Analysis of TCP over Optical Burst-Switched Networks with Burst Retransmission

Qiong Zhang[1], Vinod M. Vokkarane[2], Yuke Wang[3], and Jason P. Jue[3]
[1] Mathematical Sciences and Applied Computing, Arizona State University West, USA
[2] Department of Computer and Information Science, University of Massachusetts Dartmouth, USA
[3] Department of Computer Science, The University of Texas at Dallas, USA
E-mail: qiong.zhang@asu.edu, {yuke, jjue}@utdallas.edu, vvokkarane@umassd.edu.

Abstract— **Due to the bufferless nature of OBS networks, random burst losses may occur, even at low traffic loads. For optical burst-switched (OBS) networks in which TCP is implemented at a higher layer, these random burst losses may be mistakenly interpreted by the TCP layer as congestion in the network, leading to serious degradation of the TCP performance. In this paper, we reduce random burst losses by a burst retransmission scheme in which the bursts lost due to contention in the OBS network are retransmitted at the OBS layer. The OBS retransmission scheme can then reduce the probability that the TCP layer falsely detects congestion, thereby improving the TCP throughput. We analyze the TCP throughput when OBS networks employ the burst retransmission scheme and develop a simulation model to validate the analytical results. Based on our simulation results, we show that an OBS layer with burst retransmission provides an improvement of up to ten times the TCP throughput over an OBS layer without burst retransmission. This significant improvement is primarily because the TCP layer triggers fewer time-out based retransmissions when the OBS retransmission scheme is used.**

I. Introduction

Optical Burst Switching (OBS) [1] is a promising switching technology that efficiently utilizes the raw bandwidth provided by dense wavelength division multiplexing (DWDM), and at the same time, avoids the need for optical buffering while handling bursty traffic. In an OBS network, a data burst consisting of multiple IP packets is switched through the network all-optically. A *Burst Header Packet* (BHP) is transmitted ahead of the burst in order to reserve the data channel and configure the switches along the burst's route. In the *Just-Enough-Time* (JET) signaling scheme [1], the burst transmission follows an out-of-band BHP after a predetermined offset time. The offset time allows the BHP to be processed before the burst arrives at the intermediate nodes; thus, the burst does not need to be delayed at the intermediate nodes. The BHP also specifies the duration of the burst in order to let a node know when it may reconfigure its switch for the next arriving burst. Other OBS signaling techniques, such as Just-In-Time (JIT) [2] are also implemented in a one-way unacknowledged manner.

Due to the bufferless nature of OBS core network and the one-way based signaling scheme, the OBS network will suffer from random burst losses, even at low traffic loads. One problem arises when TCP traffic traverses OBS networks: the random burst loss may be falsely interpreted as network congestion by the TCP layer. For example, if a burst that contains all of the segments of a TCP sending window is dropped due to contention at a low traffic load, then the TCP sender times out, which leads to false congestion detection. This false congestion detection is referred to as a *False Time Out* (FTO) in [3]. When the TCP sender detects this false congestion, it will trigger the *slow start* congestion control mechanism, which will result in the TCP throughput being reduced. Another example is when a random burst loss triggers TCP fast retransmission for the case in which segments in a TCP sending window are assembled into multiple bursts. The burst loss will be interpreted as light network congestion and will trigger one or more TCP-layer fast retransmissions. The amount of time that it takes for the TCP layer to recover the segments in the lost burst through fast retransmission is referred to as *fast retransmission period*.

Recently several works have evaluated TCP throughput over an OBS network [4], [5], [6]. However, these works assume a constant random burst loss probability in the OBS network, and do not take into account TCP false congestion detection. The work in [3] proposes several schemes for detecting FTOs and reacting with a fast retransmission for each FTO detection. However, these schemes require either that the TCP sender is capable of estimating the maximum number of packets assembled into a burst, or that the OBS nodes are able to send the TCP packet information in a burst back to the TCP sender.

In this paper, we employ a burst retransmission scheme for OBS networks that attempts to avoid TCP FTOs and reduces the TCP fast retransmission period without requiring the OBS layer to maintain information for each individual TCP flow. In the following discussions, we refer to the burst which fails to make a successful channel reservation due to contention at a core node as the *contending burst*. In the OBS retransmission scheme, contending bursts are retransmitted by their source OBS nodes. A contending burst may be retransmitted multiple times until either the burst reaches the egress node, or the burst retransmission process exceeds a delay constraint. If a retransmission results in the burst exceeding its delay constraint, then the burst will simply be dropped.

