Performance Evaluation of TCP Implementations in OBS Networks

Xiang Yu†, Chunming Qiao†, Yong Liu* and Don Towsley*
†Department of Computer Science and Engineering, State University of New York at Buffalo
*Department of Computer Science, University of Massachusetts at Amherst

***Abstract*— Since TCP traffic is and may remain as the most popular traffic type in the Internet, it is important to evaluate the performance of TCP in networks employing optical burst switching (OBS), a promising paradigm for the next generation Optical Internet. This work is the first comprehensive study of the TCP performance in OBS networks. Our results provide valuable insights into the interactions between the TCP congestion control mechanism and OBS-specific mechanisms such as burst assembly/disassembly and buffer-less burst switching. In particular, we identify various factors that result in TCP throughput gains and penalties, and determine optimal burst assembly times to be used in OBS networks. In addition, TCP throughput models are developed for the three most popular TCP implementations, i.e., SACK, Reno and New-Reno, and are validated through simulations.**

I. Introduction

IP over Wavelength Division Multiplexing (WDM) networking is a promising architecture to support the expected huge bandwidth demand. Optical Burst Switching (OBS) integrates IP and WDM by leveraging the intelligence and processing capability of electronics as well as the virtually unlimited capacity and low per-bit cost of optical communications (see e.g., [1], [2], [3]). Since OBS combines the best of optical circuit switching (wavelength-routing) and optical packet switching while avoiding their shortcomings, it has received a lot of attention (see http://www.cse.buffalo.edu/ qiao/wobs and http://www.obsforum.org for related links to activities and publications).

In this paper, we study TCP performance in OBS networks, motivated in part by the fact that TCP/IP [4] is a prevailing mechanism for data transmission today and will likely remain so in the next generation Optical Internet. Such a study will also shed light on improvements that need to be made to the current TCP/IP implementation in order to take full advantage of high-speed optical networks (e.g., [5]).

In a TCP/IP over OBS network, the TCP sender/receiver is connected to an OBS network through several IP routers, which form two local IP access networks. The unique aspects of the TCP/IP over OBS network that are relevant to this study are as follows. Firstly, several IP packets from a TCP sender are assembled into a "burst" at an ingress node of the OBS network. Secondly, this burst is then switched (as a whole) inside a *bufferless* OBS core (since no optical RAM exists today, nor is it likely to appear in the near future). Thirdly, the burst is disassembled into IP packets at an egress node of the OBS network and forwarded to the corresponding TCP receiver.

As a result, the performance of TCP in an OBS network can differ from that in a packet switched network. This is because, for example, the burst assembly mechanism in OBS not only introduces an extra delay to incoming IP packets, but also changes the incoming IP packet traffic processes and in particular, enlarges the transmission unit from a packet to a burst. As will be shown later, while assembly delay introduces some *penalty* to the TCP throughput, the combination of burst assembly and the bufferless nature of the OBS core can also delay the first packet loss event for a given TCP flow, thus enabling its TCP congestion window to grow for a longer period of time before being halved due to a packet loss indication. Such a delayed first loss (DFL) in turn results in *gains* in the TCP throughput. In particular, with bufferless switching inside an OBS network, data (burst) losses in an OBS network occur randomly, mostly due to the short range burstiness of the assembled burst traffic, as opposed to correlated packet losses due to long range burstiness in the packet traffic and buffer overflow in electronic packet switched networks.

So far, only a few papers have addressed TCP performance in OBS networks, and none is thorough enough in analyzing the impact (e.g., gains and penalties) of the burst assembly and bufferless switching mechanisms in OBS networks on the TCP throughput. For example, the authors in [6] studied the impact of several burst assembly algorithms on the throughput of TCP in OBS networks through simulations only. The authors in [7] carried limited analysis of TCP Reno's throughput in OBS networks, but the analytical results therein are mostly confined to the case with a single TCP flow whose access bandwidth is either very low or very high relative to the assembly time.

In this paper, we conduct both analytical and simulation studies of the interactions between the TCP's congestion control mechanism and the unique burst assembly and bufferless switching operations within the OBS network. We develop closed form throughput models for the most common TCP implementations such as SACK, Reno and New-Reno. Several nonintuitive differences in the performance of these TCP flavors and the sensitivity of their performance to the choice of the burst assembly time are also discussed.

The rest of the paper is organized as follows. In Section II, we first provide some background information including the notations to be used, and the major differences among the three common TCP implementations, i.e., SACK, Reno and New-Reno. We analyze the TCP throughput and optimal burst assembly time for SACK, Reno and New-Reno in Section III. OBS-specific factors that result in TCP throughput gains and

penalties are discussed in Section IV. Section V presents the simulation results and discussions that validate the proposed TCP throughput models. Finally, Section VI concludes this paper.

II. Background

In this study, we assume that each TCP segment is contained in one IP packet as in [8] (but in our study, multiple TCP segments may be contained in one burst). Further, a simple timer-based burst assembly algorithm will be considered, although much of the analysis and discussion will also be applicable to other burst assembly algorithms. Using such an assembly algorithm, a burst assembly timer is initialized at the beginning of each assembly cycle at an OBS ingress node. The packets which are destined to the same OBS egress node and arrive before the timer expires are assembled into the same burst. After a burst is assembled, it is then transmitted as a basic unit through a bufferless OBS core (cloud) to the destination OBS egress node, where the burst is disassembled back into IP packets, which are then sent to TCP receivers.

When using the timer-based assembly algorithm, the maximum delay for any packet inside a burst to traverse the OBS network is bounded by the assembly time and the propagation delay from the ingress to the egress (which is known if a label switched path is already chosen). In other words, there is no additional, often unpredictable queueing delay inside the OBS core. As a result, one may set the assembly timer to an appropriate value for delay sensitive (but loss-insensitive) real-time applications according to the propagation delay, the available packet delay budget, and/or the packet arrival rate. A typical value of the assemble time is between a few hundreds of nanoseconds to a few hundreds of milliseconds, depending on the applications.

Note that in the TCP/IP over OBS network, there may be packet losses in the two local IP access networks that connect TCP senders and receivers to the OBS edge nodes where TCP segments are assembled and disassembled, respectively. And the impact of such packet losses on the performance of current TCP implementations has been studied in many previous works such as [8], [9], [10].

On the other hand, inside the OBS network, there may be burst losses. Typically, retransmission of the lost bursts is not supported within the OBS network, and the lost TCP segments will be retransmitted by their TCP senders according to the congestion control mechanism employed by TCP. Accordingly, such burst losses insider the OBS network may lead to more significant loss of TCP segments than the packet losses inside an conventional electronic packet-switched network mainly due to the bufferless switching nature of the OBS network. In this paper, we will focus on the impact of burst assembly and burst losses within the OBS network on the TCP throughput. To distinguish such an impact from that of the packet losses in the access networks, we will assume that the IP access networks are lossless while the OBS network has a burst loss rate p (see Sec. III for more discussions).

A. Notations

To facilitate our presentation, the following notations will be used for a TCP flow:

λ	:	local access bandwidth to an OBS ingress (in either segments per second or Bps)
T_b	:	Burst assembly time (in seconds)
W_m	:	TCP maximum window size (in segments)
S	:	Number of segments from one TCP flow contained in one burst
B	:	TCP throughput (in segments per second)
RTT	:	TCP Round Trip Time (in seconds)
RTO	:	TCP Time Out Value (in seconds)
TDP	:	Triple Duplicate Period, a period between two triple duplicate ACK events
$\|TDP_i\|$	:	duration of the ith TDP or TDP_i (in seconds)
Y_i	:	Number of segments sent in TDP_i
X_i	:	Number of sending rounds in TDP_i
W_j	:	Sending window size in the jth sending round (in segments) in TDP_i
TOP	:	Time Out Period
$\|TOP_i\|$	:	duration of the ith TOP or TOP_i (in seconds)
H_i	:	number of segments sent in TOP_i

Note that the number of TCP segments from one TCP flow that are assembled in a burst, S, is at least 1 even when the assembly time $T_b = 0$, and is at most equal to the maximal window size W_m. That is, $S = min\{\lambda T_b + 1, W_m\}$. As to be discussed, the TCP throughput in an OBS network will be a function of S (T_b), RTT, and the burst loss rate p.

B. Reno, New-Reno, and SACK TCP Implementations

In this subsection, we briefly describe the main differences between the three common TCP implementations: Reno, New-Reno and SACK TCP. So far, only limited analysis of TCP Reno in an OBS network was carried out in [7], as mentioned earlier.

Reno TCP refers to TCP with Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery algorithms. When Reno starts, it enters the Slow-Start phase first with a congestion window of size one, and then exponentially expands its sending window after all the packets transmitted in the previous round are acknowledged. When the congestion window reaches a certain threshold, Reno enters the Congestion Avoidance phase during which the window expands by one packet per round.

Reno distinguishes two types of losses, namely timeout (TO) losses and triple duplicate (TD) losses. A TD loss occurs when a Reno sender receives three duplicate ACKs for the same packet, in which case the sender will not wait for a TO before retransmitting the lost packet. During the retransmission, the sender halves its congestion window in response to the loss (or congestion) indication. The rationale

behind this is that a TD loss only indicates light congestion. On the other hand, a TO loss is treated as an indication of heavy congestion, may occur if no more than 3 packets are successfully transmitted before the timer expires. In such a case, Reno enters the Slow Start phase (with a congestion window of one packet), followed by the Congestion Avoidance phase, to retransmit the lost packets as well as new packets. Note that when multiple packets are lost in the same round, e.g., when a burst containing a large number of packets is lost, Reno will halve its congestion window every time it successfully retransmits one lost packet and receives three new duplicate ACKs for the next lost packet in the burst. Eventually, its size may become less than 3 (e.g., this will be the case if the congestion window at the time of the burst loss is small enough). After that, since it is impossible to receive three duplicate ACKs, an additional TO event may be triggered for the remaining packets lost in the burst, which will cause Reno to enter the Slow-Start phase.

