Unscheduled Multicasts in WDM Broadcast-and-Select Networks

Eytan Modiano
MIT Lincoln Laboratory, M/S C437
244 Wood St., Lexington, MA, 02173
modiano@ll.mit.edu

Abstract

This paper considers the problem of scheduling multicast transmissions in a WDM broadcast based Local Area Network. In very high speed networks optimal scheduling may be too time consuming and complex to be executed in real time, thus we are led to consider unscheduled (random) multicast transmissions. We consider two random scheduling schemes: in the first, a message is continuously retransmitted until it is received by all of its intended recipients; and in the second, a random delay is introduced between retransmissions of the same message. We develop an exact throughput analysis for both schemes using methods from discrete-time queueing systems and show that the algorithm with random delays between retransmissions results in higher throughput. Finally, we consider a number of receiver algorithms for selecting among multiple simultaneous transmissions and show that an algorithm where the receiver selects the message with the least number of intended recipients performs better than a random selection algorithm.

I. Introduction

This paper examines the problem of scheduling multicast transmissions in Wavelength Division Multiplexing (WDM) local area networks. Recently, multicasting has received a great deal of attention in the context of higher layer protocols for wide area networks [1-2]. Increasingly applications and protocols come to depend on the underlying networks for providing a multicast capability. The emergence of WDM LAN technology [5] gives rise to the multicasting problem in this environment. In order for WDM LANs to gain widespread use they must support a multicasting capability.

Most LANs use a broadcast medium which makes multicasting simple and efficient. Every transmission is received by all of the nodes in the network. In WDM broadcast LANs, such as a broadcast star, the problem is significantly more complicated. This is because in WDM the fiber supports multiple channels (each corresponding to a different wavelength). In order to listen to a transmission on a WDM channel nodes must tune their receiver to the appropriate wavelength. If each node has only a single receiver, a transmission on one channel may not be received by nodes that are listening to a transmission on another channel.

This paper is motivated by the design of a very high speed WDM LAN extension to a wideband all-optical WDM network [3,4]. This LAN will consist of about 100 nodes connected in a broadcast star topology using 32 wavelengths (channels). Each channel will operate at a transmission rate of about 10Gbps. Each node will have one fast tunable transmitter and one fast tunable receiver. The network will use a MAC protocol that allows the nodes to efficiently share the WDM channels. There are a number of such protocols that can achieve high utilization [4-7]. The proposed system is based on the protocol in [4] which uses a simple master/slave scheduler that is able to schedule transmissions efficiently and overcome the effects of propagation delays and transceiver tuning delays. A separate control channel is used to make reservations, inform nodes when and on which wavelength to transmit and notify the intended recipients of the upcoming transmissions. The details of the proposed system and protocols are outside the scope of this paper. Here we are only concerned with the problem of scheduling multicast transmissions over this system.

There are a number of previous works that address the multicast problem in a similar context [8-12]. In [9] a reservation-based protocol that schedules multicasts to avoid any receiver conflicts is described. Since the protocol of [9] requires all of the intended receivers to be available before a message can be transmitted, it results in inefficient use of receivers when the size of a multicast group is large. Consequently, in [10] an alternative reservations protocol is described that schedules multicasts to take place in multiple transmissions, thereby reducing the amount of time that available receivers have to wait before receiving the message. Finally, in [11] a scheduling algorithm is described for performing multicasts in an N x N switch. Since this problem is similar in nature to the one we consider here, this algorithm may also be applicable to WDM broadcast LANs.

While the above algorithms attempt to increase system utilization through the design of an efficient scheduler, they require the implementation of sophisticated scheduling algorithms. In a network operating at 10 Gbps per WDM channel, millions of such schedules may have to be generated every second. This enormous throughput requirement significantly limits the sophistication of the scheduling algorithm that can be employed. We are therefore driven to consider unscheduled multicasts. Aspects of random scheduling of multicast traffic have been considered in the past in the context of an NxN switch. In [12] an approximate analysis is developed for a random packet selection policy in an NxN switch, with mixed input traffic, under the assumption that all copies of a packet have to be transmitted during the same slot.

In this paper we examine the performance of unscheduled multicast transmissions in a WDM network. This work is significant in a number of ways. First, we provide an exact throughput analysis of simple, unscheduled, multicast schemes. These throughput results are not only significant for the insight

0-7803-4383-2/98/$10.00 © 1998 IEEE.

they provide into unscheduled multicasts, but they can also be used as a benchmark for comparing the performance of other multicast scheduling algorithms. These results also hold for an NxN switching system and hence they provide additional insight into the performance of a multicast switch. Second, we compare two random scheduling schemes; in one, a message is continuously retransmitted until it is received by all of its intended recipients; and in the other, a random delay is introduced between retransmissions of the same message. We show that the algorithm with random delays between retransmission results in higher throughput. Finally, we consider a number of receiver algorithms for selecting among multiple simultaneous transmissions and show that an algorithm where the receiver selects the message with the least number of intended recipients performs better than a random selection algorithm. Hence, the results of this paper have both practical value for the implementation of simple unscheduled multicast algorithms in very high speed WDM LANs, and theoretical value for providing insight into the performance of multicast switching systems.

