SAIMD: A Novel False Congestion Detection Scheme in TCP over OBS Networks

Basem Shihada[1], Pin-Han Ho[1,2], and Qiong Zhang[3]
[1]David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, Canada
[2]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada
{bshihada,pinhan}@bbcr.uwaterloo.ca
[3]Mathematical Science and Applied Computing, Arizona State University, Phoenix, USA
qiong.zhang@asu.edu

Abstract **— Due to the bufferless nature, TCP over Optical Burst Switched (OBS) networks suffer from false congestion detection, where a packet loss event due to random burst contention in a lightly-loaded OBS network may trigger TCP congestion control mechanisms, thereby degrading the performance of TCP. This phenomenon is particularly serious for the TCP flows with very high bandwidth, where returning to slow start due to false congestion detection would critically impair the TCP services and network performance. This paper introduces a novel congestion control scheme for high-bandwidth TCP flows over OBS networks, called Statistical Additive Increase Multiplicative Decrease (SAIMD). SAIMD maintains and analyzes a number of previous round trip time (RTTs) at the TCP senders in order to identify the confidence with which a packet loss event is due to network congestion. The confidence is derived by positioning short-term RTT in the spectrum of long-term historical RTTs. The derived confidence corresponding to the packet loss is then taken in the developed policy for TCP congestion window adjustment. We will show through extensive simulation that the proposed scheme can effectively solve the false congestion detection problem and significantly outperform the conventional TCP counterparts without losing fairness. The advantage gained in our scheme is at the expense of introducing more overhead in the SAIMD TCP senders. Based on the proposed congestion control algorithm, a throughput model is formulated, and is further verified by simulation results.**

Index terms: Optical Burst Switching (OBS), TCP AIMD, Statistical AIMD, burst contention

I INTRODUCTION

Optical Burst Switching (OBS) has shown efficient and dynamic bandwidth allocation for handling the bursty and dynamic nature of the Internet traffic [1, 2]. In OBS networks, a control packet is sent by an ingress node through a predefined physical route prior to launching the corresponding data burst in order to reserve a wavelength channel along each link before the arrival of the data burst. The data burst composed of IP packets cuts through in the optical domain from the ingress node to the egress node. When the data burst passes through an intermediate node, the resource is released either according to the length of the burst (in Just-Enough-Time) or after receiving an explicit notification (in Just-In-Time). These signaling schemes mitigate the need for optical buffering, which is one of the major technical barriers for all-optical networking. Burst losses occur in OBS networks due to the bufferless characteristics, where two or more control packets try to reserve a wavelength channel in overlapped time periods.

Due to the ubiquity of Internet Protocol (IP) and the desire for an integrated IP core carrier in modern communication networks, developing an IP over OBS backbone has attracted much attention from both industry and academia in the past decade [1]. However, the adoption of the IP over OBS overlay has resulted in new challenges where the upper layer protocols such as TCP could be maliciously affected by the underlying OBS bufferless characteristic. This problem is becoming worse in presence of the fact that a burst could contain packets from numerous TCP connections and each of the connections is regulated by individual flow control and congestion control mechanisms.

Conventional TCP implementations, such as TCP Reno [3] and Sack [4], adopt Additive Increase Multiplicative Decrease (AIMD) window-based congestion control. When the launched data gets successfully acknowledged, the TCP sender linearly increases *cwnd* by one packet per round trip time (RTT). The linear growth of *cwnd* continues until either it exceeds the receiver's advertised window (*rwnd*) or a packet is lost. A packet loss is detected either when the packet transmission timeout (TO) threshold is reached or when the TCP sender receives triple-duplicated acknowledgments (TD). The former case indicates severe congestion, hence the size of *cwnd* is set to one segment and the TCP sender enters a slow start phase followed by a congestion avoidance phase. The latter case indicates a light congestion. TCP enters the fast retransmission phase by taking the half of the sender's *cwnd* as the slow start threshold *ssthresh* and setting the *cwnd* to *ssthresh* plus three segments. The TCP sender then retransmits the lost segment and increments its *cwnd* by the segment. After acknowledging the second segment, the *cwnd* is set back to the *ssthresh* [5]. For TCP over IP packet-switched networks, a packet loss can be an effective indication of network congestion, since packet losses are caused by buffer overflow, which is due to network congestion.

When OBS is deployed as the underlying switching technology, the abovementioned TCP congestion control mechanism is subject to a great challenge. In an OBS network, since data bursts cut through pre-configured intermediate core

nodes without being stored, burst contention may happen due to traffic burstiness even when the network load is light. Also, TCP packets along with other protocol packets, such as UDP, could be assembled in a single data burst. Therefore, the delay or loss of a burst in the OBS layer may affect multiple TCP segments from a single or multiple TCP senders. Note that burst delay could take place in the OBS domain due to burst assembly delay, fiber delay lines [6], burst retransmission [7, 8], and burst deflection routing [9], other than network congestion. In such a circumstance, the TCP senders could take an improper reaction and reduce sending rate unnecessarily. The lack of ability in accurately identifying congestion in the OBS domain could significantly downgrade the TCP throughput and leave network resources under-utilized. This problem is particularly critical for the high-bandwidth CP flows, where a false congestion detection event will cause a significant impairment to the applications.

A number of TCP implementations have been proposed for detecting and controlling network congestion in various network environments, including mobile wireless networks [10], ad hoc networks [11], and optical networks [12,13]. Each TCP enhancement has its own design premises, and could be very effective in one circumstance while being much outperformed in another. It has also been proved that jointly considering the characteristics of the whole network environment in the design of the TCP modifications or extensions is necessary [14]. These facts are especially distinguished when TCP is extended to OBS networks and taken to support mission critical applications such as Grid, where multiple TCP segments can be lost when the OBS network is not congested.

In this paper, a novel congestion control mechanism, called Statistical Additive Increase Multiplicative Decrease (SAIMD), is proposed for high-bandwidth TCP flows in the IP over OBS networks. With SAIMD, a TCP sender collects and statistically analyzes a certain number of historical RTTs and dynamically adjusts the multiplicative parameters according to the proposed policy. The proposed scheme only requires the statistical information of RTTs measured at a TCP sender along with some additional computation when a packet loss is encountered, which achieves a clean separation between the TCP and OBS domains. Based on the proposed SAIMD scheme, the throughput performance is modeled, which is further validated through extensive simulation. We will compare the proposed scheme with some of the well-known TCP implementations to verify its efficiency, along with fairness and stability with the coexisting conventional TCP. Since high-bandwidth and fast TCP flows are usually less in number while dominating the bandwidth consumption in the networks, the higher processing overhead caused by SAIMD is a trade-off for the performance gain in having better accuracy in the congestion detection.

The rest of the paper is organized as follows. Section II provides the related work and the background. Section III introduces the proposed SAIMD scheme. In Section IV, we formulate an analytical model for TCP throughput based on the SAIMD scheme. In Section V, the proposed TCP congestion control mechanism is compared with well-known TCP implementations based on AIMD (*1,0.5*) through extensive simulation. Section VI concludes the paper.

II BACKGROUND

In this section, we present the related work on the enhancement and evaluation of TCP over OBS. We also introduce the framework of the Generalized AIMD, which is the basis of the proposed congestion control mechanism.

