The Delay-Constrained Minimum Spanning Tree Problem*

Hussein F. Salama
Cisco Systems
170 W. Tasman Dr.
San Jose, CA 95134
hsalama@cisco.com

Douglas S. Reeves
N.C State University
CSC Department
Box 8206, Raleigh, NC 27695
reeves@eos.ncsu.edu

Yannis Viniotis
N.C. State University
ECE Department
Box 7911, Raleigh, NC 27695
candice@eos.ncsu.edu

Abstract

We formulate the problem of constructing broadcast trees for real-time traffic with delay constraints in networks with asymmetric link loads as a delay-constrained minimum spanning tree (DCMST) problem in directed networks. Then, we prove that this problem is NP-complete, and we propose an efficient heuristic to solve the problem based on Prim's algorithm for the unconstrained minimum spanning tree problem. Simulation results under realistic networking conditions show that our heuristic's performance is close to optimal. Delay-constrained minimum Steiner tree heuristics can be used to solve the DCMST problem. Simulation results indicate that the fastest delay-constrained minimum Steiner tree heuristic, DMCT, is not as efficient as the heuristic we propose, while the most efficient delay-constrained minimum Steiner tree heuristic, BSMA, is much slower than our proposed heuristic and does not construct delay-constrained broadcast trees of lower cost.

1 Introduction

Real-time applications will heavily utilize the network resources in the near future. Real-time traffic is usually bandwidth intensive and requires quality of service (QoS) guarantees from the underlying network. Many real-time applications will involve all nodes in a given network. Distributed real-time control applications and the broadcasting of critical network state information are just a few examples of such applications. Therefore, there is a need for efficient routing algorithms which define cost as a function of the utilized link bandwidth, and are capable of constructing low-cost broadcast trees (spanning trees) that satisfy the constraints imposed by the QoS requirements. The delay constraint, i.e., the upper bound on end-to-end delay, is an important QoS requirement, because most real-time applications, and the interactive ones in particular, are delay-sensitive. No previous work has been reported on delay-constrained broadcast trees. However, a number of delay-constrained multicast routing algorithms have been proposed during the past few years. A delay-constrained shortest path heuristic was proposed in [1], and several cost-efficient, but quite complex, delay-constrained minimum Steiner tree heuristics were proposed in [2, 3, 4, 5]. A thorough evaluation of different multicast routing algorithms, unconstrained and delay-constrained, can be found in [6]. The destination set of the minimum Steiner tree problem may be any subset of the network nodes. In the special case when the destination set includes all nodes in the network, multicasting reduces to broadcasting, and the minimum Steiner tree problem reduces to the, usually less complex, minimum spanning tree (MST) problem. The delay-constrained minimum Steiner tree problem is *NP*-complete [2], and it remains *NP*-complete even after the delay constraint is removed [7]. We prove in this paper that the delay-constrained MST (DCMST) problem is also *NP*-complete. However, several polynomial time algorithms exist for the unconstrained MST problem [8, 9].

In this paper, we propose a DCMST heuristic, the bounded delay broadcast heuristic (BDB). BDB is designed for general networks with asymmetric link loads (directed networks). We evaluate BDB's performance using simulation. Since no other DCMST heuristics have been proposed in the literature, we compare BDB's performance to two delay-constrained minimum Steiner tree heuristics: BSMA [5] (Bounded Shortest Multicast Algorithm), the most cost-efficient delay-constrained minimum Steiner tree heuristic [6], and DMCT [3], a fast and distributed heuristic. Note, however, that DMCT was designed for networks with symmetric link loads (undirected networks) only.

We show that the heuristic we propose outperforms DMCT, not only in the case of directed networks, but also in the special case of undirected networks. We also compare our heuristic to BSMA.

