A Novel Congestion Detection Scheme in TCP over OBS Networks

Basem Shihada, Pin-Han Ho, and Qiong Zhang

Abstract — This paper introduces a novel congestion detection 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 advantages gained in our scheme are 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 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 (with Just-Enough-Time signaling) or after receiving an explicit notification (with Just-In-Time signaling). These signaling schemes mitigate the need for optical buffering, which is one of the major technical barriers for all-optical networking. Burst contentions 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 the 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 architecture has resulted in new challenges. In this case, the upper layer protocols, such as TCP, could fail to function in desired manners due to the underlying OBS bufferless characteristics. This problem can become worse since a burst could contain TCP packets from numerous different TCP connections, where each of the connections is regulated by individual flow control and congestion control mechanism.

Conventional TCP implementations, such as TCP Reno [3] and Sack [4], adopt Additive Increase Multiplicative Decrease (AIMD) window-based congestion control, where a packet loss event can be an effective indication of network congestion. When OBS is deployed as the underlying switching technology, the conventional TCP congestion control mechanism may behave in an undesired manner. 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. The end-to-end delay of bursts may increase due to the deployment of OBS contention resolution schemes, such as fiber delay lines [6], burst retransmission [7, 8], and burst deflection routing [9]. Burst losses could also occur in a lightly-loaded OBS network if the burst contentions could not be resolved. The delay or loss of a burst in the OBS layer may affect one or multiple TCP senders, if the TCP packets from those senders are assembled in a single burst. 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 TCP flows, since a false congestion detection event may significantly reduce the cwnd size, requiring a long time for the cwnd recovery.

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 runs over OBS networks and supports mission-critical applications, such as Grid.

In this paper, we propose a novel congestion control mechanism, called Statistical AIMD (SAIMD), 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. SAIMD only requires the statistical information of RTTs measured at a TCP sender along with some additional computation when packet loss is encountered, which achieves a clean separation between the TCP and OBS domains. We also model the throughput performance of SAIMD, which is further validated through extensive simulation. The proposed scheme is then compared with the existing TCP implementations to verify its efficiency and fairness.
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 some well-known TCP implementations through extensive simulation. Section VI concludes the paper.

II BACKGROUND

A few studies have been reported for enhancing the conventional TCP to cope with the false congestion detection problem in OBS networks. The study in [15] investigated TCP false Time Out (TO) detection due to random burst contention under a wide range of traffic load. Several approaches were proposed for avoiding false TO detection, which are either based on the estimation of the number of TCP packets assembled in a burst or based on the information received from the OBS network. The authors in [7, 8] proposed the retransmission of lost bursts at ingress nodes in order to reduce burst loss probability and to avoid TCP false congestion detection, where edge nodes are required to buffer incoming bursts within a time window.

A Threshold-based TCP Vegas (T-Vegas) [16] was proposed for TCP identifying network congestion in OBS networks. It was shown that the T-Vegas can effectively detect network congestion. But additional network information may be needed for pre-configuring the optimal parameters, such as the threshold. Some schemes have been proposed for improving the performance of TCP over OBS networks. The paper [17] proposed an adaptive burst assembly algorithm and investigated the impact of configuration parameters on the performance of both TCP and UDP traffic over OBS networks. In [18], a TCP throughput analytical model was proposed in the presence of a burst acknowledgment mechanism. In [19], a modified TCP decoupling approach was introduced 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 packet losses and improves the TCP throughput.

Generalized AIMD (GAIMD) was proposed to reduce the saw-like behaviour of TCP for multimedia applications [20]. Instead of increasing the cwnd size by one for successful packet delivery and cutting the cwnd size by half for a packet loss event, GAIMD additively increases cwnd by α packets when no packet is lost in a single RTT, while multiplicatively decreasing by β if either a TD or TO packet loss event occurs. TCP friendliness is ensured among competing GAIMD flows by regulating the values of α and β [20]. In our study, we take α as 1 while β as a variable subject to manipulation.

