Delay Analysis of Broadcast Routing in Packet-Switching Networks

by

Gita Krishnan and J.W. Wong

Research Report CS-81-05

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

February 1981

Faculty of Mathematics

University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

Abstract

Broadcast addressing is the capability to send a packet from a source node to all other nodes in the network. Store-and-forward, packet-switching networks are not inherently designed to carry broadcast packets, and broadcasting has to be implemented by some sort of routing algorithm. In this paper, the source based forwarding algorithm is considered. With this algorithm, a spanning tree is defined for each node, and broadcast packets are sent along the branches of these trees. Approximation methods are presented to obtain a lower bound and estimates of the mean broadcast time for Kleinrock's packet-switching model. The accuracy of these methods is evaluated by comparison with simulation. The effect of the choice of spanning trees for the source based forwarding algorithm on the mean broadcast time is also studied.

1. Introduction

In a store-and-forward packet-switching network, broadcast addressing is the capability to send a packet from a source node to all other nodes. This is accomplished by executing a routing function at each node which determines the outgoing channels an incoming packet should take so that all nodes eventually receive the packet. There are several applications in which the broadcast facility is required. There are also applications in which multicasting is necessary. Multicasting allows a packet to be sent to more than one destination node (broadcasting is therefore a special case of multicasting). In [1], it was mentioned that multicast finds applications in the Advanced Research Projects Agency (ARPA) network user authentication and billing scheme [2], and in corporations where the headquarter broadcasts news and directives to the branch offices via a private network.

Another important application of broadcasting is in distributed database systems. The idea of having multiple copies of a database at various sites linked together by a packet-switching network is very attractive. This improves reliability since several copies of critical portions of a database exist and can be accessed even if a site fails or the links to a site break down making the site inaccessible. Replicated databases also enhance the responsiveness of the system since data may be stored near to where it is most frequently accessed. Distributing the database however introduces the problems of consistency and synchronization of updates [3]. It is necessary to perform updates in a manner such that the mutual consistency of the redundant copies and the internal consistency of each copy are both preserved. Several algorithms for concurrency control for multiple copy databases have been proposed. Examples of such algorithms are Thomas' majority consensus algorithm [4], Gray's two-phase commit algorithm [5], and those implemented in SDD-1 (A System for Distributed Databases) [6] and the distributed version of Ingres [7]. These algorithms present different approaches to the same problem but they have one thing in common -- they all require that an update request be broadcast to a number of destination nodes, and the performance of these algorithms would be greatly improved if broadcasting of packets in the communication net-

work can be done efficiently.

Since store-and-forward packet-switching networks are not inherently designed to carry broadcast packets, broadcasting has to be implemented by some sort of routing algorithms. In [1], Dalal and Metcalfe have studied the following algorithms :

(a) Seperately Addressed Packets : One copy of the broadcast packet is made for each destination node and these packets are delivered as point-to-point packets.

(b) Multidestination Addressing : Packets have a fixed length bit map to carry multiple destination addresses. At each node, copies of a packet are transmitted to one or more outgoing channels according to the packet's destination addresses. The set of destination addresses for each copy become smaller and the destination address fields of these copies are modified accordingly.

(c) Hot Potato Forwarding : At each node the broadcast packet is copied onto all channels except the one by which it arrived at the node. Provisions are made to avoid flooding the network.

(d) Spanning Tree Forwarding : A spanning tree is imposed upon the network, and the broadcast packet is transmitted along the branches (or channels) of the spanning tree except the one on which it arrives.

(e) Source Based Forwarding : Same as spanning tree forwarding except there is a separate spanning tree for each source node rather than the same tree for all nodes.

(f) Reverse Path Forwarding : When a broadcast packet arrives at a node it is processed if and only if the incoming channel is the best way (based on some criterion) to get from the node to the source of the broadcast packet. If this condition does hold, the broadcast packet is copied onto all the channels except the incoming channel.

(g) Extended Reverse Path Forwarding : Same as Reverse Path Forwarding except that information from neighbouring nodes is used to reduce the number of outgoing channels over which copies

of a broadcast packet are transmitted.

A key performance measure of broadcast routing is the mean broadcast time which is the mean delay before all nodes receive the broadcast packet. Dalal and Metcalfe [1] have found that Source Based Forwarding, Reverse Path Forwarding, and Extended Reverse Path Forwarding give good performances. Their analysis, however, is based on the assumption of no queueing delays at the channels (except for Separately Addressed Packets). This assumption is not realistic although it simplifies the analysis significantly.

In this paper, we study broadcast routing without making the simplifying assumption that there are no queueing delays at the channels. Due to the complexity of the mathematics involved, we will restrict our attention to the Source Based Forwarding algorithm only. Our analysis is based on an extension of Kleinrock's model for packet-switching networks [8] to include the Source Based Forwarding algorithm. Both point-to-point and broadcast packets are considered. The results are directly applicable to Spanning Tree Forwarding since it is a special case of Source Based Forwarding. They are also applicable to Extended Reverse Path Forwarding if we make the assumption that the reverse path tree is known a priori.

In section 2, the queueing model used in this study is defined. The exact analysis of mean broadcast time is very difficult. Attempts are therefore made to obtain bounds and approximate results. In section 3, an approximate lower bound for mean broadcast time is presented. Analytic results based on two other approximation methods are also derived. These derivations are given in sections 4 and 5 respectively. In section 6, the accuracy of these approximate methods is evaluated by comparison with simulation results. The effect of the choice of spanning trees for Source Based Forwarding on mean broadcast time is also studied.

2. Queueing Model

2.1. Model Description

Our study is based on an extension of Kleinrock's packet-switching network model to include broadcast packets and Source Based Forwarding. In this model, the delay experienced by a packet is approximated by the waiting time and data transmission time at the channels. The processing time at the switching nodes and the propagation delay are assumed to be negligible. Let M be the number of channels and C_i the capacity of channel i, i = 1,2,...,M. Each channel is modelled by a single server queue. It is assumed that all channels are error free and all nodes have unlimited buffer space.

There are 2 types of packets - broadcast (from a source node to all other nodes in the network) and point-to-point (from a source node to a single destination node). Let N be the number of nodes. For s = 1,2,...,N, the arrival process of broadcast and point-to-point packets from outside the network to source node s is assumed to be Poisson with mean rates γ_s^b *and* γ_s^p respectively. A point-to-point packet is assumed to be sent with equal probability to any one of the N-1 destination nodes. Also, the lengths of all packets are assumed to have the same exponential distribution with mean $\frac{1}{\mu}$. The data transmission time of any packet at channel i is therefore exponential with mean $\frac{1}{\mu C_i}$.

The routing algorithm for broadcast packets is Source Based Forwarding. There is a spanning tree for each source node. A broadcast packet is transmitted along the spanning tree according to the node at which the broadcast is originated. The routing algorithm for point-to-point packets is assumed to be fixed and uses the same paths as the broadcast packets. Finally, Kleinrock's independence assumption [8] is used to make the mathematical analysis tractable. This assumption states that each time a packet (broadcast or point-to-point) enters a node, a new length is generated for this packet from the exponential packet length distribution.

2.2. Performance Measure

As mentioned previously, the performance measure of interest is the mean broadcast time. Specifically, the mean broadcast time for source node s (denoted by B_s) is the mean delay before all the N-1 destinations receive a broadcast packet from node s. The mean broadcast time for all nodes in the network is then given by :

$$B = \frac{\sum_{s=1}^{N} \gamma_s^b B_s}{\sum_{s=1}^{N} \gamma_s^b} \tag{1}$$

In what follows, we derive an approximate lower bound and two other approximate expressions for B_s and B.

3. Approximate Lower Bound for Mean Broadcast Time

We assume that the network topology and the external arrival rates of both broadcast and point-to-point packets are all specified. With Source Based Forwarding, the route taken by a broadcast packet from node s can be represented as a spanning tree with node s as the source. We call this the *broadcast tree* for node s. As discussed earlier, a fixed routing algorithm based on this tree is used for point-to-point packets.