We assume a slotted system where the size of a slot is equal to the message size. The network consists of N nodes and W wavelengths and each message is addressed (multicast) to k receivers randomly chosen from the N nodes. The unscheduled multicast protocol retransmits messages repeatedly until they are received by all of their intended receivers.

In this context, the system throughput can be expressed as the average number of multicast message transmissions completed per slot per wavelength (channel). In other words, it is the inverse of the average number of transmissions required per successful multicast. A lower bound on this number can be easily obtained by observing that, even with an optimal scheduling algorithm, during every slot each node can *at most* receive one message transmission.

The system consists of W channels simultaneously transmitting multicast messages each intended for k randomly chosen nodes. Since there are N nodes, on average each node has kW/N transmissions intended for it. Since each node can only receive one transmission at a time, $\overline{T}$, the average number of transmissions required per message, is lower bounded by

$$\overline{T} > \max(\frac{kW}{N}, 1). \qquad (1)$$

Therefore, regardless of whether or not a scheduling algorithm is used, or which algorithm is used, the system throughput in multicasts per slot per wavelength cannot exceed $1/\overline{T}$. It is interesting to note that when kW is less than N the system is channel limited; that is, there are not enough channels to keep all of the receivers busy. In the channel limited case receivers can never be fully utilized. While when kW is greater than N, the system is receiver limited; that is, the number of receivers is too small to keep all of the channels busy with new transmissions. In the receiver limited case channels can never be fully utilized and the lower bound gives the average number of required transmissions when receivers are fully utilized.

The throughput efficiency of a multicast algorithm can therefore be defined as the ratio of the lower bound from equation 1 to the average number of transmissions that the algorithm requires per successful multicast. When the system is channel limited (lower bound =1), this simply measures the throughput of the channels (the average number of completed multicasts per slot/wavelength). However, when the system is receiver limited this measure compares the channel throughput to the maximum achievable throughput. We believe that this measure gives a better indication of the efficiency of a scheduling algorithm than simply measuring channel utilization.

In this paper we consider unscheduled multicasts and compare their performance to this lower bound. We start, in Section II, by considering a protocol where a message is repeatedly transmitted until it is received by all of its intended recipients. This protocol leads to a form of Head of Line (HOL) blocking where a message waiting on a busy receiver blocks the channel from being used by other messages that are addressed to unoccupied receivers. To remedy this problem, in section III we consider a protocol which retransmits a message only after waiting a random amount of time to allow busy receivers to become unoccupied. We show that this protocol offers a throughput improvement over the one in section II. Finally, in section IV we show that an additional throughput improvement can be achieved if receivers intelligently select the transmission to which they listen.

II. Persistent protocol - throughput analysis

The simplest protocol we consider is one in which a message is repeatedly transmitted until it has been received by all of its intended recipients (all members of the multicast group). In this protocol, which we call persistent, no new multicast transmission can begin on a channel until the previous one is completed. In order to analyze the achievable throughput of this protocol we assume that the system is constantly backlogged. That is, as soon as the transmission of one message is completed, a new message is immediately available for transmission[1]. Each message is destined to exactly k receivers chosen at random from the N nodes. Hence, each packet has a probability k/N of being addressed to any given node. When a node has more than one packet addressed to it, it must choose only one of them to receive. The choice of which packet a receiver should choose is addressed in more detail in section IV. Here we make the assumption that the receiver selects the packet based on the time it was first transmitted in a First Come First Served (FCFS) order. If two or more packets were first transmitted in the same slot, then the receiver can choose

[1] Here we are not concerned with the channel assignment problem and assume that a protocol, such as the one described in [4], is used that can coordinate the transmissions so that the channels are fully utilized.

amongst them at random. Here, we use this FCFS ordering mainly because it simplifies the analysis. However, in section IV we will show through simulation that FCFS ordering results in shorter delays and increased throughput over a random selection policy.

Consider the mth time slot, and let Q_m^i be the number of messages addressed to node i at the end of the mth time slot. Since during any time slot node i can only receive one message it follows that

$$Q_m^i = \max(0, Q_{m-1}^i + A_m^i - 1), \tag{2}$$

where A_m^i is the number of new messages arriving before the start of the mth time slot and destined to node i. A new message arrives, at the beginning of a time slot, only when the transmission of a previous message had been completed at the end of the previous slot, and that new message is destined to node i with probability k/N. Hence, if we let C_{m-1} denote the number of completed multicast transmissions during the $(m\text{-}1)$th time slot then C_{m-1} also denotes the number of newly arrived messages at the beginning of the mth time slot. Now, A_m^i has the binomial distribution

$$\Pr[A_m^i = l] = \binom{C_{m-1}}{l} \left(\frac{k}{N}\right)^l \left(1 - \frac{k}{N}\right)^{C_{m-1}-l}, \tag{3}$$