III SAIMD TCP OVER OBS NETWORKS

A. Scheme Overview

The proposed 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 packets from single or multiple TCP senders is lost, the corresponding TCP senders will be notified of the packet 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 our 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 packet loss for the short-term statistics. The average of the N RTTs, denoted by \( \text{avg}_N \), is compared with the average of the M RTTs, denoted by \( \text{avg}_M \), in a TCP session, in order to determine how likely the packet loss is due to network congestion or due to random burst contention at a lightly-loaded OBS network. In a packet loss event caused by random burst contention, \( \text{avg}_N \) is expected to be close to \( \text{avg}_M \) since the high utilization of network resources remains only a short time period in the N RTTs. A larger \( \text{avg}_N \) can be considered that a packet loss event is more likely due to network congestion rather than random burst contention.

![Fig. 1. A TCP over OBS overlay network.](image)

The relationship between \( \text{avg}_M \) and \( \text{avg}_N \) is based on the following observations: (1) in a TCP over OBS overlay network (as shown in Fig. 1), network congestion can occur in IP access networks, at OBS edge nodes, and in the OBS core network. The difference between network congestion and random burst contention in the OBS core network is that network congestion suffers from high resource utilization for a much longer period; (2) in the high-utilization state, the RTT of each packet delivery will be much higher than that in the low-utilization state. This is due to the fact that, in IP access networks, the high-utilization state will cause longer queueing delay. Also, in an OBS core network with contention resolution schemes, such as burst retransmission and deflection, bursts will more likely to be retransmitted or deflected in the high-utilization state, which results in a longer average burst delay in the OBS core network.
We further quantify the relation between the long-term and short-term statistics in order to define the confidence with which a packet loss is due to network congestion. We assume that the $M$ consecutive RTTs are random with a mean $\text{avg}_{-} \text{RTT} \_M$ and a variance $\text{Var}(\text{RTT})$. 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 three network scenarios: one is a barebone OBS network, where delay variation takes place in IP access networks and at OBS edge nodes; the second scenario is an OBS network with the burst deflection scheme, where delay variation takes place in IP access networks, at OBS edge nodes, and in the OBS core network due to burst deflection or burst retransmission. The null hypothesis in the Chi-square test is: “the distribution of the $M$ consecutive RTTs cannot be modeled as normal $N(\mu, \sigma)$, where $\mu = \text{avg}_{-} \text{RTT} \_M$, and $\sigma = \sqrt{M \cdot \text{Var}(\text{RTT})}$”. The distributions derived in the network scenarios are shown in Fig. 2. The experiment parameters are given in the numerical analysis section. We found that the null hypothesis can be rejected with $95\%$ confidence, which validates our assumption that the $M$ consecutive RTTs can be approximately modeled as a Normal distribution.

![TCP RTT distribution histogram](image)

Fig. 2. TCP RTT distribution histogram.

B. 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 burst statistics about the short-term network status when a packet is lost. If $N$ is chosen too small or too large, the short-term network status may not be accurately represented. Our approach in selecting $N$ employs an autocorrelation function:

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

where $\text{RTT}(i)$ is the RTT of the $i$th packet. The numbering of RTTs is shown in Fig. 3. The autocorrelation function can reflect how smooth a process is. $R(0,N)$ has the maximum value when $N = 0$. Also, a stronger correlation among RTTs results in a larger value of $R(0,N)$. In our scheme, the value of $N$ is selected such that $R(0,N) = R(0,0) \cdot \gamma^\%$, where $\gamma$ 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 $\gamma \%$. 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 $\gamma$ should be close to $1$. In our study, $\gamma$ is taken as $90\%$.

![Numbering of RTTs](image)

Fig. 3. Numbering of RTTs, where RTT$(0)$ denotes the RTT right before a packet loss occurs.

![The relation of $z_i$, $u_i$, and $\beta$ in the SAIMD scheme](image)

Fig. 4. The relation of $z_i$, $u_i$, and $\beta$ in the SAIMD scheme.

C. SAIMD Congestion Detection for TCP over OBS

After a proper $N$ is selected based on the approach in the previous subsection, the value of $\text{avg}_{-} \text{RTT} \_N$ can be obtained. Then, we can define the confidence with which the current packet loss event of a TCP session is due to network congestion by positioning $\text{avg}_{-} \text{RTT} \_N$ in the Normal distribution spectrum. The derived confidence is used to dynamically adjust $\beta$ at a TCP sender under a developed policy such that $\beta$ can represent the current network status.

