Multidestination Communication Over Tunable-Receiver Single-Hop WDM Networks

George N. Rouskas, *Member, IEEE*, and Mostafa H. Ammar, *Senior Member, IEEE*

***Abstract*— We address the issue of providing efficient mechanisms for multidestination communication over one class of lightwave wavelength division multiplexing (WDM) architectures, namely, single-hop networks with tunability provided only at the receiving side. We distinguish a number of multicast traffic types, we present a number of alternative broadcast/multicast time-division multiple-access (TDMA) schedules for each type, and we develop heuristics to obtain schedules that result in low average packet delay. One of our major contributions is the development of a suite of adaptive multicast protocols which are simple to implement, and have good performance under changing multicast traffic conditions.**

***Index Terms*— Adaptive multicast protocols, multidestination communication, optical WDM networks.**

I. Introduction

THE single-hop architecture for lightwave networks [11] employs wavelength division multiplexing (WDM) to divide the enormous information-carrying capacity of single mode fiber into multiple concurrent channels. Single-hop networks are all-optical in nature, i.e., the entire path between the source and the destination is optical. For a successful packet transmission, one of the transmitters of the source and one of the receivers of the destination must operate on the same wavelength. Thus, tunable transceivers are required, as well as some form of coordination among nodes wishing to communicate. We focus on a wavelength-time assignment of the optical bandwidth, whereby time is slotted and each node may transmit only in slots specified by a predetermined schedule.

Several single-hop architectures have been proposed in the literature and have been extensively studied for *single-destination* traffic [4], [7], [8], [10]. In particular, the authors have developed a general framework for analyzing and optimizing the throughput [13] and delay [14] performance of single-hop networks using time-division multiple-access (TDMA) schedules, for any number of wavelengths, any transceiver tunability characteristics, and general (potentially nonuniform) traffic patterns. It is, however, widely believed [16], [17] that with the advent of computer applications and telecommunication services such as distributed data processing, broadcast information systems, and teleconferencing, a significant portion of the overall traffic in future broadband networks will be of the *multidestination* type. In the context of single-hop networks, the issue of multidestination traffic has been addressed in [5], where a control-channel based multicast protocol is presented, and in [12], where we investigated the throughput characteristics of various multicast schemes.

In this paper, we suggest, analyze, and optimize a number of alternative approaches to performing efficient broadcast/multicast over single-hop lightwave networks employing tunable receivers. The main difference between this work and the one in [12] is our use of a more realistic model to evaluate the protocols and the emphasis on the delay performance of the various approaches.

This paper is organized as follows. In Section II we introduce the network model, and the various schedules we consider. In Section III we present a classification of multidestination traffic, and we argue for the need to provide a variety of multicast mechanisms addressing the requirements of each type of multidestination traffic. Section IV describes an optimization heuristic for constructing schedules that minimize the average packet delay, and in Section V a suite of adaptive multicast protocols is developed. We present numerical results in Section VI and conclude with a summary of our work in Section VII.

II. System Model

We consider a network of N nodes, each equipped with one transmitter and one receiver, interconnected through a passive broadcast optical medium that can support $C \leq N$ wavelengths, $\lambda_1, \lambda_2, \cdots, \lambda_C$. We consider systems with fixed transmitters, and receivers that can be tuned to any and all wavelengths λ_c, $c = 1, \cdots, C$. We let $\lambda(i) \in \{\lambda_1, \cdots, \lambda_C\}$ denote the channel assigned to the fixed transmitter of node i. We also let X_c denote the set of transmitters assigned to channel λ_c.

The network operates in a slotted mode with a slot time equal to the packet transmission time plus the tuning latency; the latter is defined as the time it takes a receiver to tune from one wavelength to another. A *collision* occurs when two or more transmitters access the same channel during a slot. When two or more sources transmit to the same destination on different channels the result is a *destination conflict*. All packets involved in a collision or destination conflict are considered lost.[1]

Manuscript received January 15, 1996; revised September 15, 1996. An earlier version of this paper was presented at the IEEE INFOCOM'94 Conference, Toronto, Ontario, Canada, June 14–16, 1994.

G. N. Rouskas is with the Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8206 USA.

M. H. Ammar is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0280 USA.

Publisher Item Identifier S 0733-8716(97)02265-8.

[1] In some cases, one of the packet involved in a destination conflict may be

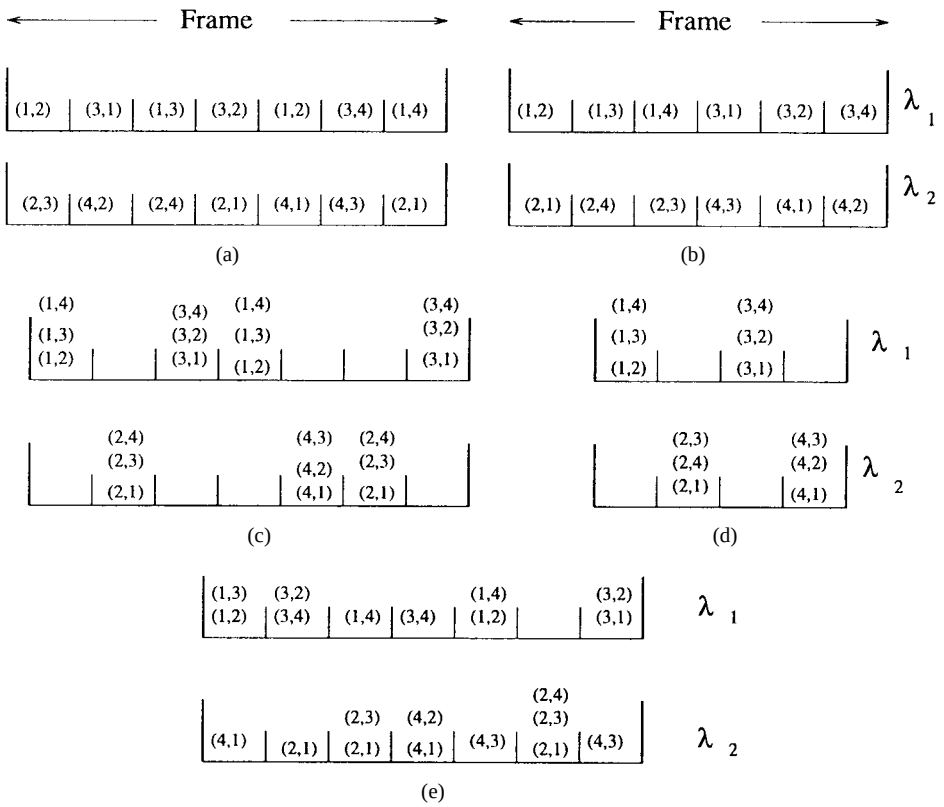

Fig. 1. Schedules with: (a) and (b) unicast, (c) and (d) broadcast, and (e) multicast slot for a network with N = 4 notes, C = 2 wavelengths, $X_1 = \{1, 3\}, X_2 = \{2, 4\}$: (a) general schedule with unicast slots, (b) cyclic schedule with unicast slots, (c) general schedule with broadcast slots, (d) cyclic schedule with broadcast slots, and (e) schedule with multicast slots. (i, j) denotes that i has permission to transmit to j in this slot.

