

TCP-ENG: Dynamic Explicit Congestion Notification for TCP over OBS Networks

Basem Shihada¹, Pin-Han Ho^{1,2}, and Qiong Zhang³

¹David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, Canada

²Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada

³Mathematical Science and Applied Computing, Arizona State University, Phoenix, USA

Abstract— Transport Control Protocol (TCP) has served as a reliable, self-regulated, and congestion tolerant transport protocol for many Internet applications. Relatively, limited knowledge has been gained in terms of the impacts encountered in the TCP when Optical Burst Switching (OBS) is adopted in the network backbone. A novel scheme, called TCP with Explicit Notification Generalized Additive Increase Multiplicative Decrease (TCP-ENG), is introduced, which is considered as the first study that integrates the explicit notification platform with the GAIMD approach. The proposed scheme aims to solve the false congestion detection problem in the IP over OBS networks. An analytical model is developed for the proposed scheme and is verified through extensive simulation.

I. INTRODUCTION

Optical Burst Switching (OBS) has been considered as one of the most promising supporting technologies for the next-generation all-optical networks. In OBS networks, data bursts of variable lengths are launched by the OBS edge nodes and asynchronously switched along predetermined physical routes to support the corresponding services. To transport a data burst across the OBS domain, the corresponding control packet is initiated at the OBS edge node and is sent prior to the data burst in order to configure the switch fabrics of each intermediate node along the route [3][5][6].

Due to its unacknowledged resource reservation mechanism and dynamic characteristics, OBS has been widely considered to provision UDP-based real-time services. However, the Internet is essentially dominated with data traffic, in which the TCP bulk data transfer mechanism with strict QoS requirements in terms of data integrity, reliability, flow and congestion control are of extremely high importance [1][2]. To our best knowledge, only a few studies have been reported dedicatedly to the TCP design over the OBS networks [3][5], [11][12], and a relatively limited amount of understandings on the TCP behaviours over the OBS networks have been gained in the presence of burst dropping at different traffic loads.

A number of TCP implementations have been proposed for detecting and controlling network congestion and segment losses in order to solve some specific problems. In the past years, TCP has been extensively modified and/or extended to cope with the problems incurred by data transmission in a number of different network environments, including mobile wireless networks, ad hoc networks, and some lower layer networks with a specific transmission technology such as optical circuit switched (OCS) and optical burst switched

(OBS) networks. It is clear that each of the TCP enhancements has its own design premises, and could be very effective in one circumstance while being much outperformed by another. It has also been proven that jointly considering the characteristics of the whole network environment in the design of the TCP modifications/extensions is necessary. These facts are especially distinguished when TCP is extended to the OBS networks, where multiple TCP segments are assembled in a single burst at the edges which traverses through the core nodes without any buffering facility.

Generally, TCP takes packet loss events as an indication of network congestion. However, in lightly loaded OBS, packets can be occasionally dropped due to contention, where two or more bursts try to access the same output port simultaneously. In this case, the corresponding TCP will be informed of network congestion which consequently results in the reduction of congestion window size (*cwnd*) even if the network is under-utilized. This is also referred to as *false congestion detection*, which may significantly impair the TCP throughput.

TCP false congestion detection may impose a fundamental vicious impact on the throughput performance in IP over OBS networks. It could be far from appropriate to simply cut *cwnd* by a half or turn to the slow-start stage in response to a TCP segment loss without regarding the OBS domain congestion status. This problem is getting more serious for those TCP senders of high-bandwidth and long-lasting flows, where the recovery from slow-start could take hours or days according to the currently widely adopted additive-increase framework. Therefore, an enhanced TCP implementation for the end-user *cwnd* adjustment that is responsive to the underlying OBS functionality and burst delivery mechanisms is critical to the success of employing the OBS infrastructure in the modern communication carrier networks with mission-critical services such as Grid applications.

Very few research efforts have addressed the performance of TCP over OBS. BTCP [3] is one of the TCP implementations proposed to handle the issue of detecting false timeout, where false reactions to the TCP timeouts are performed in the TCP senders at a burst loss event under various traffic loads. The authors in [3] also proposed using TCP segmentation strategies, such that a TCP agent located at the ingress node sends Acks whenever a TCP segment is received. This approach simply violates the end-to-end TCP semantics since an ACK reaches the sender before actual completion of the packet delivery. Also in [3], a more complicated approach has been proposed, where each core

node maintains a TCP agent. Whenever, a burst dropping occurs, this agent disassembles the burst and sends back negative Acks to the sender. Obviously, this approach introduces intolerably high complexity at the switching core. In [7], the authors proposed to persistently retransmit a lost burst. This approach mainly requires the edge node to buffer a copy of the all the bursts within a certain time window until they are successfully delivered.

