Multicasting for High-Bandwidth Delay-Sensitive Multimedia Applications

Vachaspathi P. Kompella
Joseph C. Pasquale
George C. Polyzos

Computer Systems Laboratory
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, California 92093-0114

E-mail: pasquale@cs.ucsd.edu

Abstract

We investigate high bandwidth delay-sensitive multicast routing as an optimization problem. A solution would be especially useful for multi-user multimedia applications that have high performance requirements. Typically, such an application consists of a source that sends a large amount of data to a subset of nodes of the network. We desire that the least amount of bandwidth be used, while ensuring that the end-to-end delay from the source to any destination node be bounded by a specified delay tolerance. The problem is *NP*-complete. We present a polynomial-time approximation algorithm using a tree forwarding technique. Empirical results on random graphs show the solution to be near optimal.

Keywords: multicast, tree forwarding, Steiner tree, multimedia applications.

1 Introduction

Multicasting is the transmission of identical data from one source to several destinations, as opposed to unicasting which is from one source to one destination, and broadcasting, which involves communication from one source to all nodes in the network. A multicast channel is a more natural group communication construct because it sends data transparently to members of the group. With communication over computer networks becoming more and more popular, it is expected that this form of communication will be an important mode of interaction between geographically separated entities.

Multicasting has been recognized to be a useful communication paradigm for several reasons:

- Inter-process communication among a group of cooperating processes.
- Data transfer to all parties involved in a conference.
- Locating any one resource belonging to a class of resources all addressable by a common group network address[2].

Multicasting has been used in several networks and distributed systems, such as the V kernel [6], the Andrew file system [19], group activities, *e.g.*, conferencing, shared editing, *etc.* [9,18]. In addition, it is expected that when it becomes more popular, multimedia communication will promote other applications such as public news bulletin boards (multimedia versions of Videotex, ticker tape, USENET), department store catalogs, closed circuit television, shared blackboards/editors, group debugging sessions with audio support.

All these applications transmit high-bandwidth data and expect the delay to be low enough to enable the user to experience aural and visual continuity. To appreciate the magnitude of the problem of continuous media communication [3], consider the following figures. For continuous motion video, the screen is refreshed 30 times per second. A 640x480 screen requires 7 megabytes of frame buffer memory. Thus, uncompressed motion video requires data rates of 220 megabits/sec.

Telephone quality sound with a spectrum of 4 KHz requires data rates of about 64 kilobits/sec (although 16 kilobits/sec has been reported to be sufficient [7]), while CD quality sound requires about 1.4 megabits/sec. With compression techniques, the bandwidth requirements for both audio and video traffic reduce by about 1 or 2 orders of magnitude. Thus, even with high-speed networks like B-ISDN or FDDI, bandwidth is still a scarce resource in the context of multimedia applications.

Delay tolerance, on the other hand, is a very subjective measure. Delay tolerance limits range from about 40 to 600 milliseconds [5]. The propagation delay alone can play an important role in determining the quality of communication. For example, propagation delay across the United States is about 20 milliseconds. Satellites delays are even larger and magnify the problem of global communication. These figures show that unless low delay routes are found, interactive multimedia communication may not be possible.

Earlier work done on multicasting has emphasized network management details, *e.g.*, modifications to existing network protocols to support multicasting [1,8,20], operating systems support for multicasting [15], and protocols for reliable multicasting [12]. Some work has been done in constructing optimal or near optimal trees for multicasting using tree forwarding techniques[1]. These include:

- Constructing a near optimal Steiner tree, *i.e.*, a tree that spans a given subset of nodes in the network, and has minimum total cost [21]. The weight of the tree could correspond to the bandwidth that the multicast uses.

- Constructing a multicast tree with shortest paths from the source to each destination [8].

[1] In tree forwarding multicasting, the data from the source is forwarded along the edges of a tree that spans all the nodes in the multicast group.

This could correspond to minimizing the delay on the path from source to each destination.

A good discussion of multicasting techniques, their advantages and disadvantages, can be found in [11].

We believe that a multicast routing algorithm that optimizes on bandwidth utilization will be essential for multimedia networking. However, in addition to minimizing the utilized bandwidth, the routing algorithm must also determine routes that bound the delay from source to destination.

This paper describes a routing problem that we believe will be an important factor in the success of multimedia communication. Section 2 defines the problem, along with the assumptions about the network and the traffic. Section 3 describes a heuristic we have developed to determine good multicast routes. Section 4 explains our approach to investigating the performance of the algorithm. Section 5 presents our results, and section 6 outlines the conclusions, and discusses our future work in refining the algorithm.