We distinguish between single- and multidestination packets; the latter need to be delivered to a number of nodes, members of a *multicast group*. We define σ_i and ρ_i as the probability that a new single-destination and multidestination packet, respectively, is generated at node i during a slot time. We let p_{ij} denote the probability that a new single-destination packet is destined to node j. Similarly, q_{ig} will denote the probability that a new multidestination packet is addressed to multicast group $g \subseteq \{1, \cdots, N\}$.

The buffer capacity at each node is organized into N queues. One queue is associated with each of the possible $N-1$ destinations; a single-destination packet addressed to j joins the queue for this node. The Nth queue (to be called the *multicast queue* from now on) is used for storing arriving multidestination packets regardless of their multicast group.[2] Having separate queues eliminates the head of line effects of a single buffer and helps to drastically improve the delay and throughput characteristics as demonstrated in [4] and [14].

A. Transmission Schedules

The media access scheme we consider is an extension of weighted TDMA over a multichannel environment. In such a scheme, time slots are grouped in frames of $M \geq N$ slots. A *transmission schedule* indicates, for all i and j, which slots during a frame can be used for transmissions from i to j. If node i has permission to transmit to node j in slot t, then receiver j will tune to channel $\lambda(i)$ in slot t.

Based on the permissions given to the various sources, a slot t can be classified as the following.

1) *Unicast slot:* Exactly C nodes are given permission to transmit in slot t, each on a different channel and receiver.
2) *Broadcast slot:* Exactly one node k is allowed to transmit in slot t, and all receivers have to tune to $\lambda(k)$ in this slot. We will call k the *owner* of slot t; a packet transmitted by k will reach all nodes in the network, thus the name "broadcast slot."
3) *Multicast slot:* A number m, $1 \leq m \leq C$, of nodes are given permission to transmit in slot t, each on a different channel. One of the nodes, say k, may transmit to the receivers of a multicast group g, while the other $m-1$ nodes may transmit to exactly one receiver which is not a member of g. We say that k is the owner of the slot, and its transmission will reach all nodes in g.

Fig. 1 shows schedules with unicast, broadcast, and multicast slots for a fixed transmitter, tunable receiver network with $N = 4$ nodes and $C = 2$ wavelengths. Channel λ_1 is shared by the transmitters of nodes 1 and 3 ($X_1 = \{1, 3\}$), while channel λ_2 is shared by the transmitters of nodes 2 and 4 ($X_2 = \{2, 4\}$). The cyclic schedules also shown are special

successfully received; this will happen if it is the only packet transmitted on the channel the receiver listens to in that slot.

[2] Due to the large number of potential multicast groups, it is not practical to dedicate one queue for each such group. If a certain node, as is typical of real systems, maintains only a small number of active multicast sessions, say m, we may use m queues for multidestination packets, one for each multicast session.

cases, whereby each node may transmit to each possible destination exactly once per frame (for unicast slots), or is the owner of exactly one broadcast slot per frame.

We will be concerned with evaluating the performance of the various schedules in terms of average packet delay and aggregate throughput. Packet delay is defined as the number of slots elapsed between the generation of a packet at its source and the slot in which it is transmitted. Throughput is defined as the expected number of packets successfully *received* per slot. This definition assumes that the tuning latency (which is included as part of every slot) is small compared to the packet transmission time. This assumption is reasonable for systems with a small number of channels C and for local area networks/multiple area networks (LAN's/MAN's) with relatively large packet sizes (a few thousand bytes), since the tuning latency would only take up a small fraction of the slot time. On the other hand, including the tuning latency in each slot would be highly inefficient for environments characterized by very small packet sizes [e.g., asynchronous transfer mode (ATM) cells]. In these situations, the packet transmission time may be only a fraction of the tuning latency of even the fastest currently available tunable optical filters, and techniques to construct schedules to hide or overlap the tuning latency (see [3], [6], [15]) would be more appropriate.

III. Schedules for Mixed Single-/Multidestination Traffic

When only single-destination traffic is offered to the network, schedules with unicast slots only are sufficient. These schedules may also be used for multidestination traffic. In this case, a source would have to transmit a multidestination packet multiple times, once to each member in the packet's multicast group. Obviously, this approach does not take advantage of the inherent multicast capabilities of tunable receiver single-hop networks. We now describe how we may use broadcast or multicast slots to carry multidestination packets.

Let g be the multicast group of a multidestination packet originating at node i. Typically, the members of g are not known in advance; also, group membership may change during the life of the multicast communication. One way to guarantee that a packet will be received by all current members of its multicast group would be to have node i broadcast the packet to the entire network. The receivers would then use the multicast address to filter out any packets they do not need. This can be achieved by setting aside some of the slots of the schedule as broadcast slots with i as their owner; source i would then use its broadcast slots to transmit its multidestination packets. However, if the average size of a multicast group is small compared to the number of nodes in the network (a situation that often arises in distributed computing systems), an approach that attempts to deliver all multicast packets to all nodes in the network would be extremely wasteful in terms of bandwidth. Ideally, we would like to have schedules that allow a source to deliver a packet only to the current members of a packet's multicast group.

To this end, we may allocate a number of slots in the schedule as multicast slots with i as their owner. Only members of a multicast group g for which i acts as a source will tune their receivers to $\lambda(i)$, i's transmit wavelength, in these slots. Other nodes may tune their receivers to wavelengths other than $\lambda(i)$, allowing sources other than i to transmit single-destination packets in i's multicast slots. Thus, unlike broadcast slots in which all receivers are tuned to a certain source's wavelength and no other communication may take place, multicast slots provide for transmission concurrency and higher channel utilization.

Allocating at least one slot per frame for transmissions to each possible multicast group would be impractical even for networks of moderate size, as the number of possible multicast groups increases very rapidly with the number of nodes. Typically, though, at any given time, a source participates in a small number of multicast sessions. But as multicast groups of existing sessions change over time, or as active sessions terminate and new ones start, the schedule has to be modified to reflect these changes. Thus, unless a mechanism for dynamically updating the permissions in the multicast slots of the frame is available, the only practical approaches to multicasting would be to use either unicast or broadcast slots.

As we can see, the nature of multidestination traffic is of crucial importance in designing the schedules. We may classify this traffic along two dimensions: the average length of multicast sessions and the average size of multicast groups. In a typical distributed computing environment (such as when multicast is used for "response collection" applications [1]) the multicast session is short, involving the exchange of a few packets. In video and audio conferencing applications, however, the multicast session may be quite long, requiring the transmission of a large number of packets. On the other hand, the average group size is often small in distributed computing applications. But applications that require the sending of messages to all or a large number of the nodes in the network do exist (for instance, the update of routing information).

We now distinguish three types of multidestination traffic.

- *Type 1*) Multicast sessions are relatively short, and multicast groups are likely to include a large number of nodes. Schedules with both unicast (for single-destination packets) and broadcast (for multidestination packets) slots are most appropriate.
- *Type 2*) Multicast sessions are relatively short, while the average multicast group contains few nodes. Since broadcast slots may lead to substantial under utilization of bandwidth, schedules with unicast slots only should be used, and multidestination packets should be transmitted to each of their destinations individually.
- *Type 3*) Multicast sessions are relatively long. Schedules with both unicast and multicast slots may be used, and we have developed a suite of protocols to dynamically update the permissions in each multicast slot according to the current multicast group.

