A Performance Analysis of Multicast Routing in Simple Mesh-Based Networks

Tim Harrison
Carey L. Williamson
Department of Computer Science
University of Saskatchewan
57 Campus Drive
Saskatoon, SK, CANADA
S7N 5A9
Phone: (306) 966-8656
FAX: (306) 966-4884
email: carey@cs.usask.ca

August 28, 1995

Abstract

This paper addresses the problem of multicast (*i.e.,* multipoint) routing in simple mesh-based networks. In particular, we derive a closed form mathematical expression for computing the average distance from a traffic source to each of the recipients of a multicast call, in a two-dimensional rectangular mesh.

The analytical expression is used to evaluate the relative performance of three simple multicast routing techniques: M-Unicast, Conference Call, and Core-Based Trees. Simulation experiments are then used to assess the accuracy of the analytical model.

The results show that, among the three routing models considered, the Core-Based Trees model provides the lowest cost routes for multicast calls, in terms of the average links required per multicast recipient. Simulation results show that the analytic model is highly accurate: always within 10%, and often within 5% of the results obtained by simulation. The accuracy of the analytical model degrades only slightly as the network load (and thus the call blocking probability) is increased.

Keywords: Networking Systems; Multicast Routing; Analysis; Simulation

1 Introduction

Multicast is the selective broadcast of a message from a source node S to a group G of M recipients, where the group G can consist of an arbitrary subset of the total nodes in a network. Multicast is distinct from *unicast*, which provides point-to-point (*i.e., one-to-one*) communication, and distinct from *broadcast*, which provides *one-to-everyone* communication. In fact, unicast and broadcast are special cases of multicast, where $N = 1$ and $N = \infty$, respectively.

The goal in multicast transmission is for a sender to only have to send *one* copy of each message, regardless of how many group members there are. In other words, rather than making multiple copies of each message and sending one copy, in turn, to each recipient, the sender relies on the network itself to make the copies, as needed, at branch points in the *multicast delivery tree*, so that one copy is delivered to each recipient. A related goal is that messages should *only go where they are needed* in the network. Again, the sender relies on the network to achieve this: the network should route a message to each member of the group, while trying to avoid routing any messages into portions of the network where no intended recipients reside.

Multicast thus has several key advantages over broadcast. These include: savings in network bandwidth, since messages are routed only where they need to go; control of information distribution, for the same reason; and fewer incoming messages for most network nodes, since fewer[1] irrelevant messages arrive.

Unfortunately, *optimal* multicast routing in an arbitrary network is computationally infeasible: it is precisely the *Steiner Tree* problem [9], which has been studied extensively in graph theory, and is known to be NP-complete.

Fortunately, in most computer networks, it is sufficient to find *good* multicast routes, even if the routes are not provably optimal. For example, the routing protocol in the Internet [8] does not guarantee optimal routing even for unicast traffic: the routing protocols are designed more for robustness (*i.e.,* finding *a* route that works) than optimality[2]. Heuristics thus play a central role in most computer network routing algorithms, for unicast and multicast.

There are several proposed routing techniques for multicast network traffic [4, 11, 12, 13, 14]. One of these is *truncated reverse path forwarding* [3]. This algorithm, which is used in the Internet multicast backbone (MBONE) [2, 6], initially broadcasts (advertises) the existence of a multicast group throughout the network, and then prunes off (*i.e.,* truncates) branches of the network where no recipients reside (*i.e.,* no subscription requests are received for the multicast group), resulting in a minimum spanning tree for the multicast group, rooted at

[1]Some "irrelevant" multicast messages may still need to be received (and forwarded) by a non-member of the multicast group if, for example, the node lies along a path from the multicast source to an intended multicast recipient.

[2]Even if an algorithm computes "shortest paths" using techniques like Dijkstra's algorithm, the routes chosen are only "good" if the inputs to the routing process (link connectivity, link distance, traffic load, etc.) are accurate and up-to-date. Thus the Internet Protocol (IP) makes no claims about optimal routing.

the source. The multicast routers, which form the backbone and the branch points of the spanning tree, need only keep track of the reverse path to the source in order to know how to forward multicast messages.

While reverse path forwarding does result in good multicast delivery trees, the approach incurs significant overhead for the setup of a multicast call. Three other techniques, which are much less costly in terms of call setup overhead, are M-Unicast, Conference Call, and Core-Based Trees [1]. These latter three are the approaches evaluated in this paper. They are described in more detail in Section 2. A recent proposal for Protocol Independent Multicast (PIM) [5] in the Internet is similar in flavour to the Conference Call and Core-Based Trees models, and thus is not evaluated separately.

This paper presents a mathematical analysis of the three multicast routing algorithms mentioned above, on simple mesh-based network topologies, such as that shown in Figure 1. While not necessarily representative of realistic network scenarios, the mesh-based networks do provide two key advantages. First, they provide a network scenario in which optimal routing distances *can* be computed *a priori*, providing a baseline for evaluating the performance of other routing heuristics. Second, they provide a simple enough environment to interpret and fully understand the (mathematical and simulation) results of the performance analysis.

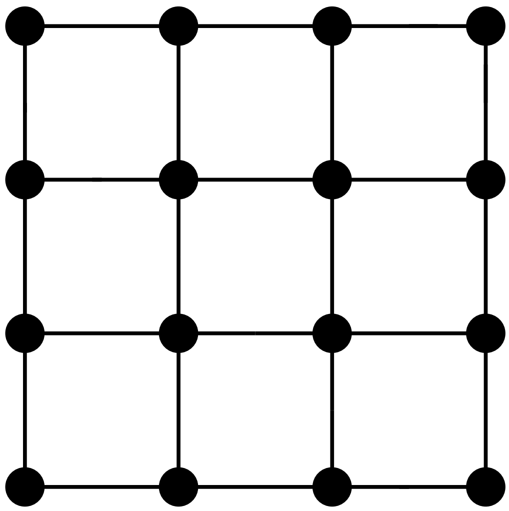

Figure 1: An Example of an N x N Mesh Network ($N = 4$)

The remainder of this paper is organized as follows. Section 2 describes the multicast routing algorithms evaluated in the paper. Section 3 presents a mathematical analysis of the multicast routing problem, first for a (trivial) one-dimensional network, and then for two-dimensional rectangular mesh networks (the case of interest). Section 4 presents a simulation experiment designed to evaluate the effectiveness of the multicast routing algorithms, and the accuracy of the analytical model. Finally, Section 5 concludes the paper.