For positioning $\text{avg}_{-} \text{RTT} \_N$ in the Normal distribution spectrum, a function $z_i = r_{\text{conf}}(u_i)$ is defined, where $u_i$ is the confidence level. The $r_{\text{conf}}(u_i)$ returns a RTT value (denoted by $z_i$) which is larger than a proportion $u_i$ ($0 < u_i \leq 1$) 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:

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

The $cdf(z_i)$ and $\text{pmf}(z_i)$ denote the cumulative density function (CDF) and probability mass function (PMF) in the RTT spectrum given the RTT value of $z_i$ [27]. The one-to-one mapping between $u_i$ and $z_i$ is shown in Fig. 4. 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.

The proposed policy for adjusting $\beta$ is as follows. When $\text{avg}_{-} \text{RTT} \_N$ is smaller than $z_i = r_{\text{conf}}(u_i)$, a low confidence of network congestion is indicated, which yields no adjustment
of the cwnd in response to a packet loss, i.e., $\beta = 1$. When $\text{avg}_N > \text{rtt}_u$ $> \text{rtt}_u$, it is a strong indication of network congestion. Hence, the TCP sender cuts the cwnd by half, i.e., $\beta = 0.5$ in response to a packet loss. When $\text{avg}_N$ falls in the interval $[z_1, z_2]$, $\beta = f(u_i)$, where $f(u_i) = 1 - \frac{u_i - u_1}{2(u_i - u_1)}$. Note that $u_i$ and $u_1$ 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, $u_i$ and $u_1$ are set to 50% and 90%, respectively. The policy-based cwnd adjustment scheme can be summarized in the following equation:

$$\beta = \begin{cases} 
0.5 & \text{avg}_N > \text{rtt}_u \\
\frac{f(u_i) > \text{avg}_N > \text{rtt}_u} {\text{rtt}_u} & \frac{\text{avg}_N > \text{rtt}_u} {\text{rtt}_u} \\
1 & \text{avg}_N \leq \text{rtt}_u
\end{cases} \quad (1)$$

The dynamic adjustment of $\beta$ based on the confidence level $u_i$ is also illustrated in Fig. 4. The flow chart for the proposed SAIMD scheme is shown in Fig. 5. Note that the adjustment of $\beta$ will be triggered only if a TCP sender detects a TD or TO.

We now discuss two extreme cases for the SAIMD scheme. 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 $\text{avg}_N$ and $\text{avg}_M$ obtained by the TCP sender are very close. Hence, $\beta$ will be close to 1. For a TD packet loss, the size of the cwnd will not be reduced enough. The SAIMD scheme will then cause persistent congestion in the network and TO packet 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 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 $\text{avg}_N$ and $\text{avg}_M$ obtained by the TCP sender are also close. When the network is congested, the cwnd will not be reduced enough as a response to the received TDs. The TCP flow will eventually timeout for the 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 packets 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 to its previous size could be very long (especially if the cwnd size is additively increased), 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 time spent for the cwnd recovery from the slow-start phase caused by false congestion detection. This is considered essential for those high-bandwidth TCP flows which may take hours or days to recover from a slow-start phase. Note that the computation for the autocorrelation and confidence interval is required only when a packet 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-duration and high-bandwidth TCP flows instead of short-duration TCP such as HTTP web services; thus, the additional overhead to the whole network is expected to be trivial.

![Fig. 5. The proposed SAIMD congestion control scheme.](image)

**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]. The following table lists the notations used in the analytical model.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$</td>
<td>packet loss probability</td>
</tr>
<tr>
<td>$b$</td>
<td>number of packets that are acknowledged by receiving an ack</td>
</tr>
<tr>
<td>$B$</td>
<td>TCP throughput</td>
</tr>
<tr>
<td>$H$</td>
<td>number of packets transmitted during TO</td>
</tr>
<tr>
<td>$\text{RTT}$</td>
<td>average round trip time</td>
</tr>
<tr>
<td>$\text{TOP}$</td>
<td>timeout period</td>
</tr>
<tr>
<td>$\text{TDP}$</td>
<td>triple duplicate period</td>
</tr>
<tr>
<td>$\text{RTO}$</td>
<td>retransmission timeout</td>
</tr>
<tr>
<td>$Z^{\text{TO}}$</td>
<td>duration of a sequence of TOs</td>
</tr>
<tr>
<td>$X$</td>
<td>number of successful rounds in a TDP</td>
</tr>
<tr>
<td>$Y$</td>
<td>number of packets sent before TD or TO expiration</td>
</tr>
</tbody>
</table>
\( W \): current congestion window size in packets