The TCP over OBS network with burst retransmission has two levels of loss recovery. One level is TCP-layer packet loss recovery through fast retransmission and time-out based retransmission. The other level is through OBS-layer burst retransmission. By setting a very high delay constraint, the OBS-layer retransmission scheme can potentially recover all of the burst losses; however, if the delay constraint is too high, then the TCP layer may recover the lost packets before the retransmitted burst reaches the destination. In this paper, we will determine how to choose a proper delay constraint in order to minimize redundant packet retransmission.

We also evaluate the TCP performance over an OBS network with burst retransmissions through analysis and simulation. TCP flows are classified into fast, medium, and slow flows as in [4]. For a fast flow, all the segments in a TCP source's sending window are assembled in a single outgoing burst. Thus, if the burst is dropped, even after OBS retransmission attempts, then the TCP source will time out. For a slow flow, at most one segment in a TCP source's sending window is included in any given outgoing burst. Thus, the loss of a burst will correspond to a single TCP segment being lost. For a medium flow, the number of segments of a TCP source included in any burst is somewhere between that of a fast flow and that of a slow flow. In this paper, we focus on TCP fast and medium flows, similar to [3]. This assumption is based on a high-speed access network and a reasonable burst assembly time.

The remainder of the paper is organized as follows. Section II provides background information including TCP congestion control mechanisms for different flavors of TCP and the retransmission scheme in the OBS layer. Section III determines a delay constraint for TCP over an OBS network with burst retransmission. Section IV presents an analysis evaluating the TCP throughput over an OBS network with burst retransmission. Section V presents numerical results from the analysis and simulation, and also compares the TCP performance over OBS networks with and without burst retransmission. Section VI concludes the paper.

II. Background

In this section, we describe TCP congestion control mechanisms implemented by two well-known flavors of TCP: Reno and SACK. We also present the burst retransmission scheme in OBS networks.

A. TCP Reno and SACK

TCP congestion control mechanisms include slow start, congestion avoidance, fast retransmission, and fast recovery. If a TCP segment is lost, there are two types of loss indications: *Time Out* (TO) and *Triple Duplicates/partial ACKs* (TD). A TO loss is detected by a *Retransmission Time Out* (RTO), when an acknowledgment for a segment is not received within a certain period of time. TCP interprets a TO loss as a loss due to heavy network congestion; hence, the TCP sender retransmits the lost segment and enters into a *slow start* phase. A TD loss is detected when a TCP sender receives three duplicate ACKs, which indicates that a packet is lost due to light network congestion; hence, the TCP sender enters into *fast retransmission* and *fast recovery* without waiting for RTO.

Several flavors of TCP, such as Reno and SACK, are widely deployed. These TCP variations differ in terms of fast retransmission and fast recovery [7]. In TCP Reno, after a TD loss is detected, the source retransmits one lost segment, reduces the size of congestion window by half, and enters into a fast recovery phase. During the fast recovery phase, the source increases the congestion window by one segment for each duplicate ACK that it receives. After receiving half a window of duplicate ACKs, the congestion window size will be the same as the window size prior to the TD detection. Thus, the source can send a new packet for each additional duplicate ACK that it receives. The source exits

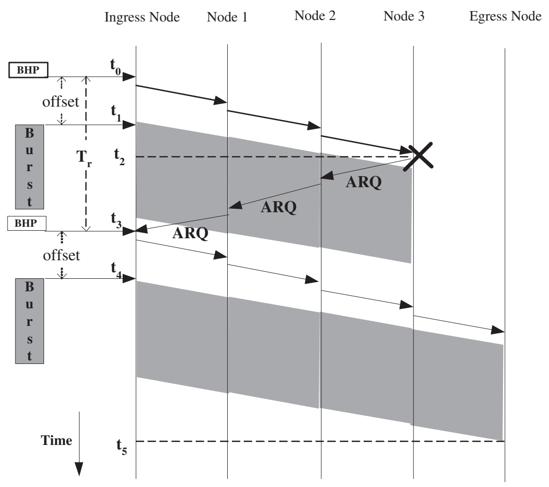


Fig. 1. OBS retransmission scheme.

fast recovery upon the receipt of the ACK that acknowledges the retransmitted lost segment, and enters into a *congestion avoidance* phase. TCP Reno is suitable for single segment loss in the sending window, but does not handle multiple losses well.