The authors in [10] have introduced a TCP scheme based on the framework of GAIMD [8] for the congestion window adjustment, in which both functions of identifying burst contention caused burst drop and multiple data segment losses in a single round trip are supported. With BAIMD, the TCP senders simply consider every burst drop as due to congestion. To mitigate the false identification of network congestion status that may irrelevantly shrink $cwnd$ in the TCP senders, the factors α and β are manipulated such that the summarized effect of a burst drop event due to both contention and congestion is evaluated at each TCP sender. BAIMD is characterized by a clean separation between the signaling between the TCP senders and OBS edge nodes, in which no extra signaling overhead is introduced between the two layers.

Based on the amount of asynchronous bursty bandwidth demand at each ingress-egress pair, a new scheme for AIMD should be devised such that decreasing the window size only responds to the event of true network congestion. The idea has been exercised by [4] in wireless environments, where explicit loss notification (ELN) is devised for the state leakage between the lower layer and the TCP layer, similar to the approaches proposed in [3]. With ELN, the cause of each burst loss is reported to the TCP sender, which could be due to congestion or other transmission conditions. In this way, the TCP sender adjusts its $cwnd$ according to the network status of the lower layer.

In this paper, we solve the false congestion detection problem and avoid the unnecessary $cwnd$ reduction by introducing a novel TCP implementation, called TCP with Explicit Notification Generalized AIMD (TCP-ENG). The proposed scheme measures the link utilization on each OBS network path and adjusts TCP $cwnd$ based on the link utilization and burst dropping information carried in the explicit notification messages. This is considered as the first study that integrates the explicit notification platform with the GAIMD

This paper is organized as follows. Section II introduces the TCP-ENG scheme for the OBS networks. Section III presents an analytical model for the throughput performance of TCP-ENG. In Section IV, simulation is conducted to verify the proposed performance model and compares the proposed scheme with a number of counterparts, including Reno, Sack, and BAIMD. Section V concludes the paper.

II. DYNAMIC EXPLICIT CONGESTION NOTIFICATION FOR OBS NETWORKS (TCP-ENG)

It has been shown that the previously reported TCP implementations are subject to numerous limitations and negatively affected by the OBS bufferless characteristic. In

order to further improve the TCP performance in the IP over OBS networks, a novel TCP scheme, called Explicit Congestion Notification GAIMD (TCP-ENG), is introduced. The proposed TCP-ENG scheme intends to mitigate the impact due to the false congestion detection problem without losing the TCP friendliness. In terms of the design premise and novelty, the proposed TCP-ENG scheme takes the best of BAIMD and BTCP, where the AIMD window-based congestion control paradigm and the mechanisms of explicit notification and/or signaling between the TCP and the OBS layer are jointly exercised. We will demonstrate how the scheme is implemented followed by a performance modeling.

The proposed TCP-ENG scheme can solidly improve the BAIMD scheme [10] in the aspect of dealing with the burst losses containing a large number of TCP segments. Note that when the $cwnd$ is small due to a high dropping rate, the BAIMD flows are subject to a lower throughput compared with the existing conventional TCP flows. This is due to the smaller sending rate α in BAIMD compared with the fixed $\alpha = 1$ in the conventional TCP. In the case where no burst dropping occurs, the BAIMD $cwnd$ is increased by one packet in every $1/\alpha$ RTT. If burst dropping occurs during this time due to random contention, the $cwnd$ will be decreased before it successfully completed one increase. Thus, the actual increase rate will never reach α . The situation is getting worse while considering fast TCP flows [11] since the burst loss results in a complete loss of the entire $cwnd$ segments causing these flows to return to slow-start. As a result, the BAIMD flows will hardly reach that of the traditional TCP throughput.

TCP-ENG aims to solve the above problem by allowing the TCP senders to dynamically react to the received explicit loss notifications from the OBS edges in tuning the control parameter pair (α, β) . In the OBS domain, it is reasonable to assume that the utilization of a path connecting two OBS edge nodes can be easily identified by the edge nodes. First of all, we take the burst dropping rate along a path as the evaluation of the utilization of the path. Each OBS ingress edge maintains the burst dropping rate in a small time window (i.e., Δt) for each path. When the burst dropping rate along the path is less than a threshold, the path is taken as under-utilized; otherwise, the path is taken as over-utilized. Once a burst dropping event occurs while the path is under-utilized, the edge node is responsible for sending *burst explicit notifications (BENs)* to the TCP senders.

