Efficient Sequencing Techniques for Variable-Length Messages in WDM Networks

Babak Hamidzadeh, Ma Maode, and Mounir Hamdi, *Member, IEEE*

***Abstract*— Message sequencing and channel assignment are two important issues that need to be addressed when scheduling variable-length messages in a wavelength division multiplexing (WDM) network. Channel assignment addresses the problem of choosing an appropriate data channel via which a message is transmitted to a node. This problem has been addressed extensively in the literature. On the other hand, message sequencing which addresses the order in which messages are sent, has rarely been addressed. In this paper, we propose a set of scheduling techniques for single-hop WDM passive star networks, which address both the sequencing aspect and the assignment aspect of the problem. In particular, we develop two priority schemes for sequencing messages in a WDM network in order to increase the overall performance of the network. We evaluate the proposed algorithms, using analytical modeling and extensive discrete-event simulations, by comparing their performance with state-of-the-art scheduling algorithms that only address the assignment problem [9]. We find that significant improvement in performance can be achieved using our scheduling algorithms where message sequencing and channel assignment are simultaneously taken into consideration. This suggests that, when scheduling messages in WDM networks, one has to consider message sequencing, as well as channel assignment. As a result, we anticipate that this research will open new directions into the problem of on-line scheduling in WDM networks.**

***Index Terms*— On-line scheduling, optical networks, wavelength division multiplexing (WDM).**

I. Introduction

WAVELENGTH division multiplexing (WDM) is an effective way of utilizing the large bandwidth of an optical fiber. By allowing multiple messages to be transmitted in parallel, on a number of channels, this technique has the potential to significantly improve the performance of optical networks. The nodes in such a network can transmit and receive messages on any of the available channels using one or more tunable transmitter(s) and/or tunable receiver(s). Several topologies have been proposed for WDM networks [1], [2], a popular one being the single-hop, passive star-coupled topology [3].

To unleash the potential of single-hop, WDM passive star networks, efficient access protocols and scheduling algorithms are needed to allocate and coordinate system resources optimally, while satisfying message and system constraints [1]. Most of these protocols and algorithms can be divided into two main classes, namely preallocation-based [3]–[6] and reservation-based [7]–[11] techniques. Preallocation-based techniques use all channels of a fiber to transmit messages. These techniques assign transmission rights to different nodes in a static and predetermined manner. Reservation-based techniques allocate a channel as the control channel, to transmit global information about messages to all nodes in the system. Once such information is received, all nodes invoke the same scheduling algorithm to determine when to transmit/receive a message and on which data channel. Reservation-based techniques have a more dynamic nature and assign transmission rights based on the run-time requirements of the nodes in the network. In this paper, we focus our attention on reservation-based techniques.

Most of the scheduling algorithms proposed for reservation-based techniques can only schedule fixed-length packets for transmissions. Recently, many researchers have relaxed this constraint by allowing their scheduling algorithms to schedule variable-length messages [9], [12]–[14]. As a result, these variable-length scheduling algorithms are more general than fixed-length scheduling algorithms and adapt better to various traffic characteristics (e.g., bursty). We adopt the same strategy in this paper by allowing our scheduling algorithms to handle variable-length messages. There are two fundamental aspects that a variable-length message scheduling algorithm should efficiently solve, namely, channel assignment and message sequencing. The assignment aspect of a scheduling algorithm addresses the problem of selecting an appropriate channel and a time slot on that channel to transmit a message, while message sequencing addresses the order in which messages are selected for transmission. The assignment aspect of this problem has been addressed extensively in the literature. The sequencing aspect of this problem, however, has not received much attention. In particular, all the above proposed variable-length messages scheduling algorithms schedule messages individually and independently of one another [9], [12]–[14]. These scheduling algorithms attempt to schedule each message *immediately* after receiving the control information about that message. However, useful information for improving the schedule quality exists when a batch of messages are considered for scheduling together rather than individually, as will be shown in this paper.

In this paper, we propose and evaluate a set of scheduling techniques that address the sequencing, as well as the assignment aspect of the scheduling problem. Our techniques are more globally optimizing than the existing approaches,

Manuscript received December 5, 1997; revised March 19, 1999.

B. Hamidzadeh is with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, B.C. V6T 1Z4 Canada.

M. Maode and M. Hamdi are with the Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.

Publisher Item Identifier S 0733-8724(99)04517-X.

since they not only share global information about each message among receiving and transmitting nodes, but also consider multiple messages from different transmitting nodes simultaneously when scheduling. The scheduling algorithms are invoked after control information about multiple transmitting nodes is received by all the nodes in the network. This approach not only provides each node with more global information about messages, but also reduces the frequency at which the scheduling algorithms are invoked. The reduction in invocation frequency results in lower scheduling overheads and permits longer times for transmitter–receiver tuning.

As part of our scheduling techniques we adopt policies, such as *longest-job-first* (LJF) and *shortest-job-first* (SJF), that impose a priority on the order in which messages are transmitted. LJF is a priority scheme that is used in the parallel processing community to balance the load among processing elements [15], [16]. We adopt this policy to balance the transmission load among the communication channels of our network. Despite its load balancing capabilities, LJF is known to result in relatively poor average delays among messages (or jobs) in a queue. SJF, on the other hand, is a priority scheme that results in reduced average delays. This scheme, however, performs poorly in terms of balancing the load among the communication channels. Studying the performance tradeoffs between these two priority schemes is another aspect of the scheduling problem which is addressed in the paper. As we shall see, these simple schemes impose strong implications on performance. The combination of simplicity and superiority in performance makes the proposed techniques a powerful and viable choice for WDM networks.

We have developed a theoretical model to analyze the performance of the techniques discussed in this paper. In addition, we evaluated our techniques by comparing their performance with a recently proposed scheduling algorithm [9] using extensive discrete-event simulations. The results of these experiments demonstrate the significant improvements that can be obtained by using techniques that address sequencing and assignment simultaneously. The experiments also show a comparison and tradeoff between using techniques that are better for load balancing (e.g., LJF) with techniques which are more suitable for reducing average delay (e.g., SJF).

The remainder of this paper is organized as follows. Section II specifies our WDM system model and the scheduling problem to be addressed. Section III discusses our scheduling techniques and Section IV provides a detailed example to demonstrate the operations involved in the proposed techniques. Section V provides an analytical model of the proposed techniques. Section VI provides an experimental evaluation of these techniques' performance. Finally, Section VII concludes the paper with a summary of the results and a discussion of our future work.

