Segmentation-Based Nonpreemptive Channel Scheduling Algorithms for Optical Burst-Switched Networks

Vinod M. Vokkarane, *Member, IEEE*, and Jason P. Jue, *Senior Member, IEEE*

***Abstract*—One of the key components in the design of optical burst-switched nodes is the development of channel scheduling algorithms that can efficiently handle data burst contentions. Traditional scheduling techniques use approaches such as wavelength conversion and buffering to resolve burst contention. In this paper, we propose nonpreemptive scheduling algorithms that use burst segmentation to resolve burst contentions. We propose two segmentation-based scheduling algorithms, namely, nonpreemptive minimum overlapping channel (NP-MOC) and NP-MOC with void filling (NP-MOC-VF), which can significantly reduce the loss experienced in an optical burst-switched network. We further reduce packet loss by combining burst segmentation and fiber delay lines (FDLs) to resolve contentions during channel scheduling. We propose two types of scheduling algorithms that are classified based on the placement of the FDL buffers in the optical burst-switched node. These algorithms are referred to as delay-first or segment-first algorithms. The scheduling algorithms with burst segmentation and FDLs are investigated through extensive simulations. The simulation results show that the proposed algorithms can effectively reduce the packet-loss probability compared to existing scheduling techniques. The delay-first algorithms are suitable for applications that have higher delay tolerance and strict loss constraints, while the segment-first algorithms are suitable for applications with higher loss tolerance and strict delay constraints.**

***Index Terms*—Burst segmentation, burst switching, channel scheduling, fiber delay lines (FDLs), IP/wavelength division multiplexing (WDM), optical networks, wavelength conversion.**

I. Introduction

THE RAPID growth of the Internet will result in an increased demand for higher transmission rates and faster switching technologies. In order to efficiently utilize the amount of raw bandwidth in wavelength division multiplexing (WDM) networks, an all-optical transport method, which avoids electronic buffering while handling bursty traffic, must be developed. Optical burst switching (OBS) is one such method for transporting traffic directly over a bufferless WDM network [1].

Manuscript received December 14, 2004. This work was supported in part by NSF Grant No. ANI-01-33899. Portions of this paper have appeared in IEEE ICC '03, Anchorage, AL, May 2003, and in the First International Workshop of Optical Burst Switching (WOBS), Dallas, TX, Oct. 2003.

V. M. Vokkarane is with the Department of Computer and Information Science, University of Massachusetts Dartmouth, North Dartmouth, MA 02747 USA (e-mail: vvokkarane@umassd.edu).

J. P. Jue is with the Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083 USA (e-mail: jjue@utdallas.edu).

Digital Object Identifier 10.1109/JLT.2005.856265

In OBS networks, bursts of data consisting of multiple packets are switched through the network all optically. A burst header packet (BHP) is transmitted ahead of the burst in order to configure the switches along the burst's route. The BHP and the data burst are separated at the source, as well as subsequent intermediate nodes, by an offset time, as shown in Fig. 1. The offset time allows for the BHP to be processed at each node before the data burst arrives at the intermediate node; thus, no fiber delay lines (FDLs) are necessary at the intermediate nodes to delay the burst while the BHP is being processed. The BHP may also specify the duration of the burst in order to let a node know when it may reconfigure its switch for the next burst. This signaling technique is known as just enough time (JET) [1]. In this paper, we will consider an OBS network that uses the JET technique. Each WDM link consists of control channels used to transmit BHPs, and data channels used to transmit data bursts. The channel scheduling algorithms considered can also be easily modified to work with other commonly used signaling techniques such as just in time (JIT) [2]–[4]. In this paper, we assume that every channel consists of a wavelength and that each OBS core router has wavelength-conversion capability.

One of the primary OBS core network issues is contention resolution. When two or more bursts are destined for the same output port at the same time, contention occurs. There are many contention-resolution schemes [5] that may be used to resolve the contention. The primary contention-resolution schemes are optical buffering, wavelength conversion, deflection routing, and burst segmentation. In optical buffering, FDLs are used to delay the burst for a specified amount of time, proportional to the length of the delay line, in order to avoid the contention [6]. In wavelength conversion, if two bursts on the same wavelength are destined to go out of the same port at the same time, then one burst can be shifted to a different wavelength [7]. In deflection routing, one of the two bursts will be routed to the correct output port (primary) and the other to any available alternate output port (secondary). The deflected packets may end up following a longer path to the destination, leading to higher end-to-end delay, and packets may also arrive at the destination out of order [8]–[10]. A combination of contention-resolution techniques may be used to provide high throughput, low delay, and low packet-loss probability. In burst segmentation [11], the burst is divided into basic transport units called segments. Each of these segments may consist of a single IP packet or multiple IP packets, with each segment defining the possible partitioning points of a burst when the burst experiences contention in the optical

0733-8724/$20.00 © 2005 IEEE

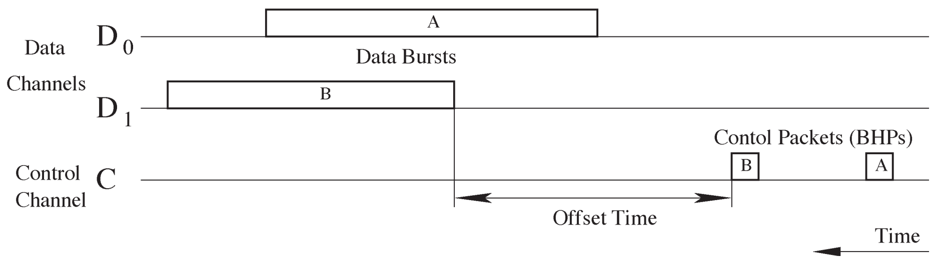


Fig. 1. Data and control channels in a WDM link.

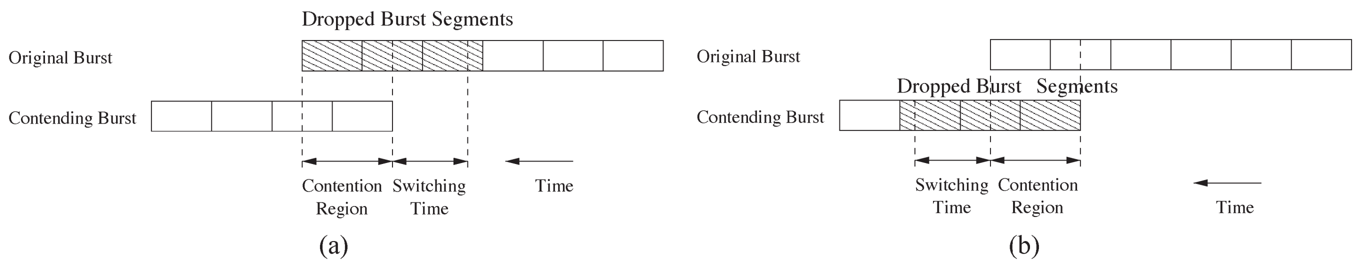


Fig. 2. Selective segment dropping for two contending bursts: (a) tail-dropping policy and (b) head-dropping policy.

network. All segments in a burst are initially transmitted as a single burst unit. However, when contention occurs, only those segments of a given burst that overlap with segments of another burst will be dropped, as shown in Fig. 2. If switching time is not negligible, then additional segments may be lost when the output port is switched from one burst to another. There are two approaches for dropping burst segments when contention occurs between bursts. The first approach, tail dropping, is to drop the tail of the original burst [Fig. 2(a)], and the second approach, head dropping, is to drop the head of the contending burst [Fig. 2(b)] [11].

Another important issue at every OBS core router is the scheduling of data bursts onto outgoing data channels. The scheduling algorithm must find an available data channel for each incoming burst in a manner that is quick and efficient, and that minimizes data loss. In order to minimize data loss, the scheduling algorithm may use one or more contention-resolution techniques. Data-channel scheduling algorithms that use wavelength conversion and FDLs include Horizon, latest available unscheduled channel (LAUC), and LAUC with void filling (LAUC-VF) [12], [13]. However, these techniques drop the burst completely if all of the data channels are occupied at the arrival time of the burst. Instead of dropping the burst in its entirety, it is possible to drop only the overlapping parts of a burst using the burst-segmentation technique.

Due to the inherent property of segmentation, the segmentation-based channel scheduling algorithms can be either nonpreemptive or preemptive. In the nonpreemptive approach, existing channel assignments are not altered, while in preemptive scheduling algorithms, an arriving unscheduled burst[1] may preempt existing data-channel assignments, and the preempted bursts (or burst segments) may be rescheduled or dropped.

[1]Bursts that have been assigned a data channel are referred as the scheduled bursts, and the burst that arrives to the node waiting to be scheduled, as the unscheduled burst.

The advantage of a nonpreemptive approach is that the BHP of the segmented unscheduled burst can be immediately updated with the corresponding change in the burst length and arrival time (offset time). Also, in nonpreemptive channel scheduling algorithms, once a burst is scheduled on the output port, it is guaranteed to be transmitted without being further segmented. The advantage of the preemptive approach can be observed while incorporating QoS into channel scheduling. In this case, a higher priority unscheduled burst can preempt an already scheduled lower priority data burst [14]. In this paper, we consider nonpreemptive segmentation and optical buffering techniques for resolving contentions during channel scheduling.

In order to implement nonpreemptive schemes with non-void-filling-based scheduling algorithms, head dropping must be applied to unscheduled bursts. To implement nonpreemptive schemes with void-filling-based scheduling algorithms, both the head and tail of the unscheduled burst may need to be dropped. In order to implement preemptive schemes with non-void-filling-based scheduling algorithms, we need to use tail dropping on the scheduled burst. At the same time, we may have to drop both the head and tail of the overlapping scheduled burst for void-filling-based scheduling algorithms. In the void-filling case, if the unscheduled burst overlaps more than two bursts, then we have to execute the above procedure on a per-burst basis.