and $\overline{A}$, the average number of new arrivals at a node (per slot) is equal to $\overline{C}k/N$, where $\overline{C}$ is the average number of completed multicast transmissions per slot. It can be shown [13] that, for finite values of k, as W (the number of channels) approaches infinity A_m^i becomes Poisson of rate $\overline{A}$. Also, noticing that equation (2) is the same as that of an M/D/1 queue [14], we can express $\overline{Q}$, the average number of messages destined to a receiver, according to the well known M/D/1 formula [14]

$$\overline{Q} = \overline{A} + \frac{(\overline{A})^2}{2(1-\overline{A})}.$$

Now let's turn our attention to the computation of $\overline{C}$, the average number of completed multicast transmissions per slot. We arrive at this number by first computing the average number of transmissions (slots) required per successful multicast message. For a multicast message to be successful, it must be received by all k nodes for which it was intended. Consider a message destined to nodes $n_1 \ldots n_k$ and suppose that when the message arrives it finds $q_1 \ldots q_k$ other messages waiting to be transmitted to nodes $n_1 \ldots n_k$, respectively. These values represent the number of messages waiting to be received by nodes $n_1 \ldots n_k$. They can be viewed as implicit output queues at these nodes. Of course, actual output queues do not exist at the nodes, but these "implicit" queues can be used for the analysis. Clearly, because of the FCFS receiver algorithm this message will have to wait $\max(q_1, \ldots q_k)$ before it can be received by all of the nodes. Accounting for an additional transmission time of one slot and for the remaining service time of the message at the head of the queue[2] its total wait would then be

$$T = 1 + \max(q_1, \ldots q_k) + R_{\max}. \tag{4}$$

In order to compute $\overline{T}$ we must first compute the average value of $\max(q_1, \ldots q_k)$. It can be shown that for k finite, as N approaches infinity, $q_1 \ldots q_k$ behave as k independent M/D/1 queues. Hence we wish to compute the average value of the maximum queue size of k independent M/D/1 queues.

Let $Q_{\max} = \max(q_1, \ldots q_k)$. Then

$$\begin{aligned} &\Pr(Q_{\max} \le l) = \\ &= \Pr(q_1 \le l)\Pr(q_2 \le l) \ldots \Pr(q_k \le l) = \Pr(q_1 \le l)^k \end{aligned} \tag{5}$$

and

$$\begin{aligned} &\Pr(Q_{\max} = l) = \Pr(Q_{\max} \le l) - \Pr(Q_{\max} \le l-1) \\ &= \Pr(q_1 \le l)^k - \Pr(q_1 \le l-1)^k \end{aligned} \tag{6}$$

Thus,

$$\begin{aligned} \overline{Q}_{\max} &= \sum_{l=1}^{l=\infty} l \times \Pr(Q_{\max} = l) \\ \overline{Q}_{\max} &= \sum_{l=1}^{l=\infty} l \times [\Pr(q_1 \le l)^k - \Pr(q_1 \le l-1)^k] \end{aligned} \tag{7}$$

Lastly, $\overline{R}_{\max}$, the average remaining service time of the message at the head of the queue, is simply equal to 1/2 times the probability that the busiest (implicit) queue is not empty[3]. Since the busiest queue is empty only if all of the queues are empty, we have

$$\overline{R}_{\max} = \frac{1 - (1-\overline{A})^k}{2}. \tag{8}$$

---

[2] In a slotted system, where all messages arrive and depart at slot boundaries, this remaining service time does not actually exist; however, this quantity does represent the number of messages that arrived during the same slot as our message, but were placed ahead of it in the queue.

[3] Notice that while the system is constantly backlogged, an output queue can still be empty if none of the transmissions on the w wavelengths are intended for it.

It can be shown that in a slotted system where all messages arrive at once at the beginning of a slot, $\overline{R}_{\max}$ of equation (8) is equal to the average number of messages that arrived during the same slot as our message but were placed ahead of it in the queue. Now, putting equations (4)-(8) together we obtain,

$$\overline{T} = 1 + \overline{R}_{\max} + \sum_{l=1}^{l=\infty} l \times [\Pr(q_1 \le l)^k - \Pr(q_1 \le l-1)^k]. \quad (9)$$

Hence, we have expressed $\overline{T}$ in terms of the probability distribution of the number of messages in a single M/D/1 queue. This distribution can be obtained by solving the corresponding Markov chain [14]. While we do not know of closed form results for this distribution, numerical evaluation is rather straight forward. Since the transmission takes place over W channels in parallel, $\overline{C} = W/\overline{T}$. Finally we obtain an expression for $\overline{A}$:

$$\overline{A} = \left(\frac{W}{\overline{T}}\right)\left(\frac{k}{N}\right). \quad (10)$$

Equation (10) gives the arrival rate of new messages to each of the independent M/D/1 queues representing the number of messages waiting to be received by a given node. This equation is expressed in terms of $\overline{T}$, which is given by equation (9). However, equation (9) is expressed in terms of the statistics of an M/D/1 queue with arrival rate $\overline{A}$. Hence, the two equations must be solved together in order to obtain the values of $\overline{A}$ and $\overline{T}$. This can be done using an iterative algorithm similar to the one described in [15,16].

