Multi-Source Spanning Tree Problems

**Arthur Farley, Paraskevi Fragopoulou,
David Krumme, Andrzej Proskurowski,
and Dana Richards**

**CIS-TR-99-02
February 1999**

Department of Computer and Information Science
University of Oregon

Multi-Source Spanning Tree Problems

Arthur M. Farley* Paraskevi Fragopoulou† David Krumme‡
Andrzej Proskurowski§ Dana Richards¶

Abstract.
We consider a model of communication in a network, where multiple sources have messages to disseminate among the network membership. We propose that all the messages (one from each source) be disseminated along the same spanning tree of the network and consider the problem of constructing such a tree. One evaluative measure for suitability of the construction is the sum of distances from each source for each vertex. We show that finding the exact solution in this case is NP-hard, for any selection of the sources. We therefore investigate this problem for some restricted classes of graphs and give efficient solution algorithms for those. We also consider alternative measures of goodness of spanning communication trees.

1. Introduction

In computer networks (*e.g.*, the Internet), an increasingly prominent communication paradigm is *multicast*, whereby any of a set of network vertices called *senders* broadcasts its message to the set of *(multicast group) members*. In current Internet practice, the multicast protocol may require building an "optimal" routing tree from each of the senders to all group members; this protocol suffers an explosive growth of resource usage with the network size. We will discuss possible modifications to the above multicast protocol, whereby all senders utilize the same, "shared" routing tree.

We consider classes of multicast applications that restrict the sets of senders and receivers. For the purpose of our discussion, we will assume that the group membership includes all N vertices of the network (modeled as a simple undirected graph) and that some k of those ($1 \leq k \leq N$) are designated as sources. We will also assume that all communications take place along edges of a *spanning tree* of the network, *i.e.*, an acyclic subgraph of the network graph that includes all vertices of the network.

*Computer and Information Science Department, University of Oregon, Eugene, OR 97403 USA

†Laboratoire de recherche en informatique PRISM, Université de Versailles St-Quentin en Yvelines, 45 Avenue des Etas-Unis, Versaille CEDEX, France

‡Computer Science Department, Tufts University, Medford, MA 02155

§Computer and Information Science Department, University of Oregon, Eugene, OR 97403 USA

¶Computer Science Department, George Mason University, 400 University Drive, Fairfax, VA 22030-4444

There are several extant measures of goodness of a spanning tree, all with respect to a possible length function on the edges of the graph. One measure involves a simple sum of lengths of all edges of the spanning tree, resulting in the so called *minimum spanning tree*. Another takes under consideration the distance between vertices of the graph, being the sum of the lengths of edges along a shortest path between the two vertices (unique in a tree). For a single source, this defines a *(single source) shortest-paths tree*. Because of the possible attributes and requirements of multicast applications, several other measures come to mind. We will consider two cost measures, namely, the sum of distances from each source to all vertices and the maximum distance from each source, over all vertices. We will also discuss the complexity of the problem as parameterized by the number k of sources ($1 \leq k \leq N$).

2. Definitions

We model a network as a simple, undirected graph without loops, $G = \langle V, E \rangle$, where V is the set of N *vertices* and $E \subseteq \binom{V}{2}$ is the set of *edges*. There is a *length function* defined for the edges, $\ell : E \to Z^+$. We define a family of decision problems called *k-Source Shortest-Paths Spanning Tree (k-SPST)*, parameterized by a positive integer k.

k-SPST:

Instance: A graph $G = \langle V, E \rangle$ with the length function ℓ, k sources $s_1, \ldots, s_k \in V$, a positive integer K

Question: Is there a spanning tree T of G whose cost (the sum of edge lengths on the paths from all sources to all vertices) does not exceed K?

For $k = 1$, this problem becomes the single source shortest path problem, with well known efficient solutions (for instance, the Dijkstra-Prim algorithm). For the set of sources identical with the set of vertices of the instance graph, we have a problem whose uniform edge lengths version has been defined before as *Shortest Total Path Length Spanning Tree* (*cf.* [ND3] in [1]) and shown to be NP-complete. In a recent paper [3], Wu et al. address the all source problem and propose an efficient approximation scheme.