TCP SACK is a conservative extension of Reno. An ACK contains a number of SACK blocks, where each SACK block reports a non-continuous set of packets that has been received and queued at the receiver side. After detecting a TD loss, the sender retransmits one lost segment and enters the fast recovery phase. The TCP sender selectively retransmits one or more lost segments that are reported by a SACK block for each *partial ACK* it receives. A partial ACK is an ACK that acknowledges a new segment, but not the segment with the highest sequence number when fast recovery was triggered. When an ACK acknowledges the highest sequence number sent when fast retransmission was triggered, TCP SACK exits the fast recovery phase and enters congestion avoidance. By giving the SACK information, the sender can avoid unnecessary delays and retransmissions as in Reno, resulting in improved throughput.

B. Burst Retransmission in OBS Networks

The burst retransmission scheme retransmits contending bursts in the OBS layer rather than having higher-level protocols, such as TCP, recover lost data. In this scheme, an ingress node stores the duplicates of transmitted bursts for possible retransmissions. If the BHP of a burst fails channel reservation, the core node will send an ARQ to the ingress node in order to report the reservation failure. Upon receiving an ARQ, the ingress node retransmits a duplicate of the requested contending burst preceded by a corresponding BHP. In order to identify the contending burst which needs to be retransmitted, each data burst should be assigned a unique *burst id*.

We illustrate a retransmission scenario in Fig. 1. In this figure, the BHP is transmitted at time t_0, while the burst is duplicated and stored at the ingress node before being transmitted. The burst is transmitted at time t_1 after some offset time. At t_2, the burst reservation fails at Node 3, triggering Node 3 to send an ARQ back to the ingress node. The ingress node receives the ARQ at t_3, then sends a new BHP and retransmits a duplicate burst at t_4 after some offset time. Assuming the second transmission is successful, at t_5 the burst arrives at the egress node. A burst duplicate may be retransmitted multiple times until the burst successfully reaches the egress node.

We observe from Fig. 1 that the retransmission scheme results in an extra delay, T_r, referred to as *retransmission delay*. The retransmission delay is the time elapsed between the initial BHP transmission of a burst and the last ARQ receipt for the corresponding burst, i.e., $t_3 - t_0$. The retransmission delay can be bounded by a delay constraint, notated as δ. Once the ingress node receives an ARQ for a contending burst, the ingress node calculates T_r for the contending burst and decides if it is necessary to retransmit the burst. If $T_r \geq \delta$, the ingress node ignores the ARQ and does not retransmit the contending burst.

If the network is lightly loaded, the retransmission scheme has a good chance of successfully retransmitting contending bursts. If the network is heavily loaded, the retransmitted bursts have a lower probability of being successfully received. Hence, burst losses in an OBS network with burst retransmission can potentially provide a more accurate indication of network congestion to the TCP layer. Compared to an OBS network without burst retransmission, an OBS network with burst retransmissions will have a higher traffic load due to the retransmitted bursts in the network, leading to higher burst contention probability. However, the burst is allowed to experience multiple contentions, which leads to a lower burst loss probability, particularly at lower loads.

In the OBS retransmission scheme, each ingress node must store a copy of each transmitted burst for possible retransmission. Therefore, electronic buffering at each ingress node is necessary. Since the retransmission scheme only reports contention but not the successful receipt of the bursts, the ingress node does not know when to purge the buffer of unwanted bursts. Given the delay constraint, δ, the ingress node can purge the bursts that have been in the buffer for δ units of time. Hence, if we assume that the duration of the burst staying in the retransmission buffer is δ and that the buffer can store k bursts, we can model the retransmission buffer as a $M/G/k/k$ queuing system. Let P_b be the buffer blocking probability, or the probability that a burst could not be stored in the buffer, and let k be the number of bursts that the buffer needs to store in order to satisfy a given buffer blocking probability. Using the Erlang-B formula, we can obtain the relationship between k and P_b.

III. TCP over OBS Retransmission

TCP performance over an OBS layer is different from TCP performance over traditional packet-switched network since the OBS layer introduces additional burst assembly delay as well as random burst loss due to the bufferless core. The impact of assembly delay on TCP over an OBS layer has been investigated in [4], [5]. In an OBS network with burst retransmission, the delay constraint, δ, not only determines the required buffer size at ingress nodes, but can also impact the TCP performance. In this section, we attempt to determine a reasonable delay constraint independent of the different TCP flows. We denote the delay incurred in the access network as T_a, the burst assembly and disassembly delay as T_b, and the one-way propagation delay incurred in the OBS network as T_p.

