

DUALCAST: A Scheme for Reliable Multicasting

Sudhir Aggarwal and Amritansh Raghav

Department of Computer Science

State University of New York at Binghamton, NY 13902-6000

{sudhir , raghav}@cs.binghamton.edu

Abstract

With the rapid deployment of high speed networks and the increased use of applications that blend voice, data and video traffic, protocols have to be looked at anew. The problem of trying to conserve bandwidth is being replaced by the problems of overcoming latency and processing bottlenecks. In this paper, we introduce the conflicting demands in implementing reliable multicast and then present a new heuristic that balances the tradeoffs by using redundancy while not loading the network excessively

Keywords: Reliable Multicasting, Steiner Trees, Redundancy Routing, Fault Tolerant Routing

1. Introduction

With the current trends in communication networks, and the services that are likely to be supported, it seems almost certain that newer networks will need to offer different types of service. For such services, new protocols will need to be developed to facilitate connections providing broadcasting, multicasting (one to many) and multipoint multicasting (many to many). Issues of reliability will also need to be looked at anew.

In the context of traditional data networks, reliability means that a data packet reaches its destination correctly. In a large public switched telecom network, on the other hand, reliability implies that a grade of service is maintained for a particular stream, and that a circuit remain established for the duration required. Multimedia applications, which blend data, voice and video streams, for example, require a large degree of synchronization and can have severe timing constraints. To satisfy the delay constraints such applications run under, protocol performance will have to be enhanced. Much of the previous research on protocols for multicasting has been designed to take advantage of the low error rates on LANs and WANs. However, under noisy conditions, or should the application be under stringent time and reliability constraints, the performance of these protocols suffers.

One way to provide reliability in a network is to build redundancy in the transmission paths. Dispersity routing, in which the message to be sent is broken up into segments which are routed via geographically different paths, has the advantage of distributing the load on the network and thus reducing the queueing delays at the nodes [1]. The messages can be coded in such a format that a missing segment can be regenerated. In redundant dispersity routing, the segments are duplicated and sent via different routes. While this increases the total traffic put on the network, random transmission errors and link failures do not have as great an impact on the throughput, and the robustness of the system is increased. In older telecommunication networks, limited bandwidth made fault tolerance through multiple transmission paths a prohibitively expensive option. However, with fiber becoming the medium of choice, the need to pack greater and greater information onto a limited bandwidth channel is no longer a necessity and reliable protocols that make effective use of the greater bandwidth are now practical.

The high speed and enormous bandwidth available in these connections however, also result in latency problems [2]. As propagation delays become comparable to transmission times, and the high bandwidth implies that a large volume of data is actually buffered on the links, it becomes costlier to retransmit erroneous packets. Because a channel is latency limited rather than transmitter speed limited, the greater the amount of control information that must be passed between the source and the destination, the more wasteful retransmitting erroneous packets becomes. It has also been seen that the bottleneck now lies in the processing of the protocol which has generated research in the area of lightweight protocols [3]. If an increase in the cost of the hardware is acceptable, then a way to deal with this bottleneck is the parallelization of the protocol processing [4]. The DUALCAST scheme we introduce in this paper will be seen to be inherently parallel.

Handling a multicast as a set of independent unicast connections places a larger load on the system than necessary. To minimize the traffic generated by the connections, a mul-

unicast is implemented by using a minimum weight tree that spans the set of multicasting destinations. With multicast copy switches becoming a reality this means that if a link is common to two destinations, then it need carry only a single packet of data till the bifurcation point. Hence a more efficient multicast attempts to reduce the number of links used in the connection, and thus the number of packets generated. However, if redundancy is used to incorporate fault tolerance, the number of packets transmitted is increased. Clearly, reliable multicasting exerts conflicting demands.

The concept of using multiple paths over which to transmit data has been studied before and algorithms exist for finding pairs of disjoint paths between two nodes in a network [5] [6] [7]. We adapt redundant dispersity routing to the problem of multicasting. In the scheme we propose, we take advantage of the large amount of bandwidth available and send two copies of each packet to each destination. This is done by generating two Steiner multicasting trees which span the set of destination nodes. The approximate Steiner trees are sufficiently diverse so that the duplicate transmission offers reliability, but not so sub optimal that the network is loaded excessively or that timing constraints cannot be met.