II. WDM System Model and the Scheduling Problem

As mentioned previously, in this paper we consider message transmission in a single-hop, WDM optical network, whose nodes are connected via a passive star coupler. The star coupler supports C channels and N nodes in the network. $C-1$ channels, referred to as data channels, are used for message transmission. The other channel, referred to as the control channel, is used to exchange global information among nodes about the messages to be transmitted. The control channel is the basic mechanism for implementing the reservation scheme. Each node in the network has two transmitters and two receivers. One transmitter and one receiver are fixed and are tuned to the control channel. The other transmitter and receiver are tunable and can tune into any of the data channels to send and receive data on those channels. This is similar to the network proposed in [9].

The nodes are assumed to generate messages with variable lengths which can be divided into several equal-sized packets. The basic time interval on the data channels is the transmission time of one packet. In our model, we assume that the basic transmission unit is one message. The nodes are divided into two nondisjoint sets of source (transmitting) nodes s_i and destination (receiving) nodes d_j. A queue for the messages waiting to be transmitted is assumed to exist at each source node s_i.

A time division multiple access (TDMA) protocol is used on the control channel to access that channel. According to this protocol, each node can transmit a control packet during a predetermined time slot. The basic time interval on the control channel is the transmission time of a control packet. N control packets make up one control frame on the control channel. Thus, each node has a corresponding control packet in a control frame, during which that node can access the control channel. The length of a control packet is a system design parameter and depends on the number of messages l about which each node is allowed to broadcast control information, and the amount of control information about each message (e.g., the address of the destination node, message length).

The existence of message queues at each source node and the value of parameter l hold important implications for the design of the scheduling algorithms in our WDM model. Values greater than one for the parameter l signify a situation in which a source node s_i can transmit information about multiple messages in its queue to all nodes through a control packet i in a control frame. This enables the nodes to schedule l messages per source node in one scheduling invocation. Increasing the number of messages that are considered in each scheduling phase facilitates a more globally optimizing approach to the problem.

Fig. 1 demonstrates some of the basic concepts used in our model. In this model, we have ignored the transmitter and receiver tuning times. The reason for doing so is to provide clear insight into the salient features of our scheduling techniques, which in principle, are independent of tuning times. The model and the proposed scheduling techniques can easily be extended to consider tuning times without loss of generality, as was illustrated in [9]. We expect a similar performance from our techniques in a model that considers tuning times.

Simultaneous access and transmission on multiple data channels create a parallelism in message transmission. Transmitting variable-length messages in parallel may result in variable channel utilization on different data channels, which

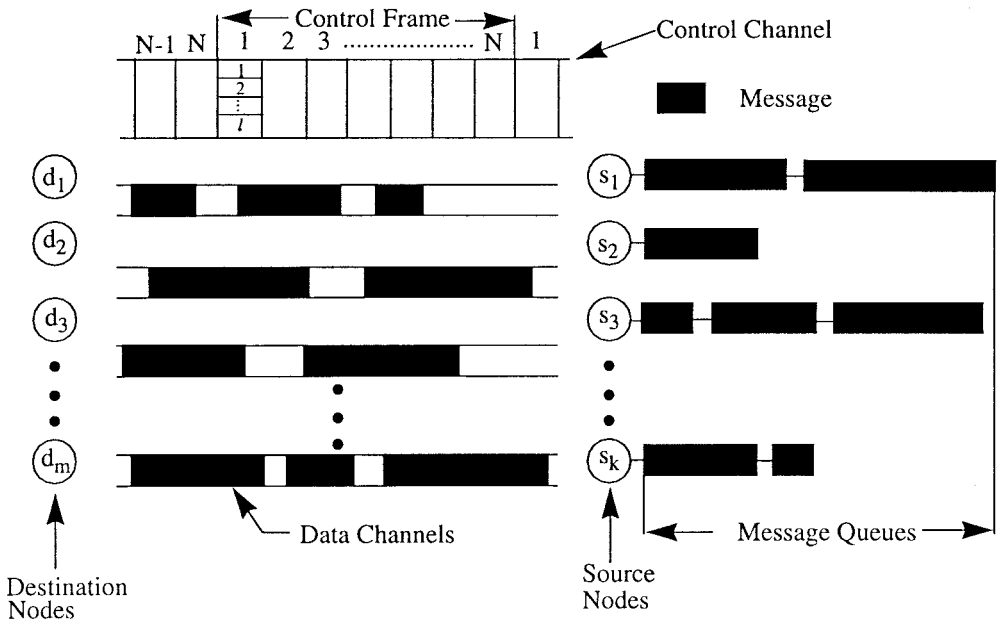

Fig. 1. Data and control channel configuration, and message queues at transmitting nodes.

in turn, leads to uneven schedule finish times on different channels. The completion time of all message transmissions is the time at which the last message is transmitted and is minimized by equitably dividing the loads among data channels, such that the last scheduled messages on all data channels reach their destinations at about the same time. This concept is widely known as *load balancing* in the parallel processing community. Balancing the load among data channels constitutes one of our objectives, since it is expected to reduce channel access delay and improve throughput. Another important concept to note when sending variable-length messages is the channel access delays caused on average for all messages, due to transmission of long messages. Minimizing the average delay constitutes another one of our objectives in this paper.

III. Scheduling Techniques

In this section, we discuss the basic steps of our scheduling techniques and discuss some of their performance tradeoffs. During the transmission of a control frame, each source node s_i sends a control packet during time slot i on the control channel to all other nodes. The control packet contains information about one (at the head of s_i's message queue) or more messages it intends to transmit. The larger the number l of messages that are represented in a control packet, the more globally optimizing our scheduling algorithms will be. Larger values of l result in longer durations, but less frequent invocations of the scheduling algorithms.

After $R + F$ time units, where R is the round-trip propagation delay between a node and the star coupler and F is the time duration of a control frame, all the nodes in the network will have information contained in a control frame about messages to be transmitted. At this point, an identical copy of a distributed scheduling algorithm is invoked by all nodes, so as to assign the messages represented in the control frame to the appropriate data channels to be transmitted at a point in time. The technique for assignment of data channels and transmission time may vary based on different models. Examples of such techniques that are receiving attention are EATS, CDS, and TTAS as proposed in [9]. EATS (*Earliest Available Time Scheduling*) is a basic channel assignment algorithm that does not consider tuning time as part of its criteria for assigning messages to channels. CDS and TTAS, on the other hand, do consider tuning time. For the sake of simplicity, and to be able to clearly illustrate the importance of sequencing messages in these networks, we adopt EATS as our basic channel assignment mechanism. However, the choice of channel assignment technique in our approach is independent of our sequencing algorithms. In other words, our sequencing mechanisms work as well had we adopted CDS or TTAS. EATS assigns a message to the data channel that has the earliest available time among all the channels. Once the data channel is assigned, the message is scheduled to transmit as soon as that channel becomes available. This algorithm effectively resolves destination conflicts. Its conflict-resolution characteristics are conveniently inherited by our algorithms, as well.