New-Reno attempts to recover from multiple losses in a round without halving the window each time when retransmitting a lost packet. Even when multiple packets from a single window of data are lost, New-Reno may recover without a TO by retransmitting one lost packet per RTT upon receiving each partial ACK (which acknowledges a packet with a new but not the highest sequence number), without waiting for three duplicate ACKs. It does not halve the congestion window until all the lost packets from that window have been retransmitted. With the above changes, New-Reno improves the TCP throughput over Reno in a packet switched network. However, in an OBS network, when a large burst is lost, New-Reno can significantly prolong the retransmission period during which no new packets can be sent, therefore its TCP throughput may be worse than that of Reno.

The congestion control mechanisms used in SACK is a conservative extension of Reno's congestion control in that it uses the same algorithms for increasing and decreasing the congestion window. The difference is that the option field in SACK contains a number of SACK blocks, where each SACK block reports a non-contiguous set of data that has been received and queued. With the block information in the ACKs, the TCP sender will be able to send more than one lost packets at a time, which helps in improving the TCP performance in OBS networks.

III. Throughput Model for SACK, Reno and New-Reno

In this section, we develop the throughput models for the three common TCP implementations: SACK, New-Reno and Reno, in an OBS network. We assume that the OBS network has a random loss probability p, which is insensitive to the TCP transmission rate. Accordingly, for each TCP implementation, we only need to consider a single TCP flow as the results for this TCP flow will be representative of the other TCP flows with the same TCP implememtation.

The throughput models to be developed are based on that developed for an electronic packet switched network in [8] where taking into consideration several fundamental differences caused by burst assembly and bufferless burst switching. Note that, it was assumed in [8] that a packet may be lost with probability p, but once a packet is lost, all subsequent packets in the same round are also lost. This implies that p is not exactly the actual packet loss probability. In this paper, every burst is assumed to have a loss probability of p. Accordingly, the actual *packet* loss probability in an OBS network may be different from that in an electronic packet-switched network. Nevertheless, one may compare the TCP throughput in an OBS network using our models with that in a packet-switched network using the model in [8] by assuming the same p. In addition, our models facilitate the comparison between TCP throughput in an OBS network and that in an optical packet-switched (OPS) network as in both the OBS and OPS networks, there is no (optical) buffer at any intermediate nodes, so the burst losses or packet losses will be random. The only difference between the two networks, as far as the TCP throughput is concerned, is that in OPS, there is no burst assembly. Accordingly, for the purpose of evaluating the TCP throughput, OPS may be treated as the special case of OBS where the assembly time T_b is set to 0.

As to be discussed later, the models developed below are also useful in that an optimal assembly time T_b that can achieve the best TCP throughput in OBS networks for a given p and access bandwidth λ can be derived from these models.

A. SACK TCP

In SACK, a loss may be indicated by the missing block information contained in ACKs if any burst in the middle of a sending round is lost due to burst contention. Such a loss is similar to a triple duplicate (TD) loss in Reno and thus will be treated as a TD loss in the following discussion. A loss can also be indicated by "timeout" (TO) when there are no segments delivered successfully in the last round. We obtain the TCP throughput in an OBS network based on the model in [8] with both "TD" and "TO" losses as follows:

$$B(p, T_b) = \frac{E[Y] + Q \times E[H]}{E[|TDP|] + Q \times E[|TOP|]} \tag{1}$$

where Q denotes the ratio between the probability of a TO loss and that of a TD loss (which also equals to the probability that the loss indication ending a TDP is a TO). $E[Y]$ and $E[H]$ are the average number of segments transmitted in TDP and TOP, respectively, while $E[|TDP|]$ and $E[|TOP|]$ denote the average duration of TDP and TOP, respectively. In the rest of this section, we will explain how to derive $E[Y]$, $E[|TDP|]$, $E[H]$, $E[|TOP|]$ and Q.

1) A TD Loss: Suppose that the $(\beta_i + 1)$th burst in TDP_i is the first burst lost in TDP_i, whose first segment is the (α_i+1)th segment in TDP_i. Suppose also that X_i is the round where the first loss occurs, and W_{X_i} is the window size at the end of TDP_i. After burst β_i+1 is sent and lost, γ_i additional segments will be sent in the same round. Then after receiving three duplicate ACKs, the TCP sender will retransmit all the segments contained in the lost burst in the following round according to the information contained in the ACKs. In the retransmission round, $W_{X_i} - S$ new segments can be sent out as shown in Figure 1.

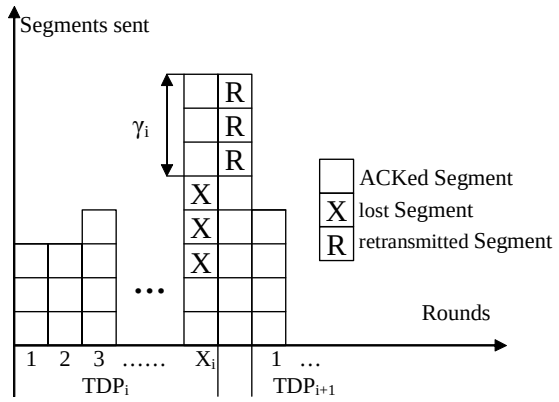


Fig. 1. TCP SACK Retransmission over OBS networks

After the lost burst is successfully retransmitted, TDP_{i+1} starts with a sending window size of $W_{X_i}/2$. The total number of transmitted segments in TDP_i is thus $Y_i = \alpha_i + \gamma_i + W_{X_i} - S$ according to the above analysis. Since $0 \leq \gamma_i \leq W_m$, we can approximate $E[\gamma_i]$ by $E[W_X]/2$ and thus have

$$E[Y] = E[\alpha] + \frac{3}{2}E[W_X] - S \tag{2}$$

Since the burst losses in an OBS network are independent events, the probability of successfully transmitting β_i bursts before a loss happens is

$$P[\beta_i = k] = (1-p)^k p, \;\; k = 1, 2, ... \tag{3}$$

Given that $\alpha_i = S\beta_i$, we have:

$$E[\alpha] = SE[\beta] = S\sum_{0}^{\infty}(1-p)^k pk = S/p \tag{4}$$

Note that in a packet switched network, $E[\alpha] = 1/p$ [8], which is smaller than S/p for $S > 1$. *This increase in the number of segments that can be sent in a TDP before the first loss results in the Delayed First Loss (DFL) gain in TCP throughput as mentioned eariler.*

Substituting (4) in (2) yields:

$$E[Y] = \frac{3}{2}E[W_X] + \frac{1-p}{p}S \tag{5}$$

Next, we further distinguish two subcases with a TD loss. The first is when the maximum window limitation W_m is relatively large such that W_m is rarely reached, or in other words, $W_X < W_m$ most of the time. In the second subcase, p or W_m is relatively small such that $W_X = W_m$ in most rounds. We will discuss these two subcases in the following subsections.

i). When $W_X < W_m$

Let b denote the number of ACKed rounds before the sending window size is increased [8], we have:

$$W_{X_i} = \frac{W_{X_{i-1}}}{2} + \frac{X_i}{b} \tag{6}$$

from which we get:

$$E[X] = \frac{b}{2}E[W_X] \tag{7}$$

Since we can also express Y_i by summing the number of segments sent out in all the previous X_i rounds, and that in the additional S rounds, which is $W_{X_i} - S$, we have:

$$\begin{aligned} Y_i &= \sum_{k=0}^{X_i/b-1}(\frac{W_{X_{i-1}}}{2} + k)b + W_{X_i} - S \\ &= \frac{X_i}{2}(\frac{W_{X_{i-1}}}{2} + W_{X_i} - 1) + W_{X_i} - S \end{aligned}$$

Therefore, we get:

$$E[Y] = \frac{3bE[W_X]^2}{8} + (1 - \frac{b}{2})E[W_X] - S \tag{8}$$

Note that in the above equation, we have assumed X_i to be independent of $W_{X_{i-1}}$. Such an assumption is in general acceptable especially because the burst losses occur randomly without a large variance in the length of the interval between two consecutive losses [11].