2. The Steiner Tree Problem

The general problem of determining the minimum weight tree that spans a given set of nodes M in a graph G is known as the Steiner tree problem. Although for generalized random graphs the Steiner problem can be shown to be NP-complete [8], heuristics exist that aim at an approximate solution in polynomial time [9] [10]. For a discussion of the Steiner problem, see [11].

3. The Algorithm

As was stated in the introduction, our aim is to find two approximate Steiner trees which satisfy certain conditions of disjointness and optimality. These Steiner trees can be generated using any polynomial time heuristic. In trial runs we used the Floyd Warshall algorithm to generate the minimum cost paths in the graph and then the Takahashi Matsuyama heuristic [10], which is similar to standard greedy algorithms, to find the approximate Steiner trees.

Let the network be depicted by the undirected, finite connected graph G consisting of a set of n vertices $V = v_1 \dots v_n$ and m edges $E = e_1 \dots e_m$. With each of these edges is associated a cost $C(e)$.

Of these vertices let one be a distinguished vertex which is the source from which the multicast transmission is to take place.

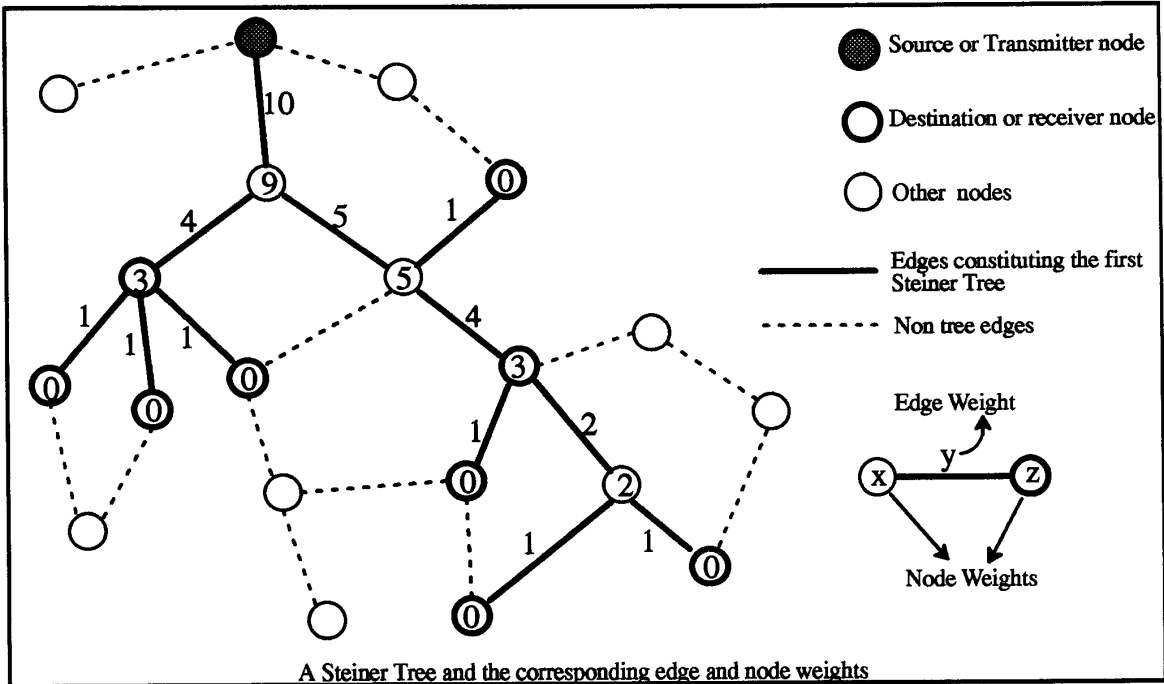
Then this heuristic aims to find two trees S_1 and S_2 (both S_1 and $S_2 \subset G$) which span the k multicasting nodes in M .

STEP I The minimum cost from every node to every other node is calculated using the Floyd Warshall algorithm on the cost of the edges

STEP II Then an approximate Steiner tree for the k multicasting nodes is constructed using the Takahashi Matsuyama heuristic. This is the first Steiner tree S_1 .