At two points in our model, it is possible to sequence messages according to a priority scheme, before assigning them to a data channel for transmission. One such point is at the message queues associated with each of the source nodes. At certain intervals, an algorithm can be invoked to sequence the messages at each source node, before a control packet is sent out about such messages. Sequencing messages at this point imposes an order on the choice of messages about which control information is transmitted. Another point at which messages can be sequenced for transmission is after the entire control frame has been received by all nodes. As part of the distributed scheduling algorithms in the nodes, we can apply a sequencing mechanism to the messages represented in a control frame. This sequencing mechanism will impose an order in which messages are assigned to channels.

We have selected two priority schemes for sequencing messages in our scheduling algorithms, namely, the SJF and the LJF schemes. By scheduling shorter messages first, SJF is expected to reduce average delays. SJF's ability to reduce average delays has been demonstrated. In an environment where messages can be transmitted in parallel on different data channels, however, SJF is expected to result in a poorly balanced load among different channels. This is because the larger messages that are scheduled last may have large differences in size, which will lead to a coarser schedule with uneven loads among channels. This is why we have chosen LJF as an alternative priority scheme to see the tradeoff between load balancing and reducing average delays in our algorithms. LJF is expected to balance the load by first scheduling long messages on data channels and then filling the uneven loads with smaller messages.

Sequencing messages at the source-node message queues and/or messages represented in the control frame, can lead to a number of different scheduling policies, some of which we have adopted and evaluated. In the following subsections we discuss some of these strategies and their characteristics.

A. Frame Scheduling

Using this strategy, each message queue, at the source nodes, is maintained as a first-come first-served (FCFS) queue. During each time slot i, control information about the message at the head of s_i's queue is placed in packet i of a frame. After all packets of a frame reach all nodes in the network, a sequencing algorithm based on a priority scheme (e.g., SJF or LJF) is called to sort the messages represented in that frame according to their priorities. Once the order of message transmissions is determined, a channel assignment algorithm (e.g., EATS) is invoked to assign the channel and time of transmission. The source nodes will then know on which channel to transmit the message at the head of their message queues and at what time. The receiver nodes will also know to which channel they should tune and at what time to receive the appropriate message.

Prioritizing message transmissions in frame scheduling does not lead to starvation, since this prioritization takes place in batches all of whose messages receive service before the next batch of messages is scheduled/serviced. A delay (e.g., $c/\max\{\lambda_i\}$, where c is a constant and λ_i are the message arrival rates at the source nodes s_i) can be introduced in transmitting control information at each node to allow messages to arrive at the message queues, so that most of the control packets in a control frame are likely to carry information about the messages to be transmitted. This will allow the scheduling algorithms to be applied to a larger number of messages and, thus, obtain better quality schedules. This gain in schedule quality is traded off with the artificial delay that is introduced. Note that as arrival rates increase, the need for such a delay diminishes and the above suggested formula for calculating this delay automatically reduces the delay.

It is clear that the frame scheduling technique does not need to assume the existence of message queues at the source nodes for its correct operation. This technique can work with a model that allows at most one arrived message to be represented at each source node. As we shall see, the techniques presented next do depend on the existence of message queues in order to operate correctly.

B. Frame-and-Queue Scheduling

Using this strategy, sequencing is done at two points; once at the message queues of the source nodes and once at the time the messages of a control frame are scheduled. The message queues at the source nodes are maintained according to some priority scheme (e.g., SJF or LJF). Thus, the head of each queue contains the message m with the highest priority among all the messages that have arrived at a source for transmission. During time slot i, therefore, control information about message m will be placed in the appropriate control packet of the control frame. Once all packets of the frame have reached all nodes, a sequencing algorithm (based on the same priority scheme as the one at the message queues) is applied again to sequence the messages represented in that frame.

A point to note is that the frame-and-queue scheduling technique may lead to starvation for some messages at the message queues. This is because a higher priority message can always arrive into a message queue and replace existing lower priority messages at the head of the queue. Thus, some form of aging mechanism should be adopted for this kind of technique, to increase the priority of messages as they stay longer in the message queues.

As in frame scheduling, a delay can be introduced before control information is transmitted to allow messages to arrive at the source nodes. The constant c in the formula $c/\max\{\lambda_i\}$ can represent the number of messages that we expect (or desire) to arrive at the message queues.

C. Multiple-Messages-per-Node Scheduling

This technique attempts to do scheduling at a more global level than the previous two approaches. To do this, it represents a number $l(l \geq 1)$ of messages in each control packet. Thus, control information about multiple messages at each source node's message queue can be placed in each of the corresponding control packets of a frame. This technique performs sequencing once at the time the frame has reached all the nodes. The sequencing algorithm is applied to all messages of the frame and an order is imposed on the messages according to some priority scheme, as was discussed earlier. After scheduling, each source node will know which message in its queue is to be transmitted next. The source nodes will also know on which channel to transmit the message and at what time.

Like the frame scheduling technique, the multiple-messages-per-node technique is free from starvation. This is again attributed to the fact that this technique schedules messages in independent batches. Since this technique schedules more messages in each scheduling phase than the previous two approaches, its scheduling time is higher. The frequency of scheduling invocations, on the other hand, is lower since a larger number of messages is scheduled each time. The quality

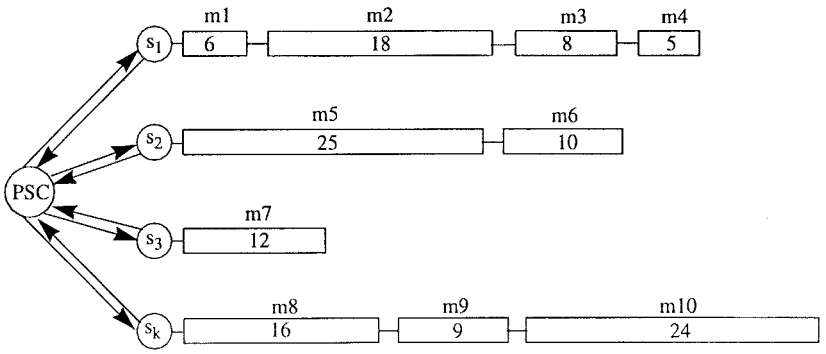

Fig. 2. Message queues at transmitting nodes.

of schedules are expected to improve as l increases. This is mainly because a larger number of messages are compared against one another using the priority queuing schemes.

