Performance Analysis of k-Fold Multicast Networks

Zhenghao Zhang and Yuanyuan Yang, *Senior Member, IEEE*

***Abstract*—Multicast involves transmitting information from a single source to multiple destinations, and is an important operation in high-performance networks. A k-fold multicast network was recently proposed as a cost-effective solution to providing better quality-of-service functions in supporting real-world multicast applications. To give a quantitative basis for network designers to determine the suitable value of system parameter k under different traffic loads, in this paper, we propose an analytical model for the performance of k-fold multicast networks under Poisson traffic. We first give the stationary distribution of network states, and then derive the throughput and blocking probability of the network. We also conduct extensive simulations to validate the analytical model, and the results show that the analytical model is very accurate under the assumptions made. The analytical and simulation results reveal that by increasing the fold of the network, network throughput increases very fast when the fanouts of multicast connections are relatively small, compared with the network size.**

***Index Terms*—Blocking probability, Markov process, multicast communication, performance analysis, quality of service (QoS), switching networks, throughput.**

I. Introduction and Previous Work

MULTICAST involves transmitting information from a single source to multiple destinations, and is an important operation in high-performance networks. Multicast will be increasingly used to support various interactive applications, such as multimedia, teleconferencing, web servers and electronic commerce on the Internet, as well as communication-intensive applications in parallel and distributed computing systems, such as distributed database updates and cache coherence protocols. Many of these applications require not only multicast capability, but also predictable communication performance, called quality of service (QoS). The combination of the nonuniform nature of multicast traffic and the requirement of QoS guarantees makes the problem very challenging.

There has been much work in the literature on multicast communication in various networks, see, for example, [3]–[13]. Several researchers [3]–[7] have considered supporting *multicast assignments* in switch-based networks (or switching networks) in a nonblocking or rearrangeable fashion. A multicast assignment is a mapping from a subset of network source nodes to a subset of network destination nodes, with no overlapping allowed among the destinations of different sources. However, in reality, there are usually many instances of various multicast applications running on the same network. In this case, what we can observe at the physical layer of the network is that many independent multicast connections are being routed through the switching network, which leads to the situation that the combined multicast traffic in the network is not necessarily a multicast assignment, and overlapping among destinations of different multicast connections is quite possible. A simple example is that a destination node may be simultaneously involved in two multicast connections. Such connections will be blocked in a network which is designed to be nonblocking or rearrangeable for only multicast assignments.

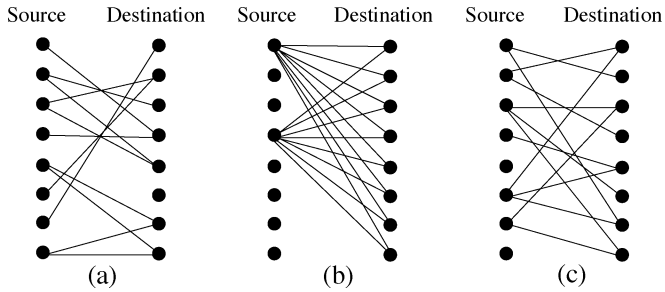


Fig. 1. Examples of twofold multicast assignments in an eight-node network.

To overcome this problem, recently [13] presented a design for a nonblocking k-fold multicast network, which can provide better QoS functions for arbitrary multicast communication. Specifically, the network can realize multiple, say, k multicast assignments in a single pass with a guaranteed latency. A *k-fold multicast assignment* is defined as a mapping from a subset of network source nodes to a subset of network destination nodes, with up to k-fold overlapping allowed among the destinations of different sources. In other words, any destination node can be involved in multicast connections, from up to k different sources at a time. Fig. 1 gives several examples of twofold multicast assignments in an eight-node network. A network which can realize any k-fold multicast assignments in a single pass is referred to as a *k-fold multicast network*. Clearly, an ordinary multicast network is a onefold multicast network. Here, k is an adjustable parameter in network design, and an appropriate value of k should be determined by the multicast traffic in the target multicast applications, especially by the statistics of destination overlapping in multicast connections. Note that although k-fold multicast assignments can be realized by simply stacking k copies of onefold networks together, the k-fold network designed in [13] has a much lower hardware cost. In fact, the cost of the former is about 3–k times of a k-fold network for any k. Thus, a k-fold network is a cost-effective choice to provide better QoS functions in supporting arbitrary multicast communication.

Paper approved by T. T. Lee, the Editor for Wireless Communication Theory of the IEEE Communications Society. Manuscript received March 10, 2003; revised April 28, 2004. This work was supported in part by the U.S. National Science Foundation under Grants CCR-0073085, CCR-0207999, and ECS-0427345.

The authors are with the Department of Electrical and Computer Engineering, State University of New York, Stony Brook, NY 11794 USA (e-mail: zhhzhang@ece.sunysb.edu; yang@ece.sunysb.edu).