2 The Multicast Tree Problem

In this section, we describe the environment of an application of high-bandwidth delay-sensitive multicasting. We are given a network G with node set V and edge set E. Each edge in the network has two characteristics, a cost and a delay. The cost of an edge defines the charge for using that edge. The delay on the edge is the sum of the processing and switching delays, as well as the transmission and propagation delays.

The network is a point-to-point network, with each node representing the main gateway of an institution to the outside world. Within a node, there may be a number of local area networks. Multicasting to these will be a local problem, following Deering's concept of hierarchical multicasting[8]. The links out of a node represent the physical connections that it has to other nodes. We will look at how to multicast between the nodes of the point-to-point network.

A multicast on this network is defined by four parameters, a source node s, a set of destination nodes S, a flow of data f from the source, and a delay tolerance D. The problem is to find a tree that spans each of the nodes in S, has the least cost, and ensures the delay along the path from s to each node in S is bounded above by D. We call this tree an optimal constrained multicast tree (OCMT).

We assume that the data reaching any destination follows exactly one path. The paths that represent optimal routing from the source to all the destinations form a tree. We associate a cost with each edge, which could be any measure, *e.g.*, the utilized bandwidth on the edge, the charge for using the link, or even a cost inversely proportional to the residual capacity of the link (to take into account congestion). The cost of the optimal constrained multicast tree is the sum of the costs of its edges.

The flow f describes a constant bandwidth requirement of the application. For example, the flow could be the minimum bandwidth needed to maintain reasonable quality communication. There is an inclination to believe such parameters may be necessary to ensure that the network knows what minimum quality of service it must provide the application[10].

The delay tolerance D defines the sensitivity of the application to delay. The application has perfect behavior if the data reaches the destination within the delay constraint, and fails if the data reaches too late. This is a reasonable condition because human perception of delay is not significant when the delay is low. For example, experiments[14] have shown that as long as the delay is small, clipping (when words are cut off because the data does not reach in time to fill the audio chip's buffer) is not perceptible. However, for larger delays, clipping is sufficiently noticeable as to be distracting.

The optimal constrained multicast tree takes into account both delay and cost. We present next an algorithm which finds an optimal constrained multicast tree given a network, a set of multicast group nodes, and a delay tolerance.

In this study, we shall assume the cost of an edge is directly proportional to the bandwidth utilized. Given a constant flow f, the total cost of communication is the product of the flow and the number of edges in the multicast tree. Thus, we can consider f to be a unit flow, and all edges to contribute a unit cost each[2].

3 The Multicast Tree Algorithm

The problem of finding a low-cost multicast tree has already been investigated[21]. The routing algorithm makes use of a very useful graph theoretic solution known as the Kou-Markowsky-Berman (KMB) algorithm[17]. The KMB algorithm generates an approximate Steiner tree, *i.e.*, a tree that spans a subset S of nodes in the graph, and has minimum cost. The Steiner tree problem has been shown to be NP-complete, even if the edge weights are all equal to 1[13,16].

The first step of the KMB algorithm is to construct a complete graph on the nodes in S. The edges between these nodes are weighted by the length of the shortest path between the endpoints of the edge in the original graph. Then, the minimum spanning tree of the complete graph is determined. Finally, each edge of the tree is expanded into the shortest path it represents.

The strength of the KMB algorithm lies in condensing most of the relevant information of the network into a complete graph on the nodes in S, using polynomial time. It then operates on a very much smaller graph to find a good approximation to the Steiner tree in polynomial time. However, the KMB algorithm does not address the optimal constrained multicast tree problem. A Steiner tree has no source or directionality. It does not account for delays on a paths. For this reason, it is

[2] We note that the problem is NP-complete, even with the simplification outlined. A proof appears in Appendix A.

possible to construct an example where the KMB algorithm fails to generate an OCMT. Consider the graph in Fig. 1. The source is node A, the set S is {B, C, D}, and the delay constraint is 14. The (cost, delay) parameters for each edge are shown in Fig. 1(a). Fig. 1(b) shows the complete graph on S, and Fig. 1(c) shows the final Steiner tree. Note that the path AB-BC-CD has a total delay of 15, violating the delay constraint.