The choice of l is important and can have implications on the performance of the algorithm and its scheduling overhead. Assuming that messages arrive in each source node s_i's queue at a rate λ_i, it might be beneficial, as a rule of thumb, to choose larger values for l as λ_i becomes larger. The upper bound on the value of $l$can be determined by the maximum time the scheduling algorithm is allowed to consume in each invocation. The maximum time of scheduling messages of a frame can be bounded by $R+F$ time units which is the time it takes for the next control frame to arrive at a node and is the time for the next invocation of the scheduling algorithm.

If the arrival rates λ_i are small, a delay can be introduced in transmitting control information at each node so as to allow more messages to arrive at the source node queues. A formula similar to the one suggested for the other techniques (i.e., $c/\max\{\lambda_i\}$) can be used to determine the length of this delay. In this formula, $c = l$, would allow an expected l number of messages to arrive at the nodes before the control information is transmitted. This value raises the probability that each frame transmits control information at that frame's maximum capacity.

IV. Examples and Illustrations of the Scheduling Techniques

In this section, we discuss the details of the proposed scheduling techniques in the context of an example. Fig. 2 shows a network of four nodes and a set of ten messages to be transmitted at the source nodes s_i. The boxes and the numbers inside them represent messages and their lengths, respectively.

We start our discussion by observing the behavior of the EATS algorithm [9] in this example. As mentioned earlier, this algorithm is a basic channel assignment algorithm and does not sequence messages in any particular order. EATS (also referred to as first-control-packet first-served (FCPFS) in this paper) starts by assigning the message represented in the first control packet of a frame to the data channel with the earliest available time. It then proceeds to assign the message represented by the second control packet of a frame to the next channel with the earliest available time, and so on.

Fig. 3 shows each of the control frames, and their packets that transmit global information about the messages. The

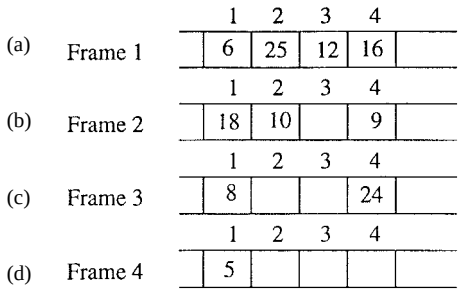

Fig. 3. Control frames during EATS/FCPFS, and frame scheduling.

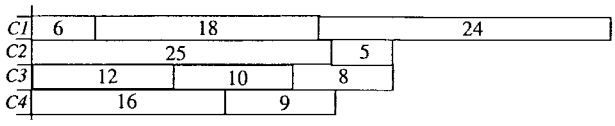

Fig. 4. Schedule using EATS/FCPFS.

numbers on top of the control packets of each frame designate the corresponding source nodes and the numbers inside the packets indicate the message lengths. Each control packet in a frame transmits information about the message at the head of the queue at its corresponding source node. For example, the frame shown in Fig. 3(a) contains control information about messages m1, m5, m7, and m8 in control packets 1, 2, 3, and 4, respectively.

In this example, we assume that the data channels are initially idle. EATS can schedule each message after its corresponding control packet reaches all the nodes, or it can schedule all the messages represented by a frame after the entire frame has been transmitted to all the nodes. The effect of these alternatives is semantically the same and leads to the same schedules. As Fig. 4 shows, EATS initially provides the schedule {(m1, C1), (m5, C2), (m7, C3), (m8, C4)} which assigns message m1 to data channel C1, message m5 to data channel C2, message m7 to data channel C3, and message m8 to data channel C4. Message m2 of frame 2 is then assigned to channel C1 which has the earliest available time. Similarly, m6 is assigned to the next channel with the earliest available time, namely C3. Message m9 is then assigned to C4, since this channel has the earliest available time at the time of assignment. EATS continues to assign messages represented in frames 3 and 4 in a similar fashion. The result of this

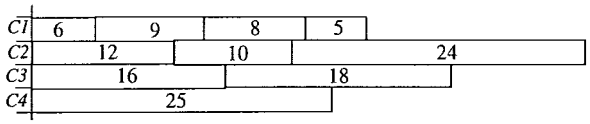

Fig. 5. Schedule using frame scheduling.

channel assignment leads to the schedule shown in Fig. 4. The average delay of the 10 messages in this example using EATS is calculated as: $(6 + 25 + 12 + 16 + 24 + 30 + 22 + 25 + 48 + 30)/10 = 238/10 = 23.8$.

The frame scheduling algorithm uses the same control information as EATS, as shown in Fig. 3. This algorithm invokes the distributed scheduling algorithm after an entire control frame has been transmitted to all the nodes. After receiving control frame 1, the algorithm first sorts the messages represented in the frame according to some priority scheme (i.e., SJF in this example) and then uses the EATS technique to assign messages to data channels. Based on this description, messages m1, m5, m7, and m8 are sorted according to SJF into m1, m7, m8, and m5, in increasing order of their lengths, namely 6, 12, 16, and 25. They are then assigned to channels C1, C2, C3, and C4, represented by the schedule {(m1, C1), (m7, C2), (m8, C3), (m5, C4)}. Messages m2, m6, and m9 in frame 2 are scheduled similarly by first being sorted into m9, m6, and m2. At this time, C1 has the earliest available time (i.e., 6), so it will be assigned to transmit m9 at time 6. Similarly, m6 and m2 are assigned to C2 and C3, with the earliest available times, respectively. The result of scheduling all messages using frame scheduling with SJF priority scheme is shown in Fig. 5. The average delay of the 10 messages using frame scheduling with SJF is calculated as: $(6 + 12 + 16 + 25 + 15 + 22 + 34 + 23 + 46 + 28)/10 = 227/10 = 22.7$. It is evident from the result of this example that frame scheduling improves average delay when compared to the EATS algorithm. This is attributed to frame scheduling's ability to address both sequencing and assignment simultaneously so as to better optimize the results.

Next we discuss the result of applying the frame-and-queue scheduling technique to our example. Using this technique, the message queues at the source nodes are first sorted according to a priority scheme (i.e., SJF in this example). The result of this step of the algorithm is shown in Fig. 6. Next, the control frames are transmitted as shown in Fig. 7. After receiving control frame 1, the algorithm first sorts the messages m4, m6, m7, and m9 represented in the frame according to some priority scheme (i.e., SJF) to obtain m4, m9, m6, m7. It then assigns them to data channels according to their earliest available time to obtain the schedule {(m4, C1), (m9, C2), (m6, C3), (m7, C4)}. Similarly, {(m1, C1), (m8, C2), (m5, C3)} results from scheduling messages represented in frame 2. The result of scheduling all messages using frame-and-queue scheduling with SJF is shown in Fig. 8. The average delay of the ten messages using this technique is calculated as: $(5 + 9 + 10 + 12 + 11 + 25 + 35 + 36 + 19 + 37)/10 = 199/10 = 19.9$.