\( W_m \): TCP maximum window size

\( S \): number of packets 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 \( \text{cwnd} \) (in packets) and waits until either it receives an acknowledgement or the TO expires. We also define a TD loss as a packet loss detected by triple duplicates and define a TO loss as a packet loss detected after the 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 subsections, we will derive \( E[Y] \), \( E[TDP] \), \( E[TOP] \), \( E[H] \), and \( Q \) for TD and TO losses respectively.

A. Triple Duplicate (TD) Losses

As per the model in [8], suppose that the \( (c+1) \)th burst is the first burst lost in the \( i \)th TDP, \( TDP_i \), which contains the first \( (a_i + 1) \)th lost packet in the \( TDP_i \). As shown in Fig. 6, \( h_i \) additional packets will be sent in the same round after the \( (c+1) \)th burst is sent and lost. After receiving TD, the TCP sender retransmits all the missing packets contained in the lost burst in the next round. Therefore, in the next round, \( W_{X_i} - S \) new packets will be sent, where \( W_{X_i} \) is the \( \text{cwnd} \) size in the \( X_i \)th round in the \( TDP_i \). After recovering all the packets lost in the burst, a new round \( TDP_{i+1} \) starts with the \( \text{cwnd} \) being cut by a factor of \( \beta \). The total number of packets successfully transmitted during the \( TDP_i \) is \( Y_i = a_i + h_i + W_{X_i} - S \). \( E[h] \) is approximately equal to \( E[\beta]E[W_{X_i}] \), since \( 0 \leq h_i \leq W_{X_i} \) and the \( \text{cwnd} \) is reduced by a factor of \( \beta \) for every TD loss. Thus, we have

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

As per Eq. (1), \( \beta \) is a function of the \text{avg.rtt.N} and the confidence level \( u_i \). Considering Fig. 4, we derive \( E[\beta] \) as follows,

\[
E[\beta] = \bar{\beta} = cdf(z_i) + \sum_{i=1}^{n} f(z_i) \cdot pmf(z_i) + \frac{1 - cdf(z_i)}{2} \tag{4}
\]

\[
u_i + \sum_{i=1}^{n} \left( \frac{1 - u_i - u_i}{2(u_i - u_i)} \right)(u_i - u_i) + \frac{1 - cdf(z_i)}{2} \tag{4}
\]

\[
u_i + \frac{1 - u_i}{2} + \sum_{i=1}^{n} \frac{u_i - u_i}{2(u_i - u_i)} (u_i - u_i) + \sum_{i=1}^{n} \frac{u_i(u_i - u_i)}{2(u_i - u_i)} \tag{4}
\]

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 \( z_i \). An alternative way for solving \( E[\beta] \) can be through historical collection of the \( \beta \) values, which yields:

\[
E[\beta] = \bar{\beta} = \sum_{i} \beta \cdot p(\beta_i) \tag{5}
\]

where \( p(\beta_i) \) is the probability of the distinct values \( \beta_i \) to exist.

In order to derive \( E[a] \), 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,

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

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

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

1) For high packet losses \( (W_X < W_m) \)

In the presence of a high packet loss probability, the \( \text{cwnd} \) will remain less than the maximum size \( W_m \). Recall that \( b \) denotes the number of packets that are acknowledged by receiving an \text{ack}. During the \( TDP_i \), the \( \text{cwnd} \) increases between \( \beta W_{X_i} \) and \( W_{X_i} \). Since the increase of \( \text{cwnd} \) 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

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

Since \( Y_i \) can be derived by summarizing the number of packets sent in \( X_i \) successful rounds and the additional \( (W_{X_i} - S) \) packets in the next round of \( X_i \), as shown in Fig. 6, we have:
\[ Y_i = \sum_{k=0}^{X_i/b-1} (\beta W_{X_{i-k}} + k)b + W_{X_i} - S \]

\[ = \frac{X_i}{2} (2\beta W_{X_{i-1}} + \frac{X_i}{b} - 1) + W_{X_i} - S \]