Our conclusions are summarized in Table I. The next section describes a way to construct schedules for Type 1 traffic so that the average packet delay is minimized, while Section V

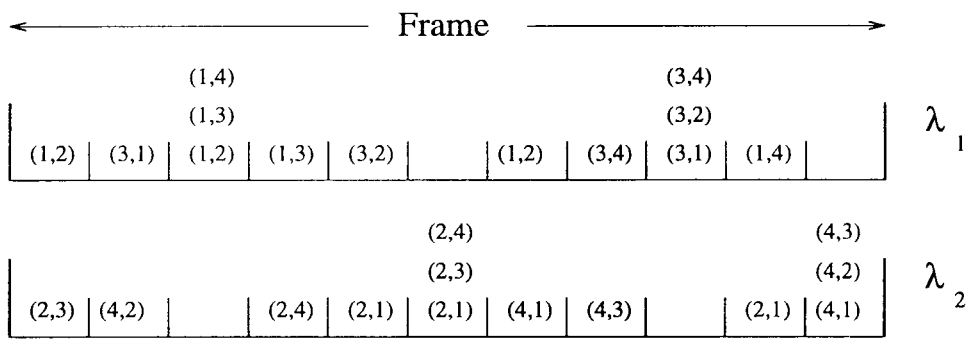

Fig. 2. Schedule merging: (i, j) denotes that i has permission to transmit to j in this slot.

TABLE I
SCHEDULES APPROPRIATE FOR THE VARIOUS TYPES OF MULTIDESTINATION TRAFFIC

		Average Duration of Multicast Session	
		Short	Long
Average Size of Multicast Group	Small	Unicast Slots	Unicast and Multicast Slots
	Large	Unicast and Broadcast Slots	

presents a number of adaptive protocols for Type 3 traffic. How to obtain schedules with unicast slots only that minimize the average packet delay for Type 2 traffic is discussed in Appendix A.

IV. DELAY MINIMIZATION FOR TYPE 1 TRAFFIC

Our objective is to determine schedules with unicast and broadcast slots such that the network-wide average packet delay is minimized for a given set of traffic parameters σ_i, ρ_i, p_{ij}, and q_{ig}, $\forall i, j, g$. From previous experience with a similar problem (namely, the problem of constructing optimal schedules with unicast slots only for tunable transmitter, fixed receiver systems [14]) we expect Problem 1 to be a very hard allocation problem. We now describe a heuristic to obtain schedules that not only perform well, but also guarantee a designer-specified level of performance for each type of traffic (single- or multidestination).

The heuristic is based on a decomposition of the problem into two manageable subproblems, namely, the problems of finding optimal schedules assuming each type of traffic is offered to the network in isolation. Optimization techniques to obtain unicast slot schedules for single-destination traffic only ($\rho_i = 0\ \forall i$) are presented in Appendix A, while near-optimal broadcast slot schedules for multidestination traffic only ($\sigma_i = 0\ \forall i$) are derived in Appendix B. Then, the two schedules are appropriately *merged* into a final schedule for the mixed traffic at hand.

Let S_1 and S_2 be two schedules of frame lengths M_1 and M_2, respectively. Without loss of generality, assume that $M_1 \geq M_2$ and $M_1 = mM_2$. If m is an integer, merging of S_1 and S_2 is performed by inserting one slot of S_2 after every m slots of S_1, resulting in a new schedule S, of frame length $M = M_1 + M_2$. Schedule merging can be easily generalized to situations where M_1 is not an integer multiple of M_2. In Fig. 2 we show the result of merging the unicast slot schedule of Fig. 1(a) ($M_1 = 7$), with the cyclic broadcast slot schedule of Fig. 1(d) ($M_2 = 4$).

Consider a unicast slot schedule S for single-destination traffic only. Since there are no broadcast slots, multidestination traffic experiences infinite delay. If we merge S with l, $l \geq 1$, frames of a broadcast slot schedule S', we effectively provide slots in which multidestination packets may be transmitted, thus improving their delay performance. As l increases, the merged schedule will tend to favor multidestination packets (note that as $l \to \infty$, the resulting schedule will be indistinguishable from an S' schedule, in which case single-destination traffic will suffer). Therefore, we must choose an l such that both the overall delay, D_{overall}, is low and the delay of single-destination traffic, D_{single}, and the delay of multidestination traffic, D_{multi} is acceptable. Our approach is outlined in the following schedule-merging heuristic (SMH). Note that the stopping rule in Step 4 guarantees that the final schedule will contain broadcast slots.

Schedule-Merging Heuristic (SMH):

1) Given single-destination traffic parameters σ_i and p_{ij}, and the number of wavelengths, C, obtain an unicast slot schedule S_0 of length M, as in Appendix A.
2) Given ρ_i and C obtain a broadcast slot schedule S'_0 of frame length M', as in Appendix B. Set $l \leftarrow 1$.
3) Merge one frame of S_0 with l frames of S'_0 to produce a new schedule, S_l, of frame length $M + lM'$.
4) If $l = 1$ or $[D_{\text{overall}}(S_l) < D_{\text{overall}}(S_{l-1})$ *and* $D_{\text{single}}(S_l)]$ and $D_{\text{multi}}(S_l)$ acceptable, set $l \leftarrow l+1$ and repeat from Step 3. Otherwise, stop; the best schedule to use is S_{l-1}.

V. ADAPTIVE MULTICAST PROTOCOLS FOR TYPE 3 TRAFFIC

We now present adaptive multicast protocols which assume that each node i is the owner of b_i multicast slots per frame. The protocols are adaptive in the sense that the transmissions allowed in these multicast slots are not specified in advance; instead, they are dynamically updated to reflect the current members of multicast groups.

A. The Basic Idea

The operation of the protocols is based on the assumption that a source i will transmit P consecutive packets, $P > 1$,

to the same multicast group, g. Typically, P takes a value between some minimum and maximum values $P_{\min}$ and $P_{\max}$, respectively. We will now describe the basic idea behind the operation of the protocols by considering the transmissions in node i's multicast slots; similar observations can be made for other nodes' multicast slots. We will also assume that i transmits to only one multicast group at a time. The protocols, however, can be easily extended to handle a source that maintains a small number, m, of multicast sessions simultaneously; in these situations, it would be beneficial to the source to have m rather than one multicast queues, one for the packets belonging to each session.

In the first multicast slot with i as its owner all nodes tune their receivers to $\lambda(i)$, the transmit wavelength of i. Let g be the multicast group to which the packet transmitted by i in that slot is addressed, and let P be the total number of packets i will transmit to the same group; $g = \phi$ if no packet is transmitted by i in that slot. Suppose that $|g| < N - 1$, and consider a node $j \neq i$. If $j \in g$ then j will continue listening to $\lambda(i)$ in subsequent multicast slots of i. However, if $j \notin g$, j is free to tune its receiver to the transmit wavelength of another node, k, in subsequent multicast slots of i. If k has a single-destination packet for j, and provided that $\lambda(k) \neq \lambda(i)$, k will transmit to j in i's multicast slots, thus increasing channel utilization.

After i transmits all P packets to the same multicast group g, it will not be able to transmit to a group $g' \neq g$, unless all nodes not in g are somehow notified. We therefore require that all nodes tune their receivers to $\lambda(i)$ in specified multicast slots of i, called *synchronization* slots[3] (as explained, the first multicast slot is a synchronization slot). The F multicast slots of i between synchronization slots are called *free* as receivers not in g are free to tune to any wavelength other than $\lambda(i)$. F is a network-wide constant and thus, all nodes can synchronize by tuning to $\lambda(i)$ in synchronization slots.