We can see from the result of this example that frame-and-queue scheduling improves average delay when compared to frame scheduling. This is attributed to the fact that the former algorithm uses global information about a larger number of messages than the latter algorithm. Note that in the frame-and-queue algorithm, the messages are still not compared directly with one another. Rather, it sorts them in two disjointed phases. Next we shall see how the multiple-messages-per-node technique uses a more globally optimizing strategy to improve the results further.

As is shown in Fig. 9, the multiple-messages-per-node technique transmits control information about multiple messages in one frame. In this example, control information about all the messages is contained in one frame. The value of parameter $l = 4$ as each control packet has the ability to represent up to four messages originated at its corresponding source node. Once the entire frame has been received by all nodes, the scheduling algorithm collectively sorts all messages m1–m10, represented in a frame, according to the SJF priority scheme to obtain the sequence m4, m1, m3, m9, m6, m7, m8, m2, m10, m5.

Next, the messages are assigned, in order of their sequence, to data channels with the earliest available times to obtain the schedule shown in Fig. 10. The average delay of the ten messages using this technique is calculated as: $(5 + 6 + 8 + 9 + 15 + 18 + 24 + 27 + 39 + 43)/10 = 194/10 = 19.4$. This result is the best among all the candidate sequencing algorithms in this example. This is expected, since the multiple-messages-per-node technique compares and schedules a larger number of messages in an aggregate manner. In the following sections, we validate our expectations further regarding the general performance of these algorithms, through analytical modeling and in a number of experimental studies.

V. Analytical Model

In this section, we present a simple analytical model for our WDM network that follows and extends the analytical model originally proposed in [9]. This model and that of [9] have been simplified using some underlying assumptions in order to make the WDM model mathematically tractable. Nevertheless, it can be used to reveal some important insights into our scheduling algorithms, and to make sure that they conform to our simulation results as well. The performance metric of our interest in this model is the average message delay in the network.

In order to make our WDM model mathematically manageable, several assumptions have been adopted as follows:

1) the tuning time is negligible;
2) we have a finite message population, M, at the head of each node's queue;
3) the message generation process at each node is a Poisson process with a mean arrival rate of λ;
4) a message transmitted by a node is destined to every other node with equal probability;
5) for each of the nodes i, the message length is exponentially distributed with a mean value of $1/\mu_i'$;
6) for each of the nodes, the probability that each node has one message is approximately $1/N$.

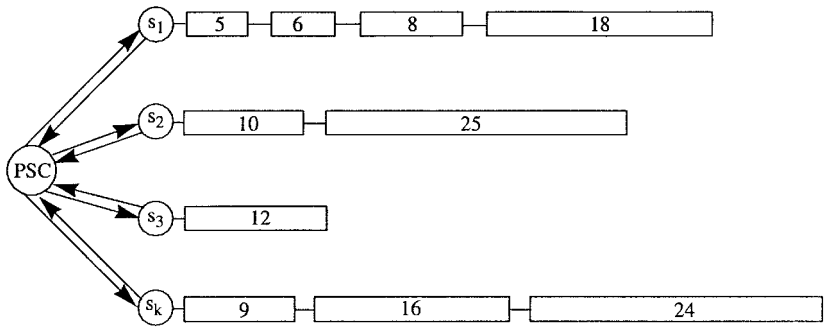

Fig. 6. Message queues after sorting; done as part of frame-and-queue scheduling.

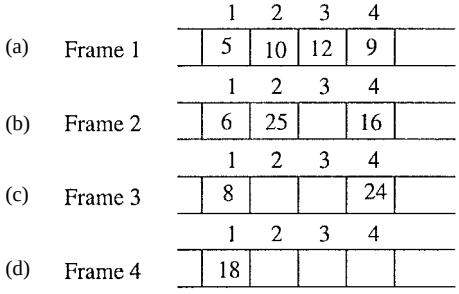

Fig. 7. Control frames during frame-and-queue scheduling.

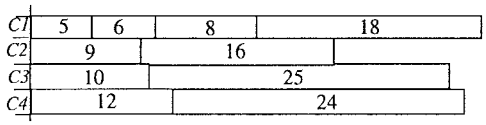

Fig. 8. Schedule using frame-and-queue scheduling.

The frame scheduling algorithms introduced in the previous sections provide mechanisms to sequence the messages according to length-based priorities assigned to each message (i.e., SJF or LJF). As a result, a WDM system that adopts one of these algorithms can be modeled as an M/G/1 with a priority queuing system [17]–[19]. The population of the system queue in this model is bounded by the number of nodes, since we consider the system queue as being composed of every first message at each node (i.e., the head of every node's queue). The servers of the queue can be considered as the set of data channels in the system with different service rates. The service rate of a channel depends on the messages it serves, combined with the restriction on message destinations.

The message population is limited to one per node, with the same arrival rate λ and probability $1/N$. The arrival rate of the system can be approximated by $(N-k)\times\lambda/N$, where the system state k is the number of messages in the system and $k\in\{0,1,\ldots,N-1\}$.

The service rate, which is the inverse of the service time of the system server, can be considered as a function of two factors. One is the mean message service rate μ_i' of the message with the ith priority. The other is $a_i(k)$ which denotes the probability that out of k messages, i messages are destined to different nodes. This term signifies that the destination of a message plays a role in determining the service rate of the

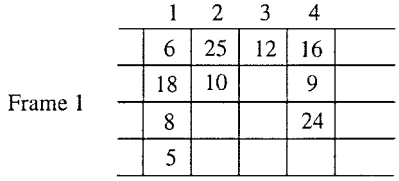

Fig. 9. Control frame during multiple-messages-per-node scheduling.

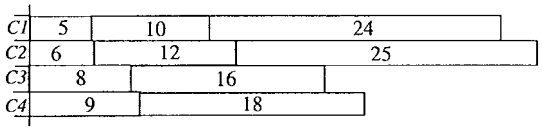

Fig. 10. Schedule using multiple-messages-per-node scheduling.

system server. The service rate μ_k of the server, when kth priority message is served in the system, can be expressed by the following [18], [19].