Consider a packet originating from source node s. Let $\tilde{t}_{s,d}$ be the delay for this packet to reach destination node d, $d \neq s$, and $T_{s,d}$ be its mean. The mean broadcast time is given by :

$$B_s = E\left[\max_{d \in L_s} \{\tilde{t}_{s,d}\}\right] \tag{2}$$

where L_s is the set of "leaf" nodes in the broadcast tree for node s, and $E[\tilde{x}]$ stands for the mean of $\tilde{x}$. A simple lower bound for B_s is :

$$B_s \geq \max_{d \in L_s} \{T_{s,d}\} \tag{3}$$

We now derive an expression for $T_{s,d}$. Let $\gamma_{s,d}$ be the external arrival rate of (s,d) packets, i.e. packets originating at node s with destination d. Since a broadcast packet is sent to all nodes and a point-to-point packet is assumed to be sent with equal probability to any of the N-1 possible destinations, we have :

$$\gamma_{s,d} = \gamma_s^b + \frac{\gamma_s^p}{N-1} \tag{4}$$

The arrival process of these (s,d) packets is Poisson. This is a result of the assumption that both broadcast and point-to-point packet arrivals are Poisson.

Let $\lambda_{i;s,d}$, (i=1,2,...,M) be the mean arrival rate of (s,d) packets to channel i. Also let $\pi_{s,d}$ be the ordered set of channels forming the path from node s to node d as defined by the broadcast tree for node s. $\lambda_{i;s,d}$ is given by :

$$\lambda_{i;s,d} = \begin{cases} \gamma_{s,d} & \text{if } i \in \pi_{s,d} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Let $\rho_{i;s,d}$ be the utilization of channel i by (s,d) packets.

$$\rho_{i;s,d} = \frac{\lambda_{i;s,d}}{\mu C_i} \tag{6}$$

The total utilization of channel i is then given by :

$$\rho_i = \sum_{s=1}^{N} \sum_{d=1}^{N} \rho_{i;s,d} \tag{7}$$

It is necessary that $\rho_i < 1$ for i=1,2,...,M. This is the condition for the existence of stochastic equilibrium.

Assuming the model with broadcast packets belongs to the class of models analyzed by Baskett et al. [9], the following approximate expression is obtained for $T_{s,d}$ [10] :

$$T_{s,d} \approx \sum_{i \epsilon \pi_{s,d}} \frac{1}{\mu C_i (1-\rho_i)} \tag{8}$$

Equation (8) can be used in Equations (3) and (1) to obtain approximate lower bounds for B_s and B.

4. First Approximation Method

Our first approximation method is based on the assumption that the random variables $\tilde{t}_{s,d}$ $(d \neq s)$ in Equation (2) are independent. Let $T_{s,d}(x)$ be the cumulative distribution function of $\tilde{t}_{s,d}$ and $T^*_{s,d}(\xi)$ its Laplace Transform, i.e.,

$$T^*_{s,d}(\xi) = \int_0^\infty e^{-\xi x} dT_{s,d}(x) \tag{9}$$

Assuming again that the network model with broadcast packets belongs to the class of models analyzed by Baskett et al. [9], $T^*_{s,d}(\xi)$ is given by [10] :

$$T^*_{s,d}(\xi) \approx \prod_{i \in \pi_{s,d}} \frac{\mu C_i (1-\rho_i)}{\xi + \mu C_i (1-\rho_i)} \tag{10}$$

where ρ_i is given by Equation (7). $T^*_{s,d}(\xi)$ can easily be inverted to give the probability density function from which we can obtain $T_{s,d}(x)$.

Since the broadcast time for node s is $\max_{d \in L_s} \{ \tilde{t}_{s,d} \}$, its cumulative distributive function (denoted by $B_s(x)$) is given by [11] :

$$B_s(x) \approx \prod_{d \in L_s} T_{s,d}(x) \tag{11}$$

From $B_s(x)$ we get the following approximate expression for B_s :

$$\begin{aligned} B_s &= \int_0^\infty (1 - B_s(x))\, dx \\ &\approx \int_0^\infty [1 - \prod_{d \in L_s} T_{s,d}(x)]\, dx \end{aligned} \tag{12}$$

We also get an approximation for B by using Equation (12) in Equation (1).

5. Second Approximation Method

This method is best illustrated by the example broadcast tree shown in Figure 1. This tree is broken up into levels and the mean broadcast time is calculated starting from the bottom level (i.e. the level farthest away from the source node) and working up successively to the source node. At a given level, the nodes are either a leaf node of the broadcast tree or the root node of some subtree. For the case of a subtree, the root node's out-going channel(s) form a queueing system with one or more servers. Upon receiving a broadcast packet from the source node, the root node routes the packet to all its outgoing channel(s) simultaneously. Since the choice of broadcast trees for other source nodes is arbitrary, broadcast packets from these nodes can be routed to any combination of out-going channels also. We thus have a multiple server queueing system where an arriving packet can be routed to any subset of servers simultaneously.

In our analysis, the number of outgoing channels at each node is limited to a maximum of three. This is due to the fact that the complexity in obtaining numerical results grows quickly with the number of channels (this will become obvious later). In the remainder of this section, we first present a general method for solving 1-server, 2-server and 3-server queueing systems with simultaneous arrivals. We then outline how the solutions can be used to obtain an approximate expression for the mean broadcast time.

5.1. One-Server Queueing Model

The one-server model is for subtrees whose root node has only one outgoing channel (e.g., node 4 of Figure 1). Let this channel be i; the model is shown in Figure 2. The arrival process is assumed to be Poisson with mean rate ϕ_i. ϕ_i includes the rate of broadcast packets originating from node 1 and those which originate from other nodes but have channel i in their broadcast tree. Using the notation defined in Section 3, ϕ_i is given by :

$$\phi_i = \sum_{all\ s,d} \lambda_{i;s,d} \tag{13}$$

In this model packets are served by channel i in first-come, first-served (FCFS) order and then

experience a constant delay D_i. D_i represents the additional delay to broadcast a packet to the remaining nodes in the subtree after leaving channel i. In the special case that the subtree consists of only one channel, $D_i = 0$.

The mean response time of this model (denoted by b_i) is the mean time to broadcast a packet through the subtree. It is approximated by the sum of D_i and the mean response time of an M/M/1 model with parameters ϕ_i and μC_i, i.e.,

$$b_i = \frac{1}{\mu c_i - \phi_i} + D_i \tag{14}$$

5.2. Two-Server Queueing Model

The two-server model is shown in Figure 3. It is a generalization of the one-server model to two-servers, representing subtrees whose root node has two outgoing channels (e.g. node 2 of Figure 1). There are three types of packets. Type 1 packets (with arrival rate ϕ_{ij}) are broadcast to both channels simultaneously. Types 2 and 3 (with arrival rates ϕ'_i and ϕ'_j) are sent to one channel only. Let G_s be the set of channels in the broadcast tree for source node s.

Formally, the arrival rates are given by :

$$\begin{aligned} \phi_{ij} &= \sum_{s:i,j \epsilon G_s} \gamma_s^b \\ \phi'_i &= \phi_i - \phi_{ij} \\ \phi'_j &= \phi_j - \phi_{ij} \end{aligned} \tag{15}$$

The two channels can be considered as two independent single server queues. Let $P(n_i,n_j)$ be the equilibrium probabilities that there are n_i *and* n_j packets (of any type) at channels i and j respectively.