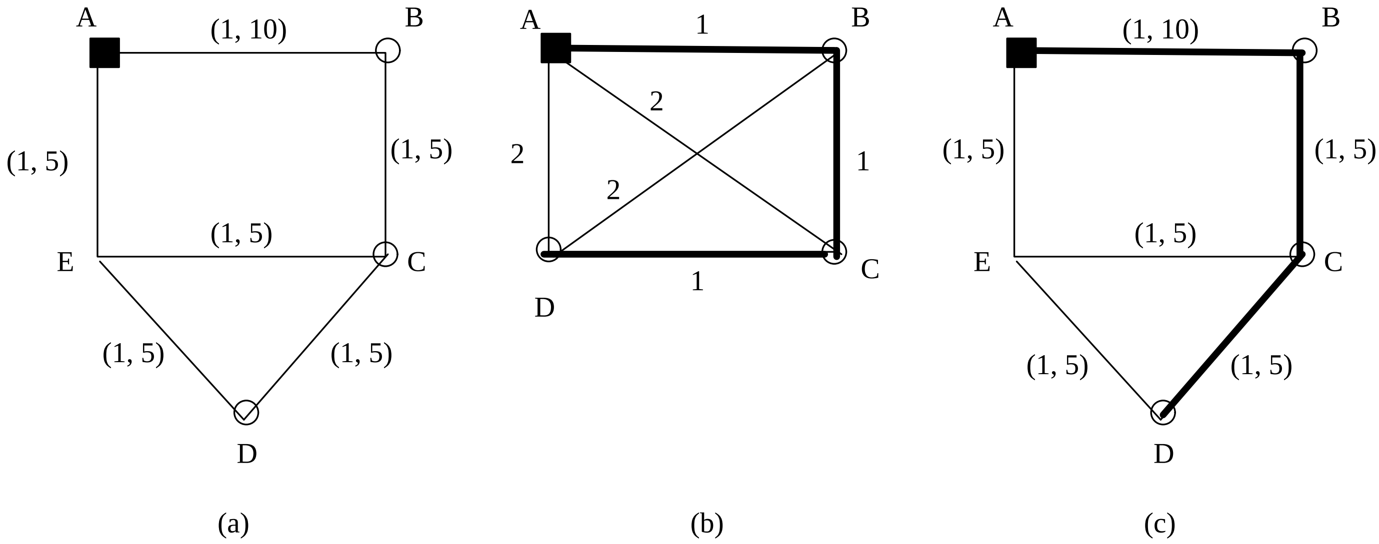

Figure 1: The Steiner tree does not produce a multicast tree with delay tolerance D = 14. a) The given graph. b) The minimum spanning tree of the complete graph on the set S. c) The Steiner tree.

We now describe the algorithm we developed for generating an optimal constrained multicast tree. The first step is to create a complete graph G' on the nodes in S and the source s. The cost of edge (v, w) is the weight of the least cost path from v to w which has delay under D. Thus, if $Weight[(v,w)]$ is the weight on the edge (v,w), and $W(v,w,d)$ is the weight of the least cost path from v to w with delay less than d, then

$$Weight[(v,w)] = \min_{k \,\epsilon\, S, 0 \leq d \leq D} \{W(v,k,d) + W(k,w,D-d)\}$$

This is a simple extension of Floyd's all-pairs shortest path algorithm. The delay along edge (v, w) is the sum of the delays on the edges forming the path from v to w.

The next step is to construct a spanning tree on the complete graph G'. The tree is built using Dijkstra's method of constructing spanning trees. It starts from s, and adds edges to the partially constructed spanning tree so that at all times the set of edges is connected. The choice of edges is decided by a critical measure, M_C, based both on the edge weight and the edge delay:

$$M_C = \frac{Weight[(v,w)]}{D - (P[v] + Delay[(v,w)])}$$

where

$$\begin{aligned} Weight[(v,w)] &= \text{weight of the edge } (v,\ w) \text{ in } G' \\ P[v] &= \text{delay on the path from } s \text{ to } v \\ Delay[(v,w)] &= \text{Delay on the edge } (v,\ w) \text{ in } G' \end{aligned}$$

$Weight[(v,\ w)]$ is the number of edges that will carry the data, if the path from v to w is used. This factor is to be minimized. The denominator of M_C is the residual delay tolerance along the path from s to w. We maximize this value because the more the residual delay tolerance, the less constrained the solution space is. The edge with the least value of M_C is the next edge to be added to the spanning tree. The final step is to expand the edges in the tree to the paths in G, and eliminate any loops. A description of the algorithm is given in Fig. 3 at the end of the paper.

Fig. 2(b) shows the complete graph G' and the spanning tree based on the critical measure M_C. Fig. 2(c) shows the OMCT. Note that the OMCT has four edges, although the minimum Steiner tree has only three edges.