A very general problem of *Optimum Communication Spanning Tree* (*cf.* [ND7] in [1]) has been defined by Hu [2] based on a complete graph G with length $\ell(u, v)$ and requirement $r(u, v)$ for each edge (u, v). In this problem, the cost measure of a spanning tree T of G is defined as the sum of vertex distances in T weighted by the corresponding requirements: $\sum_{u,v \in V} d_T(u, v) \cdot r(u, v)$. By restricting the requirements $r(u, v) = 0$ for all vertices u except the sources s_i, and allowing "very large" edge lengths for $(u, v) \notin E$, the problem reduces to k-SPST.

In the next section, we will show that the 2-SPST problem is NP-complete. It remains so even in the uniform edge lengths case.

We also consider the complexity of the multi-source problem with alternative cost measure being the maximum eccentricity of a source (distance from any vertex), the *k-Source Maximum-Eccentricity Spanning Tree (k-MEST).*

k-MEST:

Instance: A graph $G = (V, E)$ with length function ℓ, k sources $s_1, \ldots, s_k \in V$, a positive integer K

Question: Is there a spanning tree T of G whose cost (maximum eccentricity of a source) does not exceed K?

We show later that certain restrictions of the problem are efficiently solvable.

3. NP-Completeness Reduction

Before presenting a reduction, we state an observation about the structure of our measure of goodness for a solution of the 2-SPST problem.

OBSERVATION 1. The cost of a spanning tree T of a graph G with N vertices, two of which, s_1 and s_2, are sources is equal to $N \cdot d(p) + 2\sum_{v \in V} d(v, p)$, where p is the unique path between the sources, $d(p)$ its length, and for a vertex v, $d(v, p)$ is the shortest distance of v to a vertex of p.

PROOF. For any vertex v of G, the shortest paths to the two sources include twice the path between v and the nearest vertex w of p and the two subpaths of $p : (w, s_1)$ and (w, s_2), whose lengths add to $d(s_1, s_2)$. Summing over all vertices of G gives the above formula. □

We will use the well-known 3-SAT decision problem to reduce to the 2-SPST problem.

3-SAT ([LO2] in [1]):

Instance: A set of disjunctive clauses, C_i, $1 \leq i \leq m$ involving literals of boolean variables $x_1, \ldots, x_n$

Question: Is there a truth value assignment to the variables that satisfies all the clauses?

THEOREM 3.1. *2-SPST is NP-complete even when all edge lengths are equal.*

PROOF. A reduction from 3-SAT problem. Given an instance $C_1, \ldots, C_m$ of 3-SAT, construct an instance of 2-SPST by combining the following "gadgets":

A gadget corresponding to the variable x_i consists of a four-cycle with vertices labeled, in order, $x_i', x_i^T, x_i'', x_i^F$. For each i, $1 \leq i < n$, identify vertices x_i'' and x_{i+1}'.

A gadget corresponding to a clause $C_j = (y_{j_1}, y_{j_2}, y_{j_3})$ consists of a vertex labeled c_j, adjacent to $L = 2nm$ degree-1 vertices; this makes it costly to not

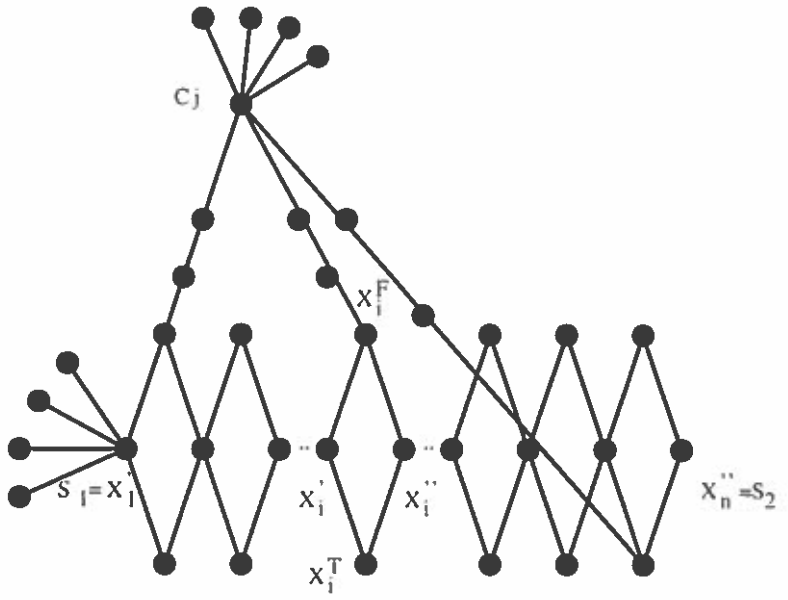

Fig. 3.1: The construction; a clause $c_j = \bar{x}_1 \wedge \bar{x}_i \wedge x_n$.