STEP III The graph G is transformed into another graph G' . To aid in this transformation a NODE WEIGHT and an EDGE WEIGHT is given to all the nodes and edges that form the Steiner tree. These weights can be defined recursively, starting from the root node:

- The node weight of a node is the sum of the edge weights of all outgoing edges from that node. The node weight of all leaves is 0. (We use the term *outgoing edges* even though the graph is undirected because due to the fact that the transmission is taking place from the root, tree itself is directed. Thus in this regard, an outgoing edge from a node is an edge on which a node will transmit information to the destinations.)
- The edge weight of an edge is the node weight of the node it is incident upon *if the node is not a multicasting destination*. In case the node is an intended receiver of the transmission and not just a relay or switching node, then the weight of the edge on the incident node is one greater than the node weight of the node. Again the reason for using the term *incident* in an undirected graph is the same as the one given above. We also wish to clarify here that non-leaf nodes may also be multicasting destinations



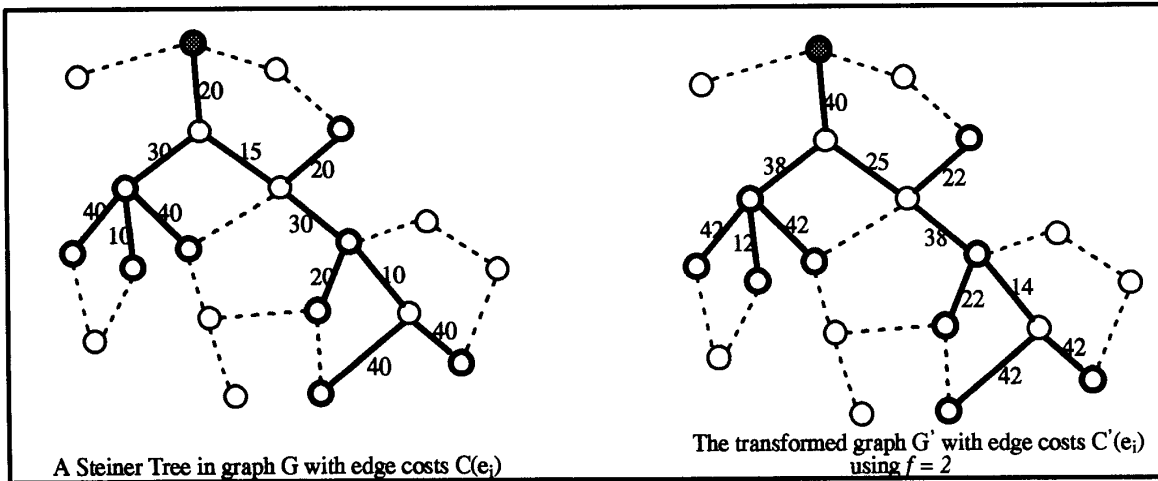
Once the edge and node weights are given a new graph G' is constructed consisting of vertices $V' = V$ and edges $E' = E$. The cost of the edges $C'(e)$ is defined as

$$C'(e_i) = \begin{cases} C(e_i) & \text{if } e_i \notin S_1 \\ C(e_i) + f \times EW(e_i) & \text{if } e_i \in S_1 \end{cases}$$

where f is a multiplication factor whose choice is discussed later.

STEP IV Now the Floyd Warshall algorithm is used to generate the minimum cost paths in G' and the Takahashi Matsuyama heuristic is used to generate the second Steiner tree S_2 .

Now we can discuss the rationale behind the Dualcast heuristic. The edge weight of an edge is the number of receivers that get the data over that edge. (The node weight is just a temporary variable to help calculate the edge weights.) This weight can be looked on as a *criticality factor* for the specific



edge, because if that edge or link fails, or causes an error on the data transmit on it, then the number of receivers that will suffer due to this is nothing but the edge weight. A naive way to get totally disjoint Steiner trees would be to remove all the edges constituting the the first tree and then find a second Steiner tree. This method would, however, often result in a very sub optimal second tree, and would not offer any control over the tradeoffs we mentioned in the introduction. So we do not remove the edges but instead increase the cost of the edges in proportion to their *criticality* which will decrease the chance of those edges being in the second Steiner tree.