Digital Object Identifier 10.1109/TCOMM.2004.841983

To provide a quantitative basis for network designers to choose the suitable value of system parameter k under different traffic loads, in this paper, we propose an analytical model for analyzing the performance of a k-fold multicast network. Although there has been a lot of research on performance analysis in various networks under multicast traffic, see, for example, [8]–[12], none of them has considered network performance in terms of k-fold multicast assignments. In this paper, we derive the *throughput* and the *blocking probability* of a k-fold multicast network under Poisson traffic and validate the model through simulations. Based on the assumptions we make, we first show that the number of ongoing multicast connections in the network is a continuous-time Markov chain. The network throughput and blocking probability can then be obtained in terms of the stationary distribution of the Markov chain. We also conduct extensive simulations to validate the analytical results.

The rest of the paper is organized as follows. Section II describes some definitions and assumptions used in the paper. Section III derives the stationary distribution of network states. Section IV gives the throughput and blocking probability of a k-fold multicast network. Section V discusses some possible generalizations of the results. Section VI shows the simulation results along with analytical results and gives some observations. Finally, Section VII concludes the paper.

II. Preliminaries

In this section, we give the definitions and assumptions we will use in this paper.

First, we will need following definitions.

Definition 1: A multicast connection refers to a source node being connected to multiple destination nodes simultaneously in the network, and sending the same message to these destination nodes. In this paper, we sometimes simply refer to it as a connection.

Definition 2: We assume that there are no buffers at source nodes. Input blocking refers to the case that a multicast connection request arrives at a busy source node and is blocked.

Definition 3: If a destination node is the destination of m source nodes, we say that this destination node is of degree m.

Definition 4: Output blocking refers to the case that a multicast connection request arrives at an idle source node but is blocked because at least one of its destination nodes is already of degree k.

Definition 5: We define output blocking ratio as the ratio of the requests blocked due to output blocking over all the requests blocked.

Definition 6: If a group of multicast connections can be transmitted simultaneously through the network without any blocking, we say that they are mutually compatible and abbreviate it as $m.c.$ Clearly, in a k-fold multicast network, only those multicast connections that can fit into a k-fold multicast assignment are $m.c.$

Definition 7: We define the average number of successful multicast connection requests carried by the network in a unit time as carried throughput, or simply throughput.

In addition, throughout this paper, we make following assumptions on the multicast traffic we consider.

- The probability of a destination node being involved in an incoming multicast connection request is θ and is independent of other destination nodes.
- Multicast connection requests at different source nodes are independent of each other.
- Holding time of each multicast connection is exponentially distributed with parameter μ and is independent of each other.
- Multicast connection requests arrive at each source node according to a Poisson process with intensity λ and are independent of each other.

III. Stationary Distribution of Network States

In this section, we will find the stationary distribution of the network, by which we can obtain network throughput and blocking probability.

First, if there are i multicast connection requests, let $p_{\deg}(i,m)$ be probability that a destination node is the destination of exactly m of the multicast connection requests, or, is of degree m under these i multicast connection requests. The probability that any multicast connection request chooses this destination node is θ and is independent of other multicast connections. Thus, we have

$$p_{\deg}(i,m) = \binom{i}{m}\theta^m(1-\theta)^{i-m}, \quad m \in \{0,1,\ldots,i\} \quad (1)$$

which is a binomial random variable. Under the assumptions, each destination node has the same distribution given by (1). Furthermore, from the assumptions, whether a destination node is chosen by a multicast connection is independent of other destination nodes. Thus, in addition to having the same distributions, the degrees of the destination nodes are also independent of each other. In other words, they are a group of independent, identically distributed (i.i.d.) random variables.

Let $P_{\mathrm{mc}}(i)$ be the probability that i multicast connection requests are $m.c.$ in a k-fold multicast network. Recall that a set of multicast connection requests are $m.c.$ when none of the destination nodes has a degree more than k when realized simultaneously in the network. From (1), we know that the probability of a destination node having a degree less than or equal to k is $\sum_{m=0}^{k} p_{\deg}(i,m)$ for $i > k$, and 1 for $i \le k$, because when $i \le k$, no destination node can have a degree more than k. Since the degrees of destination nodes are independent of each other, we have

$$P_{\mathrm{mc}}(i) = \begin{cases} \left(\sum_{m=0}^{k} p_{\deg}(i,m)\right)^N, & i > k \\ 1, & \text{otherwise.} \end{cases} \quad (2)$$

Suppose a new multicast connection request arrives when there are i multicast connections in the network. If this new connection can be realized along with those ongoing connections, we say it can "join" them. Let $P_{\mathrm{jn}}(i)$ be the probability that a new multicast connection can join i ongoing connections. We have

$$P_{\mathrm{jn}}(i) = \frac{P_{\mathrm{mc}}(i+1)}{P_{\mathrm{mc}}(i)}. \quad (3)$$