The $P(n_i,n_j)$'s satisfy

$$\begin{aligned}
P(0,0)(\phi'_i+\phi'_j+\phi_{ij}) &= P(1,0)\mu C_i + P(0,1)\mu C_j \\
P(n_i,0)(\phi'_i+\phi'_j+\phi_{ij}+\mu C_i) &= P(n_i+1,0)\mu C_i + P(n_i,1)\mu C_j \\
&+ P(n_i-1,0)\phi'_i \qquad n_i > 0 \\
P(0,n_j)(\phi'_i+\phi'_j+\phi_{ij}+\mu C_j) &= P(0,n_j+1)\mu C_j + P(1,n_j)\mu C_i \\
&+ P(0,n_j-1)\phi'_j \qquad n_j > 0 \\
P(n_i,n_j)(\phi'_i+\phi'_j+\phi_{ij}+\mu C_i+\mu C_j) &= P(n_i-1,n_j)\phi'_i + P(n_i,n_j-1)\phi'_j \\
&+ P(n_i-1,n_j-1)\phi_{ij} + P(n_i+1,n_j)\mu C_i \\
&+ P(n_i,n_j+1)\mu C_j \qquad n_i,n_j > 0
\end{aligned} \tag{16}$$

It is not possible to get a closed form solution to these balance equations. A solution can only be obtained by numerical methods. We can restrict the total number of states by looking at only the states (n_i,n_j) such that $n_i < N_i$ *and* $n_j < N_j$ for chosen values of N_i *and* N_j. This means that we assume $P(n_i,n_j) = 0$ for $n_i \geq N_i$ *or* $n_j \geq N_j$. We then have N_iN_j equations in N_iN_j unknowns. One of the equations is dependent, and can be eliminated with the following normalization condition

$$\sum_{n_i=0}^{N_i-1}\sum_{n_j=0}^{N_j-1} P(n_i,n_j) = 1 \tag{17}$$

The set of equations can then be solved numerically.

N_i *and* N_j have to be carefully selected so that

$$\sum_{n_i=N_i}^{\infty}\sum_{n_j=N_j}^{\infty} P(n_i,n_j) << 1 \tag{18}$$

It is not possible to choose these values analytically. We have to select N_i *and* N_j on the basis that the computed approximate values of the probabilities $P(n_i,n_j)$ do not vary very much when the value of N_i *or* N_j is increased.

Choosing reasonable values of N_i *and* N_j usually yields quite a large set of equations to solve. These equations are solved using SPARSPAK [12] to obtain the state probabilities.

Let $\tilde{b}_{ij}$ be the time to broadcast a packet to all nodes in the subtree with two outgoing channels and b_{ij} be its mean. Also let $A(n_i,n_j)$ be the probability that an arrival finds the system

in state (n_i, n_j). Since the arrival process is assumed to be Poisson, we have [13]:

$$A(n_i, n_j) = P(n_i, n_j) \tag{19}$$

Conditioned on an arrival finding the system in state (n_i, n_j), the response time for the system is given by :

$$\tilde{b}_{ij}(n_i, n_j) = \max(\tilde{y}_i^*, \tilde{y}_j^*) \tag{20}$$

where

$$\tilde{y}_l^* = \tilde{y}_l + D_l \quad l = i,j \tag{21}$$

and $\tilde{y}_l$ follows an (n_l+1)- stage Erlang distribution, each stage having mean $\frac{1}{\mu C_l}$.

So we can infer

$$Pr\,[\,\tilde{y}_l^* \leqslant y\,] = Pr\,[\,\tilde{y}_l \leqslant y - D_l\,]$$

$$= 1 - \sum_{r_l=0}^{n_l} \frac{(\mu C_l (y - D_l))^{r_l} e^{-\mu C_l (y - D_l)}}{r_l!} \quad l = i,j \tag{22}$$

Assuming that $\tilde{y}_i^*$ *and* $\tilde{y}_j^*$ are independent, the cumulative distributive function (cdf) of the response time is given by [11] :

$$Pr\,[\,\tilde{y}_i^* \leqslant y\,] \,.\, Pr\,[\,\tilde{y}_j^* \leqslant y\,] \tag{23}$$

and the conditional mean response time can then be obtained from

$$b_{ij}(n_i, n_j) = \int_0^\infty (1 - Pr\,[\,\tilde{y}_i^* \leqslant y\,].Pr\,[\,\tilde{y}_j^* \leqslant y\,])\, dy$$

$$= \int_0^\infty (1 - \prod_{l=i,j} Pr\,[\,\tilde{y}_l \leqslant y - D_l\,])\, dy \tag{24}$$

Let D = max $\{ D_i, D_j \}$. Then, simplifying the integral (24), we have

$$b_{ij}(n_i, n_j) = D + X_i + X_j - X_i X_j \tag{25}$$

where

$$X_l = e^{-\mu C_l (D - D_l)} \sum_{r=0}^{n_l} \sum_{p=0}^{r} \frac{[\mu C_l (D - D_l)]^p}{\mu C_l p!} \quad l = i,j \tag{26}$$

Assuming $D_i \geqslant D_j$ (if not, exchange indices i and j), we have

$$X_i X_j = \sum_{r_i=0}^{n_i} \sum_{r_j=0}^{n_j} \frac{\mu C_i^{r_i} \mu C_j^{r_j}}{r_i! r_j!} e^{-\mu C_i (D - D_i)} e^{-\mu C_j (D - D_j)} H_{ij}$$

where

$$H_{ij} = \sum_{p=0}^{r_j} \binom{r_j}{p} (D_i - D_j)^{r_j - p} \sum_{q=0}^{r_i + p} \frac{(r_i + p)!\ (D - D_i)^{r_i + p - q}}{(r_i + p - q)!\ (\mu C_i + \mu C_j)^{q+1}} \tag{27}$$

There are 3 special cases that can lead to a simpler expression for $X_i X_j$ and possibly X_i (*or* X_j). These are given in the Appendix.

Removing the condition on (n_i, n_j), the mean response time for the subtree is given by :

$$b_{ij} = \sum_{n_i=0}^{N_i - 1} \sum_{n_j=0}^{N_j - 1} P(n_i, n_j)\ b_{ij}(n_i, n_j) \tag{28}$$

5.3. Three-Server Queueing Model

Generalizing the two-server model to three servers, we can model subtrees whose root node has 3 outgoing channels (denoted by i, j, and k). There are seven types of packets. The arrival rates of each type, together with the channel(s) along which the packets are sent, are shown in Table 1.

Type	Arrival Rate	Channel(s)
1	ϕ_{ijk}	all three channels
2	ϕ'_{ij}	i and j
3	ϕ'_{ik}	i and k
4	ϕ'_{jk}	j and k
5	ϕ''_{i}	i
6	ϕ''_{j}	j
7	ϕ''_{k}	k

Table 1

The arrival rates to the three-server model can be formally defined in a manner similar to the one-

server and two-server models.

Let b_{ijk} be the mean broadcast time in the subtree. Extending the approximation technique for the two-server model to three servers, we have

$$b_{ijk} = \sum_{n_i=0}^{N_i-1} \sum_{n_j=0}^{N_j-1} \sum_{n_k=0}^{N_k-1} P(n_i,n_j,n_k) b_{ijk}(n_i,n_j,n_k) \tag{29}$$

where $P(n_i,n_j,n_k)$ is the equilibrium state probability obtained by numerical methods, and

$$b_{ijk}(n_i,n_j,n_k) = D + X_i + X_j + X_k - X_iX_j$$

$$- X_jX_k - X_iX_k + X_iX_jX_k \tag{30}$$

The expressions for X_i, X_j, X_k, X_iX_j, X_jX_k *and* X_iX_k are analogous to Equations (26) and (27), and for $D_i \geqslant D_j \geqslant D_k$,

$$X_iX_jX_k = \sum_{r_i=0}^{n_i} \sum_{r_j=0}^{n_j} \sum_{r_k=0}^{n_k} \frac{\mu C_i{}^{r_i} \mu C_j{}^{r_j} \mu C_k{}^{r_k}}{r_i!\, r_j!\, r_k!} e^{-\mu C_j(D_i-D_j)-\mu C_k(D_i-D_k)} E_{ijk}$$

where

$$E_{ijk} = \sum_{p=0}^{r_j} \sum_{q=0}^{r_k} \binom{r_j}{p} \binom{r_k}{q} (D_i-D_j)^{r_j-p} (D_i-D_k)^{r_k-q} \frac{(r_i+p+q)!}{(\mu C_i+\mu C_j+\mu C_k)^{r_i+p+q+1}} \tag{31}$$

There are two special cases where one can get a simpler expression for $X_iX_jX_k$. They are given in the appendix.