A. Enhancement and Evaluation of TCP over OBS

A few studies were reported for enhancing the conventional TCP to cope with the false congestion detection problem in OBS networks. The paper [15] investigated TCP false TO detection due to random burst contention under a wide range of traffic load. Three approaches are proposed in this paper in order to avoid false TO detection. The first approach is based on the estimation of the number of TCP segments assembled in a burst without knowing the burst assembly mechanism deployed in the OBS layer. Along with the *cwnd*, a TCP sender also maintains a burst congestion window *burst_wd*. When the TCP sender detects a TO, it first compares its *cwnd* with the *burst_wd*. If the $cwnd \leq burst\_wd$ and $burst\_wd > 3$, then the TCP sender considers this TO as a false TO, hence halfs its *cwnd* and performs fast retransmission for the missing segments. If the $cwnd > burst\_wd$ or $burst\_wd \leq 3$, the TCP sender considers this TO event as a true TO, hence performs the normal TCP retransmission procedure. In the second approach, each TCP segment is acknowledged by a TCP agent located at OBS edge nodes (i.e. ingress nodes). This approach can effectively prevent TCP from false TO detection; however, it violates the end-to-end TCP semantics since ACKs reach TCP senders before the actual completion of segment delivery. The last approach is to maintain a TCP agent at each OBS core node. Whenever a burst is dropped at a core node, the TCP agent disassembles the burst and sends a negative *ack* to the corresponding TCP sender. Therefore, the missing segments and the network congestion state are explicitly informed to TCP senders. The abovementioned solutions could introduce high signaling overhead and may not be practical for handling the Internet traffic. The authors in [7, 8] have proposed to retransmit lost bursts from ingress nodes in order to reduce burst loss probability and to avoid TCP false congestion detection. Also, an analytical model for the throughput of TCP Sack over OBS networks with burst retransmission has been proposed in [16]. However, the burst retransmission approach requires edge nodes to buffer incoming bursts within a time window.

Some schemes have been proposed for improving the performance of TCP over OBS networks. The paper [17] has proposed an adaptive burst assembly algorithm and has investigated the impact of configuration parameters, such as burst assembly time, burst size, and adaptive assembly queue length threshold, on the performance of both TCP and UDP traffic over OBS networks. The authors have shown that the adaptive assembly algorithm can achieve the best TCP performance among all the investigated assembly schemes. In

[18], a TCP throughput analytical model has been proposed in presence of a burst acknowledgment mechanism. The authors have also proposed an error recovery mechanism for electronic buffering at the edge nodes in order to improve the TCP throughput [16]. In [19], a modified TCP decoupling approach was introduced. This approach controls the burst contention probability at the OBS network bottleneck link by taking the advantage of the TCP self-clocking property, where the burst sending rate is controlled through the arrival time of TCP decoupling management packets. Thus the approach avoids unnecessary segment losses and improves the TCP throughput.

Several research works have been reported to evaluate the throughput performance of different TCP flavours over OBS networks. In [20] the authors examined the effect of the burst assembly algorithms at OBS ingress nodes on the performance of TCP Reno. In this paper, TCP flows are classified into three categories: fast, medium, and slow. The fast flows have a sufficiently high IP-access bandwidth that allows the entire transmitted segments (i.e. *cwnd*) to be assembled in a single burst. On the other hand, the medium to slow flows have less number of configurative segments assembled in a single burst. The authors have shown that for medium to slow flows, a performance penalty is introduced on TCP sending rate due to assembly delay, while the aggregation of multiple TCP segments due to burst assembly has given TCP a throughput performance gain. The study in [21] derived a packet-oriented throughput model for TCP Reno, New Reno, and Sack over OBS networks. The study has shown that the three examined TCP flavours have failed to deal with burst losses where each burst contains a large number of TCP segments assembled from a single TCP source. The abovementioned performance degradation of TCP over OBS has also been shown through simulation in [22]. In [23], a fixed-point method based on the TCP feedback mechanism was adopted to accurately compute the TCP throughput in OBS networks.

B. Generalized AIMD congestion control

Generalized AIMD (GAIMD) has been proposed to reduce the saw-like behaviour of TCP for multimedia applications [24, 25]. Instead of increasing the *cwnd* size by one for successful segment delivery and decreasing the *cwnd* size by half for a segment loss event, GAIMD additively increases *cwnd* by α segments when no segment is lost in a single RTT and multiplicatively decreases by β if either a TD or TO segment loss event occurs. In order to ensure TCP friendliness among the competing GAIMD flows, the relations which regulates the values of α and β can be obtained from [24, 25]. Our study focuses on the multiplicative decrease β parameter, while α is kept as 1.

III SAIMD TCP OVER OBS NETWORKS

The SAIMD scheme adopts the framework of GAIMD to enhance the responsiveness of TCP upon any burst loss event that is not caused by congestion. In SAIMD, when a data burst consisting of many TCP segments from single or multiple TCP senders is lost, the corresponding TCP senders will be notified of the segment loss through the receiving of a TD or TO. In either case, instead of halving the *cwnd* or even throttling to slow-start phase, TCP senders reduce the size of *cwnd* by the multiplicative factor β. The factor β is dynamically determined by positioning the short-term RTT statistics in the spectrum of long-term historical RTTs. Here, the "statistics" refers to mean, standard deviation, and correlation function in this study, and will be further detailed as follows.

We introduce two parameters in this scheme, *M* and *N*. The parameter *M* is the number of consecutive RTTs measured for the long-term statistics. *M* should be sufficiently large such that the derived statistics (i.e., the mean and standard deviation) can fully represent the intrinsic characteristics of the network topology, routing policy, and traffic distribution/pattern. The parameter *N* is the number of consecutive RTTs measured prior to a segment loss for the short-term statistics. The average of the *N* RTTs, denoted by *avg_rtt_N*, is compared with the average of the *M* RTTs, denoted by *avg_rtt_M*, in a TCP session, in order to determine how likely the segment loss is due to network congestion or due to random burst contention at a lightly-loaded OBS network. In a segment loss event caused by random burst contention, *avg_rtt_N* is expected to be close to *avg_rtt_M* since the high utilization of network resources remains only a short time period in the *N* RTTs. A larger *avg_rtt_N* can be considered that a segment loss event is more likely due to network congestion rather than random burst contention.

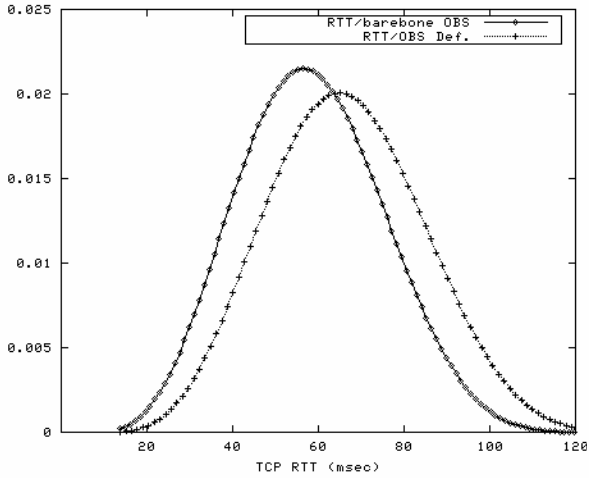

Fig. 1. TCP RTT distribution histogram.

The relationship between *avg_rtt_M* and *avg_rtt_N* is based on the following observations: (1) in TCP over OBS networks, segment losses can be caused by random burst contention in OBS core networks or network congestion along the route of IP access networks and OBS core networks. The difference between random burst contention and network congestion is that network congestion suffers from high resource utilization for a longer period; (2) in the high resource utilization state, the RTT of each segment delivery will be much higher than that in the low-utilization state. This is due to the fact that high-utilization will cause longer queueing delay in IP access networks. Also, in an OBS core network with contention

resolution schemes, such as burst retransmission [7, 8] and deflection [9] schemes, bursts will have a high probability of being retransmitted or deflected, which results in a longer average burst delay in the OBS network.