satisfy the clause (see below). Additionally, the clause vertex c_j is connected to the variable gadget vertices that correspond to the literals y_{j_1}, y_{j_2} and y_{j_3} through paths of $n-1$ vertices each. That is, for each j, $1 \leq j \leq m$, identify the vertex representing literal y_{j_i} of C_j with the F or T vertex of the corresponding variable gadget, depending on whether y_{j_i} is satisfied by assigning `FALSE` or `TRUE` to the variable.

The two source vertices are $s_1 = x_1'$ and $s_2 = x_n''$. This ensures that in any spanning tree, the path between s_1 and s_2 is of length at least $2n$ (and equal to $2n$ only if the path does not pass through a clause vertex). Finally, one of the sources, say s_1, is adjacent to $M = 4m(n+1)(L+1)$ degree-1 vertices; this makes it costly to have a long path from s_1 to s_2 in the spanning tree. This graph has $N = M + 3n + 1 + Lm + 3nm - 2m$ vertices. Completing the construction, the constant K of the instance of 2-SPST is defined to be $2nN + 2M + 2n + m(n+1)(2L+3n) - 4m$.

Let us assume that the instance of 3-SAT is satisfiable. We construct a spanning tree by connecting s_1 and s_2 through the variable gadgets, for each i including the vertex x_i^T or x_i^F (call those "essential variable vertices") according to the satisfying truth assignment. The length of this path p is thus $2n$. We complete the tree by removing edges between a clause vertex and two of its degree-2 neighbors, leaving a connection to an essential variable vertex, for each clause vertex.

There are $3m$ paths in the graph connecting clause vertices to the s_1, s_2-path. We designate each one as "short" or "long" according to whether it attaches to an essential or inessential variable vertex. Let P denote the

number of long paths. Since each clause has at least one short path, $P \leq 2m$. By Observation 1, the cost of the tree is $2nN + 2\sum_{v \in V} d(v, p)$, where the component $2\sum_v d(v, p)$ equals

0	for vertices v on the s_1, s_2-path
$2M$	for M degree-one vertices v adjacent to s_1
$2n$	for n non-essential variable vertices v
$2nm$	for m clause vertices v
$2Lm(n+1)$	for mL degree-1 vertices v adjacent to clause vertices
$P(n(n+1)-2)$	for vertices v in long paths
$(3m-P)(n-1)n$	for vertices v in short paths

This totals $2nN + 2M + 2n + 2Lm(n+1) + 3n^2m - nm + 2P(n-1) \leq K$.

We call a spanning tree T *legal* if it corresponds to a truth assignment (i.e., if no clause vertex lies on the path from s_1 to s_2 in T). If the instance of 3-SAT is not satisfiable, then the cost of a legal tree is higher than K, since the degree-1 neighbors of an unsatisfied clause are at distance $n+2$ from the path between s_1 and s_2; they would contribute an extra cost of $2L$, which would make the cost exceed K even if P were 0. (P must be at least 3 because of the unsatisfied clause.)

If T is not legal, the path from s_1 to s_2 includes a clause vertex and has length at least $2n+2$. The M degree-1 neighbors of s_1 contribute an extra $2M$ to the cost of T. The cost of this tree is therefore more than:

$$\begin{aligned} N(2n+2)+2M &> 2Nn+4M \\ &\geq 2Nn+2M+8m(n+1)(L+1) \\ &\geq 2Nn+2M+2Lm(n+1)+6Lm(n+1)+8m(n+1) \\ &\geq 2Nn+2M+2Lm(n+1)+3nm(n+1)+2n > K. \end{aligned}$$

COROLLARY 1. *The 2-SPST problem is NP-complete.*

COROLLARY 2. *The k-SPST problem is NP-complete for any $k > 1$.*

4. Algorithms for 2-Source Spanning Trees

4.1 Distance sum measure

In light of the difficulty of solving the 2-SPST problem for general graphs, we will be satisfied, for the time being, to present efficient solution algorithms for some fairly restricted classes of graphs. To start with an almost trivial class of graphs, we consider first unicyclic graphs consisting of cycle vertices x_i, $0 \leq i < n$ and the corresponding trees T_i rooted in cycle vertices (*cf.* Figure 4.1). We use x_n as a synonym for x_0.