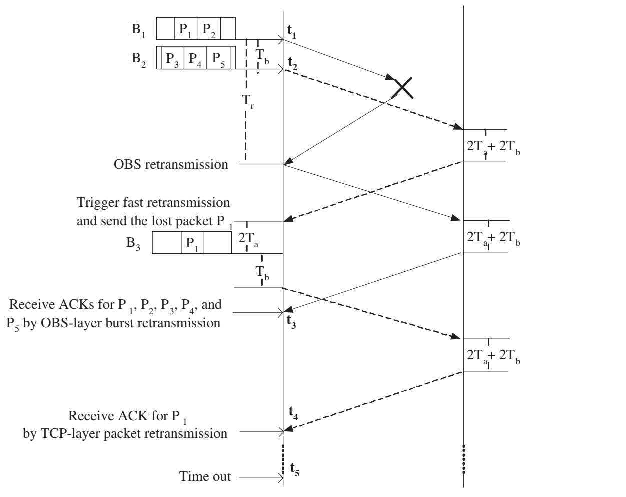

B1
P1
P2
B2
P3
P4
P5
Tb
t1
t2
Tr
2Ta + 2Tb
OBS retransmission
Trigger fast retransmission and send the lost packet P1
B3
P1
2Ta
Tb
2Ta + 2Tb
Receive ACKs for P1, P2, P3, P4, and P5 by OBS-layer burst retransmission
t3
2Ta + 2Tb
Receive ACK for P1 by TCP-layer packet retransmission
t4
Time out
t5

Fig. 2. Two levels of loss recovery for a TCP medium flow.

We first consider a TCP fast flow over an OBS network with burst retransmission. For a TCP fast flow, if a burst experiences contention and if the burst retransmission is successful, the TCP receiver will acknowledge all the packets contained in the burst. The OBS burst retransmission does result in additional retransmission delay; however, as long as the acknowledgments for the corresponding packets arrive at the TCP sender prior to RTO, slow start will not be triggered. Hence, the delay constraint δ can be set to $(RTO - RTT)$ such that the packets in the successfully retransmitted burst are acknowledged earlier than RTO. In TCP, the value of RTO is generally several times the RTT. If we assume $RTO = 2RTT$ when deciding the value of δ, then we have $\delta = RTT = 2T_p + 4T_b + 4T_a$, which is suitable for most TCP flows.

For a TCP medium flow, a burst contention may trigger fast retransmission even when burst retransmission is employed. In this case, packets from a given TCP flow may be spread across multiple bursts. Since the retransmitted burst incurs an extra retransmission delay, bursts that are sent after the contending burst may actually reach the egress node prior to the retransmitted burst. The earlier arrival of these other bursts will result in the generation of duplicate ACKs, leading to the triggering of fast retransmission at the source. Once fast retransmission is triggered, the TCP sender will retransmit a lost packet. At the same time, the OBS layer also attempts to recover the packet losses through burst retransmission. Therefore, there may be redundant retransmissions of packets in the network.

Fig. 2 presents two levels of loss recovery for a TCP medium flow. Packets P_1, P_2, P_3, P_4, and P_5 belong to the same sending window. Packets P_1 and P_2 are assembled into Burst B_1, and Packets P_3, P_4, and P_5 are assembled into Burst B_2. The five packets are acknowledged at time t_3 due to burst retransmission. However, the TCP sender triggers fast retransmission due to the retransmission delay, and retransmits the lost packet P_1. The acknowledgement of Packet P_1 that is retransmitted by the TCP layer during fast retransmission is received at time t_4. Hence, if $t_3 \leq t_4$, the TCP layer will not retransmit additional redundant packets (i.e. Packet P_2) and the TCP fast retransmission period will be reduced. We have $(T_r + 2T_p + 2T_a + 2T_b) \leq (4T_p + 6T_a + 6T_b)$, thus $T_r \leq (2T_p + 4T_a + 4T_b)$. δ can then be chosen to be $(2T_p + 4T_a + 4T_b)$ for a TCP medium flow, which is same as for a TCP fast flow. Since the value of T_a may be variant for different TCP flows and the value of T_b may differ when different burst assembly mechanisms are used, we only use the knowledge of OBS core networks to determine δ. Therefore, we can obtain $\delta = 2T_p$ without considering the values of T_a and T_b. Since the impact of δ on the TCP throughput is independent of T_a, we assume $T_a = 0$ in this paper.