We further quantify the relation between the long-term and short-term statistics in order to define the confidence with which a segment loss is due to network congestion. We assume that the M consecutive RTTs are random with a mean *avg_rtt_M* and a variance $Var(RTT)$. We also assume that the M consecutive RTTs can be approximately modeled as a Normal distribution. To validate this assumption, we statistically analyze 14,000 TCP consecutive RTTs with a Chi-square test under the following two network scenarios: one is a barebone OBS network, where delay variation takes place in IP access networks and burst assembly at OBS edge nodes, the other scenario is an OBS network with the burst deflection scheme [9], where delay variation takes place in IP access networks, burst assembly at OBS edge nodes, and burst deflection due to a longer route. The *null hypothesis* in the Chi-square test is: "the distribution of the M RTTs cannot be modeled as normal $N(\mu,\sigma)$, where $\mu = avg\_rtt\_M$, and $\sigma = \sqrt{M \cdot Var(RTT)}$". The distribution derived in both network scenarios is shown in Fig. 1. The experiment parameters are given in the numerical analysis section. We found that the null hypothesis can be rejected with 5% confidence, which validates our assumption that the M consecutive RTTs can be approximately modeled as a Normal distribution.

A. Autocorrelation for Determining a Proper Value of N

Selecting a proper value of N is important, since the N RTTs are expected to provide sufficient information about the short-term network status when a segment is lost. If N is chosen too small or too larger, the short-term network status may not be accurately represented.

Our approach in selecting N employs an autocorrelation function:

$$R(0,N) = \frac{1}{N} \cdot \sum_{i=0}^{N} RTT(i) \cdot RTT(i+N),$$

where $RTT(i)$ is the RTT of the *ith* segment. The numbering of RTTs is shown in Fig. 2. The autocorrelation function can reflect how smooth a process is. $R(0,N)$ has the maximum value when N = 0. Also, the stronger correlation a group of RTTs has, a larger value of $R(0,N)$ outcomes [26].

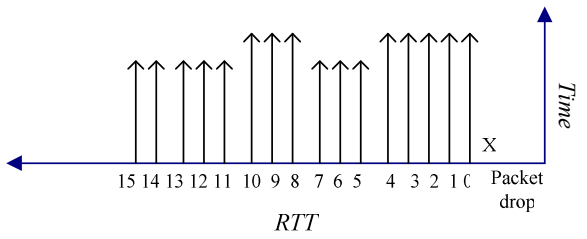

Fig. 2. Numbering of RTTs, where RTT(0) denotes the RTT right before a segment loss occurs.

In our scheme, the value of N is selected such that $R(0,N) = R(0,0) \cdot \gamma\%$, where γ is the threshold that determines N based on the autocorrelation function. In other words, the value of N is selected such that $R(0,N)$ decays from its peak value by no less than γ %.

In order to well-represent the short-term network status, the N RTTs should have a strong correlation with each other. Hence, the value of γ should be close to 1. In our study, γ is taken as 90%.

B. Congestion by SAIMD for TCP over OBS

After a proper N is selected based on the approach in the previous subsection, the value of *avg_rtt_N* can be obtained. Then, we can define the confidence with which the current segment loss event of a TCP session is due to network congestion by positioning *avg_rtt_N* in the Normal distribution spectrum of the M RTTs (shown in Fig. 1). The derived confidence is used to dynamically adjust β at a TCP sender under a developed policy such that β can represent the current network status.

For positioning *avg_rtt_N* in the Normal distribution spectrum, a function $z_i = rtt_{conf}(u_i)$ is defined, where u_i is the confidence level. The $rtt_{conf}(u_i)$ returns a RTT value (denoted by z_i) which is larger than a proportion u_i ($0 < u_i \le 1$) of all RTTs in the Normal distribution curve. A one-to-one mapping between u_i and z_i exists, as shown in the following expression:

$$u_i = cdf(z_i) = \sum_{j=0}^{i} pmf(z_j).$$

The $cdf(z_i)$ and $pmf(z_i)$ denote the cumulative density function (CDF) and probability mass function (PMF) in the RTT spectrum given the RTT value of z_i [26]. The one-to-one mapping between u_i and z_i is shown in Fig. 3. In this figure, for example, if the RTT is higher than the mean RTT with u_i = 90% confidence, then the RTT value of z_i is in the range of 100 to 120 ms.

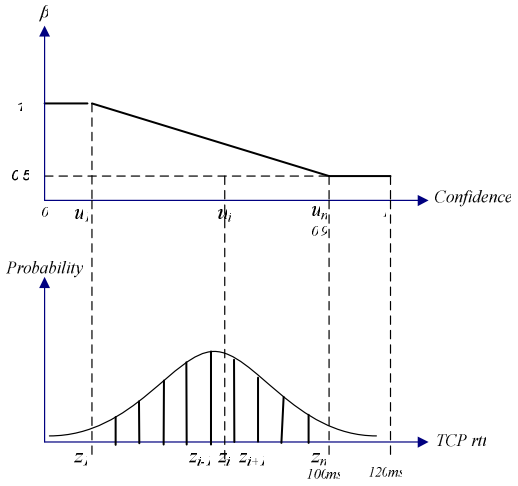

Fig. 3. The relation of z_i, u_i, and β in the SAIMD scheme.

The proposed policy for adjusting β is as follows. When *avg_rtt_N* is smaller than $z_1 = rtt_{conf}(u_1)$, a low confidence of network congestion is indicated, which yields no adjustment of the *cwnd* in response to a segment loss, i.e., β = 1. When $avg\_rtt\_N > z_n = rtt_{conf}(u_n)$ $(u_n > u_1)$, it is a strong indication of network congestion. Hence, the TCP sender cuts the *cwnd* by half, i.e., β = 0.5 in response to a segment loss. When *avg_rtt_N* falls in the interval $[z_1, z_n]$, $\beta = f(u_i)$, where $f(u_i) = 1 - \frac{u_i - u_1}{2(u_n - u_1)}$. Note that u_1 and u_n are two parameters given in advance in order to distinguish network congestion and random burst contention at a lightly-loaded OBS network. In this study, u_1 and u_n are set to 50% and 90%, respectively. The policy-based *cwnd* adjustment scheme can be summarized in the following equation:

$$\beta = \begin{cases} 0.5 & avg\_rtt\_N > rtt_{conf}(u_n) \\ f(u_i) & rtt_{conf}(u_n) > avg\_rtt\_N > rtt_{conf}(u_1) \\ 1 & avg\_rtt\_N \le rtt_{conf}(u_1) \end{cases} \quad (1)$$

The dynamic adjustment of β based on the confidence level u_i is also illustrated in Fig. 3. The flow chart for the proposed SAIMD scheme is shown in Fig. 4.