This paper is organized as follows. In section 2, we formulate the DCMST problem in directed networks and prove that it is *NP*-complete. In section 3, we present our heuristic for solving the problem. Then in section 4, we evaluate the average performance of the heuristic and compare it to optimum, DMCT, and BSMA using simulation. Section 5 concludes the paper.

2 Problem Formulation

A communication network is represented as a directed network $G = (V, E)$, where V is a set of nodes and E is a set of directed links. Any link $e \in E$ has a cost $c(e)$ and a delay $d(e)$ associated with it. $c(e)$ and $d(e)$ may take any positive real values.

A spanning tree $T(s) \subseteq E$ is rooted at a source node $s \in V$ and contains a path from s to any node $v \in (V - \{s\})$. The total cost of a tree $T(s)$ is simply:

$$Cost(T(s)) = \sum_{t \in T(s)} c(t) \qquad (1)$$

A path $P(T(s), v) \subseteq T(s)$ is the set of tree links connecting

*This work was supported in part by the Center for Advanced Computing and Communication at North Carolina State University, and by AFOSR grant F49620-96-1-0061.

s to $v \in V$. The cost of the path $P(T(s), v)$ is:

$$Cost(P(T(s), v)) = \sum_{t \in P(T(s), v)} c(t) \quad (2)$$

and the end-to-end delay along that path is:

$$Delay(P(T(s), v) = \sum_{t \in P(T(s), v)} d(t) \quad (3)$$

Thus the maximum end-to-end delay of a spanning tree is:

$$Max\_Delay(T(s)) = \max_{v \in V} \left(Delay(P(T(s), v)) \right) \quad (4)$$

The DCMST problem in directed networks constructs the spanning tree $T(s)$ rooted at s that has minimum total cost among all possible spanning trees rooted at s which have a maximum end-to-end delay less than or equal to a given delay constraint Δ. The same problem can be expressed as a decision problem as follows.

Delay-Constrained Minimum Spanning Tree (DCMST) Problem: *Given a directed network $N = (V, E)$, a nonnegative cost $C(e)$ for each $e \in E$, a nonnegative delay $D(e)$ for each $e \in E$, a source node $s \in V$, a positive delay constraint Δ, and a positive value B, is there a spanning tree $T(s)$ that satisfies:*

$$Cost(T(s)) \leq B, \quad (5)$$

$$Max\_Delay(T(s)) \leq \Delta? \quad (6)$$

Theorem 1 *DCMST is NP-complete unless all link costs are equal.*

Proof. Clearly DCMST is in *NP*, because a nondeterministic algorithm can guess a set of links to form the tree, then it is possible to verify in polynomial time that these links do form a tree, that this tree spans all nodes in the network, that the total cost of the tree is less than B, and that the maximum end-to-end delay from the source node s to any node $v \in V$ is $\leq \Delta$.

When all link costs are equal, the solution to the polynomial time least-delay tree problem can be used to answer the DCMST decision problem.

The next step is to transform a known *NP*-complete problem to DCMST. We will use the Exact Cover by 3-Sets (X3C) problem which has been shown to be *NP*-complete in [10]. It can be stated as follows.

Exact Cover by 3-Sets (X3C) Problem: *Given a finite set $X = \{x_1, \ldots, x_{3p}\}$ and a collection $Y = \{y_1, \ldots, y_q\}$, $q \geq p$, of 3-element subsets of X, is there a subcollection $Y' \subseteq Y$ such that every element of X occurs in exactly one member of Y', i.e., the members of Y' are pairwise disjoint, and $\bigcup_{y \in Y'} y = X$?*

Given an arbitrary instance of X3C, we construct the network $N = (V, E)$ for the corresponding instance of DCMST as follows:

