graph G with sources S and a tree T^* minimizing $cost_5$ over all spanning trees of G. A SPST rooted at χ, the midpoint of a diametral path of T^*, is also optimal for k-SSET.

Proof. Let T^* be an optimal tree, let q be the diameter of T^*, and let x and y be the endpoints of a diametral path. By Lemma 3.8, all s_ic_i-paths include χ, and x or y is critical for each source. Let T_χ be a SPST rooted at χ. Let c_i be a critical vertex for s_i in T^* and c_i^χ a critical vertex for s_i in T_χ. Then

$$
\begin{aligned}
cost_5(T_\chi) &= \sum_i (d_{T_\chi}(s_i, c_i^\chi)) \\
&\leq \sum_i (d_{T_\chi}(s_i, \chi) + d_{T_\chi}(\chi, c_i^\chi)) \\
&\leq \sum_i (d_{T^*}(s_i, \chi) + \frac{1}{2}q) \\
&= \sum_i (d_{T^*}(s_i, \chi) + d_{T^*}(\chi, c_i)) \\
&= cost_5(T^*)
\end{aligned}
$$

By the assumption that T^* is optimal, $cost_5(T) = cost_5(T_\chi)$ so T_χ is optimal as well. □

From Theorems 3.9, 3.1 and 3.2, the next theorem follows directly.

Theorem 3.10. k-SSET $\in \mathcal{P}$.

4 Conclusion

We have filled the gaps in the complexity status of certain problems of constructing optimal multi-source spanning trees with different distance related cost metrics. One of these problems was recently shown to be $\mathcal{NP}$-hard and another was shown to have an efficient solution. We have resolved the complexity status of the problems for related metrics, and two of these metrics were shown to yield $\mathcal{NP}$-hard problems while the cost under the other two metrics could be minimized in polynomial time.

Further research may include streamlining efficient algorithms for the polynomially solvable problems and finding efficient approximation algorithms and efficient algorithms on restricted classes of graphs for the $\mathcal{NP}$-hard problems. There has been some work on approximation algorithms for the more general Optimum Communication Spanning Tree in [9], and [1] presents a polynomial algorithm for Shortest Total Path Length Spanning Tree for distance hereditary graphs. Both of these results may be applied to the k-SPST problem. Relating these theoretical results to the practical applications (as those of multicast routing trees) would also be of interest.

References

[1] E. Dahlhaus, P. Dankelmann, W. Goddard, and H. C. Swart. MAD trees and distance hereditary graphs. In *Proceedings of JIM'2000*, pages 49–54, 2000.

[2] A. M. Farley, P. Fragopoulou, D. Krumme, A. Proskurowski, and D. Richards. Multi-source spanning tree problems. *Journal of Interconnection Networks*, 1(1):61–71, 2000.

[3] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York, NY, 1979.

[4] T. C. Hu. Optimum communication spanning trees. *SIAM Journal of Computing*, 3(3):188–195, 1974.

[5] D. S. Johnson, J. K. Lenstra, and A. H. G. Rinnooy Kan. The complexity of the network design problem. *Networks*, 8(4):279–285, 1978.

[6] D. W. Krumme and P. Fragopoulou. Minimum eccentricity multicast trees. Submitted, 1999.

[7] B. McMahan and A. Proskurowski. Multi-source spanning trees: Algorithms for minimizing source eccentricities. Submitted, 1999.

[8] R. Ravi, R. Sundaram, M. V. Marathe, D. J. Rosenkrantz, and S. S. Ravi. Spanning trees – short or small. *SIAM Journal of Discrete Mathematics*, 9(2):178–200, 1996.

[9] B. Y. Wu, K. Chao, and C. Y. Tang. Approximation algorithms for some optimum communication spanning tree problems. *Discrete Applied Mathematics*, 102:245–266, 2000.