Traffic Statistics and Performance Evaluation in Optical Burst Switched Networks

Xiang Yu, Jikai Li, Xiaojun Cao, Yang Chen, and Chunming Qiao

***Abstract*—Optical burst switching (OBS) is a promising switching technology to exploit the potential benefits of optical communication and, at the same time, support statistical multiplexing of data traffic at a fine granularity. To quantify its benefits, the paper describes several typical burst assembly algorithms and studies their impact on the assembled burst traffic characteristics as well as the performance of TCP traffic. Also described is a proactive burst scheduling algorithm, called burst overlap reduction algorithm (BORA), which schedules locally assembled bursts in such a way as to reduce burst contention at downstream nodes in OBS networks. Furthermore, to provide analytical insights into performance evaluation of OBS networks, a burst loss model at an OBS node and its extension to different reservation protocols are presented.**

***Index Terms*—Assembly, optical burst switching, reservation, scheduling, TCP, traffic characteristic.**

I. Introduction

SEVERAL optical network paradigms for future Internet backbone have been under intensive research. Among these paradigms, optical circuit switching (OCS) is relatively easy to implement but lacks flexibility to cope with the fluctuating traffic and the changing link status. On the other hand, optical packet switching (OPS) provides statistical multiplexing of bursty traffic at packet level in the backbone, but the required technologies such as optical buffer and optical logic are too immature for it to happen anytime soon. A promising approach called optical burst switching (OBS) that combines the advantage of OCS and OPS was proposed by researchers [1]–[3] and has received an increasing amount of attention from both academia and industry worldwide [4]–[7]. Similar to OCS, OBS reserves bandwidth in advance via out-of-band signaling; thus, it eliminates the need for an optical buffer or even fiber delay lines (FDLs) required by OPS and enables data to be switched "transparently" (in terms of bit rates, coding format, and protocols) throughout the network. Unlike OCS, it provides statistical multiplexing at burst level at intermediate nodes without having to incur the circuit setup delay (which is equal to at least the round-trip propagation time) [3].

Manuscript received November 26, 2003; revised June 21, 2004.

X. Yu and C. Qiao are with the Department of Computer Science and Engineering, State University of New York at Buffalo, Buffalo, NY 14260 USA (e-mail: xiangyu@cse.buffalo.edu; qiao@cse.buffalo.edu).

J. Li is with the Department of Computer Science, The College of New Jersey, Ewing, NJ 08628 USA (e-mail: jikaili@computer.org).

X. Cao is with the Department of Information Technology, Rochester Institute of Technology, Rochester, NY 14623 USA (e-mail: cao@it.rit.edu).

Y. Chen is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: yangchen@cc.gatech.edu).

Digital Object Identifier 10.1109/JLT.2004.833527

An OBS network is a collection of interconnected OBS nodes. An ingress OBS node assembles packets from local access networks, for example, Internet protocol (IP) packets, into bursts and sends out a corresponding control packet (CP) for each data burst. This CP carries specific information about the corresponding burst such as the offset time and the burst length and is delivered out-of-band and leads the data burst by an offset time. Based on the information in the CP, each intermediate node along the way, from the ingress node to the egress node, searches for an available time period on a wavelength at the desired output port and makes a reservation for the arriving burst (this is also called *burst scheduling*). When the burst reaches the intermediate node, the burst is switched to the output port without any extra delay or optical–electrical–optical (O/E/O) conversion. The burst will be disassembled at the egress node and reach the destination via local access networks. Accordingly, the function of a general OBS node can be divided into two parts as shown in Fig. 1, namely, edge node function (which is also called assembly node in the following discussion) and core node function. In OBS networks using one-way reservation protocols, such as tell-and-go (TAG) [8] and just-enough-time (JET), burst loss is a major concern [1] because bursts encountering contention cannot be buffered at any intermediate node due to the lack of optical RAM (an FDL, if available at all, can only provide a limited delay and contention resolution capability).

Since multiple packets are assembled into a burst, the burst traffic is expected to have different statistical characteristics from those of the input packet traffic [9], [10]. Different traffic characteristics (e.g., burst length and interarrival time distribution) will have a different impact on the burst loss rate, which is a critical performance metric in OBS networks. Thus, it is essential to study the traffic characteristics in OBS networks. Given that TCP is today's prevailing transport protocol and also likely to be adopted in future optical networks, it is interesting to find out how congestion control and retransmission mechanisms in different TCP implementations, i.e., Reno, New-Reno, and SACK [11]–[13], perform in OBS networks as a result of their interaction with burst assembly. In addition, in order to reduce burst loss and improve TCP performance, it is important to design efficient scheduling algorithms that can avoid and/or resolve burst contention. In this paper, we also investigate several typical burst scheduling algorithms and introduce an efficient scheduling algorithm called burst overlap reduction algorithm (BORA), which reduces burst loss by proactively avoiding burst contentions at downstream nodes. Finally, a model to analyze the burst loss rate at an OBS node

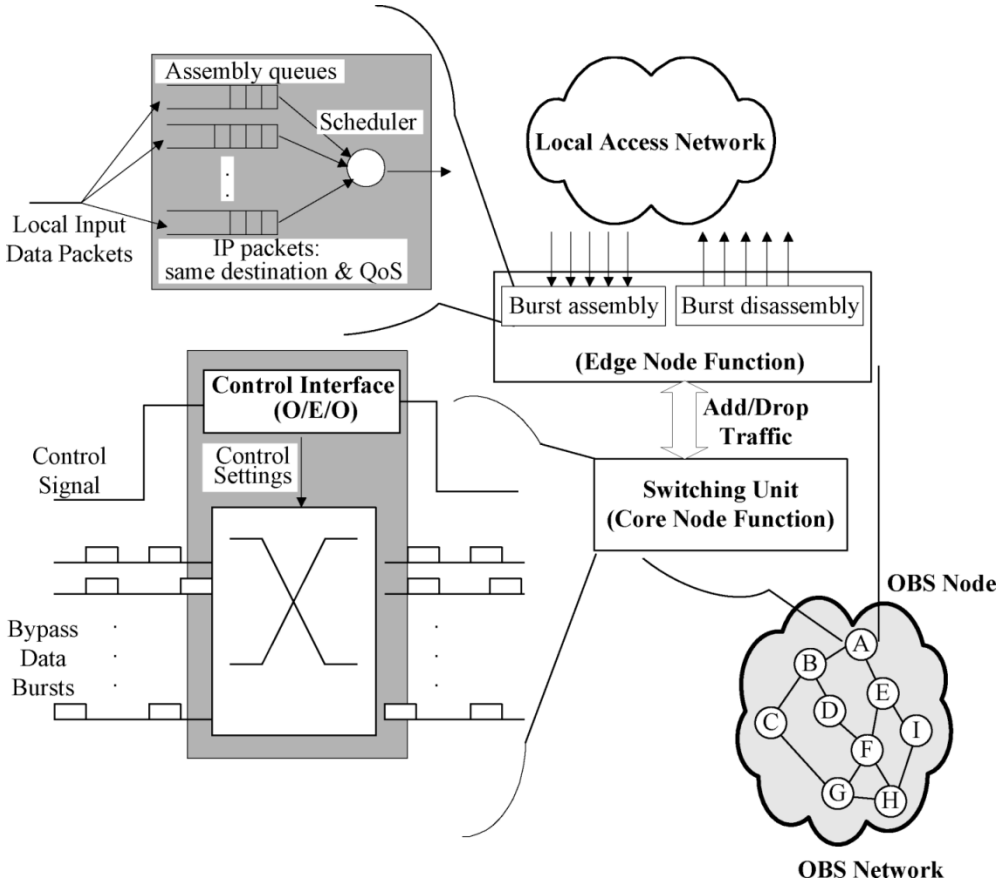


Fig. 1. OBS node architecture.

and its applications to different reservation protocols are also provided.

This paper is organized as follows. We introduce various assembly algorithms in Section II and elaborate on the assembled burst traffic characteristics and their impact on the performance of OBS network in Section III. The performance of TCP traffic in OBS networks is presented in Section IV. Different burst reservation protocols are discussed in Section V, and a proactive scheduling algorithm is presented in Section VI. Based on the statistical study, a burst loss model is provided, which is compared with conventional loss models in Section VII. Section VIII concludes this paper.

II. Assembly Algorithms in Optical Burst Switched Networks

Various burst assembly algorithms have been proposed to aggregate packets into bursts that are the basic transmission units in OBS networks. In general, each assembly node maintains multiple queues to hold the data packets from local access networks according to their destinations and quality-of-service (QoS) requirements. The assembly algorithms (to be introduced subsequently) choose a number of packets from an assembly queue and send them out as a burst to the OBS core node.

Most of the assembly algorithms use either assembly time or burst length or both as the criterion to create bursts [9], [14]. The parameters involved include a time threshold T, a burst length threshold B, and a minimum burst length requirement $B_{\min}$. Threshold T limits the delay of packets in the assembly queue within a maximum value when traffic load is low, and B reduces the unnecessary delay of a burst when traffic load is high (as will be explained subsequently). If the assembled burst length is shorter than $B_{\min}$ by the time the burst is to be sent out, the burst is padded to meet the requirement specified by $B_{\min}$ [15]. The time and length criteria can be either fixed or adjusted dynamically. We can classify the assembly algorithms into four categories, described hereafter.

The first category contains time-based assembly algorithms, whereby a fixed threshold T acts as the primary criterion to create a burst. At the same time, the burst size is also required to be no smaller than $B_{\min}$ and is sent out with padding bits up to $B_{\min}$ otherwise. The second category contains burst-length-based assembly algorithms, whereby a burst is sent out once the burst length reaches or exceeds the given burst length threshold B.

However, each of the above two types of assembly algorithms has its shortcomings under certain circumstances. Since a burst-length-based algorithm does not have a constraint on a packet's waiting time in the assembly queue, when the traffic load is low, there is no bound on either the maximum or average delay for packets as shown by the dotted line in Fig. 2. A time-based algorithm can solve this problem and will form the burst at point P_2. However, under heavy traffic load, it will have a longer average delay than the burst-length-based algorithm, as shown by the dashed line, because the latter will send out the burst at point P_1. In addition, the burst length could have a large variance for the

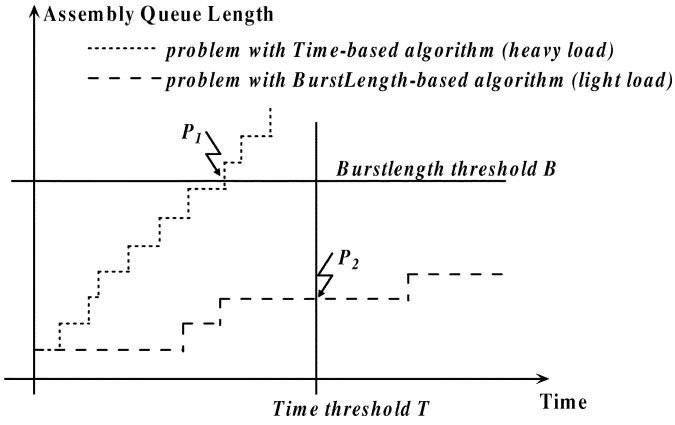


Fig. 2. Burst assembly process.

time-based algorithm, which may affect the scheduling performance at OBS core nodes. It is preferable to have an assembly algorithm that performs well under all load conditions, i.e., a burst will be sent at point P_1 in heavy load and at point P_2 when the load is low. This is the basic idea of the third category of assembly algorithms called the mixed time/burst-length assembly algorithm, which uses all the three thresholds: time threshold T, burst length threshold B, and minimum burst-length requirement $B_{\min}$ [9]. Note that if the burst length threshold B is much larger than the average amount of data that arrives in time T, i.e., $T \times \lambda_p \ll B$ (where λ_p is the average packet arrival rate to the assembly queue), the assembled burst's length will never reach B before it is sent at point P_2, as illustrated in Fig. 2 by the dotted line. In such a case, mixed time/burst-length assembly algorithm will function in the same way as the time-based burst assembly algorithm with a fixed time threshold T. On the other hand, if the burst-length threshold B is much smaller than the average data that arrives in time T, i.e., $T \times \lambda \gg B$, the time threshold T will never be reached before a burst's length reaches B at point P_1, as illustrated in Fig. 2 by the dashed line, and the mixed time/burst-length assembly algorithm operates like the burst-length-based assembly algorithm.