The KMB algorithm takes $O(n^3)$ to find the shortest paths, and $O(n^2 \log n)$ for determining the tree, where n is the number of nodes in the graph The most expensive part of the MTA algorithm is the computation of the shortest paths under the delay constraint. This takes time $O(n^3D^2)$, where D is the delay constraint. The construction of the OMCT takes $O(n^3)$ time. The optimal solution

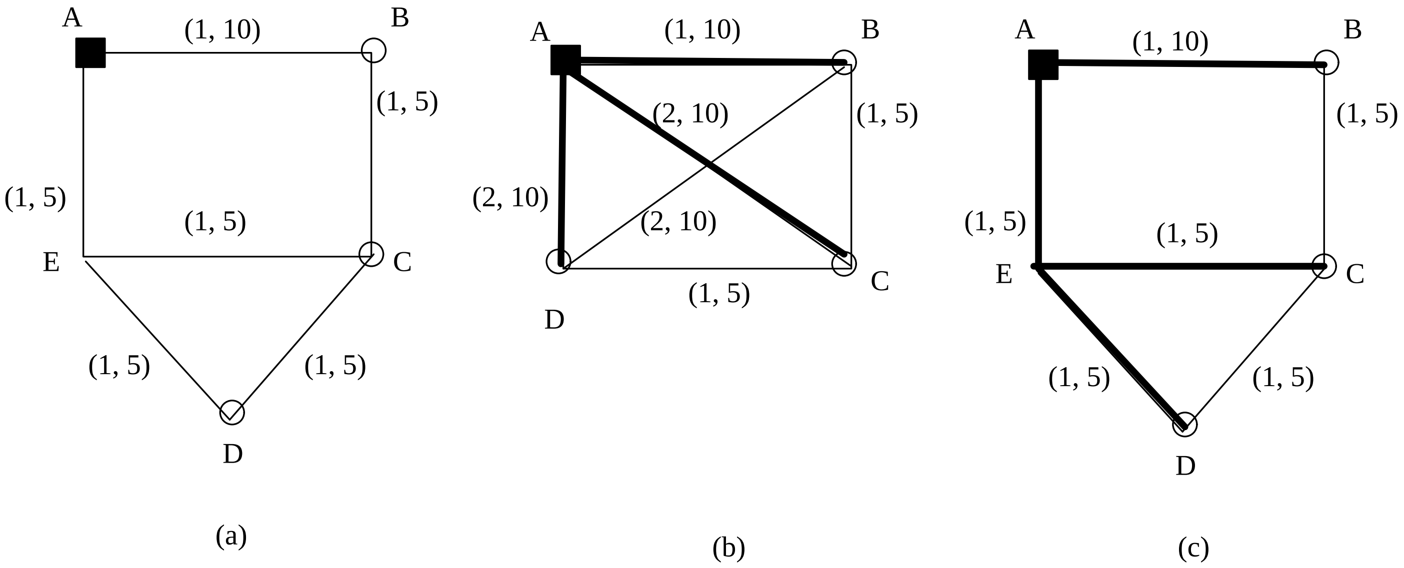

Figure 2: The multicast tree algorithm generates a solution with delay tolerance $D = 14$. a) The given graph. b) The spanning tree using the measure M_C. c) The optimal constrained multicast tree.

could take as much time as needed to examine all the spanning trees in the graph, or $O(n^{n-2})$ for a complete graph.

4 Experiments

One of the important factors in determining the utility of the routing algorithm is its average case performance. The KMB algorithm has an average case performance far better than its worst case performance [21]. We found that MTA also has this property. To see what the average case performance of MTA was, we ran the algorithm on randomly generated graphs.

We needed some reasonable figures to measure the algorithm against. One obvious choice was the optimal solution. The optimal solution is the least cost tree spanning all nodes in S, that satisfies the delay constraint. The solution is found by examining all spanning trees using a branch-and-bound depth first search. In the process of constructing the optimal tree, if the cost of the subtree is greater than the minimum solution found, or the delay constraint is not satisfied, then the subtree can never be part of a solution. At this point, the DFS can backtrack and take another branch.

A second choice was the random solution. This was constructed by building a tree using randomly selected edges in the graph G until all the nodes in S were spanned. We felt that any algorithm that did not perform better than a random solution was not an acceptable algorithm. In other words, we consider the cost of the random solution to be a realistic performance bound [3].

The graphs were randomly generated with the following parameters. Each graph had to be connected, with every node having degree between 1 and 4, with most nodes having degree 4. (Typical installations have degrees of the same order). The delay on each edge was a randomly chosen value between 1 and 8.[4]

An experiment generated $\mathcal{R}$ graphs with the same number of nodes, multicast group size, and delay tolerance. The result of the multicast tree algorithm, the optimal solution and the random solution for each graph were compared. The experiments had the following parameters:

number of runs $\mathcal{R}$:	1 000
number of nodes n:	10, 11, 12, ..., 20
degree d:	$1 \leq$ degree ≤ 4
size of S (including s):	3, 4, 5
delay on each edge:	$1 \leq$ delay ≤ 8, uniformly random
delay tolerance D:	15, 20, 25, 30

Table 1: Parameters for the experiments.