From the TCP senders' perspective, the parameter pair (α, β) is maintained and possibly updated at the end of each RTT. In our scheme, the value of α is defined according to the size of $cwnd$ and the value of β . At the slow-start phase where $cwnd = 1$, the TCP has $(\alpha, \beta) = (1, 0.5)$. When $cwnd$ is in the middle size, i.e., $1 < cwnd < 1/(1 - \beta)$, α is set to 1 if the TCP sender did not receive any *BEN* in the previous round. Otherwise, α is determined according the triple duplicate or timeout equations of GAIMD, when the TCP sender encounters a packet drop that is associated with a *BEN*. The GAIMD *sending rate* equations are illustrated in the following sub-section.

To summarize, one of the unique features in our scheme is that a TCP-ENG sender takes the OBS domain as the link-

layer, and each path in the OBS domain is taken as a link for the upper layer protocols. With such a design, the proposed scheme does not need to know the burst dropping occurs at which core node of a path. To enable a simplified signaling mechanism and explicit notification, a TCP agent is placed at each OBS ingress node to summarize the congestion status of each path sourced at the ingress by calculating the burst dropping rate of the path. This is considered as the most straightforward approach for acquiring the congestion status of the path in the OBS domain. On the other hand, *BEN* is performed at the edge nodes in such a way that the TCP senders simply take every segment loss as due to congestion if not notified. In other words, the TCP agent in the OBS edge keeps quiet on any burst dropping event classified as due to congestion, while a *BEN* is sent to the corresponding TCP senders only when the burst loss is judged as due to contention. We can clear observe that the scheme can achieve a much better scalability and implementability than BTCP in [3] in terms of signalling architecture design since BTCP reports every burst loss from core nodes, leading to a vast number of NAKs sent from OBS edge nodes.

III. PERFORMANCE MODELING

In this section, we analyze the throughput performance of the TCP-ENG flows in an OBS network. We have selected TCP SACK for our analytical model since it has been widely deployed in the current operating systems and has shown the best throughput performance over OBS networks compared to TCP Reno and New Reno [3]. The following table lists the notations used in the analytical model.

p	: packet loss probability
b	: number of packets that are acknowledged by receiving an <i>ack</i>
B	: TCP throughput
H	: number of packets 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 packets 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 *ack* or the timeout (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 TCP sender timeouts.

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

$$B = \frac{E[Y] + Q \times E[H]}{E[TDP] + Q \times E[TOP]} \quad (1)$$

It is worth to recall that TCP-ENG uses the GAIMD congestion control mechanism. The relation between α and β for ensuring friendliness among other existed TCP flows in the case of triple-duplicate *acks* is obtained from [8] as:

$$\alpha = \frac{3(1-\beta)}{1+\beta} \quad (2)$$

while in the case of timeout,

$$\alpha = \frac{4}{3}(1-\beta^2) \quad (3)$$

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.

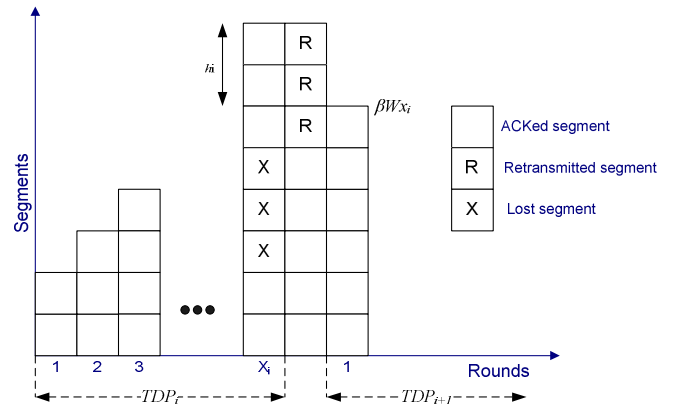


Fig. 2. Evolution of TCP-ENG Sack congestion window over OBS networks.