The fourth category of assembly algorithms is based on dynamic threshold, where either the time threshold or the burst-length threshold or both are adjusted dynamically according to real-time traffic measurements and prediction [14], [16]. These dynamic assembly algorithms are more adaptive to network traffic conditions and can provide better performance for strongly correlated traffic such as TCP or long-range-dependent traffic in OBS networks, as will be shown later in Section IV. Note that the dynamic assembly algorithms involve more computational complexity (in predicting and adjusting the thresholds) than the first three categories of assembly algorithms.

One interesting issue involved in the dynamic threshold assembly algorithm as well as other assembly algorithms is how to predict the next burst length (or the next assembly time). With such a prediction, the CP of a burst can be sent to the OBS network before the creation of a burst, and thus the pretransmission delay introduced by burst assembly is reduced. The prediction is especially important for the dynamic threshold assembly algorithm because the thresholds for either the next burst length or the next assembly time or both need to be set to a proper value according to predicted incoming traffic rate. A linear prediction method to predict the next burst length based on traffic correlations was proposed in [17].

III. Assembled Burst Traffic Characteristics

In this section, we describe the characteristics of assembled burst traffic, which is the output of an OBS assembly node as well as the input to an OBS core node. We will focus on both short-range characteristics and long-range dependency. The short-range characteristics include the distribution of burst size and burst interarrival time, as well as the variance and correlation of the arrival process in short time scales. While the long-range dependency is in terms of correlation structures of the traffic in large to infinity time scales.

A. Short-Range Characteristics

Since both time-based and burst-length-based assembly algorithms assemble multiple packets that arrive in a certain time period to a burst, most packets arriving at an assembly queue have to wait some amount of time in the electronic buffer until a threshold is reached. This buffering process changes the characteristics of the data traffic, more specifically, from packet size to burst size, and from packet interarrival time to burst interarrival time. In order to analyze the performance of an OBS network, we need to understand the assembled burst traffic process first. Hereafter, we will describe the distributions of burst interarrival time and burst length, respectively, for both time-based and burst-length-based assembly algorithms. Since the packet traffic arriving at an assembly queue is an aggregated process from many independent packet traffic flows, we assume the aggregated packet traffic to be a Poisson traffic, which has an exponentially distributed interarrival time t with mean value $\mu = 0.1$. For simplicity, we also assume that the data packet size is fixed as $P = 10$ units.

With a time-based assembly algorithm, one burst is sent every T (time units). The interarrival time of bursts is also fixed as T. The burst size equals the sum of all packets arriving in T. Since the packet size is fixed as P according to our assumption, the burst size is $l = P \times n_p$, where n_p is the number of packets contained in the burst. The distribution of burst size can thus be calculated as

$$\begin{aligned} Pr\{\text{Burstlength} = l\} &= Pr\{n_p \times P = l\} \\ &= \frac{\left(\frac{T}{\mu}\right)^{l/P}}{\left(\frac{l}{P}-1\right)!} e^{-T/\mu} \end{aligned} \tag{1}$$

which approaches a Gamma distribution and will approach Gaussian distribution with mean $PE[n_p]$ and variance $P^2E[n_p]$ when T/μ is large enough. Fig. 3(a) illustrated the Gaussian-distributed burst size (normalized with respect to the average number of packets contained in a burst $E[n_p] = T \times \lambda_p$, where $\lambda_p = 1/\mu$ is the average number of packets arrived in a unit time) in time-based assembly algorithm. In the simulation, Gaussian distribution (1) has a mean value $10 \times 10 = 100$ and variance $10^2 \times 10 = 1000$ and Gaussian distribution (2) has a mean value $10 \times 50 = 500$ and

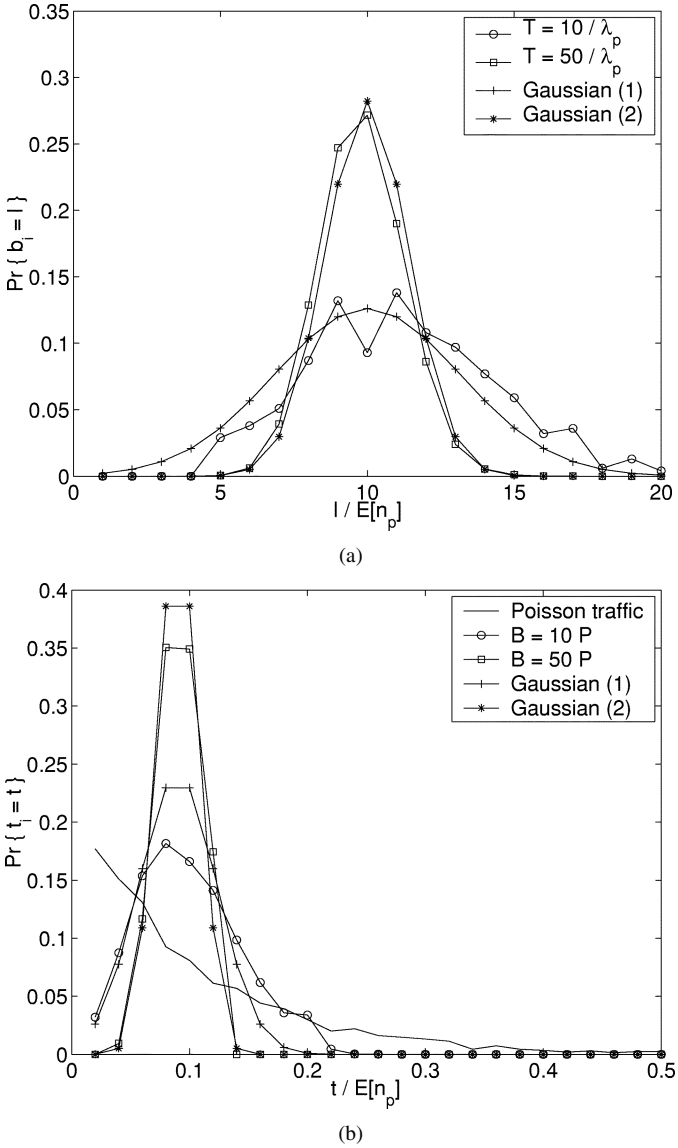


Fig. 3. Short-range statistics. (a) Burst size distribution in time-based assembly algorithm. (b) Burst interarrival time distribution in burst-length-based assembly algorithm.

a variance $10^2 \times 50 = 5000$. Therefore, the assembled burst traffic from time-based assembly algorithm has a fixed burst interarrival time and Gaussian-distributed burst lengths.

With a burst-length-based assembly algorithm, since the packet size P and burst-length threshold B are both fixed, the number of packets contained in a burst should also be fixed as $n_p = E[n_p] = B/P$. Since packets' interarrival times are independent random variables, the burst interarrival time, which is the sum of the interarrival times of all packets aggregated in the burst, is the sum of n_p such variables and, from the central limit theorem, should have a Gaussian distribution (when n_p is large) with mean $\mu E[n_p]$ and variance $\mu^2 E[n_p]$. Fig. 3(b) illustrates the Gaussian-distributed burst interarrival time (normalized with respect to $E[n_p]$) from simulation with varying burst-length thresholds. In the simulation, Gaussian distribution (1) has a mean value $0.1 \times 10 = 1$ and a variance of $0.1^2 \times 10 = 0.1$ and Gaussian distribution (2) has a mean value $0.1 \times 50 = 5$ and a variance of $0.1^2 \times 50 = 0.5$. Therefore, the assembled burst traffic from burstlength-based assembly algorithm has a fixed burst size and Gaussian distributed burst inter-arrival times.

The traffic characteristic of the assembled burst traffic from a mixed time-burstlength assembly algorithm is a mixture of those from the above time-based and burstlength-based assembly algorithms, i.e., a "Gaussian distribution" with bounded marginal distributions, because the outcome of the mixed time-burstlength based algorithm would be the same as either of the above two assembly algorithms for any given burst.

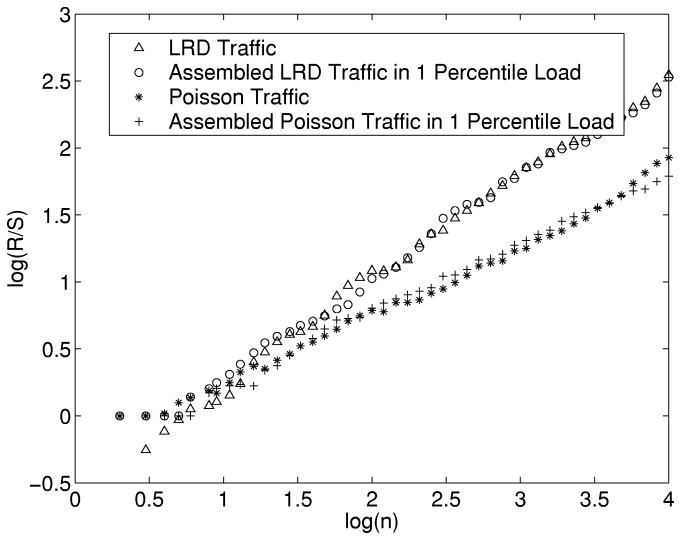
Using dynamic assembly algorithms, both the burst inter-arrival time and burst length depend on the traffic arrival rate as well as the predicted assembly parameters, which vary from time to time. Thus the short range traffic characteristics of the assembled burst traffic from these dynamic assembly algorithms will be quite different from the above results, and beyond the scope of this paper.

B. Long Range Dependency

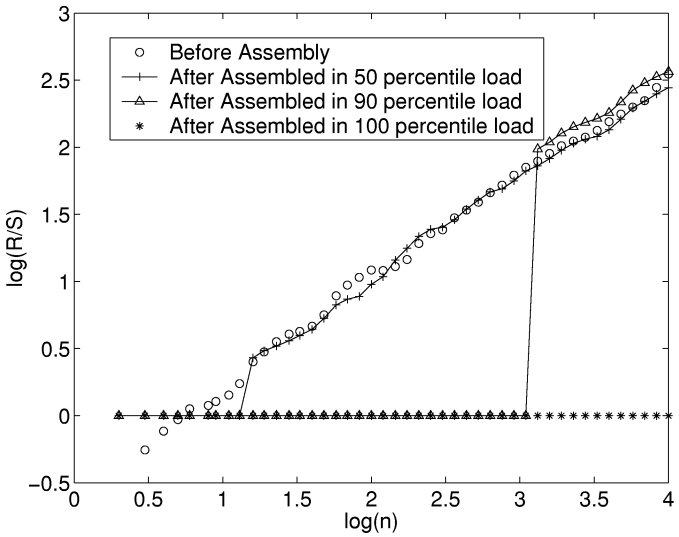
Another important aspect of today's Internet traffic is the long range dependency [18]. As mentioned earlier, the long range dependency exists in traffic processes that correlate at large or even infinite time scales. It degrades network performance and decreases network resource utilization because of increased data loss and heavy delays. Previous work [15] claimed that burst assembly algorithms could reduce the long range dependency in the unassembled packet traffic, whereas [19] and [9] pointed out that the long range dependency will not change after assembly. Here we analyze the long range dependency in the assembled burst traffic from the time-based assembly algorithm in two traffic scenarios with a light traffic load and a medium to heavy traffic load, respectively at an assembly node.

In the scenario with a light traffic load, the burst size is small. Therefore, the processing time of a burst at an assembly node (including the time to schedule and transmit the burst) is relatively short compared to the burst inter-leaving time. The assembly queues are always empty and the next burst is created after the previous burst is sent out. Clearly, traffic statistics only change within the assembly time period because previous bursts will not affect (or delay) the following bursts departure time. Compared to the large or even infinity time scales of interest in long range dependency study, this assembly time period is trivial. Hence, the assembled burst traffic still has the same long range dependency.

In the scenario with a medium to heavy traffic load at an assembly node, the burst size is large and the burst processing time is relatively large compared to the burst inter-leaving time, and thus the assembly queues may not be empty when the next burst is created and ready to depart. In this case, the previous bursts will affect (or delay) the departure time of the following bursts, and the queuing process in the assembly buffer further changes the assembled burst traffic from Gaussian distributed (long range dependent) process to an almost constant rate process as long as there are enough arriving packets to keep the assembly queue occupied. However, this change, or smoothing effect, is similar to that of the buffering of any traffic in electronic switched networks in that the extent of the smoothing



(a)



(b)

Fig. 4. RS plot. (a) Poisson and LRD traffic before and after assembled under light load at assembly node. (b) LRD traffic assembled under medium to heavy load in assembly node.

effect depends on the load. In particular, a buffer with finite queue length cannot reduce the long-range dependency in a traffic process. More specifically, with the increase in the load, the assembled traffic may become more like a constant rate flow. However, the correlation structure at large to infinity time scales still does not change, as will be shown in subsequent simulations.