2 Multicast Routing Models

2.1 Problem Statement and Assumptions

The purpose of a multicast routing algorithm is to find a routing path from a designated source node S in a network to a designated set of recipient nodes in the network. The performance objectives in multicast routing are to route the call as efficiently as possible (*i.e.,* using as few network links as possible), and to keep the call blocking probability as low as possible.

In our work, the set of multicast recipients is known at the time of call arrival, and remains static for the duration of the call. If all intended recipients can be reached successfully without exceeding the network capacity on any network link, then the call is accepted. Otherwise, the call is blocked (rejected), recorded as such, and removed from further consideration by the network. Note that accepting a call implies that the call must be routed through the network, which reduces the available network capacity on the chosen route for the duration of the call. The available capacity on the network links correspondingly increases whenever a call completes.

Note that the multicast routing problem is distinctly different from that of the *optimal flow assignment problem* in operations research. In our context, it does *not* make sense to use *backtracking* at all: the decision to accept or reject a call must be made (once and for all) at the time of call arrival. There is no backtracking allowed to change the acceptance decision or the route chosen for previous (existing) calls.

Defining optimality in the context of multicast routing is also difficult or impossible to do. For one thing, one must specify what performance metric is to be optimized. The two obvious ones are *call blocking probability* (which should be as low as possible) and *links used per successful call* (which should be as low as possible). However, these two metrics are not independent, and an improvement in one metric often comes at the expense of the other metric. For example, link usage can clearly be optimized by rejecting all calls that require a distance greater than 1 link. Or, as a more realistic example, one could reject all calls that, because of the current loading on the network, require sub-optimal routes relative to the known shortest path derived from the static network topology. Furthermore, one must always deal with a question like the following: "Should I reject the 'big' multicast call X arriving right now in the hopes that I can accommodate two (or more) 'small' (multicast or unicast) calls (say, Y and Z) that *might* arrive in the near future? (*i.e.,* in the time before X completes)" This decision obviously impacts the call blocking probability, and could affect link usage in unpredictable ways (resulting in an increase, a decrease, or no change).

In light of this discussion, we focus our attention only on *greedy* approaches to multicast routing: route the arriving call if a path exists to accommodate the call right now. Otherwise, reject the call. This paper considers three such approaches to multicast routing: M-Unicast, Conference Call, and Core-Based Trees.

2.2 M-Unicast Model

The M-Unicast routing algorithm achieves multicast transmission by using M separate unicast transmissions, one to each recipient. That is, the source makes M copies of each message, and sends one copy to each group recipient using unicast. An example of M-Unicast is electronic mail sent to a mailing list in the Internet: a separate copy is routed to each recipient on the distribution list, regardless of where in the network they reside.

M-Unicast is thus *not* a multicast routing algorithm at all, since it violates the goals of multicast that were outlined in Section 1. Nevertheless, it does provide a useful baseline for comparison: if a proposed multicast routing algorithm A provides performance results that are *worse* than M-Unicast, then clearly A is not a practical multicast routing algorithm.

2.3 Conference Call Model

The conference call routing model uses a specialized conference hub, statically located in the network, as a rendezvous point for the sender and the recipients. This model is used in the telephone system for (voice) conference calls. Participants must dial in to a pre-assigned number on a pre-assigned conference hub (bridge). For economic reasons, conference bridges are usually available only in selected (large) metropolitan areas. However, this fact means that sub-optimal routing can easily result when receivers are clustered elsewhere in the network.

In our version of the conference call model, the source of a multicast call sends *one* copy of a message to the conference hub. The hub node is responsible for replicating the message, and delivering one copy separately to each of the M recipients (using M-Unicast). In our simulator, the number of hubs can be scaled with network size (*e.g.,* 1 hub node in a small 4 x 4 mesh, and 4 equally spaced hub nodes in a large 10 x 10 mesh). In the experiments in this paper, only 1 hub is used, regardless of the network size. The location of the hub node is statically assigned, and only interior nodes of the mesh (never corner nodes or edge nodes) are eligible to be conference hubs.

2.4 Core-Based Trees Model

The Core-Based Trees (CBT) [1] model is similar to the conference call model in that it uses a hub (core) to effect the delivery of multicast messages. However, rather than using a static hub, the CBT model chooses the location of the hub *dynamically* for *each* multicast call, based on where the group members reside. As in the conference call model, the source sends one copy of each message to the hub (core), and the hub is responsible for replicating the message and sending a copy separately to each group member (using M-Unicast).

In our implementation, the location of the hub is determined by averaging the X and Y coordinates of all group participants (source and recipients) and rounding off to the nearest node in the mesh. With this approach, corner nodes in the mesh can never be chosen as hubs. Edge nodes can be chosen as a hub only if a large number of group participants lie along the same edge of the mesh. Thus the vast majority of hub nodes chosen will be interior nodes of the mesh.

2.5 Summary

This section has presented three simple routing techniques for handling multicast calls in mesh-based networks. Note that there is no "link sharing" in any of the routing models[3] described here. That is, we do *not* fully exploit the true potential of multicast as described in Section 1. With link sharing, multicast calls that use common links in routing a multicast group use only one unit of channel capacity for all recipients on those common links, regardless of how many recipients use that link. This "no link sharing" assumption is made to simplify the mathematical analysis presented in Section 3. A full set of performance results for multicast routing algorithms, including a study of the effects of link sharing, is available in [7].

3 Mathematical Analysis

This section presents a mathematical analysis of the multicast routing problem. In particular, we derive a closed form expression for the average (*i.e.,* expected value) for the distance between two randomly chosen distinct points in a regular mesh-based network. We begin with an analysis of a one-dimensional network (as a simple warmup exercise), and then proceed to the two-dimensional case of interest. Generalizing the result to higher dimensionality (*e.g.,* hypercube networks) is left as future work.

3.1 One-Dimensional Networks

Consider a one-dimensional network with N ($N > 1$) points, with all points unit distance from their neighbours (*e.g.,* the points represent consecutive integers on the number line). An example of such a network is shown in Figure 2. If two (distinct) points are chosen at random, then what is the expected value $E[D]$ for the distance D between the two points?