Note that i may start transmitting packets to a new multicast group only in a synchronization slot. If F is large relative to P, the number of consecutive packets to the same multicast group, i will, on average, have to wait for a considerable number of slots to start transmitting to a new group. On the other hand, if F is very small relative to P there will be unnecessarily many synchronization slots in which no transmissions by nodes other than i are allowed. F will, in general, be a function of P, as well as of the propagation delay (more on this later), and must be carefully selected in order to maximize the overall throughput.

We have not yet discussed how a receiver $j \notin g$ selects a transmitter $k \neq i$ to tune to in i's free multicast slots. There are two issues that need to be considered. First, $\lambda(k)$ must be different than $\lambda(i)$ to prevent packet loss due to collisions in free multicast slots. Second, k must also be informed of j's decision. Real-time negotiation between j and other nodes to determine k is impractical because of the propagation delays involved.

To solve the first problem we start with a unicast slot schedule S under which no collisions are possible, such as the one in Fig. 7. Let a_i be the number of slots per frame in which i may transmit under S. We then specify b_i, $b_i < a_i$, of these slots as multicast slots with i as their owner. Let t be one of these b_i slots and consider a node $j \neq i$ which, according to the initial schedule S, must tune its receiver to node k in slot t. If t is a synchronization slot, or if t is a free slot but $j \in g$, j will ignore the permissions specified by S and, in slot t, it will tune to $\lambda(i)$ instead. However, if t is a free multicast slot and $j \notin g$, j will tune to $\lambda(k)$ as S specifies. Note that, since no collisions are possible under S and both i and k are given permission to transmit in the same slot t, we have that $\lambda(i) \neq \lambda(k)$.

[3] Synchronization slots are actually broadcast slots.

B. Determining Group Membership

Since all nodes execute the same protocol, the problem of informing k about j's decision is now partially solved: k knows that j will tune to $\lambda(k)$ in slot t if a) t is a free slot, and b) $j \notin g$. Deciding about a) is done by k as part of the protocol for tuning its own receiver. Thus the problem reduces to how k may determine whether j is in the multicast group g or not. We now describe two protocols which differ in their assumptions about k's knowledge regarding membership in the multicast groups of packets originating at node $i \neq k$.

Global-Knowledge Multicast Protocol (GMP): Node k maintains tables to map a multicast address in a packet originating at i into the node-members of the multicast group. These tables are initialized and updated by a multicast group management protocol that operates independently of the actual multicast packet transmissions. By listening to a synchronization slot of i it can tell whether j is in the multicast group or not.

Control-Packet Multicast Protocol (CMP): Node k uses tables to map a multicast address into the members of the multicast group, but no independent multicast group management protocol is employed. Before transmitting a packet to a new multicast group g, i will first transmit, in a synchronization slot, a control packet with information about the members of g. Following the control packet transmission, i will transmit the P packets to g as discussed above. Upon receiving the control packet, node k updates its tables to associate g with the group members. This protocol incurs the overhead of one extra packet, but this is not expected to be a problem, especially if $P \gg 1$.

Both protocols assume that nodes have knowledge of the receivers in the various multicast groups. If nodes maintain no information about group membership (as in the IP multicast model), two approaches are possible. Node k may use a multicast group member server (similar to the MARS architecture [2] for IP multicast over ATM) to determine the members of a multicast group. Alternatively, k may have no way of determining whether receiver j belongs to the multicast group g. In this case, one possible approach would be for node k to transmit a packet, if it has one, to j in a free multicast slot of i with probability r. This approach is simple to implement, but the value of r must be selected so as to minimize packet loss due to destination conflicts [if $j \in g$, j will tune to $\lambda(i)$ in free multicast slots of i and k's transmissions in these slots will be wasted]. In general, r should represent the probability that

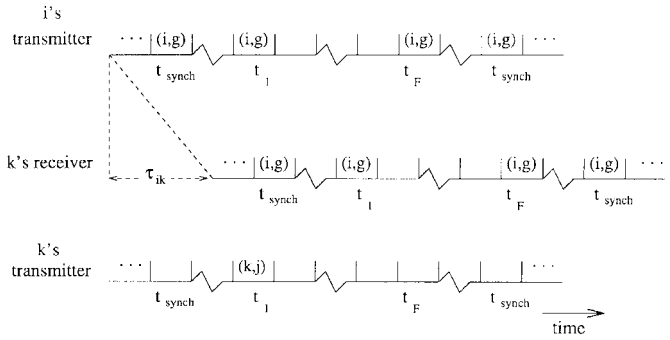

Fig. 3. Effect of the propagation delay (not in scale): (i, j) denotes that i may transmit to multicast group g in this slot, (k, j) denotes that k may transmit to j in this slot, and τ_{ik} is the propagation delay from i to k.

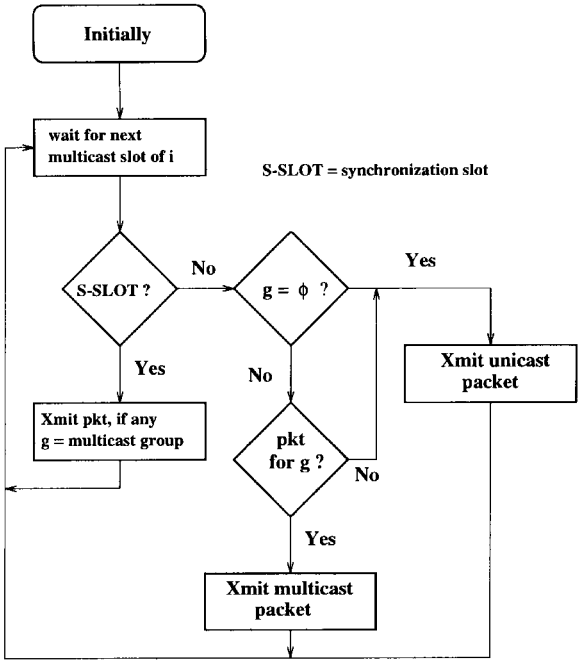

Fig. 4. Algorithm executed by i's transmitter for transmission in i's multicast slots.

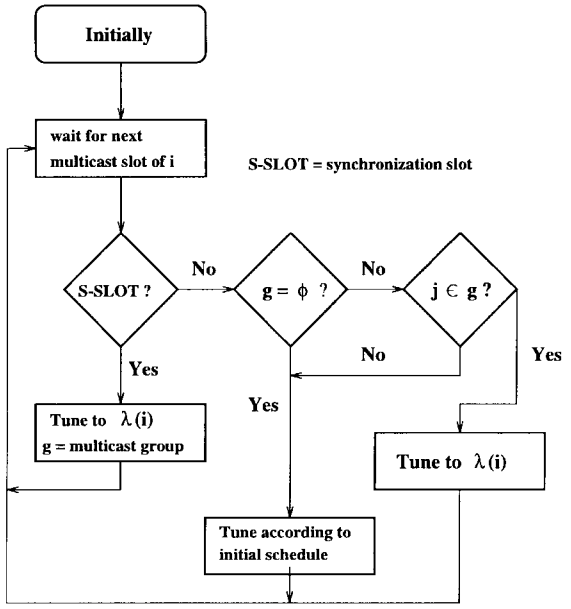

Fig. 5. Algorithm executed at j's receiver for tuning in i's multicast slots.