1) When $k<C-1$:

$$\mu_k=\sum_{i=1}^{k}\mu_i'\times i\times a_i(k).$$

2) When $k\geq C-1$:

$$\mu_k=\sum_{i=1}^{C-1}\mu_i'\times i\times a_i(C-1).$$

These formulas demonstrate the effect of the number of channels on the service rate. According to assumption 4, $a_i(k)$ can be computed as follows [9]:

$$a_i(k)=\frac{\binom{N}{i}\times i!\times S(k,i)}{N^k}$$

where $S(k,i)$ is the Stirling number.

The system traffic intensity or the load P_k, which is the ratio of the messages arrival rate to the service rate of the system, when kth priority message is served by the server, can be expressed by the following relationship:

$$P_k=\frac{\lambda_k}{\mu_k}=\frac{(N-k)\times\lambda/N}{\sum_{i=1}^{k}\mu_i'\times i\times a_i(k)}$$

Applying Little's result [18] to our M/G/1 priority queuing system, we can obtain the relationships between the average delay D_k of the kth priority message, and the average waiting time DW_k of the kth priority message in the queue. Finally, we can derive the average delay time D of all the messages in the system. In particular, the waiting time D_k of the kth priority message can be expressed as follows:

$$DW_k = \frac{\sum_{i=1}^{k} \frac{p_i}{u_i}}{2 \times \left(1 - \sum_{i=1}^{k} p_i\right) \times \left(1 - \sum_{i=1}^{k-1} p_i\right)}.$$

The delay time D_k of the kth priority message can be obtained by adding that message's service time $1/\mu_k$, to the waiting time DW_k of the kth priority message in the queue:

$$D_k = \frac{1}{\mu_k} + DW_k.$$

Based on the above formula, we can calculate the average delay time of all messages in the system as

$$D = \frac{1}{N} \times \sum_{k=1}^{N} D_k.$$

This model is general in that it is based on the notion of priority queueing which can accommodate a variety of priority schemes (e.g., SJF and LJF). As a result, it can be used to give us insights into the performance of any proposed priority scheme for sequencing messages in our WDM model. Even though this model has been made simple intentionally, it nevertheless agrees fairly well with our discrete-event simulation results as will be shown in the next section.

VI. Experimental Evaluation

In this section, we discuss the results of a set of experiments that evaluate the performance of the proposed scheduling techniques and also compare them with the scheduling scheme adopted in [9]. The experiments were conducted using a discrete-event simulator. In one experiment, we study the effect of message arrival rates at the source nodes on the performance of our WDM network. In another experiment, we investigate the effect of varying the number of channels on the performance of our WDM network. We also compare the results obtained from theoretical analysis with those of the simulation experiments, to validate the experiments and the mathematical model. The following subsections provide a discussion of the design of these experiments and their results.

A. Experiment Design

The parameters involved in the design of our WDM system include the number of nodes, which was chosen to be 50, and the number of channels, which ranges from 4 to 10. Tuning latencies were not considered in these experiments to focus the results on the salient features of the proposed scheduling algorithms. As was pointed out before, the techniques can easily be extended to account for tuning latencies. In Section III, we discussed the possibility of introducing a delay to allow message queues to fill up before the scheduling algorithms are invoked. We have not considered the effects of such delays in the results reported in this paper. We plan to study these effects, as part of our future work. Round-trip propagation delay is another system parameter which was set to 10 in the experiments.

Message lengths vary according to an exponential distribution with a mean of 20 packets per message each of which is a single time unit long. An exponential message arrival rate across all the nodes was considered, which ranges from 0.002 to 0.005 messages per unit time for each node in the network. Destination nodes for messages were chosen according to a uniform probability distribution. The behavior of the candidate algorithms was observed over a simulation period of 100 000 time units. Each point in the performance graphs is the average of ten independent runs. A metric of performance in the experiments is *average delay*, defined as the average duration between the time a message is scheduled for transmission and the time at which it is received at its destination. Another metric of performance is *throughput*, which is defined as the number of packets that are transmitted per unit of time.

The channel assignment strategy chosen for all candidate algorithms is the EATS technique, as proposed in [9]. This technique assigns a message to the data channel with the earliest available time. The candidate algorithms for the performance-comparison experiments were first-control-packet first-served (FCPFS), Frame scheduling with SJF (F-SJF) and with LJF (F-LJF) priority schemes, Frame-and-Queue scheduling with SJF (FQ-SJF) and with LJF (FQ-LJF) priority schemes, and multiple-messages-per-node scheduling with SJF (MMN-SJF) and with LJF (MMN-LJF) priority schemes. The operation of F-SJF, F-LJF, FQ-SJF, FQ-LJF, MMN-SJF, and MMN-LJF were discussed in previous sections. The value of parameter l in MMN algorithms was set to five. The FCPFS algorithm is the basic algorithm against which our proposed algorithms are compared which was originally given in [9]. This algorithm does not sequence messages in any particular order and assigns them to the data channels according to the index of their control packets in the control frame. This means that a message originated at source node s_1 whose corresponding control packet in the control frame is packet 1, will be scheduled before a message originated at source node s_2 with the second control packet as its corresponding slot in the control frame.

B. Experimental Results

Fig. 11 compares the average delay of the algorithms under varying loads (arrival rates) in a system with four channels. As the figure shows, the algorithms which perform both sequencing and assignment (e.g., F, FQ, and MMN) significantly outperform those which perform only assignment (e.g.,

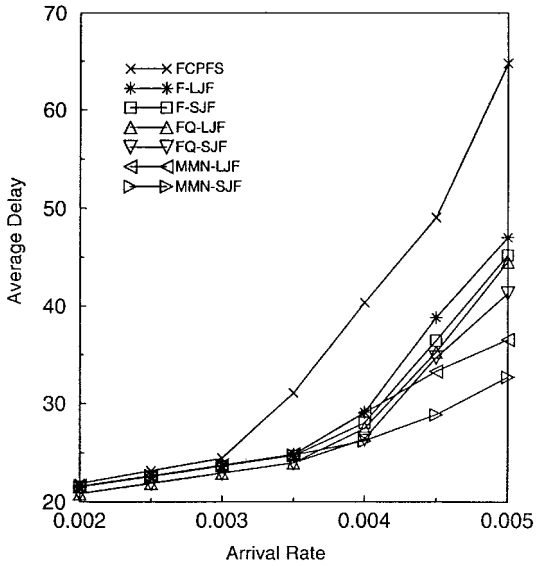

Fig. 11. Comparison of average delays versus average arrival rates.