To see this, let E_1 be the event that a new multicast connection request and the existing i multicast connections are $m.c$, E_2 be

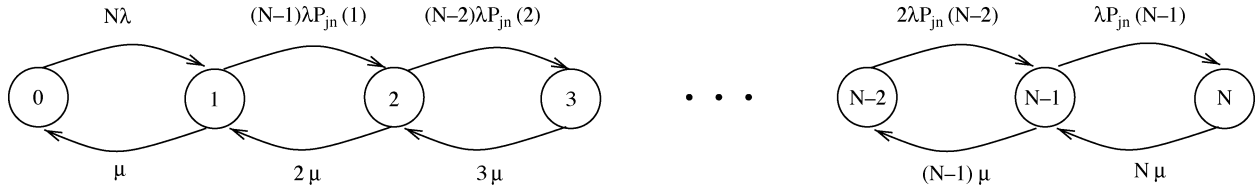


Fig. 2. State transition diagram of a k-fold multicast network.

the event that i multicast connections are $m.c$, and E_3 be the event that $i+1$ multicast connections are $m.c$. We have

$$P_{\rm jn}(i) = P(E_1|E_2) = \frac{P(E_1,E_2)}{P(E_2)} = \frac{P(E_3)}{P(E_2)} = \frac{P_{\rm mc}(i+1)}{P_{\rm mc}(i)}.$$

Suppose the network is carrying i multicast connections. Since the arrival process is Poisson and the holding time follows the exponential distribution, and the probability that a newly arrived connection request can be accepted can be uniquely determined by the number of currently ongoing connections, the number of ongoing connections in the network is a continuous-time Markov chain. The state of this Markov chain may change when some connection requests arrive and can join the ongoing connections, or when some ongoing connections are terminated. To be specific, state i may transit to state $i+1$ when a new connection request arrives at an idle source node and can join the i ongoing connections, or may transit to state $i-1$ when one of the ongoing connections terminates.

We now derive the state transition rate. First, consider the transition from state i to state $i+1$. We know that in state i, there are $N-i$ idle source nodes in the network, and under the fourth assumption in the previous section, the arrival processes at these nodes are Poisson with intensity λ and independent of each other. Hence, the combined arrival process is also Poisson and with intensity $(N-i)\lambda$. A connection request is accepted with probability $P_{\rm jn}(i)$ and is rejected with probability $1-P_{\rm jn}(i)$. Thus, the combined Poisson process is then randomly split into two random processes: one is the process of the arrivals that can be successfully realized in the network, and another is the process of the arrivals that are blocked. Since a split Poisson process is still Poisson, the arrival process of the connection requests that can join the i ongoing connections is Poisson with intensity $(N-i)\lambda P_{\rm jn}(i)$, which is the transition rate from state i to state $i+1$. Now consider the transition from state i to state $i-1$. In state i, there are i ongoing multicast connections, whose holding times are exponentially distributed with parameter μ and independent of each other. Thus, the transition rate from state i to state $i-1$ is $i\mu$. A complete state transition diagram, including the boundary conditions for $i=0$ and $i=N$, is shown in Fig. 2. Clearly, this is a birth–death process. Letting π_i, $i=0,1,\ldots,N$, be the stationary distribution, we have

$$\pi_{i+1}(i+1)\mu = \pi_i(N-i)\lambda P_{\rm jn}(i), \quad i\in\{0,1,\ldots,N-1\}.$$

Let $\rho=\lambda/\mu$

$$\begin{aligned}\pi_i &= \pi_0\frac{N(N-1)\cdots(N-i+1)}{i!}\rho^i\prod_{j=0}^{i-1}P_{\rm jn}(j)\\ &= \pi_0\binom{N}{i}\rho^i P_{\rm mc}(i), \quad i\in\{0,1,\ldots,N\} \qquad (4)\end{aligned}$$

where π_0 is the probability that the Markov chain is in state 0, which can be determined by

$$\pi_0 = \frac{1}{\sum_{i=0}^{N}\binom{N}{i}\rho^i P_{\rm mc}(i)}.$$

Note that by the discussions in this section, we are actually using one single variable, the number of ongoing connections, to represent the state of the whole network. The reason for this is that we are only interested in the probability of a new arriving request being blocked, and as explained earlier in this section, this probability can be determined by the number of ongoing connections. There are, of course, other ways to model the network. The most accurate one is to let each possible connection pattern be a state. However, this will be highly unnecessary, and can be shown to be equivalent to using the number of ongoing connections as follows. First, note that much information about the connection patterns is redundant for the purpose of determining the blocking probability of a new connection request. This is because to find the blocking probability, only the degrees of the outputs are needed. Connection patterns that result in the same degrees of the outputs will have exactly the same effect on the new connection request, and therefore, can be merged into the same state. The network state can then be reduced to a vector, which can be called the "degree vector," with each element representing the degree of an output port. Furthermore, based on the assumption on the multicast connections, the probability that the network is in a state represented by a particular degree vector can be uniquely determined by the number of active input ports. Therefore, the state of the network can be further reduced to the number of ongoing connections.