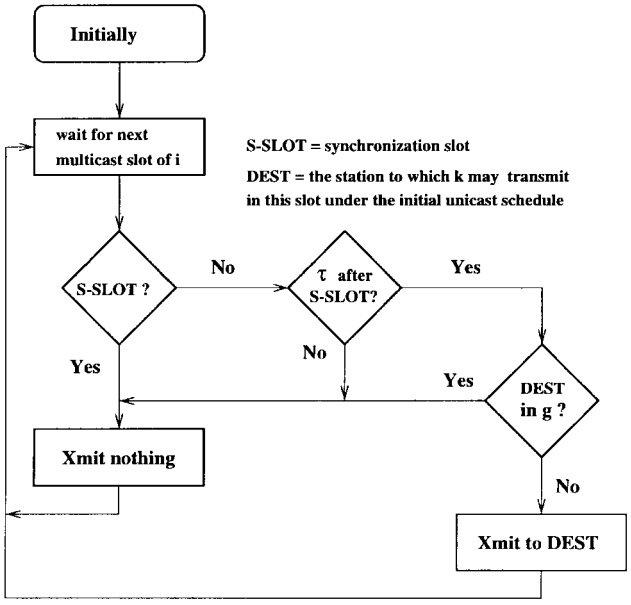

Fig. 6. Algorithm executed by k's transmitter for transmission in i's multicast slots ($k \neq i$).

j does *not* belong to g. Hence, if $\bar{\eta}$ is the average number of stations in a multicast group, r should be set to $1-\bar{\eta}/(N-1)$.

C. Effect of Propagation Delay on Throughput

Under either global-knowledge multicast protocol (GMP) or control-packet multicast protocol (CMP) a node $k \neq i$ must receive the packet transmitted by i in a synchronization slot before it can determine whether the nodes to which it is scheduled to transmit in the next free multicast slots of i belong to g or not. Fig. 3 illustrates how propagation delay may become a problem. In this figure we show a synchronization slot of i followed by F free slots and another synchronization slot; the horizontal axis represents time increasing from left to right. The transmitters of both i and k are synchronized at the beginning of each slot. But a packet transmitted by i will not be heard by the receiver of k until τ_{ik} slots later, where τ_{ik} is the propagation delay from i to k in slots. In the scenario of Fig. 3, by the time k receives the packet transmitted by i in the first synchronization slot, free slot t_1 has already passed by its transmitter. Since at the beginning of t_1, k does not know whether $j \in g$ it may not transmit a packet to it.

As a result of the propagation delays, some of the free multicast slots may not be used for single-destination transmissions; the longer the propagation delays the less free slots that may be utilized. In the extreme case when all F free slots are within a propagation delay, neither GMP nor CMP will be able to capitalize on the availability of free slots to improve the throughput. Thus, F is indeed a function of the propagation delay as mentioned earlier. Observe, though, that

0	0.70	0.05	0.05	0.05	0.05	0.05	0.05
0.05	0	0.70	0.05	0.05	0.05	0.05	0.05
0.05	0.05	0	0.70	0.05	0.05	0.05	0.05
0.05	0.05	0.05	0	0.70	0.05	0.05	0.05
0.05	0.05	0.05	0.05	0	0.70	0.05	0.05
0.05	0.05	0.05	0.05	0.05	0	0.70	0.05
0.05	0.05	0.05	0.05	0.05	0.05	0	0.70
0.70	0.05	0.05	0.05	0.05	0.05	0.05	0

(1,2)	(1,2)	(1,6)	(1,2)	(1,3)	(1,8)	(1,2)	(1,5)	(1,2)	(1,3)	(1,7)	(1,2)	(1,4)
(2,3)	(2,3)	(2,7)	(2,3)	(2,4)	(2,1)	(2,3)	(2,6)	(2,3)	(2,4)	(2,8)	(2,3)	(2,5)
(3,4)	(3,4)	(3,8)	(3,4)	(3,5)	(3,2)	(3,4)	(3,7)	(3,4)	(3,5)	(3,1)	(3,4)	(3,6)
(4,5)	(4,5)	(4,1)	(4,5)	(4,6)	(4,3)	(4,5)	(4,8)	(4,5)	(4,6)	(4,2)	(4,5)	(4,7)
(5,6)	(5,6)	(5,2)	(5,6)	(5,7)	(5,4)	(5,6)	(5,1)	(5,6)	(5,7)	(5,3)	(5,6)	(5,8)
(6,7)	(6,7)	(6,3)	(6,7)	(6,8)	(6,5)	(6,7)	(6,2)	(6,7)	(6,8)	(6,4)	(6,7)	(6,1)
(7,8)	(7,8)	(7,4)	(7,8)	(7,1)	(7,6)	(7,8)	(7,3)	(7,8)	(7,1)	(7,5)	(7,8)	(7,2)
(8,1)	(8,1)	(8,5)	(8,1)	(8,2)	(8,7)	(8,1)	(8,4)	(8,1)	(8,2)	(8,6)	(8,1)	(8,3)

Fig. 7. Ring-type matrix and corresponding 13 slot schedule for $C = 8$, $\sigma = 0.5$.

0	0.30	0.30	0.30	0.025	0.025	0.025	0.025
0.30	0	0.30	0.30	0.025	0.025	0.025	0.025
0.30	0.30	0	0.30	0.025	0.025	0.025	0.025
0.30	0.30	0.30	0	0.025	0.025	0.025	0.025
0.025	0.025	0.025	0.025	0	0.30	0.30	0.30
0.025	0.025	0.025	0.025	0.30	0	0.30	0.30
0.025	0.025	0.025	0.025	0.30	0.30	0	0.30
0.025	0.025	0.025	0.025	0.30	0.30	0.30	0

Fig. 8. Two-community-type matrix.

0	0.20	0.20	0.20	0.10	0.10	0.10	0.10
0.40	0	0.08	0.08	0.20	0.08	0.08	0.08
0.40	0.08	0	0.08	0.20	0.08	0.08	0.08
0.40	0.08	0.08	0	0.20	0.08	0.08	0.08
0.10	0.10	0.10	0.10	0	0.20	0.20	0.20
0.20	0.08	0.08	0.08	0.40	0	0.08	0.08
0.20	0.08	0.08	0.08	0.40	0.08	0	0.08
0.20	0.08	0.08	0.08	0.40	0.08	0.08	0

Fig. 9. Two-server-type matrix.

the propagation delay will have a negative effect only if it increases beyond the number of slots between *consecutive multicast* slots with the same owner. Going back to Fig. 3, if τ_{ik}, in slots, is less than the distance between the first synchronization slot and t_1, k will be able to transmit in t_1, as well as in all other free slots (if $j \notin g$). Otherwise, k will still be able to use slots $t_2, \cdots, t_F$ as long as τ_{ik} is further increased by less than the distance between t_1 and t_2, and so on. By assigning multicast slots to i so that they are spaced out in the frame we can make the distance between two consecutive multicast slots much larger than one slot. We have shown [12] that this technique makes GMP and CMP largely insensitive to propagation delays.

The algorithms used by the various transmitters and receivers for transmissions in i's multicast slots are shown in Figs. 4–6. The algorithms are very simple to implement, and thus suitable for the high-speed environment we are considering.