Fig. 4(a) and (b) shows the RS plot (which estimates the Hurst parameter by measuring the traffic-load variance in different time scales, and the slope of the plot illustrates the degree of long-range dependency in a process. For details of the RS method, readers are referred to [20] for different traffic-load scenarios. In the light traffic-load scenario, as illustrated in Fig. 4(a), the RS slopes for the assembled burst traffic and unassembled input packet traffic (either long-range-dependent or Poisson traffic) are the same, except on the small time scales (when $\log(n) < 0.6$ or there is on average less than $10^{0.6} < 4$ bursts within each time unit). In other words, the long-range

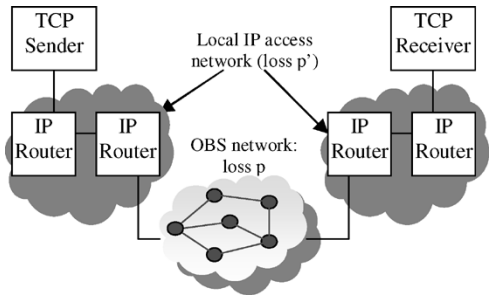


Fig. 5. TCP/IP over OBS networks.

dependency cannot be reduced by the burst assembly process. The difference in the slopes for the assembled and unassembled long-range-dependent (LRD) traffic for $\log(n) < 0.6$ is due to the fact that the burst assembly procedure does change the traffic statistics on the small time scale (e.g., comparable to the assembly period) in the light-load scenario. Fig. 4(b) illustrates that in the medium-to-heavy-load scenario, the slope of the RS plot is reduced to zero on some time scales larger than those comparable to the assembly period. This implies that on these time scales (with respect to the traffic load), the assembly process smooths the input traffic to a constant-rate burst traffic, for which the variance (or the correlation) of the traffic arriving within a burst assembly period is too small to show up in the RS plot. However, for a given load (e.g., 50 or 90 percentile), the RS slope of the assembled burst traffic on larger time scales (e.g., when there are more than $10^{1.2}$ or $10^{3.1}$ bursts within each time unit) is the same as that of the unassembled packet traffic, implying that they have the same long-range dependency. Note that in the extreme case where the load is 100 percentile, the assembled burst traffic becomes a strictly constant rate flow, and hence the RS slope becomes zero.

Although these studies on the long-range dependency in the assembled burst traffic are for time-based assembly algorithms, other assembly algorithms with a finite assembly time will also change the short-range characteristics only, without changing long-range dependency in the assembled burst traffic.

IV. Performance of TCP Traffic in OBS Networks

Since TCP traffic is and may remain to be the most popular traffic type in the Internet backbone, it is important to evaluate the performance of TCP in OBS networks. In a TCP/IP over OBS network, a TCP sender and receiver are connected to an OBS network through local IP access networks, as shown in Fig. 5. There are losses in both of the two local IP access networks as well as in the OBS network. The packet losses in the IP access networks affect TCP performance in the same way as in packet-switched networks as studied in [21], [13], and [22], while the burst losses in the OBS network could have a different impact.

As the TCP sender dynamically adjusts its sending window according to perceived network congestions, TCP performance in OBS networks could be affected not only by the loss rate but also by the assembly algorithms, burst loss pattern, and TCP implementations, which will be discussed in the following subsections.

A. Impact of Burst Assembly

The TCP's performance (e.g., throughput) is sensitive to burst assembly delay (in addition to other factors) as its sender may retransmit and adjust its congestion window according to a timer set on each sent segment. Therefore, a time-based assembly algorithm could perform better than burst-length-based assembly algorithm for a TCP Reno flow, as shown in Fig. 6.

The value of the parameter T in the time-based algorithm or B in the burst-length-based algorithm can affect the throughput of TCP in an OBS network. In an OBS network with a fixed burst loss rate (e.g., $p = 0.01$), there could exist an optimal T for the time-based algorithm or B for the burst-length-based algorithm that maximizes the throughput, as shown in Fig. 6. (In terms of loss performance for general Poisson input traffic, [23] also reported the existence of optimal burst length threshold.) For a mixed time/burst-length-based algorithm, since the burst assembly can be triggered by either T or B, we could expect its performance to be between that of the previous two algorithms.

As mentioned in Section II, a dynamic assembly algorithm adjusts the threshold values (e.g., T or B or both) according to the traffic statistics collected in real time to fit better with TCP dynamic congestion control and thus could have a better performance than the time-based assembly algorithms. Fig. 7(a) and (b) shows the performance improvements in terms of goodput (i.e., the effective data that the destinations receive) and loss rate for TCP Reno traffic using a dynamic assembly algorithm [14], where T changes according to traffic arrival rates.

To highlight the impact of burst assembly on TCP throughput, we focus our study on the simple time-based assembly algorithm under the assumption that a TCP sender/receiver has a constant access bandwidth λ_p to the OBS assembly node via the local IP access networks.

A burst is the transmission unit in OBS networks, which contains a number of consecutive packets/segments.[1] Burst losses in an OBS network differ from packet losses in conventional packet-switched networks in two aspects: 1) a burst loss could result in the loss of several consecutive packets contained in the burst and 2) due to the bufferless nature of the OBS core, burst losses are the results of burst contentions and are less correlated than packet losses caused by buffer overflows.

When a burst is lost, TCP senders will be notified of the loss of a number of consecutive packets contained in the lost burst, either by information from ACKs or by an expired timer set for each packet sent out. The ACKs indicating a loss could be a duplicate or partial ACKs (which acknowledge a new packet but not the packet with the highest sequence number sent) as in Reno and New-Reno or ACKs with missing block information in receiver's window as in SACK, and such a loss is denoted as a triple duplicate (TD) loss. Accordingly, the loss indicated by an expired timer is denoted as a time-out (TO) loss. The consecutive packet loss pattern in OBS networks triggers different retransmission pattern and results in delayed first loss (DFL) gain and retransmission penalty, which will be discussed subsequently. These OBS-specific factors will affect TCP throughout.

[1]In this paper, we assume that one TCP segment is always contained in one IP packet, and the packet and segment will be interchangeably used.

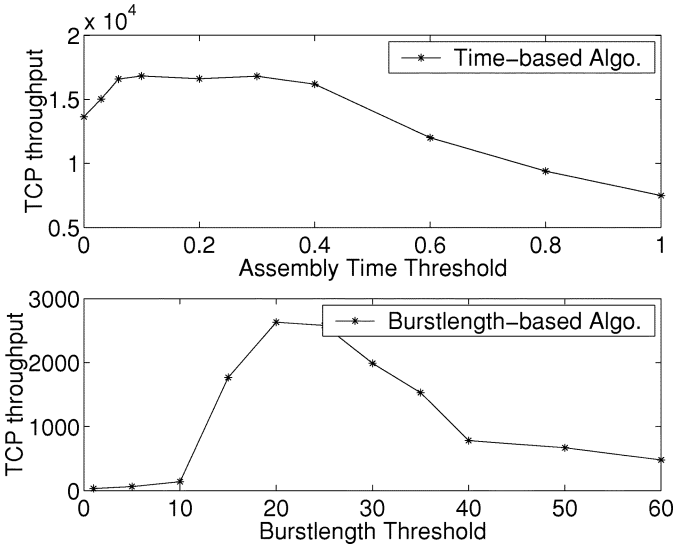


Fig. 6. Impact of parameters from assembly algorithm on TCP Reno performance.

In the remaining part of this subsection, we will discuss the TCP throughput gains and penalties in OBS networks as identified in [24]

1) Delay Penalty and Loss Penalty: The delay penalty is mainly caused by burst assembly and disassembly delays, which increase the TCP round-trip time (as well as TCP TO value) and decrease TCP throughput.

Given the assumption that a constant assembly time T is used for a time-based assembly algorithm, we may denote the TCP round-trip time as $\mathrm{RTT} = \mathrm{RTT}_0 + 2T$, where RTT_0 is the round-trip time without burst assembly (or, equivalently, when $T = 0$). Similarly, the TCP TO value, RTO, is also enlarged in proportion to $\mathrm{RTT}_0 + 2T$.

In addition to the delay penalty, there is another penalty, called the *loss penalty*, which is very intuitive since the larger the burst loss rate within the OBS network, the smaller the TCP throughput.

2) DFL Gain: The enlarged transmission unit from a packet to a burst could delay the first loss in a TD period (TDP) between two TD losses, which in turn enables TCP to reach a larger sending window and send more data in a TDP, which is thereafter called "DFL gain." This is because of the geometric burst loss assumption that all the bursts have the same loss probability p regardless of the number of segments from one TCP flow contained in each burst. This assumption is valid in OBS networks because the real burst size is decided by packets from many aggregated flows in the edge. However, for simplicity, we just denote the virtual burst size l seen by one TCP flow as the number of segments from the same TCP flow contained in the burst. Therefore, the average number of successfully transmitted bursts before a lost burst should be the same in OBS networks as the average number of successfully transmitted packets before a lost packet in packet-switched networks. Thus, the larger a burst is, the more packets can be successfully transmitted before the first packet loss occurs. Although the DFL could also enlarge the duration of TDP, its effect is always less than the increased data sent, and thus the DFL gain is always positive. It has been

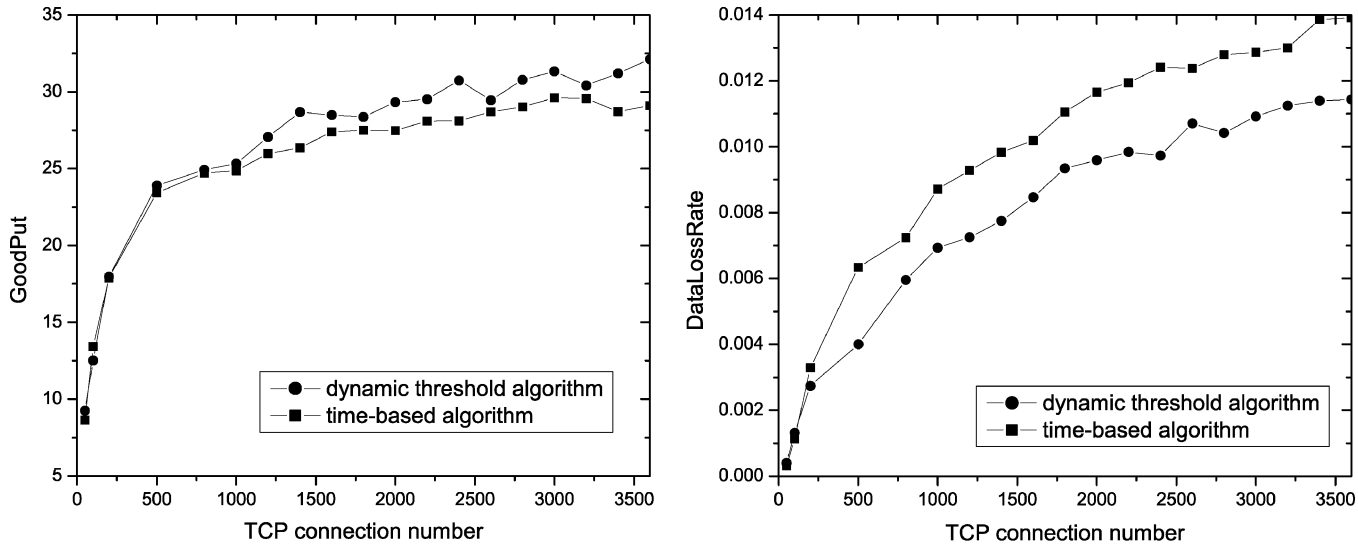


Fig. 7. TCP's performance with different assembly algorithms. (a) Goodput versus TCP number. (b) Loss rate versus TCP number.

proven by analysis and simulation that the DFL gain is proportional to the square root of the burst length l [24].

3) Retransmission Penalty: Different TCP implementations have different retransmission mechanisms, resulting in different retransmission phase. In a retransmission period, the TCP congestion window cannot increase, and TCP may send only a few or no new packets in each round. Since retransmitting a burst will usually require a longer retransmission period than retransmitting a packet, there could be retransmission penalties for a lost burst compared with a lost packet. We will discuss retransmission penalties for three common TCP implementations: Reno, New-Reno, and SACK in the following.