IV. Throughput and Blocking Probability

Given the stationary distribution of network states, in this section, we derive the throughput and blocking probability of a k-fold multicast network. We consider a long time period $[0,T]$, such that the values under consideration have converged to their average values. First we have the following lemma.

Lemma 1: For a long time period $[0,T]$, the total number of successful multicast connection requests carried by the network during $[0,T]$ is given by

$$N_{\rm succ} = T\lambda\sum_{i=0}^{N}\pi_i(N-i)P_{\rm jn}(i). \qquad (5)$$

Proof: The average dwelling time of state i is the inversion of the rate that the network departs from state i

$$\frac{1}{(N-i)\lambda P_{\rm jn}(i)+i\mu}.$$

It holds for all $i\in\{0,1,\ldots,N\}$. Since the total time the system spent in state i during $[0,T]$ is $\pi_i T$, the number of visits to state

i during $[0,T]$ is $T\pi_i((N-i)\lambda P_{\rm jn}(i)+i\mu)$. Notice that the number of visits is also the number of times the network departs from state i. A departure may be caused by the arrival of a successful connection request or the termination of an ongoing connection. The first case has the probability

$$\frac{(N-i)\lambda P_{\rm jn}(i)}{(N-i)\lambda P_{\rm jn}(i)+i\mu}.$$

Thus, the total number of times the network departs from state i due to the arrival of a successful connection request is $T\pi_i(N-i)\lambda P_{\rm jn}(i)$. This is also the total number of successful connection requests arriving at the network when the network is in state $i(i\in\{0,1,\ldots,N\})$ during $[0,T]$. Therefore, the total number of successful connection requests carried by the network during $[0,T]$ is obtained by summing over i

$$N_{\rm succ}=T\lambda\sum_{i=0}^{N}\pi_i(N-i)P_{\rm jn}(i).$$

■

Proposition 1: The *throughput* of a k-fold multicast network is given by

$$TH=\lambda\sum_{i=0}^{N}\pi_i(N-i)P_{\rm jn}(i) \tag{6}$$

which can be directly obtained from *Lemma 1*.

Note that the total number of connection requests arriving at the network during $[0,T]$ is $N_{\rm total}=N\lambda T$, among which only $N_{\rm succ}$ connection requests given in *Lemma 1* are successful, and the rest

$$N_{\rm bl}=N_{\rm total}-N_{\rm succ}=T\lambda\sum_{i=0}^{N}\pi_i\left(N\left(1-P_{\rm jn}(i)\right)+iP_{\rm jn}(i)\right)$$

connection requests are blocked. The blocking probability is $N_{\rm bl}/N_{\rm total}$. Thus, we have the following proposition.

Proposition 2: The blocking probability of a k-fold multicast network is

$$PB=\frac{1}{N}\sum_{i=0}^{N}\pi_i\left(N\left(1-P_{\rm jn}(i)\right)+iP_{\rm jn}(i)\right). \tag{7}$$

To further study the performance of a k-fold multicast network, we consider input blocking and output blocking separately. Input blocking depends on the number of busy source nodes in the network. Since a larger fold means more busy source nodes, increasing k will generally increase the probability of input blocking. Input blocking can be reduced only by adding buffers at each source node, which is not considered in this paper. On the other hand, increasing k will reduce the chance of output blocking. In order to calculate the output blocking probability, first we have the following lemma.

Lemma 2: Let $p_{\rm bl}(i,m)$ be the probability that exactly m requests arriving at idle source nodes are blocked during a visit to state i. We have

$$p_{\rm bl}(i,m)=\alpha(1-\alpha)^m \tag{8}$$

where

$$\alpha=\frac{(N-i)\lambda P_{\rm jn}(i)+i\mu}{(N-i)\lambda+i\mu}.$$

Proof: Define the following events.

- E_1: the first m connection requests arrive before any of the i ongoing connections terminates.
- E_2: none of the m connection requests can be realized along with the i ongoing connections.
- E_3: one of the ongoing connections terminates after the arrival of the mth request, and before the arrival of the $(m+1)$th request.
- E_4: the $(m+1)$th request arrives before any ongoing connection terminates, and is $m.c.$ with the i ongoing connections.