VI. Numerical Results

We consider the 8-node ring-type, two-community-type, and two-server-type single-destination traffic matrices with probabilities p_{ij} as shown in Figs. 7–9, respectively. We let $\sigma_i = \sigma \ \forall i$, and $\rho_i = \rho \ \forall i$; this does not compromise the generality of our results as a) the single-destination traffic characteristics are determined by p_{ij}, and b) we are interested in the behavior of our schedules as the relative amount of single- and multidestination traffic varies, and this is captured by the relative values of σ and ρ, and the average multicast group size, $\bar{\eta}$. In the simulations, the number of consecutive multicast packets is uniformly distributed in the range $[P_{\min}, P_{\max}]$. Multicast groups for a source i are constructed by randomly selecting nodes in the set $\{1, 2, \cdots, i-1, i+1, \cdots, N\}$ (ignoring duplicates) until the group size has been reached. All simulation results presented in this section were obtained

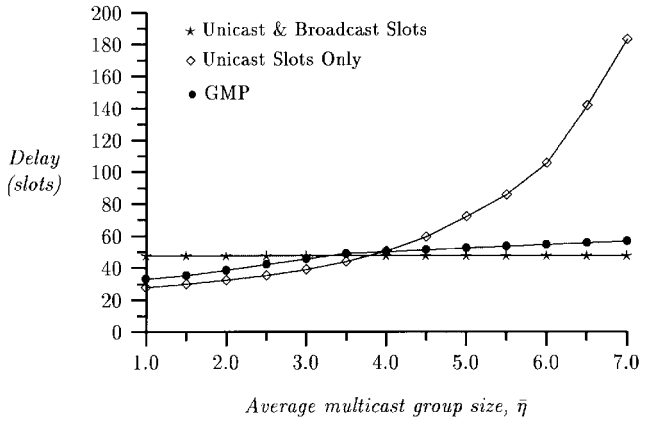

Fig. 10. Delay versus average multicast group size, two-community-type matrix, $N = 8$, $C = 2$, $\sigma = 0.1$, $\rho = 0.01$ (GMP was run with: $P_{\min} = 30$, $P_{\max} = 50$, $F = 50$).

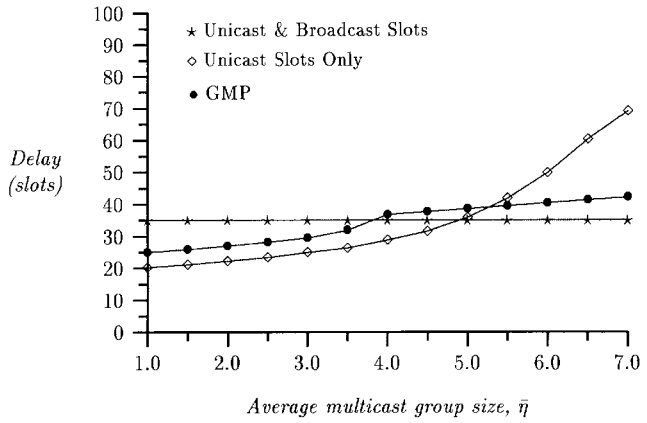

Fig. 12. Delay versus average multicast group size, two-server-type matrix, $N = 8$, $C = 4$, $\sigma = 0.3$, $\rho = 0.01$ (GMP was run with: $P_{\min} = 30$, $P_{\max} = 50$, $F = 50$).

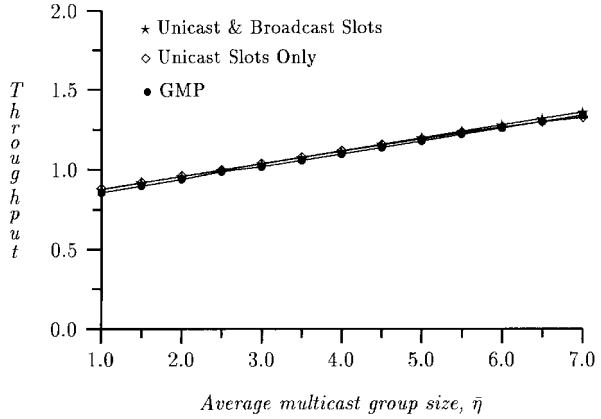

Fig. 11. Throughput versus average multicast group size, two-community-type matrix, $N = 8$, $C = 2$, $\sigma = 0.1$, $\rho = 0.01$ (GMP was run with: $P_{\min} = 30$, $P_{\max} = 50$, $F = 50$).

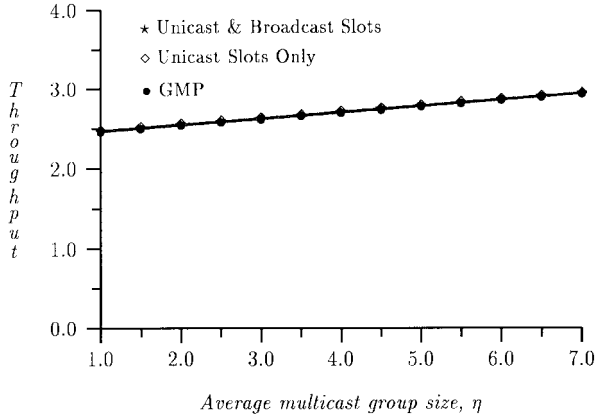

Fig. 13. Throughput versus average multicast group size, two-server-type matrix, $N = 8$, $C = 4$, $\sigma = 0.3$, $\rho = 0.01$ (GMP was run with: $P_{\min} = 30$, $P_{\max} = 50$, $F = 50$).

with a confidence of 95% in less than 1% variation from the mean.

Figs. 10, 12, and 14 plot the delay, and Figs. 11, 13, and 15 plot the throughput against the average multicast group size for the stated traffic matrix and values for parameters C, σ, and ρ. Each figure shows three delay or throughput curves, each curve corresponding to one of the multicast approaches presented earlier, namely, schedules with unicast slots only, schedules with unicast and broadcast slots, and schedules with unicast and multicast slots with the GMP adaptive multicast protocol. Similar results have been obtained for various other traffic matrices and a wide range of values for parameters C, σ, and ρ, as well as for larger values of N.

We note that throughput increases with average multicast group size, and that the three approaches have almost identical performance in terms of throughput. However, the three approaches differ drastically in terms of their delay behavior; this behavior can be explained by considering the way multicast packets are handled. First observe that when schedules with unicast and broadcast slots are used, the delay is independent of the multicast group size. Recall that these schedules are constructed using the SMH of Section IV, and that SMH inserts a number of broadcast slots according only

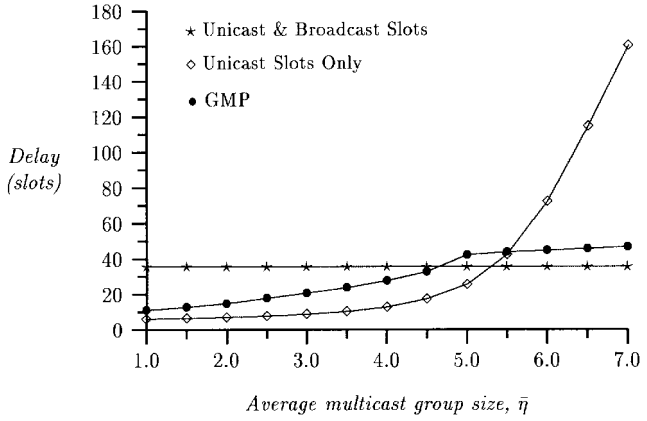

Fig. 14. Delay versus average multicast group size, ring-type matrix, $N = 8$, $C = 8$, $\sigma = 0.5$, $\rho = 0.02$ (GMP was run with: $P_{\min} = 30$, $P_{\max} = 50$, $F = 50$).