TCP Reno detects a TD loss by triple duplicate ACKs only and treats multiple TD losses separately. After each lost packet is retransmitted, Reno will immediately halve its congestion window. Therefore, multiple packet losses in a burst can halve the congestion window a consecutive number of times, and TCP Reno may finally result in a TO retransmission. For TCP New-Reno, successfully retransmitting one of the lost packets (in the lost burst) will not end the fast retransmission phase. Instead, if the new ACK received after retransmitting a lost packet is still a partial ACK, which does not acknowledge the packet with the highest sequence number, New-Reno will automatically retransmit the next unacknowledged packet, and still stays in the fast recovery phase without halving its congestion window. In this way, New-Reno improves over Reno in packet-switched networks when dealing with multiple TD losses [13]. However, New-Reno insists on retransmitting only one packet in each retransmission round, during which there is no increase in the congestion window; thus, few new packets can be sent. Since the time for retransmitting all the packets in a large burst can be long for New-Reno while Reno can end the long retransmission phase by a TO event, the overall throughput of New-Reno may be lower than that of Reno when the burst size is large.

The problems in Reno and New-Reno with multiple packet losses are similar because both of them assume that a duplicate/partial ACK only implies that the next unacknowledged packet is lost, and therefore the sender only retransmits one lost packet in the next round upon receiving one or more duplicate/partial ACKs. Some recent TCP implementations such as Selective Acknowledgment (SACK) by Floyd [13] and Forward Acknowledgment (FACK) by Mathis [22] are capable of handling multiple losses more efficiently. Specifically, both SACK and FACK add the information of missing segment blocks in the receiver's window to ACKs. As soon as the sender receives the ACKs, it can retransmit multiple lost packets in one round and quickly recover from the retransmission phase. Since the number of retransmission rounds needed is almost the same whether it is a burst loss or a packet loss, SACK/FACK has no retransmission penalty and outperforms Reno and New-Reno in OBS networks.

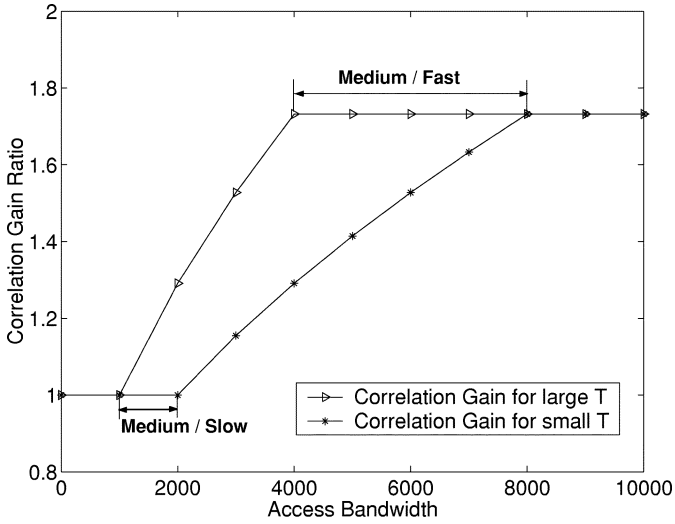


Fig. 8. Correlation gain versus arrival rate (fix T and p).

It is noted that the combined effect from the DFL gain and retransmission penalty was called the "correlation gain" [25], which is always positive as long as the burst size is reasonably large, i.e., $l > 3$.

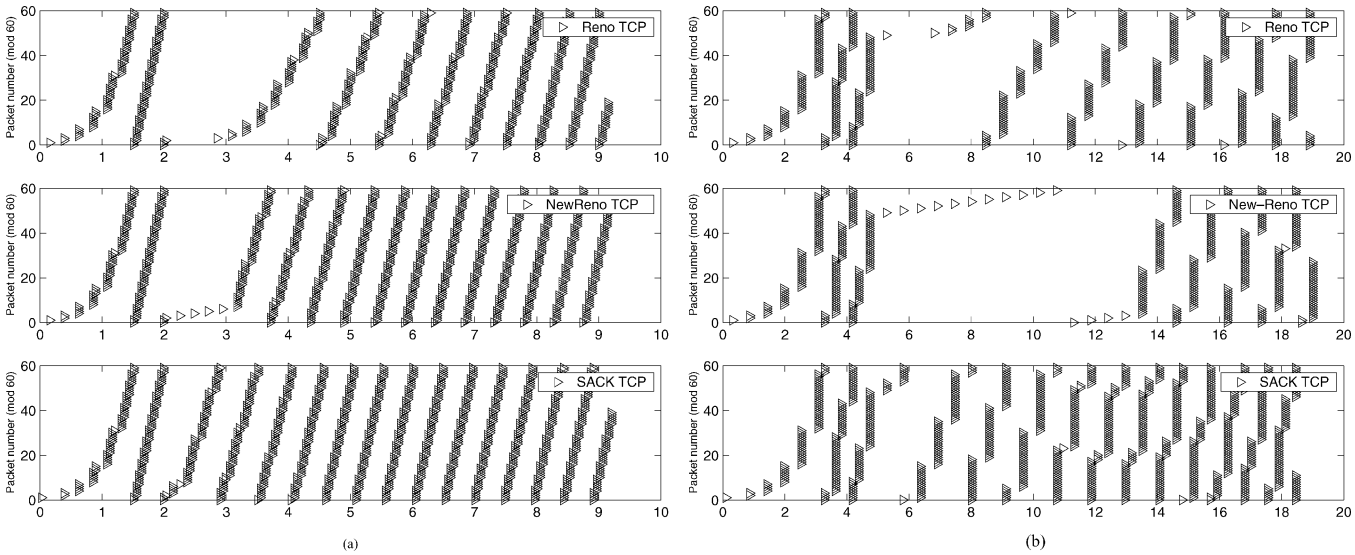


Fig. 9. Packet traces with one burst loss ($W = 40$ when a burst is lost). (a) $T = 0.04$ s. (b) $T = 0.2$ s.

4) Effect of Access Bandwidth: Note that the different access bandwidth in the local IP access networks could also affect TCP throughput. With a relatively slow access bandwidth (and a relatively small assembly time T), only one segment from a given TCP flow may be assembled into a burst. Such a flow is called a slow TCP flow, and the performance of a slow TCP flow in OBS networks is similar to that in packet-switched networks except for the delay penalty, which increases with T. With a large access bandwidth (and a relatively large assembly time T), all TCP segments from one sending window of a TCP flow could always be assembled into one burst even when the congestion window W reaches its maximum. Such a flow is called a fast TCP flow. Although the DFL gain of a fast TCP flow is maximized, the delay penalty still increases with T. Hence, the throughput of both slow and fast TCP flows decreases with the assembly time T. If a TCP flow has an access bandwidth (as well as the assembly time) between a slow TCP flow and a fast TCP flow, it is called a medium-rate TCP flow. The TCP segments from a medium-rate TCP flow contained in the same burst increase with T and so does the DFL gain. Since the delay penalty still increases with T, it will offset the DFL gain, and TCP throughput will have a maximal value for some optimal assembly time.

It is worth noting that whether a flow is slow, medium rate, or fast depends on the assembly time T, the access bandwidth λ_p, and the maximum sending window size W_m. More specifically, for a given λ_p and W_m, a slow or medium-rate TCP flow could be considered as a medium rate when a small T is used or fast TCP flow when a large T is used. Fig. 8 shows the effect of different access bandwidths on the ratio of the correlation gain for TCP Reno as an example [24].

B. Impact of TCP Implementations in OBS Networks

In this section, we present the performance results of Reno, New-Reno, and SACK in OBS networks via simulations using the NS-2 network simulator [26].

Fig. 9 illustrates the packet traces inside bursts departing from an OBS ingress node with a medium-rate TCP flow as an input, whose access bandwidth is 125 packets/s. For this and the following graphs in this section, the X axis represents the bursts' departure time in seconds, and the Y axis shows the packets' number mod 60. Fig. 9 illustrates the packet traces inside bursts departing from an OBS ingress node with a medium-rate TCP flow as an input, whose access bandwidth is 125 packets per second. For this and the following graphs in this section, the x axis represents the bursts' departure time in seconds, and the y axis shows the packets' number mod 60. Fig. 9(a) compares different TCP implementations upon one burst loss when both l and W are small (5 and 40 in the simulation, respectively). It can be seen that SACK has the best performance because the ACK can indicate the block of packets lost, and the sender sends out all the lost packets again upon receiving the ACKs. New-Reno generally performs better than Reno because New-Reno detects the loss of next packets upon receiving a partial ACK, without waiting for three duplicate ACKs or a TO (while Reno will have a TO as discussed previously). In addition, New-Reno's congestion window will only be halved once after it is recovered (i.e., $W = 40/2 = 20$), which enables New-Reno to recover quickly from the loss of a small burst. We note that if l is small, but W at the time when burst loss occurs is relatively large compared with l as in Fig. 9(a), Reno will not have a TO, and hence its performance will be comparable to that of New-Reno.

Fig. 9(b) shows that when the lost burst's length is large ($l = 15$ in simulation) while W is still relatively small compared with l when the burst loss occurs (e.g., 40), New-Reno performs worse than Reno. This is because New-Reno only retransmits one lost packet in one round and hence needs a long time $15 \times$ RTT to finish all 15 retransmissions, during which no new packets can be sent. Reno, on the other hand, will have a TO as before, but new packets may be transmitted before the TO, and in addition, the RTO value is much smaller than the duration of New-Reno's fast retransmission phase, and Reno's transmission after TO is much more efficient because it exponentially increases the sending window size. Also from Fig. 9(b), we can

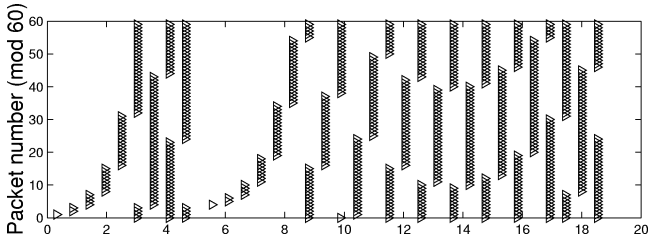


Fig. 10. Packet trace with one burst dropping for a fast TCP flow with $W_m = 40$.

TABLE I
THROUGHPUT RATIO OF NEW-RENO AND SACK OVER RENO, $T = 0.04$ s

$log(p)$	-3	-2.5	-2	-1.5	-1	-0.5
$NewReno/Reno$	1.0	1.1	1.29	1.19	1.03	0.93
$SACK/Reno$	1.0	1.07	1.58	1.4	1.33	1.07

TABLE II
THROUGHPUT RATIO OF NEW-RENO AND SACK OVER RENO, $T = 0.2$ s

$log(p)$	-3	-2.5	-2	-1.5	-1	-0.5
$NewReno/Reno$	1.0	0.94	0.9	0.76	0.93	1.06
$SACK/Reno$	1.0	1.03	1.02	1.1	1.0	1.0

see that in this case, SACK has a much better performance than Reno and New-Reno due to its selective acknowledgment.

To simulate a fast TCP flow, we assume that the access bandwidth is 1000 packets per second, and the burst assembly time is $T = 0.2$ s. In this case, since there is only one burst (containing all the packets) in one round, a burst loss will trigger a TO event in Reno, New-Reno, and SACK, which have exactly the same performance as they all use slow start for retransmission after a TO loss, as shown in Fig. 10.

Tables I and II show the throughput ratios of New-Reno over Reno as well as SACK over Reno, as the burst loss rate varies from low to high. In general, for a low or high loss rate, all three TCP implementations have a similar performance. This is because the performance difference for different TCP implementations come from the TD retransmission stage only. More specifically, since a high burst loss rate usually leads to a higher TO probability with which there is no difference in TO retransmissions in the three TCP implementations. On the other hand, a low burst loss rate leads not only to a low TO probability but also to a low TD loss probability. Accordingly, there will also be little performance difference between different TCP implementations. However, with a medium-low to medium-high loss rate, the probability of TD losses can be relatively high (compared with that of TO losses), and accordingly, the performance difference among different TCP implementations will be more significant, as illustrated in the middle two columns of Tables I and II.

Tables I and II also show the effect of the burst assembly time T (or burst length) on the performance ratios of New-Reno over Reno as well as SACK over Reno, respectively. It is noted that SACK always has the best performance, and New-Reno has a better performance for a smaller assembly time (0.04 s for possibly smaller burst size, e.g., up to 5), while Reno has a better performance for a larger assembly time (0.2 s for possibly smaller bust size, e.g., up to 25), as illustrated in Tables I and II, respectively. Also note that when assembly time T (or burst length) is large, SACK is still better than Reno, but its advantage diminishes due to the fact that TO can happen more frequently, in which case all TCP implementations use the same retransmission mechanisms.