3.1 Multiplication Factor

When the cost of an edge was transformed to a new cost, a multiplication factor of f was used. If we review the equation

$$C'(e_i) = \begin{cases} C(e_i) & \text{if } e_i \notin S_1 \\ C(e_i) + f \times EW(e_i) & \text{if } e_i \in S_1 \end{cases}$$

it can be seen that the degree of disjointness of the two trees is proportional to the value of this multiplication factor. If f is large then the transformed cost will be higher for the edges and the second tree will probably not include those edges.

In fact the independence of the two trees essentially depends on this factor. During the tests of the scheme this value was varied from 0% to 50% of the mean cost of the edges of the network.

Let f be $x\%$ of the mean cost of the edges, i.e.,

$$f = x \times \overline{C(e)}$$

where $\overline{C(e)}$ is the mean cost of an edge in the network, then

$$C'(e_i) = C(e_i) + x \times \overline{C(e)} \times EW(e_i)$$

So an estimate of the increase in $C(e_i)$ is:

$$\frac{C'(e_i) - C(e_i)}{\overline{C(e)}} = x \times EW(e_i)$$

4. Performance Evaluation

In this section we compare *Dualcast* to standard multicasting using a single Steiner tree. We first develop a measure of the outage in the network due to various error conditions like the breakdown of a link, discarding of a packet, transmission error etc. Next we define two bases for comparison and develop inequalities which show under what conditions

Dualcast works better. Lastly we plot the results of our simulations of this routing strategy.

4.1 Expectation of outage

Given that a packet of data is transmitted along one or both of the Steiner trees so found, we can find an estimate of the expectation of outage. We define this expectation as the average number of multicasting destinations (receivers) that will receive erroneous packets due to an error on a link, or will not receive packets due to packet loss or link failure, given that an error or link failure does occur. As our goal is fault tolerance, a quantification of this measure is obviously necessary.

Now for the first scheme in which only one Steiner tree is used, this is a fairly straight forward calculation:

Consider the space defined by the event that there is exactly one error on one of the links which constitute the Steiner tree S_1 . Let us define a random variable X such that $x = i$ is the event that the i^{th} edge is the faulty one. Then assuming uniform distribution of error, the probability of this event is:

$$p_i = Prob\{X = i\} = \frac{1}{L_1} \quad \forall i$$

For each edge e_i in S_1 let us define a corresponding n_i , the number of receivers dependent on that edge. A receiver is dependent on an edge when the edge lies on the path from the source to that receiver. Now we can find the expected number of receivers that fail when one link in the Steiner tree is faulty. This expectation is:

$$\overline{n_1} = \sum_{S_1} n_i \times p_i$$

which can be written as

$$\overline{n_1} = \frac{\sum_{S_1} n_i}{L_1}$$

To calculate the expectation for *Dualcast*, let us assume, as before that a single link is in error, but in this case it can be on either of the two trees. Let the number of edges in S_1 be L_1 and in S_2 be L_2 . Given this, the probability that the i^{th} edge is the faulty one is

$$p_i = Prob\{X = i\} = \frac{1}{L_1 + L_2} \quad \forall i$$

Now we need to calculate the number of receivers dependent on each edge. By the assumption of a single link failure, we can see that a receiver will fail only if the faulty edge lies

on both the paths from the destination to that receiver. We call such links “critical” links. It has to be noted here that an edge e_i being in $S_1 \cap S_2$ is a necessary but not a sufficient condition for it to be “critical”. The edge is “critical” only if it lies on both the paths to any multicasting node m_j .

Let CL be the set of links or edges that are critical and for all $e_i \in CL$ let f_i be the number of receivers that fail when edge e_i fails. Thus f_i is the number of receivers that have edge e_i on both of their paths from the distinguished or transmitting vertex.

Then the expected number of receivers that fail is:

$$\pi_2 = \sum_{CL} f_i \times p_i$$

which is

$$\pi_2 = \frac{\sum f_i}{L_{1+2}}$$

However we must note here that this expression is calculated given that an error occurs in either S_1 or S_2 whereas for the single tree case the expectation was conditioned on an error occurring in S_1 alone.