In figure 1 we plot the throughput efficiency vs. k when W=N=Infinity. Notice that in this case the throughput efficiency is simply equal to $\overline{A} = k/\overline{T}$. This is the special case where the number of channels is equal to the number of nodes. This case has received much attention in the literature because it also applies to an NxN switch. As can be seen from the figure, with k=1 the maximum throughput of 0.586 can be achieved. This is the well known result of [13]. It is interesting to note, however, that as we increase k, the throughput increases. This increase in channel utilization is due to an increase load on the receivers. As k increases receivers become heavily utilized, but at the same time the probability of receiver conflicts is increased. However, the significance of this result is that with large k there is very little to be gained by trying to efficiently schedule multicast traffic. This is a surprising result that leads us to think that an unscheduled multicast protocol can achieve good throughput efficiency.

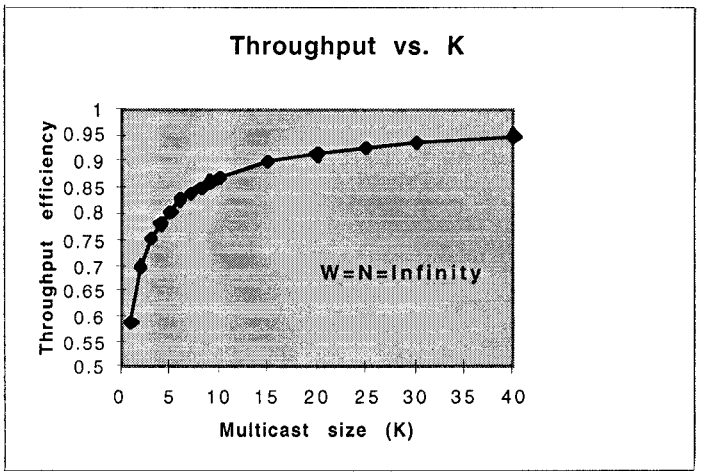

Figure 1. Throughput efficiency of unscheduled multicasts vs. multicast group size.

The results shown in figure 1 are obtained by solving equations (9) and (10), which are only exact for infinite values of N and W. With N and W finite, the analysis leading to equations (9) and (10) is approximate for two reasons: first, with finite W the arrivals at the different output queues would no longer be Poisson; and second, with finite N the output queues would not be independent. We rely on both of these assumptions to arrive at our results. However, it is interesting to note that this analysis provides a rather accurate approximation for finite values of N and W. In table 1 we compare the exact results for W=N=Infinity with those arrived at via simulation for finite values of N and W. As can be seen from the table, with W=N=100, the exact results (for W=N=infinity) are well within 1% of the simulation results. Even with a value of W=N=10 the simulations are within 5% of the exact results (for W=N=infinity). Hence, it appears that the exact results for a system with an infinite number of nodes and channels provide a rather good approximation for a system with a finite (even small) number of nodes and channels.

k	Exact W=N=Inf	Simulation W=N=10	Simulation W=N=32	Simulation W=N=100
1	0.586	0.612	0.593	0.588
2	0.700	0.719	0.704	0.702
3	0.750	0.769	0.756	0.752
4	0.784	0.803	0.789	0.786
5	0.808	0.826	0.813	0.810
6	0.826	0.844	0.831	0.828
7	0.841	0.857	0.846	0.843
8	0.853	0.868	0.857	0.854
9	0.863	0.878	0.867	0.864
10	0.871	0.886	0.875	0.872

Table 1. Comparison of the exact analysis (infinite nodes) to simulation with finite number of nodes and channels.

In figure 2 we consider what may be a more relevant case for a WDM network. In this figure we plot the throughput efficiency

vs. W/N for k=5. Here we define the throughput efficiency as the ratio of the lower bound of equation (1) to $\overline{T}$, the average number of transmissions per successful multicast. What we see from figure 2 is that when the number of channels is very small the efficiency approaches 1. This is because when the number of channels is small, the likelihood of a receiver collision (contention) is low and hence the efficiency is high. In contrast, when the number of channels is very high we see that the throughput approaches the value of 0.808, which is the throughput with k=5 and N=W=Infinity (see figure 1 or table 1). We should point out, however, that this apparent improvement in efficiency is somewhat artificial because it is due to an increased load on the receivers. The interesting region is when the receivers are neither lightly nor heavily loaded. It is interesting to note that the efficiency is minimized when W/N = 0.2. This represents a receiver load of 1 packet per slot per receiver.