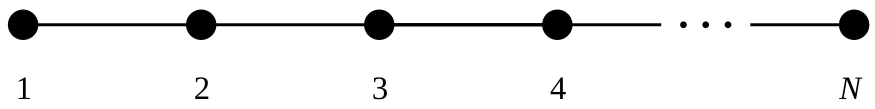

Figure 2: A One-Dimensional Network with N Points

The answer is $\frac{N+1}{3}$. From N points, there are $\binom{N}{2}$ ways to choose two points. Among these choices, there are $N-1$ ways to choose points that are only 1 unit apart (*i.e.,*

[3] The original CBT model [1] certainly uses link sharing, but we intentionally exclude that feature from our model so that we can make a fair comparison with the other two models.

$\{1,2\},\{2,3\},\{3,4\},\ldots\{N-1,N\}$), $N-2$ ways to choose points that are distance 2 apart, $N-k$ ways to choose points that are separated by distance k ($1 \le k < N$), and 1 way to choose points distance $N-1$ apart (*i.e.,* $\{1,N\}$). It is thus straightforward to compute:

$$E[D] = \frac{\sum_{i=1}^{N-1} i(N-i)}{\binom{N}{2}} = \frac{N\sum_{i=1}^{N-1} i - \sum_{i=1}^{N-1} i^2}{\binom{N}{2}}$$

$$= \frac{\frac{N^2(N-1)}{2} - \frac{N(N-1)(2N-1)}{6}}{\binom{N}{2}} = \frac{3N^2(N-1) - N(N-1)(2N-1)}{6\binom{N}{2}}$$

$$= \frac{3N^2 - 2N^2 + N}{3N} = \frac{N+1}{3} \tag{1}$$

We can derive the same result in a more long-winded fashion, the purpose of which will become clearer in Section 3.2. Our analysis makes use of a counting function $C(i,N)$, which expresses the total number of links that must be traversed to reach all other nodes in the network, using M-Unicast, starting from node i. Once $C(i,N)$ is known (for a given start node i), then the average distance from node i to all other possible recipients (of which there are $N-1$) is obtainable as

$$D(i,N) = \frac{C(i,N)}{N-1} \tag{2}$$

The overall average distance for two randomly chosen points in the network is then

$$E[D] = \frac{\sum_{i=1}^{N} D(i,N)}{N} = \frac{\sum_{i=1}^{N} C(i,N)}{N(N-1)} \tag{3}$$

The calculation of $C(i,N)$ proceeds as follows. First consider the case $i=1$. Thus the starting point is the left endpoint of the one-dimensional mesh. From this endpoint, it will take one link traversal to reach node 2, two link traversals to reach node 3 (starting from node 1, recall), and $N-1$ links to reach node N. Thus:

$$C(1,N) = \sum_{j=2}^{N}(j-1) = \sum_{j=1}^{N-1} j = \frac{N(N-1)}{2} \tag{4}$$

The same result obviously applies for the other endpoint of the mesh ($i=N$). It is nice to note that Equation 4 also works for a single node network ($N=1$), which has $C(1,1)=0$.

Now consider the case $1 < i < N$. In this case, node i divides the one-dimensional mesh into two separate pieces. There are $i-1$ nodes to the left of i, and $N-i$ nodes to the right of i. We need to consider *each* of these submeshes (with node i added in as the endpoint of the mesh of course) in turn, and simply sum the result. Thus:

$$C(i,N) = C(1,i) + C(1,N-i+1) \qquad (1 < i < N)$$

$$= \frac{i(i-1)}{2} + \frac{(N-i+1)(N-i)}{2}$$
$$= \frac{i^2 - i + N^2 - Ni + N - Ni + i^2 - i}{2}$$
$$= \frac{N^2 + N - 2Ni + 2i^2 - 2i}{2}$$
$$= \frac{N(N+1)}{2} - i(N-i+1) \quad (5)$$

For $i = 1$ or $i = N$, this result reduces to $\frac{N^2+N}{2} - N = \frac{N^2-N}{2} = \frac{N(N-1)}{2}$, as expected.

Summing Equation 5 for every possible value of i ($1 \le i \le N$) and applying Equation 3 yields the result $E[D] = \frac{N+1}{3}$.

3.2 Two-Dimensional Networks

Consider a two-dimensional rectangular mesh with r rows ($r > 1$) and c columns ($c > 1$), such as that shown in Figure 3. (An N x N mesh is simply a special case where $r = c = N$.) Assuming again that the nodes are all unit distance from their neighbours, (*i.e.,* the points reside at the integer coordinates in a Cartesian plane), then what is the average distance between two randomly chosen (distinct) points within the mesh?

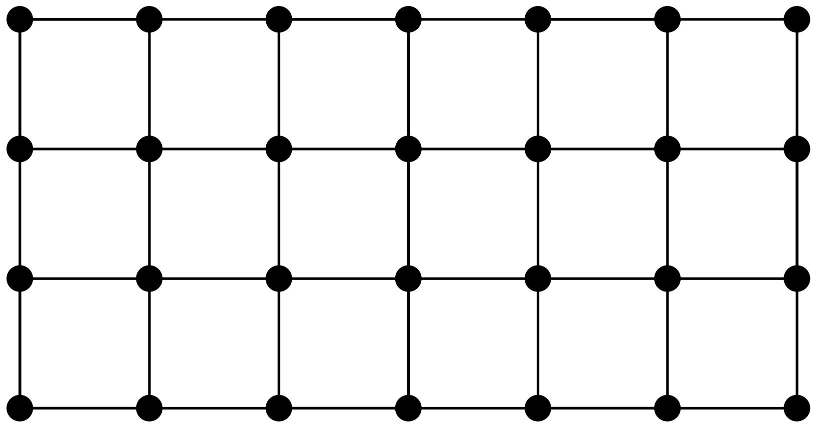

Figure 3: A Two-Dimensional Network with $r = 4$ Rows and $c = 7$ Columns

At first glance, it appears that the answer can be obtained by simply doubling the result from Section 3.1. That is, the result should be $E[D] = \frac{2(N+1)}{3}$.

However, this approach would result in only an approximate solution (and a slight over-estimate of the expected distance). The reason lies in the fact that Section 3.1 assumes the nodes are distinct in that dimension, and here in Section 3.2 we do not require that nodes be distinct in *both* dimensions. It is sufficient for the two chosen points to differ in *at least one* dimension (either the row coordinate, or the column coordinate, or both). This observation complicates the analysis a bit, since we desire an exact closed form solution. Furthermore, we would like our result to apply for general r and c, and not just $r = c = N$.