Wlog., we can assume that, in the given input graph G with $N = \sum_{0 \leq i < n} |T_i|$ vertices, the sources are the roots x_0 and x_j of trees T_0 and T_j, $j > 0$ (for $j = 0$, an optimal spanning tree T of G is identical with the single-source

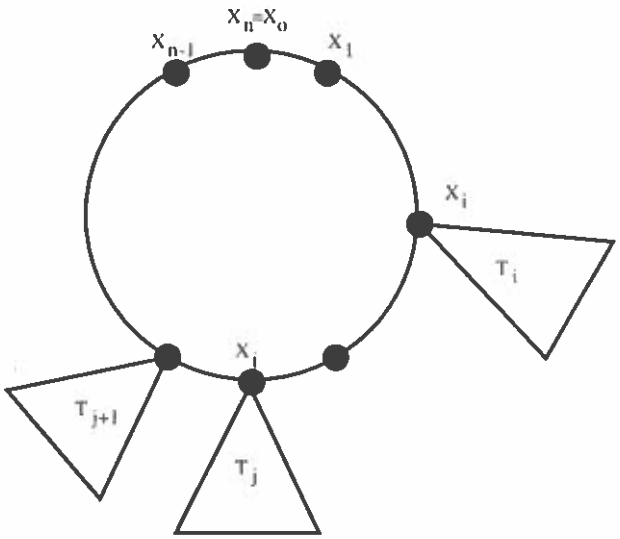

Fig. 4.1: A unicyclic graph.

shortest-paths spanning tree of G.) As we noted before, the cost of T consists of the length of the shortest path between the sources taken N times plus the distances of all vertices to that path. For each tree T_i, we denote by N_i the number of vertices in T_i and by S_i the sum of their distances to the root x_i.

Let us define function $L(x_i, x_k)$ to denote the "clockwise" distance from x_i to x_k; for instance, $L(x_i, x_k) = \sum_{i \le m < k} l(x_m, x_{m+1})$ for $0 < i \le k$ and similarly for other cases of i and k. Removing the edge (x_k, x_{k+1}) from the only cycle of G results in a tree, say $T^{(k)}$. The cost of $T^{(k)}$ is equal to, according to Observation 1, N times the sum of the distance between the sources, $L(x_0, x_j)$ when $j \le k < n$, or $L(x_j, x_0)$ when $0 \le k < j$, plus twice the distance sums S_i in T_i and twice the following sums:

$$\sum_{j < i \le k} N_i L(x_j, x_i) + \sum_{k < i < n} N_i L(x_i, x_n) + \qquad \text{for } j \le k < n \tag{4.1}$$

or

$$\sum_{0 < i \le k} N_i L(x_0, x_i) + \sum_{k < i < j} N_i L(x_i, x_j) \qquad \text{for } 0 \le k < j. \tag{4.2}$$

Finding a k to minimize the cost of $T^{(k)}$ can be done in linear time. To design such an incremental update algorithm we observe that the difference between costs of $T^{(k)}$ for two consecutive values of k can be computed in constant time.

A generalization of unicyclic graphs may involve multiple cycles C_i connected in a tree-like manner through common articulation points. Such graphs are sometimes called *cacti*. Let G be a cactus with sources s_1

and s_2. There exists a path of cycles $C_1, \ldots, C_n$ between s_1 $(= x_{1,0})$ and s_2 $(= x_{n,j_n})$, such that all (s_1, s_2)-paths pass through articulation vertices $x_{1,j_1} = x_{2,0}, \ldots, x_{n-1,j_{n-1}} = x_{n,0}$ of the consecutive intermediate cycles $C_2, \ldots, C_{n-1}$.

We will describe an algorithm to eliminate an edge from each cycle in order to obtain a minimum cost spanning tree T of G.