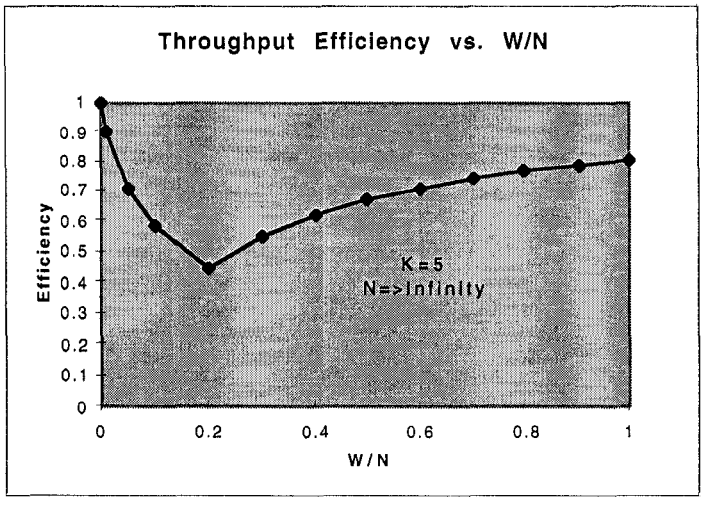

Figure 2. Throughput efficiency vs. W/N.

III. Random back-off protocol

The throughput of the persistent protocol is limited by a form of HOL blocking which results from a channel being unused while the message being transmitted on that channel is waiting for receivers to become available. An almost obvious solution to this problem is to allow new messages to be transmitted while other messages are waiting for receivers to become available. In this section we analyze a variation of this idea where messages that have not been received by all of their intended receivers are retransmitted after a random delay. This random delay removes the dependence that exists between the transmissions on the different channels, thereby alleviating the HOL blocking problem. Of course, this comes at the expense of the additional delay associated with a back-off algorithm.

Here we are not concerned with the protocol involved in the coordination of these transmissions, nor with the details of the back-off algorithm. We assume, as is done in the standard analysis of the Aloha protocol [17], that retransmissions take place sometime in the future and, hence, the transmissions on the different channels are independent of one another. We also assume, for the purpose of simplifying the analysis, that when a receiver has multiple transmissions to choose from it selects a transmission at random. The issue of received selection policy will be addressed in more detail in the next section. Again, we are interested in computing the average number of times a message has to be transmitted in order for it to be received by all of its intended recipients.

Let X be the average number of intended recipients for a multicast transmission during a slot (this transmission may be a retransmission of an older message and therefore may be addressed to fewer than k nodes). The first time a message is transmitted X=k. In subsequent transmissions the number of intended recipients decreases as some recipients have already received the message previously. Clearly, $1 \le X \le k$.

Let P be the probability that an arbitrary node is addressed by an arbitrary multicast transmission (occurring on one of the W channels) during a slot. Clearly

$$P = \frac{X}{N}. \tag{11}$$

Let N_j be the number of transmissions intended for node j. Then, the probability distribution of N_j is given by

$$P(N_j = l) = \binom{W}{l} P^l (1-P)^{W-l}. \tag{12}$$

Suppose now that node j is addressed by a multicast transmission, the probability that node j will choose to receive this particular transmission is equal to the probability that node j selects this transmission out of all of the transmissions addressed to it during this slot. This probability is given by,

$$P_s = \sum_{l=0}^{W-1} \left(\frac{1}{l+1}\right)\binom{W-1}{l} P^l (1-P)^{W-l-1}, \tag{13}$$

and $P_{ns} = 1 - P_s$, is the probability that the node does not select this particular transmission. Next we must compute X, the average number of nodes addressed by a multicast transmission. This number represents the average number of intended recipients on each wavelength.

Let X^i be the number of nodes addressed during the *ith* transmission attempt of a message. Then, $X^1 = k$ and X^2 is equal to the number of nodes that failed to receive the transmission in the first attempt. In general, X^r is equal to the number of nodes that have not received the message by the *rth* attempt.

The probability that a node requires r or more transmissions is equal to the probability that a node failed to receive the message during the first r-1 transmissions and is equal to $(1-P_s)^{r-1} = (P_{ns})^{r-1}$.

Hence,

$$P(X^r = l) = \binom{k}{l}\left[(P_{ns})^{(r-1)}\right]^l\left[1-(P_{ns})^{(r-1)}\right]^{k-l}. \quad (14)$$

The probability that a message is transmitted exactly r times is equal to the probability that r or more transmissions are required minus the probability that r+1 or more transmissions are required. Now, the probability that at least r transmissions are required is equal to the probability that at least one node requires r or more transmissions and is given by

$$P_t(r \le \#trans) = 1-(1-(P_{ns})^{r-1})^k. \quad (15)$$

Finally, the probability that exactly r transmissions are needed is given by

$$P_t(r) = (1-(P_{ns})^r)^k - (1-(P_{ns})^{r-1})^k. \quad (16)$$

Therefore, the average number of transmissions required is

$$E[Trans] = \sum_{r=1}^{\infty} r[(1-(P_{ns})^r)^k - (1-(P_{ns})^{r-1})^k]. \quad (17)$$