Our analysis for the two-dimensional case makes use of recurrence relations, and the same analysis idea that was introduced in Section 3.1. Suppose for the moment that we have chosen a source node at position (i, j) in the mesh (row i, column j).

First, consider the case where the source node is a corner node of the mesh. By symmetry we need only consider the case $i = 1$, $j = 1$. In this case, we can use the counting formula[4] idea $C(i, N)$ from Section 3.1 to determine the number on links involved in doing M-Unicast from the source node to: (1) the $c - 1$ remaining nodes in the first row of the mesh; (2) the $r - 1$ remaining nodes in the first column of the mesh; and (3) the $(r - 1)(c - 1)$ remaining nodes in the rest of the mesh (other than the first row or column). Part (1) is simply $C(1, c) = c(c - 1)/2$, and part (2) is simply $C(1, r) = r(r - 1)/2$.

Part (3) is the only remaining unknown, and it can be solved recursively: it takes 2 links to get from node (1,1) to node (2,2), and then node (2,2) can be treated as a corner point of another (smaller) $\hat{r}$ x $\hat{c}$ mesh, where $\hat{r} = r - 1$ and $\hat{c} = c - 1$. The recursion continues until you have either a one-dimensional mesh remaining ($r \neq c$ initially), which is directly solvable (from Section 3.1), or a single point remaining ($r = c$ initially), which has $C(1, 1) = 0$.

The full recurrence relation for a corner node on an r x c mesh is thus

$$C(1,1,r,c) = \begin{cases} \frac{c(c-1)}{2} & \text{if } r = 1 \\ \frac{r(r-1)}{2} & \text{if } c = 1 \\ C(1,1,r,1) + C(1,1,1,c) + 2(r-1)(c-1) + C(1,1,r-1,c-1) & \text{otherwise} \end{cases} \tag{6}$$

Solving this recurrence relation yields:

$$C(1,1,r,c) = \frac{rc(r+c-2)}{2} \tag{7}$$

It is comforting to note that for $r = 1$, this equation reduces to $\frac{c(c-1)}{2}$, the same as mentioned in Section 3.1. A similar observation applies for $c = 1$. Also, for $r = c = N$, we obtain $C(1, 1, N, N) = N^2(N - 1)$ which can be shown by induction for square meshes.

We now turn our attention to the average internode distance for a source node (i, j), for the general case where $1 < i < r$ and $1 < j < c$. Note that, under this assumption, the source node (i, j) is an interior node of the mesh, and thus (i, j) partitions the entire mesh into four separate quadrants, as shown in Figure 4. Even more delightful is the fact that node (i, j) is now a corner node of each submesh. It is thus possible to apply the result from

[4] In the ensuing discussion, we will generalize the counting formula to two dimensions, using the form $C(i, j, r, c)$. For absolute clarity, one should use $C_1(i, N)$ and $C_2(i, j, r, c)$ to distinguish the one-dimensional case from the two-dimensional case. However, to avoid cluttering our notation, we drop the subscripts, relying on the context (and the number of parameters) to indicate which version of $C()$ is intended.

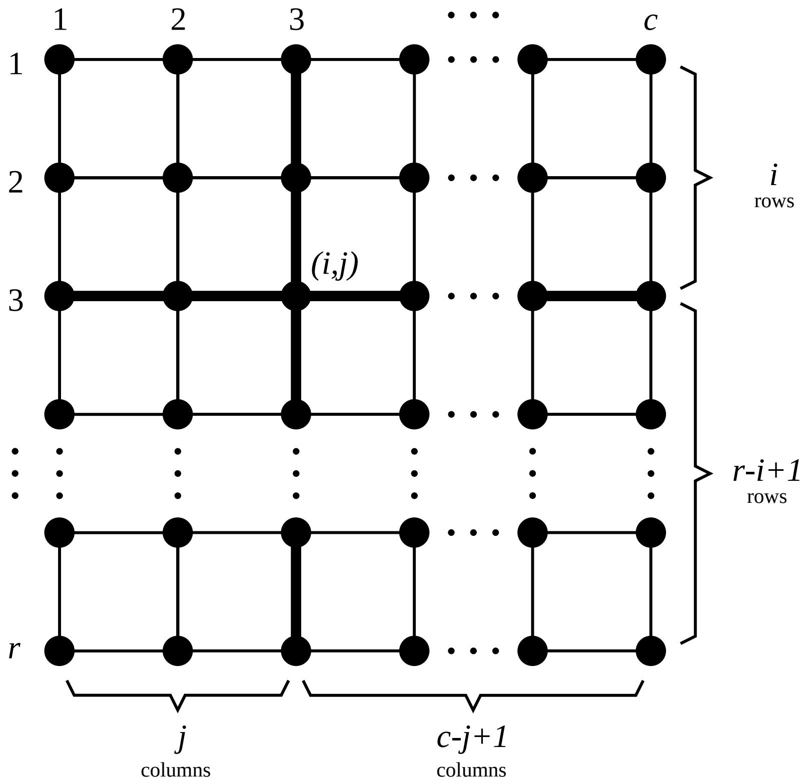

Figure 4: A Mesh Partitioned into 4 Submeshes by Node (i,j)

the previous section to each of these four quadrants, which may be different in size. The exact formula for $1 < i < r$ and $1 < j < c$ is:

$$C(i,j,r,c) = C(1,1,i,j) + C(1,1,i,c-j+1) + C(1,1,r-i+1,j) \\ + C(1,1,r-i+1,c-j+1) - C(i,1,r,1) - C(1,j,1,c) \tag{8}$$

The latter two (subtracted) terms are necessary to eliminate the double counting that happens for nodes in row i and column j (*i.e.,* the thick lines in Figure 4).

There is one final case to clean up: when the node (i,j) lies along an edge of the mesh (but not a corner). By symmetry, we can assume that the source lies along the topmost edge of the mesh ($i=1$). In this case, the mesh is partitioned into only two submeshes, with the source at a corner of each submesh. The formula[5] becomes:

$$C(i,j,r,c) = C(1,1,r,j) + C(1,1,r,c-j+1) - C(1,1,r,1) \tag{9}$$

Again, the latter term is necessary to eliminate the double counting that happens for nodes in column j.