5.4. Calculation of Mean Broadcast Time

We now illustrate how the results of the last three sections can be used to compute the mean broadcast time. We will again use the broadcast tree in Figure 1 and give the steps required to get an approximation for B_1, the mean broadcast time from node 1. The steps are as follows :

(a) Compute b_7 with $D_7 = 0$

(b) Compute b_{89} with $D_8 = D_9 = 0$

(c) Compute b_{45} with $D_4 = b_7$ *and* $D_5 = 0$

(d) Compute b_6 with $D_6 = b_{89}$

(e) Compute b_{123} with $D_1 = b_{45}$, $D_2 = 0$, *and* $D_3 = b_6$

B_1 is given by b_{123}.

5.5. Other Remarks

This approximation method is not specific to any network topology. It is rather general and can be used for an arbitrary network with any mixture of broadcast and point-to-point packets. It is also applicable to multicasting because the routing tree for multicasting is merely a subtree of some broadcast tree.

This approximation method considers the broadcast tree of a node in its entirety and also takes into effect the simultaneous arrivals of packets to more than one channel. Thus, it is likely to provide better results than the first approximation.

One drawback of this method is the amount of computation required to get the results. In the three-server case, the operation count is $O(n^5)$ where n is the number of packets in a queue in the multiserver queueing system. The amount of computation may be practically infeasible for models with more than three servers. That is why this method is presented for broadcast trees with a maximum of three outgoing channels per node only.

6. Results

We now evaluate the accuracy of the approximation methods described above. Our evaluation will be based on the two example networks shown in Figures 4 and 5. In both networks, the path length between any two nodes is at most 3. The 5-node network in Figure 4 has a couple of channels which could be potential bottlenecks. For example, all packets for nodes 1 and 5 have to be routed along channels 2 and 9 respectively. Each node in both networks has at most 3 outgoing channels. The 8-node network, as shown in Figure 5, has enough links to allow us to observe the effect of varying the spanning trees used in the broadcast routing algorithm.

There are two types of computer networking environments which are of interest to this study. In the first environment, there is one primary computer connected to a number of secondary processors. In this type of logically centralized networks, only one node generates broadcast packets. All nodes in the network can generate point-to-point packets. This environment is applicable to situations where a corporation headquarters desires to broadcast directives to the branch offices. We are interested in how the mean broadcast time for the network (i.e. for the node doing the broadcasting) varies as the arrival rate of broadcast packets increases. In the second computing environment, every node may generate broadcast packets. This is applicable to a distributed database management system for example.

We will study these two types of environments in more detail, applying the lower bound and approximation methods developed previously.

6.1. First Environment

Two cases are considered - one with a heavy background load of point-to-point packets and the other with a light background load. For the 5-node network, node 1 is assumed to be the node generating broadcast packets. The other nodes initiate only point-to-point packets. Similarly, we assume node 7 is the node doing broadcasts in the 8-node network.

Tables 2 and 3 give the mean broadcasting times obtained for the two networks for the first

γ_r^p	Sim	Lower Bd	Err	1st App	Err	2nd App	Err
0.5	0.550	0.545	-0.9	0.603	9.6	0.558	1.5
1.0	0.616	0.600	-2.6	0.662	7.5	0.612	-0.6
1.5	0.670	0.667	-0.4	0.733	9.4	0.679	1.3
2.0	0.753	0.750	-0.4	0.822	9.2	0.761	1.0
2.5	0.868	0.857	-1.3	0.937	7.9	0.866	-0.2
3.0	1.019	1.000	-1.9	1.088	6.8	1.007	-1.2
3.5	1.228	1.200	-2.3	1.300	5.9	1.204	-2.0
4.0	1.512	1.500	-0.7	1.616	6.9	1.497	-1.0

5-node network, node 1 broadcasting
γ_s^p = 5.0 for all nodes.

Table 2

γ_r^p	Sim	Lower Bd	Err	1st App	Err	2nd App	Err
0.5	0.625	0.490	-21.6	0.655	4.8	0.562	-9.4
1.0	0.678	0.533	-21.4	0.712	5.0	0.608	-10.3
1.5	0.732	0.586	-19.9	0.779	6.4	0.664	-9.3
2.0	0.817	0.650	-20.4	0.860	5.3	0.731	-10.5
2.5	0.900	0.730	-18.9	0.961	6.8	0.815	-9.4
3.0	1.013	0.833	-17.8	1.089	7.5	0.921	-9.1
3.5	1.177	0.971	-17.5	1.259	7.0	1.061	-9.9
4.0	1.373	1.167	-15.0	1.493	8.7	1.253	-8.7

8-node network, node 7 broadcasting
γ_s^p = 4.0 for all nodes.

Table 3

case in which the background load of point-to-point packets is heavy. It can be seen that the two approximation methods give accurate results. The worst case error is 10.5%. The first method is more accurate for the 8-node network while the second is more accurate for the 5-node network. The first method also yields results which are consistently higher than those obtained from simulation. The lower bound is not bad either when it is used as an approximation. It is, on the average, 1.3% and 19.1% away for the 5-node and 8-node network respectively.

We next consider the case in which the background load of point-to-point packets is light. Tables 4 and 5 tabulate the results. It is observed that the two approximation methods also yield accurate results for this case. The second approximation is much better for both networks (this is different from the case of a heavy background load of point-to-point traffic). The first method is again giving results which are consistently higher than those obtained by simulation.

We will make some general comments about these results after discussing the second type of networking environment.

6.2. Second Environment

In these examples it is assumed that the arrival rates of broadcast packets at all the nodes are equal. For convenience, we assume there are no point-to-point packets in the network. Tables 6 and 7 give the results obtained using the approximation techniques. Once again, we observe that the approximation results are accurate. The second approximation method is better for both networks, and the first approximation yields results that are pessimistic estimates of the mean broadcast time.

6.3. Discussion of Results

The approximation method using the bottom-up approach yields values very close to the simulation results. It is at worst 10.5% away from the simulation values. In some cases there is less than 1% difference between the two. On the average they differ by about 3-5%, which is quite good. This approximation method is specially accurate for the second type of computing environment.

γ_1^p	Sim	Lower Bd	Err	1st App	Err	2nd App	Err
1.0	0.435	0.429	-1.4	0.456	4.8	0.445	2.3
2.0	0.509	0.500	-1.8	0.531	4.3	0.517	1.6
3.0	0.603	0.600	-0.5	0.636	5.5	0.618	2.5
4.0	0.760	0.750	-1.3	0.793	4.3	0.767	0.9
5.0	1.002	1.000	-0.2	1.054	5.2	1.016	1.4
6.0	1.542	1.500	-2.7	1.572	1.9	1.506	-2.3

5-node network, node 1 broadcasting
γ_s^p = 2.5 for all nodes.

Table 4

γ_1^p	Sim	Lower Bd	Err	1st App	Err	2nd App	Err
1.0	0.513	0.410	-20.0	0.563	9.7	0.480	-6.4
2.0	0.588	0.474	-19.4	0.646	9.9	0.551	-6.3
3.0	0.694	0.564	-18.7	0.764	10.1	0.650	-6.3
4.0	0.844	0.694	-17.8	0.939	11.3	0.794	-5.9
5.0	1.099	0.905	-17.7	1.214	10.5	1.020	-7.2
6.0	1.530	1.300	-15.0	1.721	12.5	1.432	-6.4

8-node network, node 7 broadcasting
γ_s^p = 2.0 for all nodes.

Table 5

γ_s^b	Sim	Lower Bd	Err	1st App	Err	2nd App	Err
0.5	0.294	0.277	-5.8	0.333	13.3	0.309	5.1
0.75	0.322	0.302	-6.5	0.362	12.4	0.335	4.0
1.0	0.358	0.333	-7.0	0.399	11.5	0.367	2.5
1.25	0.414	0.375	-9.4	0.448	8.2	0.409	-1.2
1.5	0.482	0.435	-9.7	0.519	7.7	0.475	-1.5
1.75	0.590	0.529	-10.3	0.632	7.1	0.576	-2.4
2.0	0.809	0.708	-12.5	0.853	5.5	0.771	-4.8

5-node network, all nodes broadcasting
γ_s^p = 0.0 for all nodes.