We have

$$p_{\rm bl}(i,m)=P(E_1\cap E_2)P(E_3\cup E_4|E_1\cap E_2).$$

Events E_1 and E_2 are independent of each other, because knowing the arrival time of a connection request does not provide any information on whether it can join the i ongoing connections. Thus, $P(E_1\cap E_2)=P(E_1)P(E_2)$. Since events E_3 and E_4 cannot occur at the same time

$$P(E_3\cup E_4|E_1\cap E_2)=P(E_3|E_1\cap E_2)+P(E_4|E_1\cap E_2).$$

Thus

$$p_{\rm bl}(i,m)=P(E_1)P(E_2)\left(P(E_3|E_1\cap E_2)+P(E_4|E_1\cap E_2)\right).$$

Event E_1 occurs when the summation of m independent exponentially distributed random variables with parameter $(N-i)\lambda$ is less than another exponentially distributed random variable with parameter $i\mu$. Thus

$$P(E_1)=\left(\frac{(N-i)\lambda}{(N-i)\lambda+i\mu}\right)^m.$$

Since connection requests are independent of each other, we have $P(E_2)=(1-P_{\rm jn}(i))^m$. Due to the memoryless property of exponentially distributed random variables, after the arrival of the mth blocked request, the system starts over again. Thus, $P(E_3|E_1\cap E_2)$ is simply the probability that an exponentially distributed random variable with parameter $i\mu$ is less than another exponentially distributed random variable with parameter $(N-i)\lambda$. Therefore

$$P(E_3|E_1\cap E_2)=\frac{i\mu}{(N-i)\lambda+i\mu}.$$

Similarly, we have

$$P(E_4|E_1\cap E_2)=\frac{(N-i)\lambda}{(N-i)\lambda+i\mu}P_{\rm jn}(i).$$

Define

$$\alpha=\frac{(N-i)\lambda P_{\rm jn}(i)+i\mu}{(N-i)\lambda+i\mu}.$$

We have $p_{\rm bl}(i,m)=\alpha(1-\alpha)^m$. ■

From this lemma, we know that the number of the requests blocked due to output blocking during a visit to state i is a geometric distribution random variable with parameter α. It has mean

$$\frac{1-\alpha}{\alpha} = \frac{(N-i)\lambda\,(1-P_{\rm jn}(i))}{(N-i)\lambda P_{\rm jn}(i)+i\mu}. \tag{9}$$

From the proof of *Lemma 1*, the number of visits to state i during a long time period $[0,T]$ is $T\pi_i((N-i)\lambda P_{\rm jn}(i)+i\mu)$. Thus, the average total number of requests blocked due to output blocking when the network is in state i during $[0,T]$ is $T\pi_i(N-i)\lambda(1-P_{jn}(i))$. Then the total number of requests blocked by output blocking during $[0,T]$ is

$$N_{\rm outbl} = T\lambda\sum_{i=0}^{N}\pi_i(N-i)\,(1-P_{\rm jn}(i))\,.$$

Thus, the total number of requests blocked due to input blocking during $[0,T]$ is the total number of the requests blocked minus $N_{\rm outbl}$

$$N_{\rm inbl} = N_{\rm bl} - N_{\rm outbl} = T\lambda\sum_{i=0}^{N}\pi_i i.$$

Proposition 3: The output blocking ratio, which is defined as the ratio of blocked requests due to output blocking over all blocked requests, is

$$\frac{\sum_{i=0}^{N}\pi_i(N-i)\,(1-P_{\rm jn}(i))}{\sum_{i=0}^{N}\pi_i\,(N\,(1-P_{\rm jn}(i))+iP_{\rm jn}(i))\,.} \tag{10}$$

V. Some Generalizations

In this section, we discuss some possible generalizations of the results obtained in previous sections.

First, although we have mainly focused on an $N\times N$ k-fold multicast network, the results can be easily extended to an asymmetrical k-fold multicast network with N source nodes and M destination nodes. The only modification we need to make is (2). For an $N\times M$ k-fold multicast network, the probability that i multicast connection requests are $m.c.$ is

$$P_{\rm mc}(i) = \begin{cases} \left(\sum_{m=0}^{k} p_{\rm deg}(i,m)\right)^{M}, & i>k \\ 1, & \text{otherwise.} \end{cases} \tag{11}$$

All other results and discussions still hold.

This result can be directly applied to wavelength-division multiplexing (WDM) optical multicast networks [15], where the inputs and outputs are optical fibers, with k wavelengths on each fiber. An $N\times N$ WDM multicast network, with each input fiber equipped with a *full-range wavelength converter*, which converts a wavelength to any of the k wavelengths in the network, can be modeled as an asymmetrical $Nk\times N$ k-fold multicast network. The throughput and blocking probability of this WDM multicast network can then be obtained by using the results in this paper.