V. BURST RESERVATION PROTOCOLS

In this section, we will give an overview of burst transmission protocols and also introduce the existing protocols for OBS networks.

In [8], the author evaluated two burst-level admission control mechanisms for asynchronous transfer mode (ATM) networks: tell-and-wait (TAW) and TAG. In TAW, when the source has a burst to transfer, it first tries to reserve the bandwidth/wavelength from source to destination by sending a short "request" message. Every intermediate node receiving this message will make reservation on a certain output link. If the requested bandwidth is successfully reserved on all the links along the path, an ACK will be sent backward to inform the source to send out the burst immediately; otherwise, a negative acknowledgment (NAK) will be returned to release previous bandwidth reservations and initiate the retransmission of the "request" message after a back-off time. In TAG, the source transmits bursts without making any bandwidth reservation in advance. At an intermediate node, the burst needs to be delayed before the switch control unit makes reservation on outgoing links. If the reservation fails at any intermediate node, an NAK will be sent back to the source to initiate the burst retransmission after a back-off time.

Various performance (e.g., throughput and delay) comparisons between these two conceptual approaches were given in [8]. It has been found that TAW outperforms the TAG when the propagation delay is negligible with respect to the burst length (in terms of transmission time). The opposite outcome holds when the propagation delay is significant compared with the burst length.

The concept of TAG forms the basis of terabit burst switching (TBS) [3]. With TBS, in order to compensate for the CP processing time and prevent a burst from entering the switching fabric before its configuration is finished, a fixed delay is inserted into the data path using an FDL at each input port. On the other hand, just-in-time (JIT), which was first proposed in [27], can be considered as a variant of TAW as it requires each burst transmission request to be sent to a central scheduler. The scheduler then informs each requesting node the exact time to transmit the data burst. Here, JIT means by the time a burst arrives at an intermediate node, the switching fabric has been configured. This concept is later applied and extended to wavelength-routed OBS (WR-OBS) network [5]. Since centralized protocols are neither scalable nor robust, [28] provided a distributed version of JIT protocol called reservation with JIT (RIT), which requires a copy of the request to be sent to all switches (each has a scheduler) concurrently. These schedulers are not only synchronized in time, but also share the same global link state information, which makes the implementation difficult. The authors of [29] proposed another distributed version of JIT protocol based on hop-by-hop reservation which adopts some features of the just-enough-time (JET) protocol [1], [2].

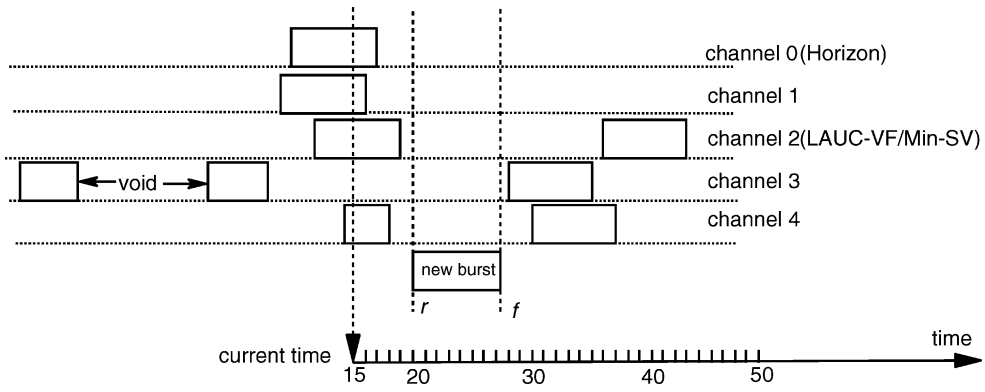


Fig. 11. JET protocol.

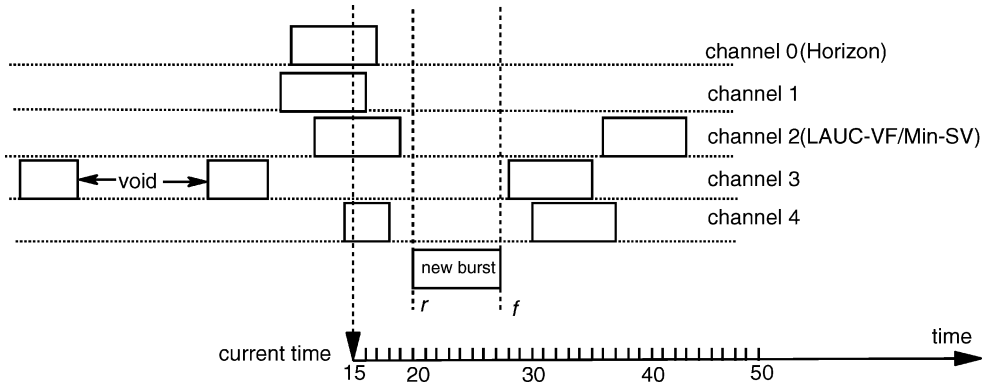


Fig. 12. New burst is scheduled to different channels by different algorithms.

JET is the prevailing distributed protocol for OBS networks today that does not require any kind of optical buffering or delay at each intermediate node [1], [2]. It accomplishes this by letting each CP carry the offset time information and make the so-called delayed reservation (DR) for the corresponding burst, i.e., the reservation starts at the expected arrival time of the corresponding burst. In the example shown in Fig. 11, the bandwidth is reserved for the first burst starting from the burst arrival time instead of the arrival time of CP. At each intermediate node, the offset time is updated (reduced) to compensate for the actual header processing/switch configuration time (see Fig. 2). Note that the delay experienced by a CP might vary due to different reasons. In addition, when we consider deflection routing [30] in an OBS network, the minimal offset time for the primary path might not be enough if the burst takes a longer alternate path. In such a case, an extra offset time can be added [2].

Another important feature of JET is that the burst-length information is also carried by the CP, which enables it to make close-ended reservation (i.e., only for the burst duration with automatic release) instead of open-ended reservation (i.e., which will not be terminated until a release signal is detected). This close-ended reservation will help the intermediate node make intelligent decisions on whether it is possible to make reservation for a new burst, and thus effective bandwidth utilization can be achieved. An example is shown in Fig. 11, where the reservation for the second burst arrival in cases 1 and 2 can succeed if and only if at the time when the second CP arrives, the intermediate node makes close-ended reservations for both the first and second bursts.

VI. Scheduling Algorithms

In an OBS network, it is possible that a control packet may arrive o units of time (which is called the offset time) before the corresponding burst b arrives. In such a case, the reservation for the burst will not start at the current time (t), but at $r = o + t$ (i.e., when the burst actually arrives). If the burst's length is l (time units), the reservation will be made until $f = r + l$. Because bursts may have different offset times at the core nodes and may not arrive in the same order as their corresponding CP, each channel[2] is likely to be fragmented with several reservation periods, separated by idle (also called void) intervals (see channels 2–4 in Fig. 12, for example). More specifically, each of the k channels initially corresponds to an open interval (considered to be a special case of a void interval) from time 0 to positive infinite. Let each void interval I_j be modeled as an ordered pair (s_j, e_j), where s_j and e_j are the starting and ending time of the void interval I_j, respectively, with $e_j > s_j$. We say a void interval I_j is feasible to a data burst $b = (r, f)$, if and only if $s_j \leq r$, and $e_j \geq f$. Once the reservation is made using a feasible interval I_j, up to two new void intervals may be created, which are (s_j, r) and (f, e_j), respectively.

The availability of FDLs at each node further complicates the design of scheduling algorithms. More specifically, assume that there are F different delays ($d_1 < d_2 < \cdots < d_F$) that a burst can obtain via FDLs at a node. Then, the possible offset time values are $o, o^1, o^2, \ldots, o^F$, where $o^j = o + d_j$ for

[2]The terms *channel* and *wavelength* are used interchangeably in this paper.

$1 \leq j \leq F$. This in turn leads to $F+1$ different starting times $r, r^1, \ldots, r^j, \ldots, r^F$ (where $r^j = r + d_j$) and finishing times $f, f^1, \ldots, f^j, \ldots, f^F$ (where $f^j = r^j + l$) of the reservation period for the burst. A scheduling algorithm thus needs to examine up to $F+1$ possible ways to satisfy a reservation request for each burst at one node.

In addition to being efficient in terms of bandwidth utilization and loss rate, a scheduling algorithm needs to be fast. For example, if the minimum burst length is 1 unit time (e.g., a millisecond), and the OBS switching fabric has 64 input fibers, each multiplexed with 100 channels, the maximal number of control packets (or reservation requests) that may need to be processed is 6.4 million/s. Hence, a feasible scheduling algorithm must have comparable processing speed.

The scheduling algorithms can be roughly divided into two categories: those that do not consider the impact on downstream bursts brought by scheduling the current burst, which we call *reactive scheduling algorithms*, and those that try to avoid burst contention in the future, which we call *proactive scheduling algorithms*.

A. Reactive Scheduling Algorithms

We first describe several typical reactive scheduling algorithms, which have been studied in the literature.

Horizon: The Horizon scheduling algorithm was proposed in [3]. In this algorithm, a scheduler only keeps track of the so-called horizon for each channel, which is the time after which no reservation has been made on that channel. The scheduler assigns each arriving data burst to the channel with the latest horizon as long as it is still earlier than the arrival time of the data burst (this is to minimize the void interval to be created between the current horizon and the starting time of the new reservation period). In Fig. 12, the horizons of channels 0 through 4 are 17, 16, 43, 35, and 37, respectively, and the new burst is $b = (20,27)$. As a result, only channels 0 and 1 contain a feasible interval, and in the end, the new burst will be scheduled onto channel 0.

For a link with k channels, the best implementation of the Horizon scheduling algorithm takes $O(\log k)$ time to schedule a burst. Accordingly, the Horizon algorithm is relatively simple and has a reasonably good performance in terms of its execution time. However, the Horizon scheduling algorithm results in a low bandwidth utilization and a high loss rate. This is due to the fact that the horizon algorithm wastes all the void intervals.

LAUC-VF: A channel scheduling algorithm, called latest available unused channel with void filling (LAUC-VF), was proposed in [4]. LAUC-VF keeps track of all void intervals (including the interval between the horizon and positive infinity) and assigns to a burst arriving at time r a large enough void interval whose starting time s_i is the latest but still earlier than r. In Fig. 12, each of the five channels contains a feasible interval, but among them, the one on channel 2 has the latest starting time (of about 19). As a result, the new burst will be scheduled on channel 2.

LAUC-VF yields a better bandwidth utilization and loss rate than the Horizon algorithm. However, even the best known implementation of LAUC-VF has a much longer execution time than Horizon, especially when the number of voids m is significantly larger than k (which in general is the case). For example, a straightforward implementation of the LAUC-VF algorithm described in [4] takes $O(m)$ time to schedule a burst, where m is the number of voids. The time complexity becomes $O(Fm)$ when there are F different delays a burst can obtain via the use of FDLs. Searching for a suitable void interval in this way might take a longer time than that allowed by the offset time of a burst, thus resulting in unnecessary burst dropping.

Min-SV: [31] proposed several efficient algorithms for selecting channels for incoming data bursts using different criteria. A representative algorithm proposed in [31] is minimum starting void (Min-SV). Min-SV 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 doing so, Min-SV enjoys a loss rate as low as LAUC-VF and a processing time almost the same as Horizon.

B. Proactive Scheduling Algorithms

A proactive scheduling algorithm called priority-based wavelength assignment (PWA) was proposed in [32]. In PWA, each ingress node keeps a wavelength priority database for every destination node. When a burst arrives, the ingress node searches the wavelength priority database. If the wavelength with the highest priority is available, the burst is sent out on that wavelength; otherwise, the algorithm checks the wavelength with the second highest priority. The priority of each wavelength is updated dynamically according to its burst loss profile (i.e., how many percents of the bursts from the ingress node to the egress node have been lost on channel λ). Simulation shows PWA can reduce loss rate in an OBS network. However, PWA is only meaningful in an OBS network without wavelength conversion capability.

Li and Qiao [33] took this a step further by proposing several proactive scheduling algorithms that can work with or without wavelength conversion. The algorithms proposed in [33] are collectively called BORA. The idea of BORA is based on the observation that if the total number of simultaneously arriving bursts at an output port exceeds the number of channels at that port, burst loss will be inevitable (assuming that there are no FDLs). Thus, if we can reduce the total number of simultaneously arriving bursts at each port, it is likely that the burst loss will be reduced.

To facilitate discussion, we use *overlapping degree* to describe the number of bursts that arrive at one link simultaneously. Clearly, the *overlapping degree* is directly related to the burst loss. The larger the *overlapping degree* is, the more likely an incoming burst will be dropped.