to parameters σ and ρ (corresponding to the offered unicast and multicast traffic load, respectively). These broadcast slots are used for sending multicast packets, thus the delay of each packet is independent of the number of recipients. On the other hand, when schedules with unicast slots only are used, a multicast packet has to be transmitted multiple times,

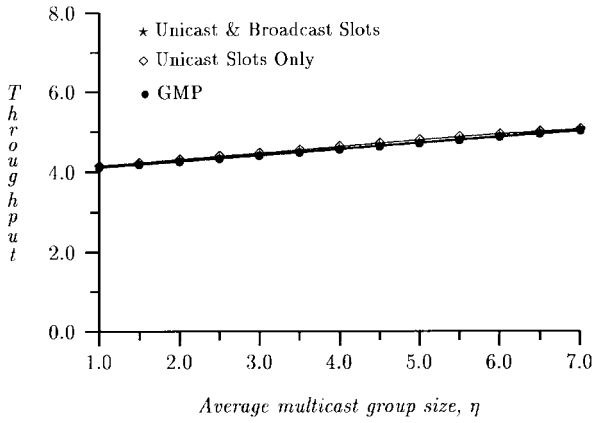

Fig. 15. Throughput versus average multicast group size, ring-type matrix, $N = 8, C = 8, \sigma = 0.5, \rho = 0.02$ (GMP was run with: $P_{\min} = 30, P_{\max} = 50, F = 50$).

once to each of the members in its multicast group. Hence, as the size of the multicast groups increases, delay also increases, resulting in the curves shown in Figs. 10, 12, and 14. As we can see, for small multicast group sizes, there is a penalty for using broadcast slots (all receivers tune to a certain wavelength, despite the fact that the multicast packet is addressed to a fraction of all possible destinations) making these schedules inferior to schedules with unicast slots only. However, with large multicast groups, transmitting a packet multiple times results in a waste of network resources; consequently, there is a point at which the two curves intersect, and beyond which it is preferable to use schedules with both unicast and broadcast slots. The point at which one approach outperforms the other depends on the traffic parameters.

Let us now turn our attention to the delay behavior of schedules with unicast and multicast slots running the GMP. (The performance of CMP is very similar and is omitted; recall that CMP differs from GMP only in that the source transmits, in a synchronization slot, an initial control packet containing information about the multicast group, before it transmits the actual multicast data packets.) The most important observation is that, although these schedules do not achieve lower delay compared to the ones discussed above, they *always* incur a delay close to that of the best static schedule. This behavior of GMP can be understood by examining the two extreme cases, namely, very small and very large multicast groups.

Let us assume that all multicast groups consist of one member, and consider the multicast slots of source i. In such a system, the free multicast slots of i behave almost like unicast slots, as one receiver (the one belonging to the current multicast group) will have to tune to $\lambda(i)$, while all others are free to tune to the home channels of other sources. However, GMP does incur some overhead compared to the pure unicast schedule, as all receivers must tune to $\lambda(i)$ during the synchronization slots of i. On the other hand, if the size of multicast groups tends to $N-1$ (a broadcast scenario), the behavior of the multicast protocol becomes similar to that of a static schedule with unicast and broadcast slots. The difference in this case is that, once all packets to the current multicast group have been transmitted, the source must wait for the next synchronization slot before it can transmit packets to a new multicast group. As a result, GMP incurs some overhead compared to the static schedule. The behavior of GMP between these two extremes is determined by which overhead is the dominant one. Thus, depending on the average multicast group size, the performance of GMP matches that of the schedule that incurs the lowest delay.

VII. Concluding Remarks

We have addressed the problem of carrying both multi- and single-destination traffic over single-hop WDM networks with tunability provided only at the receiving end. We have presented a classification of multidestination traffic that takes into account the size of multicast groups and the length of multicast sessions, and we have designed multicast mechanisms addressing the requirements of each type of traffic. We have also developed a suite of adaptive multicast protocols which are simple to implement, and which can be useful under changing traffic conditions. Overall, our results indicate that the multicast protocols can successfully adapt to a wide range of multicast group sizes. However, this is only possible when the number of packets to be transmitted to the same multicast group is relatively large. Otherwise, a near-optimal (static) schedule with unicast or unicast and broadcast slots (depending on the expected group size) will achieve the best performance.

Appendix
Schedule Optimization Heuristics

A. Schedules with Unicast Slots for Single-Destination Traffic

Let us consider the problem of finding a schedule that minimizes the network-wide average packet delay when only single-destination traffic is offered to the network (i.e., $\rho_i = 0\ \forall i$). There are three dimensions to this problem: a) the sets of transmitters, X_c, sharing wavelength λ_c, $c = 1, \cdots, C$, must be constructed, b) the number of slots per frame, a_{ij}, allocated to each source-destination pair (i, j) must be obtained, and c) a way of placing the a_{ij} slots within the frame, for all i, j, must be determined. This problem is somewhat different than the tunable transmitter, fixed receiver one addressed in [14], but it is still a hard allocation problem. As in [14], we first construct sets X_c using a heuristic which attempts to balance the traffic load across all channels. Given these sets X_c, we now present a heuristic to obtain near-optimal schedules.

Let us consider channel λ_c in isolation. Since only nodes in X_c may transmit on λ_c, and the packet arrival at node i is described by σ_i, this is exactly the single-channel problem in [9]. It was shown there that the average packet delay is minimized when the fraction of time node i is permitted to transmit on channel λ_c is [9]

$$x_i = \sigma_i + \left(1 - \sum_{k \in X_c} \sigma_k\right) \frac{\sqrt{1-\sigma_i}}{\sum_{k \in X_c} \sqrt{1-\sigma_k}} \quad \forall i \in X_c. \quad (1)$$

Note that x_i are independent of the frame length M. Given x_i and M, we may obtain the number of slots per frame in

which $i \in X_c$ may access channel λ_c as

$$\lfloor Mx_i \rfloor < \alpha_i \leq \lceil Mx_i \rceil \quad \forall i \in X_c; \qquad \sum_{i \in X_c} \alpha_i = M. \quad (2)$$

We then need to determine how these α_i slots should be allocated for transmissions from i to each of the potential receivers. Note that the $N-1$ queues at i do not interact with each other; thus we have again an equivalent single-channel problem, which dictates that the fraction of time node i transmits to each destination be

$$y_{ij} = \sigma_i p_{ij} + \left(1 - \sum_{l=1}^{N} \sigma_i p_{il}\right) \frac{\sqrt{1-\sigma_i p_{ij}}}{\sum_{l=1}^{N} \sqrt{1-\sigma_i p_{il}}}$$
$$\forall i \in X_c,\ j = 1, \cdots, N. \quad (3)$$

Therefore, the number of slots allocated to the (i, j) pair should be

$$\lfloor \alpha_i y_{ij} \rfloor < a_{ij} \leq \lceil \alpha_i y_{ij} \rceil; \qquad \sum_{j=1}^{N} a_{ij} = \alpha_i \quad \forall i \in X_c. \quad (4)$$

Furthermore, no receiver should be assigned to receive in more than M slots in a frame

$$\sum_{i=1}^{N} a_{ij} \leq M \quad \forall j. \quad (5)$$

According to the results in [9], for each source-destination pair (i, j), the a_{ij} slots in which i transmits to j should be equally spaced within the frame. However, this is not possible in general. To overcome this problem, a golden-ratio policy was developed in [9], which requires that the frame length be a Fibonacci number. Our approach is to use the golden ratio policy to place the permissions within each channel independently of the others. However, considering channels in isolation may cause a receiver to be assigned to tune on two or more channels in the same slot. If this occurs, we must rearrange the schedule to remove these violations. To this end, we use algorithm REARRANGE, described in [13], with a worst case complexity of $O(N^2M^2)$. Our heuristic can be described by these steps.