Combining equations (8) and (5) yields:

$$\frac{3b}{8}E[W_X]^2 - \frac{b+1}{2}E[W_X] - \frac{S}{p} = 0 \tag{9}$$

Solving the above equation for $E[W_X]$, we have:

$$E[W_X] = \frac{2(b+1)}{3b} + \sqrt{(\frac{2b+2}{3b})^2 + \frac{8S}{3bp}} \tag{10}$$

By substituting (10) into (5), we obtain the following expression of $E[Y]$:

$$E[Y] = \frac{1-p}{p}S + \frac{b+1}{b} + \sqrt{(\frac{b+1}{b})^2 + \frac{6S}{bp}} \tag{11}$$

In addition, by substituting (10) into (7), we obtain

$$E[X] = \frac{b}{2}E[W_X] = \frac{b+1}{3} + \sqrt{(\frac{b+1}{3})^2 + \frac{2bS}{3p}} \tag{12}$$

To derive $E[|TDP|]$, we consider again TDP_i. Define r_{ij} to be the duration (round trip time) of the jth round of TDP_i. Then the duration of TDP_i is $|TDP_i| = \sum_{j=1}^{X_i+1} r_{ij}$. Therefore,

$$E[|TDP|] = (E[X] + 1)E[r] \tag{13}$$

where $E[r] = RTT$. From both (12) and (13), we get the expression of $E[|TDP|]$ as follows:

$$E[|TDP|] = RTT(\frac{b+4}{3} + \sqrt{(\frac{b+1}{3})^2 + \frac{2bS}{3p}}) \tag{14}$$

Note that, for a small loss rate p, $E[|TDP|]$ can be approximated by:

$$E[|TDP|] \approx RTT\sqrt{\frac{2bS}{3p}} \tag{15}$$

ii). When $W_X = W_m$

In this subcase, we can no longer estimate the number of transmission rounds as in (6). As the congestion window size is saturated at W_m, we can obtain $E[Y]$ by plugging $E[W_X] = W_m$ in (5) to obtain

$$E[Y] = \frac{3}{2}W_m + \frac{1-p}{p}S \tag{16}$$

Since there are a total of S/p segments successfully transmitted before a TD loss happens, and during each TDP, the window size starts at $W_m/2$ and ends at W_m, we have the following relationship:

$$(\frac{W_m}{2} + W_m)/2 + W_m(X - \frac{W_m}{2}) - \gamma_i = \frac{S}{p} \quad (17)$$

from which E[X] is calculated as follows:

$$E[X] = (\frac{S}{p} + \frac{W_m^2}{8})/W_m + \frac{1}{2} = \frac{S}{pW_m} + \frac{W_m}{8} + \frac{1}{2} \quad (18)$$

and the expected length of TDP can be calculated by substituting (18) into (13) as follows:

$$E[|TDP|] = RTT(\frac{S}{pW_m} + \frac{W_m}{8} + \frac{3}{2}) \quad (19)$$

2) A TO Loss: Since only one segment is retransmitted after each TO event, the probability that the TO event occurs R_i times, and only after the last occurrence of the TO event, a segment is transmitted successfully, is

$$P[R_i = k] = p^{k-1}(1-p) \quad (20)$$

Accordingly, we can compute $E[R]$ as follows:

$$E[R] = \sum_{k=1}^{\infty} kP[R = k] = \frac{1}{1-p} \quad (21)$$

And the average number of segments transmitted in TO is

$$E[H] = E[R] - 1 = p/(1-p) \quad (22)$$

Since the length of the first six timeout intervals in one TOP are $2^{i-1} \times RTO$, where $i = 1, ..., 6$, respectively, and the length of all the subsequent timeouts is $64 \times RTO$, the duration of a sequence with k timeouts is

$$L_k = \begin{cases} (2^k - 1)RTO & \text{for } k \leq 6 \\ (63 + 64(k-6))RTO & \text{for } k > 6 \end{cases} \quad (23)$$

Accordingly, we obtain the expected length of the duration of TOP, which is denoted by $E[|TOP|]$, as follows:

$$\begin{aligned} E[|TOP|] & \\ &= \sum_{k=1}^{\infty} L_k P[R = k] \\ &= RTO\frac{1+p+2p^2+4p^3+8p^4+16p^5+32p^6}{1-p} \\ &= RTO\frac{f(p)}{1-p} \end{aligned} \quad (24)$$

where

$$f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6$$

Note that in an OBS network, the TO events occur in a pattern that can be quite different from that in a packet-switched IP network. This is because in the latter, packet loss at IP routers is usually due to buffer overflow [8] and thus one packet loss is often followed by the loss of all subsequent packets sent *during the same round*. Accordingly, the probability for a TO event to occur can be approximated with the probability that fewer than three packets are successfully delivered in the last round. On the other hand, in an OBS network, if T_b is large enough such that each burst contains at least three segments (i.e., $S \geq 3$), a TO event occurs *if and only if* all the bursts in the last round are lost. In addition, since there is no buffer at any OBS core node, the correlation between burst losses is small. Accordingly, after a burst is lost, it is difficult to determine how many additional rounds there will be before the TO event happens since subsequent burst loss(es) may happen during additional rounds.

Instead of assuming that once a packet is lost, all subsequent packets in the same round are also lost as in [8], here, we assume that no subsequent burst losses in additional rounds after a TD loss. In other words, a TO event only occurs when all the bursts in the last round (X_i) of a TDP are lost, whose probability is

$$P(W_X) = p^{\frac{W_X}{S}-1}$$

Since a TDP will be followed by either a TOP (with probability $P(W_X)$) or a TDP (with probability $1 - P(W_X)$), and a TOP will always be followed by a TDP, the expected TO versus TD loss ratio is:

$$Q(E[W_X]) = E[P(W_X)] = E[p^{\frac{W_X}{S}-1}] \approx p^{\frac{E[W_X]}{S}-1} \quad (25)$$

3) Throughput Estimation: For the case $W_X < W_m$, by substituting (11), (14), (22), (24) and (25) in (1), we obtain the SACK TCP throughput as follows :

$$\begin{aligned} B(p, T_b) &= \frac{E[Y]+Q(E[W_X])E[H]}{E[|TDP|]+Q(E[W_X])E[|TOP|]} \\ &= \frac{\frac{3E[W_X]}{2}+\frac{1-p}{p}S+Q(E[W_X])\times\frac{p}{1-p}}{RTT(bE[W_X]+1)+Q(E[W_X])RTO(\frac{f(p)}{1-p})} \end{aligned} \quad (26)$$

If p is small, the TO event probability Q will be very small, and the second term in both the numerator and denominator in (1) can be ignored, and (26) can be simplified to:

$$\begin{aligned} B(p, T_b) &= \frac{\frac{S}{p}}{RTT(\sqrt{\frac{2bS}{3p}} + 1)} + o(\frac{1}{\sqrt{p}}) \\ &= \frac{1}{RTT_0 + 2T_b}\sqrt{\frac{3S}{2bp}} + o(\frac{1}{\sqrt{p}}) \end{aligned} \quad (27)$$

where RTT_0 is the TCP round trip time value without burst assembly. And $RTT = RTT_0 + 2T_b$ because both data segments and ACKs experience an assembly delay of T_b.

For the case where $W_X = W_m$, the throughput model is obtained by plugging (16), (19), (22), (24) and (25) in (1):

$$B(p, T_b) = \frac{\frac{3W_m}{2} + \frac{1-p}{p}S + Q(W_m) \times \frac{p}{1-p}}{RTT(\frac{S}{pW_m} + \frac{W_m}{8} + \frac{3}{2}) + Q(W_m)RTO(\frac{f(p)}{1-p})} \quad (28)$$

For a small loss rate p, (28) can be approximated by:

$$B(p, T_b) = \frac{W_m}{RTT} \quad (29)$$

Note that during the lifetine of a TCP connection, both cases discussed above two cases can occur, and hence,the expected throughput will be inbetween the results from (26) and (28).

Note also that the above throughput models can work well for all values of T_b and the access bandwidth λ. For example, if the product $\lambda \times T_b$ is so small that $S = 1$, which is the case referred to as having a "slow" TCP flow in [7], the above models simplify to the same model in packet switched networks as Eq. (31) in [8].

4) Optimal Assembly Time T_b: In this subsection, we consider a practical case in an TCP over OBS network where the maximum number of segments contained in a burst is $1 < S = T_b\lambda + 1 < W_m$, and hence the larger the T_b, the larger the S. In such a case, both the numerator and denominator of (27) contain T_b. The T_b in the numerator represents the DFL gain (in the form of $\sqrt{S}$) in the TCP throughput, in that the TCP throughput increases with the assembly time T_b. On the other hand, the T_b in the denominator represents the delay penalty from the burst assembly process, in that the TCP throughput decreases with the assembly time. Since both the DFL gain and delay penalty are present, there may exist an optimal assembly time that maximizes the TCP throughput.

Note that for both (27) and (29), even though the numerator scales sub-linearly with T_b while the denominator scales linearly with T_b, an optimal T_b exists when the constant factor RTT_0 in the denominator is large.

More specifically, for the case $W_X < W_m$, by computing the root of the following equation, we get the optimal $S = T_b\lambda + 1$ that maximizes B in (27) as follows:

$$\max\{B(T_b)\} \sim \max\{\frac{\sqrt{3\lambda T_b}}{RTT_0 + 2T_b}\} \sim \max\{\frac{1}{\frac{RTT_0}{\sqrt{T_b}} + 2\sqrt{T_b}}\} \Rightarrow \frac{RTT_0}{\sqrt{T_b}} = 2\sqrt{T_b}$$

$$T_b^{opt} = \frac{RTT_0}{2} \tag{30}$$

And the optimal throughput is:

$$B_m(p) \simeq \frac{1}{4RTT_0}\sqrt{\frac{3\lambda RTT_0}{bp}} \tag{31}$$

For the case $W_X = W_m$, we obtain T_b^{opt} by maximizing B in (28) as follows:

$$\max\{B(T_b)\} \sim \max\{\frac{S/p}{(RTT_0 + 2T_b)(\frac{S}{pW_m} + \frac{W_m}{8} + \frac{3}{2})}\} \sim \max\{\frac{1}{\frac{2S}{\lambda pW_m} + \frac{RTT_0(\frac{W_m}{8}+\frac{3}{2})}{S} + (\frac{W_m+12}{4\lambda} + \frac{RTT_0}{pW_m})}\}$$
$$\Rightarrow 2S^2 = \lambda pW_m(RTT_0 + \frac{W_m}{8} + \frac{3}{2}) \Rightarrow$$

$$T_b^{opt} = \sqrt{\frac{pW_m}{2\lambda}(RTT_0 + \frac{W_m}{8} + \frac{3}{2})} \tag{32}$$

And the optimal throughput is:

$$B_m(p) \simeq \frac{1}{\frac{p(W_m+12)}{4\lambda} + \frac{RTT_0 p}{W_m} + \sqrt{\frac{8p(RTT_0 + \frac{W_m}{8} + \frac{3}{2})}{\lambda W_m}}} \tag{33}$$

Note that if the access bandwidth is so low or high that $S = 1$ or $S = W_m$, the DFL gain $\sqrt{S}$ is fixed (which is 1 or $\sqrt{W_m}$). But the delay penalty (in RTT) increases with T_b. Therefore, TCP throughput decreases monotonically with T_b.