In this paper, we propose new segmentation-based nonpreemptive scheduling algorithms. These nonpreemptive scheduling algorithms perform significantly better in terms of packet-loss probability compared to existing scheduling algorithms. The paper is organized as follows. In Section II, we discuss the OBS network architecture and describe two core-node architectures with FDLs. Section III gives an overview of the channel scheduling algorithms in the OBS literature. Section IV describes the proposed nonpreemptive segmentation-based channel scheduling algorithms with and without void filling. Section V discusses the two new families

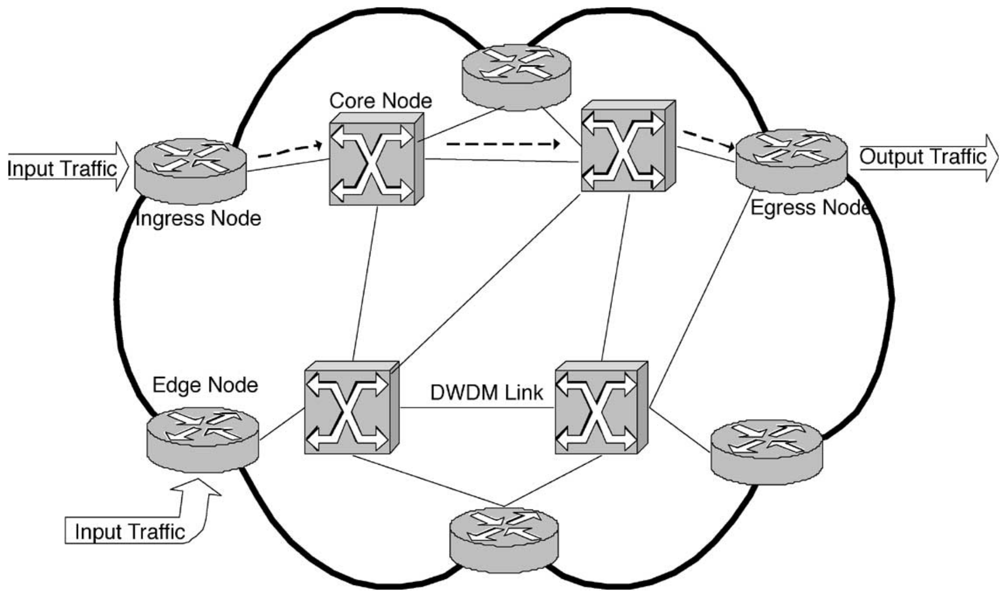


Fig. 3. OBS transport network architecture.

of segmentation-based scheduling algorithms with FDLs (delay first and segment first). Section VI provides numerical results for the different scheduling algorithms. Section VII concludes the paper and proposes directions for future research.

II. OBS Network Architecture

An OBS network consists of a collection of edge and core routers (Fig. 3). The edge routers assemble the electronic input packets into an optical burst, which is sent over the OBS core. The ingress edge node assembles incoming packets from the client terminals into bursts. The bursts are transmitted all optically over OBS core routers without any storage at intermediate nodes within the core. The egress edge node, upon receiving the burst, disassembles the bursts into packets and provides the packets to the destination client terminals. Basic architectures for core and edge routers in an OBS network have been studied in [13].

In the network architecture, we assume that each node can support both new input traffic as well as all-optical transit traffic. Hence, each node consists of both a core router and an edge router, as shown in Fig. 4(a) and (b).

Fig. 4(a) shows a typical architecture of an optical burst-switched node, where optical data bursts are received and sent to the neighboring nodes through physical fiber links. The architecture consists primarily of wavelength converters, variable FDLs, an optical space switch, and a switch control module. We assume that all the header packets incur a fixed processing time at every intermediate node. The switch control module processes the BHPs and sends the control information to the switching fabric to configure the wavelength converters, space switch, and broadcast-and-select switch for the associated data burst. In the case of a data burst entering the optical crossconnect (OXC) before its control packet, the burst is simply dropped (referred to as early burst arrival problem). It is important to note that the arrangement of the key components depends on the architecture of the OBS node considered. A number of different OBS-node architectures are possible using FDLs as optical buffers.

The edge router [Fig. 4(b)] performs the functions of presorting packets, buffering packets, aggregating packets into burst, and deaggregating bursts into its constituent packets. Different burst-assembly policies, such as a threshold policy or a timer mechanism, can be used to aggregate bursty data packets into optical bursts and to send the bursts into the network. The architecture of the edge router consists of a routing module (RM), a burst assembler (BA), and a scheduler. The RM selects the appropriate output port for each packet and sends each packet to the corresponding BA module. Each BA module assembles bursts consisting of packets that are headed for a specific egress router. In the BA module, there is a separate packet queue for each class of traffic. The scheduler creates a burst based on the burst-assembly technique and transmits the burst through the intended output port. At the egress router, a burst-disassembly module disassembles the bursts into packets and send the packets to the upper network layers.

From Fig. 4, we can observe that channel scheduling is a critical function of both the edge node as well as the core node. The edge node schedules freshly assembled data bursts from the electronic input buffer onto outgoing WDM data channels. This process is similar to traditional IP-based scheduling and those well-known techniques such as first come first served and round robin [15] can be used. In this paper, we consider the challenging problem of core-node scheduling. Core scheduling is challenging since the core nodes do not have the luxury of storing the incoming bursts in electronic buffers (RAMs).

We study core scheduling wherein the core nodes are equipped with all-optical full wavelength converters and limited FDLs. We consider two OBS-node architectures with FDLs for realizing the proposed scheduling algorithms. The architecture in Fig. 5(a) shows an input-buffered FDL OBS node with FDLs dedicated to each input port, while Fig. 5(b) shows an output-buffered FDL OBS node with FDLs dedicated to each output port.

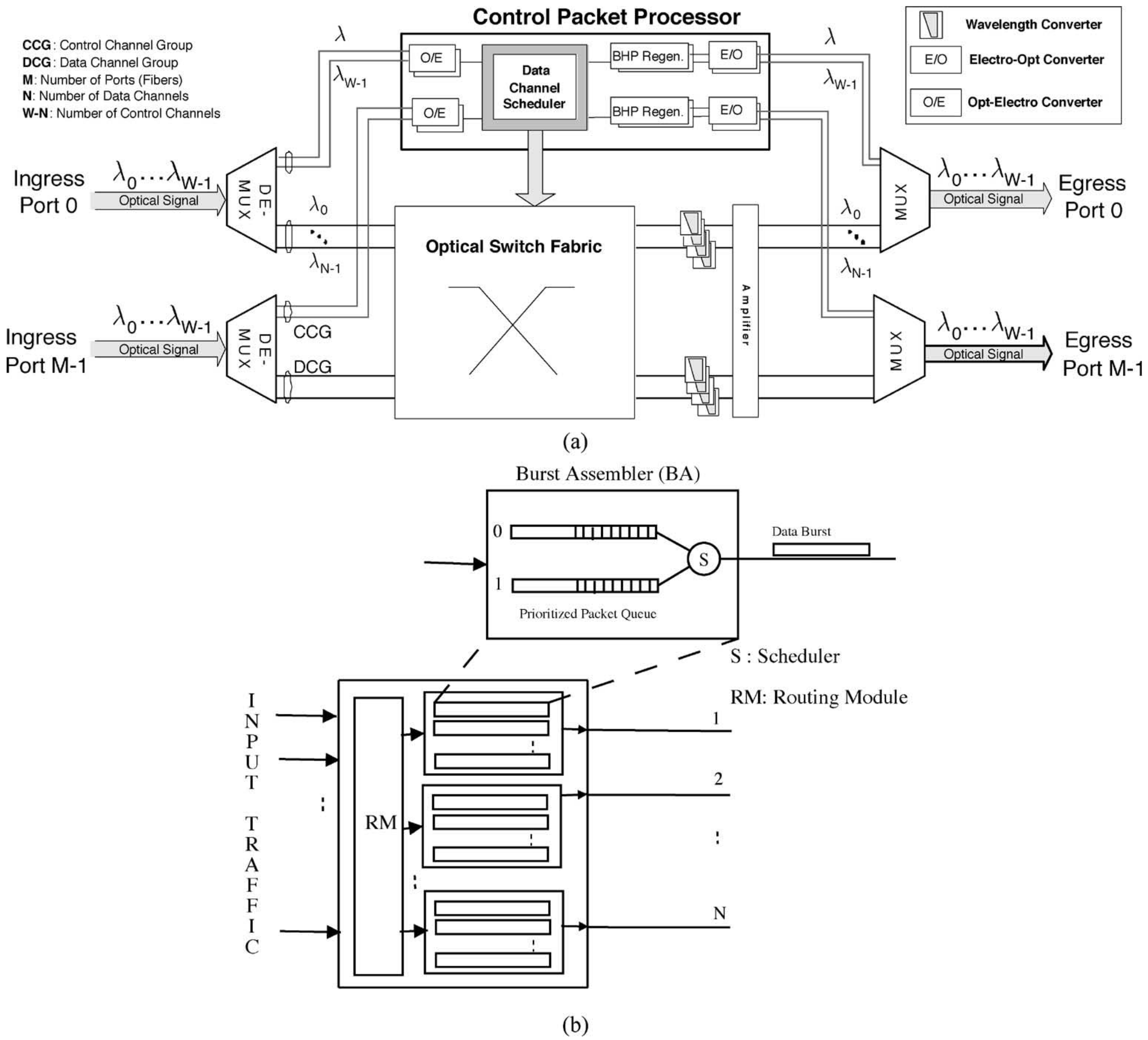


Fig. 4. (a) Architecture of core router and (b) architecture of edge router.