By substituting Eq. (9), we have

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

By assuming zero correlation between \( \beta \) and \( W_{X_i} \), after substituting Eq. (10), we get

\[ E[Y] = \frac{b(1-\bar{\beta}^2)E[W_{X_i}^2 - b(1-\bar{\beta})E[W_{X_i}]} + E[W_{X_i}] - S \quad (11) \]

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

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

\[ E[W_{X_i}] \] can be then obtained as

\[ \bar{\beta} - \frac{b(\bar{\beta} - 1)}{2} + \sqrt{\frac{b(\bar{\beta} - 1) - \bar{\beta}^2 + 2Sb(1-\bar{\beta})}{p}} \]

\[ = \frac{b(1-\bar{\beta})}{2} \quad (12) \]

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

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

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

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

\( E[TDP] \) is then obtained as

\[ E[TDP] = RTT(E[X] + 1) \]

\[ = RTT(\frac{2\bar{\beta} - b(\bar{\beta} - 1)}{2} + \sqrt{\frac{b(\bar{\beta} - 1) - \bar{\beta}^2 + 2Sb(1-\bar{\beta})}{p}} + 1) \quad (15) \]

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,

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

During each TDP, the \( cwnd \) size linearly increases from \( \beta W_m \) 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 packets 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 packets that are successfully transmitted before a packet loss is \( S/p \). Hence we have

\[ \frac{(W_m - \beta W_m)^2}{2} + W_m(X_i - W_m + \beta W_m) - h_i = \frac{S}{p} \quad (17) \]

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

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

The duration of the TDP is obtained as,

\[ E[TDP] = RTT(E[X] + 1) \]

\[ = RTT(W_m(1 - \bar{\beta}) - \frac{W_m(1 - 2\bar{\beta} + \bar{\beta}^2)}{2} + \frac{S}{W_m p} + 1) \quad (19) \]

B. 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

\[ E[H] = E[R] - 1 - \frac{p}{1-p} \quad (20) \]

\[ E[TO] = RTO \frac{f(p)}{1-p} \quad (21) \]

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

\[ Q(E[W_{X_i}]) \approx \frac{w_i}{S} \quad (22) \]

C. SAIMD TCP SACK over OBS Throughput Estimation

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

\[ B_z = \frac{2\bar{\beta} - b(\bar{\beta} - 1)}{2} + \sqrt{\frac{b(\bar{\beta} - 1) - \bar{\beta}^2 + 2Sb(1-\bar{\beta})}{p}} + \frac{1-p}{p} S + \frac{p}{1-p} \]

\[ = \frac{2\bar{\beta} - b(\bar{\beta} - 1)}{2} + \sqrt{\frac{b(\bar{\beta} - 1) - \bar{\beta}^2 + 2Sb(1-\bar{\beta})}{p}} + 1 + p \frac{w_i}{S} \frac{RTO \cdot f(p)}{1-p} \quad (23) \]

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
\[ B_S = \frac{(\bar{\beta} + 1)W_m + \frac{(1-p)S}{p} + \frac{W}{S^{\frac{1}{2}}}}{RTT(W_m(1-\bar{\beta}) - \frac{W_m(1-2\bar{\beta} + \bar{\beta}^2)}{2} + \frac{S}{W_m}p + 1) + \frac{W}{S^{\frac{1}{2}}}RTOf(p)} {1-p} \]

\[
(24)
\]

V NUMERICAL RESULTS

Simulation is conducted using NS-2, where the NSF network topology shown in Fig. 7 is adopted as the OBS core network. The distance along each link is in km. Each link has 8 wavelengths operating at 10 Gbps. One bi-directional control channel is allocated along each link. Control-packet processing time is set to 1 \( \mu \)s at both core and edge nodes. The offset time is set to 6 \( \mu \)s which is sufficient for bursts to traverse a minimum of 4 hops. There is no fiber delay line at OBS core nodes. The mixed time/length based burst assembly algorithm is adopted, where 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 packets, and each packet has the size of 1 KB. The TCP throughput is obtained over a simulation period of 10^4 seconds. The simulated TCP flows are fast and medium flows. Depending on the number of competing TCP flows, the packet loss probability varies between 10^-5 and 10^-2. We compare the SAIMD scheme with the conventional TCP implementations, such as Reno and Sack, along with the Threshold-based TCP Vegas (T-Vegas) [16], in which the \( cwnd \) size is adjusted based on the number of long RTTs exceeds the threshold \( T \) within \( N \) RTTs.