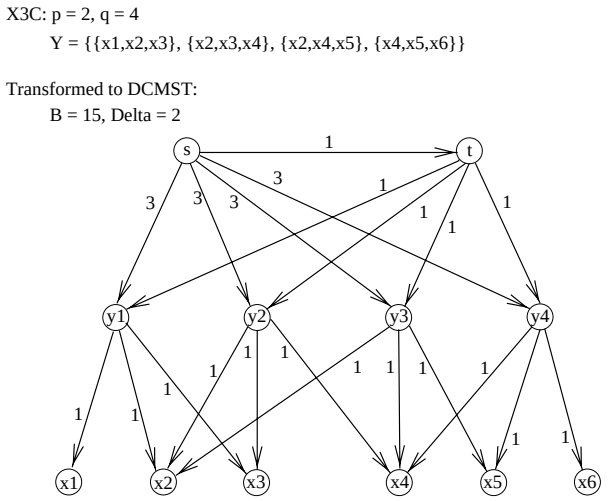

Figure 1: Equivalent instances of X3C and DCMST. Link costs are shown. All link delays are set to 1 (not shown).

- Every element of X is represented by a node in the network, and also every member of Y is represented by a node. Two additional nodes are introduced: a source node s and another node t. Therefore:

$$V = s \cup t \cup X \cup Y \quad (7)$$

- Add the following set of directed links E to interconnect the nodes:

$$\begin{aligned} E \quad = \quad & (s,t) \cup \{(s, y_i) : i = 1, \ldots, q\} \cup \\ & \{(t, y_i) : i = 1, \ldots, q\} \cup \\ & \{(y_i, x_j) : x_j \in y_i, i = 1, \ldots, q, \\ & \qquad j = 1, \ldots, 3p\} \end{aligned} \quad (8)$$

- Assign the following costs to the links:

$$C(e) = \begin{cases} 3 & e \in \{(s, y_i) : i = 1, \ldots, q\} \\ 1 & otherwise \end{cases} \quad (9)$$

- Unit delay is assigned to all links:

$$D(e) = 1 \quad \forall e \in E \quad (10)$$

- Set the delay constraint Δ of DCMST to:

$$\Delta = 2 \quad (11)$$

- Set the positive value B to:

$$B = 5p + q + 1 \quad (12)$$

Figure 1 illustrates this transformation. It is clear that it can be done in polynomial time. The final step is to show that a feasible spanning tree exists for the above instance of DCMST if and only if the 3-set collection Y has an exact cover Y'. If for a given instance of X3C, any element $x \in X$ is not in any member y of the collection Y then

that instance does not have an exact cover. The above transformation, when applied to such an instance, will result in an unconnected network, thus no spanning tree can be constructed.

If, however, for a given instance of X3C, each element $x \in X$ appears in at least one member y of Y, the resulting network will be connected. Since all link delays are set to 1 and the delay constraint Δ is set to 2, the number of hops along any path starting at s in any feasible solution $T(s)$ can not exceed 2. As a result all nodes x_i, $i = 1, \ldots, 3p$, must be leaf nodes in any feasible solution. Each node x_i will be attached to some node $y_j \in Y$ via a unit cost link. The y_js used to reach the x_is must therefore be directly attached to s via the expensive links of cost 3. Lower cost but longer paths, via the node t, can be used from s to the y_js which are not used to reach the x_is. Let there be m nonleaf y_j nodes (y_js that are used to reach the x_is) and n leaf y_j nodes, where $q = m + n$. Thus the total cost of any spanning tree, $T(s)_{sat\_\Delta}$, that satisfies the delay constraint Δ is:

$$\begin{aligned} Cost(T(s)_{sat\_\Delta}) &= \overbrace{3p*1}^{x_i s} + \overbrace{m*3}^{\text{nonleaf } y_j s} + \overbrace{n*1}^{\text{leaf } y_j s} + \overbrace{1*1}^{t} \\ &= 3p + 3m + (q - m) + 1 \\ &= 3p + 2m + q + 1 \end{aligned} \quad (13)$$