In the input-buffered OBS-node architecture shown in Fig. 5(a), each input port is equipped with an FDL buffer containing N delay lines. The input-buffered architecture supports the delay-first scheduling algorithms. The n data channels are demultiplexed from each input fiber link and are passed through wavelength converters whose function is to convert the input wavelengths to wavelengths that are used within the FDL buffers. The use of different wavelengths in the FDL buffers and on the output links helps to resolve contentions among multiple incoming data bursts competing for the same FDL and the same output link. In the design of FDL buffers, we can have fixed-delay FDL buffers, variable-delay FDL buffers, or a mixture of both. In this paper, we follow the architecture of variable-delay FDL buffers.

In the output-buffered OBS-node architecture, shown in Fig. 5(b), the FDL buffers are placed after the switch fabric. The output-buffered architecture supports the segment-first scheduling algorithms. The input wavelength converters are used to convert the input wavelengths to the wavelengths that are used within the switching fabric. The functions of the output wavelength converters are the same as described in the input-buffered FDL architecture.

In this paper, we only considered the above-described per-port FDL architectures. In order to minimize switch cost, a per-node FDL architecture can be adopted, in which a single set of FDLs can be used for all the ports in a node. This lowering of switch cost results in lower performance with respect to packet loss.

III. Channel Scheduling—Background

Another type of scheduling in optical burst-switched networks is channel scheduling. In channel scheduling, multiple wavelengths are available on each link, and the problem is to

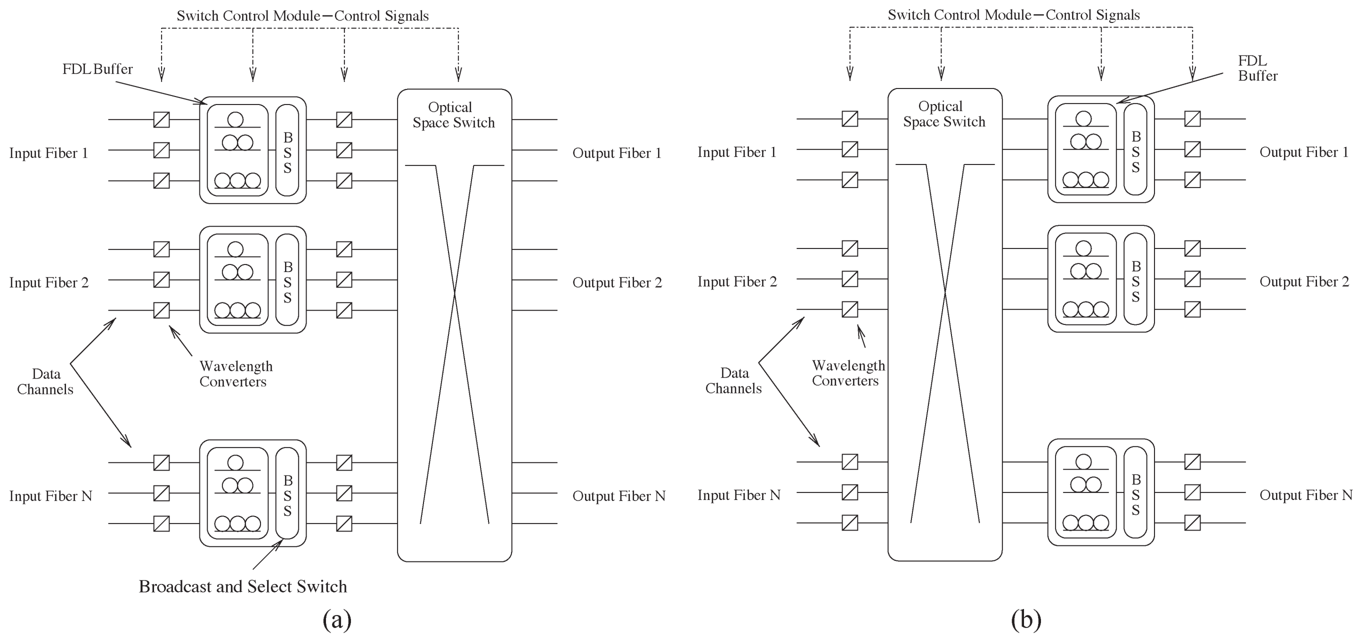


Fig. 5. (a) Input-buffered and (b) output-buffered FDL architecture.

assign an incoming burst to an appropriate channel or wavelength on the outgoing link. In this problem, all-optical wavelength conversion is assumed to be available at each node, and the scheduling occurs at intermediate core nodes as well as ingress nodes. The primary objective in this type of scheduling is to minimize the "gaps" in each channel's schedule, where a gap is the idle space between two bursts, which are transmitted over the same output wavelength. Channel scheduling in OBS networks is different from traditional IP scheduling. In IP, each core node stores the packets in electronic buffers and schedules them on the desired output port. In OBS, once a burst arrives at a core node, it must be sent to the next node without storing the burst in electronic buffers. We assume that each OBS core node supports full optical wavelength conversion.

When a BHP arrives at a core node, a channel scheduling algorithm is invoked to assign the unscheduled burst to a data channel on the outgoing link. The channel scheduler obtains the burst arrival time and duration of the unscheduled burst from the BHP. The algorithm may need to maintain the latest available unscheduled time (LAUT) or the horizon, gaps, and voids on every outgoing data channel. Traditionally, the LAUT of a data channel is the earliest time at which the data channel is available for an unscheduled data burst to be scheduled. A gap is the time difference between the arrival of the unscheduled burst and ending time of the previously scheduled burst. A void is the unscheduled duration (idle period) between two scheduled bursts on a data channel. For void-filling algorithms, the starting and the ending time for each burst on every data channel must also be maintained.

The following information is used by the scheduler for most of the scheduling algorithms.

L_{b}	Unscheduled burst length duration.
t_{ub}	Unscheduled burst arrival time.
W	Maximum number of outgoing data channels.
N_{b}	Maximum number of data bursts scheduled on a data channel.
D_i	ith outgoing data channel.
LAUT_i	LAUT of the ith data channel, $i = 1, 2, \ldots, W$, for non-void-filling scheduling algorithms.
$S_{(i,j)}$ and $E_{(i,j)}$	Starting and ending times of each scheduled burst j on every data channel i for void-filling scheduling algorithms.
Gap_i	If the channel is available, gap is the difference between t_{ub} and LAUT_i for scheduling algorithms without void filling, and is the difference between t_{ub} and $E_{(i,j)}$ of previous scheduled burst j for scheduling algorithms with void filling. If the channel is busy, Gap_i is set to 0. Gap information is useful to select a channel for the case in which more than one channel is free.
$\mathrm{Void}_{(i,k)}$	Duration of the kth void on the ith data channel. This information is relevant to void-filling algorithms. A void is the duration between the $S_{(i,j+1)}$ and $E_{(i,j)}$ on a data channel. Void information is useful in selecting a data channel in case more than one channel is free.

Data-channel scheduling algorithms can be broadly classified into two categories: with and without void filling. The algorithms primarily differ based on the type and amount of state information that is maintained at a node about every channel. In data-channel scheduling algorithms without void filling, the LAUT_i on every data channel D_i, $i = 0, 1, \ldots, W$, is maintained by the channel scheduler. In void-filling algorithms, the starting time $S_{(i,j)}$ and ending time $E_{(i,j)}$ are maintained for each burst on every data channel, where $i = 0, 1, \ldots, W$ is

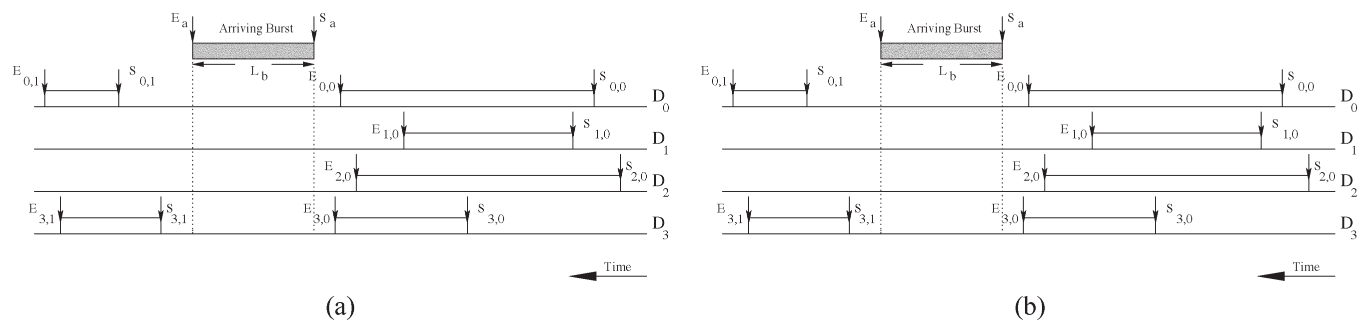


Fig. 6. Initial data-channel status (a) without void filling, (b) with void filling.