[5]An analogous formula obviously applies for the case $j=1$.

Putting these three results all together, we have our final recurrence relation for the count function for node (i, j) in an r x c mesh:

$$C(i,j,r,c) = \begin{cases} \frac{rc(r+c-2)}{2} & \text{if } i = 1 \text{ and } j = 1 \\ C(1,1,r,c) & \text{if } i = 1 \text{ and } j = c \\ C(1,1,r,c) & \text{if } i = r \text{ and } j = 1 \\ C(1,1,r,c) & \text{if } i = r \text{ and } j = c \\ C(1,1,1,j) + C(1,1,1,c-j+1) & \text{if } r = 1 \\ C(1,1,i,1) + C(1,1,r-i+1,1) & \text{if } c = 1 \\ C(1,1,r,j) + C(1,1,r,c-j+1) - C(1,1,r,1) & \text{if } i = 1 \text{ or } i = r \\ C(1,1,i,c) + C(1,1,r-i+1,c) - C(1,1,1,c) & \text{if } j = 1 \text{ or } j = c \\ C(1,1,i,j) + C(1,1,i,c-j+1) & \\ +C(1,1,r-i+1,j) + C(1,1,r-i+1,c-j+1) & \\ -C(i,1,r,1) - C(1,j,1,c) & \text{otherwise} \end{cases} \tag{10}$$

After much algebraic manipulation, this recurrence relation reduces to:

$$C(i,j,r,c) = \frac{rc(r+c-2(i+j-1))}{2} + rj(j-1) + ci(i-1) \tag{11}$$

Note the equivalent role played by r and c in this formula (as one would expect). It is also comforting to note that for $i = 1$ and $j = 1$, the formula reduces to Equation 7, since the last two terms of Equation 11 disappear.

Equation 11, as it turns out, will be the most important one in our evaluation of multicast routing algorithms in Section 4, since once the source node is known (for the M-Unicast algorithm), or the hub node is known (for the Conference Call model and the Core-Based Trees model), Equation 11 tells you the expected link cost of routing to each recipient in a multicast group.

From Equation 11 it is a simple matter to compute the average internode distance for source node (i, j) in an r x c mesh:

$$D(i,j,r,c) = \frac{C(i,j,r,c)}{rc-1} \tag{12}$$

The average distance between two randomly chosen distinct nodes in the whole network can be obtained by considering each possible source node (i, j):

$$E[D] = \frac{\sum_{i=1}^{r}\sum_{j=1}^{c} D(i,j,r,c)}{rc} \tag{13}$$

In the general case $(r \neq c)$, the formula evaluates to $E[D] = \frac{r+c}{3}$. For $r = c = N$, the formula predicts $E[D] = \frac{2N}{3}$.

3.3 Analytic Results

This section relates the mathematical analysis of the previous section to the three multicast routing algorithms considered in this paper, namely M-Unicast, Conference Call, and Core-Based Trees. The performance analysis in the remainder of the paper assumes $r = c = N$. The analysis in this section also assumes a network with infinite link capacity, so that no call blocking (or suboptimal routing) ever occurs. That is, all calls are successful, and all calls always use a shortest path route.

The analysis in Section 3.2 is exact for the M-Unicast routing algorithm. Thus the average number of links used per multicast call will be a strictly linear function of the group size (since recipients are assumed to be chosen uniformly at random, independently). The slope of this line will be $\frac{r+c}{3} = \frac{2N}{3}$, which depends only on the network size N. The intuition here is simple: the larger the mesh size, the further apart the recipients are (on average), and the greater the incremental cost of routing to a multicast recipient.

An exact analysis of the Conference Call model requires knowing the location of the conference hub(s) in the mesh. With this information, Equation 11 can be used to determine the (expected) cost of routing a multicast call: the source has to reach the hub, and the hub has to reach all of the recipients. This cost function will again be linear in the number of recipients, but may have a slightly higher starting point (because of the cost of connecting the source to the hub), and a lower slope than the M-Unicast algorithm (since a well placed hub is in the interior of the network, and never at an edge or corner node). For example, for 1 hub at location (2,2) in a 4 x 4 network, we expect a slope of 2.13 for the Conference Call model (instead of $\frac{2N}{3} = 2.67$ for M-Unicast), and for 1 hub at location (5,5) in a 10 x 10 network, we expect a slope of 5.05 instead of 6.67.

An exact mathematical analysis of the Core-Based Trees model is not possible, since the hub location is chosen dynamically for each group, and it makes little sense to calculate the "average" core location. Nevertheless, we do still expect the links used per call to be linear in the number of group recipients, and we can place a fairly tight upper bound on the expected slope of the line. For example, the CBT model should do about as well as (but slightly better than) having the core node always fixed at an innermost interior node in the network.

3.4 Summary

This section has presented a mathematical analysis of three simple multicast routing strategies for mesh-based networks. The approaches differ in the presence or absence of a conference hub, and in whether the hub location is static or dynamic. Mathematical analysis (and intuition) suggests that the M-Unicast model will provide the worst performance, the Core-Based Trees model will provide the best performance, and the Conference Call model will provide an intermediate level of performance.

There is one key assumption to bear in mind in the analytic results presented here: they all assume optimal shortest path routing for all recipients in all calls, as based on the static global network topology. In other words, network links have infinite capacity: no calls are ever blocked, and no links ever saturate, causing non-optimal "detour" routes to be chosen. This assumption does not necessarily hold in the simulation results for the same routing models, presented in the next section.

4 Experimental Method and Results

4.1 Simulation Methodology

A set of simulation experiments was conducted to evaluate the performance of the three multicast routing algorithms described in Section 2. The purpose of these experiments was to assess the accuracy of the analytic results derived in Section 3.

The experiments were conducted using *call level simulation* on simple mesh topologies of size 4 x 4, 6 x 6, 8 x 8, and 10 x 10. A separate workload file was constructed for each mesh size, containing 100,000 synthetically generated calls. Workload parameters included the call arrival rate, the mean call duration, the mesh size, the mean multicast group size, and the percentage of the total calls that are to be multicast calls (the remainder are unicast calls). Call arrivals are assumed to be Poisson, and call durations are exponentially distributed. The number of recipients for each multicast call is drawn from a geometric distribution (a modeling assumption) with the specified mean. The experiments reported here use a mean group size of 4 recipients, and 10% multicast traffic. For each mesh size, the exact same workload file is run on each multicast routing algorithm, so that performance differences in the results obtained are due to differences in the algorithms, and not differences in the workload.