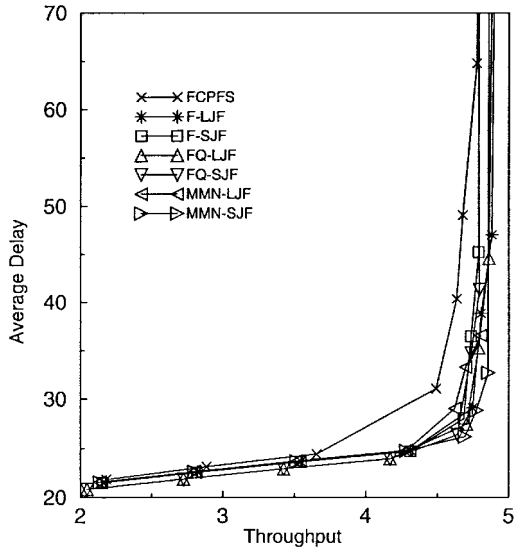

Fig. 13. Comparison of average delays versus throughputs.

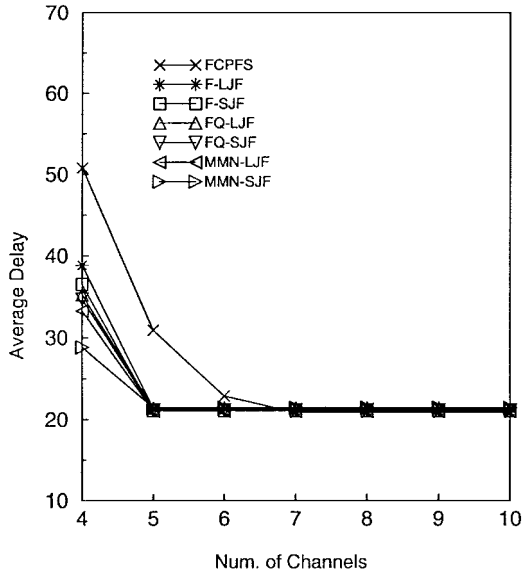

Fig. 12. Comparison of average delays versus number of channels.

FCPFS), as arrival rates increase. The figure also reveals that as the degree of globally optimizing behavior increases, the algorithms' performance consistently improves (i.e., MMN outperforms FQ and FQ outperforms F).

The F algorithms can outperform FCPFS by as much as 50%. The FQ and MMN algorithms outperform FCPFS by as much as 60 and 100%, respectively. This is an affirmation of the importance of efficiently sequencing messages in variable-length message scheduling algorithms. As a general trend, algorithms using the SJF priority scheme perform slightly better, in terms of reducing average delay, than those employing LJF.

Fig. 12 shows the average delay of the different algorithms as the number of channels vary. The load in this set of experiments was set to 0.0045. The figure shows that the sequencing-and-assignment techniques outperform the FCPFS algorithm significantly, when the number of channels is small. The figure also shows that the margin of performance narrows among different algorithms as the number of channels increases.

Based on the current state of technology the number of channels in a WDM network is expected to be much smaller than the number of nodes. Under such circumstances, our results show that the sequencing-and-assignment scheduling algorithms can significantly improve performance in such networks, by exploiting the limited resources effectively. When the number of channels is small, F and FQ algorithms can outperform FCPFS by as much as 30%. The MMN algorithms, on the other hand, outperform FCPFS by as much as 75–100%, when the number of channels is small. Once again, we observe that the algorithms using the SJF priority scheme perform slightly better, in reducing average delay, than those which use the LJF scheme.

In Fig. 13, the average delay of the candidate algorithms is plotted against their throughput in a system with four channels. As the figure shows, the average delay is consistently lower for the F, FQ, and MMN algorithms, at different throughput levels, than for the FCPFS algorithm. Finally, Fig. 14 compares the results of the analytical model developed in the previous section with those obtained through our discrete-event simulations in a system with four channels. The analytical results agree well with the experimental results and confirm that F-SJF demonstrates improved performance with respect to FCPFS. These results also verify the accuracy of the analytical model and its usefulness in predicting the performance of various priority schemes for sequencing messages.

It was shown in [9] that EATS (or FCPFS) outperforms most state-of-the-art scheduling algorithms for WDM networks, such as [3], [4]. For that reason, it has been receiving considerable attention from the research community. Needless to say, we expect our algorithms which consider message sequencing mechanisms, as well as channel assignment, to outperform those algorithms by an even wider margin.

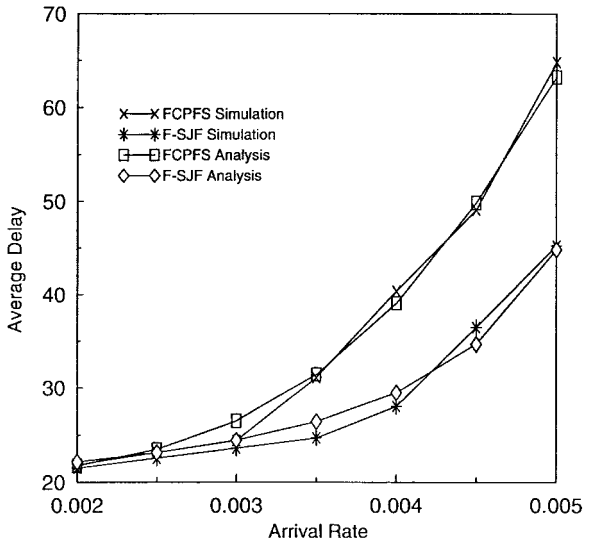

Fig. 14. Comparison of analysis and simulation results.

VII. Conclusion

In this paper, we proposed a set of reservation-based techniques for scheduling variable-length messages in a single-hop, WDM passive star network. Unlike many existing reservation-based techniques, the proposed techniques address both message sequencing and channel assignment aspects of the scheduling problem simultaneously. Two priority schemes, namely SJF and LJF, were used for imposing an order on the message sequences so as to improve the overall performance. The SJF scheme reduces the average message delay, but performs poorly in balancing message loads among data channels. The LJF, on the other hand, balances the message loads among data channels but it performs poorly in reducing average delay among messages in a queue. We formulated a mathematical model to study the performance of the proposed techniques. We also evaluated the performance of the proposed techniques and the tradeoffs between the priority schemes in a number of experiments. These experiments compared the proposed algorithms with another scheduling technique which only addresses channel assignment problem (no message sequencing) [9]. The results of our experiments show significant improvements when compared with this channel assignment technique, and the results of our mathematical analysis support these conclusions as well. The results also show that more globally optimizing techniques, which consider a larger number of messages during sequencing and assignment decisions, perform consistently better than others which do not take an aggregate view of the scheduling problem. As a general trend, we observed that the SJF priority scheme performs better than LJF in reducing average delays. In terms of throughput, however, SJF and LJF perform similarly. As part of our future work, we plan to study in detail the effect of introducing a delay to allow message queues to fill before the scheduling algorithms are invoked. Furthermore, we plan to extend our model and our techniques to account for receiver/transmitter tuning times.