First we observe that for a subgraph separated from the sources by a single vertex $x_{i,j}$ of a cycle C_i, $0 \leq i \leq n$, the single-source shortest-paths spanning tree (with the articulation vertex being the source for the subproblem) provides a subtree of the optimal tree T for the input cactus G. The number of vertices in such a subtree becomes an attribute $w(x_{i,j})$ of the articulation cycle vertex $x_{i,j}$ (analogous to N_i in the unicyclic case). For each cycle C_i with vertices $x_{i,j}$ weighted by $w(x_{i,j})$, we have a separate optimization problem of choosing an edge $(x_{i,k}x_{i,k+1})$ to remove, according to the formulae (4.1) and (4.2) above. This algorithm computes the optimal spanning tree in linear time.

We observe further that this algorithm applies to any instance of the problem in which the subgraph induced by all (s_1, s_2)-paths is a cactus. Any biconnected component of the instance graph sharing an articulation vertex s with such a cactus can be optimally spanned by a shortest-paths tree with the single-source s.

A simple algorithm constructs an approximate solution to the 2-SPST problem with cost at most twice the optimal. Given a graph G with two sources s_1 and s_2, the algorithm constructs a shortest (s_1, s_2)-path p, contracts its edges collapsing the path's vertices into one "super vertex" s and then constructs a single-source s shortest paths tree T' spanning the derived graph G'. We claim that the spanning tree T of G consisting of the path p and the edges of G corresponding to edges of T' has the cost at most twice that of an optimal spanning tree T^* of G. Assume that the (s_1, s_2)-path p^* in T^* has length M. Any vertex v of G is at most $M/2$ further away from p than from p^* (this because a vertex of p^*, namely a source, can be reached from v within this extra distance). Thus in T we over-estimate the cost of T^* by at most $2n(M/2) = Mn$, which is a lower bound on the cost of T^*. This gives the following theorem.

THEOREM 4.1. *The optimal solution to the 2-SPST problem can be efficiently approximated within a factor of 2.*

4.2 Maximum eccentricity measure

Now, let us consider the alternative cost measure for an optimal spanning tree, namely, the maximum eccentricity of a source. For a cycle vertex x_i of the unicyclic graph G as above, let $d(x_i)$ be the depth of the tree T_i. Let the function $L(x_i, x_k)$ be defined as before to denote the clockwise distance around the cycle. The maximum eccentricity of a source is minimized with

the following expressions (when choosing the removed edge (x_k, x_{k+1})):

$$L(x_j, x_n)+\max\{\max_{0\le i\le k}(L(x_0, x_i)+d(x_i)), \max_{k<i\le j}(L(x_i, x_j)+d(x_i))\} \text{ for } 0 \le k < j \tag{4.3}$$

$$L(x_0, x_j)+\max\{\max_{k<i<n}(L(x_i, x_n)+d(x_i)), \max_{j\le i\le k}(L(x_j, x_i)+d(x_i))\} \text{ for } j \le k < n \tag{4.4}$$

The expressions (4.3) and (4.4) can be evaluated in time proportional to n by separately computing values of L for x_0 and x_j (as one of the arguments) and then incrementally updating them with $d(x_i)$, to obtain the relevant maximum values. A final scan of those values will yield the minimum maximum source eccentricity and thus a linear time algorithm for the 2-MEST problem on unicyclic graphs.

If the sources are located not on the cycle but in the trees, T_0 and T_j, then the distances of the sources to the cycle, $D_1 = d(s_1, x_0)$ and $D_2 = d(s_2, x_j)$, respectively, would modify the formulae (4.3) and (4.4) in the obvious manner. This suggests a pseudo-polynomial time solution algorithm for the 2-MEST problem on cacti. (For unit edge lengths or lengths with magnitude polynomially bounded by the input length, this represents an efficient algorithm. We are unable, as yet, to solve the problem on cacti with arbitrary edge lengths.) Namely, for each cycle C_i, as above, solve the problem for the *unicyclic* case for *all* possible values of D_1 and D_2 and then use the dynamic programming approach to find an optimal spanning tree T.

The simple single-source shortest-paths algorithm that produces an approximation to the 2-SPST problem gives also an approximate solution to the 2-MEST problem, as follows. Let T' be a single-source s shortest-paths tree in the graph G' obtained from the instance graph G by collapsing vertices of a shortest path between the sources s_1 and s_2 into vertex s. We obtain a spanning tree T of G by adding to the original edges of T' the contracted edges of the (s_1, s_2)-path. The maximum source eccentricity, M, of a vertex in T is not larger than the distance between the sources, m, plus the maximum distance d from any vertex to the single source s of the derived graph G'. Being shortest path distances in G, both d and m are lower bounds on the maximum source eccentricity M^* in the optimal spanning tree T^* of G. Thus we have $M \le 2M^*$.