In our simulation, the TCP senders and receivers are connected to the OBS edge nodes through IP access networks. Burst losses occur at the OBS core network due to burst contention. Burst retransmission and deflection routing are implemented. We examine the TCP throughput over barebone OBS, OBS with burst retransmission, and OBS with burst deflection. If a burst experiences contention after the second retransmission or deflection attempt, the burst will be dropped. In the simulation, RTT increases primarily due to the buffering delay at the OBS edge nodes, the burst assembly, and the deployment of the burst retransmission and deflection.

Fig. 7. NSF network topology adopted in the study.

A. SAIMD Throughput Performance

Fig. 8 shows the simulation result for the relationship between the average TCP RTT versus the packet loss probability. We can see that a significant amount of extra delay was introduced to the retransmitted and deflected bursts. It is notable that the retransmitted or deflected bursts can successfully reach their destinations under low packet loss probabilities. Under high packet loss probabilities, very few packets can reach their destinations and compute their RTT value.

Fig. 9 shows the simulation result on the average value of \( \beta \) in the SAIMD scheme at various packet loss probabilities. We observe that with higher packet loss probability, the average value of \( \beta \) decreases. When the packet loss probability is close to 0.01, the average values of \( \beta \) are close to 0.5. This is due to the fact that higher network utilization results in higher packet loss probability and higher packet delay, leading to higher confidence of network congestion.

Fig. 8. Average TCP RTT vs. packet loss probability.

Fig. 10 demonstrates the throughput of the proposed SAIMD scheme obtained by the developed analytical model and by simulation, respectively. It is clear that the TCP throughput decreases rapidly as the packet loss rate increases, which meets our expectation. Also, our analytical model is validated since the results from both analysis and simulation match well with each other.

Fig. 11 shows the simulation results for the throughput of Sack/Sack-SAIMD, Reno/Reno-SAIMD, and T-Vegas 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 packet loss probability is low. This is because the Sack and Reno senders always unnecessarily halve the \( cwnd \) for a packet loss event at low traffic loads. On the other hand, the SAIMD Sack senders have achieved up to 37% throughput improvement compared with that by the AIMD senders since they do not rigidly react to a packet loss event at low traffic loads. Instead, the factor \( \beta \) is adjusted at each SAIMD sender to guarantee a smaller \( cwnd \) as a contention resolution scheme in the OBS network. Retransmission in the OBS domain [8] has been reported to be able to reduce the overall packet loss probability.
at the expense of longer packet delay (as shown in Fig. 8). By integrating the burst retransmission scheme with the SAIMD scheme, we can observe up to 81% improvement in TCP throughput compared to Reno and Sack. We can also see that Sack-SAIMD performs better than T-Vegas, and Reno-SAIMD perform similar to T-Vegas. Similar results have been obtained when enabling burst deflection in the OBS network, which is not included in this paper due to the space limitation.

B. SAIMD Fairness

In this section, we examine the fairness among Reno-SAIMD, Sack-SAIMD, and the widely deployed TCP implementations, such as Reno and Sack. For this purpose, we use Jain’s fairness index which is defined as \( \frac{\sum_{i=1}^{n} B_i^2}{n \sum_{i=1}^{n} B_i^2} \), 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. Figs. 13 and 14 show the fairness index of TCP flows by Reno, Sack, and Reno-SAIMD, and Sack-SAIMD, over the barebone OBS and the OBS with burst retransmission. We can observe that the SAIMD has a much better fairness index compared to 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. The simulation results with burst deflection enabled in the OBS network are similar to the results of burst retransmission. Due to the space limitation, we did not include the results of burst deflection.
Fig. 14. Fairness Index of Reno, Sack, Reno-SAIMD, and Sack-SAIMD in OBS with burst retransmission.

VI CONCLUSIONS

The paper introduced a novel TCP congestion control framework, called SAIMD, 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 detection due to the bufferless characteristic in the OBS domain. The proposed scheme collects and analyzes the historical RTTs and adjusts $\beta$ according to the statistics of the collected RTTs at the occurrence of any packet 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 TCP flows.

REFERENCES