Proactive Loss Recovery using Forward Segment Redundancy in Optical Burst-Switched (OBS) Networks

Deepak Chandran[+], Neal Charbonneau[*], and Vinod M. Vokkarane[*]
[+] *Enterasys Networks Inc., Andover, MA*
[*] *Department of Computer and Information Science, University of Massachusetts, Dartmouth, MA*
E-mail: dchandra@enterasys.com, u_ncharbonne@umassd.edu, and vvokkarane@umassd.edu

***Abstract*— Data burst contentions occur in optical burst-switched (OBS) networks due to one-way signaling and bufferless nature of the core network. A single burst loss containing multiple TCP segments can trigger drastic reactions from the corresponding TCP sources. TCP sender interprets segment loss as network congestion, resulting in throttling send rate. In this paper, we propose a new *forward segment redundancy* (FSR) mechanism that minimizes segment loss during burst contentions in the core and also recovers from segment loss in the forward direction using redundant segments placed in each data burst. In FSR mechanism, redundant TCP segments are added to each burst assembled at the ingress node before transmission to the destination. The segment losses due to contentions are minimized using a modified burst segmentation mechanism, and most lost segments are recovered in the forward direction using redundant segments. Using FSR, we observe significant improvement in TCP performance over OBS networks.** [1]

Keywords: Loss Recovery, TCP, and OBS.

I. Introduction

Optical burst switching (OBS) is a promising candidate for supporting next-generation Internet traffic. Data packets are assembled into bursts that are switched through the network optically. Each burst has an associated control packet called the burst header packet (BHP) and the BHP is transmitted ahead of time in order to configure the core switches along the bursts' route. BHPs carry information about the data burst, such as source, destination, burst duration, and offset time. Offset time is the amount of time by which the data burst and BHP are separated at the source and at the subsequent intermediate nodes. The offset time allows for the BHP to be electronically processed at each intermediate node before the data burst arrives. The BHP configures the optical cross-connects (OXC) so that the data burst can cut-through without incurring any delay. Data channel bandwidth is reserved only for the burst duration using a reservation technique called just-enough-time (JET) [1].

The primary issue in the OBS core network is contention resolution, since the core does not have any buffers. Contention occurs when two or more bursts contend for the same output port at the same time. There are several contention resolution techniques, such as FDLs, wavelength conversion, deflection routing, burst segmentation [2], retransmission [3], and forward error correction (FEC) [4]. In this paper, we propose forward segment redundancy, a proactive loss recovery mechanism for OBS networks. Forward segment redundancy (FSR) mechanism is comprised of modified burst segmentation and forward redundancy (no need for complicated FEC).

TCP-based applications account for majority of data traffic in the Internet; thus understanding and improving the performance of TCP implementations over OBS networks is critical. The fundamental assumption of the various TCP versions is that the underlying physical medium is electronic and the packets experience queuing delay at IP routers. The various techniques TCP employs, to deal with and to avoid congestion, ultimately provide fairness across all flows in the network. While these techniques are successful in providing fairness in an electronic network, they are unsuccessful in an OBS network when there are random burst contentions. It is important to resolve burst contentions in OBS networks, a burst loss having multiple TCP segments could adversely affect TCP performance [3], [5].

In this paper, we will evaluate TCP performance over an OBS network with FSR. The remainder of the paper is organized as follows. In Section II we describe the proactive FSR mechanism to resolve contentions and to recover from packet losses in OBS networks. Section III discusses the simulations results and Section IV concludes the paper.

II. Forward Segment Redundancy (FSR)

In this section, modified burst segmentation and forward redundancy are combined to provide FSR. In this paper, we refer to the burst already scheduled at the output port as the *original burst* and the later arriving burst to the same port as the *contending burst*. Additionally, priority is assigned to each burst (high/low) based on the number of redundant segments contained in a burst. A burst with no redundant segments is given higher priority than a burst with redundant segments. Priorities help minimize the impact of redundant traffic (additional network load) on original traffic.

FSR mechanism involves appending redundant packets to the burst generated at the ingress node, before transmitting the burst into the core. The core node employs modified burst segmentation using tail-drop and/or head-drop policies to resolve data burst contentions. Based on the data traffic loss requirement, either complete forward redundancy $>= 100\%$ or partial forward redundancy $< 100\%$ can be implemented.

FSR involves two components: forward redundancy in the edge and modified burst segmentation in the core. FSR, being a proactive loss recovery mechanism, is better in scenarios where end-to-end delay is significant. FSR avoids feedback from the receiver to the sender about packet losses. FSR mechanism aims to eliminate the primary limitation of burst retransmission by avoiding large ingress electronic buffers at the edge and the additional delay incurred in retransmitting the original burst lost during contention in the core. Bandwidth utilization is not affected because redundant burst segments are dropped when they contend with the original burst segments already scheduled on the data channel by using burst priorities.

Traditional burst segmentation involves dropping only the overlapping segments of a burst by employing head-drop or tail-drop policy [2]. Traditional burst segmentation resolves the contention by splitting the two contending bursts into three burst segments, and scheduling only two non-overlapping

[1] This work was supported in part by the National Science Foundation (NSF) under grant CNS-0626798.