B. New-Reno TCP

The major difference between New-Reno and SACK is that after each burst loss, New Reno retransmits only one lost packet in the lost burst after each partial ACK is received. During the fast retransmission phase, with only a partial ACK the number of unacknowledged outstanding packets will soon exceed the sending window and few new packets can be sent. Accordingly, we may ignore the number of new packets sent in the retransmission phase in our analysis as shown in Figure 2.

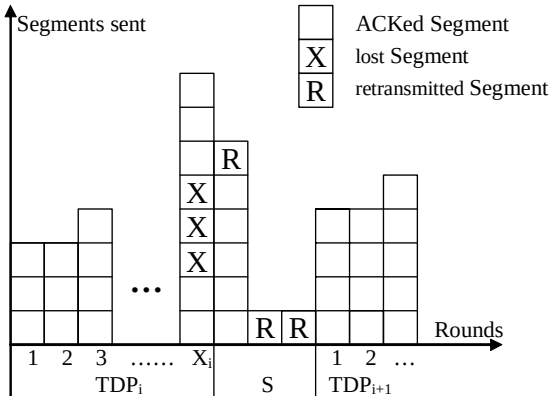


Fig. 2. TCP New-Reno Retransmission over OBS networks

The total packets transmitted in a TDP is the same as SACK but the TDP in New-Reno contains S additional rounds. Similar to (27), when $W_X < W_m$, the throughput of New-Reno can be calculated as follows (assuming p is small):

$$B(p, T_b) = \frac{\frac{S}{p}}{RTT(\sqrt{\frac{2bS}{3p}} + S)} + o(\frac{1}{\sqrt{p}}) \tag{34}$$

which is smaller than the throughput of SACK in (27). However, if the average window size is much larger than the burst length, i.e., $E[W_X] >> S$ or equivalently $\sqrt{\frac{2bS}{3p}} >> S$, S can be ignored in the denominator and (34) is the same as (27), or in other words, New-Reno and SACK have the same throughput. If $W_X = W_m$, $\sqrt{\frac{2bS}{3p}} >> S$ for a small p, and thus the throughput of New-Reno is the same as (29).

C. Reno TCP

It is noted that the retransmission mechanism in Reno is more complicated than in SACK and New-Reno as Reno only uses triple duplicate ACKs to indicate a TD loss. As a result, the retransmission phase could be different for different burst lengths S. More specifically, we discuss two different loss patterns in a TDP, i.e., a TD loss only and mixed TD and TO losses, respectively, in the following subsections.

1) A TD loss only: When the lost burst length S is small, Reno can recover from fast retransmission phase by retransmitting multiple lost packets in each round. This is illustrated in Figure 3.

While the model in SACK ignored the retransmission period following a TDP, here we need to analyze the case with S retransmissions as follows. The first retransmission will

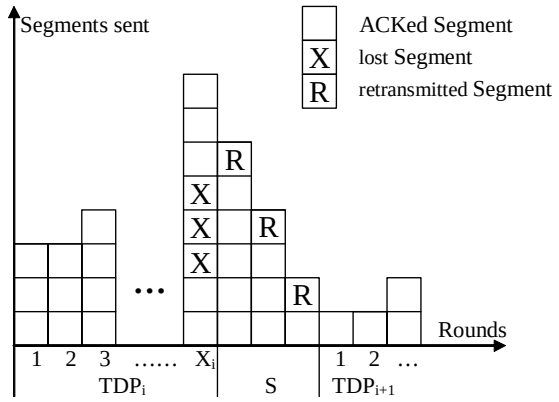


Fig. 3. TCP Reno Retransmission over OBS networks

occur at the end of the 1st additional round, and after each subsequent round, the sender will reduce its window size by half as illustrated in Figure 3. In the TD loss only case, Reno always maintains a congestion window larger than 3 such that duplicate ACKs can be received after each segment retransmission in the lost burst. Suppose that the loss rate p is small and only one burst is lost in the round X_i, the total number of new segments Y_a sent in the additional sending rounds following X_i will be

$$Y_a = \sum_{k=0}^{S-1} \frac{1}{2^k} W_{X_i} - S = (2 - \frac{1}{2^{S-1}}) W_{X_i} - S$$

After a new ACK arrives, TDP_{i+1} starts with a sending window size of $W_{X_i}/2^S$. The total number of transmitted segments in TDP_i is thus $Y_i = \alpha_i + \gamma_i + Y_a$, and similar to the calculation for SACK, we have

$$E[Y] = \frac{1-p}{p} S + (\frac{5}{2} - \frac{1}{2^{S-1}}) E[W_X] \tag{35}$$

Also, the sending window size should be calculated as:

$$W_{X_i} = max\{1, \frac{W_{X_{i-1}}}{2^S}\} + \frac{X_i}{b} \tag{36}$$

For the TD loss only case, the congestion window can be halved multiple times but must still remain larger than 3 before all the S segments are successfully retransmitted, i.e., $logW_X > S$, and the start window of the next TDP is always larger than 1. Therefore,

$$W_{X_i} = \frac{W_{X_{i-1}}}{2^S} + \frac{X_i}{b} \tag{37}$$

from which we get:

$$E[X] = bE[W_X] - \frac{bE[W_X]}{2^S} \tag{38}$$

Since we can also express Y_i by summing the number of segments sent out in all the previous X_i rounds and in the S additional rounds, we have:

$$\begin{aligned} Y_i &= \sum_{k=0}^{X_i/b-1} (\frac{W_{X_{i-1}}}{2^S} + k)b + \\ &\quad [W_{X_i} - S + \sum_{j=1}^{S-1} \frac{1}{2^j} W_{X_i} - S] \\ &= \frac{X_i W_{X_{i-1}}}{2^S} + \frac{X_i}{2}(\frac{X_i}{b} - 1) + \\ &\quad (2 - \frac{1}{2^{S-1}}) W_{X_i} - 2S \end{aligned}$$

as long as S is reasonably large such that $\frac{1}{2^S} \ll 1$. In addition, by taking the expectation of both sides of the above equation, and applying (38) to it, we get:

$$\begin{aligned} E[Y] &= \frac{E[X]^2}{2b} + \frac{E[X]}{2} + 2E[W_X] - 2S \\ &= \frac{(bE[W_X] - b)^2}{2b} + \frac{bE[W_X] - b}{2} + 2E[W_X] - 2S \\ &= \frac{bE[W_X]^2}{2} - \frac{bE[W_X] - 4}{2} - 2S \end{aligned} \tag{39}$$

Combining equations (39) and (35), we have:

$$\frac{b}{2} E[W_X]^2 - \frac{b+1}{2} E[W_X] - \frac{1+p}{p} S = 0 \tag{40}$$

Solving the above equation, we get:

$$E[W_X] = \frac{1}{2} + \frac{1}{2b} + \sqrt{(\frac{1}{2} + \frac{1}{2b})^2 + \frac{2S}{bp}} \tag{41}$$

By plugging (41) in (35), we get the expression of $E[Y]$ as follows:

$$\begin{aligned} E[Y] &= \frac{5}{2} E[W_X] + \frac{S}{p} - S \\ &= \frac{5}{4} + \frac{5}{4b} + \sqrt{(\frac{5}{4} + \frac{5}{4b})^2 + \frac{10S}{bp}} + \frac{1-p}{p} S \end{aligned} \tag{42}$$

In addition, by plugging (41) in (38), we obtain $E[X]$ as follows:

$$E[X] = bE[W_X] - b = -\frac{b}{2} + \frac{1}{2} + \sqrt{(\frac{b}{2} + \frac{1}{2})^2 + \frac{2bS}{p}} \tag{43}$$

And similarly, we get the expression of $E[|TDP|]$ as follows:

$$E[|TDP|] = RTT(-\frac{b}{2} + \frac{1}{2} + S + \sqrt{(\frac{b}{2} + \frac{1}{2})^2 + \frac{2bS}{p}}) \tag{44}$$

With a small loss rate p, $E[|TDP|]$ can be approximated by:

$$E[|TDP|] \approx RTT(\sqrt{\frac{2bS}{p}}) \tag{45}$$

2) Mixed TD and TO Losses in Reno: When the lost burst length S is large, retransmission of multiple lost packets may result in a TO event. Such a loss situation in Reno where a TO event is caused by consecutive TD losses is illustrated in Figure 4. Since after each TD loss, the congestion window will be cut in half, with several consecutive cuts, the congestion window will be reduced to 1 and no duplicated ACKs can be received before all the lost segments can be retransmitted. Therefore, Reno's sending window is stalled, and eventually the timer expires and a timeout retransmission is triggered. *Note that such a loss situation was not considered in [8] since it ignored the retransmission following a TDP.* This mixed TD

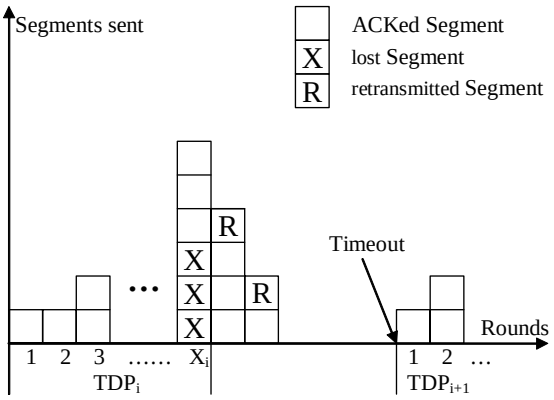


Fig. 4. TCP Reno Retransmission over OBS networks

and TO loss scenario happens when the congestion window is reduced to be less than 3 before all the lost segments are retransmitted, i.e., $logW_X < S$. In this case, a TDP is followed by $logW_X - 1$ retransmissions and a TOP. And the next TDP will start from 1. The total number of segments sent during the TDP and following TOP are $E[Y^*] = E[Y] + E[H]$, with the duration being $E[|TDP^*|] = E[|TDP|] + E[|TOP|]$.