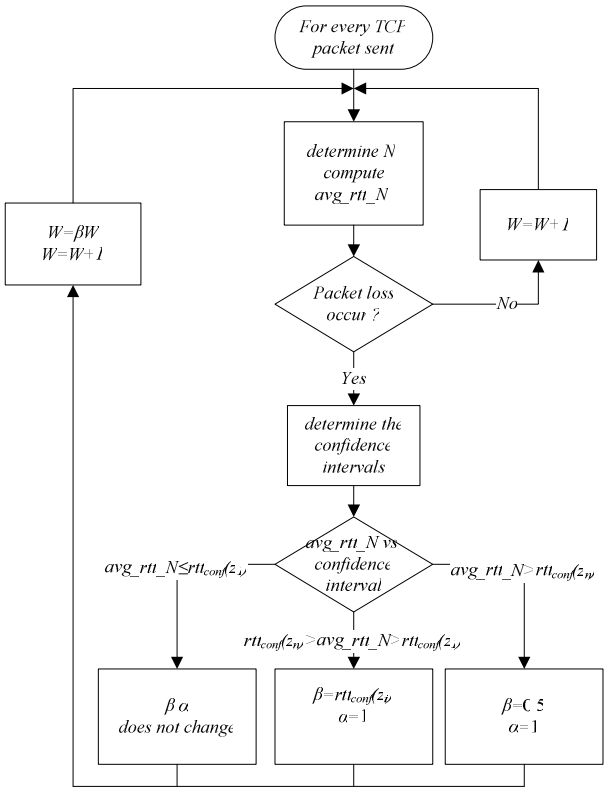

Fig. 4. The proposed SAIMD congestion control scheme.

We now discuss two extreme cases for the SAIMD scheme. Note that the adjustment of β will only be trigged if a TCP sender detects a TD segment loss. The first extreme case is that a TCP sender starts the data transmission while the network is congested. In this case, the measured RTTs are large at the beginning of the TCP session and the *avg_rtt_N* and *avg_rtt_M* obtained by the TCP sender are very close. Hence, β will be close to 1. For a TD segment loss, the size of the *cwnd* will not be reduced enough. The SAIMD scheme will then cause persistent congestion in the network and TO segment losses will occur. The TCP sender then enters a slow start phase and sets the size of the *cwnd* to be 1. As a result, the network congestion will be possibly relieved. The second extreme case is when there is no RTT variation in a network, which is rare in the real situation. In this case, the *avg_rtt_N* and *avg_rtt_M* obtained by the TCP sender are also very close. As a result, the *cwnd* will not be reduced enough as a response to collecting TD. Instead, the TCP flow will timeout as a response to persistent congestion.

The SAIMD scheme is particularly suitable for the high bandwidth TCP flows that operate for a relatively long period of time and a large *cwnd*. These high-bandwidth TCP flows are expected to take an important role in some mission-critical applications such as Grid. Depending on the number of TCP segments assembled in a contended burst, these flows may either trigger a TO or cut the *cwnd* to half as a response to the receiving of TDs. Once there is a false congestion detection event which leads to TO or TD, the time required for increasing the *cwnd* (most likely through an additive increase) to its previous size could be very long, which downgrades the TCP performance and impairs the desired application scenario.

Compared with the conventional AIMD based TCP scheme, the SAIMD causes additional overhead for maintaining the *M* RTTs along with the efforts in computing the autocorrelation and confidence intervals for the *N* RTTs. The cost is nonetheless a trade-off with the long convergence time in recovery from slow-start caused by false congestion detection. This is considered with essential importance for those high-bandwidth TCP flows which may take hours or days to recover from a slow-start. Note that the computation for the autocorrelation and confidence interval is required only when a segment loss event occurs, and the computation complexity is almost a constant regardless of *M* and *N*. In addition, the proposed SAIMD scheme is mainly for the long and high-bandwidth TCP flows instead of short TCP such as HTTP web services; thus, the resultant additional overhead to the whole network is expected to be trivial.

IV PERFORMANCE ANALYSIS

In this section, we analyze the throughput of the SAIMD fast flows in an OBS network. We have selected TCP Sack [4] for our analytical model since it has been widely deployed in the current operating systems and has the best throughput performance over OBS networks compared to TCP Reno and New Reno [15],[21]. The following table lists the notations used in the analytical model.

p	:	segment loss probability
b	:	number of segments that are acknowledged by receiving an *ack*
B	:	TCP throughput
H	:	number of segments transmitted during TO
$\overline{RTT}$	:	average round trip time
TOP	:	timeout period
TDP	:	triple duplicate period
RTO	:	retransmission timeout
Z^{TO}	:	duration of a sequence of TOs
X	:	number of successful rounds in a TDP
Y	:	number of segments sent before TD or TO expiration
W	:	current congestion window size in segments
W_m		TCP maximum window size
S	:	number of segments belonging to a single TCP flow being assembled in the current burst
Q	:	ratio between the probability of TO loss and TD loss

In our model, we define a round as when the TCP sender emits the current *cwnd* (in segments) and waits until either it receives an acknowledgement or the TO expires. We also define a TD loss as a segment loss detected by triple duplicates and define a TO loss as a segment loss detected after TCP sender timeouts.

We obtain the Statistical AIMD TCP Sack throughput in OBS network for both TD and TO losses as follows:

$$B_S = \frac{E[Y] + Q \times E[H]}{E[TDP] + Q \times E[TOP]} \tag{2}$$

In the following two sections, we will derive $E[Y]$, $E[TDP]$, $E[TOP]$, $E[H]$, and Q in the presents of TD and TO losses respectively.

C. Triple Duplicate (TD) Losses

As per the model in [8], suppose that the (c_i+1)th burst is the first burst lost in the ith TDP, TDP_i, which contains the first (a_i+1)th lost segment in the TDP_i. As shown in Fig. 5, h_i additional segments will be sent in the same round after the (c_i+1)th burst is sent and lost. After receiving TD, the TCP sender retransmits all the missing segments contained in the lost burst in the next round. Therefore, in the next round, $W_{X_i} - S$ new segments will be sent, where W_{X_i} is the *cwnd* size in the X_i th round in the TDP_i. After recovering all the segments lost in the burst, a new round TDP_{i+1} starts with the *cwnd* being cut by a factor of β. The total number of segments successfully transmitted during the TDP_i is $Y_i = a_i + h_i + W_{X_i} - S$. $E[h]$ is approximately equal to $E[\beta]E[W_X]$, since $0 \le h_i \le W_{X_i}$ and the *cwnd* is reduced by β for every TD loss. Thus, we have

$$E[Y] = E[a] + (E[\beta] + 1)E[W_X] - S \tag{3}$$

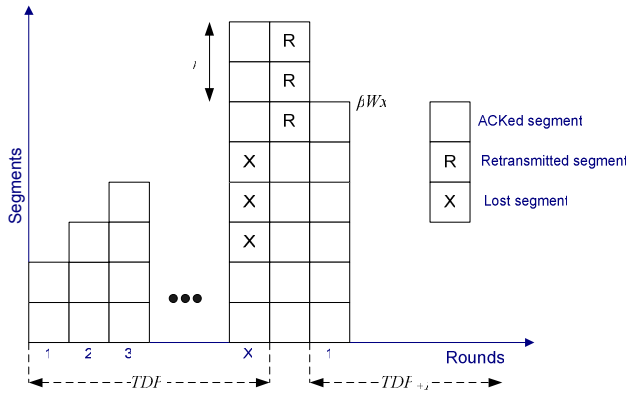

Fig. 5. Evolution of SAIMD TCP Sack congestion window over OBS networks.