Table 6

γ_s^b	Sim	Lower Bd	Err	1st App	Err	2nd App	Err
0.5	0.434	0.359	-17.3	0.486	12.0	0.420	-3.2
0.75	0.483	0.399	-17.6	0.539	11.4	0.467	-3.5
1.0	0.549	0.450	-18.0	0.603	9.8	0.524	-4.6
1.25	0.634	0.518	-18.3	0.698	10.1	0.602	-5.0
1.5	0.743	0.615	-17.1	0.821	10.5	0.710	-4.3
1.75	0.919	0.765	-16.8	1.017	10.7	0.871	-5.2
2.0	1.273	1.042	-18.1	1.381	8.5	1.167	-8.3

8-node network, all nodes broadcasting
γ_s^p = 0.0 for all nodes.

Table 7

The disadvantage of this method is the large amount of computation involved in getting numerical results.

The approximate lower bound gives a very tight estimate in some cases (with less than 1% difference between the simulation and the lower bound values), while in other cases it is off by a larger amount. For all the examples that were tried, the "worst case" difference between the lower bound and the simulation is 21.6%. The lower bound is meant as a rough estimate and the fact that it is so easy to compute is a point in its favour.

It is observed that the first approximation method consistently gives results higher than the simulated values. Similar observations were made for other networks not reported in this paper. We thus conjecture that the first approximation method gives an upper bound for the mean broadcast time for the network. We feel strongly that this is the case but have not been able to prove it so far. The difference between the simulation results and the conjectured upper bound ranges between 1.9% and 13.3% with the average being 7-9%. This is quite a tight bound if it indeed is an upper bound.

We conclude from the above discussion that the second approximation is very accurate, but is expensive to compute. One can sacrifice some accuracy and get upper and lower bounds instead. These bounds are easy to compute and simulation experiments have shown that they are tight bounds.

6.4. Spanning Trees

In order to analyze broadcast routing in store-and-forward networks, we have restricted ourselves to the Source Based Forwarding algorithm described in the first section. Each node has a spanning tree along which broadcast packets initiated by that node are routed. We are interested in how the choice of the spanning tree affects the mean broadcasting time experienced by the node. We define a measure of "flatness" (F) for a spanning tree as follows:

F = Maximum depth of tree / Number of leaves

Intuitively, the flatter the tree, the lower is the mean broadcast time.

We consider again the 8-node network shown in Figure 5. Figures 6 and 7 show 5 different spanning trees for node 6 and node 7. The spanning trees were chosen such that the 5 trees for a node all have different values of F. The graphs in figures 8 and 9 show the relationship between F and the simulation result for mean broadcast time. As expected, the smaller the value of F, the smaller is the mean broadcasting time for the node. So, if the mean broadcasting time is to be minimized, a good rule is to select spanning trees with low F.

7. Conclusion

In this paper, we have studied broadcast routing in a packet-switching network and analyzed the performance of the source based forwarding algorithm. Our analysis is based on an extension of Kleinrock's packet-switching network model to include broadcast packets. A simple, but approximate lower bound for the mean broadcast time was first obtained. Two other methods were also presented to obtain approximations for the mean broadcast time. Comparison with simulation has shown that the first approximation method consistently yields pessimistic estimates. This leads us to conclude that it can be used as an upper bound. Results based on the first approximation method are also easy to compute. One can therefore obtain upper and lower bounds to the mean broadcast time easily. Simulation experiments have shown that these bounds are rather tight. If more accurate results are required, the second approximation method can be used. This method has the disadvantage that numerical results are more costly to compute.