IV. Performance Analysis

In this section, we analyze the TCP throughput for TCP over an OBS network with burst retransmission. In the existing analysis of TCP throughput over an OBS network without burst retransmission [4], [5], burst loss probability is equal to burst contention probability. When burst retransmission is employed, the burst loss probability differs from the burst contention probability since a burst is only considered lost if it experiences contention and it is not successfully retransmitted. We analyze TCP throughput for a TCP fast flow and a TCP medium flow. For the TCP fast flow, all three TCP flavors have the same behavior. For the TCP medium flow, since TCP SACK performs the best, we only analyze TCP SACK throughput.

As defined in [8], a TCP sending *round* refers to the period during which all packets in the sending window are sent and the first ACK for one of the packets in the sending window is received. We assume that the time needed to send all the packets in the sending window is less than RTT. Hence, the duration of a round is equal to RTT. We also assume that the number of packets acknowledged by a received ACK is one (b = 1 in [5]).

We introduce the following notation for a TCP flow:

p_c:	burst contention probability.
p_d:	burst dropping probability.
B:	TCP throughput.
W_m:	TCP maximum window size (in packets).
Z^{TO}:	duration of a sequence of TOs.
H:	no. of segments sent in Z^{TO}.

1) TCP fast flow: Our analysis of a TCP fast flow is similar to that in [4]. However, in our analysis, with the burst retransmission, the successfully retransmitted bursts are treated differently from the bursts that do not experience any contention. The retransmitted bursts suffer from an extra retransmission delay, which has a negative effect on the TCP throughput.

Since a TCP fast flow does not trigger TDs, multiple successful sending rounds are only followed with one or multiple lossy rounds. Therefore, as in [4], a given time-out period includes a sequence of successful rounds and a sequence of lossy rounds. In this time-out period, let X be the number of successful rounds, Y be the number of segments sent before the first lossy round, and A be the duration of the sequence of successful rounds. We can then calculate the TCP throughput as

$$B^f = \frac{E[Y] + E[H]}{E[A] + E[Z^{TO}]}. \tag{1}$$

The sequence of successful rounds consists of a portion of rounds in which the burst does not experience contention and a portion of rounds in which the burst experiences contention, but is successfully retransmitted. Hence, we obtain the probability of a successful round in which a burst experiences contention but is successfully retransmitted as

$$p_{sr} = \frac{p_c - p_d}{1 - p_d}. \tag{2}$$

The probability of a successful round in which there is no burst contention can be calculated as

$$p_{nc} = \frac{1 - p_c}{1 - p_d}. \tag{3}$$

We assume that each retransmission of a burst takes an average time of T_p. Then, the average number of retransmissions for a retransmitted burst, given that the burst needs to be retransmitted at least once and the retransmission is successful, is

$$E[r] = \sum_{i=1}^{\lfloor \delta/T_p \rfloor - 1} i p_c^{i-1}(1 - p_c) + \lfloor \frac{\delta}{T_p} \rfloor p_c^{(\lfloor \frac{\delta}{T_p} \rfloor - 1)}. \tag{4}$$

Hence, the average round trip time experienced by a successfully retransmitted burst is

$$RTT_r = RTT + E[r]T_p. \tag{5}$$

We then obtain $E[A]$ as

$$E[A] = p_{sr}E[X]RTT_r + p_{nc}E[X]RTT. \tag{6}$$

Based on equations (14),(16), (18), and (28)in [4], we have

$$E[Z^{TO}] = RTO\frac{f(p_d)}{1 - p_d}, \tag{7}$$

where, $f(p_d) = 1 + p_d + 2p_d^2 + 4p_d^3 + 8p_d^4 + 16p_d^5 + 32p_d^6$.

$$E[H] = \frac{p_d}{1 - p_d}, \tag{8}$$

$$E[X] = \frac{1 - p_d}{p_d}, \tag{9}$$

and

$$E[Y] = \begin{cases} \frac{1}{p_d^2} & p_d > \frac{1}{W_m} \\ \frac{W_m}{p_d} & otherwise. \end{cases} \tag{10}$$

Since only burst losses result in TOs for a fast flow, the burst loss probability in an OBS network with burst retransmission is applied in the above equations.