To estimate $E[Y]$, we still follow the same method used for the case with a TD loss only, but change (35) to:

$$E[Y] = (\frac{5}{2} - \frac{1}{2^{logW_X}})E[W_X] - logW_X + \frac{S}{p} \quad (46)$$

and (38) to:

$$E[X] = bE[W_X] \quad (47)$$

assuming that the slow start threshold is 1.

Similarly to the TD loss only case, we obtain $E[W]$ in the same way as (41) and have

$$E[Y] = \frac{5}{2}E[W_X] + \frac{S}{p} - logW_X \approx \frac{S}{p} \quad (48)$$

and

$$E[|TDP|] = (E[X]+logW_X)RTT \approx RTT(\sqrt{\frac{2bS}{p}}+logW_X) \quad (49)$$

And therefore,

$$E[Y^*] = E[Y] + E[H] \approx \frac{S}{p} + \frac{p}{1-p} \approx \frac{S}{p} \quad (50)$$

and thus we have

$$\begin{aligned} E[|TDP^*|] &= E[|TDP|] + E[|TOP|] \\ &\approx RTT(\sqrt{\frac{2bS}{p}} + logW_X) + RTO(\frac{f(p)}{1-p}) \\ &\approx RTT(\sqrt{\frac{2bS}{p}} + log\sqrt{\frac{2S}{bp}}) + RTO \end{aligned} \quad (51)$$

3) Throughput Estimation: For the case where $logW_X > S$, or $log\sqrt{\frac{2S}{bp}} > S$, by plugging (42), (44), (22), (24) and (25) in (1), we can obtain the throughput in TCP Reno for a small p as follows:

$$B(p, T_b) = \frac{\frac{S}{p}}{RTT(\sqrt{\frac{2bS}{p}} + S)} + o(\frac{1}{\sqrt{p}}) \quad (52)$$

For the case where $logW_X < S$, or $log\sqrt{\frac{2S}{bp}} < S$, by plugging (50), (51), (22), (24) and (25) in (1), and with a small p as well, we can obtain the TCP throughput as follows:

$$B(p, T_b) = \frac{\frac{S}{p}}{RTT(\sqrt{\frac{2bS}{p}} + log\sqrt{\frac{2S}{bp}}) + RTO} + o(\frac{1}{\sqrt{p}}) \quad (53)$$

It is noted that for the first case where $log\sqrt{\frac{2S}{bp}} > S$, Reno always has a smaller throughput than New-Reno as they have the same number of retransmission rounds while Reno receives more retransmission penalty by halving the next TDP start window multiple times. For the second case where $log\sqrt{\frac{2S}{bp}} < S$, there are tradeoffs in Reno and New-Reno's throughput as we can see from (34) and (53).

Comparing to (52), the denominator of (53) is enlarged by one more factor RTO, where the TCP throughput is degraded by the timeout retransmission following multiple TD retransmissions. When p is small and S is large such that $RTTlog\sqrt{\frac{2bS}{p}} >> RTO$, (53) approaches (52).

In addition, if $E[W_X]$ is sufficiently large such that $E[W_X] >> \min(S, logW_X)$ and $RTT \times E[W_X] >> RTO$, then both (52) and (53) (as well as (34)) can be further simplified to:

$$\begin{aligned} B(p, T_b) &= \frac{\frac{S}{p}}{RTT\sqrt{\frac{2bS}{p}}} + o(\frac{1}{\sqrt{p}}) \\ &= \frac{1}{RTT_0 + 2T_b}\sqrt{\frac{S}{2bp}} + o(\frac{1}{\sqrt{p}}) \end{aligned} \quad (54)$$

IV. Throughput Gains and Penalties in OBS Networks

Armed with the TCP throughput models developed above, we can now quantify the gains and penalties in the throughput of TCP flows in OBS networks. In this section, besides the delay penalty and delayed first loss (DFL) gain described above, we also identify what we call the loss penalty, and TCP Reno and New-Reno's retransmission penalty that affect the TCP throughput in OBS networks. In the following subsections, we consider gains and penalties by fixing all other factors except the one under consideration.

A. Loss Penalty

The Loss Penalty (LP) is the reduction in throughput due to burst loss (whose rate is p). Intuitively, the larger the burst loss rate, the smaller the TCP throughput. The LP *ratio* is defined as:

$$\begin{aligned}\text{LP Ratio} &= \frac{B(\text{with no loss})}{B(\text{with a burst loss rate } p)} \\ &\approx \frac{\frac{W_m}{RTT}}{\frac{1}{RTT}\sqrt{\frac{3S}{2bp}}} = W_m\sqrt{\frac{2bp}{3S}} \end{aligned} \quad (55)$$

for a small loss rate p.

B. Delay Penalty

As mentioned earlier, the Delay Penalty (DP) is mainly caused by the burst assembly process, which increases the TCP round trip time and decreases the TCP throughput. In particular, the larger the T_b, the larger the DP. Below, we quantify the DP ratio by fixing everything including the time of the first packet loss (thereby excluding any DFL gain) except RTT which increases with T_b:

$$\begin{aligned}\text{DP Ratio} &= \frac{B(\text{with original } RTT_0)}{B(RTT \text{ prolonged due to burst assembly})} \\ &\approx \frac{RTT_0 + 2T_b}{RTT_0} \end{aligned} \quad (56)$$

C. Retransmission Penalty

Retransmission Penalty (RP) is the penalty from prolonged retransmission period that is caused by multiple retransmission rounds, during which fewer new packets can be sent. Since SACK TCP can always retransmit all the packets in the lost burst in one (or a few) round (as the missing block information is contained in the received ACKs), there is no RP in that the number of retransmissions is approximately the same regardless of whether a burst or a packet is lost. For New-Reno TCP, however, multiple retransmission rounds (S) are needed for S packets contained in a lost burst before the next TDP starts. From (27) and (34), we can calculate the RP ratio (RPR) as follows:

$$\begin{aligned}\text{RPR(New-Reno)} &= \frac{B(\text{one retransmission})}{B(S \text{ retransmissions})} \\ &\approx 1 + \frac{S-1}{\sqrt{\frac{2b}{3Sp}}+1} \approx 1 + \sqrt{\frac{3Sp}{2b}} \end{aligned} \quad (57)$$

for a small p and large S. Note that (57) increases with S. For Reno TCP, the retransmission not only prolongs the TDP duration as in New-Reno, but also decreases the start window of the next TDP by halving the congestion window size multiple times during retransmission. For the case where $log\sqrt{\frac{2S}{bp}} > S$, we can calculate the RPR using the model in (52) as follows:

$$\begin{aligned}\text{RPR(Reno)} &= \frac{B(\text{one retransmission})}{B(S \text{ retransmissions})} \\ &\approx \sqrt{3}(1+\sqrt{\frac{Sp}{2b}}) \end{aligned} \quad (58)$$

for a small p.

And for the case where $log\sqrt{\frac{2S}{bp}} < S$, we can calculate the RPR using the model in (53) as follows:

$$\text{RPR(Reno)} \approx \sqrt{3}[1 + (\frac{RTO}{RTT} + \log\sqrt{\frac{2S}{bp}}) \times \sqrt{\frac{p}{2bS}}] \quad (59)$$

for a small p.

We note that for the case where $log\sqrt{\frac{2S}{bp}} > S$, RPR(Reno)>RPR(New-Reno), and thus New-Reno always exhibits a higher throughput than Reno. In the second case where $log\sqrt{\frac{2S}{bp}} < S$, if $1 + \sqrt{\frac{3Sp}{2b}} > \sqrt{3}[1 + (\frac{RTO}{RTT} + log\sqrt{\frac{2S}{bp}}) \times \sqrt{\frac{p}{2bS}}]$, New-Reno exhibits a higher throughput, otherwise, Reno has a better throughput.

D. DFL Gain

If we compare (27) which is the TCP SACK throughput model in an OBS network with the throughput model below from [8] without burst assembly (and without considering the W_m limitation)

$$B(p) = \frac{1}{RTT_0}\sqrt{\frac{3}{2bp}} + o(\frac{1}{\sqrt{p}}) \quad (60)$$

and ignore the difference in RTT, we can see that the larger the S in (27), the larger the throughput and the larger the DFL gain, whose ratio can be quantified as follows (note that, as before, we should ignore the delay penalty when considering only DFL gain):

$$\begin{aligned}\text{DFL Gain Ratio} &= \frac{B(\text{the first loss is delayed})}{B(\text{the first loss not delayed})} \\ &\approx \sqrt{S} \text{ (for a fixed small loss } p) \end{aligned} \quad (61)$$

Note that the combination of DFL gain and retransmission penalty is equivalent to what is termed "Correlation Gain" in [7], which did not analyze any of the three TCP implementations in detail.