Second, note that by our assumptions, the fanout of multicast connection is a random variable that follows binomial distribution. If the fanout follows other types of distributions, for example, geometric distribution, the only thing we need to do is to recalculate $P_{\rm mc}(i)$. However, in this case, the degrees of output nodes will no longer be independent of each other. When the network size is large, finding $P_{\rm mc}(i)$ analytically becomes impractical. Thus, we have to generate $P_{\rm mc}(i)$ by simulations. After obtaining $P_{\rm mc}(i)$, the throughput and blocking probability can be obtained immediately from our analytical model.

VI. Simulations and Observations

We have conducted extensive simulations for several network sizes under different fanouts and arrival rates to validate our analytical results. In the simulation, we use F to represent the fanout of a multicast connection request. F is a binomial random variable with parameter (N,θ) and mean $E(F)=N\theta$. Without loss of generality, we let service rate $\mu=1$. The time interval between two consecutive arrivals at an input node is a random variable that follows exponential distribution with parameter λ, and is independent of network state.

Initially, all input nodes are set to be idle, and the degrees of all output nodes are set to be zero. The system time is set to be zero. The program runs in an event-driven mode: it looks for the first event that is about to occur, either an arrival or a departure. If it is an arrival, a random variable that follows exponential distribution with parameter λ is generated as the time interval until next arrival. If the connection request arrives at a busy input node, it is immediately rejected. For a connection request arriving at an idle input node, its destinations are generated and are represented by a $1\times N$ binary vector. If one of its destination nodes is found to have degree k already, the connection request is rejected. Otherwise, the connection request is accepted, and the degrees of all its destination nodes are incremented by one. The input node is then set to state "busy." A random variable that follows exponential distribution with parameter μ is generated as the holding time of this connection. If the event is a departure, the degrees of all its destination nodes are decremented by one, and the input node is set to state "idle." After that, system time is set to the time when the event occurs. The program then looks for the next nearest event. After the system time exceeds a certain value T, the simulation terminates. For arrival rate $\lambda=0.5$ and network size $N=32$, if $T=4000$, there are about 64 000 arrivals. In our simulation, we ensure that there are at least 50 000 arrivals.

Due to limited space, we only illustrate the simulation results along with the analytical results for network size 32. In Fig. 3(a) and (b), we plot network throughput as a function of network fold k. In Fig. 3(a), we keep arrival rates at 0.5 and vary the fanout distribution. First, let's look at the growth of the throughput with respect to k for different values of mean fanout $E(F)$. We observe that when $E(F)$ is relatively small, the throughput grows indeed very fast, but when $E(F)$ becomes larger, throughput grows much more slowly. Next, we can see that for any $E(F)$, the throughput grows more rapidly when k is small, and tends to converge to some value when k further increases. For example, in Fig. 3(a), for the average fanout $E(F)=3.2$, the throughput increases by 75% when network fold k increases from one to two, but when $k>5$, it stops increasing and remains at about 10.6. This is because when k becomes very large, almost every connection request arriving at an idle source node can go through. Thus, the network throughput