So far we have obtained the average number of transmission attempts per message. In order to compute X, the average number of intended recipients per attempt, we must first obtain NR, the average number of intended recipients for a message over all transmission attempts. Recalling that X^r is equal to the number of intended recipients in the rth transmission of the message, we have

$$NR = \sum_{r=1}^{\infty} \overline{X^r} = \sum_{r=1}^{\infty} k(P_{ns})^{r-1} = k\sum_{r=0}^{\infty}(P_{ns})^r = \frac{k}{1-P_{ns}}.$$

Thus, X, the average number of recipients per transmission, is given by

$$X = \frac{k}{(1-P_{ns})E[Trans]}. \quad (18)$$

Now, equations (11)-(13) provide an expression for $P_{ns} = 1-P_s$ that is in terms of X, and equation (18) gives an expression for X in terms of P_{ns}. The two equations can be solved together to obtain P_{ns}. Once P_{ns} is obtained, equation (17) can be used to obtain the average number of times that a message must be transmitted in order to be received by all of the multicast group members.

Figure 3 shows these values for a case with 100 nodes and 5 members per multicast group. Also shown in the figure is the lower bound on the number of required transmissions from equation (1). It is interesting to observe that, at least in the case examined here, the random back-off protocol only needs at most one more transmission (on average) than the lower bound. As the receiver load increases with increasing number of wavelengths, the efficiency of the random protocol also increases. More interestingly, figure 4 compares the transmission efficiency of the random back-off algorithm to that of the persistent algorithm. It is interesting to note that, while the algorithms exhibit similar behavior, the back-off algorithm always results in higher throughput.

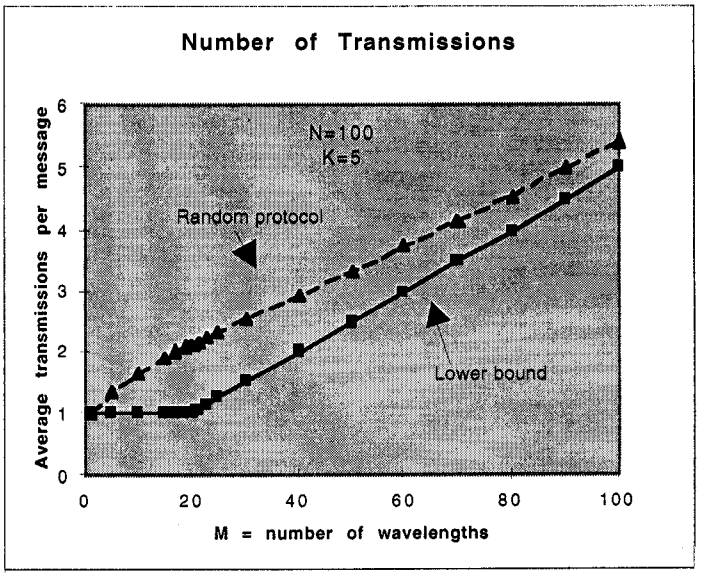

Figure 3. Average number of transmissions per multicast message for unscheduled multicasts with random back-off delay.

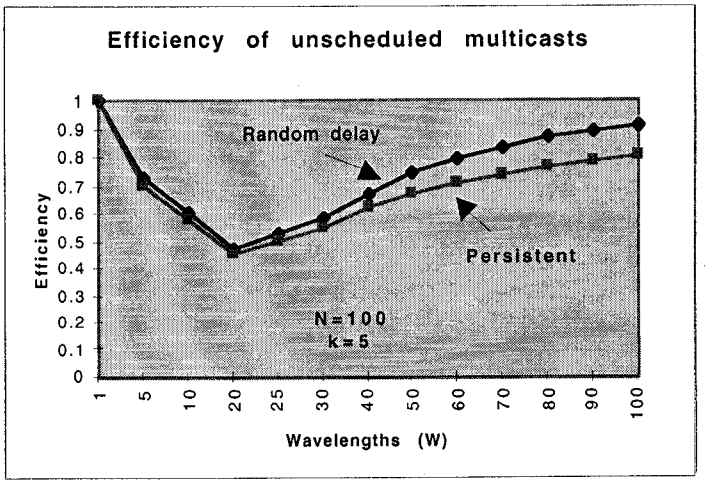

Figure 4. Throughput efficiency for random back-off and persistent protocols.

IV. Receiver Algorithm

We have seen so far that the throughput of unscheduled multicasts can be improved by introducing a random delay between retransmissions of a message. An additional improvement in the performance of these unscheduled multicast can be achieved by having receivers intelligently, rather than randomly, select the message they receive in the event of a conflict. So far, in the persistent algorithm the receiver selected the message on a FCFS basis and in the random back-off algorithm the receiver selected the message at random.