References

[1] B. Mukherjee, "WDM-based local lightwave networks—Part I: Single-hop systems," *IEEE Network*, pp. 12–27, May 1992.
[2] ——, "WDM-based local lightwave networks—Part II: Multi-hop systems," *IEEE Network*, July 1992, pp. 20–32.
[3] 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. 590–603, May 1993.
[4] A. Ganz and Y. Gao, "Time-wavelength assignment algorithms for high performance WDM star based systems," *IEEE Trans. Commun.*, vol. 42, pp. 1827–1836, Feb./Mar./Apr. 1994.
[5] G. N. Rouskas and M. H. Ammar, "Analysis and optimization of transmission schedules for single-hop WDM networks," *IEEE/ACM Trans. Networking*, pp. 211–221, Apr. 1995.
[6] M. S. Borella and B. Mukherjee, "Efficient scheduling of nonuniform packet traffic in a WDM/TDM local lightwave network with arbitrary transceiver tuning latencies," *IEEE J. Select. Areas Commun.*, pp. 923–934, June 1996.
[7] K. Bogineni and P. W. Dowd, "A collisionless multiple access protocol for a wavelength division multiplexed star-coupled configuration: Architecture and performance analysis," *J. Lightwave Technol.*, vol. 10, pp. 1688–1699, Nov. 1992.
[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] F. Jia, B. Mukherjee, and J. Iness, "Scheduling variable-length messages in a single-hop multichannel local lightwave network," *IEEE/ACM Trans. Networking*, vol. 3, pp. 477–487, Aug. 1995.
[10] C. S. Li, M. S. Chen, and F. F. K. Tong, "POSMAC: A medium access protocol for packet-switched passive optical networks using WDMA," *J. Lightwave Technol.*, vol. 11, pp. 1066–1077, May/June 1993.
[11] N. Mehravari, "Performance and protocol improvements for very high-speed optical fiber local area networks using a passive star topology," *J. Lightwave Technol.*, vol. 8, pp. 520–530, Apr. 1990.
[12] J. H. Lee and C. K. Un, "Dynamic scheduling protocol for variable-sized messages in a WDM-based local network," *J. Lightwave Technol.*, pp. 1595–1600, July 1996.
[13] H. Jeon and C. Un, "Contention-based reservation protocols in multiwavelength optical networks with a passive star topology," in *Proc. IEEE ICC*, June 1992, pp. 1473–1477.
[14] A. Muir and J. J. Garcia-Luna-Aceves, "Distributed queue packet scheduling algorithms for WDM-based networks," in *Proc. IEEE INFOCOM*, 1996, pp. 938–945.
[15] C. D. Polychronopoulos and D. J. Kuck, "Guided self-scheduling: A practical scheduling scheme for parallel supercomputers," *IEEE Trans. Comput.*, vol. C-36, Dec. 1987.
[16] S. F. Hummel, E. Schonberg, and L. E. Flynn, "Factoring: A practical and robust method for scheduling parallel loops," in *Proc. Supercomputing Conf.*, 1991, pp. 610–619.
[17] L. Kleinrock, *Queueing Systems.* New York: Wiley, pp. 1975–1976.
[18] I. Mitrani, *Modeling of Computer And Communication Systems.* New York: Cambridge University Press, 1987.
[19] M. K. Molloy, *Fundamentals of Performance Modeling.* New York: Macmillan, 1989.

Babak Hamidzadeh received the M.S. and Ph.D. degrees in computer science and engineering from The University of Minnesota, Minneapolis, in 1989 and 1993, respectively.

During that period, he also worked as a Research Associate at The Systems and Research Center of Honeywell Inc., and as a Research Scientist at The Research and Technology Center of Alliant Techsystems, Inc., for over three years. From 1993 to 1996, he was an Assistant Professor of Computer Science and Computer Engineering at The Hong Kong University of Science and Technology. Currently, he is an Assistant Professor of Electrical and Computer Engineering at The University of British Columbia, Vancouver, Canada. His areas of research include real-time computing, parallel and distributed processing, multimedia, and communication networks.

Dr. Hamidzadeh is a member of IEEE Computer Society.

Ma Maode received the B.E degree in computer engineering from Tsing Hua University, Beijing, China, in 1982, and the M.E. degree in computer engineering from Tianjin University, Tianjin, China, in 1991. He is currently working towards the Ph.D. degree from The Department of Computer Science of The Hong Kong University of Science and Technology, Kowloon, Hong Kong.

From 1982 to 1985, he was a Project Engineer in the computer industry in China. From 1986 to 1991, he was a system engineer for The Department of Computer Engineering of Tianjin University. From 1991 to 1994, he was a Faculty Member with The Department of Computer Engineering of Tianjin University. His current research interests include WDM optical computer networks, QOS for computer communications, real-time system scheduling, system modeling and simulation, applications of queuing theory, Markov decision processes, and applications of discrete event system algebra.

Mounir Hamdi (M'91) received the B.Sc. degree with distinction in electrical engineering and computer engineering from the University of Southwestern Louisiana, Lafayette, in 1985 and the M.Sc. and Ph.D. degrees in electrical engineering from the University of Pittsburgh, Pittsburgh, PA, in 1987 and 1991, respectively.

While at the University of Pittsburgh, he was a Research Fellow involved with various research projects on interconnection networks, high-speed communication, parallel algorithms, switching theory, and computer vision. In 1991, he joined the Computer Science Department at Hong Kong University of Science and Technology, as an Assistant Professor. He is currently an Associate Professor of Computer Science and the Director of the Computer Engineering Programme in that university. His main areas of research are ATM packet switching architectures, high-speed networks, and wireless networking, and parallel computing. He has published over 90 papers on these areas in various journals, conference proceedings, and books' chapters. He was Guest Editor of a special issue of *Informatica* on Optical Parallel Computing.

Dr. Hamdi received the best paper award at the 12th International Conference on Information Networking. He recently received the "Best Ten Lecturers Election Award" from the Hong Kong University of Science and Technology. He co-Founded and co-Chaired the International Workshop on High-Speed Network Computing, is on the editorial board of the IEEE COMMUNICATIONS MAGAZINE and *Parallel Computing*, and has been on the Program Committee of various International Conferences. He is a member of the ACM.