We have two measures of performance for the algorithm. First, in an absolute sense, we can compute the average normalized difference in tree cost between the multicast and optimal trees.

[3] The actual random solution cost was an average of 50 random solutions on the same graph.

[4] This range was chosen because we noticed that the longest path had about 3-4 edges. Thus, for a delay tolerance of 15, we could not have the average edge delay be more than about 4 in order to generate graphs that contained solutions.

We call this measure the average normalized distance between the multicast tree and the optimal solution, denoted by δ, where

$$\delta = \frac{1}{\mathcal{R}} \sum_{i=1}^{\mathcal{R}} \frac{M_i - O_i}{O_i}$$

This measure can be thought of as the additional cost incurred in using the MTA tree instead of the optimal tree. It is an indication of how frugal the algorithm is, when the penalty of being nearly correct is amortized over a number of multicasts.

Our second measure of the algorithm is the efficiency, η, the ratio of the difference in the total random cost and the total multicast cost to the difference in the total random cost and the total optimal cost over the $\mathcal{R}$ experiments, *i.e.*,

$$\eta = \frac{\sum_{i=1}^{\mathcal{R}} (R_i - M_i)}{\sum_{i=1}^{\mathcal{R}} (R_i - O_i)}$$

This measure indicates whether it pays to put some effort into constructing a route, given that picking a random route requires the least computation.

In the above,

- M_i is the tree cost in the multicast algorithm solution in the i^{th} run,
- O_i is the tree cost in the optimal solution in the i^{th} run,
- R_i is the tree cost in the random solution in the i^{th} run.

Since the value of δ ranged from 1% to 4%, a confidence interval of 0.5 was chosen. Using values of standard deviation from initial experiments, a confidence interval of 0.5 with a 95% confidence level required that $\mathcal{R} = 1\ 000$.

5 Results

The efficiency ratio, η, is a measure of how close to perfect we are on a scale of 0 to 1, with 0 being the worst (as good as random), and 1 being the best (as good as optimal). We notice in Fig. 4 that the efficiency increases with an increase in delay tolerance. In fact, if D becomes infinite, the algorithm reduces to the KMB algorithm, since the numerator is the only significant factor in M_C. This is a desirable property in view of the fact that the KMB algorithm is the basis for the best known solution for the Steiner tree problem.

We see in Fig. 5 that efficiency decreases with increase in nodes, but approaches a stable value. Since the average delay on an edge is about 4, if we consider a large network, then the nodes distant from the source will usually be unreachable within the delay bound. In all the experiments, we considered only those graphs that had solutions. Thus, nodes further away from the source than the delay constraint D do not affect the the performance of the algorithm.

The average normalized distance of the multicast solution from the optimal, δ, is a measure of the additional cost that would be incurred if the multicast routing tree were used instead of the optimal. We see from Fig. 6 that δ is quite low. This, together with the values of η, show that the performance of the algorithm is quite good both in an absolute sense and a relative sense.

The biggest advantage of the multicast algorithm is that it takes polynomial time to compute a solution, whereas the optimal takes exponential time to complete. The entire suite of experiments was run on a DecStation 5000 and took 7 days to complete. Almost all of the time was spent in obtaining the optimal solutions.

A final observation is that over the 100 000 random graphs generated, the multicast algorithm did not produce a tree that was more than twice the weight of the optimal. This suggests that the worst case performance bound of the KMB algorithm may also apply to our algorithm.

6 Conclusions

We have described a new problem in route determination for high-bandwidth delay-sensitive applications. Current network bandwidths have limited capacity to carry multiple multimedia data streams, compressed or uncompressed. Furthermore, existing modes of communication do not take into account the delay involved. We believe that the solution to the problem will facilitate multimedia communication enough to popularize it. We have presented one possible solution to the problem. Our empirical results show that the multicast tree algorithm is a step in the right direction.

We have shown that the multicast tree algorithm generates near optimal constrained multicast trees. It also shows nice behavior with larger networks and group sizes. The algorithm shows good average case performance if we use the average normalized distance from the optimal as the measure of performance. In a relative sense, the algorithm performs well on a scale ranging from the random algorithm (poor) to the optimal solution (excellent).

We are considering some other critical measures that govern edge selection. In particular, we are looking at those measures that can be tailored to provide bias towards either bandwidth or delay, depending on which is more critical. We are also examining modifications to the algorithm to provide low-cost tree updates when multicast groups change. Some work has been done in this area[4]. We are also working on a distributed version of the algorithm that will reduce the effort needed to construct the least cost paths, and also take into account status changes in the network.

7 Appendix A

Statement: The multicast tree problem is NP-complete.

Given: A graph G(V, E), with edge weights and edge delays.