As per Eq. (1), β is a function of the *avg_rtt_N* and the confidence level u_i. Considering Fig. 3, we derive $E[\beta]$ as follows,

$$\begin{aligned} E[\beta] = \bar{\beta} &= cdf(z_1) + \sum_{i=1}^{n} f(z_i) \cdot pmf(z_i) + \frac{1 - cdf(z_n)}{2} \\ &= u_1 + \sum_{i=1}^{n} \left[1 - \frac{u_i - u_1}{2(u_n - u_1)} \right] \cdot (u_i - u_{i-1}) + \frac{1 - cdf(z_n)}{2} \\ &= u_1 + \frac{1 - u_n}{2} + \sum_{i=1}^{n} \frac{2u_n - u_1}{2(u_n - u_1)} (u_i - u_{i-1}) + \sum_{i=1}^{n} \frac{u_i (u_i - u_{i-1})}{2(u_n - u_1)} \end{aligned} \tag{4}$$

where $cdf(z_i)$ and $pmf(z_i)$ denote the cumulative density function (CDF) and probability mass function (PMF) in the RTT spectrum given the RTT value of z_i [26]. An alternative way for solving $E[\beta]$ can be through historical collection of the β values, which yields:

$$E[\beta] = \bar{\beta} = \sum_i \beta_i p_\beta(\beta_i) \tag{5}$$

where $p_\beta(\beta_i)$ is the probability of the distinct values β_i to exist.

In order to derive $E[\alpha]$, we consider a random process $\{c_i\}$, which is the average number of bursts sent in the TDP_i till the first burst loss. Assume that burst contentions in OBS networks occur independently. The probability of $c = k$ (or the case where k–1 bursts are successfully delivered before a burst loss is encountered) can be written as:

$$P[c = k] = (1 - p)^{k-1} \cdot p \tag{6}$$

Given that $a_i = Sc_i$ we have,

$$E[a] = S\,E[c] = S \sum_{k=1}^{\infty} k(1-p)^{k-1} p = \frac{S}{p} \tag{7}$$

By substituting Eq. (7) into Eq. (3), we have

$$E[Y] = (\bar{\beta} + 1)E[W_X] + \frac{1 - p}{p} S \tag{8}$$

1) For high segment losses ($W_X < W_m$)

In the presents of a high segment loss probability, the *cwnd* will remain less than the maximum size W_m. Recall that b denotes the number of segments that are acknowledged by receiving an *ack*. During the TDP_i, the *cwnd* increases between $\beta W_{X_{i-1}}$ and W_{X_i}. Since the increase of the *cwnd* is linear with slop 1/b, thus,

$$W_{X_i} = \beta W_{X_{i-1}} + \frac{X_i}{b} \tag{9}$$

By reversing Eq. (9), we have

$$E[X] = b(1-\bar{\beta})E[W_X] \tag{10}$$

Since Y_i can be derived by summarizing the number of segments sent in X_i successful rounds and the additional $(W_{X_i} - S)$ segments in the next round of X_i as shown in Fig. 5, we have:

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

By substituting Eq. (9), we have

$$Y_i = \frac{X_i}{2}(\beta W_{X_{i-1}} + W_{Xi} - 1) + W_{X_i} - S$$

By assuming zero correlation between β and W_X, after substituting Eq. (10), we get

$$E[Y] = \frac{b(1-\bar{\beta}^2)E[W_X]^2 - b(1-\bar{\beta})E[W_X]}{2} + E[W_X] - S \tag{11}$$

By combining Eq. (11) and Eq. (8), we have,

$$\frac{b(1-\bar{\beta}^2)}{2}E[W_X]^2 + (\frac{b\bar{\beta}}{2} - \frac{b}{2} - \bar{\beta})E[W_X] - \frac{S}{p} = 0$$

$E[W_X]$ can be then obtained as

$$E[W_X] = \frac{\bar{\beta} - \frac{b(\bar{\beta}-1)}{2} + \sqrt{(\frac{b(\bar{\beta}-1)}{2} - \bar{\beta})^2 + \frac{2Sb(1-\bar{\beta}^2)}{p}}}{b(1-\bar{\beta}^2)} \tag{12}$$

By substituting Eq. (12) into Eq. (8), we obtain $E[Y]$ as:

$$E[Y] = \frac{\bar{\beta} - \frac{b(\bar{\beta}-1)}{2} + \sqrt{(\frac{b(\bar{\beta}-1)}{2} - \bar{\beta})^2 + \frac{2Sb(1-\bar{\beta}^2)}{p}}}{b(1-\bar{\beta})} + \frac{1-p}{p}S \tag{13}$$

Also, by substituting Eq. (12) into Eq. (10), we obtain $E[X]$ as,

$$E[X] = \frac{\bar{\beta} - \frac{b(\bar{\beta}-1)}{2} + \sqrt{(\frac{b(\bar{\beta}-1)}{2} - \bar{\beta})^2 + \frac{2Sb(1-\bar{\beta}^2)}{p}}}{1+\bar{\beta}} \tag{14}$$

$E[TDP]$ is then obtained as

$$E[TDP] = \overline{RTT}(E[X]+1)$$
$$= \overline{RTT}(\frac{2\bar{\beta} - \frac{b(\bar{\beta}-1)}{2} + \sqrt{(\frac{b(\bar{\beta}-1)}{2} - \bar{\beta})^2 + \frac{2Sb(1-\bar{\beta}^2)}{p}}}{1+\bar{\beta}} + 1) \tag{15}$$

2) For low burst losses ($W_X = W_m$)

For a very low burst loss probability, the *cwnd* size will most likely remain to be the maximum *cwnd* size, W_m, before a burst loss event occurs. From Eq. (8) we can obtain,

$$E[Y] = (\bar{\beta}+1)W_m + \frac{1-p}{p}S \tag{16}$$

During each TDP, the *cwnd* size linearly increases from βW_m to W_m for $(W_m - \beta W_m)$ rounds and then stays at W_m for $(X_i - (W_m - \beta W_m))$ rounds, hence we can obtain the number of segments that are transmitted before a TD loss as $\frac{(W_m - \beta W_m)^2}{2} + W_m(X_i - W_m + \beta W_m) - h_i$. On the other hand, from Eq. (7), the total number of segments that are successfully transmitted before a segment loss is S/p. Hence we have

$$\frac{(W_m - \beta W_m)^2}{2} + W_m(X_i - W_m + \beta W_m) - h_i = \frac{S}{p} \tag{17}$$

By reversing Eq. (17), we can obtain $E[X]$ as

$$E[X] = W_m(1-\bar{\beta}) - \frac{W_m(1-2\bar{\beta}+\bar{\beta}^2)}{2} + \bar{\beta} + \frac{S}{W_m p} \tag{18}$$

The duration of the *TDP* is obtained as,

$$E[TDP] = \overline{RTT}(E[X]+1)$$
$$= \overline{RTT}(W_m(1-\bar{\beta}) - \frac{W_m(1-2\bar{\beta}+\bar{\beta}^2)}{2} + \bar{\beta} + \frac{S}{W_m p} + 1) \tag{19}$$

D. Timeout (TO) Losses

The behavior of TCP SAIMD for a TO loss is same as that of TCP Sack. Hence, the analysis of TO losses is same as the analysis in [8]. From [8], we have

$$E[H] = E[R] - 1 = \frac{p}{1-p}, \tag{20}$$

$$E[TOP] = RTO\frac{f(p)}{1-p}, \tag{21}$$

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

$$Q(E[W_X]) \approx p^{\frac{W_X}{S}-1}. \tag{22}$$

E. SAIMD TCP SACK over OBS Throughput Estimation

In the case of $W_X < W_m$, we can obtain the SAIMD throughput by substituting Eqs. (13), (15), (20), (21), and (22) into Eq. (2), which yields

$$B_S = \frac{\frac{\bar{\beta}-\frac{b(\bar{\beta}-1)}{2}+\sqrt{(\frac{b(\bar{\beta}-1)}{2}-\bar{\beta})^2+\frac{2Sb(1-\bar{\beta}^2)}{p}}}{b(1-\bar{\beta})}+\frac{1-p}{p}S+\frac{p^{\frac{W_X}{S}}}{(1-p)}}{\overline{RTT}(\frac{2\bar{\beta}-\frac{b(\bar{\beta}-1)}{2}+\sqrt{(\frac{b(\bar{\beta}-1)}{2}-\bar{\beta})^2+\frac{2Sb(1-\bar{\beta}^2)}{p}}}{1+\bar{\beta}}+1)+p^{\frac{W_X}{S}-1}RTO\frac{f(p)}{1-p}} \quad (23)$$