Here we consider an algorithm where the receiver selects the message with the smallest number of (remaining) intended receivers. The intuition behind the algorithm is that by selecting the message with the smallest number of intended receivers the probability that a message will be released (having been received by all of its intended recipients) is maximized, thereby making way for the transmission of a new message. If, alternatively, a message is selected with a larger number of intended recipients, it is more likely that message will have to be retransmitted anyhow. For example, suppose a receiver has two messages to choose from, one which is intended for 10 receivers and a second which is only intended for 1 receiver. By selecting the message intended for one receiver we are assured that its channel will be freed, making way for a new multicast transmission. On the other hand, had we selected the message with 10 intended receivers it is highly likely that both messages will require retransmission.

The performance of this simple algorithm, obtained via simulation, is plotted in figure 5. Also plotted in figure 5 are the performance of random selection for the back-off algorithm (obtained via the analysis of section III), the performance of FCFS selection for the persistent algorithm (obtained via the analysis of section II) and the performance of random selection for the persistent algorithm (obtained via simulation). As shown in the figure, the worst performance is that of the random selection persistent algorithm. The FCFS selection, which was analyzed in section II, offers a small improvement. An additional improvement is obtained when we introduce the random back-off protocol. Finally, the best throughput efficiency is obtained by the random back-off protocol with the priority selection algorithm described above. In all, for the case examined here (100 nodes and k=5) we see an improvement of as much as 40% in a heavily loaded system when going from the random selection persistent protocol to the priority selection random back-off protocol.

In figure 6 we plot the multicast efficiency for the special case of N = W (as in an NxN switch) with an infinite number of nodes and channels. As can be seen from the figure, the back-off algorithm reduces the effect of HOL blocking. This is a well known result for the unicast case [13]. Here we extend this result to show that for multicast transmissions over an NxN switch utilization can be improved by introducing a form of randomization to the inputs. We also show a further improvement in utilization when receivers select the transmission to which they listen by giving priority to the message with the least number of remaining intended recipients. It is interesting to note that for a system employing this strategy nearly 100% utilization can be achieved when k is greater than 6.

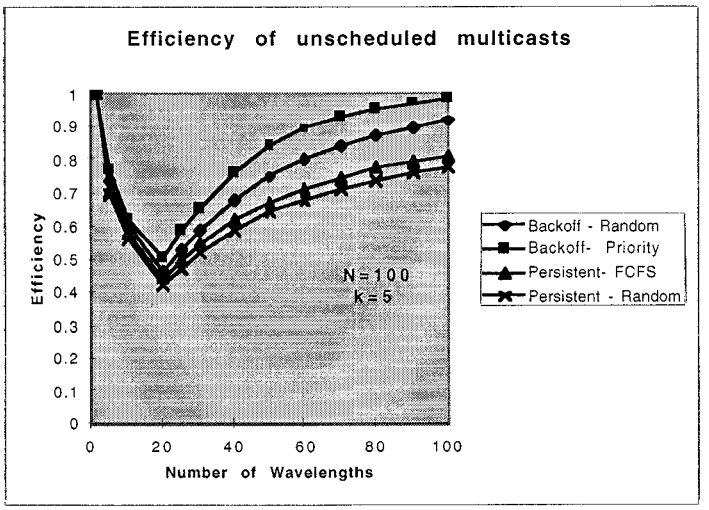

Figure 5. Efficiency for different receiver selection algorithms.

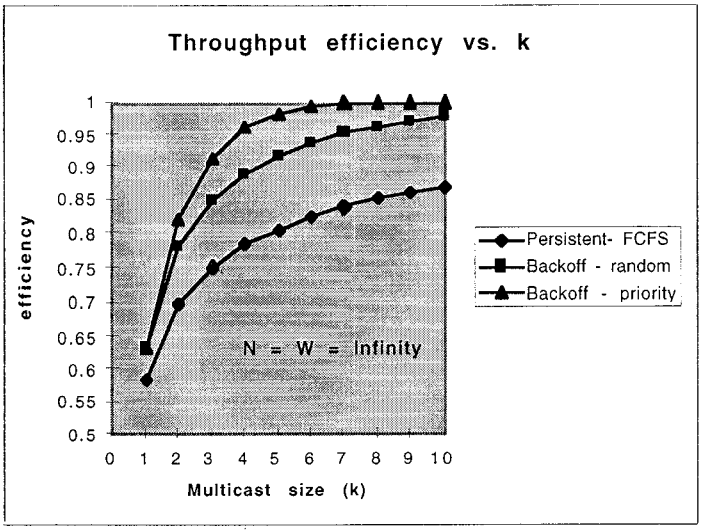

Figure 6. Efficiency for different algorithms with infinite number of nodes and channels.