In our simulation setup, we can intentionally scale the workload to make the call blocking probability as low or as high as desired. In the experiments reported here, call blocking is intentionally low ($\leq$ 1.0%), so that link usage (*i.e.,* average number of links used for each successfully routed call) is the only performance metric of interest. A more complete description of simulation experiments, call blocking behaviour, and other performance results is available in [7].

4.2 Simulation Results

Figures 5, 6, and 7 summarize the results of our simulation experiments. Figure 5 presents link usage results (both analytic and simulation) for the M-Unicast routing model, for the four different mesh sizes considered. Figures 6 and 7 do the same for the Conference Call and Core-Based Trees models, respectively.

M-Unicast Model

Figure 5 shows that the average number of links used per successful call in the M-Unicast model is strongly linear in the number of group recipients. The simulation results (plotted with mean and 95% confidence intervals[6]) for each multicast group size agree very closely with the analytic predictions. The values predicted by the analytic model are always well within the simulation confidence intervals for groups of less than 10 recipients.

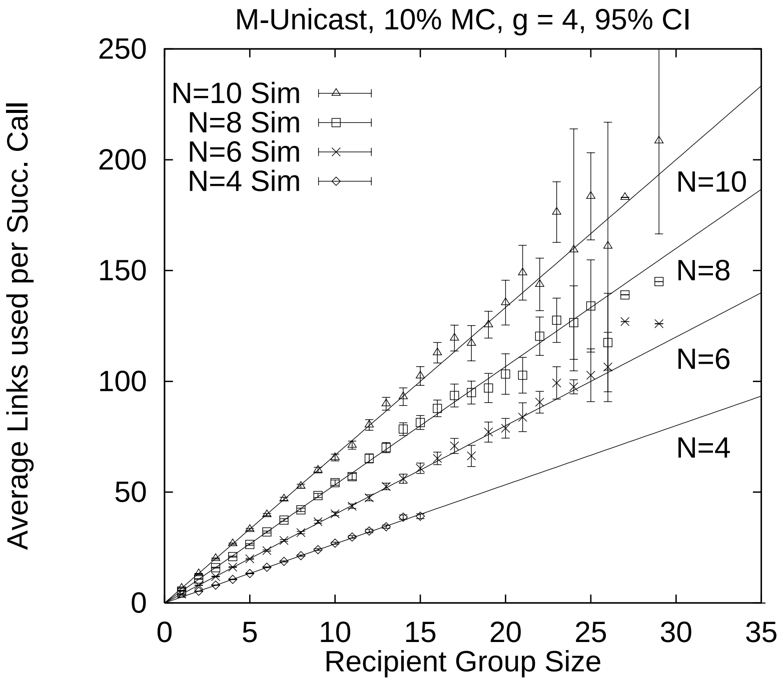

Figure 5: Link Usage Results for M-Unicast Model. Data points show the mean and 95% confidence intervals for simulation results. Solid lines denote analytically derived values.

Table 1 further confirms these observations. A linear regression fit of the simulation data yields a very high R^2 value (0.98-0.99). The regression shows a slope close to $\frac{2N}{3}$ for $N = \{4, 6, 8\}$. The slope value for the $N = 10$ case is distorted somewhat by the link usage results for several large multicast groups, as evidenced by a relatively large (negative) y-intercept value. For a given recipient group size, larger meshes generally show higher variability in the link usage results because of the greater range of possibilities for the physical placement of those group members.

[6]The confidence intervals naturally get larger for large group sizes because our workload (intentionally) generates fewer calls with a large number of recipients.

Table 1: Linear Regression Results for M-Unicast Model

Mesh Size	Slope	y-intercept	R^2
4 x 4	2.65	0.20	0.993
6 x 6	3.97	0.21	0.977
8 x 8	5.37	0.08	0.984
10 x 10	7.24	-4.29	0.981

Conference Call Model

Figure 6 shows the results for the Conference Call model. Again, the line is strongly linear (see Table 2), and again the analytic results match very well with the simulation data.

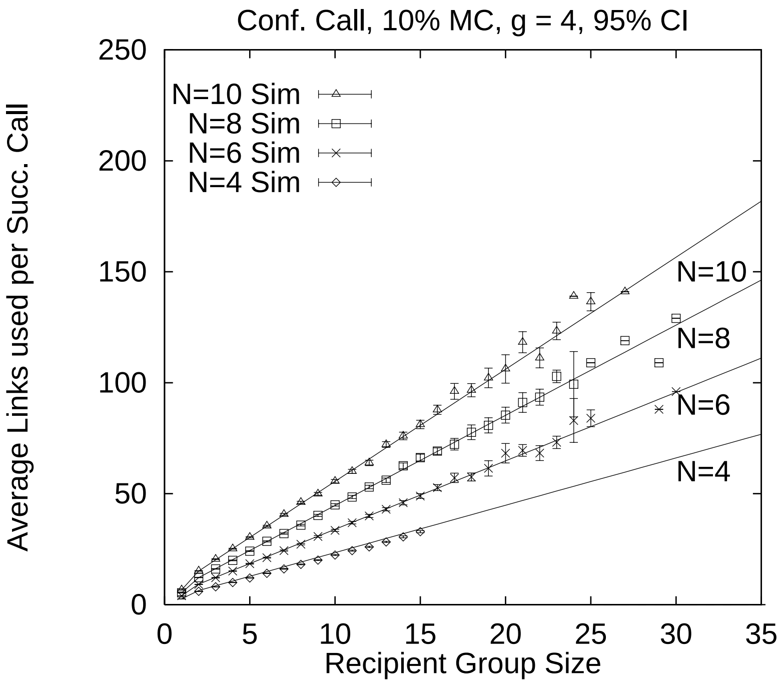

Figure 6: Link Usage Results for Conference Call Model. Data points show the mean and 95% confidence intervals for simulation results. Solid lines denote analytically derived values.