In the case of $W_X = W_m$, TCP SAIMD throughput can be obtained by substituting Eqs. (16), (19), (20), (21), and (22) into Eq. (2), which yields

$$B_S = \frac{(\bar{\beta}+1)W_m+\frac{(1-p)S}{p}+\frac{p^{\frac{W_m}{S}+1}}{1-p}}{\overline{RTT}(W_m(1-\bar{\beta})-\frac{W_m(1-2\bar{\beta}+\bar{\beta}^2)}{2}+\bar{\beta}+\frac{S}{W_m p}+1)+p^{\frac{W_m}{S}-1}RTO\frac{f(p)}{1-p}} \quad (24)$$

V NUMERICAL RESULTS

To verify the proposed statistical AIMD scheme, simulation is conducted using *NS-2*, where the NSF network topology shown in Fig. 6 is adopted as the OBS core network. The distance presented along each link is in *kilometres*, which corresponds to a bi-directional control channel and a fiber link for data burst transfer. The link consists of 8 wavelengths operating at 10 *Gbps* transmission rate. Each OBS core node is equipped with three fiber delay lines (FDL) operating at 2, 5, and 8 *ms* delay granularities, which can delay 3 data bursts simultaneously. The burst offset time is set to 4μs. The mixed time/length based burst assembly algorithm is adopted, where the burst timeout threshold is 5*ms* and the maximum burst length is 50*KB*. The core nodes implement the LAUC-VF channel scheduling algorithm. The File Transfer Protocol (FTP) application is used for generating TCP traffic. The maximum congestion window size of a TCP flow is 128 segments, and each segment has the size of 1*KB*. The TCP throughput is obtained over a simulation period of 10^4 *seconds*.

The TCP senders and receivers are attached to the OBS edge nodes. Burst losses occur at the OBS core network due to burst contention. Burst retransmission and deflection routing have been implemented. In our simulation, we examined the TCP throughput over barebone OBS, OBS with burst retransmission, and OBS with burst defection routing. If the contented burst fails to deliver after the second retransmission attempt, the burst will be dropped. Similarly, if the deflected burst fails to transmit after reaching the second node, it will be dropped. In the simulation, RTT increases due to the buffering delay at the edge nodes, the burst assembly delay, the deployment of the burst retransmission and deflection.

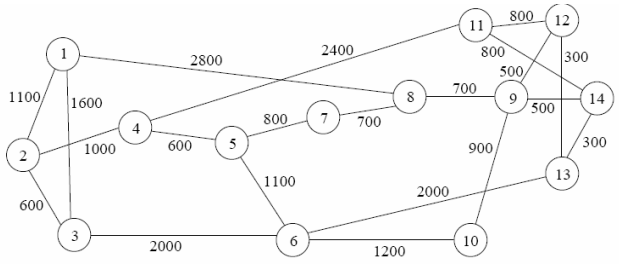

Fig. 6. NSF network topology adopted in the study.

A. SAIMD Throughput Performance

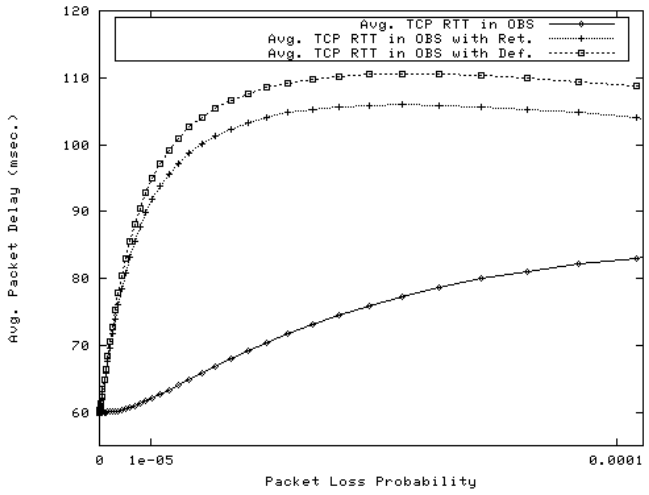

Fig. 7. Average TCP RTT vs. segment loss probability.

In Fig. 7 we show the relationship between the average of TCP RTT versus the segment loss probability. It is notable that a significant amount of extra delay was introduced to the retransmitted and deflected bursts.

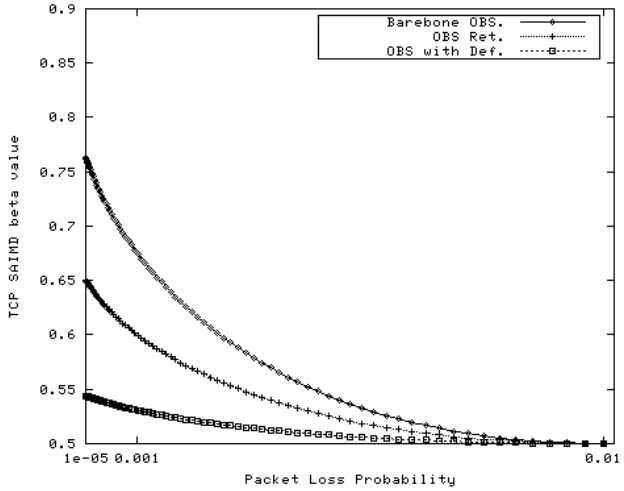

Fig. 8. Segment loss probability vs. the average values of β in the SAIMD scheme.

In Fig. 8 we show the average values of β in the SAIMD scheme at various segment loss probabilities. We can see that with higher segment loss probability, the average value of β decreases. When the segment loss probability is close to 0.01, the average values of β are close to 0.5. This is due to the fact that a higher segment loss probability results in a higher confidence of congestion.

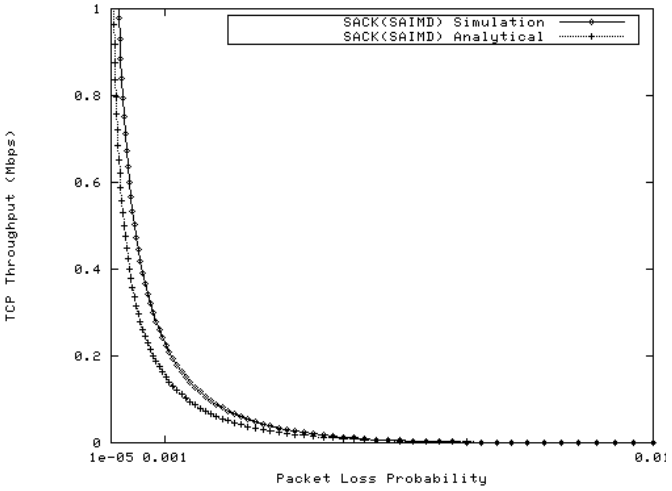

Fig. 9. SAIMD throughput from the simulation model vs. the analytical model.

Fig. 9 compares the results from the analytical model and the simulation model. We can see that the throughput performance from the two approaches match, which validates our analytical model.