Proof: It is obviously in NP. A non-deterministic algorithm can "guess" a solution, *i.e.*, a set of edges T that make up the multicast tree, in polynomial time. It takes deterministic polynomial time to verify that T is indeed a solution, because we have to check:

a) all nodes in $S \cup \{s\}$ are covered by T.

b) the edges of T form a tree.

c) the path delay from s to each node in S is bounded above by D.

Assume there exists a deterministic polynomial-time algorithm A to solve the multicast tree problem. Given a Steiner tree problem, *i.e.*, a graph, a set of Steiner points S, and the edge weights, we can construct a multicast tree problem with the following parameters. The graph is G, the multicast group is S. The source is any element of S. The delays are any finite values. The delay tolerance is ∞. The solution given by A to this multicast tree problem is exactly the Steiner tree solution. Thus, the multicast tree problem is NP-hard, and so NP-complete. We note that this proof holds even for edge weights = 1, since the Steiner tree problem is NP-complete for edge weights = 1 [13,16].

References

[1] Aguilar, L., "Datagram Routing for Internet Multicasting," *ACM Computer Comm. Rev.*, vol. 14, no. 2, 1984, pp. 58-63.

[2] Ahamad, M., Ammer, M.H., Bernabèu-Aubán, J.M., Khalidi, M.Y., "Using Multicast Communication to Locate Resources in a LAN-based Distributed System," *Proc. of the 13th Conf. on Local Computer Networks*, Oct. 1988, pp. 193-202.

[3] Anderson, D.P., Tzou, S.-Y., Wahbe, R., Govindan, R., and Andrews, M., "Support for Continuous Media in the DASH System," Tech. Rep., University of California, Berkeley, Oct. 1989.

[4] Belkeir, N.E. and Ahamad, M., "Low Cost Algorithms for Message Delivery in Dynamic Multicast Groups," *Proc. of the Ninth Int'l Conf. on Dist. Comp. Sys.*, Jun. 1989, pp. 110-117.

[5] Bell Telephone Laboratories, *Engineering and Operations in the Bell System*, AT&T Bell Labs, 1984.

[6] Cheriton, D.R. and Zwaenpoel, W., "Distributed Process Groups in the V Kernel," *ACM Trans. on Comp. Sys.*, vol. 3, no. 2, May 1985, pp. 77-107.

[7] Decina, M. and Modena, G., "CCITT Standards on Digital Speech Processing," *IEEE Journal on Selected Areas in Communication*, vol. 6, no. 2, Feb. 1988, pp. 227-234.

[8] Deering, S.E., and Cheriton, D.R., "Multicasting Routing in Internetworks and Extended LANs," *ACM Trans. on Comp. Sys.*, vol. 8, no. 2, May 1990, pp. 85-110.

[9] Ensor, J.R., *et al*, "The Rapport Multimedia Conferencing System – A Software Overview," *Proc. of the Second IEEE Conf. on Comp. Workstations*, Washington, Mar. 1988, pp. 52-58.

[10] D. Ferrari, D. C. Verma "A Scheme for Real-Time Channel Establishment in Wide-Area Networks," *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 3, April 1990, pp. 368-379.

[11] Frank, A.J., Wittie, L.D., and Bernstein, A.J., "Multicast Communication on Network Computers," *IEEE Software*, vol. 2, no. 3, 1985, pp. 49-61.

[12] Garcia-Molina, H. and Spauster, A., "Ordered and Reliable Multicast Communication," CS-TR-184-88, Computer Science Dept., Princeton Univ., Oct. 88.

[13] Garey, M.R. and Johnson, D.S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Co., N.Y., 1979.

[14] Gruber, J.G. and Strawczynski, L., "Subjective Effects of Variable Delay and Speech Clipping in Dynamically Managed Voice Systems," *IEEE Trans. on Comm.*, vol. 33, no. 8, Aug. 1988, pp. 801-808.

[15] Hughes, L., "A Multicast Interface for UNIX 4.3," *Software Practice and Experience*, John Wiley & Sons, vol. 18, no. 1, Jan. 1988, pp. 15-27.

[16] Karp, R.M., "Reducibility Among Combinatorial Problems," in *Complexity of Computer Computations*, Plenum Press, N.Y., 1972, pp. 85-103.

[17] Kou, L., Markowsky, G., and Berman, L., "A Fast Algorithm for Steiner Trees," *Acta Informatica*, no. 15, 1981, pp. 141-145.

[18] Sarin, S. and Greif, I., "Computer-Based Real-Time Conferencing Systems," *IEEE Computer*, Oct. 1985, pp. 33-45.