A. TCP-ENG in Triple Duplicate (TD) Losses

As per the model in [12], suppose that the (c_i+1) th burst is the first burst lost in the i th TDP, TDP_i , which contains the first (u_i+1) th lost segment in the TDP_i . As shown in Figure 2, h_i additional segments will be sent in the same round after the (c_i+1) th burst is sent and lost. 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 and the receiving *BEN*, a new round TDP_{i+1} starts with the *cwnd* being cut by a factor of β . The sending rate α is determined according to Eq. (2) and Eq. (3). The total number of segments successfully transmitted during the TDP_i is $Y_i = u_i + h_i + W_{X_i} - S$. $E[h]$ is approximately equal to $\beta E[W_X]$, since $0 \leq h_i \leq W_{X_i}$ and the *cwnd* is reduced by β for every TD loss.

Thus, we have

$$E[Y] = E[u] + (\beta + 1)E[W_X] - S \quad (4)$$

In order to derive $E[u]$, 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:

$$P[c = k] = (1 - p)^{k-1} \cdot p \quad (5)$$

Given that $a_i = Sc_i$ we have,

$$E[a] = S E[c] = S \sum_{k=1}^{\infty} k(1-p)^{k-1} p = \frac{S}{p} \quad (6)$$

By substituting Eq. (6) into Eq. (4), we have

$$E[Y] = (\beta + 1)E[W_X] + \frac{1-p}{p} S \quad (7)$$

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

In the presents of a high packet loss probability, the $cwnd$ will remain less than the maximum size W_m . Recall that b denotes the number of packets 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 α/b , thus,

$$W_{X_i} = \beta W_{X_{i-1}} + \frac{\alpha X_i}{b} \quad (8)$$

By reversing Eq. (8), we have

$$E[X] = \frac{b(1-\beta)E[W_X]}{\alpha} \quad (9)$$

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 Figure 2, we have:

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

By substituting Eq. (8), we have

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

after substituting Eq. (9), we get

$$E[Y] = \frac{b(1-\beta^2)E[W_X]^2 - b(1-\beta)E[W_X]}{2\alpha} + E[W_X] - S \quad (10)$$

By combining Eq. (10) and Eq. (7), we have,

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

$E[W_X]$ can be then obtained as

$$E[W_X] = \frac{\alpha\beta - \frac{b(\beta-1)}{2} + \sqrt{(\frac{b(\beta-1)}{2} - \alpha\beta)^2 + \frac{2\alpha S b(1-\beta^2)}{p}}}{b(1-\beta^2)} \quad (11)$$

By substituting Eq. (11) into Eq. (7), we obtain $E[Y]$ as,

$$E[Y] = \frac{\alpha\beta - \frac{b(\beta-1)}{2} + \sqrt{(\frac{b(\beta-1)}{2} - \alpha\beta)^2 + \frac{2\alpha S b(1-\beta^2)}{p}}}{b(1-\beta)} + \frac{1-p}{p} S \quad (12)$$

Also, by substituting Eq. (11) into Eq. (9), we obtain $E[X]$ as,

$$E[X] = \frac{\alpha\beta - \frac{b(\beta-1)}{2} + \sqrt{(\frac{b(\beta-1)}{2} - \alpha\beta)^2 + \frac{2\alpha S b(1-\beta^2)}{p}}}{1+\beta} \quad (13)$$

$E[TDP]$ is then obtained as

$$\begin{aligned} E[TDP] &= \overline{RTT}(E[X]+1) \\ &= \overline{RTT} \left(\frac{2\alpha\beta - \frac{b(\beta-1)}{2} + \sqrt{(\frac{b(\beta-1)}{2} - \alpha\beta)^2 + \frac{2\alpha S b(1-\beta^2)}{p}}}{1+\beta} + 1 \right) \end{aligned} \quad (14)$$

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. (7) we can obtain,

$$E[Y] = (\beta + 1)W_m + \frac{1-p}{p} S \quad (15)$$

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 $(\alpha X_i - (W_m - \beta W_m))$ rounds, hence we can obtain the number of segments that are transmitted before a TD loss as $\frac{(W_m - \beta W_m)^2}{2} + W_m(\alpha X_i - W_m + \beta W_m) - h_i$. On the other

hand, from Eq. (6), the total number of segments that are successfully transmitted before a packet loss is S/p . Hence we have

$$\frac{(W_m - \beta W_m)^2}{2} + W_m(\alpha X_i - W_m + \beta W_m) - h_i = \frac{S}{p} \quad (16)$$