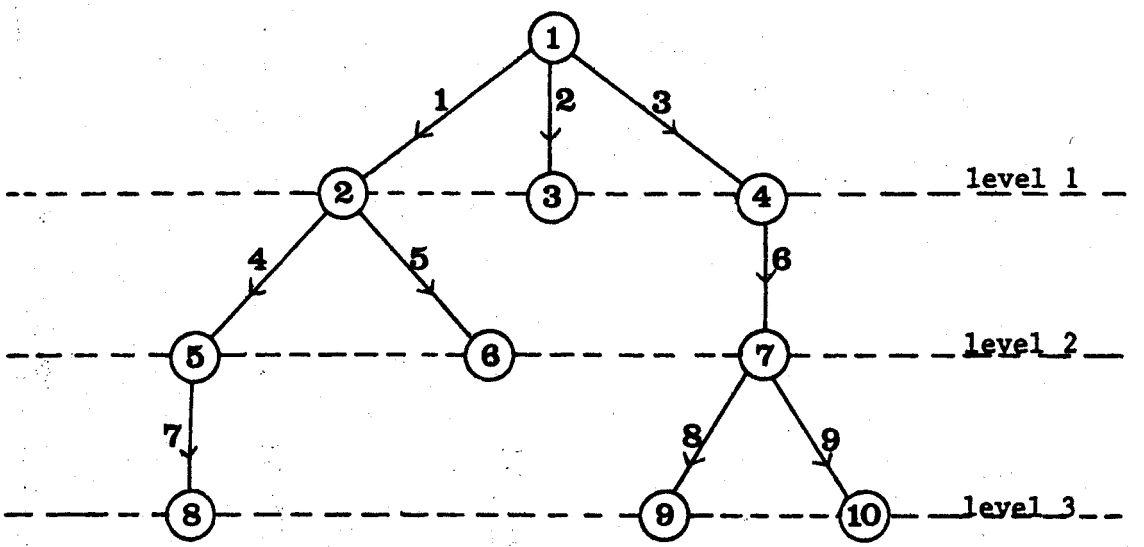


Figure 1

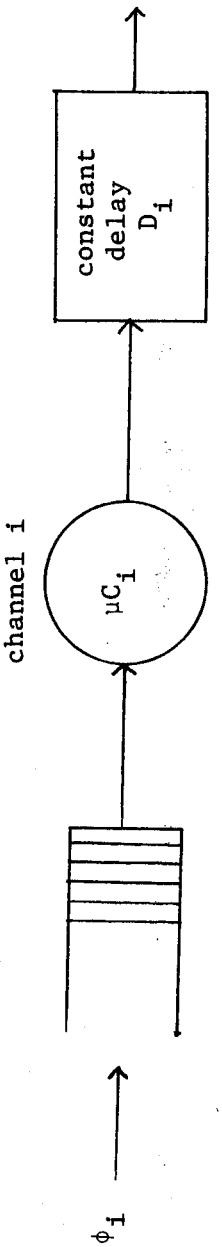


Figure 2

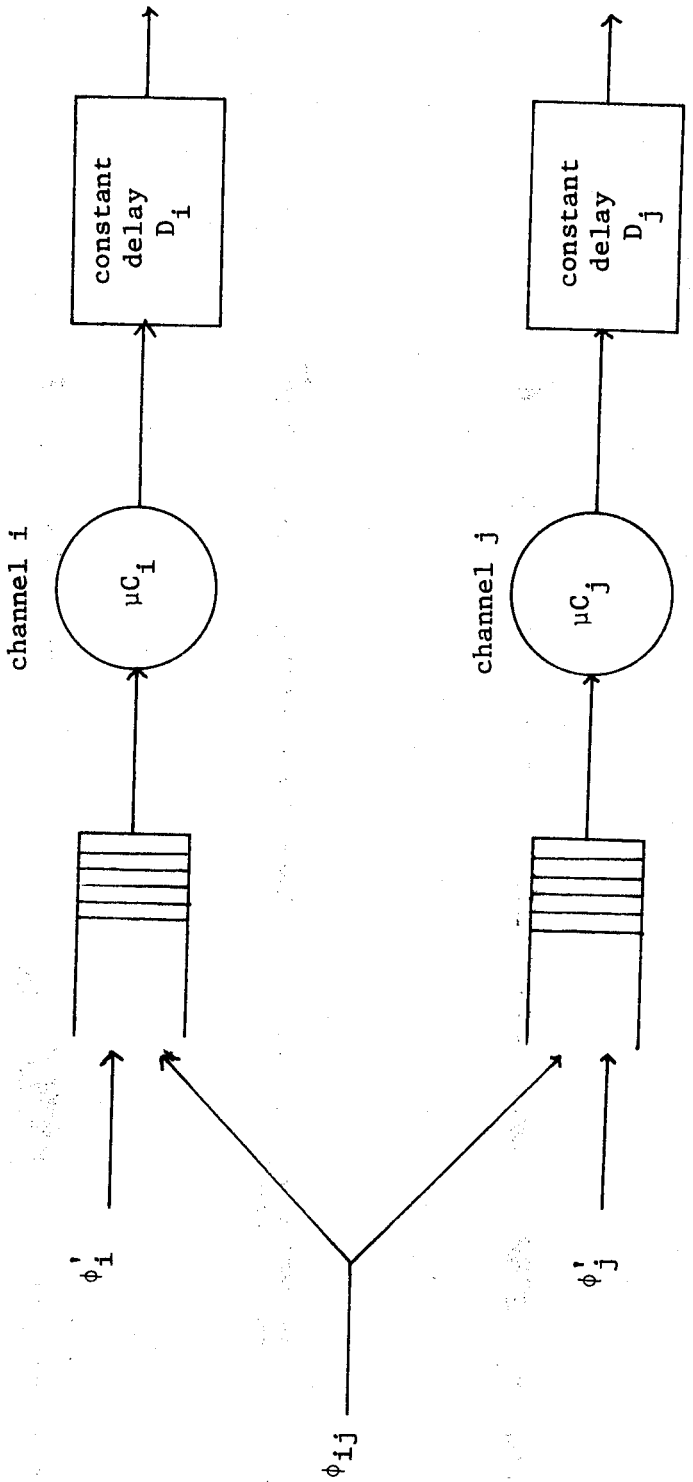

channel i
channel j
constant delay D_i
constant delay D_j
μC_i
μC_j
ϕ'_i
ϕ'_j
ϕ_{ij}

Figure 3

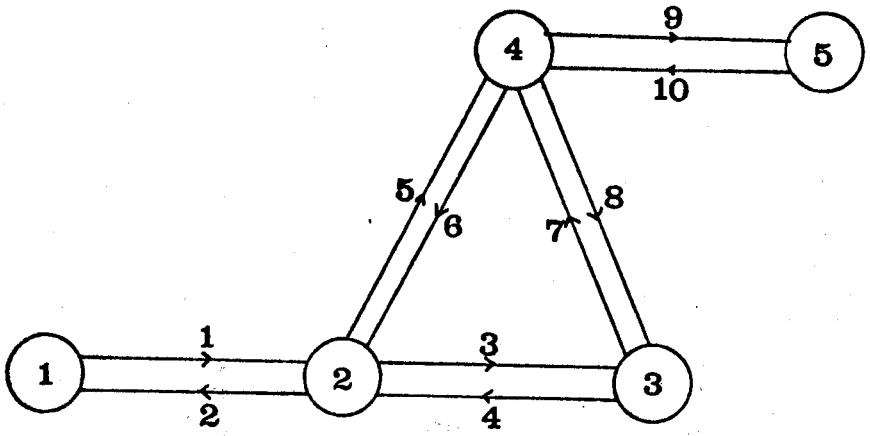


Figure 4

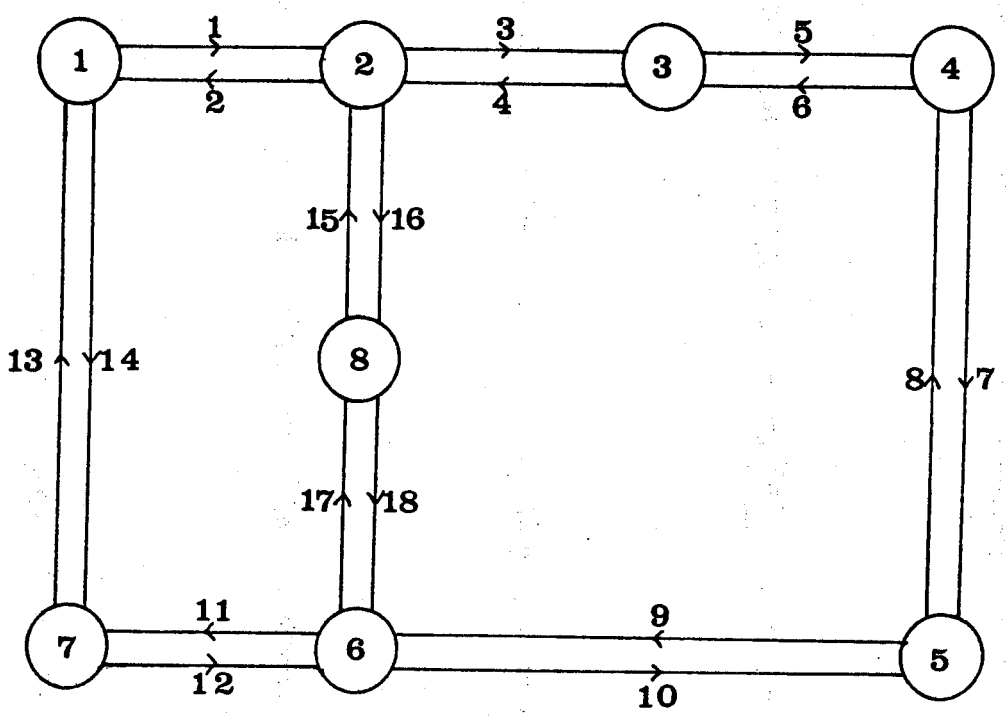
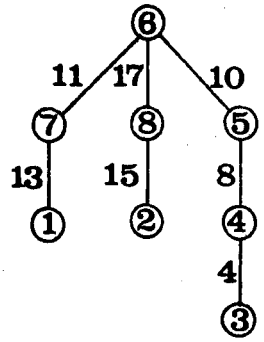
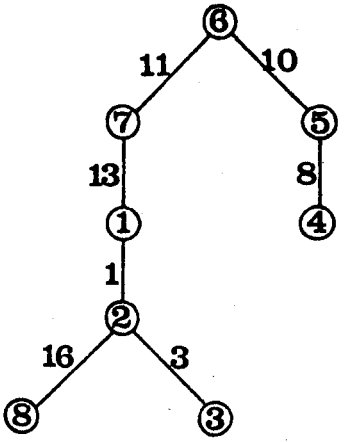
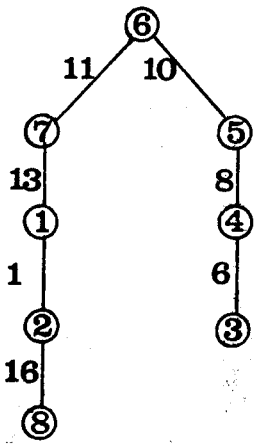
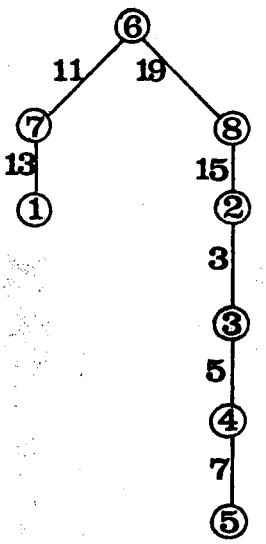