If the distribution of errors is uniform then

$$\frac{\text{Prob}\{\text{faulty link} \in S_1 \cup S_2 \mid \text{a link fails in the network}\}}{\text{Prob}\{\text{faulty link} \in S_1 \mid \text{a link fails in the network}\}} = \frac{L_{1+2}}{L_1}$$

As it is more likely that an error will occur in the latter case where there are two trees, to compare the two we use scale π_2 to $\overline{\pi_2}$ where

$$\overline{\pi_2} = \pi_2 \times \frac{L_{1+2}}{L_1}$$

which upon substitution is

$$\overline{\pi_2} = \frac{\sum f_i}{L_{1+2}} \times \frac{L_{1+2}}{L_1} = \frac{\sum f_i}{L_1}$$

The values of f_i and n_i are calculated for each tree from the simulation.

Throughout the calculation we have assumed that the probability of error on a link is independent. This is only partially true. If the cost of a link in the network is a measure of the time the packet will stay on the link then the probability of error for that link should be a function of the cost. This is especially when we consider cell discarding in an ATM net-

work. A high cost for a link implies that it is highly loaded so the chances of a cell being discarded on that link are higher. However to keep the calculation simple we have not taken this cost into account.

5. Comparison

Clearly our scheme in which the packet of data is transmitted over two trees will generate more load on the network, whence load is measured as the number of packets transported by the network. However, as we mentioned earlier, with the advent of fiber optics and the enormous bandwidth it offers, we might not mind offering a higher load to the network, say if there were strict timing constraints on the application and transmitting over two paths reduced the total time taken to finish the transmission.

So we compare the schemes under two criteria, viz, which of the schemes takes less total time to finish transmitting data generated by an application process, and which of the two has less “network usage”. These criteria are clearly highly dependent on the actual protocol employed and factors like packet size, window size, buffer and frame size etc. However if we make the following simplifying assumptions, we can make a comparison between the two schemes without considering any specific protocols.

- Let the amount of data, say K bytes, after which an ack or request for retransmission is sent, be the same for both the schemes.
- Let the timeout period be the same.
- The retransmission is assumed to be unicast in both cases.
- In our scheme where the transmission is duplicated, the receiver acks only the first packet it receives.
- The time for control packets that are sent back to the transmitter asking for a retransmission of the lost/erroneous packet is disregarded.
- The total transmission time is more dependent on propagation delays/queuing times, than on the speed of the transmitter.

5.1 Network Usage

In this section we explore the load offered to the network by our scheme. We measure this in terms of a network usage and assume that the cost of an edge in the graph is some abstract measure of the per byte usage of the link depicted by

that edge. So the network usage incurred by transmitting K bytes over a link with cost C is $K \times C$. Also, the usage in multicasting a packet of K bytes to the destinations along the Steiner tree spanning them is just the weight of the spanning tree times K .

Let the weight (the sum of the cost of the edges comprising the tree) of the first Steiner tree S_1 be W_1 and that of the second tree be W_2 .

If there is only the first Steiner tree, and corresponding to that the expected outage is \bar{n}_1 , and if every one in x packets suffers some error in some part of the tree, then the total time of network usage is given by

$$(x \times W_1 \times K) + (\bar{n}_1 \times \text{the usage of the network by retransmitting the packet the destination})$$

The retransmission usage in the worst case will be the cost of the longest path to a receiver $x K$. An upper bound for the cost of the longest path is W_1 . So the equation now becomes

$$(x \times W_1 \times K) + (\bar{n}_1 \times W_1 \times K)$$

For the case where simultaneous transmission is occurring over S_1 and S_2 the network usage becomes:

$$(x \times (W_1 + W_2) \times K) + (\bar{n}_2' \times W_1 \times K)$$

where n_2 is the expected outage using Dualcast.. (This is so because the retransmission is assumed to be unicast and the upper bound for the cost of the longest path is the weight of the tree)

So for our scheme to have lower network usage:

$$(x \times (W_1 + W_2) \times K) + (\bar{n}_2' \times W_1 \times K) < (x \times W_1 \times K) + (\bar{n}_1 \times W_1 \times K)$$

Simplifying

$$\bar{n}_2' < \bar{n}_1 - \frac{x \times W_2}{W_1}$$

So as we increase the disjointness of the two paths, \bar{n}_2' should decrease; however, weight of the second tree will increase causing the right hand side of the above inequality to decrease. Thus the results should show a region of values of f the multiplication factor, where the inequality is specified. If lower values are chosen, the two trees will not be disjoint enough and \bar{n}_2' will be too high to realize the benefits of the double transmission. If values above this region are used then the second tree, though substantially different from the first, is of such a high cost that the larger costs offsets any gain from the reduction of errors.