By reversing Eq. (16), we can obtain $E[X]$ as

$$E[X] = \frac{W_m(1-\beta)}{\alpha} - \frac{W_m(1-2\beta+\beta^2)}{\alpha^2} + \frac{\beta}{\alpha} + \frac{S}{\alpha W_m p} \quad (17)$$

The duration of the TDP is obtained as,

$$\begin{aligned}
E[TDP] &= \overline{RTT}(E[X]+1) \\
&= \overline{RTT} \left(\frac{W_m(1-\beta)}{\alpha} - \frac{W_m(1-2\beta+\beta^2)}{\alpha^2} + \frac{\beta}{\alpha} + \frac{S}{\alpha W_m p} + 1 \right)
\end{aligned} \quad (18)$$

B. Timeout (TO) Losses

The behavior of TCP-ENG for a TO loss is same as that of TCP Sack. Hence, the analysis of TO losses is same as the analysis in [12]. TCP-ENG transmits the same number of packets between two TOs as compared with traditional TCP. However, for TCP-ENG, the packet retransmission starts as soon as the *BEN* is received, which is the *RTT* after the loss round. Thus, TCP-ENG waits for $TOP = RTT/p$. From [12], we have:

$$E[H] = E[R] - 1 = \frac{p}{1-p}, \quad (19)$$

$$E[TOP] = \frac{RTT(f(p))}{p(1-p)}, \quad (20)$$

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

$$Q(E[W_X]) \approx p \frac{W_X - 1}{S}. \quad (21)$$

C. TCP-ENG SACK over OBS Throughput Estimation

In the case of $W_X < W_m$, we can obtain the TCP-ENG throughput by substituting Eqs. (12), (14), (19), (20), and (21) into Eq. (1), which yields

$$\begin{aligned}
B &= \frac{\alpha\beta - \frac{b(\beta-1)}{2} + \sqrt{\left(\frac{b(\beta-1)}{2} - \alpha\beta\right)^2 + \frac{2\alpha S b(1-\beta^2)}{p}}}{b(1-\beta)} + \frac{1-p}{p} S + \frac{p^{\frac{W_X}{S}}}{(1-p)} \\
&= \frac{2\alpha\beta - \frac{b(\beta-1)}{2} + \sqrt{\left(\frac{b(\beta-1)}{2} - \alpha\beta\right)^2 + \frac{2\alpha S b(1-\beta^2)}{p}}}{RTT \left(\frac{W_m(1-\beta)}{\alpha} - \frac{W_m(1-2\beta+\beta^2)}{2\alpha} + \frac{\beta}{\alpha} + \frac{S}{\alpha W_m p} + 1 \right) + p \frac{W_X - 1}{S} \frac{RTT(f(p))}{p(1-p)}}
\end{aligned} \quad (22)$$

In the case of $W_X = W_m$, TCP-ENG throughput can be obtained by substituting Eqs. (15), (18), (19), (20), and (21) into Eq. (1), which yields

$$\begin{aligned}
B &= \frac{(\beta+1)W_m + \frac{(1-p)S}{p} + \frac{p^{\frac{W_m+1}{S}}}{1-p}}{RTT \left(\frac{W_m(1-\beta)}{\alpha} - \frac{W_m(1-2\beta+\beta^2)}{2\alpha} + \frac{\beta}{\alpha} + \frac{S}{\alpha W_m p} + 1 \right) + p \frac{W_m - 1}{S} \frac{RTT(f(p))}{p(1-p)}}
\end{aligned} \quad (23)$$

IV. NUMERICAL RESULTS

In this section, we evaluate the throughput performance of the proposed TCP-ENG implementation in the IP over OBS environment. We compare the TCP-ENG throughput with the BTCP and BAIMD as well as the TCPs that were originally designed for general packet-switched networks, including TCP Reno and SACK. The network simulator *NS-2* is used and the NSF network topology is implemented as shown in Figure 3. The distances shown are in *km*.