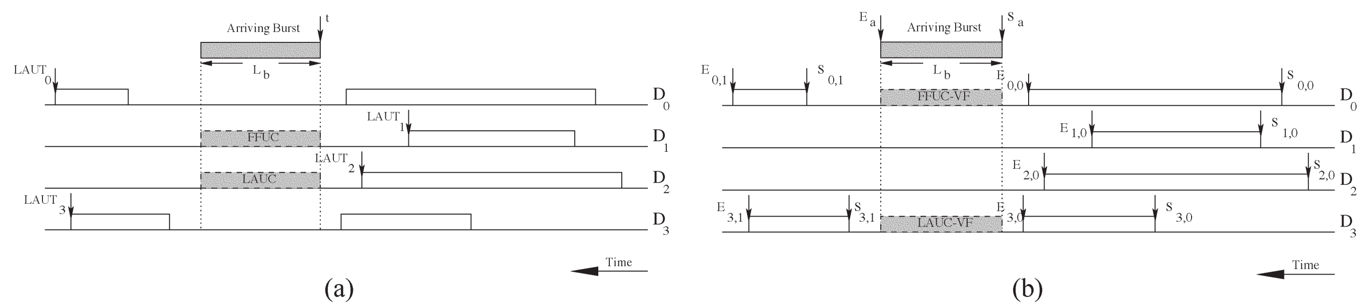


Fig. 7. Channel assignment after using (a) non-void-filling algorithms (FFUC and LAUC), and (b) void-filling algorithms (FFUC-VF and LAUC-VF).

the ith data channel and $j = 0, 1, \ldots, N_{\mathrm{b}}$ is the jth burst on channel i.

Let the initial data-channel assignment for the channel scheduling algorithms without void filling and with void filling be as shown in Fig. 6(a) and (b), respectively. In Fig. 6(a), the LAUT_i on every data channel D_i, $i = 0, 1, \ldots, W$, is maintained by the scheduler. In Fig. 6(b), the starting time $S_{(i,j)}$ and the ending time $E_{(i,j)}$, where i refers to the ith data channel and j is the jth burst on channel i, are maintained for each burst on every output data channel. In the following sections, we will describe traditional non-void-filling scheduling algorithms, such as first-fit unscheduled channel (FFUC) and LAUC, and traditional void-filling scheduling algorithms, such as FFUC with Void Filling (FFUC-VF) and LAUC-VF.

A. First Fit Unscheduled Channel (FFUC)

The FFUC scheduling algorithm keeps track of the LAUT (or horizon) on every data channel. A wavelength is considered for each arriving burst when the unscheduled time (LAUT) of the data channel is less then the burst arrival time. The FFUC algorithm searches all the channels in a fixed order and assigns the first available channel for the new arriving burst. The primary advantage of FFUC is the simplicity of the algorithm and that the algorithm needs to maintain only one value (LAUT_i) for each channel. The FFUC algorithm can be illustrated in Fig. 7(a). Based on the LAUT_i, data channels D_1 and D_2 are available for the duration of the unscheduled burst. If the channels are ordered based on the index of the wavelengths $(D_0, D_1, \ldots, D_W)$, the arriving burst is scheduled on outgoing data channel D_1. The time complexity of the FFUC algorithm is $O(\log W)$. The primary drawback of FFUC is the high burst-dropping probability as a tradeoff for simplicity in scheduling. The following algorithms aim at reducing the burst-dropping probability at the expense of increased algorithm complexity.

B. Horizon or LAUC

The LAUC or Horizon [12] scheduling algorithm keeps track of the LAUT (or horizon) on every data channel and assigns the data burst to the latest available unscheduled data channel. The LAUC algorithm can be illustrated in Fig. 6(a). Based on the LAUT_i, data channels D_1 and D_2 are available for the duration of the unscheduled burst. Also, we observe that $\mathrm{Gap}_1 > \mathrm{Gap}_2$; thus, the arriving burst is scheduled on the outgoing data channel with the minimum gap, i.e., D_2. The time complexity of the LAUC algorithm is $O(W)$.

C. FFUC With Void Filling (FFUC-VF)

The FFUC-VF scheduling algorithm maintains the starting and ending times for each scheduled data burst on every data channel. The goal of this algorithm is to utilize voids between two data-burst assignments. The first channel with a suitable void is chosen. The FFUC-VF algorithm is illustrated in Fig. 6(b). Based on $S_{i,j}$ and $E_{i,j}$, all the data channels D_0, D_1, D_2, and D_3 are available for the duration of the unscheduled burst. If the channels are ordered based on the index of the wavelengths $(D_0, D_1, \ldots, D_W)$, the arriving burst is scheduled on outgoing data channel D_0. If N_{b} is the number of bursts currently scheduled on every data channel, then a binary search algorithm can be used to check if a data channel is eligible. Thus, the time complexity of the FFUC-VF algorithm is $O(W \log N_{\mathrm{b}})$.

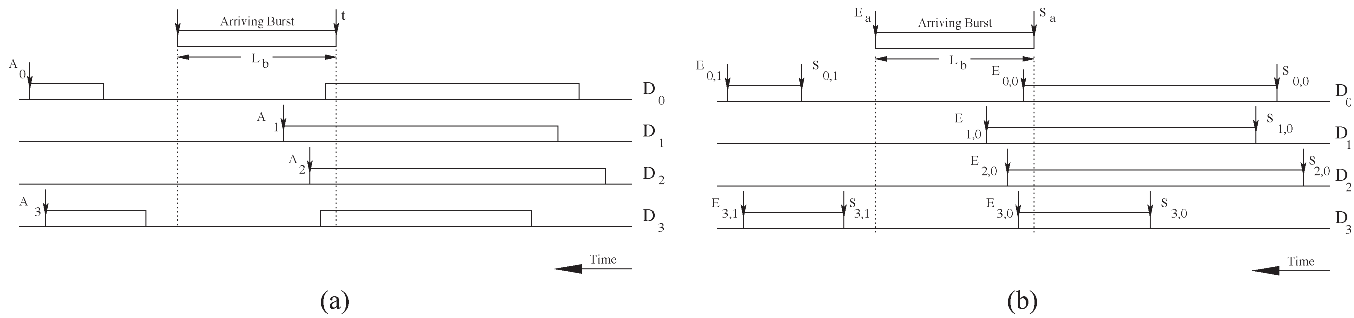


Fig. 8. Initial data-channel status (a) without void filling and (b) with void filling.

D. LAUC With Void Filling (LAUC-VF)

The LAUC-VF [16] scheduling algorithm maintains the starting and ending times for each scheduled data burst on every data channel. The goal of this algorithm is to utilize voids between two data-burst assignments. The channel with a void that minimizes the gap is chosen. The LAUC-VF algorithm is illustrated in Fig. 6(b). Based on $S_{i,j}$ and $E_{i,j}$, all the data channels D_0, D_1, D_2, and D_3 are available for the duration of the unscheduled burst. Also, we observe that D_3 had the least gap Gap_3; thus, the arriving burst is scheduled on D_3. If N_{b} is the number of bursts currently scheduled on every data channel, then a binary search algorithm can be used to check if a data channel is eligible. Thus, the time complexity of the LAUC-VF algorithm is $O(W \log N_{\mathrm{b}})$.

Recently, researchers have proposed several optimizations for the above-described scheduling algorithms. In [17], a minimizing voids unscheduled channel (MVUC) algorithm is proposed, with the objective of minimizing voids generated by arriving bursts at each core node. In the MVUC algorithm, when the burst that has arrived at the optical core router at a certain time can be transmitted in some data channels by using the unused data-channel capacity, the MVUC algorithm selects the data channel in which the newly generated void after scheduling the arriving burst becomes minimum. The authors claim through computer simulations that the MVUC performs better than LAUC-VF in terms of data loss.

The authors of [18] propose the minimum starting void (Min-SV) algorithm for selecting channels for incoming data bursts. The advantage of Min-SV is that it has the same scheduling criteria as LAUC-VF. However, the data structure of Min-SV is constructed by augmenting a balanced binary search tree. By constructing this tree, Min-SV achieves a loss rate as low as LAUC-VF and processing time as low as Horizon (LAUC).

The Look-ahead Window (LAW) [19] or a Group-based Scheduling algorithm [20], takes advantage of the separation between the data bursts and the BHPs (offset time). By receiving BHPs one offset time prior to their corresponding data bursts, it is possible to construct a look-ahead window. The authors believe that such a collective view of multiple BHPs results in more efficient decisions with regard to which incoming bursts should be discarded or reserved. Also, the use of FDLs for any lost time in the offset, due to the creating of a window, is suggested.

There has also been substantial work on scheduling using FDLs in OBS [12], [13], [21]. In the next section, we propose several new scheduling algorithms that are based on burst segmentation [14], with and without FDLs. We show that our proposed algorithms can achieve significantly lower loss than all the above scheduling algorithms [22]. We describe our new scheduling algorithms using LAUC and LAUC-VF as the baseline algorithms. The proposed modifications can be applied to any of the above channel scheduling algorithms so as to improve loss performance.

IV. Nonpreemptive Channel Scheduling Algorithms

In this section, we will explain the details of the implementation of LAUC-based algorithms and the proposed scheduling algorithms. The following channel information has to be maintained at the scheduler in addition to the ones listed in the previous section for all segmentation-based scheduling algorithms:

$\mathrm{Overlap}_i$ Duration of overlap between the unscheduled burst and scheduled burst(s). Overlap is used in non-void-filling channel scheduling algorithms. The overlap is 0 if the channel is available, otherwise, the overlap is the difference between LAUT_i and t_{ub}.

Loss_i Number of packets dropped due to the assignment of the unscheduled burst on the ith data channel. The primary goal of all scheduling algorithms is to minimize loss; hence, loss is the primary factor for choosing a data channel. In case the loss on more than one channel is the same, then other channel parameters are used to reach a decision on the selection of data channel.