5. Pseudo-polynomial Algorithm for k-MEST Problem

If the multi-source maximum-eccentricity problem seems easy, it is because it is easy – for edges with uniform length. In that case, the answer can be obtained by a brute-force low-order polynomial time algorithm based on computing single-source shortest-paths trees, as presented below. Unfortunately, in the general case the shortest-paths algorithm has to be executed a number of times that is proportional not only to the number of edges but

also to their lengths. The latter may not be polynomial in the size of the problem instance.

We will show that for an instance graph G with sources $S = \{s_1, \ldots, s_k\}$ and unit edge lengths, there exists either a vertex x, or an edge (y, z), such that the shortest-paths tree (from either x or the "midpoint" of (y, z), respectively) in G minimizes the maximum source eccentricity.

We first formulate two facts relating source eccentricities in an optimal tree of G, say T^*, to those in a shortest-paths tree, say T_x, for a certain vertex x. Let q be the maximum intra-source distance in T^* and let M^* be the maximum source eccentricity in T^*. Let P be a path of length q between a pair of sources, say s_1 and s_2, in T^*. Let x be the midpoint (equidistant from s_1 and s_2) on P. If q is odd, x will be an auxiliary vertex subdividing the middle edge (y, z) of P; we define the lengths of (x, y) and (x, z) to be $\frac{1}{2}$. We observe that this modification does not change the solution to the original problem. The following is implied by our definitions.

FACT 5.1. In T^*, no source $s \in S$ is more than $q/2$ farther away from x, i.e., $d_{T^*}(x, s) \leq q/2$. For any vertex v in G, the distance in T^* to x, $d_{T^*}(x, v) \leq M^* - q/2$.

Now, let T_x be a shortest-paths tree of G (possibly modified with the auxiliary vertex x, as per above) with the single source x. The construction implies the following fact.

FACT 5.2. For any vertex v in G, the distance in T^* to x is not smaller than the distance from v to x in T_x.

From these simple facts follows that the maximum source eccentricity in T_x does not exceed M^*.

THEOREM 5.1. *For G, $S = \{s_1, \ldots, s_k\}$, M^*, q and x, as defined above,* $\max_{v \in V, s \in S} d_{T_x}(v, s) = M^*$.

PROOF. The distance $d_{T_x}(v, s)$ is bounded by the sum of distances from x to v and s which is bounded by M^*, as per the Facts 5.1 and 5.2: $d_{T_x}(v, x) + d_{T_x}(x, s) \leq d_{T^*}(v, x) + d_{T^*}(x, s) \leq (M^* - q/2) + q/2 = M^*$. □

The above theorem implies a brute-force algorithm that requires only polynomial time (as a function of the instance size):

For G, $S = \{s_1, \ldots, s_k\}$, compute a single-source shortest-paths tree for each vertex x and the midpoint of each edge (y, z) in G (as defined above). A tree that minimizes source eccentricity is the solution to the given instance of k-MEST problem with uniform edge length.

A reduction from a general instance of k-MEST (with integral, non-uniform edge lengths) involves the construction of a derived graph, G', with edges of uniform length. G' is obtained from G by replacing each edge (y, z) with a path of $\ell(y, z)$ edges (with all new internal vertices of the path). It is easy

to see that the minimum source eccentricity in the optimal spanning tree T' of G' is the minimum source eccentricity of the spanning tree of G uniquely determined by T'. The reduction is polynomial-time only when edge lengths are bounded by a polynomial (in the number of vertices in G). This remark limits the usefulness of our algorithm in the general case of k-MEST.

Acknowledgment

We thank Mikael Goldman for letting us use his modifications to our general case proof yielding the proof for the uniform edge lengths case of 2-SPST.

References

[1] Garey and Johnson, *Computers and Intractability*, Freeman (1971)

[2] T.C. Hu, Optimum communication spanning trees, *SIAM J. Computing*, 3 (1974), 188-195;

[3] B.Y. Wu, G. Lancia, V. Bafna, K.-M. Chao, R. Ravi and C.Y. Tang, A polynomial time approximation scheme for minimum routing cost spanning trees. in *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, (1998), 21-32;