E. Impact of Multiple Gains and Penalties on TCP

Since the gain and the penalties can affect TCP throughput at the same time, we define the following special ranges in order to quantify the impact of the assembly or the burst loss rate on the TCP throughput.

1) Assembly Time Optimal (ATO) range: For a TCP flow with high access bandwidth and a given loss rate p, there may exist a range of assembly times where the DFL gain is larger than the delay and retransmission penalties as follows:

$$\text{ATO Range} = \bigcup T_b : \frac{\text{DFL Gain}}{\text{DP} \times \text{RP}} > 1$$

which we call the "assembly time optimal" (ATO) range. In this ATO range, the TCP flow usually achieves a higher throughput in an OBS network than in a packet switched network (especially an optical packet switched network) without burst assembly (note that both incur the same loss penalty). In such an ATO range, the best throughput in an OBS network can be achieved with the following optimal assembly time:

$$T_b^{opt} = \arg\max_{T_b}\{\frac{\text{DFL Gain}}{\text{RP} \times \text{DP}}\}$$

2) Assembly Time Insensitive (ATI) range: For a TCP flow with a low access bandwidth, there could exist a range of assembly times for which the DFL gain is compatible to all the delay and retransmission penalties as defined below:

$$\text{ATI Range} = \bigcup T_b : \frac{\text{DFL Gain}}{\text{RP} \times \text{DP}} \approx 1$$

which we call a "assembly time insensitive" (ATI) range. In this range, the throughput of the TCP flow is almost independent of T_b, and hence, an OBS network has an approximately the same throughput as a packet switched network without burst assembly.

3) Loss Rate Insensitive (LRI) range: If we take into account all gains and penalties (including the constant delay penalty for a fixed T_b), there will be a "loss rate insensitive" (LRI) range (defined as follows), where the TCP throughput is almost independent of the burst loss rate p.

$$\text{LRI Range} = \bigcup p : \frac{\text{DFL Gain}}{\text{DP} \times \text{RP} \times \text{LP}} \approx 1$$

The simulation results to be presented next validate the above analysis and in particular, the existence of the ATO, ATI and LRI ranges.

V. Numerical Results

In this section, we present the numerical results from both NS-2 simulations and the above analysis. In our simulations, each segment has the same unit size (as assumed in the analysis), whose default value is 1KB. The maximum window limitation W_m varies from 20 to 200 (segments), and the throughput is obtained over a period of approximately 10^6s.

When simulating the performance of a single TCP flow in an OBS network, it is sufficient to model the OBS network with two edge nodes, and two core codes which form a path of four nodes using three fiber links. Each of the three fiber links has 10ms delay in this simulation and 10GBps bandwidth, with a given loss probability p. The effect of p on the TCP throughput is studied by varying it from 10^{-4} to 10^{-1}. In addition, the TCP sender and receiver connect to the OBS edge nodes via a lossless link with 10ms propagation delay, and a constant access bandwidth that varies from 1.25KBps to 125MBps. Therefore, the round trip time (RTT) excluding the assembly time, transmission time as well as any queuing delay is $2 \times 50ms$ or 0.1s.

A. Throughput of a TCP SACK Flow

In this subsection, we illustrate the impact of the assembly time and burst loss rate on the TCP throughput using SACK as an example. We will compare the throughput results for SACK, Reno and New-Reno later.

In the simulation, we assume $W_m = 50$, and consider the cases with a high, low and medium access bandwidth λ (relative to the given W_m). To simulate the case with a high access bandwidth, λ is set to be 125MBps. The round trip time measured in simulation is approximately $RTT = 0.135 + 2T_b$ seconds[1]. With such a high access bandwidth, the maximum assembly time needed to assemble all the TCP segments in a sending round is much smaller than RTT_0, the round trip time without burst assembly. That is, $W_m/\lambda = 50KB/125MBps = 0.0004s << 0.135s$. The actual assembly time chosen by the OBS assembly node could be either less than, equal to or larger than 0.0004s, resulting in a different TCP throughput.

To simulate the case with a low access bandwidth, λ is set to be 1.25KBps, and the round trip time for a given loss rate p (from 0.003 to 0.1) varies from $18 + 2T_b$(s) to $5.4 + 2T_b$(s). With such a low access bandwidth, the maximum assembly time is much larger than RTT_0 (which varies from 5.4s to 18s). For the case with a medium access bandwidth, λ is set to be 500KBps, and the round trip time is approximately $0.145 + 2T_b$(s), where the maximum assembly time is 0.1s. As mentioned earlier, a large assembly time on the order of a hudnred of milliseconds or longer may not be practical for some real-time applications. Nonetheless, in the simualtion, we show the entire range of assembly times as a way to explain the gains and penalties in the TCP throughput that have been analyzed in the previous sections.

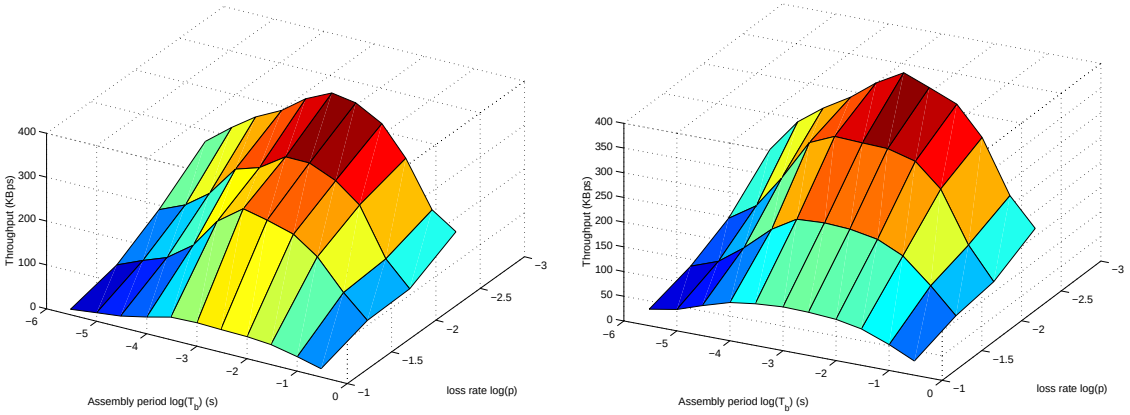
The throughput of a TCP SACK flow with high, medium or low access bandwidth as a function of the burst loss rate and assembly time is shown in Figure 5(a), 6(a) and 7(a), respectively. Figure 5(b), 6(b) and 7(b) show the results obtained from the analytic model in (26) and (28) which approximately match their corresponding simulation results.

From Figure 5, one can see that there indeed exists an optical assembly time with a high access bandwidth. More specifically, the throughput reaches its peak in an ATO range centered around an optimal assembly time $T_b \approx 10^{(} - 3)$ seconds or $1ms$ when the burst loss rate $p = 10^{-2} = 0.01$. (Note that, according to Eq. 32, $T_{opt} \approx 4ms$ but due to the log scale used which does not collect a sample at $T_b = 4ms$, the closest point is $T_b = 1ms$). With the high access bandwidth, the penalty due to the delay introduced by burst assembly is insignificant as the assembly time is much smaller compared to the round trip time without burst assembly. Thus, T_b^{opt} is where the DFL gain reaches maximum. In addition, one can also see that there is no obvious LRI range, and usually the smaller the p, the larger the throughput (for a given T_b) as shown in Figure 5(a). The throughput obtained from the simulation also matches with that from our analysis.

Figure 6 shows the throughput results for a TCP flow with a medium access bandwidth. Similar to the case with a high access bandwidth, the optimal assembly time and the throughput obtained from our simulations match well with those from our analysis.

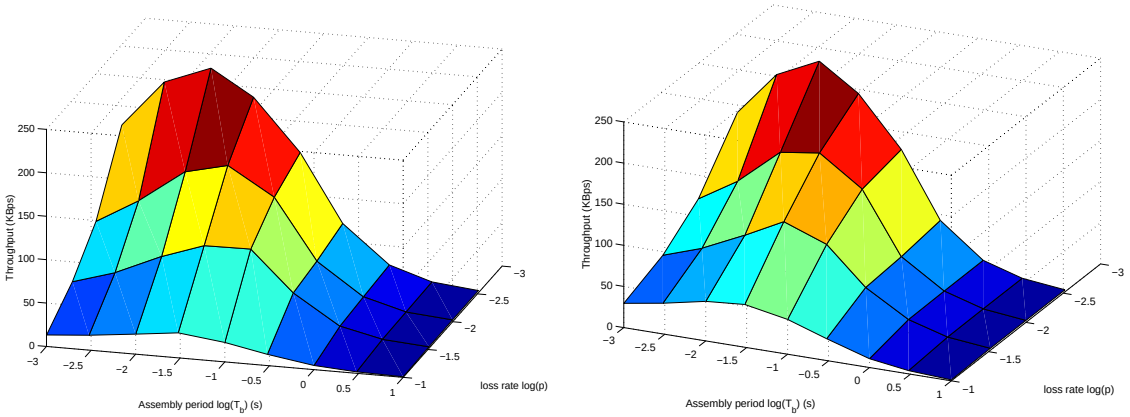
For the case with a low access bandwidth, there is no obvious ATO range since the assembly time is relatively large compared to the TCP round trip time without burst assembly, and therefore the delay penalty significantly offsets the DFL gain. Instead, there is an ATI range due to the offsetting effect. And the lower the loss rate, the larger the the ATI range. Note that here the optimal T_b^{opt} calculated from (30) is applicable (since $\lambda T_b < W_m$), which is $18/2 = 9s$ for a small loss rate (e.g., $p = 0.003$) and $5.4/2 = 2.7s$ for a large loss rate (e.g. $p = 0.1$). We can see that these T_b^{opt} are at the edge of the

[1]which includes the propagation delay, transmission delay, assembly delay and queuing delay in the access networks. Note that the queuing delay is negligible for a high access bandwidth, but could has a large variance for a low access bandwidth.