Let the initial data-channel assignment for the channel scheduling algorithms without void filling and with void filling be as shown in Fig. 8(a) and (b), respectively. In Fig. 8(a), the LAUT_i on every data channel $D_i = 0, 1, \ldots, W$ is maintained by the scheduler. In Fig. 8(b), the starting time $S_{(i,j)}$ and ending time $E_{(i,j)}$, where i refer to the ith data channel and j is the jth burst on channel i, are maintained for each burst on every data channel.

Let us now consider a channel scheduling scenario as shown in Fig. 8. The channel assignments of both LAUC and LAUC-VF algorithms are illustrated in Fig. 8(a) and (b), respectively. We see that no channel is available for the duration of the unscheduled burst in both cases. In particular, with LAUC, the entire unscheduled burst is dropped, even though the contention period with bursts on data channel D_2 is minimal.

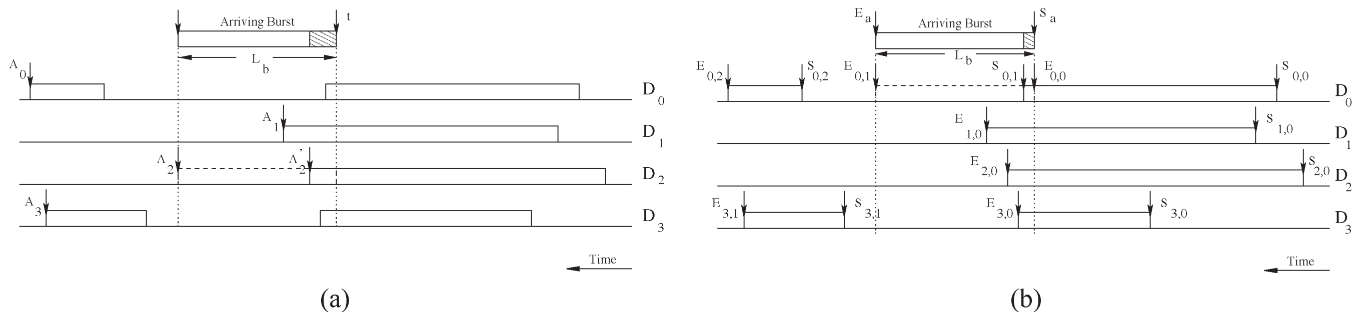


Fig. 9. Illustration of nonpreemptive (a) NP-MOC scheduling algorithm, and (b) NP-MOC-VF scheduling algorithm.

While using LAUC-VF, the entire unscheduled burst is also dropped, even though the contention period with bursts on data channel D_0 is minimal.

We now describe the new segmentation-based scheduling algorithms and also evaluate the relative performance of the algorithms under similar scenarios.

A. Nonpreemptive Minimum Overlapping Channel (NP-MOC)

NP-MOC algorithm is an improvement of the existing LAUC scheduling algorithm. The NP-MOC scheduling algorithm keeps track of the LAUT on every data channel. For a given unscheduled burst, the scheduling algorithm considers all outgoing data channels and calculates the overlap on every channel, and chooses the data channel with minimum overlap.

NP-MOC ALGORITHM (t_{ub})
tempOverlap $\leftarrow$ INFINITY;
tempGap $\leftarrow$ INFINITY;
tempChannel $\leftarrow -1$;
for each $i \in$ Data Channel
{ **if**(Overlap$_i$ is ZERO) and (Gap$_i$ < tempGap)
  { tempGap $\leftarrow$ Gap$_i$;
    tempChannel $\leftarrow i$;

if (tempChannel$\langle\rangle -1$)
  { Schedule the Unscheduled Burst on D_i;
    Stop;
else { **for each** $i \in$ Data Channel
  { **if**(Overlap$_i$ < tempOverlap)
    { tempOverlap $\leftarrow$ Overlap$_i$;
      tempChannel $\leftarrow i$;

if (tempChannel $<> -1$)
  { Resolve Contention using NP-Segmentation
    Schedule the Unscheduled Burst on D_i;
    Stop;
else { Drop Unscheduled Burst;
    Stop;

The details of NP-MOC are given above. For example, applying the NP-MOC algorithm to the example in Fig. 8(a), we see that data channel D_2 has the minimum overlap, and the unscheduled burst is scheduled on D_2 [Fig. 9(a)]. Here, only the overlapping segments of the unscheduled burst are dropped instead of the entire unscheduled burst, as in the case of LAUC. The worst time complexity of the NP-MOC algorithm is $O(W)$.

B. NP-MOC With Void Filling (NP-MOC-VF)

The NP-MOC-VF scheduling algorithm maintains the starting and ending times of each data burst on every data channel. The goal is to utilize the voids between data-burst assignments on every data channel. The data channel with a void that minimizes Gap$_i$ is chosen in case of more than one available channel. If no channel is free, the channel with minimum loss is assigned to the unscheduled burst. The details of NP-MOC-VF are given below. For example, applying the NP-MOC-VF algorithm to the example in Fig. 8(a), we see that data channel D_0 has the minimum overlap, and the unscheduled burst is scheduled on D_0 [Fig. 9(b)]. Here, only the overlapping segments of the unscheduled burst are dropped instead of the entire unscheduled burst, as in the case of LAUC-VF. The worst time complexity of the NP-MOC-VF algorithm is $O(W \log N_{\mathrm{b}})$.

NP-MOC-VF ALGORITHM (t_{ub})
tempLoss $\leftarrow$ INFINITY;
tempGap $\leftarrow$ INFINITY;
tempChannel $\leftarrow -1$;
for each $i \in$ Data Channel
{ **if** (Loss$_i$ is ZERO) and (Gap$_i$ < tempGap)
  { tempGap $\leftarrow$ Gap$_i$;
    tempChannel $\leftarrow i$;

if (tempChannel$\langle\rangle -1$)
  { Schedule the Unscheduled Burst on D_i;
    Stop;
else { **for each** $i \in$ Data Channel
  { **if** (Loss$_i$ < tempLoss)
    { tempLoss $\leftarrow$ Loss$_i$;
      tempChannel $\leftarrow i$;

TABLE I
COMPARISON OF SEGMENTATION-BASED NONPREEMPTIVE SCHEDULING ALGORITHMS

Algorithm	Time Complexity	State Information
LAUC	O(W)	$LAUT_i$, Gap_i
LAUC-VF	O($W \log N_b$)	$S_{(i,j)}$, $E_{(i,j)}$,Gap_i
NP-MOC	O(W)	$LAUT_i$, Gap_i
NP-MOC-VF	O($W \log N_b$)	$S_{(i,j)}$, $E_{(i,j)}$,Gap_i

if (tempChannel$\langle\rangle - 1$)

$$\begin{cases} \text{Resolve Contention using NP-Segmentation} \\ \text{Schedule the Unscheduled Burst on } D_i; \\ \text{Stop;} \end{cases}$$

$$\textbf{else} \begin{cases} \text{Drop Unscheduled Burst;} \\ \text{Stop;} \end{cases}$$

Table I compares all the above channel scheduling algorithms in terms of time complexity and the amount of state information stored. We observe that the time complexity of the non-void-filling algorithms is less than that of the void-filling algorithms. Also, void-filling algorithms, such as LAUC-VF and NP-MOC-VF, store more state information as compared to non-void-filling algorithms, such as LAUC and NP-MOC.

V. SEGMENTATION-BASED NONPREEMPTIVE SCHEDULING ALGORITHMS WITH FDLS

There has been substantial work on scheduling using FDLs in OBS [12], [13], [21]. In this section, we propose several segmentation-based nonpreemptive scheduling algorithms incorporating FDLs. Based on the two FDL architectures presented in Section II, we have two families of scheduling algorithms. Scheduling algorithms based on the input-buffered FDL node architecture are called delay-first scheduling algorithms, while scheduling algorithms based on the output-buffered FDL node architecture are called segment-first scheduling algorithms. In both schemes, we assume that full wavelength conversion, FDLs, and segmentation techniques are used to resolve burst contention for an output data channel. However, the order of applying the above techniques depends on the FDL architecture. In delay-first schemes, we resolve contention by wavelength conversion, FDLs, and segmentation, in that order, while in segment-first schemes, we resolve contention by wavelength conversion, segmentation, and FDLs, in that order. Before going on to the detailed description of the schemes, it is necessary to discuss the motivation for developing two different schemes. In delay-first schemes, FDLs are primarily used to delay the entire burst, while in segment-first schemes, FDLs are primarily used to delay the segmented bursts. Delaying the entire burst and then segmenting the burst keeps the packets in order; however, when delaying segmented bursts, packet order is not always maintained. In general, segment-first schemes will incur lower delays than delay-first schemes. In both the schemes, the scheduler has to additionally know MAX_DELAY, i.e., the maximum delay provided by the FDLs.

We will now describe the segmentation-based nonpreemptive scheduling algorithms that use segmentation, wavelength conversion, and FDLs.