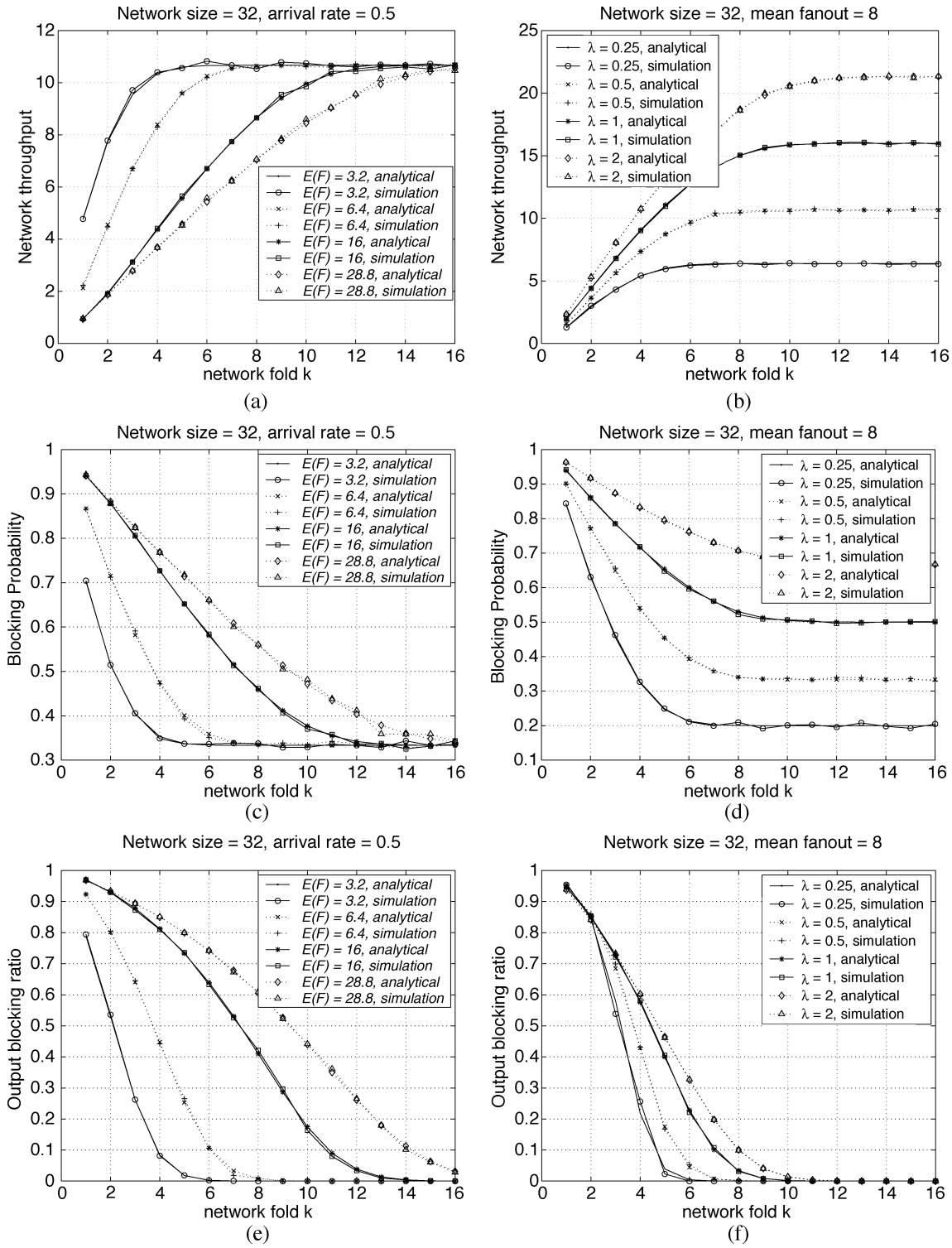


Fig. 3. Simulation and analytical results of a k-fold network for network size 32. (a) Network throughput under different fanout distributions. (b) Network throughput under different arrival rates. (c) Blocking probability under different fanout distributions. (d) Blocking probability under different arrival rates. (e) Output blocking ratio under different fanout distributions. (f) Output blocking ratio under different arrival rates.

is totally determined by the arrival rate. In this case, further increasing k will not increase the throughput.

In addition, the value of k to achieve the maximum throughput depends on the fanout distribution. In Fig. 3(a), we can see that to achieve the maximum throughput, we need to let $k = 5$ for $E(F) = 3.2$, $k = 8$ for $E(F) = 6.4$, $k = 13$ for $E(F) = 16$, and $k = 16$ for $E(F) = 28.8$, respectively. We can see that the increase is not linear to $E(F)$, because the larger the $E(F)$, the less k we need to add to the network to achieve the maximum throughput.

In Fig. 3(b), we fix the fanout distribution and study network performance under different arrival rates. Similar observations can be drawn for $\lambda < 1$. We also did experiments on $\lambda > 1$, which means arriving is faster than serving. In most networks,

this would lead to an unstable state and is not considered. However, since we assume no buffers at source nodes and any connection request arriving at a busy source node is immediately dropped, it is still valid in our case. As we can see, a larger λ leads to larger throughput, but needs a larger k to reach the maximum throughput.

In Figs. 3(c) and (d), we plot the network blocking probability as a function of network fold k. In Fig. 3(c), we keep arrival rates at 0.5 and vary the fanout distribution. First, we notice that when the average fanout $E(F)$ is relatively small, blocking probability decreases very fast when k increases. For example, in Fig. 3(c) where $N = 32$, for $E(F) = 3.2$, blocking probability drops by 30% when k increases from one to two. However, when $E(F)$ becomes larger, blocking probability decreases much more slowly and almost linearly to k. Next, we can see that blocking probability does not reach zero when k further increases. This is because of the presence of input blocking, which actually increases with k. In Fig. 3(d), we fix the fanout distribution and study network performance under different arrival rates. Again, we can see that blocking probability decreases quickly when arrival rates are small (say, less than 0.5) and much more slowly when arrival rates become large.

In Figs. 3(e) and (f), we plot the output blocking ratio as a function of network fold k. We can see that when k is larger than a certain value, the output blocking ratio tends to zero, which means no connection request is blocked due to output blocking. As in Fig. 3(e), where $N = 32$ and $\lambda = 0.5$, to make the output blocking ratio almost zero, we need only to let $k = 6$ for $E(F) = 3.2$, $k = 9$ for $E(F) = 6.4$, and $k = 14$ for $E(F) = 16$, respectively. Again, the value of k to eliminate output blocking is not a linear function of $E(F)$. The larger the $E(F)$, the less fold we need to add to the network to achieve a zero output-blocking ratio.