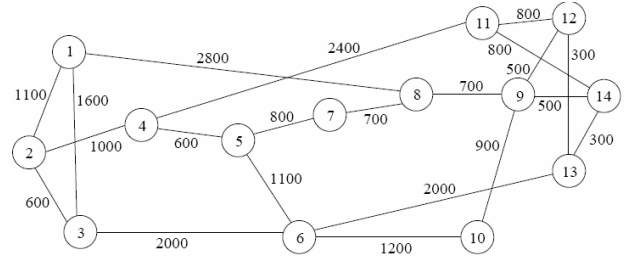


Fig. 3. NSF OBS simulation topology

The bursts are generated at the ingress edges traverse through nodes with a minimum of four hops before reaching their destinations. The number of co-existing TCP flows range from 1 to 14000 to achieve the lowest and highest congestion scenarios concerned in this study. Depending on the number of competing TCP flows, the burst dropping probability varies between 10^{-5} and 10^{-1} . At each ingress edge, the mixed time/length based burst assembly algorithm is used, where the burst timeout threshold is set to *1msec* and the maximum burst length is *50KB*. The core nodes implement the *LAUC-VF* channel scheduling algorithm. One fiber link consist of 8 wavelengths operating at *10Gbps* transmission rate is used for data burst transfer between two adjacent nodes with *10ms* propagation delay. One bi-directional control channel is allocated along each link. Control packet processing time is set to *1μs* at both core and edge nodes. The burst offset time is set to *4μs*. A File Transfer Protocol (FTP) application is used for generating TCP traffic with a *1KB* average packet size. The TCP throughput is obtained over a simulation period of 10^3 seconds. Depending on the degree of multiplexed flows at the edge nodes, the average number of packets that belong to a single flow being assembled in one burst ranges from 1 to 9 packets.

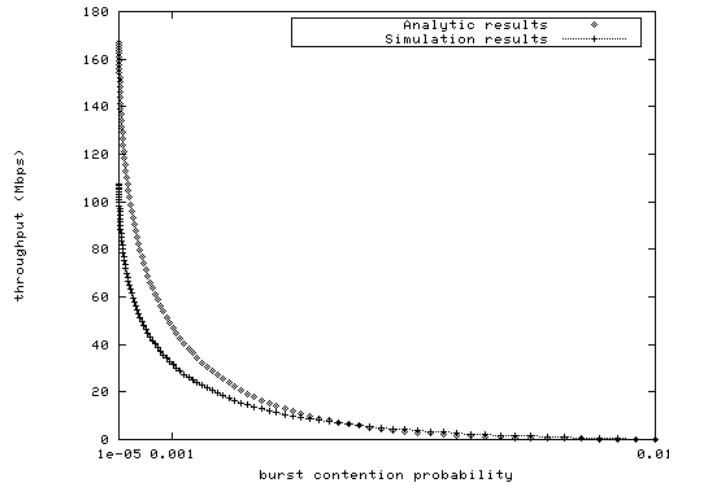


Fig. 4. TCP-ENG throughput over OBS networks.

A. TCP-ENG Numerical Results

Figure 4 compares the analytical results obtained by Eq. (22) and Eq. (23) and the simulation results obtained for TCP-ENG fast flows [11]. It is notable that the simulation results match

the analytical results. Figure 5 shows the throughput performance by TCP-ENG, TCP Reno and SACK under barebone OBS and OBS with burst retransmission. It is clear that TCP-ENG significantly outperforms the two conventional TCP schemes.

In Figure 6, TCP-ENG is compared with BAIMD and BTCP. We observe that TCP-ENG still achieves higher throughput than the other two by combining the best features of BTCP and BAIMD. The explicit notifications make the TCP-ENG senders be informed of a specific OBS domain channel status, where the two control parameters can be adjusted with the best relevance in response to a burst dropping event.

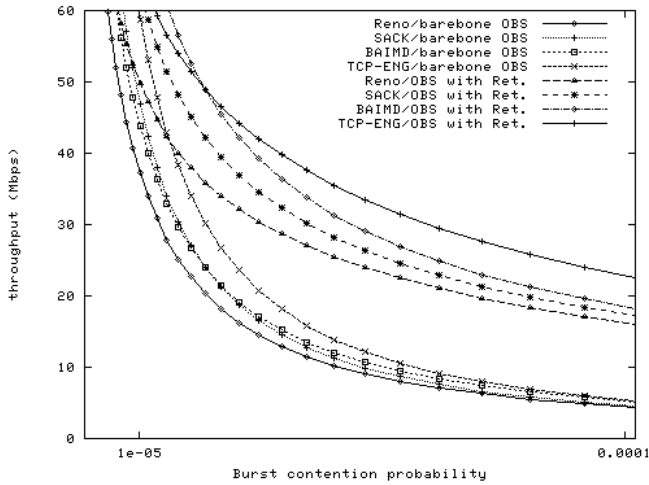


Fig. 5. TCP Reno, Sack vs. TCP-ENG throughput over barebone OBS and OBS with burst retransmission