Fig. 13 shows an example where LAUC-VF (or any other reactive scheduling algorithms) fails to schedule all bursts successfully. In this example, the intermediate OBS node has two incoming links and one outgoing link, and each link has two data channels and one control channel. Assume that four data bursts b_1, b_2, b_3, and b_4 arrive from the incoming links and they all want to go to the same outgoing link, since they overlap with each other from time t_1 to time t_2 on the outgoing link (and, accordingly, the overlapping degree during the period (t_1, t_2) is 4), two out of the four bursts will be dropped if the OBS node does not have FDLs or all FDLs have already been used by other bursts.

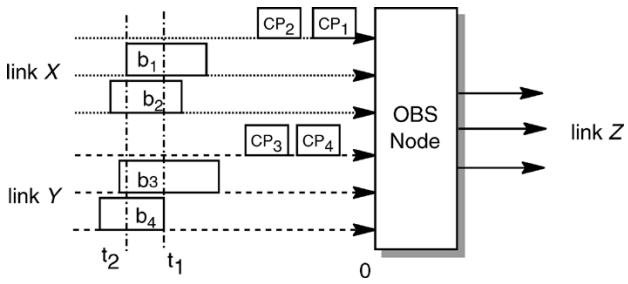


Fig. 13. Inevitable contention among the four bursts arrived simultaneously.

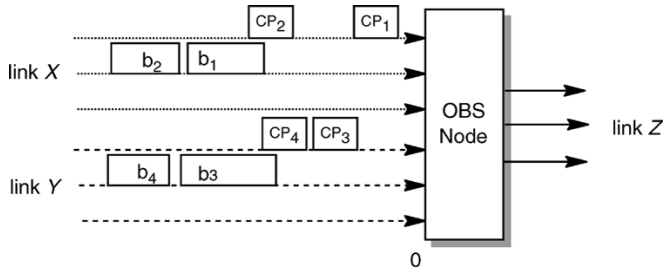


Fig. 14. Avoid contention by reducing burst overlap.

Note that if we can reduce such an overlap by delaying burst b_2 at an upstream node so that it arrives at link X after b_1 and similarly delaying b_4 so it arrives after b_3, the OBS node will not drop any burst as shown in Fig. 14.

For an OBS network, without loss of generality, we assume that there are more than one OBS path [34] passing a given link and each OBS path has one ingress node and one egress node. In order to reduce burst overlap on the given link, each OBS path should try to reduce the burst overlap on itself. In an OBS network without FDLs, the core nodes cannot delay bursts to reduce the burst overlap, but ingress nodes with an electronic buffer can by using the electronic buffer judicially.

In the example shown in Fig. 13, if b_1 and b_2 are generated by one ingress node, b_3 and b_4 are generated by another ingress node. A straightforward way to reduce burst overlap is to *delay* the transmission of burst b_2 until burst b_1 is sent out. Similarly, when a new burst is assembled and ready to be sent at the same ingress node, it will be scheduled to the same channel as b_1 and b_2. One limitation of this simple algorithm is that the delay time introduced by this algorithm could be too large. To limit the extra delaying time, we modify the algorithm to make it work with a bounded delay for each burst, say α time units.

Fig. 15 illustrates an example of how an ingress node schedules a set of locally generated bursts. Fig. 15(a) shows that four bursts b_5, b_6, b_7, and b_8 arriving at the scheduler sequentially. Fig. 15(b) shows the scheduling result when the ingress node ignores the delay time of each burst. In Fig. 15(b), the delay time of burst b_7 and b_8 exceeds maximum delay time α. Fig. 15(c) shows the scheduling result when the ingress node takes the maximum delay time into consideration. More specifically, after bursts b_5 and b_6 have been assigned to λ_0, the ingress node tries to schedule burst b_7 to λ_0, but since the delay time of burst b_7 will be larger than the maximum allowed delay time α, the ingress node assigns b_7 to λ_1. Similarly, when the ingress node schedules b_8, it checks λ_0 first. Again, because the delay time would exceed α, the ingress node tries λ_1 and reserves it for b_8.

Based on this discussion, we introduce several proactive scheduling algorithms to reduce overlapping degree, which differ mainly in how a channel is selected [33].

1) Fixed-Order Search: In the previous description, the ingress node always searches channels using a fixed order and the algorithm stops either when a suitable channel is found that satisfies the maximum delay requirement or all channels have been checked and none of them can satisfy the requirement.

Such a fixed-order search algorithm can work with or without utilizing voids that may exist on each outgoing channel of each

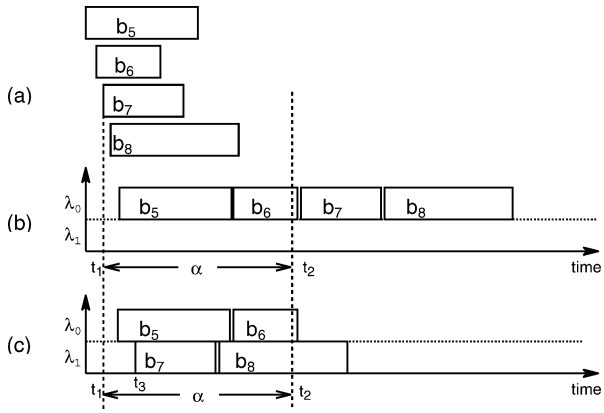


Fig. 15. Serialized bursts at an ingress node (fixed-order search).

ingress node. More specifically, if the ingress node schedules the locally generated (assembled) bursts using voids of each channel, we call it BORA with fixed-order searching and void filling (BORA-FS-VF). If the algorithm schedules the locally generated bursts to the open interval (also called horizon) of each channel only (without void filling), we simply refer to it as BORA-FS. The time complexity of BORA-FS-VF is $O(m)$ and that of BORA-FS is $O(\log k)$ [33].

2) Destination-Based Search: According to the previous discussion, we can see that the fixed-order channel searching reduces the overlapping degree associated with the locally generated bursts on the outgoing links that are connected to an ingress node. The idea behind this fixed-order search is that by reducing the overlapping degree on the first hop (link) of each OBS path, the overlapping degree on each intermediate link may also be reduced (even though bursts may belong to different OBS paths).

In the following, we modify the fixed-order channel searching algorithms into destination-based channel searching algorithms that can reduce loss rate further by taking the destination information of different OBS paths into consideration while scheduling the locally generated bursts.

Consider the example shown in Fig. 15 again. When we use BORA-FS (or BORA-FS-VF), bursts b_5 and b_6 are scheduled onto the same channel, and bursts b_7 and b_8 are assigned to another channel. In this example, if bursts b_5 and b_7 take OBS path L_0, bursts b_6 and b_8 take OBS path L_1, and then for both L_0 and L_1, the overlapping degree is 2, even on the links that L_0 and L_1 do not share.

If we modify the above algorithm as follows, the overlapping degree can be reduced. When a burst arrives, the scheduler will

first search the channel(s) preferred by the OBS path the burst will follow (called *home channel(s)*). Only if the burst cannot be accommodated by its home channel(s) then other channels (nonhome channels) will be considered. A natural way is to check nonhome channels sequentially and assign the burst to the first channel that can satisfy the burst. We call such an algorithm BORA with destination-based search (BORA-DS) if it schedules the locally generated bursts beyond the horizon of each channel. If the algorithm schedules the locally generated bursts using the voids of each channel (instead of only horizons), we name it as BORA-DS with void filling or BORA-DS-VF. The time complexity of BORA-DS is $O(\log k)$, while that of BORA-DS-VF is $O(m)$ [33]. The problem of how to the determine home channel(s) for each OBS path is, for the most part, an orthogonal issue and needs to be addressed separately.

Fig. 16 shows the scheduling result assuming the home channel for L_0 is λ_0 and that for L_1 is λ_1. The main difference between Figs. 15 and 16 is due to the fact that when the OBS node schedules b_6, it starts searching from λ_1 (its home channel) instead of λ_0 and schedules the burst on λ_1. Subsequently, it will schedule b_7 on λ_0 and b_8 on λ_1. Such a scheduling does not reduce the loss rate on the link connected to the ingress node but reduces the overlapping degree among the four bursts to one on the downstream links that are not shared by L_0 and L_1 and thus can reduce loss rate.

3) Traffic Engineering Based Search: One may improve BORA-DS algorithms by taking traffic-engineering- and routing-related issues into consideration. Note that the example shown in Fig. 16 has only two channels and two OBS paths. In a real OBS network, it is likely that each network has dozens of nodes, and each link has hundreds of channels. To reduce the overlapping degree (or loss rate) on each link, we need to reduce the overlap among the bursts belonging to not only the same OBS path, but also all the OBS paths sharing the link. The latter objective can be achieved by a load-balancing OBS path routing algorithm as a part of traffic engineering effort (such as the work in [35] and [36]). In the following discussion, we describe how to select nonhome channels (in case a burst cannot be scheduled to any of the home channels assigned to its corresponding path) such that "interference" to other OBS paths is minimized. We assume that shortest path routing algorithm is applied, but the ideas to be described can be extended to other routing algorithms.

Each ingress node in an OBS network has a spanning tree to maintain the shortest path to every egress node as shown in Fig. 17. For node A, the burst sent out from it will only pass the links of this spanning tree. Therefore, the scheduling problem becomes how to minimize the burst overlapping degree on each edge of the spanning tree.

If the traffic on each OBS path is steady (e.g., with a constant rate), then with an appropriate home channel assignment, the scheduling algorithm only needs to use the home channel(s). If the traffic on each OBS path fluctuates, it is possible that the home channels of a given OBS path are overloaded and other channels are underloaded. To avoid dropping data bursts unnecessarily and to utilize bandwidth efficiently, the ingress node can schedule a burst to some nonhome channel (which can be another OBS path's home channel). However, this may disturb other OBS paths. To minimize such disturbance, we need to search nonhome channels carefully.

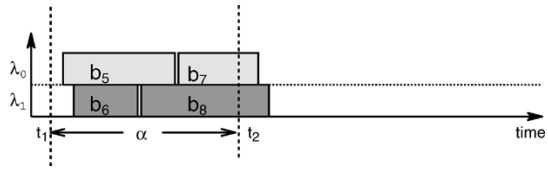


Fig. 16. Serialized bursts using destination-based searching order.

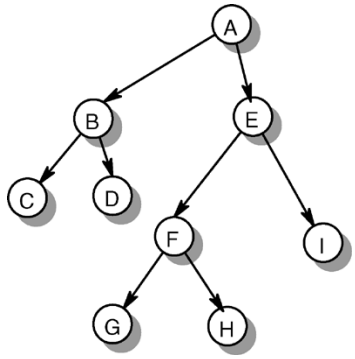


Fig. 17. Spanning tree rooted at node A.

Consider again the spanning tree rooted at node A in Fig. 17. To simplify the presentation, the home channel assigned to the OBS path from A to a node (e.g., G) will be simply referred to as G's home channel. When a burst destined to G arrives at node A, node A will assign it to G's home channel(s) if possible; otherwise, node A may try node F's (or node H's) home channels since nodes A, F, and H share two common links $\langle A, E\rangle$ and $\langle E, F\rangle$. The motivation is that by scheduling the burst in this way, it can prevent another burst destined to F (or H) from being scheduled at the same time, thus keeping the overlapping degree on links $\langle A, E\rangle$ and $\langle E, F\rangle$ unchanged.

This algorithm can be formally described as follows. Given a burst destined to node J and a spanning tree root at the ingress node, the ingress node first reserves the home channel(s) of J, and if successful, the algorithm stops; otherwise, the ingress node checks the home channels that belong to the children nodes of J and, if successful, the algorithm stops; otherwise, the ingress node checks the home channels of J's parent node, J's sibling nodes, and then the home channels of all other nodes (without having to follow any specific order). This process continues until one channel is found or until all channels have been checked and the burst has to be dropped.

Similar to the definition of BORA-FS and BORA-DS, if this algorithm assigns locally generated bursts to the horizon of each channel, it is called BORA traffic engineering (BORA-TE). If the algorithm utilizes all voids, we called it BORA-TE with void filling (BORA-TE-VF). The time complexity of BORA-TE is $O(h + \log k)$ [33], where h is the height of the spanning tree. The time complexity of BORA-TE-VF is $O(m)$.