By substituting equations (6), (7),(8), and (10) into (1),

$$B^f = \frac{p_d^3 - p_d + 1}{p_d[(1 - p_d)(p_c - p_d)RTT_r + (1 - p_d)(1 - p_c)RTT + p_d f(p_d)RTO]}, \tag{11}$$

when $p_d > \frac{1}{W_m}$. And

$$B^f = \frac{p_d^2 + W_m - W_m p_d}{(1 - p_d)(p_c - p_d)RTT_r + (1 - p_d)(1 - p_c)RTT + p_d f(p_d)RTO}, \tag{12}$$

when $p_d \leq \frac{1}{W_m}$.

2) TCP SACK medium flow: In this section, we analyze the TCP SACK throughput for a TCP medium flow over an OBS network with burst retransmission. TCP SACK triggers fast retransmission when a burst contention occurs, and exits fast retransmission if all the TCP packets that were in the sending window when fast retransmission was triggered are acknowledged. At moderate to high loads, multiple bursts that contain packets belonging to this sending window might experience contention; however, only the first contending burst will trigger a TD event. In other words, some of the burst contentions may not trigger TD events. The analysis in [5] assumes that a burst loss always triggers a TD event. In our analysis, we must obtain the probability of triggering a TD event based on both the burst loss probability and the burst contention probability. Particularly, as OBS retransmission leads to higher burst contention probability, it becomes more important to obtain the probability of triggering a TD event.

Since a TCP medium flow may trigger both TO and TD events, a sequence of TD events may follow a sequence of TO events. The analysis of a sequence of TOs is same as that of fast flow. We focus on the analysis of a sequence of TDs. A TD period is defined as the period between two consecutive TD events.

Let us define the following notation as in [5]:

S: no. of packets from a TCP flow assembled into a burst.
TDP: duration of a TD period.
Y: no. of TCP packets sent during the TD period.
W_X: sending window size of the last round in TD period.
W_b: average number of bursts containing packets from a window of size $E[W_X]$.
Q: probability that a loss indication ending a TDP is a TO.
p_{TD}: probability of triggering a TD event.

We obtain the TCP throughput by

$$B^m = \frac{E[Y] + QE[H]}{E[TDP] + QE[Z^{TO}]}. \tag{13}$$

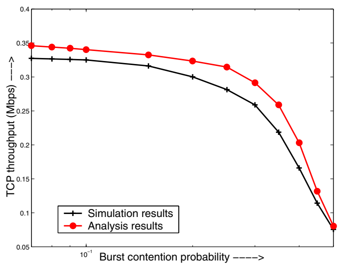
We first need to obtain p_{TD}. The packets from a window of size $E[W_X]$ will be distributed over an average of W_b bursts, where $W_b = \frac{E[W_X]}{S}$. The first burst to experience a contention will trigger a TD event. However, none of the following $W_b - 1$ bursts, regardless of whether or not they experience contention, will trigger a TD event, since TCP SACK exits fast recovery only after all the packets in the window have been acknowledged. Since burst contentions in an OBS network are independent events, the average number of remaining bursts that experience contention but do not trigger TDs is $(W_b - 1)p_c$. Note that the initial burst in W_b that triggered the TD event is not included. The ratio of the number of contentions that trigger TDs to the number of contentions that do not trigger TD is equal to the ratio of the probability of a TD event to the probability of a non-TD contention event. Thus, we have

$$\frac{1}{(W_b - 1)p_c} = \frac{p_{TD}}{p_c - p_{TD}}. \tag{14}$$

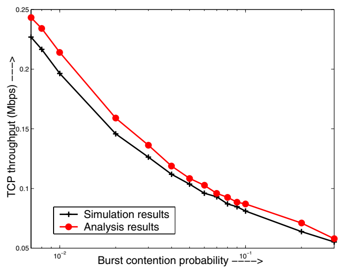
We then have

$$p_{TD} = \frac{p_c}{(\frac{E[W_X]}{S} - 1)p_c + 1}. \tag{15}$$

We use equation (9) in [5] to obtain $E[W_X]$, where



(a) TCP fast flow



(b) TCP SACK medium flow

Fig. 3. TCP throughput vs. burst contention probability for analysis.

$$\frac{3}{8}E[W_X]^2 - E[W_X] - \frac{S}{p_{TD}} = 0. \tag{16}$$

Note that we replace the burst loss probability in the original equation by p_{TD}. By substituting (15) into (16), we have

$$E[W_X] = \frac{8}{3} + \frac{4}{3}\sqrt{4 - \frac{3}{2}(S - \frac{S}{p_c})}. \tag{17}$$