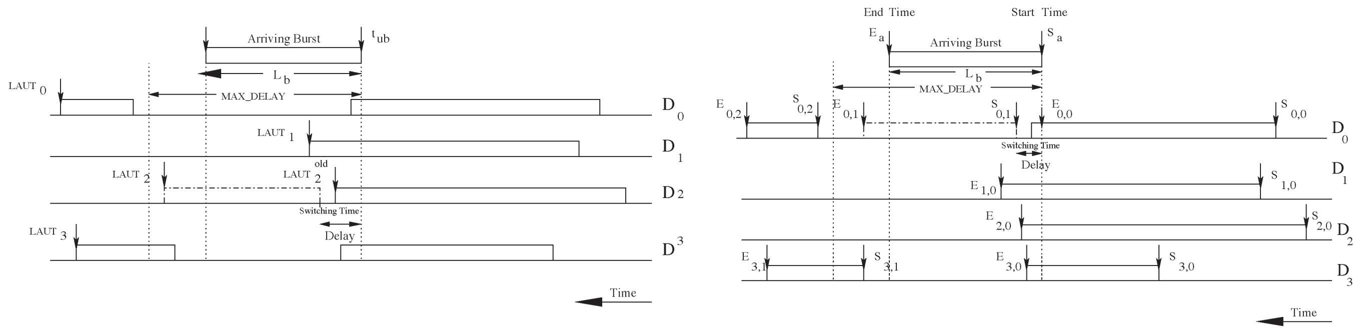
A. Delay-First Scheduling Algorithms

1) Nonpreemptive Delay-First Minimum Overlap Channel (NP-DFMOC): The NP-DFMOC algorithm calculates the overlap on every channel and then selects the channel with minimum overlap. If a channel is available, then the unscheduled burst is scheduled on the free channel with the minimum gap. If all channels are busy and the minimum overlap is greater than or equal to the sum of the unscheduled burst length and MAX_DELAY, then the entire unscheduled burst is dropped. Otherwise, the unscheduled burst is delayed for the duration of the minimum overlap and scheduled on the selected channel. In case the minimum overlap is greater than MAX_DELAY, the unscheduled burst is delayed for MAX_DELAY and the nonoverlapping burst segments of the unscheduled burst is scheduled, while the overlapping burst segments are dropped. For example, in Fig. 10(a), the data channel D_2 has the minimum overlap, thus, the unscheduled burst is scheduled on D_2 after providing a delay using FDLs.

2) Nonpreemptive Delay-First Minimum Overlap Channel With Void Filling (NP-DFMOC-VF): The NP-DFMOC-VF algorithm calculates the delay until the first void on every channel and then selects the channel with minimum delay. If a channel is available, the unscheduled burst is scheduled on the free channel with minimum gap. If all channels are busy and the starting time of the first void is greater than or equal to the sum of the end time E_a of the unscheduled burst and MAX_DELAY, then the entire unscheduled burst is dropped. Otherwise, the unscheduled burst is delayed until the start of the first void on the selected channel, where the nonoverlapping burst segments of the unscheduled burst are scheduled, while the overlapping burst segments are dropped. In case the start of the first void is greater than the sum of the start time S_a of the unscheduled burst and MAX_DELAY, then the unscheduled burst is delayed for MAX_DELAY and the nonoverlapping burst segments of the unscheduled burst are scheduled, while the overlapping burst segments are dropped. For example, consider Fig. 10(b). By applying the NP-DFMOC-VF algorithm, the data channel D_0 has the minimum delay, thus, the unscheduled burst is scheduled on D_0 after delaying the burst using FDLs. In this case, only the overlapping segments of the burst are dropped instead of the entire burst, as in the case of LAUC-VF.

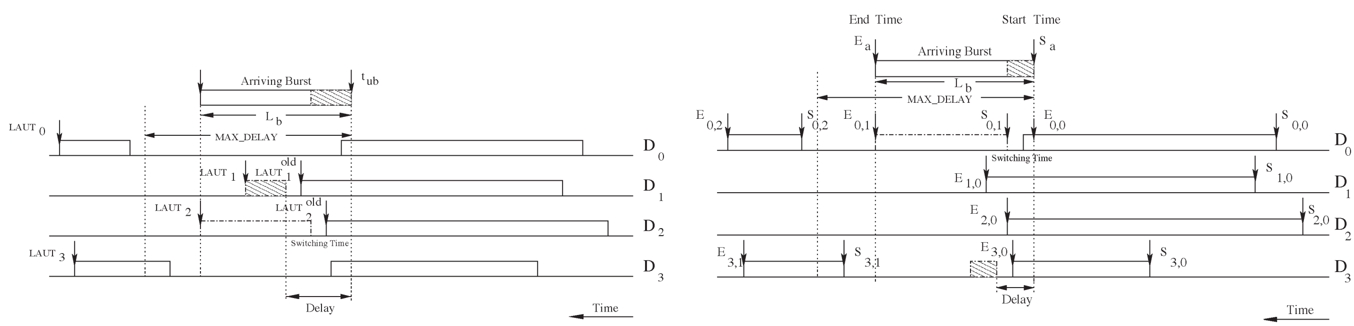
B. Segment-First Scheduling Algorithms

1) Nonpreemptive Segment-First Minimum Overlap Channel (NP-SFMOC): The NP-SFMOC algorithm calculates the overlap on every channel and then selects the data channel with minimum overlap. If a channel is available, the unscheduled burst is scheduled on the free channel with the minimum Gap_i. If all channels are busy and the minimum overlap is greater than or equal to the sum of the unscheduled burst length



(a) (b)

Fig. 10. Illustration of (a) NP-DFMOC algorithm, and (b) NP-DFMOC-VF algorithm.



(a) (b)

Fig. 11. Illustration of (a) NP-SFMOC algorithm, and (b) NP-SFMOC-VF algorithm.

and MAX_DELAY, then the entire unscheduled burst is dropped. Otherwise, the unscheduled burst is segmented (if necessary) and the nonoverlapping burst segments are scheduled on the selected channel, while the overlapping burst segments are rescheduled. Next, the algorithm calculates the overlap on all the channels for the rescheduled burst segments. The rescheduled burst segments are delayed for the duration of the minimum overlap and scheduled on the selected channel. In case the minimum overlap is greater than MAX_DELAY, then the rescheduled burst segments are delayed for MAX_DELAY and the nonoverlapping burst segments of the rescheduled burst segments are scheduled, while the overlapping burst segments are dropped. For example, in Fig. 11(a), we observe that the data channel D_2 has the minimum overlap for the unscheduled burst, thus, the unscheduled burst is scheduled on D_2, and the rescheduled burst segments are scheduled on D_1.

2) Nonpreemptive Segment-First Minimum Overlap Channel With Void Filling (NP-SFMOC-VF): The NP-SFMOC-VF algorithm calculates the loss on every channel and then selects the channel with minimum loss. If a channel is available, the unscheduled burst is scheduled on the free channel with minimum gap. If all channels are busy and the starting time of the first void is greater than or equal to the sum of the end time E_a of the unscheduled burst and MAX_DELAY, then the entire unscheduled burst is dropped. If the starting time of the first void is greater than or equal to the end time E_a of the unscheduled burst, the NP-DFMOC-VF algorithm is employed. Otherwise, the unscheduled burst is segmented (if necessary) and the nonoverlapping burst segments are scheduled on the selected channel, while the overlapping burst segments are rescheduled. For the rescheduled burst segments, the algorithm calculates the delay required until the start of the next void on every channel and selects the channel with minimum delay. The rescheduled burst segments are delayed until the start of the first void on the selected channel. The nonoverlapping burst segments of the rescheduled burst are scheduled, while the overlapping burst segments are dropped. In case the start of the next void is greater than the sum of the start time S_a of the unscheduled burst and MAX_DELAY, the rescheduled burst segments are delayed for MAX_DELAY and the nonoverlapping burst segments of the rescheduled burst are scheduled, while the overlapping burst segments are dropped. For example, in Fig. 11(b), we observe that the data channel D_0 has the minimum loss, thus, the unscheduled burst is scheduled on D_0, and the unscheduled burst segments are scheduled on D_3 (as it incurs the minimum delay) after providing a delay using FDLs.

Table II compares all of the discussed segmentation-based nonpreemptive channel scheduling algorithms with FDLs in terms of time complexity and the amount of state information stored. We can observe that the time complexity of the non-void-filling algorithms is less than the void-filling algorithms. Also, void-filling algorithms, such as LAUC-VF, NP-DFMOC-VF, and NP-SFMOC-VF, store more state information as compared to non-void-filling algorithms, such as LAUC, NP-DFMOC, and NP-SFMOC.

VI. Numerical Results

In order to evaluate the performance of the proposed channel scheduling algorithms, a simulation model is developed. Burst

TABLE II
COMPARISON OF SEGMENTATION-BASED NONPREEMPTIVE SCHEDULING ALGORITHMS WITH FDLS

Algorithm	Time Complexity	State Information
LAUC	$O(W)$	$LAUT_i$, Gap_i
LAUC-VF	$O(W \log N_b)$	$S_{(i,j)}$, $E_{(i,j)}$, Gap_i
NP-DFMOC	$O(W)$	$LAUT_i$, Gap_i
NP-DFMOC-VF	$O(W \log N_b)$	$S_{(i,j)}$, $E_{(i,j)}$, Gap_i
NP-SFMOC	$O(W)$	$LAUT_i$, Gap_i
NP-SFMOC-VF	$O(W \log N_b)$	$S_{(i,j)}$, $E_{(i,j)}$, Gap_i

arrivals to the network are Poisson, and each burst length is an exponentially generated random number rounded to the nearest integer multiple of the fixed-sized packet length of 1250 B. The average burst length is 100 μs. The link transmission rate is 10 Gbit/s. Current switching technologies provide us with a range of switching times from a few milliseconds (microelectromechanical systems (MEMS)] [23] to a few picoseconds [semiconductor optical amplifier (SOA) based] [24]. We assume a conservative switch reconfiguration time of 10 μs. The burst header processing time at each node depends on the architecture of the scheduler and the complexity of the scheduling algorithm. Based on current CPU clock speeds and a conservative estimate of the number of instructions required, we assume burst header processing time to be 2.5 μs. We know that in any optical buffer architecture, the size of the buffers is severely limited, not only by signal quality concerns, but also by physical space limitations. To delay a single burst for 5 μs requires over a kilometer of fiber. Due to this size limitation of optical buffers, we consider a maximum FDL delay of 0.01 ms. Traffic is uniformly distributed over all sender–receiver pairs. Fixed minimum-hop routing is used to find the path between all node pairs. All the simulations are implemented on the standard 14-node National Science Foundation (NSF) network shown in Fig. 12, where link distances are in kilometers.