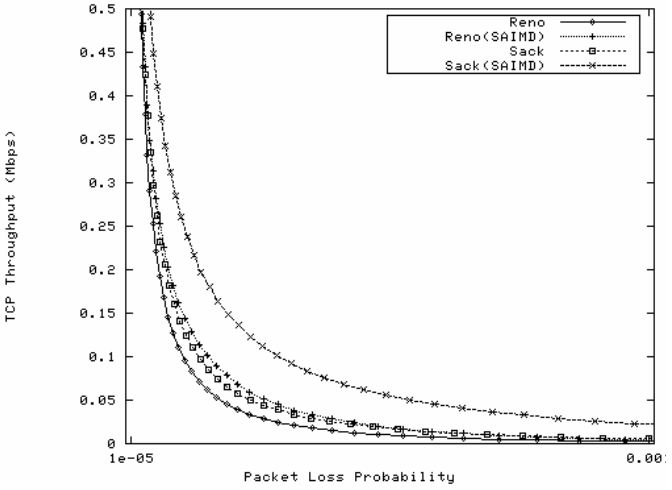

Fig. 10. Throughput of TCP Reno/Reno(SAIMD), TCP Sack/Sack(SAIMD) in the barebone OBS network.

Fig. 10 shows the simulation results for the throughput of Sack/Sack-SAIMD and Reno/Reno-SAIMD flows in the barebone OBS. We can see that the throughput from the conventional TCP Sack and Reno is much lower than that from the SAIMD scheme when the segment loss probability is low. This is because the AIMD (1, 0.5) senders always unnecessarily halve the *cwnd* for a segment loss event at low traffic loads. On the other hand, the SAIMD Sack senders have achieved up to 37% throughput improvement compared to the AIMD senders since it does not rigidly react to a segment loss at low traffic loads. Instead, the factor β is adjusted at each SAIMD sender to guarantee a smaller *cwnd* reduction in response to a segment loss at low traffic loads, which correctly reacts to the segment losses due to random burst contention.

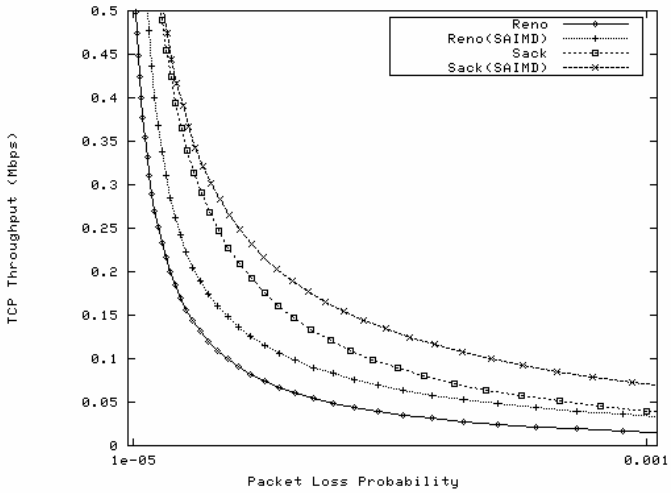

Fig. 11. Throughput of TCP Reno/Reno(SAIMD), TCP Sack/Sack(SAIMD) in OBS with burst retransmission.

Fig. 11 shows the simulation results for the throughput of TCP Sack/Sack-SAIMD, Reno/Reno-SAIMD, while enabling burst retransmission as a contention resolution scheme in the OBS network. Retransmission in the OBS domain has been reported to be able to reduce the overall segment loss probability at the expense of longer RTT (as shown in Fig. 7). By integrating the burst retransmission scheme with the SAIMD scheme, we can observe up to 81% improvement in TCP Sack and Reno throughput.

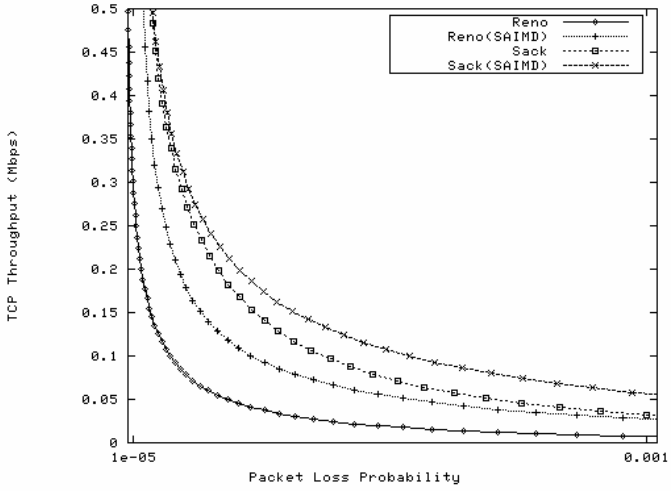

Fig. 12. Throughput of TCP Reno/Reno(SAIMD), TCP Sack/Sack(SAIMD) in the OBS network with burst deflection.

Fig. 12 shows the simulation results when we enable deflection routing in the OBS network. The results show that the SAIMD scheme can improve the TCP Sack and Reno throughput by up to 72%.

B. SAIMD Fairness

In this section, the fairness among Reno, Sack, Reno (SAIMD), and Sack (SAIMD) is examined. For this purpose, we use *Jain's fairness index* which is defined as $(\sum_{i=1}^{n} B_i)^2 \Big/ n\sum_{i=1}^{n} B_i^2$, where n is the number of competing flows and B_i is the throughput of the i^{th} flow. The competing flows share the same source-destination pairs.

Fig. 13 shows the fairness index of TCP flows by Reno, Sack, and Reno (SAIMD), and Sack (SAIMD), over a barebone OBS network. We can observe that the SAIMD has a much better fairness index compared with that by the traditional AIMD of Reno and Sack. This is due to the fact that the SAIMD congestion control mechanism has successfully and accurately identified the burst contention from the congestion, which better assists the SAIMD flows to remain close to the equilibrium.

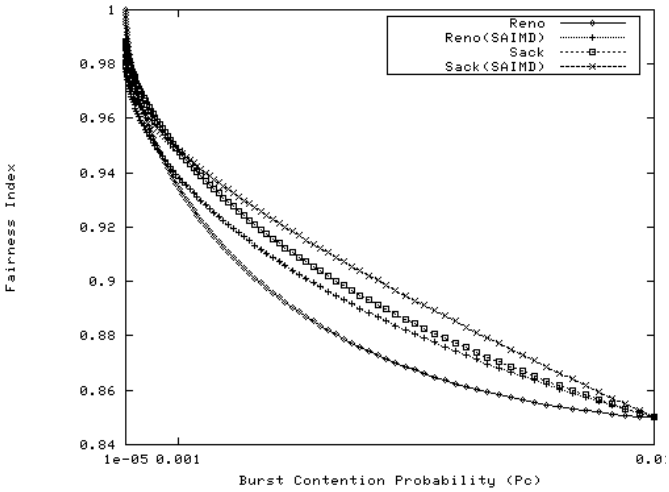

Fig. 13. Fairness Index of Reno, Sack, Reno (SAIMD), and Sack (SAIMD) in a barebone OBS network.

The fairness index has also been examined while enabling the burst retransmission mechanism for each scheme, which is shown in Fig. 14. Note that burst retransmission has been identified as an effective approach for enhancing the overall network throughput by hiding the burst loss events in the OBS domain from the TCP senders. The simulation results demonstrate that the SAIMD fairness with burst retransmission is still better than SACK and Reno. The above two experiments proved that the proposed SAIMD mechanism can maintain a friendly relation with the other TCPs.