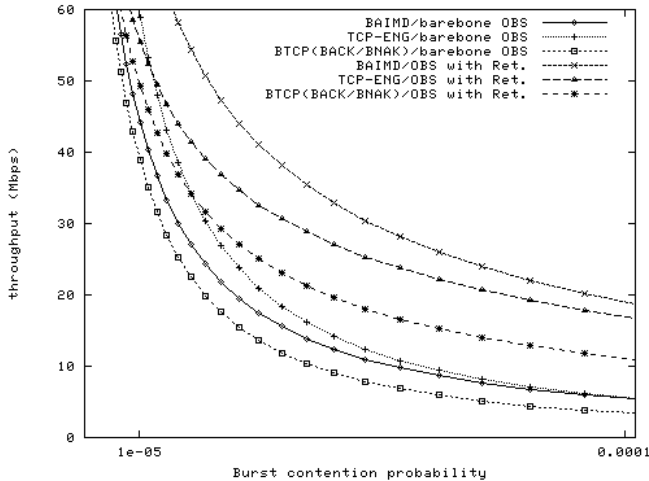


Fig. 6. BTCP, BAIMD, and TCP-ENG throughput over barebone OBS and OBS with burst retransmission.

Through the above simulation studies, we verified the proposed TCP-ENG scheme and concluded that TCP-ENG can solidly solve the false congestion detection problem and achieve better throughput than that by all the other TCP implementations based on AIMD mechanism in the carrier with OBS transmission. The overhead incurred in the

proposed scheme is the explicit notification which is required when a burst loss is determined as due to random contention. With the aid of explicit notifications and allocation of the TCP agent at the OBS edges, the TCP-ENG scheme can solidly outperform BAIMD and BTCP. In addition, we observed that the proposed scheme can effectively maintain fairness among competing AIMD flows and keep awareness on non-congestion burst dropping by manipulating the α and β values in the TCP senders, which can right serve as a candidate in the TCP implementation for the future Internet with OBS taken as the underlying infrastructure technology.

B TCP-ENG Fairness

In this section, the fairness among TCP-ENG, BAIMD, SACK, and Reno is examined. For this purpose, we use *Jain's fairness index* which is defined as $(\sum_{i=1}^n B_i)^2 / n \sum_{i=1}^n B_i^2$, where n is the number of competing flows and B_i is the throughput of the i^{th} flow. The fairness index ranges from $1/n$ to 1. If all flows have the same throughput, then the fairness index is 1 [8]. The competing flows share the same source-destination pairs.

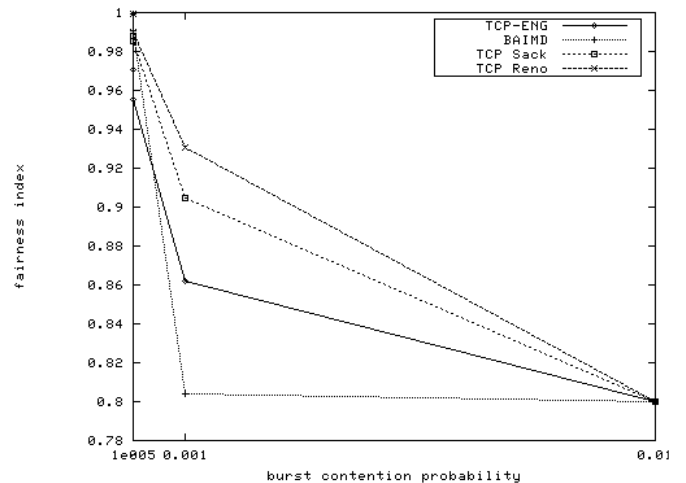


Fig. 7. Fairness Index of TCP-ENG, BAIMD, SACK, and Reno in barebone OBS.

Figure 7 shows the fairness index of flows by TCP-ENG, BAIMD, and TCP Reno. We can see TCP-ENG has a much better fairness index than BAIMD, TCP Reno, and SACK. This is due to the fact that the congestion control mechanism taken by the BAIMD, TCP Reno, and SACK does not rely on the explicit signalling between the TCP senders and the OBS edge nodes, which causes underestimation of network congestion. Thus, it results in selecting a larger multiplicative values β while keeping the additive value α at 1.