[19] Satyanarayan, M. and Siegal, E.H., "MultiRPC: A Parallel Remote Procedure Call Mechanism," Tech. Rep. CMU-CS-86-139, Carnegie-Mellon Univ., Aug. 1986.

[20] Topolcic, C., "Experimental Internet Stream Protocol, Version 2 (ST-II)" RFC 11xx, Apr. 1990.

[21] Wall, D.W. "Mechanisms for Broadcast and Selected Broadcast," Ph.D. thesis, Electr. Engg. Dept., Stanford Univ., Jun. 1980. Also Tech. Rep. STAN-CS-82-919, Stanford Univ.

```
/* let G(V, E) describe the network topology
 * Delay[e] is the delay on edge e.
 * s is the source node
 * S is the multicast group
 */
MulticastTree (G(V, E), Delay, s, S)
begin
 /* construct a complete graph G'(V', E') */
 V' ← S ∪ {s};
 for each v, w in V' do
 begin
   Weight[(v, w)] ← weight of least cost path from v to w in G such that delay on the path is ≤ D;
   Delay[(v, w)] ← sum of delays along the edges on the least cost path from v to w chosen above;
 end;
 /* C = the set of nodes already seen. */
 /* P[v] = delay on the path from s to v in the multicast tree */
 C ← {s};
 P[s] ← 0;
 /* do until all nodes in V' have been spanned */
 while (C <> V') do
 begin
   min ← ∞;
   for each v in C do
     for each w in V' - C do
     begin
       /* if delay from s to w is within limits, consider edge (v, w)
        * as a possible candidate for the next edge to be added to the tree.
        */
       if (D − (P[v] + Delay[(v, w)]) > 0) then
       begin
         M_C ← Weight[(v,w)] / (D−(P[v]+Delay[(v,w)]))
         if (M_C < min) then
         begin
           nextedge ← (v, w);
           min ← M_C;
         end;
     end /* until all possible candidate edges have been looked at */
     C ← C ∪ {w};
     P[w] ← P[v] + Delay[(v, w)];
   end /* loop until all nodes in V' have been spanned */;
end /* MulticastTree */
```