We wish to emphasize at this point that since we have not considered the time taken by the control packets, our scheme will actually work better than what is shown by the expression derived above

5.2 Time for transmission

The total time here refers to the time to transmit the packets and the time to retransmit any packets that have not reached any destination successfully. If we use the weights of the tree as the upper bound of the longest path length and use the cost as a measure of the time a packet stays on that path, then we can say that the time to transmit a packet is

$$\max\{W_1 \times K, W_2 \times K\}$$

As the second tree is longer than the first, this will be $W_2 \times K$. Now the time for retransmission of erroneous packets is the same as that calculated for the "network usage" criterion. So now we have

$$x \times W_2 \times k + \bar{n}_2' < x \times W_1 \times K + \bar{n}_1 \times W_1 \times K$$

This on simplifying gives

$$\bar{n}_2' < \bar{n}_1 - \frac{x \times (W_2 - W_1)}{W_1}$$

6. Results

The algorithm as presented above was run on different randomly generated networks with varying size of the multicasting set and the network itself. The costs of the edges of the network were normally distributed around a preselected mean. Other factors that were varied were the standard deviation of the costs of the network and the connectivity. A connectivity of $x\%$ implies that, on the average, a node is connected to $x\%$ of the nodes in a network. In this paper we present results for a graph with 100 nodes, a connectivity of 15% and the path length normally distributed around 40. The size of the multicasting set is 50.

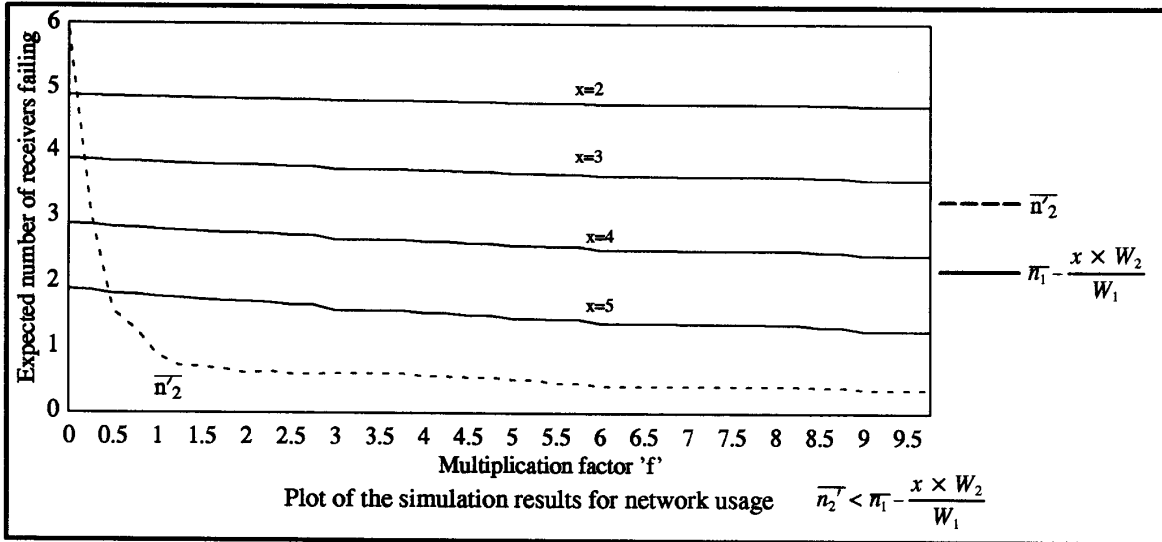
We first plot the inequalities derived for network usage and transmission time criteria, i.e, the following

$$\bar{n}_2' < \bar{n}_1 - \frac{x \times W_2}{W_1}$$

$$\bar{n}_2' < \bar{n}_1 - \frac{x \times (W_2 - W_1)}{W_1}$$

Dualcast works better in the region where \bar{n}_2' (the dotted line) is less than the solid lines. This is the region where the inequalities are satisfied.