Fig. 13(a) plots the total packet-loss probability versus load for different channel scheduling algorithms, with eight data channels on each link. We observe that the segmentation-based channel scheduling algorithms perform significantly better than algorithms without segmentation. The proposed segmentation-based scheduling algorithms perform better than the algorithms without segmentation because, when contention occurs, only the overlapping packets from one of the bursts are lost instead of the entire burst. We see that NP-MOC suffers lower loss as compared to LAUC. Also, NP-MOC-VF performs better than LAUC-VF. We can also observe that NP-MOC and NP-MOC-VF are the best algorithms without and with void filling, respectively. Also, the algorithms with void filling perform better than algorithms without void filling, as expected.

Fig. 13(b) plots the average end-to-end delay versus load for different channel scheduling algorithms, with eight data channels on each link. We observe that the segmentation-based channel scheduling algorithms have higher average end-to-end packet delay than existing channel scheduling algorithms without segmentation. The higher delay for scheduling algorithms with segmentation is due to the higher probability of a successful transmission between source–destination pairs that are farther apart, while in traditional scheduling algorithms, the entire burst is dropped in case of a contention; hence, source–destination pairs close to each other have a higher probability of making a successful transmission, which results in lower average end-to-end packet delay. We see that the NP-MOC algorithm has higher delay than the LAUC algorithm. Also, the NP-MOC-VF algorithm has higher delay than the LAUC-VF algorithm. We can also observe that LAUC has the least average end-to-end packet delay among all the algorithms.

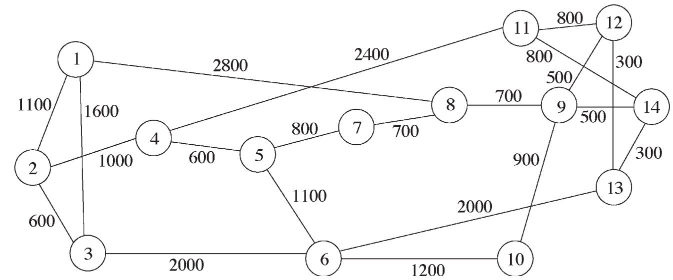


Fig. 12. Fourteen-node NSF Network.

Fig. 14(a) plots the total packet-loss probability versus load for different channel scheduling algorithms. We observe that the channel scheduling algorithms with burst segmentation perform better than algorithms without burst segmentation at most loads. Also, the delay-first algorithms have lower loss as compared to the segment-first algorithms. This is due to the possible blocking of the rescheduled burst segment by the recently scheduled nonoverlapping burst segment in the segment-first algorithms. The loss obtained by delay-first algorithms is the lower bound on delay for the segment-first algorithms. We observe that at any given load, the NP-DFMOC and NP-DFMOC-VF algorithms perform the best, since the unscheduled burst is delayed; and in case there is still a contention, the burst is segmented and only the overlapping burst segment is dropped. The segment-first algorithms lose packets proportional to the switching time every time there is a contention, while the LAUC and LAUC-VF algorithms delay the burst in case of a contention and schedule the burst if the channel is free after the provided delay. Hence, at low loads, LAUC-VF performs better than NP-SFMOC-VF, and, as the load increases, NP-SFMOC-VF performs better. Therefore, a substantial gain is achieved by using segmentation and FDLs.

Fig. 14(b) plots the average per-hop FDL delay versus load for different channel scheduling algorithms. We observe that the delay-first algorithms have higher per-hop FDL delay as compared to the segment-first algorithms, since FDL is the primary contention-resolution technique in the former and segmentation is the primary contention-resolution technique in the latter. We also observe that the per-hop FDL delay of void-filling algorithms is lower than nonvoid filling, since the scheduler can assign the arriving bursts to closer voids that incur lower FDL delay as compared to scheduling the bursts at the end of the horizon (LAUT) in the case of non-void-filling algorithms. Hence, we can carefully choose either delay-first or

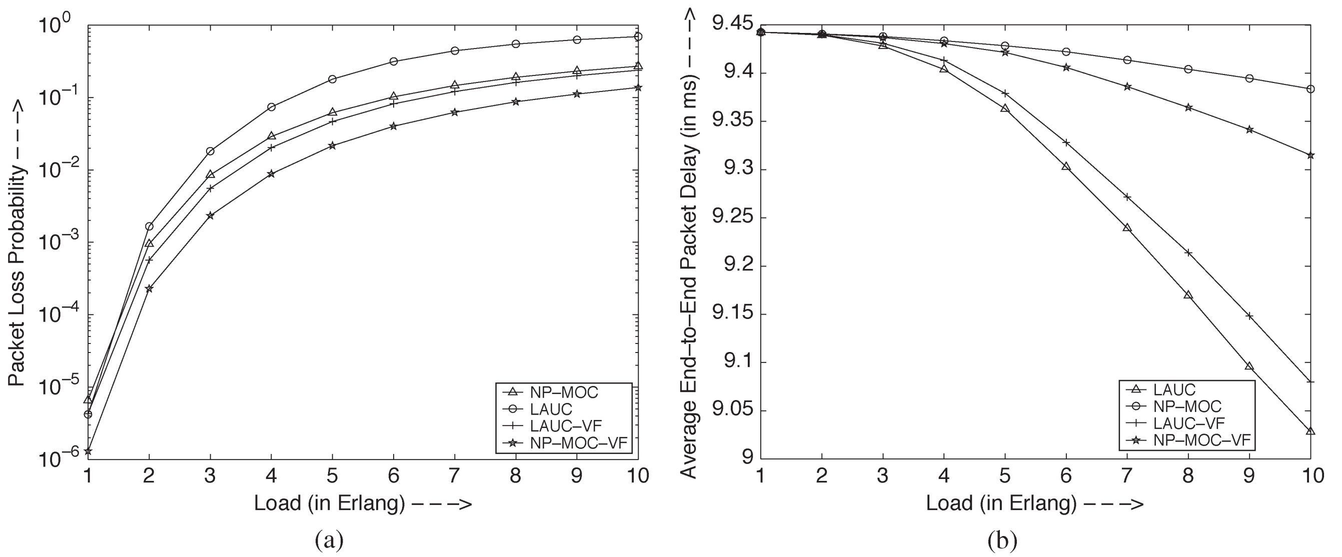


Fig. 13. (a) Packet-loss probability versus load, and (b) average end-to-end delay versus load for different scheduling algorithms with eight data channels on each link, for the NSF network.

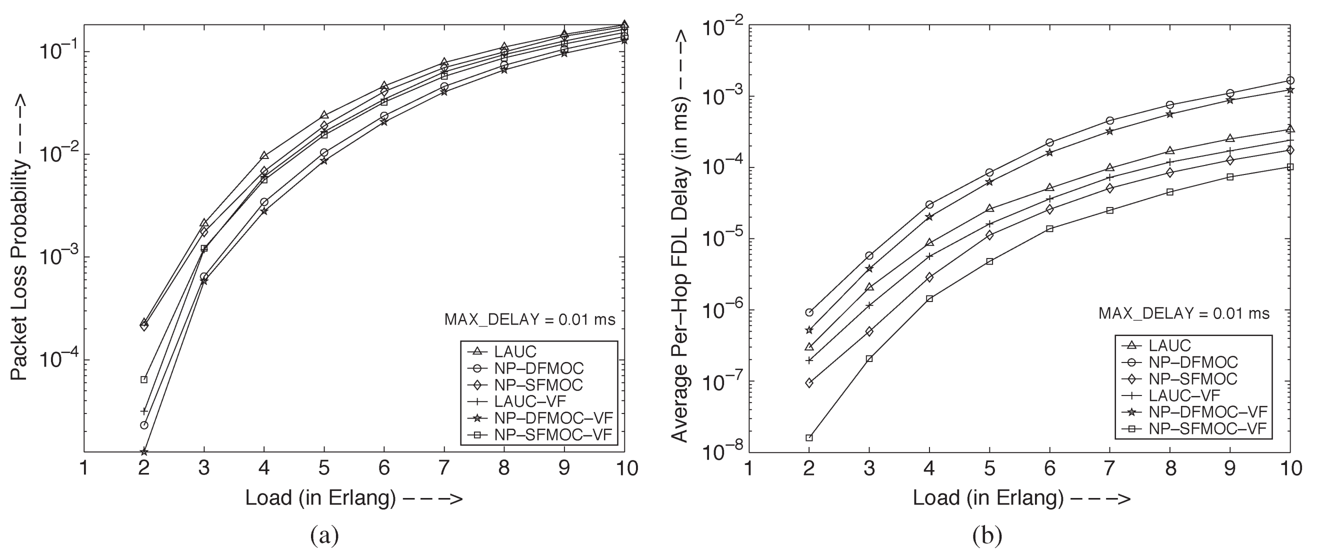


Fig. 14. (a) Packet-loss probability versus load, and (b) average per-hop FDL delay versus load for different scheduling algorithms with eight data channels on each link, for the NSF network.

segment-first schemes based on loss and delay tolerances of input IP packets.

When a high MAX_DELAY value is used, algorithms that use FDLs as the primary contention-resolution technique, such as LAUC, LAUC-VF, NP-DFMOC, and NP-DFMOC-VF, outperform the algorithms that use segmentation as the primary contention-resolution technique, such as NP-SFMOC and NP-SFMOC-VF [22].