Fig. 18 shows the simulation results from using the BORA algorithms to schedule locally generated bursts and their comparisons with the existing algorithm LAUC-VF. In all simulations, we assume that LAUC-VF is used to schedule transit

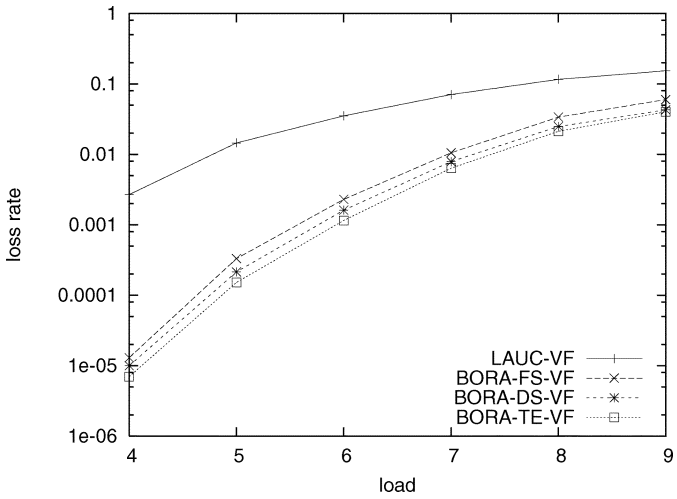


Fig. 18. Performance enhancement achieved by proactive scheduling algorithms.

(bypass) bursts as opposed to locally generally bursts. The network topology used in this simulation is the 14-node NSFNET. The α value used in this simulation is 1.0 ms. From Fig. 18, we can see that the loss rate of BORA-FS-VF is much lower than that of LAUC-VF, and the loss rate of BORA-DS-VF is about 40% lower than that of BORA-FS-VF. Furthermore, the loss rate of BORA-TE-VF is lower than that of BORA-DS-VF under all traffic loads (note that the Y axis is in log scale, so the difference between two scenarios appears small). Compared with LAUC-VF, the significant improvement achieved by BORA-FS-VF is due to the fact that BORA-FS-VF minimized the overlapping degree of locally generated bursts, whether these bursts share a path or not. The further improvement achieved by BORA-DS-VF and BORA-TE-VF are due to the fact these algorithms reduced the overlapping degree of each path wisely. The improvement achieved by BORA algorithms is especially significant when the traffic load is low to medium. When the traffic load is high, the network (or some part of the network) can be easily overloaded; hence, the effect of BORA algorithms is mitigated.

VII. Performance Analysis in OBS Networks

A. Loss Model at a Core Node

Analyzing the performance of OBS networks requires different techniques from those developed for electronic-switched networks. This is due to not only the unique burst traffic characteristics, but also the bufferless nature of optical switches. Although optical FDLs may be used, they can only provide limited and deterministic delay.

Performance evaluations of OBS networks carried out in current literature either use simulations with burst traffic generated directly from some assembly processes, which limits the potential for further theoretical analysis, or simply adopt the M/M/K/K model [37], [38] assuming that the input traffic to an OBS core node is Poisson, which is not accurate. The authors of [39] and [40] studied loss performance throughout OBS networks, with several OBS core nodes connecting source and destination. However, the input burst traffic was assumed to have an exponential burst size distribution and bounded interarrival time, which is still quite different from the traffic output of an burst assembly unit. The work in [41] modeled the traffic arrival process using a Markov process; however, the results are only applicable to OBS edge nodes, where throughput and delay are the focus of the study. Since the maximal burst size is typically small compared with the link transmission capacity, the JIT protocol with two-way reservation (i.e., with acknowledgments that the bandwidth has been reserved before a burst can be sent out) used in [41] is much less efficient than a hop-by-hop one-way JIT (i.e., without acknowledgment) and JET protocols, which only delay bursts at the edge for a small fixed offset time and allow some burst losses at OBS core nodes.

Hereafter we describe a burst loss model proposed in [42] to evaluate the performance of an OBS core node, which is developed according to the burst traffic characteristics obtained in Section III.

The bufferless nature of an OBS network means that the output traffic process will only be affected by the contention-induced loss instead of any buffer overflow. Although in electronic packet switched networks, the impact of long range dependency on the burst loss rate or delay is more obvious especially when the buffer size is large, it is not obvious in an OBS network because the loss from burst contentions in OBS networks is characterized by the traffic statistics on smaller time scales. Accordingly, given the assembled burst traffic characteristics as discussed in Section III, the following model can be used to estimate the loss rate at an intermediate OBS core node with many assembled burst traffic inputs:

$$P_{\text{loss}} \approx Ce^{-\delta'+\delta'(n-1)^{\alpha(n)}} \tag{2}$$

where C is the asymptotic constant denoting short-range burstiness and is given by the Erlang's loss formula (M/M/K/K) with k wavelengths and total load ρ as follows:

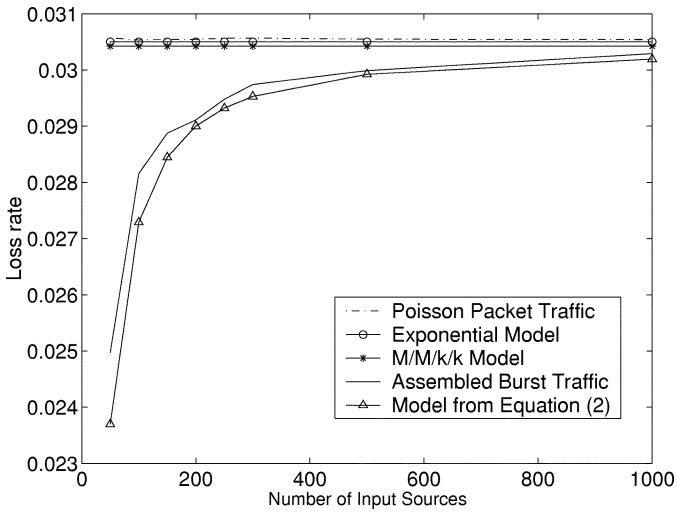
$$C = \frac{\frac{1}{k!}(\rho k)^k}{\sum_{i=0}^{k} \frac{1}{i!}(\rho k)^i}. \tag{3}$$

In addition, δ' in (2) is the decay rate related to both the short-range burstiness and the assembled burst traffic characteristics, ϕ is a small adjustable constant based on the assembled burst traffic characteristics, n is the total number of assembled burst flows at input, and $\alpha(n)$ is a log function obtained from experiments (simulations), as follows:

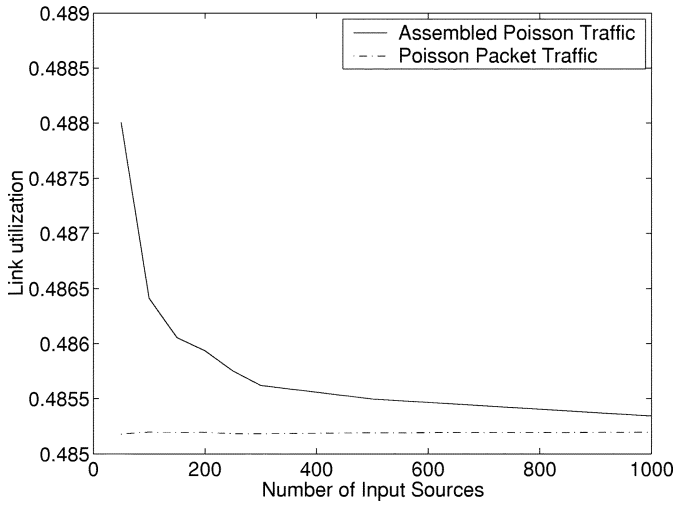
$$\alpha(n) = \frac{\phi}{\log(n+\sigma)} \tag{4}$$

where σ is also a small adjustable constant based on assembled burst traffic characteristics.

As shown in the Fig. 19(a), this model is more general and accurate than the M/M/K/K model. In particular, the latter is a special (limiting) case since $\lim_{n\to\infty} \alpha(n) \to 0$ and $P_{\text{loss}} \to Ce^{-\delta'+\delta'} = C$ as given by the M/M/K/K model, which is due to the fact that the aggregation of a large number of independent traffic flows becomes a Poisson process and the M/M/K/K model is valid in such a case. Thus, the M/M/K/K model is



(a)



(b)

Fig. 19. Loss performance of an OBS core node with a fixed load of 50%. (a) Loss rate versus number of input sources. (b) Link utilization versus number of input sources.

only a special instance of the loss model given in (2) in OBS networks.

Fig. 19(a) and (b) shows the loss rate and bandwidth utilization at an OBS core node obtained from both simulation and analysis. The results shown were obtained when the total amount of input traffic to the OBS core node is fixed at 50% (even though the number of input sources varies), whether the traffic is the unassembled packet traffic (as in a packet-switched network) or the assembled burst traffic (as in an OBS network). In particular, our analysis has shown that the loss rate increases with the number of assembled burst traffic flows as the input sources, which is different from the loss performance of Poisson packet traffic. It can be seen that the loss model from (2) (with $C = 0.0305$, $\delta' = 0.582$, $\sigma = 2$, and $\phi = 1.2$) is fairly accurate while neither the M/M/K/K nor the exponential model is when the number of input burst flows is small. The latter two models can be used only when the number of input burst flows is large, as discussed previously. In general, having the assembled burst traffic at the input results in a better performance (lower loss and higher bandwidth utilization) than having the same amount of packet traffic due to the fact that the assembled burst traffic has been smoothed in the short range as a result of the burst assembly process as discussed in Section III-A.

B. Applicability to Different Bandwidth Reservation Protocols

Note that the burst loss model provided previously for a node is applicable as long as the traffic characteristics do not change from one core node to another. In this subsection, we discuss how different reservation protocols can affect the traffic characteristics, and in turn the loss rate, in an attempt to facilitate the study of burst loss performance along a path.

As mentioned in Section V, using a non-JET-based reservation protocol, a core node makes immediate reservation for a burst after receiving the corresponding control packet (or setup message). Let t_r be the time that the bandwidth is reserved and t_b the time the burst arrives $t_b > t_r$. If the burst size is l (time units), such a non-JET reservation enlarges the effective burst size to $l + t_b - t_r$. Suppose that there are N hops from an ingress to its egress and that pipelined (or concurrent) operations involving the switch configuration at one node and control packet processing at a downstream node (as described in [2]) is allowed, the the offset time between the departure of a setup message and the corresponding burst at the assembly node is at least

$$\triangle t = N t_{\text{proc}} + T_{\text{OXC}} \tag{5}$$

where t_{proc} is the processing time of the setup message at each core node, and T_{OXC} is the switch configuration time after processing a setup message. Hence, on the mth ($1 \le m \le N$) hop, we have

$$t_b - t_r = \triangle t - m t_{\text{proc}} - T_{\text{OXC}} = (N - m) t_{\text{proc}}. \tag{6}$$

Accordingly, the equivalent burst length is $l + (N - m)t_{\text{proc}}$, which effectively increases the load ρ to $\rho \times ((l + (N - m)t_{\text{proc}})/l)$, and the asymptotic constant C in our loss model in (3) becomes

$$C = \frac{\frac{1}{k!}\left(\frac{l+(N-m)t_{\text{proc}}}{l}\rho k\right)^k}{\sum_{i=0}^{k}\frac{1}{i!}\left(\frac{l+(N-m)t_{\text{proc}}}{l}\rho i\right)^i}. \tag{7}$$

We can see that the C in (7) is larger than the C in (3) as only the numerator in (7) contains a factor $(l + (N - m)t_{\text{proc}})/l > 1$ after some simplifications. As a result, the loss rate will also increase. We can also see that the closer a core node is to the assembly node, the higher the loss rate at the node as the equivalent burst size or effective load is larger. At an OBS core node with many burst flows from different assembly nodes, if flow f has passed m_f hops before reaching the core node and has a load of ρ_f at the ingress node, the total effective load at the node is $\sum_f \rho_f \times ((l + (N - m_f)t_{\text{proc}})/l)$, which is larger than $\rho = \sum_f \rho_f$.

The JET family of bandwidth reservation protocols on the other hand make delayed reservations at a core node starting at the time the corresponding burst arrives at the core node. Therefore, the effective burst size or load is not changed. Although the offset time between a control packet and the corresponding

burst will delay a burst's arrival time, this offset time does not change the burst interarrival time of a burst traffic flow (as all the bursts belonging to the flow have the same offset time). Therefore, the traffic characteristics of a given burst flow is the same with or without using an offset time. In addition, although different burst traffic flows may use different offset times, the statistical characteristics of the aggregated traffic do not change because the statistical characteristics of each individual flow are not affected by the offset times used by other flows (or by itself). Therefore, our loss model in (2) can apply well at any core node in OBS networks with JET protocols. With low and random burst losses in OBS networks, the burst traffic characteristics of a flow will not change much from one hop to another, and hence our loss model may also be applied to obtain the end-to-end loss probability, which is also a subject of our future research.