Figure 3: The Multicast Tree Algorithm

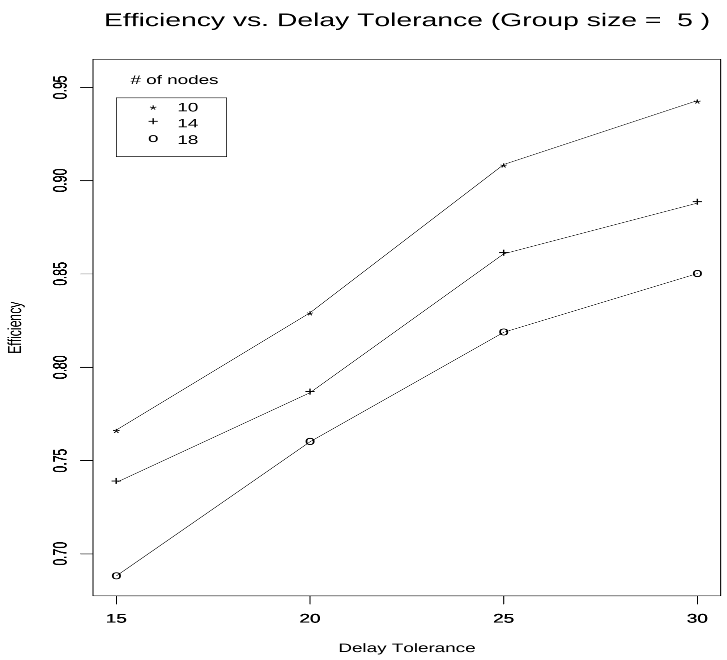

Figure 4: Efficiency as a function of delay for different number of nodes

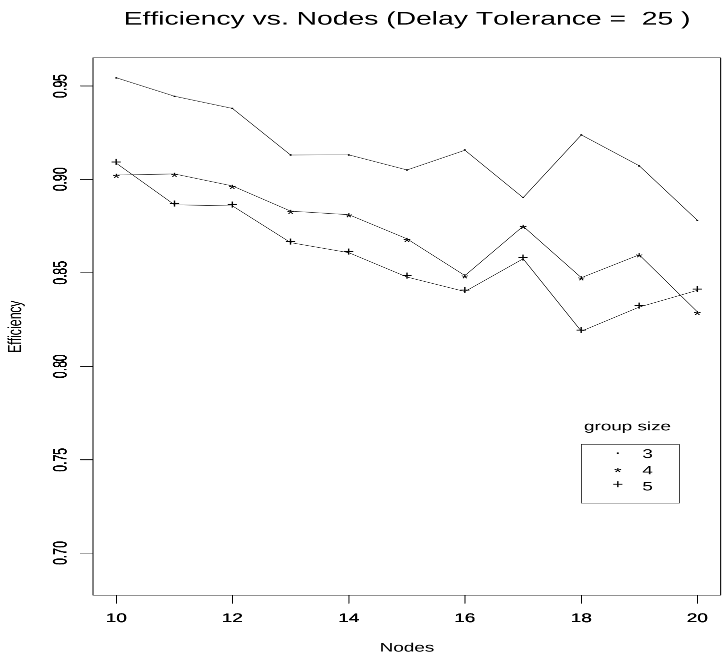

Figure 5: Efficiency as a function of nodes for different group sizes

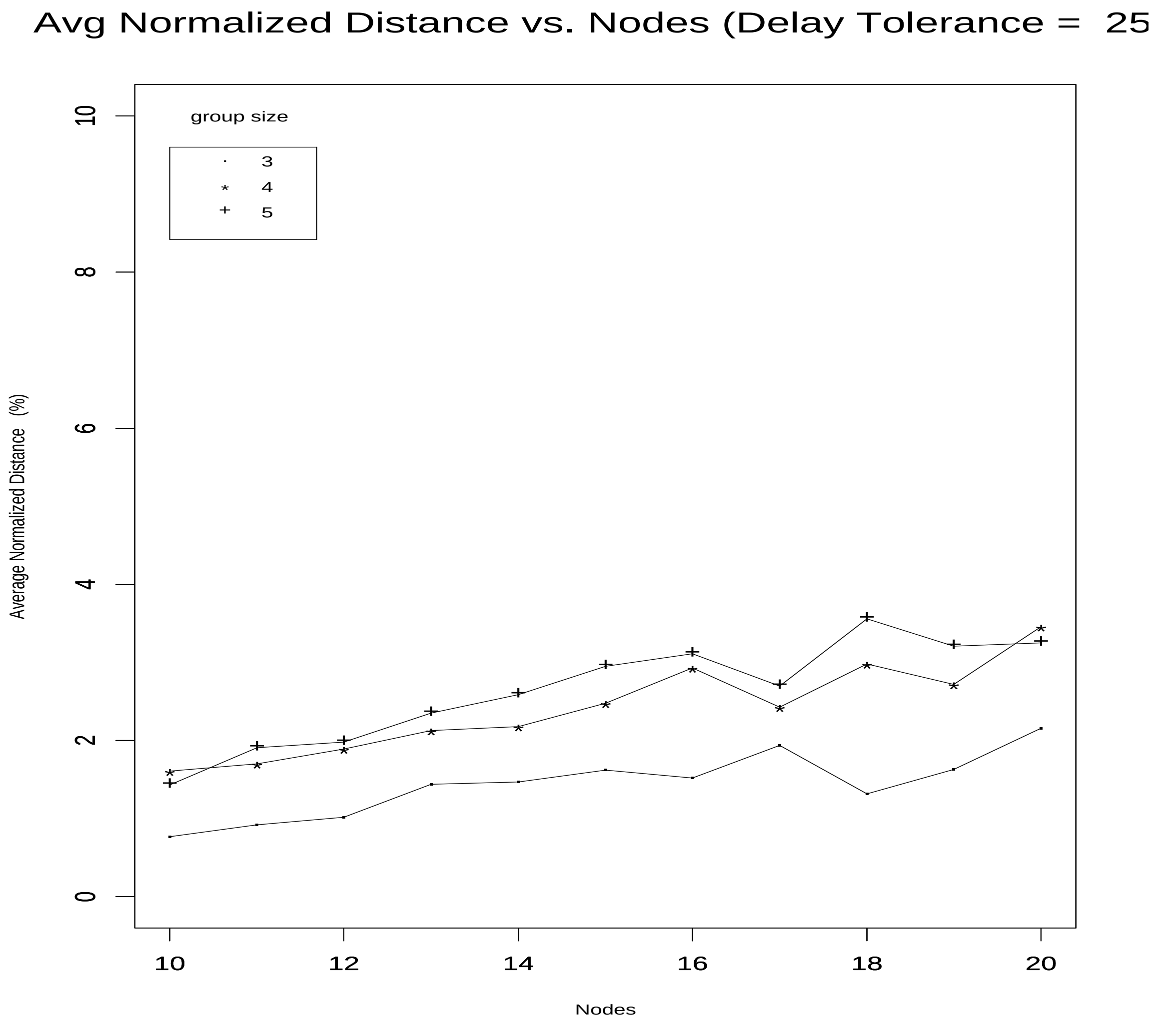

Figure 6: Normalized distance of multicast solution from optimal for different number of nodes