Each component in equation 13 represents the cost of the links used to attach a set of nodes, indicated as the label of that component, upstream towards s. From equations 5, 12 and 13 it follows that the condition

$$m \leq p \quad (14)$$

must be satisfied for the total cost of the tree to be less than B. On the other hand, each of the m nonleaf y_js can be used to reach at most 3 x_is. Therefore

$$m \geq p \quad (15)$$

nonleaf y_j nodes are needed to reach the $3p$ x_is. Combining conditions 14 and 15, we get

$$m = p \quad (16)$$

as the only possible number of nonleaf y_js that results in a feasible solution to that instance of DCMST. For such a feasible solution to exist, each of the p nonleaf y_js must be used to reach exactly 3 different x_is, or in terms of the X3C problem: the members of the collection Y corresponding to the p nonleaf y_j nodes must be pairwise disjoint. p pairwise disjoint 3-sets cover $3p$ elements, and thus form an exact cover of the set X. This means that the existence of a feasible solution of an instance of DCMST implies the existence of a feasible solution for the corresponding instance of X3C.

Conversely, if an instance of X3C has an exact cover of X, that exact cover $Y' \subseteq Y$ must have exactly $|Y'| = p$ members to cover the $3p$ elements of X. The corresponding instance of DCMST will have a feasible solution with $|Y'|$ nonleaf y_js, a maximum end-to-end delay of $2 = \Delta$, and a total cost of $3p + |Y'| * 3 + |Y - Y'| * 1 + 1 = 3p + 3p + q - p + 1 = 5p + q + 1 = B$. This completes the proof. □

In the next section we propose a simple and efficient heuristic for the DCMST problem to avoid the exponentially growing execution times of the optimal solutions.

3 The BDB Heuristic

The BDB heuristic consists of two phases; phase 1 is executed once, followed by phase 2, which is also executed once. The result of phase 1 is a moderate-cost spanning tree which satisfies the delay constraint. Phase 2 attempts to replace high-cost links in the tree with links of lower cost, without violating the delay constraint, and without introducing loops.

Phase 1 of BDB is based on Prim's MST algorithm. It starts with a subtree containing the source node, s, only, and adds one node at a time to the subtree without violating the delay constraint, as outlined above, until the subtree spans all nodes in the network. If at any point during the first phase, the heuristic can not find any node that can be added without violating the delay constraint, it resorts to delay relaxation. BDB relaxes the delays by choosing a node, n, that is already in the subtree and replaces the subtree link connecting it upstream towards s with another link, such that n remains connected to s and the end-to-end delay from s to n is reduced. If no more positive delay relaxation can be achieved and BDB fails to add any remaining unconnected nodes to the subtree, the algorithm fails.

In phase 2, expensive tree links are removed and replaced with cheaper links without violating the imposed delay constraint, and without creating any routing loops. A detailed discussion of BDB is given in [11].

Kompella's DMCT algorithm [3] resembles phase 1 of BDB, but its approach to relax the delays is different from BDB's. However, DMCT does not have an equivalent to phase 2 of BDB. DMCT's execution stops as soon as all destination nodes are included in the tree being constructed, because its aim is to construct a Steiner tree and not a spanning tree.

4 Experimental Results

In this section, we compare the performance of BDB to the optimal delay-constrained minimum spanning tree algorithm, OPT. We implemented OPT as a branch and bound algorithm. Since BDB is the first heuristic designed specifically for solving the DCMST problem, there no other DCMST heuristics to compare it to. So we compare BDB's performance to two heuristics designed for solving the more general, but more complex, delay-constrained minimum Steiner tree problem: BSMA and DMCT.

We used simulation for our experimental investigations to avoid the limiting assumptions of analytical modeling. Full duplex directed networks of different sizes with homogeneous link capacities of 155 Mbps (OC3) were used in the experiments. The positions of the nodes were fixed in a rectangle of size $4000 * 2400$ Km^2, roughly the area of the USA. A random generator was used to create links interconnecting the nodes [11]. The output of this random generator is always a connected network in which each node's degree is ≥ 2. The probability of a link to exist between any two nodes is a function of the distance between these two nodes. We adjusted the parameters of the random generator to yield networks with an average node degree of 4.