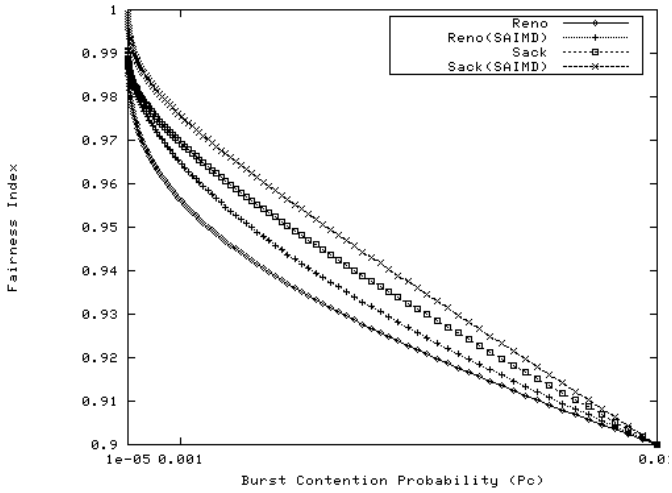

Fig. 14. Fairness Index of Reno, Sack, Reno (SAIMD), and Sack (SAIMD) in OBS with burst retransmission.

In summary, it is observed that SAIMD can effectively solve the false congestion detection problem by hiding the non-congestion losses in the OBS domain from the TCP senders while maintaining the network stability. The derivation of the β value based on the given confidnece interval is considered as a one-step advancement toward the next-generation accurate congestion prediction framework in the TCP flavor design with various and heterogeneous transmission medias.

VI CONCLUSIONS

The paper introduced a novel Statistical Additive Increase Multiplication Decrease (SAIMD) framework for TCP congestion control in the carrier networks supported by the Optical Burst Switching (OBS) technology. The proposed scheme aims to resolve the vicious effect of TCP false congestion due to the bufferless characteristic in the OBS domain. The proposed scheme collects and analyzes the historical RTTs and adjusts β according to the statistics of the collected RTTs at the occurrence of any segment loss event. Analysis was conducted to evaluate the TCP throughput using the proposed scheme. Simulations were conducted to validate the proposed TCP throughput model and to evaluate the proposed congestion control mechanism by comparing it with conventional AIMD based TCP Reno and Sack under different network scenarios, such as OBS networks with burst retransmission or burst deflection routing. Simulation results showed that the proposed SAIMD mechanism can significantly outperform the conventional TCP implementations. We conclude that the superior of the proposed SAIMD scheme comes from the better understanding on the underlying burst transmission behaviour through the analysis of collected RTT information. The merits gained by SAIMD are particularly beneficial to the high-bandwidth and fast TCP flows, in which a false congestion detection event caused by burst contention could lead to serious impairment on the TCP performance.

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, 1999.

[2] Y. Xiong, M. Vandenhoute, and H. Cankaya, "Control architecture in optical burst-switched WDM networks,' IEEE Journal on Selected Areas in Communications (JSAC), vol. 18, pp. 1838-51, 2000.

[3] S. Floyd, "Quick-Start for TCP and IP," Internet draft, draft-amit-quick-start-02.txt, 2002.

[4] J. Mahdavi, S. Floyd, and A. Romanow, "TCP selective acknowledgement options. RFC 2018, 1996.

[5] W. Stevens, "TCP/IP illustrated, Volume 1 – the protocols", 1994.

[6] I. Chlamtac, A. Fumagalli, L. G. Kazovsky, and et al., "CORD: contention resolution by delay lines," IEEE Journal on Selected Areas in Communications, vol. 14, no. 5, pp. 1014–1029, June 1996.

[7] Q. Zhang, V. Vokkarane, Y. Wang, and J. Jue, "Analysis of TCP over optical burst switched networks with burst retransmission," in the proceedings of IEEE Globecom, 2005.

[8] Q. Zhang, V. Vokkarane, Y. Wang, and J. Jue, "TCP over optical burst-switched networks with optical burst retransmission," submitted to IEEE Journal on Selected Areas in Communications (JSAC), 2005.

[9] C.-F. Hsu, T.-L. Liu, and N.-F. Huang, "Performance analysis of deflection routing in optical burst switching networks," in the proceedings of in IEEE Infocom, 2002.

[10] X. Chen, H. Zhai, J. Wang, and Y. Fang, "A Survey on Improving TCP Performance over Wireless Networks," in: Resource Management in Wireless Networking vol. 16, pp. 657-695, Kluwer Academic Publishers/Springer, 2005

[11] X. Chen, H. Zhai, J. Wang, and Y. Fang, "TCP Performance over Mobile Ad Hoc Networks", Canadian Journal of Electrical and Computer Engineering (CJECE) (Special Issue on Advances in Wireless Communications and Networking), vol. 29, no. 1/2, pp. 129-134, 2004

[12] C. Jin, D. Wei, and S. Low, "FAST TCP: motivation, architecture, algorithms, performance," Proceedings, IEEE Infocom, 2004.

[13] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (BIC) for fast long-distance networks", Proceedings, IEEE Infocom, 2004.

[14] C. Barakat, E. Alman, and W. Dabbous, "On TCP performance in a heterogenous network: a survey", IEEE Communication Magazine, vol. 38 no. 1 pp. 40-46, 2000.

[15] X. Yu, C. Qiao, and Y. Liu, "TCP implementation and false time out detection in OBS networks," in the proceedings of IEEE Infocom, 2004.

[16] S. Yao, F. Xue, B. Mukherjee, S. J. B. Yoo, and S. Dixit, "Electrical ingress buffering and traffic aggregation for optical packet switching and their effect on TCP-level performance in optical mesh networks", IEEE Communications Magazine, vol. 40, no. 9, pp. 66 –72, 2002.

[17] X. Cao, J. Li, Y. Chen, and C. Qiao, "Assembling TCP/IP packets in optical burst switched networks", in the proceedings, IEEE Globecom 2002.

[18] P. Du, S. Abe, "TCP performance analysis of optical burst switching networks with burst acknowledgment mechanism", in the proceedings of APCC, 2004.

[19] S.Y. Wang, "Using TCP congestion control to improve the performances of optical burst switched networks," in the proceedings of IEEE ICC, 2003.

[20] A. Detti and M. Listanti," Impact of segment agreegation on TCP Reno flows in optical burst switching networks, " in the proeccedings of IEEE Infocom, 2002.

[21] X. Yu, C. Qiao, Y. Liu, and D. Towsley, "Performance evaluation of TCP implemenations in OBS networks," Technical Report, 2003-13, the State University of New York at Buffalo, 2003.

[22] Gowda, R. K. Shenai, K. M. Sivalingam, and H. C. Cankaya, "Performance evaluation of TCP over optical burst-switched (OBS) WDM networks", in the proceedings of IEEE ICC, 2003.

[23] C. Cameron, J. Choiy, S. Bilgrami, et al, "Fixed-Point performance analysis of TCP over optical burst switched networks", in the proceedings of ATNAC, 2004.

[24] Y. Yang, S. Lam, "Generalized AIMD congestion control," University of Texas, Technical Report TR-2000, available at http://www.cs.utexas.edu/users/lam/NRL/TechReports/

[25] L. Cai, X. Shen, J. Pan, and J. W. Mark, "Performance analysis of TCP-friendly AIMD algorithms for multimedia applications", IEEE Transaction on Multimedia, vol. 7, no. 2, pp. 339-355, 2005.

[26] J. A. Gubner, " Probability and random processes for electrical and computer engineers", Cambridge University Press, pp 240-262, 2006