Slot Allocation Heuristic (SAH):

1) If $C < N$, use the Weight Balancing Heuristic in [14] to determine the set of transmitters, X_c, $c = 1, \cdots, C$, that share each channel.
2) Select a Fibonacci number M, and obtain $a_{ij}(M)$ from (1)–(5). Let $c = 1$, and use the golden ratio policy [9] to place the a_{ij}, $i \in X_c$, slots in a frame for transmissions on channel λ_c. Repeat for $c = 2, \cdots, C$ to obtain an initial schedule, $S_0(M)$.
3) Run algorithm REARRANGE, described in [13], on S_0 to construct a schedule $S(M)$ with no destination conflicts.
4) Repeat Steps 2 and 3 for the next Fibonacci number, up to an upper limit, $M_{\max}$. Select the frame length, M, and schedule, $S(M)$, that yields the lowest average delay.

So far we have assumed that $\rho_i = 0$, i.e., that there is only single-destination traffic. Our approach, however, can be easily adapted to obtain schedules with unicast slots only, appropriate for networks in which Type 2 multidestination traffic is also offered. In this case, in addition to σ_i and p_{ij}, we have the multidestination traffic parameters ρ_i and q_{ig}. Since unicast slots will be used to carry multicast packets, a packet for multicast group g arriving at source i is copied into all queues j of i, such that $j \in g$. To construct a schedule for such a network we can apply the Slot Allocation Heuristic to an equivalent network with single-destination traffic only, such that its parameters, σ_i' and p_{ij}', account for both the single- and multidestination traffic of the original network ($\overline{\eta}$ is the average size of a multicast group)

$$\sigma_i' = \sigma_i + \overline{\eta}\rho_i \quad \forall i,$$
$$p_{ij}' = \frac{\sigma_i p_{ij} + \rho_i \sum_{g,\, j \in g} q_{ig}}{\sigma_i'} \quad \forall i, j. \quad (6)$$

B. Schedules with Broadcast Slots for Multidestination Traffic

Recall that only one node is allowed to transmit in a broadcast slot. Given $\rho_i, i = 1, \cdots, N$, the problem of obtaining an optimal broadcast schedule is then equivalent to the single-channel problem in [9]. Therefore, the fraction of time, z_i, that node i should be given permission to transmit is

$$z_i = \rho_i + \left(1 - \sum_{k=1}^{N} \rho_k\right) \frac{\sqrt{1-\rho_i}}{\sum_{k=1}^{N} \sqrt{1-\rho_k}} \qquad i = 1, \cdots, N. \quad (7)$$

z_i is independent of the frame length M. Given a Fibonacci number $M \geq N$ [9], we assign $b_i(M)$ broadcast slots to node i such that

$$\lfloor Mz_i \rfloor \leq b_i(M) \leq \lceil Mz_i \rceil \quad \forall i \text{ and } \sum_{i=1}^{N} b_i(M) = M. \quad (8)$$

We then use the golden-ratio policy [9] to place the b_i slots, $i = 1, \cdots, N$, within the frame.

REFERENCES

[1] M. H. Ammar and G. N. Rouskas, "On the performance of protocols for collecting responses over a multiple-access channel," *IEEE Trans. Commun.*, vol. 43, pp. 412–420, Feb. 1995.

[2] G. Armitage, "Support for multicast over UNI 3.0/3.1 based ATM networks," Internet Draft, draft-ietf-ipatm-ipmc-12.txt, Feb. 1996.

[3] M. Azizoglu, R. A. Barry, and A. Mokhtar, "Impact of tuning delay on the performance of banwidth-limited optical broadcast networks with uniform traffic," *IEEE J. Select. Areas Commun.*, vol. 14, pp. 935–944, June 1996.

[4] K. Bogineni, K. M. Sivalingam, and P. W. Dowd, "Low-complexity multiple access protocols for wavelength-division multiplexed photonic networks," *IEEE J. Select. Areas Commun.*, vol. 11, pp. 590–604, May 1993.

[5] M. Borella and B. Mukherjee, "A reservation-based multicasting protocol for WDM local lightwave networks," in *Proc. IEEE ICC'95*, 1995, pp. 1277–1281.

[6] ———, "Efficient scheduling of nonuniform packet traffic in a WDM/TDM local lightwave network with arbitrary transceiver tuning latencies," *IEEE J. Select. Areas Commun.*, vol. 14, pp. 923–934, June 1996.

[7] M.-S. Chen, N. R. Dono, and R. Ramaswami, "A media-access protocol for packet-switched wavelength division multiaccess metropolitan area networks," *IEEE J. Select. Areas Commun.*, vol. 8, pp. 1048–1057, Aug. 1990.
[8] R. Chipalkatti, Z. Zhang, and A. S. Acampora, "Protocols for optical star-coupler network using WDM: Performance and complexity study," *IEEE J. Select. Areas Commun.*, vol. 11, pp. 579–589, May 1993.
[9] M. Hofri and Z. Rosberg, "Packet delay under the golden ratio weighted TDM policy in a multiple-access channel," *IEEE Trans. Info. Theory*, vol. IT-33, pp. 341–349, May 1987.
[10] P. A. Humblet, R. Ramaswami, and K. N. Sivarajan, "An efficient communication protocol for high-speed packet-switched multichannel networks," *IEEE J. Select. Areas Commun.*, vol. 11, pp. 568–578, May 1993.
[11] B. Mukherjee, "WDM-based local lightwave networks part I: Single-hop systems," *IEEE Network Mag.*, pp. 12–27, May 1992.
[12] G. N. Rouskas and M. H. Ammar, "Multi-destination communication over single-hop lightwave WDM networks," in *Proc. IEEE INFOCOM'94*, June 1994, pp. 1520–1527.
[13] ———, "Analysis and optimization of transmission schedules for single-hop WDM networks," *IEEE/ACM Trans. Networking*, vol. 3, pp. 211–221, Apr. 1995.
[14] ———, "Minimizing delay and packet loss in single-hop lightwave WDM networks using TDM schedules," in *Proc. ICC'95*, June 1995, pp. 1267–1271.
[15] G. N. Rouskas and V. Sivaraman, "On the design of optimal TDM schedules for broadcast WDM networks with arbitrary transceiver tuning latencies," in *Proc. IEEE INFOCOM'96*, Mar. 1996, pp. 1217–1224.
[16] J. S. Turner, "New directions in communications (or which way to the information age?)," *IEEE Comm. Mag.*, vol. 24, pp. 8–15, Oct. 1986.
[17] ———, "The challenge of multipoint communication," in *Proc. 5th ITC Seminar*, May 1987, pp. 263–279.

George N. Rouskas (S'92–M'95), for a photograph and biography, see this issue, p. 356.

Mostafa H. Ammar (S'83–M'85–SM'95), for a photograph and biography, see this issue, p. 276.