The propagation speed through the links was taken to be two thirds the speed of light. The propagation delay was dominant under these conditions, and the queueing component was neglected when calculating link delays and end-to-end delays.

We used variable bit rate (VBR) video for the broadcast sources . These are realistic bursty traffic sources used

in real-time applications with delay constraints. A broadcast tree that includes a link e, reserves a fraction of e's bandwidth equal to the equivalent bandwidth of the traffic generated by the corresponding broadcast source. The link cost, $c(e)$, is equal to the reserved bandwidth on that link, i.e., equal to the sum of the equivalent bandwidths of the traffic streams traversing that link.

The experiment we ran, compares the different algorithms when each of them is applied to create a broadcast tree for a given source node generating VBR traffic with an equivalent bandwidth of 0.5 Mbps, under given network loading conditions. For each run of the experiment, we generated a random set of links to interconnect the fixed nodes, we selected a random source node, and we generated random background traffic for each link. The equivalent bandwidth of each link's background traffic was a random variable uniformly distributed between B_{min} and B_{max}. This represents the cost of each link. We simulated networks with asymmetric link loads, and, therefore, the cost of a link $e = (u, v)$, $c(e)$, and the cost of the reverse link $e' = (v, u)$, $c(e')$, were independent random variables. The asymmetry of the link loads increased as the range of the link loads, i.e., the difference between B_{max} and B_{min}, increased.

The experiment was repeated with different link loading conditions, different delay constraints, and different network sizes. For each setting of these parameters, we measured the total cost of the broadcast tree. We ran the algorithms repeatedly until confidence intervals of less than 5%, using 95% confidence level, were achieved. On the average, 300 different networks were simulated in each experiment, in order to reach such confidence levels. At least 250 networks were simulated in each case. We simulated the following algorithms: BDB, DMCT, BSMA, and OPT. We could not apply OPT to networks with more than 20 nodes due to its excessive execution time. The other algorithms were applied to networks with up to 200 nodes. Therefore, we show the percentage excess cost of each algorithm relative to OPT in case of 20-node networks only. When discussing results for networks with more than 20-nodes, we show the percentage excess cost of each algorithm relative to BSMA which was previously shown in [6] to be the best performing minimum Steiner tree heuristic.

Figure 2 shows the cost of a broadcast tree versus the range of link loads, $B_{max} - B_{min}$, for 20-node networks and a tight delay constraint of 0.03 seconds. We increased the range of link loads, $B_{max} - B_{min}$, from 0 to 120 Mbps while keeping the average link load, $(B_{max} + B_{min})/2$, fixed at 65 Mbps at all times. All algorithms perform equally well in case of zero range of link loads, because all link costs are equal and there is nothing to be minimized in that case. DMCT's costs deviate further from optimal as the range of link loads increases. It is up to 23% worse than OPT when the range of link loads is large. BDB and BSMA remain close to optimal even when the range of link loads is large. The tree costs of BDB and BSMA are equal throughout the entire range of link loads simulated in this experiment. Thus BDB has a cost performance that matches the most cost efficient delay-constrained minimum Steiner tree heuristic. BDB's tree costs are within 7.5% from OPT. We found that DMCT's tree costs are comparable to the costs of the intermediate trees resulting from phase 1 of BDB. Thus the phase 2 of BDB is capable of reducing the cost of its initial tree by up to 15% in some cases.