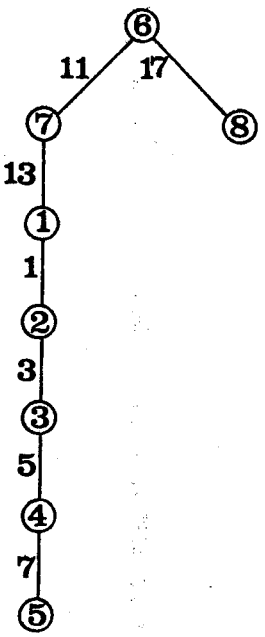
Figure 5





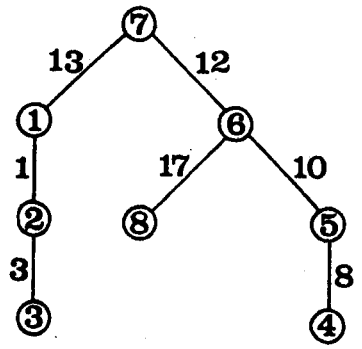


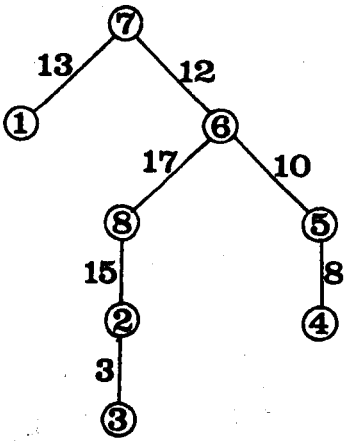


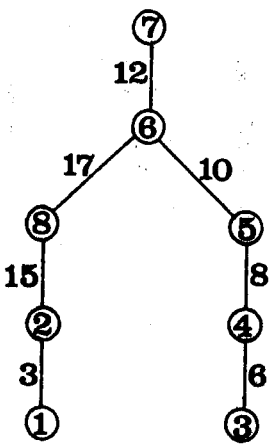


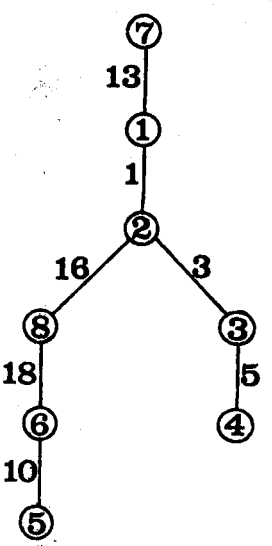
Spanning Trees for Node 6

Figure 6









Spanning Trees for Node 7

Figure 7

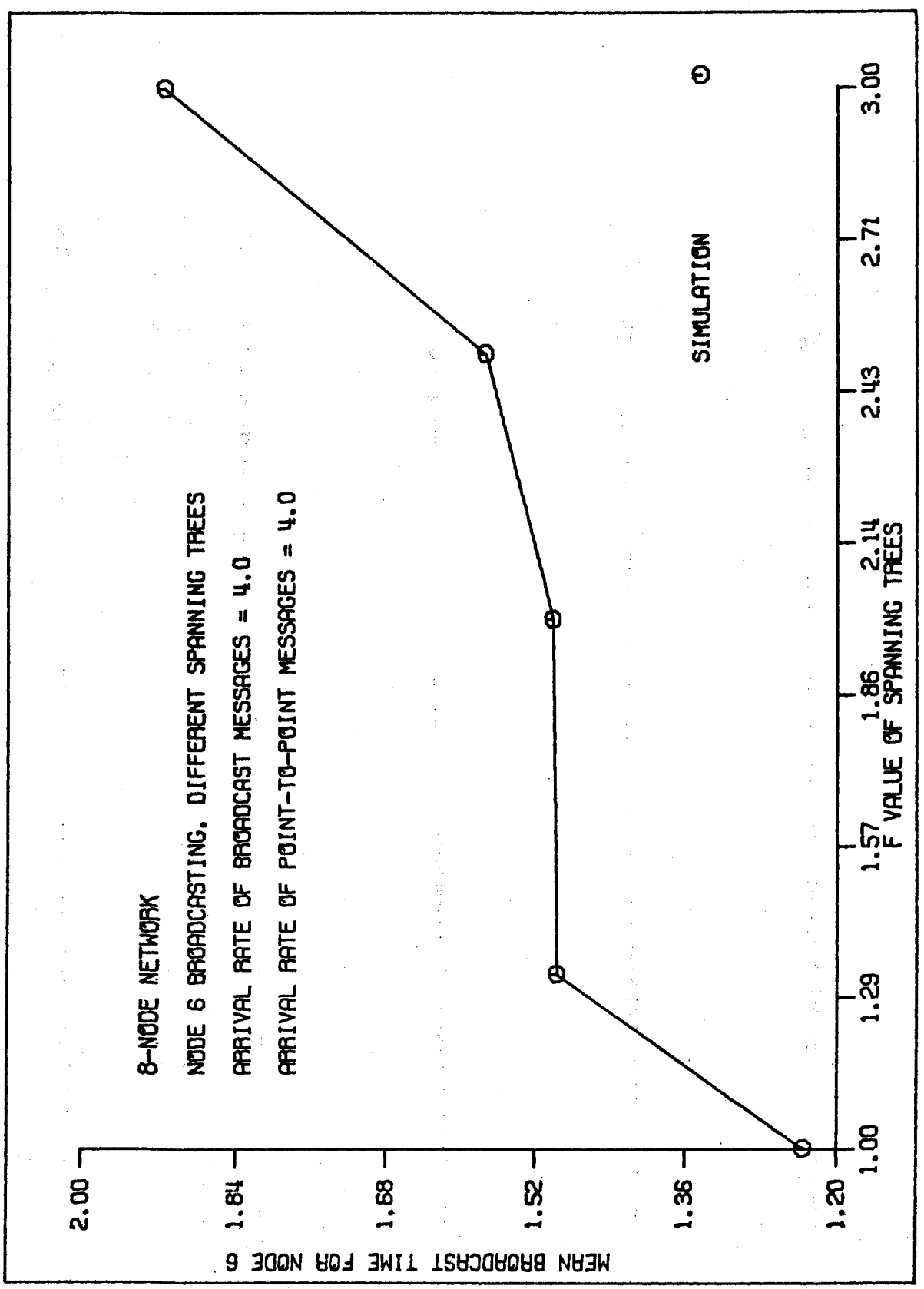

8-NODE NETWORK
NODE 6 BROADCASTING, DIFFERENT SPANNING TREES
ARRIVAL RATE OF BROADCAST MESSAGES = 4.0
ARRIVAL RATE OF POINT-TO-POINT MESSAGES = 4.0
SIMULATION
MEAN BROADCAST TIME FOR NODE 6
2.00
1.84
1.68
1.52
1.36
1.20
1.00
1.29
1.57
1.86
2.14
2.43
2.71
3.00
F VALUE OF SPANNING TREES