The fairness index has also been examined in Figure 8 while enabling the burst retransmission mechanism. Note that burst retransmission has been identified as an effective approach for enhancing the overall network throughput by recovering some contended bursts in the OBS layer while keeping the upper TCP layer unaware [8]. The simulation results have shown that the throughput of TCP-ENG is still close to SACK and

Reno, which verifies that TCP-ENG maintains a friendly relation with the other TCPs.

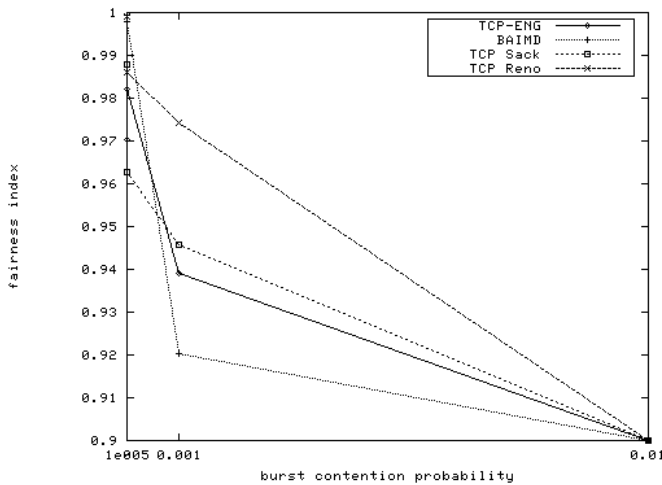


Fig. 8. Fairness Index of TCP-ENG, BAIMD, SACK, and Reno in OBS with burst retransmission.

V. CONCLUSIONS

In this paper, we proposed a new TCP implementation in IP over OBS networks, called TCP with Explicit Notification GAIMD (TCP-ENG), which aims to take the best of two recently proposed TCP enhancements: BAIMD and the BTCP. Through the simulation and the analytical modeling we demonstrated that the proposed TCP-ENG scheme can countermeasure the negative effect of the OBS bufferless characteristic and is expected to serve as a better candidate in the bufferless OBS networks compared with the conventional AIMD architecture. We have shown the superiority of the proposed scheme in terms of throughput, link utilization, and fairness.

REFERENCES

- [1] W. Stevens, "TCP/IP Illustrated, Volume1, Addison Wesley Longman, 1994.
- [2] W. Nouredine and F. Tobagi, "The transmission control protocol, an introduction to TCP and a research survey", Technical Report, July 2002.
- [3] X. Yu, C. Qiao, and Y. Liu, "TCP implementation and false time out detection in OBS networks," *Proceedings of IEEE Infocom*, pp. 774-784, 2004.
- [4] H. Balakrishnan, R. Katz, "Explicit loss notification and wireless web performance," *Proceedings of IEEE Globecom, Internet MiniConference*, 1998.
- [5] Y. Chen, C. Qiao, and X. Yu, "An optical burst switching: a new area in optical networking research," *IEEE Network*, vol. 18, no. 5, pp. 16-23, 2004.
- [6] X. Cao, J. Li, Y. Chen, and C. Qiao, "Assembling TCP/IP packets in optical burst switched networks", *Proceedings of IEEE Globecom*, 2002.
- [7] Q. Zhang, V. Vokkarane, Y. Wang, and J. Jue, "TCP over optical burst-switched networks with optical burst retransmission," *IEEE Journal on Selected Areas in Communications – Optical Communication and Networking Series*, 2005.
- [8] 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.

- [9] S. Bhandarkar and N. Reddy, "Improving the robustness of TCP to non-congestion events", internet draft draft-ietf-tcpm-tcp-dcr-01.txt, 2004
- [10] B. Shihada, P-H. Ho, F. Hou, and et al, "BAIMD: a responsive rate control for TCP over optical burst switched (OBS) networks," *Proceeding of IEEE (ICC)*, Istanbul, Turkey, 2006.
- [11] A. Detti and M. Listanti, "Impact of segment aggregation on TCP Reno flows in optical burst switching networks," in the proceedings of IEEE Infocom, 2002.
- [12] X. Yu, C. Qiao, Y. Liu, and D. Towsley, "Performance evaluation of TCP implementations in OBS networks," Technical Report, 2003-13, the State University of New York at Buffalo, 2003.
- [13] J. Li, C. Qiao, J. Xu, and D. Xu, "Maximizing throughput for optical burst switching networks", *Proceedings of IEEE Infocom*, HongKong, 2004.