Lastly, we are able to provide a simple comparison of the algorithms presented here to the Multicast Scheduling Algorithm (MSA) from [9]. Simulation results given in [9] for a system with 50 nodes and 10 wavelengths indicate a throughput efficiency of 20% with an average of 5.5 members per multicast transmission[4]. In contrast all of our algorithms result in throughput efficiency greater than 40% under similar

[4] The analysis in [9] assumed that the number of members per multicast transmission is uniformly distributed within a range of values. In this case the range of values was 1 to 10 giving an average group size of 5.5 .

circumstances (k=5). Similarly with an average of 10.5 members per multicast transmission the MSA algorithm results in throughput efficiency of 24% while all of our algorithms achieve throughputs higher than 57%. We should point out, however, that while the analysis in [9] was done with similar assumptions to the ones we make here, these assumptions are not exactly the same (e.g., the distribution of number of members per multicast transmission) and therefore this comparison is not exact. Nonetheless, assuming that these details have a small impact on the overall performance of the algorithm, this comparison is favorable to the algorithms presented in this paper. Clearly, a more detailed comparison of the algorithms presented in [8-10] to those presented here is warranted.

V. Conclusions

This paper examines the performance of unscheduled multicast transmissions in WDM broadcast LANs. This work also applies to the similar problem of multicasting in a TDM switching system. Exact throughput efficiency results were obtained for two random scheduling protocols, one that persistently retransmits a message until it has been received by all of the intended recipients and another which uses a random back-off between retransmissions. Our analysis indicates that the random back-off protocol results in improved efficiency. An important observation of this paper is that the throughput degradation due to receiver conflicts is decreased as the size of the multicast group increases. Intuitively, this is because for a given number of channels the likelihood of a receiver being idle is decreased as the number of intended recipients per transmission is increased.

We also examine the receiver algorithms for selecting among multiple simultaneous transmissions and showed, through simulation, that selecting the transmission with the smallest number of intended recipients performs better over a random selection policy. Hence we conclude that the best unscheduled multicast algorithm, among the ones we examined, is one with random back-off and priority selection.

Other areas that still require additional work include examining the mixed traffic case where transmissions are addressed to a variable number of nodes (that is, multicast groups that can vary in size). This case would also account for a system with a mix of multicast and unicast traffic. We believe that the analysis in this paper could be extended to this case in a rather straightforward manner. Lastly, it would be worthwhile to provide a more complete comparison of the performance of unscheduled multicasts to that obtained by a scheduling algorithm of the form introduced in [8-10]. In fact, an important contribution of this work is to provide a benchmark for comparison of more sophisticated multicast scheduling algorithms.

References

[1] G. J. Armitage, "IP Multicasting over ATM Networks," IEEE JSAC, April 1994.

[2] W. Stallings, "Ipv6: The New Internet Protocol," IEEE Communications Magazine, July, 1996.

[3] I.P. Kaminow, et. al., "A Wideband All-Optical WDM Network", IEEE JSAC, June, 1996.

[4] Eytan Modiano, Richard Barry and Eric Swanson , "A Novel Architecture and Medium Access Control (MAC) Protocol for WDM Networks," OFC '98, San Jose, CA, February, 1998.

[5] B. Mukherjee, "WDM-Based Local Lightwave Networks Part I: Single-Hop Systems," IEEE Network, May, 1992.

[6] N. Mehravari, "Performance and Protocol Improvements for Very High Speed Optical Fiber Local Area Networks Using a Passive Star Topology," Journal of Lightwave technology, April, 1990.

[7] K. M. Sivalingam and P.W. Dowd, "A Lightwave Media Access Control Protocol for WDM-based Distributed Shared Memory System," IEEE Infocom, March, 1996.

[8] G. N. Rouskas and M. H. Ammar, "Multidestination Communication Over Tunable-Receiver Single-Hop WDM Networks", JSAC, April 1997.

[9] M.S. Borella and B. Mukherjee, "A Reservation-Based Multicating Protocol for WDM Local Lightwave Networks," ICC '95, Seattle, WA, June, 1995.

[10] J.P. Jue and B. Mukherjee, "The advantages of Partitioning Multicast Transmissions in a Single-Hop Optical WDM Network," ICC '97, Toronto, Canada, June, 1997.

[11] K.L. Yeung, K.F. Au-Yeung and L. Ping, "Efficient Time Slot Assignment in a TDM Multicast Switching System," ICC '97, Toronto, Canada, June, 1997.

[12] M.K.M. Ali and S. Yang, "Performance Analysis of Random Packet Selection Policy for Multicast Switching," IEEE Transactions on Communications, March, 1996.

[13] M.J. Karol, M.G. Hluchyj and S.P. Morgan, "Input Versus Output Queueing in a Space-Division Packet Switch," IEEE Transactions on Communications, December, 1987.

[14] L. Kleinrock, *Queueing Systems*, Vol. 1, John Weily and Sons, 1976.

[15] E. Modiano and A. Ephremides, "A Method for Delay Analysis of Interacting Queues in Multiple Access Systems," INFOCOM 93, San Francisco, CA, March, 1993.

[16] G. N. Lance, *Numerical Methods for High Speed Computers*, London: Iliffe, 1960, pp. 134-138.

[17] Bertsekas and Gallager, *Data Networks*, prentice Hall, 1992