Figure 8

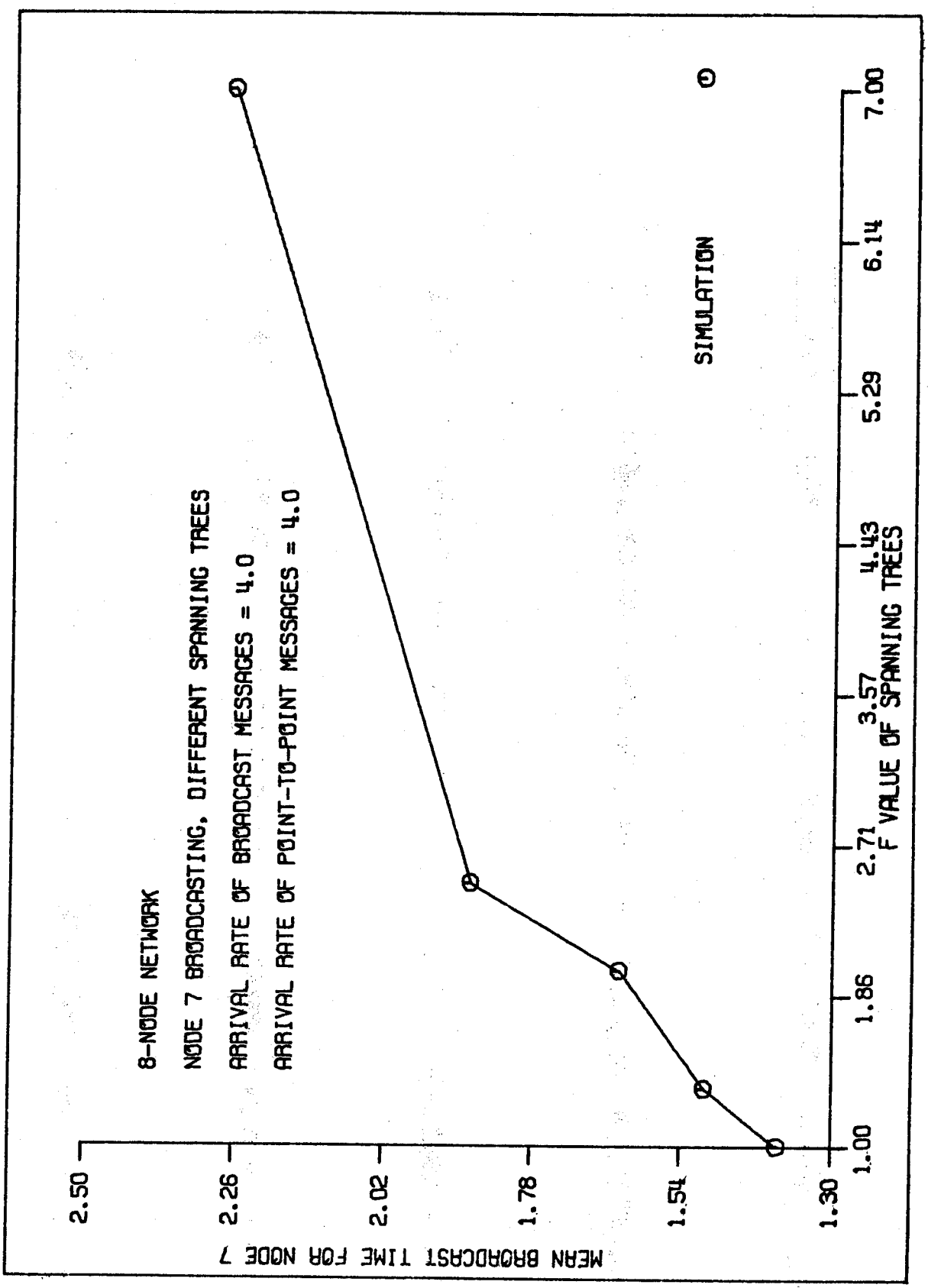

8-NODE NETWORK
NODE 7 BROADCASTING, DIFFERENT SPANNING TREES
ARRIVAL RATE OF BROADCAST MESSAGES = 4.0
ARRIVAL RATE OF POINT-TO-POINT MESSAGES = 4.0
SIMULATION
MEAN BROADCAST TIME FOR NODE 7
2.50
2.26
2.02
1.78
1.54
1.30
1.00
1.86
2.71
3.57
4.43
5.29
6.14
7.00
F VALUE OF SPANNING TREES

Figure 9

Appendix

Second Approximation Method - Special Cases

Simpler expressions for X_i and X_{ij}

1. $D = D_i$

$$X_i = \frac{n_i + 1}{\mu C_i}$$

$$X_i X_j = \sum_{r_i=0}^{n_i} \sum_{r_j=0}^{n_j} \frac{\mu C_i{}^{r_i} \mu C_j{}^{r_j} e^{-\mu C_j (D - D_j)}}{r_i!\, r_j!} Q_{ij}$$

where

$$Q_{ij} = \sum_{p=0}^{r_j} \binom{r_j}{p} (D_i - D_j)^{r_j - p} \frac{(r_i + p)!}{(\mu C_i + \mu C_j)^{r_i + p + 1}}$$

2. $D_i = D_j < D$

(This situation cannot occur for the 2-processor case, but the results will be used for the 3-server model).

$$X_i X_j = \sum_{r_i=0}^{n_i} \sum_{r_j=0}^{n_j} \frac{\mu C_i{}^{r_i} \mu C_j{}^{r_j}}{r_i!\, r_j!} e^{-(D - D_i)(\mu C_i + \mu C_j)} R_{ij}$$

where

$$R_{ij} = \sum_{p=0}^{r_i + r_j} \frac{(r_i + r_j)!\ (D - D_i)^{r_i + r_j - p}}{(r_i + r_j - p)!\ (\mu C_i + \mu C_j)^{p+1}}$$

3. $D = D_i = D_j$

$$X_i X_j = \sum_{r_i=0}^{n_i} \sum_{r_j=0}^{n_j} \frac{(r_i+r_j)!}{r_i!\, r_j!} \frac{\mu C_i{}^{r_i} \mu C_j{}^{r_j}}{(\mu C_i+\mu C_j)^{r_i+r_j+1}}$$

Simpler expressions for X_{ijk}

1. $D_i = D_j > D_k$

$$X_i X_j X_k = \sum_{r_i=0}^{n_i} \sum_{r_j=0}^{n_j} \sum_{r_k=0}^{n_k} \frac{\mu C_i{}^{r_i} \mu C_j{}^{r_j} \mu C_k^{r_k}}{r_i!\, r_j!\, r_k!} e^{-\mu C_k (D_i - D_k)} F_{ijk}$$

where

$$F_{ijk} = \sum_{q=0}^{r_k} \binom{r_k}{q} (D_i - D_k)^{r_k - q} \frac{(r_i+r_j+q)!}{(\mu C_i+\mu C_j+\mu C_k)^{r_i+r_k+q+1}}$$

2. $D_i = D_j = D_k$

$$X_i X_j X_k = \sum_{r_i=0}^{n_i} \sum_{r_j=0}^{n_j} \sum_{r_k=0}^{n_k} \frac{\mu C_i{}^{r_i} \mu C_j{}^{r_j} \mu C_k{}^{r_k}}{r_i!\, r_j!\, r_k!} \frac{(r_i+r_j+r_k)!}{(\mu C_i+\mu C_j+\mu C_k)^{r_i+r_j+r_k+1}}$$

Acknowledgement

This work was supported by the Natural Sciences and Engineering Research Council of Canada.

References

[1] Dalal, Y.K. and Metcalfe, R.M.; "Reverse path forwarding of broadcast packets", Communications of the ACM, December 1978.

[2] Cosell, B.P., Johnson, P.R., Malman, J.H., Schantz, R.E., Sussman, J., Thomas, R.H. and Walden, D.C.; "An operational system for computer resource sharing", Proceedings of Fifth Symposium on Operating Systems Principles, November 1975.

[3] Rosenkratz, D.J., Stearns, R.E. and Lewis II, P.M.; "System level concurrency control for distributed database systems", ACM Transactions on Database Systems, Vol. 3, No. 2, June 1978.

[4] Thomas, R.H.; "A majority consensus approach to concurrency control for multiple copy databases", ACM Transactions on Database Systems, Vol. 4, No. 2, June 1979.

[5] Gray, J.N.; "Notes on database operating systems", Operating Systems - An Advanced Course, Bayer, R., Graham, R.M. and Seegmuller, G., Eds., Springer Verlag, 1978.

[6] Bernstein, P.A., Rothnie, J.B., Goodman, N. and Papadimitriou, C.A.; "The concurrency control mechanism of SDD-1 : A system for distributed databases (The fully redundant case)", IEEE Transactions on Software Engineering, Vol. SE-4, No. 3, May 1978.

[7] Stonebraker, M.; "Concurrency control and consistency of multiple copies of data in distributed Ingres", IEEE Transactions on Software Engineering, Vol. SE-5, No. 3, May 1979.

[8] Kleinrock, L.; "Queueing Systems, Volume II", John Wiley and Sons, 1976.

[9] Baskett, F., Chandy, K.M., Muntz, R.R. and Palacios, F.G.; "Open, closed and mixed networks of queues with different classes of customers", Journal of the Association for Computing Machinery, Vol. 22, No. 2, April 1975.

[10] Wong, J.W.; "Distribution of end-to-end delay in message-switched networks", Computer Networks, Volume 2, No. 1, February 1978.

[11] Papoulis, A.; "Probability, Random Variables, and Stochastic Processes", McGraw-Hill Book Company, 1965.

[12] George, A., Liu, J. and Ng, E.; "User guide for SPARSPAK : Waterloo Sparse Linear Equations Package", University of Waterloo Technical Report CS-78-30.

[13] Kleinrock, L.; "Queueing Systems, Volume I", John Wiley and Sons, 1975.