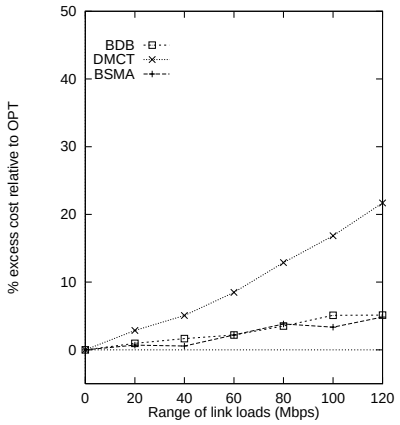

Figure 2: Percentage increase in the cost of a broadcast tree relative to optimal, variable range of link loads, 20 nodes, delay constraint $\Delta = 0.03$ seconds.

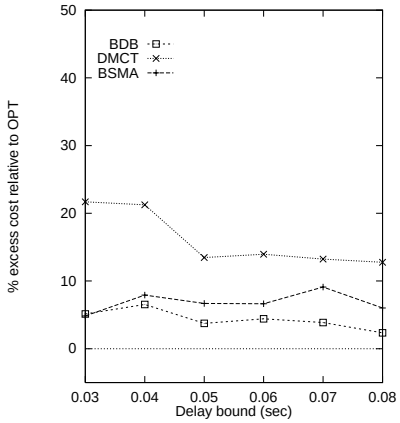

Figure 3: Percentage increase in the cost of a broadcast tree relative to optimal, variable delay constraint Δ, 20 nodes, B_{min} = 5 Mbps, B_{max} = 125 Mbps.

We repeated the experiment with a fixed range of link loads and a variable delay constraint. The results are shown in figure 3 for 20-node networks. A tight delay constraint limits the algorithms' ability to construct low-cost trees, because there aren't many possible solutions for the problem. As the delay constraint increases, its effect on restricting the algorithms' efficiency in constructing low-cost trees diminishes, and the algorithms are capable of constructing lower cost trees. When the delay constraint increases further, the effect on the resulting tree costs is minimal, because the solution of the DCMST problem approaches the solution of the unconstrained MST problem.

Figure 4 shows the percentage excess costs of BDB and DMCT relative to BSMA when the network size varies from 20 nodes to 200 nodes while keeping the delay constraint and the range of link loads fixed. BDB performs as well as BSMA does throughout the entire range of network sizes, while DMCT's performance relative to BSMA deteriorates as the network size increases. For 200-node networks, DMCT's trees are up to 35% more expensive than BSMA and BDB.

Finally, in figure 5, we present the average execution

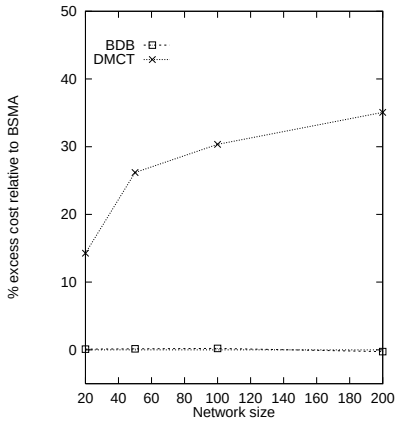

Figure 4: Percentage increase in the cost of a broadcast tree relative to BSMA's tree cost, variable network size, delay constraint $\Delta = 0.03$ seconds, B_{min} = 5 Mbps, B_{max} = 125 Mbps.

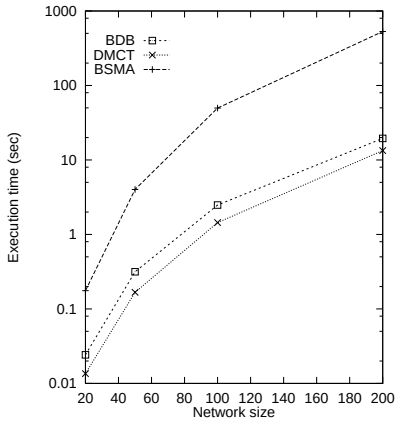

Figure 5: Average Execution time, variable network size, delay constraint $\Delta = 0.03$ seconds, B_{min} = 5 Mbps, B_{max} = 125 Mbps.

times of BDB, DMCT, and BSMA versus the network size. Note, however, that our code for these algorithms was not optimized for speed. Figure 5 shows that the average execution times of both BDB and DMCT grow at the same rate, and are always within the same order of magnitude, with BDB being constantly slower than DMCT by approximately 60%. BSMA is at least one order of magnitude slower than BDB and DMCT and its execution time grows at a faster rate.