Table 2: Linear Regression Results for Conference Call Model

Mesh Size	Slope	y-intercept	R^2
4 x 4	2.08	1.52	0.998
6 x 6	3.33	0.97	0.986
8 x 8	4.05	4.11	0.980
10 x 10	4.97	5.86	0.990

There are several other interesting observations to note about Figure 6. First, there is a slight "kink" in each line between the data points for 1 and 2 recipients. This kink, which was not present in the M-Unicast algorithm, is evident for each mesh size, and for both the simulation and analytical results. This kink effect arises in the Conference Call model because calls with 2 or more recipients are multicast calls (which must route from source to hub, and then from hub to recipients), whereas calls with 1 recipient are unicast calls (which need not route through the hub). Thus the change in slope represents the overhead of source to hub routing. This overhead (generally) increases in larger meshes (see the y-intercept values in Table 2).

A second observation is that the slopes of the lines in Figure 6 are significantly lower than those in Figure 5 (see the regression values in Tables 1 and 2). The slopes in Table 2 are 16-25% lower than the corresponding (analytically derived) slopes for the M-Unicast model, indicating that the Conference Call model, despite its additional overhead for source to hub routing (note the larger y-intercept value in Table 2 compared to Table 1), is more efficient in its routing of multicast calls.

Finally, it is interesting to note that the 95% confidence intervals of the simulated link usage values for the Conference Call model always overlap with the analytic values, except for the $N = 4$ mesh. Here, the simulation values are consistently below the analytical values (but still within 10%). The explanation for this anomaly is that the analytic model assumes that the source is distinct from the hub, and that the hub is distinct from each recipient. In the simulation, it is possible for the source to coincide with the hub (with a probability of 1/16 on a 4 x 4 mesh, and 1/100 on a 10 x 10 mesh), or for the hub to coincide with one of the recipients (a probability that depends on the group size and the mesh size). In these cases, the number of links required (in the simulation) to add that source or recipient to the multicast delivery tree is zero, while the analytic model estimates the link cost using $E[D]$ for each participant. Thus the analytic model overestimates the link cost slightly in these cases. This "error", which is most evident for small multicast groups and small mesh sizes, becomes insignificant for large groups and large meshes.

Core-Based Trees Model

Figure 7 shows the results for the Core-Based Trees model. All lines are strongly linear (see Table 3), and all simulation results match quite closely with the analytical model. Here, however, recall that the analytic prediction is only an upper bound for each line representing simulation data.

As expected, the CBT algorithm performs better than the analytic bound, especially for small multicast groups, where dynamic core placement can lead to significantly better routes than using a static hub. The simulation results approach the analytic bound for large multicast groups since the (expected) location of the dynamic core will be in the middle of the mesh (close to the location of the static hub assumed in the analytic bound). The analytic upper bound is tightest on small meshes since there is little flexibility in dynamic

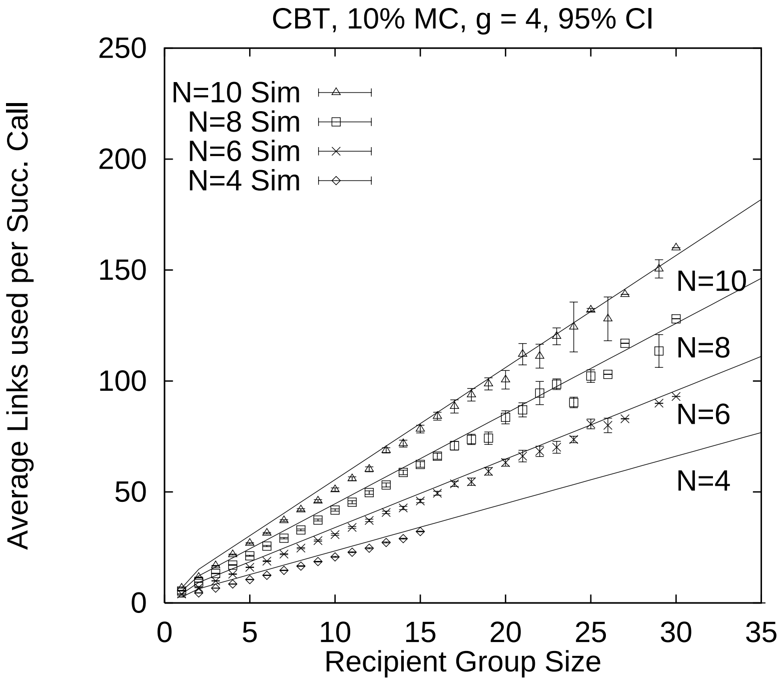

Figure 7: Link Usage Results for Core-Based Trees Model. Data points show the mean and 95% confidence intervals for simulation results. Solid lines denote analytically derived values.

Table 3: Linear Regression Results for Core-Based Trees Model

Mesh Size	Slope	y-intercept	R^2
4 x 4	2.07	0.22	0.998
6 x 6	3.09	0.35	0.994
8 x 8	4.11	0.75	0.995
10 x 10	4.94	2.55	0.986

core placement.

The regression analysis in Table 3 confirms the strong linearity in Figure 7, and shows the average slopes observed. While the slopes that result for the Core-Based Trees model are almost identical to those of the Conference Call model (compare Table 2 and Table 3), the Core-Based Trees model has lower y-intercept values, reflecting the lower cost of source-to-hub (core) routing when dynamic core placement is used. Thus the Core-Based Trees model consistently outperforms the M-Unicast and Conference Call routing models, using these performance metrics.

Effect of Network Load

Finally, we conducted a simulation experiment to determine how rapidly the predictive ability of the analytic model degrades when the network gets congested.[7] We conducted this experiment by artificially scaling the workload[8] to create call blocking probabilities ranging from 0% to 5%. In the latter case, it is known that many calls are blocked. Thus, many links must be saturated, and non-optimal routing must occur. This experiment was conducted only for the M-Unicast model (for which the analytic model is known to be exact when the network has infinite capacity), and only on the 4 x 4 mesh.