For the case in which burst contention probability and loss probability is very low, the sending window size remains at W_m for a long time, and the analysis is the same as in [5]. The TCP throughput can be approximated as

$$B^m \approx \frac{W_m}{RTT}. \tag{18}$$

We now focus on the case in which burst contention probability is moderately high, and the sending window size rarely reaches W_m. Based on equations (5), (7), and (13) in [5], we have

$$E[Y] = \frac{3}{2}E[W_X] + \frac{1 - p_{TD}}{p_{TD}}S \tag{19}$$

and

$$E[TDP] = RTT(\frac{1}{2}E[W_X] + 1). \tag{20}$$

Since TO events can only be triggered when all bursts in a sending window are dropped, from (25) in [5], we have

$$Q \approx p_d^{(\frac{E[W_X]}{S} - 1)}. \tag{21}$$

By substituting equations (7),(8),(19),(20), and (21) into (13), for a small value of p_c and $p_d \leq p_c$, we can approximate B^m as

$$B^m \approx \frac{1}{RTT}\sqrt{\frac{3S}{2p_c}}. \tag{22}$$

V. Numerical Results

We develop simulations in order to verify the analytical results and to evaluate the performance of TCP over an OBS network with burst retransmission.

Fig. 3 compares the analysis and simulation results for TCP SACK throughput in an OBS network with burst retransmission. We simulate a network with two TCP flows that share a common link. For each flow, we assume $W_m = 50$ and $T_p = 60$ ms. We assume the access bandwidth, $B_a = 5$ Mb/s and $T_b = 10$ ms for a TCP fast flow, which results in $S = 50$ and $S = W_m$. We assume $B_a = 0.5$ Mb/s and $T_b = 10$ ms for a TCP medium flow, which results in $S = 5$ and $S < Wm$. In the OBS layer, the retransmitted bursts that exceed the delay constraint, $\delta = 2T_p$, are dropped. We see that the simulation results match the analytical results well.

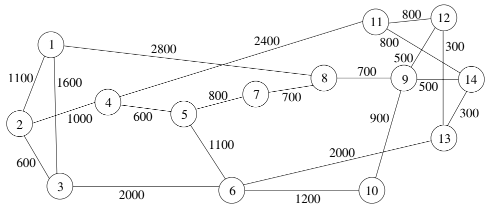
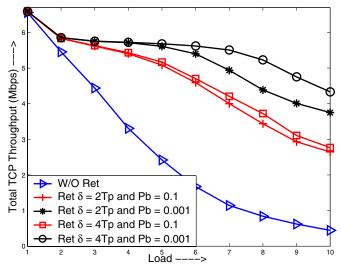


Fig. 4. NSF network.


(a) TCP throughput vs. load

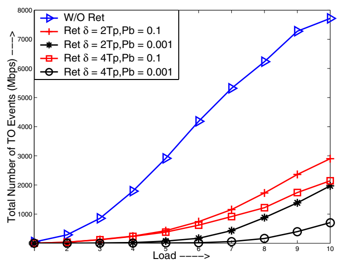
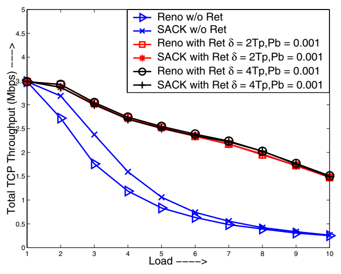

(b) Number of TO events triggered

Fig. 5. Simulation results for TCP fast flows.

We develop a network-wide simulation in order to evaluate the TCP performance over an OBS network with burst retransmission. We simulate the NSF network as shown in Fig. 4. The distances shown are in km. The number of wavelengths on each link is 4 and the transmission rate on a wavelength is 10 Gb/s. We assume core nodes have full wavelength conversion capability. The data traffic traverses through eight ingress-egress node pairs: (1,11), (3,11), (2,9), (3,9), (1,13), (2,10), (4,12), and (7,13). Burst arrivals follow a Poisson process and are uniformly distributed among the eight flows. The burst lengths are exponentially distributed with an average burst length of 100 μs. The load in each figure is the original input traffic load to the entire network. We attach a TCP flow to each of the eight burst flows across the OBS network. We assume that the OBS core network is at a steady-state such that the sending rates of the eight TCP flows do not impact the loss performance of the OBS network. For the burst retransmission scheme, we assume that there are enough buffers at ingress nodes for a given buffer blocking probability P_b and a given delay constraint δ. P_b is also the probability that a burst cannot be retransmitted due to lack of buffers.