We have also appended some statistics regarding the trees to the paper



Acknowledgements

The authors would like to thank Ms Nivedita Singhvi, discussions with whom helped formulate the ideas presented in this paper

Bibliography

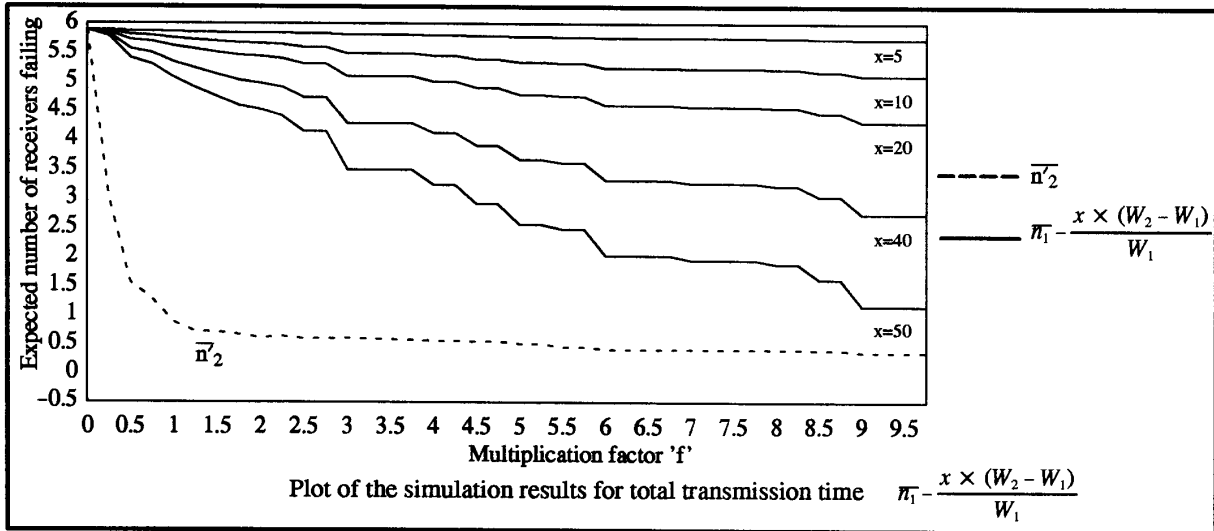
[1] N.F. Maxemchuk, "Dispersity routing in high speed networks", *Computer Networks and ISDN Systems*, vol 25, pp. 645-661,, 1993.
 [2] L. Kleinrock, "The latency/bandwidth tradeoff in gigabit networks", *IEEE Communications Magazine*, pp. 36-40, April 1992.

[3] Arun N. Netravali, W. D. Roome, K. Sabnani, "Design and implementation of a high speed transport protocol", *IEEE Trans. Comm.*, vol. 38, pp. 2010-2024, 1990.

[4] Takeshi Akakike *et al*, "Distributed multilink system for very-high-speed data link control", *IEEE JSAC*, vol. 11, pp. 540-548, 1993.

[5] J. W. Suurballe, R. E. Tarjan, "A quick method for finding shortest pairs of disjoint paths", *Networks*, vol. 14, pp. 325-336, 1984.

[6] R. G. Ogier, N. Sacham, "A distributed algorithm for finding the shortest pairs of disjoint paths," *Proc. IEEE INFOCOM*, pp. 178-182, 1989.



[7] V. Rutenberg, "Efficient distributed algorithms for computing shortest pairs of maximally disjoint paths in communication networks", *Proc. IEEE INFOCOM*, pp. 1214-1219, 1991.

[8] V. P. Kompella *et al*, "Multicasting for multimedia applications", *Proc. IEEE INFOCOM*, pp. 2078-2085, 1992.

[9] V. Rayward-Smith, "The computation of nearly minimal Steiner Trees in graphs", *Intl J. Math. Educ. Sci. Tech.*, vol. 14, pp. 15-23, 1983.

[10] H. Takahashi, A. Matsuyama, "An approximate solution for the Steiner problem in graphs", *Math. Japonica*, vol. 6, pp. 573-577, 1980.

[11] S. Hakimi, "Steiner's problem in graphs and its applications", *Networks*, vol.1, pp. 113-133, 1971.