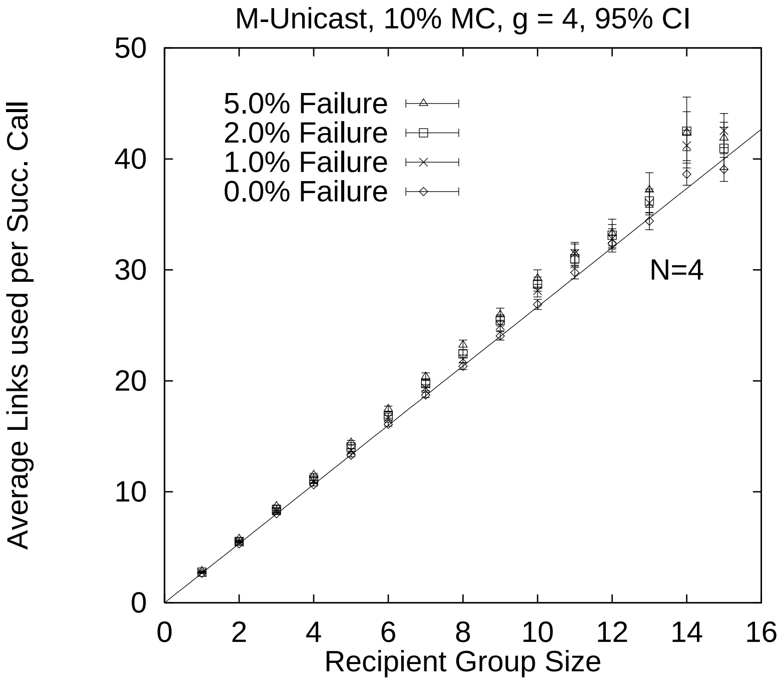

Figure 8: Effect of Network Load on Model Accuracy. Data points show the mean and 95% confidence intervals for simulation results. The solid line shows analytically derived values.

Performance results for this experiment are shown in Figure 8. The analytic model is exact (or close to exact) when there is no call blocking, as expected. The link usage results for the simulations begin to diverge further and further from the analytic predictions as the call blocking probability increases. However, the analytic model is still within 10% of the simulation results for all cases except recipient group sizes of 14. The small number of samples at this group size resulted in an average link usage 15% larger than the analytical value, when the call blocking is 5.0%.

[7]Recall that the analytic model always assumes perfect routing in an infinite capacity network, while real networks end up routing around hot spots, and blocking calls when congestion occurs.

[8]This scaling was done by simply scaling the call durations. All other workload characteristics in the workload file (*e.g.,* call arrival time, source node, group size, recipients) remained the same.

Regression analysis (Table 4) shows that the slope of the link usage plot changes only marginally as the workload increases. Thus the analytic model is deemed a useful predictor of the link cost for multicast calls under reasonable levels of network load.

Table 4: Linear Regression Results for Load Effects (M-Unicast Model)

Blocking	Slope	y-intercept	R^2
0.0%	2.68	-0.01	0.999
1.0%	2.86	-0.47	0.998
2.0%	2.84	-0.17	0.995
5.0%	2.87	0.12	0.997

4.3 Summary

This section has presented a simulation study of three simple multicast routing strategies for mesh-based networks. The simulation results are found to match very well with the analytic results predicted in Section 3.3. The analytic results are always within 10% of the simulation results, and diverge from the simulation results only when the network workload, and thus the call blocking probability, is artificially high.

5 Conclusions

Multicast traffic presents a non-trivial routing problem for point-to-point computer networks. Because the optimal multicast routing problem is NP-complete, in the general case, heuristics must be used for multicast routing in today's computer networks.

This paper presents a mathematical analysis of the multicast routing problem on simple mesh-based network topologies. Closed form solutions are obtained to measure the cost of multicast routing, in terms of link usage for successfully routed calls.

The analytic results are used in conjunction with simulation to evaluate three possible multicast routing techniques: M-Unicast, Conference Call, and Core-Based Trees. Among the three routing algorithms evaluated, the Core-Based Trees model provides the most efficient multicast routing. Throughout the performance analysis, the analytic results and the simulation results are found to agree very closely (0-10%). Simulation results diverge from the analytic results only when significant network congestion (*i.e.,* call blocking) occurs.

References

[1] T. Ballardie, P. Francis, and J. Crowcroft, "Core-Based Trees (CBT): An Architecture for Scalable Inter-Domain Multicast Routing", *Proceedings of 1993 ACM SIGCOMM Conference on Communication Architectures, Protocols, and Applications*, San Francisco, CA, pp. 85-95, September 1993.

[2] S. Casner and S. Deering, "First IETF Internet Audiocast", *ACM Computer Communications Review*, Vol. 22, No. 3, pp. 92-97, July 1992.

[3] S. Deering, "Host Extensions for IP Multicasting", RFC 1112, August 1989.

[4] S. Deering, *Multicast Routing in a Datagram Internetwork*, Ph.D. thesis, Stanford University, 1991.

[5] S. Deering, D. Estrin, D. Farinicci, V. Jacobson, C. Liu, and L. Wei, "An Architecture for Wide-Area Multicast Routing", *Proceedings of 1994 ACM SIGCOMM Conference on Communication Architectures, Protocols, and Applications*, London, UK, pp. 126-135, August 1994.

[6] H. Eriksson, "MBONE: The Multicast Backbone", *Communications of the ACM*, Vol. 37, No. 8, pp. 54-60, August 1994.

[7] T. Harrison, *Performance Evaluation of Routing Algorithms for Multicast Traffic*, M.Sc. thesis, Department of Computer Science, University of Saskatchewan, August 1995. (Available by anonymous ftp to ftp.cs.usask.ca as pub/discus/thesis-harrison.ps.Z)

[8] C. Hedrick, "Routing Information Protocol", RFC 1058, June 1988.

[9] F. Hwang and D. Richards, "Steiner Tree Problems", Technical Report, Bell Laboratories, *c.* 1987.

[10] R. Jain, *The Art of Computer Systems Performance Analysis*, John Wiley & Sons, Incorporated, New York, 1991.

[11] V. Kompella, J. Pasquale, and G. Polyzos, "Multicast Routing for Multimedia Communications", *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3, pp. 286-292, June 1993.

[12] J. Moy, "Multicast Routing Extensions for OSPF", *Communications of the ACM*, Vol. 37, No. 8, pp. 61-66, August 1994.

[13] D. Verma and P. Gopal, "Routing Reserved Bandwidth Multi-point Connections", *Proceedings of 1993 ACM SIGCOMM Conference on Communication Architectures, Protocols, and Applications*, San Francisco, CA, pp. 96-105, September 1993.

[14] D. Waitzman, C. Partridge, and S. Deering, "Distance Vector Multicast Routing Protocol", RFC 1075, November 1988.