VII. Conclusion

In this paper, we considered burst segmentation and FDLs for burst scheduling in optical burst-switched networks, and we proposed a number of channel scheduling algorithms for OBS networks. The segmentation-based scheduling algorithms perform better than the existing scheduling algorithms with and without void filling in terms of packet loss. We also introduced two categories of scheduling algorithms based on the FDL architecture. The delay-first algorithms are suitable for transmitting packets that have higher delay tolerance and strict loss constraints, while the segment-first algorithms are suitable for transmitting packets that have higher loss tolerance and strict delay constraints.

Areas of future works include extending the proposed algorithms to support QoS. In the case of providing QoS support, the priority of the burst can be stored in the BHP [25], and the scheduling algorithm can dynamically decide which contending bursts or burst segments to drop using burst segmentation, or to delay using FDLs. Hence, a combination of preemptive and nonpreemptive segmentation-based techniques can be used to provide service differentiation. It would also be useful to develop an accurate analytical model based on [26] and [27] for the proposed scheduling algorithms with realistic assumptions.

REFERENCES

[1] C. Qiao and M. Yoo, "Optical burst switching (OBS)—A new paradigm for an optical internet," *J. High Speed Netw.*, vol. 8, no. 1, pp. 69–84, Jan. 1999.

[2] A. H. Zaim, I. Baldine, M. Cassada, G. N. Rouskas, H. G. Perros, and D. Stevenson, "The JumpStart just-in-time signaling protocol: A formal description using EFSM," *Opt. Eng.*, vol. 42, no. 2, pp. 568–585, Feb. 2003.

[3] I. Baldine, G. N. Rouskas, H. G. Perros, and D. Stevenson, "Jumpstart: A just-in-time signaling architecture for WDM burst-switched networks," *IEEE Commun. Mag.*, vol. 40, no. 2, pp. 82–89, Feb. 2002.

[4] J. Y. Wei, J. L. Pastor, R. S. Ramamurthy, and Y. Tsai, "Just-in-time optical burst switching for multi-wavelength networks," in *Proc. IFIP TC6 Int. Conf. Broadband Communications*, Hong Kong, Nov. 1999, pp. 339–352.

[5] S. Yao, B. Mukherjee, S. J. B. Yoo, and S. Dixit, "All-optical packet-switched networks: A study of contention resolution schemes in an irregular mesh network with variable-sized packets," in *Proc. SPIE Optical Networking and Communications (OptiComm)*, Dallas, TX, Oct. 2000, pp. 235–246.

[6] I. Chlamtac, A. Fumagalli, L. G. Kazovsky *et al.*, "CORD: Contention resolution by delay lines," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 5, pp. 1014–1029, Jun. 1996.

[7] R. Ramaswami and K. N. Sivarajan, "Routing and wavelength assignment in all-optical networks," *IEEE/ACM Trans. Netw.*, vol. 3, no. 5, pp. 489–500, Oct. 1995.

[8] A. S. Acampora and I. A. Shah, "Multihop lightwave networks: A comparison of store-and-forward and hot-potato routing," *IEEE Trans. Commun.*, vol. 40, no. 6, pp. 1082–1090, Jun. 1992.

[9] A. Bononi, G. A. Castanon, and O. K. Tonguz, "Analysis of hot-potato optical networks with wavelength conversion," *J. Lightw. Technol.*, vol. 17, no. 4, pp. 525–534, Apr. 1999.

[10] F. Forghieri, A. Bononi, and P. R. Prucnal, "Analysis and comparison of hot-potato and single-buffer deflection routing in very high bit rate optical mesh networks," *IEEE Trans. Commun.*, vol. 43, no. 1, pp. 88–98, Jan. 1995.

[11] V. M. Vokkarane, J. P. Jue, and S. Sitaraman, "Burst segmentation: An approach for reducing packet loss in optical burst switched networks," in *Proc. IEEE Int. Conf. Communications (ICC)*, New York, Apr. 2002, vol. 5, pp. 2673–2677.

[12] J. S. Turner, "Terabit burst switching," *J. High Speed Netw.*, vol. 8, no. 1, pp. 3–16, Jan. 1999.

[13] Y. Xiong, M. Vanderhoute, and H. C. Cankaya, "Control architecture in optical burst-switched WDM networks," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 10, pp. 1838–1854, Oct. 2000.

[14] V. M. Vokkarane and J. P. Jue, "Burst segmentation: An approach for reducing packet loss in optical burst switched networks," *SPIE Opt. Netw. Mag.*, vol. 4, no. 6, pp. 81–89, Nov.–Dec. 2003.

[15] V. M. Vokkarane, Q. Zhang, J. P. Jue, and B. Chen, "Generalized burst assembly and scheduling techniques for QoS support in optical burst-switched networks," in *Proc. IEEE Global Telecommunications (GLOBECOM)*, Taipei, Taiwan, Nov. 2002, vol. 3, pp. 2747–2751.

[16] L. Tancevski, A. Ge, G. Castanon, and L. Tamil, "A new scheduling algorithm for asynchronous, variable length IP traffic incorporating void filling," in *Proc. Optical Fiber Communication Conf. (OFC)*, San Diego, CA, Feb. 1999, vol. 3, pp. 180–182.

[17] M. Iizuka, M. Sakuta, Y. Nishino, and I. Sasase, "A scheduling algorithm minimizing voids generated by arriving bursts in optical burst switched WDM network," in *Proc. IEEE Global Telecommunications (GLOBECOM)*, Taipei, Taiwan, 2002, vol. 3, pp. 2736–2740.

[18] J. Xu, C. Qiao, J. Li, and G. Xu, "Efficient channel scheduling algorithms in optical burst switched networks," in *Proc. IEEE Information Communications (INFOCOM)*, San Francisco, CA, Mar. 2003, vol. 3, pp. 2268–2278.

[19] F. Farahmand and J. P. Jue, "Look-ahead window contention resolution in optical burst switched networks," in *Proc. IEEE Workshop High Performance Switching and Routing*, Torino, Italy, Jun. 2003, pp. 147–151.

[20] S. Charcranoon, T. S. El-Bawab, H. C. Cankaya, and J. Shin, "Group-scheduling for optical burst switched (OBS) networks," in *Proc. IEEE Global Telecommunications (GLOBECOM)*, San Francisco, CA, Dec. 2003, pp. 2745–2749.

[21] C. Gauger, "Dimensioning of FDL buffers for optical burst switching nodes," in *Proc. IFIP Conf. Optical Network Design and Modeling (ONDM)*, Torino, Italy, Feb. 2002, pp. 117–132.

[22] V. M. Vokkarane, G. P. V. Thodime, V. B. T. Challagulla, and J. P. Jue, "Channel scheduling algorithms using burst segmentation and FDLs for optical burst-switched networks," in *Proc. IEEE Int. Conf. Communications (ICC)*, Seattle, WA, May 2003, vol. 2, pp. 1443–1447.

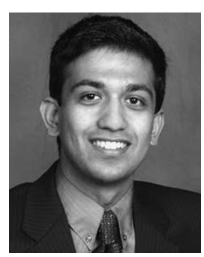
[23] A. Neukermans and R. Ramaswami, "Mems technology for optical networking applications," *IEEE Commun. Mag.*, vol. 39, no. 1, pp. 62–69, Jan. 2001.

[24] L. Rau, S. Rangarajan, D. J. Blumenthal, H.-F. Chou, Y.-J. Chiu, and J. E. Bowers, "Two-hop all-optical label swapping with variable length 80 Gb/s packets and 10 Gb/s labels using nonlinear fiber wavelength converters, unicast/multicast output and a single EAM for 80- to 10 Gb/s packet demultiplexing," in *Proc. Optical Fiber Communication Conf. (OFC)*, Anaheim, CA, Mar. 2002, pp. FD2-1–FD2-3.

[25] V. M. Vokkarane and J. P. Jue, "Prioritized routing and burst segmentation for QoS in optical burst-switched networks," in *Proc. Optical Fiber Communication Conf. (OFC)*, Anaheim, CA, Mar. 2002, vol. WG6, pp. 221–222.

[26] ——, "Prioritized burst segmentation and composite burst assembly techniques for QoS support in optical burst switched networks," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 7, pp. 1198–1209, Sep. 2003.

[27] A. Detti, V. Eramo, and M. Listanti, "Performance evaluation of a new technique for IP support in a WDM optical network: Optical composite burst switching (OCBS)," *J. Lightw. Technol.*, vol. 20, no. 2, pp. 154–165, Feb. 2002.



Vinod M. Vokkarane (S'02–M'04) received the B.Eng. degree (with honors) in computer science and engineering from the University of Mysore, Mysore, India, in 1999, and the M.S. and Ph.D. degrees in computer science from the University of Texas, Dallas, in 2001 and 2004, respectively.

He is currently an Assistant Professor at the Department of Computer and Information Science, University of Massachusetts at Dartmouth. His primary areas of research are in optical and wireless networking, with a special focus on the design of architectures and protocols for optical networks.

Dr. Vokkarane is a recipient of the University of Texas at Dallas Best Computer Science Ph.D. Dissertation Award (2003–2004).

Jason P. Jue (M'99–SM'04) received the B.S. degree in electrical engineering from the University of California, Berkeley, in 1990, the M.S. degree in electrical engineering from the University of California, Los Angeles, in 1991, and the Ph.D. degree in computer engineering from the University of California, Davis, in 1999.

He is currently an Associate Professor at the Department of Computer Science, University of Texas at Dallas. His research interests include optical networks, network control and management, and network survivability.