VIII. Conclusion

OBS has emerged as a new paradigm to exploit the benefit of transparent optical switching while achieving efficient statistical multiplexing at the burst level. In this paper, we have studied several key issues in OBS such as burst assembly, burst scheduling, and performance analysis. It has been demonstrated that the burst assembly process only smooths the input packet traffic in the short term but does not change the long-range dependency (if any) in the input traffic. In addition, the burst size using a time-based assembly algorithm or the burst interarrival time using a burst-length-based assembly algorithm will approach a Gaussian distribution.

We have described a family of scheduling algorithms, called BORA, that can reduce burst loss at core nodes. BORA tries to serialize the outgoing bursts using as few wavelengths as possible to proactively avoid burst contentions at downstream nodes and has been shown to achieve a much lower loss rate than LAUC-VF. The work described in this paper represents the first step in traffic engineering in labeled OBS networks. In such a labeled OBS network, issues such as how to assign home channels and determine the appropriate nonhome channels search order for a given LOBS path between a node pair and how to assess the benefits of using possibly different search orders for different node pairs to avoid burst contentions need further investigation.

Our studies have also identified OBS specific factors affecting TCP performance including delay penalty, DFL gain, and retransmission penalty when evaluating the impact of burst assembly and bufferless switching in OBS networks on the TCP performance. Based on the characteristics of the assembled burst traffic, a burst loss model and its applicability to different reservation algorithms have also been presented. Our loss model, which takes into consideration both the assembled traffic characteristics and bufferless switching in the core, has been demonstrated to be more accurate than the M/M/K/K model used in most literature today. Future work along this line includes the design and analysis of dynamic burst assembly algorithms, efficient TCP implementations (or other transport layer protocols) suitable for OBS networks, OBS protocols that support both periodic and nonperiodic reservations, and path (or end-to-end) burst loss analysis.

References

[1] M. Yoo and C. Qiao, "Just-enough-time (JET): A high speed protocol for bursty traffic in optical networks," in *IEEE/LEOS Summer Topical Meetings Dig. for Conf. Technologies Global Information Infrastructure*, 1997, pp. 26–27.

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

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

[4] Y. Xiong, M. Vandenhoute, and H. Cankaya, "Control architecture in optical burst-switched WDM networks," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 1838–1851, Oct. 2000.

[5] M. Düser and P. Bayvel, "Analysis of a dynamically wavelength-routed optical burst switched network architecture," *J. Lightwave Technol.*, vol. 20, pp. 574–585, Apr. 2002.

[6] K. Dolzer, C. Gauger, J. Späth, and S. Bodamer, "Evaluation of reservation mechanisms for optical burst switching," *AEÜ Int. J. Electronics Communications*, vol. 55, no. 2, Oct. 2001.

[7] L. Xu, H. G. Perros, and G. Rouskas, "Techniques for optical packet switching and optical burst switching," *IEEE Commun. Mag.*, vol. 39, pp. 136–142, Jan. 2001.

[8] I. Widjaja, "Performance analysis of burst admission-control protocols," *Proc. Inst. Elect. Eng. Communications*, vol. 142, pp. 7–14, Feb. 1995.

[9] X. Yu, Y. Chen, and C. Qiao, "Study of traffic statistics of assembled burst traffic in optical burst switched networks," in *Proc. Opticomm*, 2002, pp. 149–159.

[10] K. Laevens, "Traffic characteristics inside optical burst switched networks," in *Proc. Opticomm*, 2002, pp. 137–148.

[11] V. Jacobson, "Congestion avoidance and control," in *SIGCOMM Symp. Communications Architectures Protocols*, 1988, pp. 314–329.

[12] ——, "Modified TCP congestion avoidance algorithm," *End2end-Interest Mailing List*, Apr. 1990.

[13] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno and SACK TCP," in *ACM SIGCOMM Comput. Commun. Rev.*, vol. 26, 1996, pp. 5–21.

[14] X. Cao, J. Li, Y. Chen, and C. Qiao, "Assembling TCP/IP packets in optical burst switched networks," in *Proc. IEEE GLOBECOM*, vol. 3, Nov. 2002, pp. 2808–2812.

[15] A. Ge, F. Callegati, and L. Tamil, "On optical burst switching and self-similar traffic," *IEEE Commun. Lett.*, vol. 4, pp. 98–100, Mar. 2000.

[16] S. Oh and M. Kang, "A burst assembly algorithm in optical burst switching networks," in *Proc. Optical Fiber Communication Conf.*, 2002, pp. 771–773.

[17] D. Morató, J. Aracil, L. A. Diez, M. Izal, and E. M. na, "On linear prediction of Internet traffic for packet and burst switching networks," in *Proc. IEEE Int. Conf. Computer Communications Networks*, Oct. 2001, pp. 138–143.

[18] W. Leland, M. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of ethernet traffic (extended version)," *IEEE/ACM Trans. Networking*, vol. 2, pp. 1–15, Feb. 1994.

[19] M. Izal and J. Aracil, "On the influence of self-similarity on optical burst switching traffic," in *Proc. IEEE GLOBECOM*, vol. 3, Nov. 2002, pp. 2308–2312.

[20] B. Mandelbrot and M. Taqqu, "Robust R/S analysis of long run serial correlation," in *42nd Session Int. Statistical Institute*, vol. 2, 1979, pp. 69–99.

[21] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," *IEEE/ACM SIGCOMM*, vol. 8, pp. 133–145, 1998.

[22] M. Mathis and J. Mahdavi, "Forward acknowledgment: Refining TCP congestion control," in *Proc. ACM SIGCOMM*, 1996, pp. 281–291.

[23] V. Vokkarane, K. Haridoss, and J. Jue, "Threshold-based burst assembly policies for QoS support in optical burst-switched networks," in *Proc. Opticomm*, 2002, pp. 125–136.

[24] X. Yu, C. Qiao, Y. Liu, and D. Towsley, "Performance evaluation of TCP implementations in OBS networks,", Tech. Rep. 2003-13, CSE Dept., SUNY, Buffalo, NY, 2003.

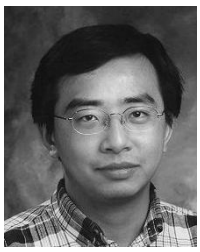
[25] A. Detti and M. Listanti, "Impact of segments aggregation on TCP Reno flows in optical burst switching networks," in *Proc. INFOCOMM*, vol. 3, 2002, pp. 1803–1812.

[26] The Network Simulator—ns-2 [Online]. Available: http://www.isi.edu/nsnam/ns/

[27] D. L. Mills, C. G. Boncelet, J. G. Elias, P. A. Schragger, and A. W. Jackson, "Highball: A High Speed, Reserved-Access, Wide-Area Network," University of Delaware, Electrical Engineering Dept., Newark, Tech. Rep. 90-9-3, 1990.
[28] G. C. Hudek and D. J. Muder, "Signaling analysis for a multi-switch all-optical network," in *Proc. IEEE Int. Conf. Communication (ICC)*, vol. 2, 1995, pp. 1206–1210.
[29] J. Y. Wei and R. I. McFarland, "Just-in-time signaling for WDM optical burst switching networks," *J. Lightwave Technol.*, vol. 18, pp. 2019–2037, Dec. 2000.
[30] C. Hsu, T. Liu, and N. Huang, "Performance analysis of deflection routing in optical burst-switched networks," in *Proc. INFOCOMM*, vol. 1, 2002, pp. 66–73.
[31] J. Xu, C. Qiao, J. Li, and G. Xu, "Efficient channel scheduling algorithms in optical burst switched networks," in *Proc. INFOCOMM*, vol. 3, 2003, pp. 2268–2278.
[32] X. Wang, H. Morikawa, and T. Aoyama, "Priority-based wavelength assignment algorithm for optical burst switched phontonic networks," in *Proc. Optical Fiber Communication Conf.*, 2002, pp. 765–766.
[33] J. Li and C. Qiao, "Schedule burst proactively for optical burst switched networks," *Comput. Netw.*, vol. 44, no. 5, pp. 617–629, 2004.
[34] C. Qiao, "Labeled optical burst switching for IP-over-WDM integration," *IEEE Commun. Mag.*, vol. 38, pp. 104–114, Sept. 2000.
[35] J. Li, G. Mohan, and K. C. Chua, "Load balancing using adaptive alternate routing in IP-over-WDM optical burst switching networks," in *Proc. Opticomm*, 2003.
[36] G. P. V. Thodime, V. M. Vokkarane, and J. P. Jue, "Dynamic congestion-based load balanced routing in optical burst-switched networks," in *Proc. IEEE GLOBECOM*, Dec. 2003, pp. 2628–2632.
[37] M. Yoo, C. Qiao, and S. Dixit, "QoS performance of optical burst switching in IP-over-WDM networks," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 2062–2071, Oct. 2000.
[38] J. Teng and G. Rouskas, "A comparison of the JIT, JET, and Horizon wavelength reservation schemes on a single OBS node," presented at the 1st Int. Workshop Optical Burst Switching, TX, Oct. 2003.
[39] Z. Rosberg, H. Vu, M. Zukerman, and J. White, "Bloking probabilities of optical burst switching networks based on reduced load fixed point approximations," in *Proc. INFOCOMM*, vol. 3, 2003, pp. 2008–2018.
[40] H. Vu and M. Zukerman, "Blocking probability for priority classes in optical burst switching networks," *IEEE Commun. Lett.*, vol. 6, pp. 214–216, 2002.
[41] L. Xu, H. Perros, and G. Rouskas, "A queueing network model of an edge optical burst switching node," in *Proc. INFOCOMM*, vol. 3, 2003, pp. 2019–2029.
[42] X. Yu, Y. Chen, and C. Qiao, "Performance evaluation of optical burst switching with assembled burst traffic input," in *Proc. IEEE GLOBECOM*, vol. 3, Nov. 2002, pp. 2318–2322.

Xiang Yu received the B.S. degree in electronics from Peking University, Beijing, China, in 1999 and the M.E. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2001. She is currently working toward the Ph.D. degree in computer science at the State University of New York at Buffalo.

Her research focuses on traffic characteristics and TCP performance in modern communication networks.



Jikai Li received the B.S. degree from the Computer Science Department of Wuhan University, Wuhan, China, in 1994, the M.E. degree in computer science from the Chinese Academy of Sciences, China, in 1998, and the Ph.D. degree in computer science from the State University of New York at Buffalo in 2004.

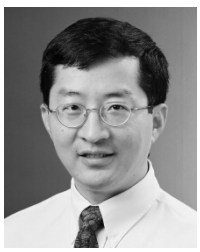
He is currently an Assistant Professor in the Department of Computer Science, the College of New Jersey, Ewing. His research interest includes optical network, optical burst switching networks, scheduling, traffic engineering, and P2P.

Xiaojun Cao received the B.S. degree from Tsinghua University, Beijing, China, in 1996, the M.S. degree from the Chinese Academy of Sciences. China, in 1999, and the Ph.D. degree from the State University of New York at Buffalo in 2004.

He is currently an Assistant Professor with the Department of Information Technology at Rochester Institute of Technology, Rochester, NY. His research interests include modeling, analysis, and protocols/algorithms design of communication networks.

Yang Chen received the B.Eng. degree in information engineering from Xi'an Jiaotong University, Xi'an, China, in 1999 and the M.S. degree in computer science and engineering from the State University of New York at Buffalo in 2003. He is currently working toward the Ph.D. degree at the College of Computing, Georgia Institute of Technology, Atlanta.

His research interests include performance analysis, resource management, and service provisioning in the next-generation Internet.



Chunming Qiao is currently a Professor at the State University of New York at Buffalo, where he directs the Laboratory for Advanced Network Design, Evaluation and Research (LANDER). He pioneered the research on optical burst switching (OBS) and integrating cellular and ad hoc relaying technologies. His research has been supported by a number of National Science Foundation grants, including two Information Technology Research (ITR) awards, and by seven other major networking research and development organizations. He has published more than 150 papers in leading technical journals and conference proceedings, authored several book chapters, and given several keynote speeches, tutorials, and invited talks.

Dr. Qiao is an editor of several journals and magazines (including the IEEE/ACM TRANSACTIONS ON NETWORKING), and a guest editor for several others, including the IEEE JOURNAL OF SELECTED AREAS IN COMMUNICATIONS and MONET. In addition, he has chaired or co-chaired several international conferences and workshops.