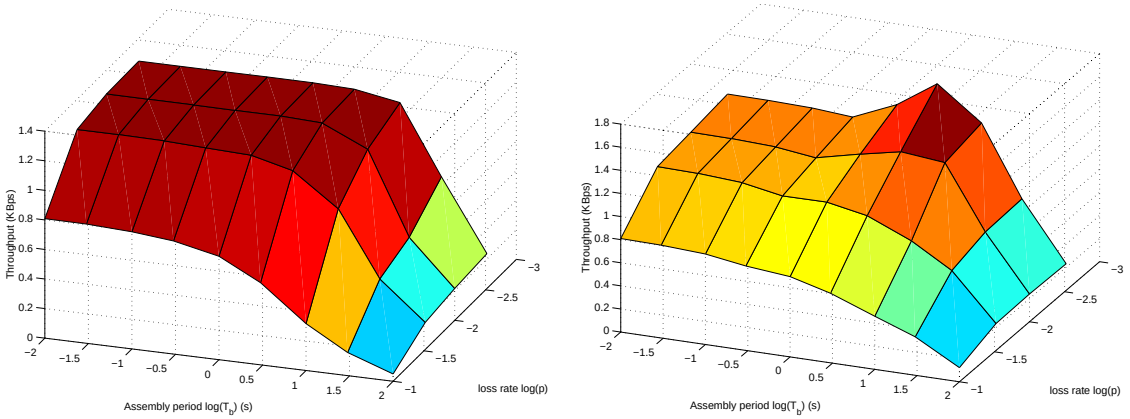
(a) Simulation Results (b) Matched Mixed Model from (26) and (28)

Fig. 5. Throughput of a TCP SACK flow with a high access bandwidth (125MBps) channel



(a) Simulation Results (b) Matched Mixed Model from (26) and (28)

Fig. 6. Throughput of a TCP SACK flow with a medium access bandwidth (500KBps) channel



(a) Simulation Results (b) Matched Mixed Model from (26) and (28)

Fig. 7. Throughput of a TCP SACK flow with with a low access bandwidth (1.25KBps) channel)

ATI range as shown in Figure 7(a). In addition, there is a LRI range for the TCP flow when $T_b < 10^{0.5}$ seconds as shown in Figure 7(a). This is because with a low access bandwidth, when T_b is small, the DFL gain is also relatively small and is more easily offset by the loss penalty.

B. Comparison Between Three TCP Implementations

In this section, we compare the performance of Reno, New-Reno and SACK in OBS networks. Previous work has evaluated the performance of these TCP implementations with only a few packet losses within a sending round, but not as many packet losses as what occurs in an OBS network with a burst loss.

Generally speaking, all three TCP implementations have the same Slow Start and Congestion Avoidance algorithms, and the DFL gain and delay penalty mentioned earlier will be the same as long as the assembly time or burst size is kept the same. The differences between the three TCP implementations come from the fast retransmission and fast recovery mechanisms, and their interactions with burst assembly in OBS networks, which are the focus of this section. Accordingly, in the rest of the study, we will ignore the case where $S = 1$ (i.e., with a low access bandwidth and a short assembly time) as this is not much different from the case without burst assembly (i.e., the case with packet-switching). We will also ignore the case where $S = W_m$ (i.e., with a high access bandwidth and a long assembly time) as any burst loss will always trigger a timeout (TO) event, and hence, the three TCP implementations will all apply the same TO retransmission mechanism (i.e., Slow Start). In other words, we will only focus on the case where $1 < S < W_m$.

To explain the performance differences between the three TCP implementations and validate some of the analysis presented earlier, we first examine the packet traces collected from our simulations where a single burst is dropped. We then examine the packet traces corresponding to multiple (random) burst losses. Finally, we present the throughput of the three TCP implementations.

1) Packet Traces with One Burst Loss: In this set of simulations, we first assume that the access bandwidth is 125KBps (low). For each graph showing the packet traces, the X-axis shows the bursts' departure time in seconds, and the Y-axis shows the packets' number mod 60.

Figure 8 illustrates the impact of the congestion window size at the time of a TD loss event in Reno, when the length of the lost burst is $S = 5$ packets. As we can see from Figure 8 (a), if the congestion window[2] is small, i.e., $W = 20$ packets, at the time when the burst is lost (i.e., 3 seconds into the trace), the window size will be reduced to less than 3 after three retransmission rounds but before all the 5 packets lost in a burst can be retransmitted. Therefore, without being able to receive three duplicate ACKs any longer, a TO occurs at time 3.8 and a Slow Start phase begins. However, in Figure 8(b) where the congestion window size has already grown to reach the maximal limit of 200 packets by the time the burst

[2]Note here with a relatively small RTT, TCP packets are pipelined and congestion window cannot be represented in the graph

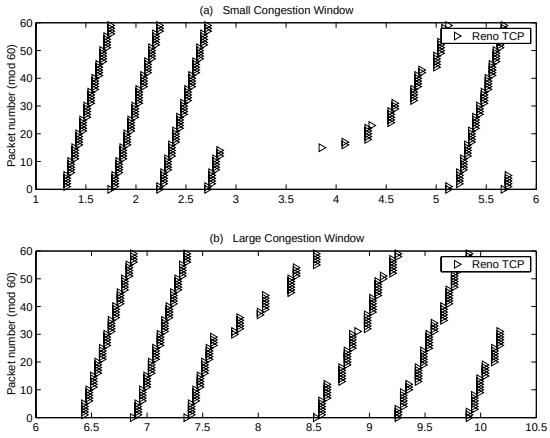


Fig. 8. Packet traces with one burst loss occurred at (a) $W = 20$ and (b) $W = 200$ for a Reno TCP flow ($S = 5$)

loss occurs, Reno can recover from the fast retransmission stage and re-enter the congestion avoidance phase without a TO event.

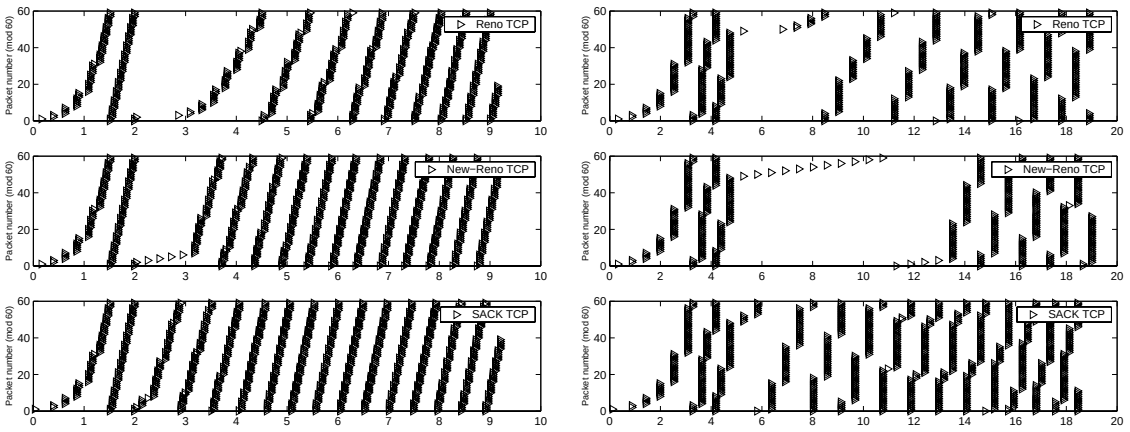
Figure 9(a) illustrates the packet traces of the three different TCP implementations upon one burst loss when both S and W are small (5 and 40 in the simulation, respectively). It can be seen that SACK has the best performance because the ACK can indicate the block of packets lost, and then the sender sends out the all the lost packet again upon receiving the ACK. In addition, New-Reno generally performs better than Reno because New-Reno detects the loss of subsequent packets upon receiving a partial ACK, without waiting for 3 duplicate ACKs or a TO (while Reno will have a TO as discussed earlier). In addition, New-Reno's congestion window will only be halved once (i.e., $W = 40/2 = 20$) after a TD loss, which enables New-Reno to recover quickly from the loss of a small burst.

Figure 9(b) shows that when the lost burst is large, ($S = 25$ in the simulation) while W is still relatively small (e.g., 40) when the burst loss occurs, 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 to finish all the 25 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 TO value is much smaller than 25 RTT in the New Reno's fast retransmission phase. Moreover, Reno's transmissions after TO is much more efficient because it exponentially increases the sending window size. Also from Figure 9(b), we can see that in this case, SACK has a much better performance than Reno and New-Reno due to its selective acknowledgements.

We note that if S is small but W is relatively large at the time that the burst loss occurs as in Fig. 8 (b), Reno will not have a TO, and hence its performance will be comparable to that of New-Reno.

2) Packet Traces with Multiple Burst Losses: In this subsection, we compare the packet traces of the three TCP implementations with multiple (random) burst losses.

Figure 10(a) illustrates the packet traces of the three different TCP implementations with a medium-high burst loss rate.



(a) $S = 5$ (b) $S = 25$

Fig. 9. Packet traces with one burst loss ($W = 40$ when a burst is lost)

When the burst losses are scattered before time 12, the three TCP implementations have very different performances due to different TD retransmission mechanisms. However, when consecutive burst losses occur one after another, the three TCP implementations have the same performance as such multiple back to back burst losses will easily trigger a TO event (after which, all three TCP implementations perform the same timeout retransmissions). This can be seen from the packet traces shown in Figure 10(a) (after time 28 in Reno, after time 21 in New-Reno and after time 16 in SACK). On the other hand, with a medium-low loss rate, Figure 10(b) shows that SACK always has the best performance as most, if not all, losses are TD losses.