Fig. 5 (a) plots the total TCP throughput versus load for TCP fast flows. We can see that, given the same buffer blocking probability, TCP fast flows perform better under higher delay constraint, which implies that the low burst loss probability can overcome the negative effect of retransmission delay in an OBS network with burst retransmission. We observe that TCP fast flows over an OBS network with burst retransmission perform much better than TCP fast flows over an OBS network without burst retransmission. This is because TCP fast flows avoid a significant number of FTOs by using OBS retransmission as shown in Fig. 5 (b). We also see that TCP throughput under $\delta = 2T_p$ and $P_b = 0.001$ is much higher than the throughput under $\delta = 4T_p$ and $P_b = 0.1$ due to lower burst loss probability.

Fig. 6 (a) plots the total TCP throughput versus load for TCP Reno and SACK medium flows. We can see that TCP medium flows also have much higher throughput with OBS retransmission than without OBS retransmission. Furthermore, the number of TO events triggered in the simulation is reduced by using OBS


(a) TCP throughput vs. load

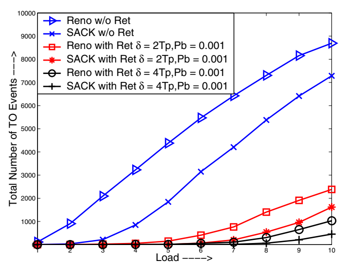

(b) Number of TO events triggered

Fig. 6. Simulation results for TCP medium flows.

retransmission as shown in Fig. 6 (b). As we compare the TCP performance under different delay constraints and $P_b = 0.001$, we observe that the TCP medium flows under low delay constraint, $\delta = 2T_p$, have similar throughput as TCP medium flows under high delay constraints, $\delta = 4T_p$. Since $\delta = 2T_p$ requires less buffers than $\delta = 4T_p$, $2T_p$ is a proper delay constraint for TCP over an OBS network with burst retransmission.

VI. Conclusion

If implemented as a high-speed core network, an OBS network must be able to present an accurate indication of the network congestion situation to the higher layers; otherwise the higher layers may falsely assume that the core network is congested and may take actions which will lead to an unnecessary degradation in performance. In this paper, we employed a simple burst retransmission scheme in an OBS network in order to not only improve the burst loss probability, but also present more accurate OBS network congestion information to the higher layers, such as TCP. We determined a proper delay constraint for TCP over an OBS network with burst retransmission. By presenting a better picture of network congestion, the OBS retransmission scheme can reduce the probability of FTOs and can also reduce the TCP fast retransmission period, thereby significantly improving the TCP throughput. We also analyzed the TCP throughput for the case in which TCP is implemented over an OBS network with burst retransmission. We found that our simulation results closely match the analytical results.

References

[1] C. Qiao and M. Yoo, "Optical Burst Switching (OBS) - A New Paradigm for an Optical Internet," *Journal of High Speed Networks*, vol. 8, no. 1, pp. 69-84, Jan. 1999.

[2] J.Y. Wei and R.I. McFarland Jr., "Just-in-time Signaling for WDM Optical Burst Switching Networks," *Journal of Lightwave Technology*, vol. 18, no. 12, pp. 2019-2037, Dec. 2000.

[3] X. Yu, C. Qiao, and Y. Liu, "TCP Implementation and False Time Out Detection in OBS Networks," *Proceedings of IEEE Infocomm*, 2004.

[4] A. Detti and M. Listanti, "Impact of Segments Aggregation on TCP Reno Flows in Optical Burst Switching Networks," *Proceedings of IEEE Infocomm*, 2002.

[5] X. Yu, C. Qiao, Y. Liu, and D. Towsley, "Performance Evaluation of TCP Implementations in OBS Networks," *Technical Report 2003-13*, The State University of New York at Buffalo, 2003.

[6] S. Gowda, R. K Shenai, K. Sivalingam, and H. C. Cankaya, "Performance Evaluation of TCP over Optical Burst-Switched (OBS) WDM Networks," *Proceedings of IEEE ICC*, 2003.

[7] K. Fall and S. Floyd, "Simulation-based comparisons of Taho, Reno, and SACK TCP," *Proceedings of ACM SIGCOMM*,1996.

[8] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, April 2000.