VII. Conclusions

A k-fold multicast network was recently proposed [13] as a cost-effective solution to provide better QoS functions in supporting real-world multicast applications. To give a quantitative basis for network designers to determine the suitable value of system parameter k under different traffic loads, in this paper, we have presented an analytical model for the performance of k-fold multicast networks under Poisson traffic. We first gave the stationary distribution of network states, and then derived the throughput and blocking probability of the network. We have also conducted simulations to validate the analytical model, and the results show that the analytical model is very accurate under the assumptions we made. From the analytical and simulation results, we can see that by increasing the fold of the network, network throughput increases very fast when the fanout of multicast connections are relatively small, compared with the network size. Our future work includes generalizing the model to other traffic distributions and other types of multicast networks.

References

[1] V. E. Benes, "Heuristic remarks and mathematical problems regarding the theory of switching systems," *Bell Syst. Tech. J.*, vol. 41, pp. 1201–1247, 1962.

[2] C. Clos, "A study of nonblocking switching networks," *Bell Syst. Tech. J.*, vol. 32, pp. 406–424, 1953.

[3] F. K. Hwang and A. Jajszczyk, "On nonblocking multiconnection networks," *IEEE Trans. Commun.*, vol. COM-34, pp. 1038–1041, Sep. 1986.

[4] Y. Yang and G. M. Masson, "Nonblocking broadcast switching networks," *IEEE Trans. Comput.*, vol. 40, pp. 1005–1015, Sep. 1991.

[5] P. Feldman, J. Friedman, and N. Pippenger, "Wide-sense nonblocking networks," *SIAM J. Discr. Math.*, vol. 1, no. 2, pp. 158–173, May 1988.

[6] C. Lee and A. Y. Oruç, "Design of efficient and easily routable generalized connectors," *IEEE Trans. Commun.*, vol. 43, pp. 646–650, Feb.-Apr. 1995.

[7] Y. Yang and J. Wang, "A new self-routing multicast network," *IEEE Trans. Parallel Distrib. Syst.*, vol. 10, pp. 1299–1316, Dec. 1999.

[8] E. W. Zegura, "Evaluating blocking probability in generalized connectors," *IEEE/ACM Trans. Networking*, vol. 3, pp. 387–398, Apr. 1995.

[9] N. McKeown, A. Mekkittijul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE Trans. Commun.*, vol. 47, pp. 1260–1267, Oct. 1999.

[10] M. Andrews, S. Khanna, and K. Kumaran, "Integrated scheduling of unicast and multicast traffic in an input-queued switch," in *Proc. IEEE INFOCOM*, 1999, pp. 1144–1151.

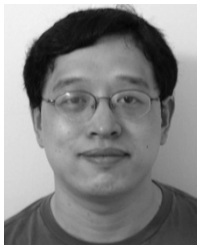
[11] Y. Yang and J. Wang, "On blocking probability of multicast networks," *IEEE Trans. Commun.*, vol. 46, pp. 957–968, Jul. 1998.

[12] Y. Yang, "The performance of multicast banyan networks," *J. Parallel Distrib. Comput.*, vol. 60, no. 8, pp. 909–923, 2000.

[13] Y. Yang and J. Wang, "Nonblocking k-fold multicast networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 14, pp. 131–141, Feb. 2003.

[14] G. Grimment and D. Stirzaker, *Probability and Random Processes*, 3rd ed. Cambridge, U.K.: Oxford Univ. Press, 2001.

[15] Y. Yang, J. Wang, and C. Qiao, "Nonblocking WDM multicast switching networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, pp. 1274–1287, Dec. 2000.



Zhenghao Zhang received the B.Eng. and M.S. degrees in electrical engineering from Zhejiang University, Hangzhou, P.R. China, in 1996 and 1999, respectively. Since 2001, he has been working toward the Ph.D. degree in the Department of Electrical and Computer Engineering, State University of New York at Stony Brook.

From 1999 to 2001, he worked in industry as a software engineer for embedded systems design. His research interest includes scheduling and performance analysis of optical networks.

Yuanyuan Yang (S'91–M'91–SM'98) received the B.Eng. and M.S. degrees in computer science and engineering from Tsinghua University, Beijing, China, and the M.S.E. and Ph.D. degrees in computer science from The Johns Hopkins University, Baltimore, MD.

She is currently a Professor of Computer Engineering and Computer Science with the State University of New York at Stony Brook. Her research interests include parallel and distributed computing and systems, high speed networks, optical and wireless networks and high performance computer architecture. She has published extensively in major journals and refereed conference proceedings, and holds six U.S. patents. She is an Editor for the *Journal of Parallel and Distributed Computing*. She has served on National Science Foundation review panels and program/organizing committees of numerous international conferences in her areas of research.

Dr. Yang is an Editor for IEEE Transactions on Parallel and Distributed Systems.