5 Conclusions

We studied the problem of constructing broadcast trees for real-time traffic with delay constraints in networks with asymmetric link loads. We formulated the problem as a DCMST problem in directed networks, and then we proved that this problem is *NP*-complete. We proposed a bounded delay broadcast (BDB) heuristic to solve the DCMST problem. The heuristic consists of two phases. The first phase is based on Prim's algorithm and constructs a moderate-cost delay-constrained spanning tree. The second phase reduces the cost of that tree by replacing tree links with lower cost links not in the tree, without violating the imposed delay constraint. Simulation results show that BDB's performance is close to optimal. Delay-constrained minimum Steiner tree heuristics can be used to solve the DCMST problem. We compared BDB to the best delay-constrained Steiner heuristic with respect to tree cost, BSMA, and the fastest delay-constrained Steiner heuristic, DMCT. Our experimental results indicate that BDB is a fast and efficient DCMST heuristic. It constructs delay-constrained broadcasts trees with close to optimal cost, and its execution times are comparable to the fastest delay-constrained minimum Steiner tree heuristic.

References

[1] Q. Sun and H. Langendoerfer, "Efficient Multicast Routing for Delay-Sensitive Applications," in *Proceedings of the Second Workshop on Protocols for Multimedia Systems (PROMS '95)*, pp. 452–458, October 1995.

[2] V. Kompella, J. Pasquale, , and G. Polyzos, "Multicasting for Multimedia Applications," in *Proceedings of IEEE INFOCOM '92*, pp. 2078–2085, 1992.

[3] V. Kompella, J. Pasquale, and G. Polyzos, "Two Distributed Algorithms for the Constrained Steiner Tree Problem," in *Proceedings of the Second International Conference on Computer Communications and Networking (IC ^{3}N '93)*, pp. 343–349, 1993.

[4] R. Widyono, "The Design and Evaluation of Routing Algorithms for Real-Time Channels," Tech. Rep. ICSI TR-94-024, University of California at Berkeley, International Computer Science Institute, June 1994.

[5] Q. Zhu, M. Parsa, and J. Garcia-Luna-Aceves, "A Source-Based Algorithm for Delay-Constrained Minimum-Cost Multicasting," in *Proceedings of IEEE INFOCOM '95*, pp. 377–385, 1995.

[6] H. Salama, D. Reeves, and Y. Viniotis, "*Evaluation of Multicast Routing Algorithms for Real-Time Communication on High-Speed Networks*." accepted for publication in the *IEEE Journal on Selected Areas in Communications*.

[7] R. Karp, "Reducibility among Combinatorial Problems," in *Complexity of Computer Computations* (R. Miller and J. Thatcher, eds.), pp. 85–103, Plenum Press, 1972.

[8] R. Prim, "Shortest Connection Networks and Some Generalizations," *The Bell Systems Technical Journal*, vol. 36, no. 6, pp. 1389–1401, November 1957.

[9] H. Gabow, Z. Galil, T. Spencer, and R. Tarjan, "Efficient Algorithms for Finding Minimum Spanning Trees in Undirected and Directed Graphs," *Combinatorica*, vol. 6, no. 2, pp. 109–122, 1986.

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

[11] H. Salama, *Multicast Routing for Real-time Communication on High-Speed Networks*. PhD thesis, North Carolina State University, Department of Electrical and Computer Engineering, 1996. Available from `ftp://osl.csc.ncsu.edu/pub/rtcomm/rtcomm.html`.