3) Throughput of SACK, Reno and New-Reno: Due to space limitation, we omit the throughput obtained from our analysis for Reno and New Reno since they match well with the throughput obtained from simulations as in the case for SACK discussed earlier. Tables I and II show the throughput ratio of New-Reno over Reno and that of SACK over Reno, obtained from our simulations where the burst loss rate varies from low to high.

In general, for a low or high loss rate, all the three TCP implementations tend to have the similar performance. This is because the performance difference between different TCP implementations comes from TD retransmissions only. Accordingly, a high burst loss rate usually leads to a higher probability of TO retransmissions in which there is no difference in the three TCP implementations. On the other hand, a low burst loss rate leads a low TO probability but also a low TD loss probability.

However, with a medium-low to medium-high loss rate, the probability of a TD event can be relatively high (compared to a TO event), and the performance difference among different TCP implementations will be more obvious, as shown in the middle two columns of Tables I and II.

Tables I and II also show the effect of the burst length S (or assembly time T_b) on the performance ratios of New-Reno over Reno, and SACK over Reno, respectively, It is interesting to note that New-Reno has a better performance with a smaller burst (up to 5 packets), while Reno has a better performance than New-Reno with a larger burst (up to 25 packets) as illustrated in Tables I and II, respectively. In general, their relative performances depend on the TCP's timeout value RTO and round trip time RTT as well as the number of packets contained in the lost burst S. More specifically, when RTO is much larger than $S \times RTT$, New-Reno has a better performance than Reno, otherwise Reno has a better performance because the interval between the time of the previous burst lost and the time to retransmit the next new packet is approximately RTO for Reno and $S \times RTT$ for New-Reno. Since RTT and RTO stablize after TCP starts for some time, S usually decides the relative performances of Reno and New-Reno. Also note that, when S (or T_b) increases, SACK is still better than Reno but its advantage is smaller due to the fact that a TO event can happen more likely.

TABLE I

THROUGHPUT RATIO OF NEW-RENO AND SACK OVER RENO, $S = 5$

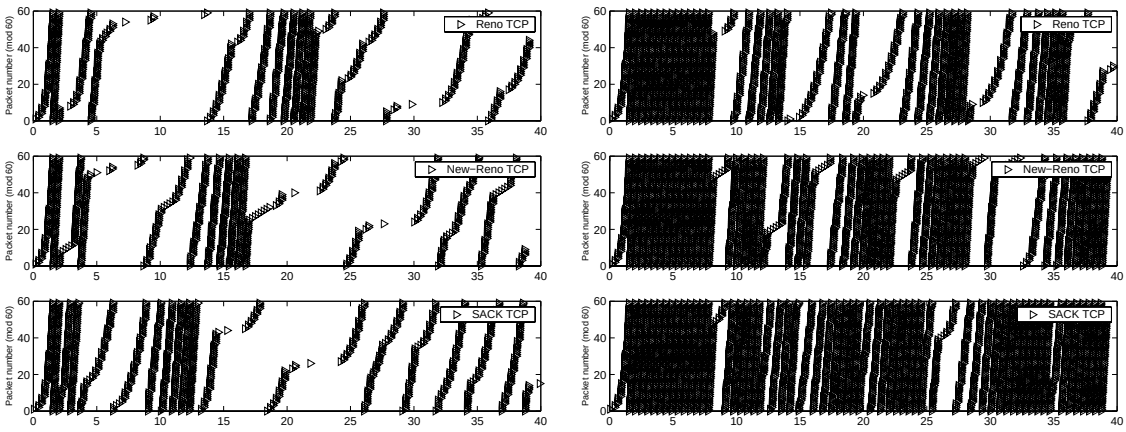
$log(p)$	-3	-2.5	-2	-1.5	-1	-0.5
$NewReno/Reno$	1.0	1.1	1.29	1.19	1.03	0.93
$SACK/Reno$	1.0	1.07	1.58	1.4	1.33	1.07

TABLE II

THROUGHPUT RATIO OF NEW-RENO AND SACK OVER RENO, $S = 25$

$log(p)$	-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

Figure 11 illustrates the throughput of the three TCP implementations as a function of the assembly time with a loss rate $p = 0.03$ and a high access bandwidth 125MBps. It can be seen that all three TCP implementations perform the same when the assembly time is larger than 1ms (this is because $S = W_m$). When the assembly time is below 1ms, SACK always has the highest throughput. New-Reno



(a) $p = 0.1$ (b) $p = 0.01$

Fig. 10. Packet traces with multiple burst losses for a medium-rate TCP flow

outperforms Reno when the assembly time is relatively small, i.e., between $10^{2.5}$ms and 10^{-1}ms, while Reno outperforms New-Reno when the assembly time is relatively large, i.e., between 0.1ms and 1ms.

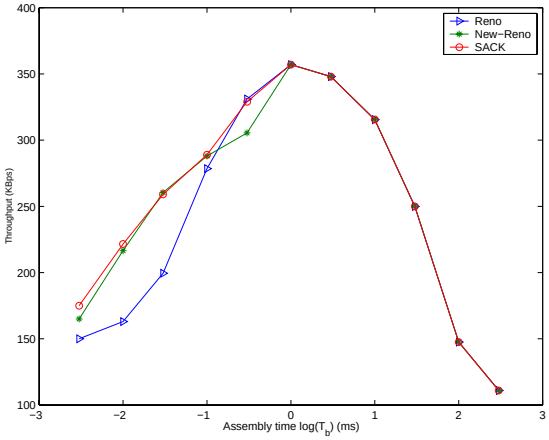


Fig. 11. Throughput comparison between Reno, NewReno and SACK TCP

VI. Concluding Remarks

The TCP throughput in OBS networks is complex to analyze because it is affected by multiple gains and penalties due to the interaction between TCP's congestion control mechanisms and OBS's burst assembly/disassembly and bufferless switching. This work represents the first comprehensive study of the throughput of the three common TCP implementations: SACK, Reno and New-Reno in OBS networks without over-simplifying the assumptions.

Our analytical (as well as simulation) results have shown that burst assembly can sometime increase the TCP throughput due to the delayed first loss (DFL) gain which allows a TCP sender to grow its sending window for a longer period of time before one or more TCP segment losses occur. In addition, there exists an optimal burst assembly time with which the TCP throughput will be maximized. Accordingly, assembling multiple TCP segments into a burst is a way to improve the TCP performance without having to change the existing TCP implementations

On the other hand, our studies have also shown that none of the three TCP implementations is particularly effective in dealing with burst losses when a burst contains a large number of TCP segments. This is due to the delay penalty introduced by the burst assembly process, and more importantly, the time-out (TO) events triggered by one or more burst losses. In this regard, sending a "jumbo" segment or increasing the Maximum Segment Size (MSS) size [12] may improve the TCP throughput more than assembling multiple "small" TCP segments into one burst. This is because when the former technique is used, there is a gain in the TCP throughput similar to the DFL gain but on the other hand, losing a jumbo packet will not likely to trigger a TO event. Nonetheless, the former technique requires modifications to the existing TCP implementations, and moreover, if/when such modifications are made, assembling these jumbo packets into a burst may still provide a further throughput improvement.

As a part of our future work, we will also extend our work to new congestion control schemes, such as HighSpeed TCP [13] and FAST [14], which are proposed specifically for high bandwidth networks.

References

[1] M. Yoo and C. Qiao, "Just-Enough-Time (JET): A high speed protocol for bursty traffic in optical networks," in *IEEE/LEOS Conf. on Technologies For a Global Information Infrastructure*, 1997, pp. 26–27.
[2] C. Qiao and M. Yoo, "Optical burst switching (OBS) - a new paradigm for an Optical Internet," *Journal of High Speed Networks*, , no. 8, pp. 69–84, 1999.
[3] J. Turner, "Terabit burst switching," *Journal High Speed Networks*, vol. 8, pp. 3–16, 1999.
[4] W. Stevens, "TCP/IP Illustrated," 1994.
[5] X. Yu, C. Qiao, and Y. Liu, "TCP implementations and false time out detection in OBS networks," in *Proceedingss of INFOCOMM*, 2004.
[6] X. Cao, J. Li, Y. Chen, and C. Qiao, "Assembling TCP/IP packets in optical burst switched networks," in *Proceedings of GLOBECOM*, 2002.

[7] A. Detti and M. Listanti, "Impact of segments aggregation of TCP Reno flows in optical burst switching networks," in *Proceedingss of INFOCOMM*, 2002, pp. 1803–1812.
[8] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proceedings of ACM SIGCOMM*, 1998.
[9] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno and SACK TCP," in *Proceedings of ACM SIGCOMM*, 1996.
[10] M. Mathis and J. Mahdavi, "Forward acknowledgement: Refining TCP congestion control," in *Proceedings of ACM SIGCOMM*, 1996.
[11] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of TCP/IP with stationary random loss," in *Proceedings of ACM SIGCOMM*, 2000.
[12] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *Computer Communications Review*, vol. 27, no. 3, 1997.
[13] S. Floyd, "Highspeed TCP for large congestion windows," in *Internet draft: draft-floyd-tcp-highspeed-02.txt*, 2002.
[14] F. Paganini, Z. Wang, S. H. Low, and J. C. Doyle, "A new TCP/AQM for stable operation in fast networks," in *Proceedingss of INFOCOMM*, 2003.