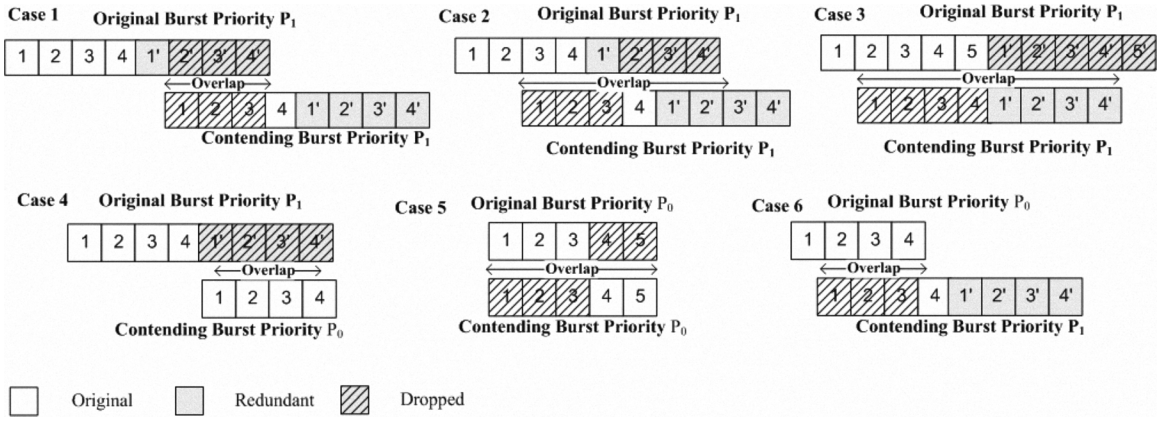

Fig. 1. Forward Segment Redundancy Scenarios.

segments of the three resultant segments on the outgoing data channel. For example, in tail-drop policy, the three burst segments are original burst's head, original burst's tail (dropped), and entire contending burst. In the proposed modified burst segmentation mechanism, contention resolution could result in four burst segments: head and tail burst segments of both contending burst and original burst. Out of the four resultant segments, only two non-overlapping segments are scheduled and the other two are dropped. Based on the contentions scenario, we choose the segments that should be scheduled and the segments that should be dropped.

We now illustrate the different contention scenarios using two burst priority classes. We assign higher priority, P_0, to burst without any redundant segments and lower priority, P_1, to burst with redundant segments. FSR mechanism for the different contentions scenarios are illustrated as shown in Fig. 1. For the sake of this illustration, we assume that the initial forward redundancy is 100% for each burst. We describe six possible contention scenarios in the presence of two burst priorities with four equal priority contentions and two different priority contentions. For equal priority contentions, we perform both head-drop and tail-drop simultaneously. In Fig. 1, Case 1, Case 2, Case 3, and Case 5 contention scenarios have original and contending bursts with the same priority. Hence, we implement both head-drop and tail-drop policies simultaneously.

In Case 1, the original burst has the same priority as the contending burst. The original burst and the contending burst have equal number of segments overlapping each other. We note that this case might not arise if we do not have FSR to begin with, but with FSR we see that there is no original segment loss because we drop only the overlapping segments (overlap $\leq$ redundancy).

In Case 2, the overlap is the same among the contending burst and the original burst. The contending burst length $>$ overlap $>$ redundancy, here we selectively drop only the overlapping redundant segments from both the bursts. There is no original segment loss in this scenario.

In Case 3, the overlap is greater than contending burst length (overlap $>$ contending burst length). We can still avoid original segment loss in this tough scenario. The original burst experiences tail-drop and the contenting burst experiences head-drop.

In Case 5, the original burst and contending burst have the same length and same priority. In order to be fair, we drop half the overlapping packets from the original burst and the remaining half from the contending burst. In this specific scenario, FSR cannot avoid original segment loss. If necessary, higher-layer TCP has to recover the lost segments.

We observe that original segments are completely recovered in most of these contention scenarios. In Fig. 1, Case 4 and Case 6 have original and contending bursts with different priorities. Therefore, depending on the priority of the bursts either a simple tail-drop or head-drop is performed.

The FSR implementation may need to reorder the burst segments that are delivered out-of-order due to head-drop policy. A simple reordering mechanism can be done at the egress node. If out-of-order TCP segments are received at the egress node, it may lead to unnecessary triple duplicates being sent back to the TCP source.

III. Simulation Results

Simulations were performed using ns2 and OWns [6] module to simulate OBS networks using a dumbell network with five edge nodes on either side (Fig. 2(a)). Each of the five nodes on the left has 4 TCP SACK flows sending a 100MB file. The following parameters have been used to obtain the results: Eight data wavelengths are used, each having a transmission rate of 1Gb/s. The propagation delay from edge nodes to core nodes is 10ms while the delay between the two core nodes is 20ms. Latest available unused channel (LAUC) scheduling is used for all the simulations. Packet length is 1KB. A timer and threshold based burst assembly mechanism is implemented with an assembly timer of 10ms and maximum burst size of 1MB.

At the ingress node, certain segments of the assembled data burst will be replicated. The redundant data segments are appended to the original data burst. The amount of data to be replicated depends upon the forward redundancy value chosen at the ingress. In this implementation we have evaluated 100% redundancy. The replicated data is appended at the tail of the original burst in a serial manner. The BHP contains the burst label (same for BHP and corresponding data burst), original data size, redundant data size, offset time, and burst priority. We simulate burst contentions at the left core node. Each simulated burst contention has an equal probability of being a head or tail-drop with a random amount of data between 1% and 100% dropped from the burst. The amount of data

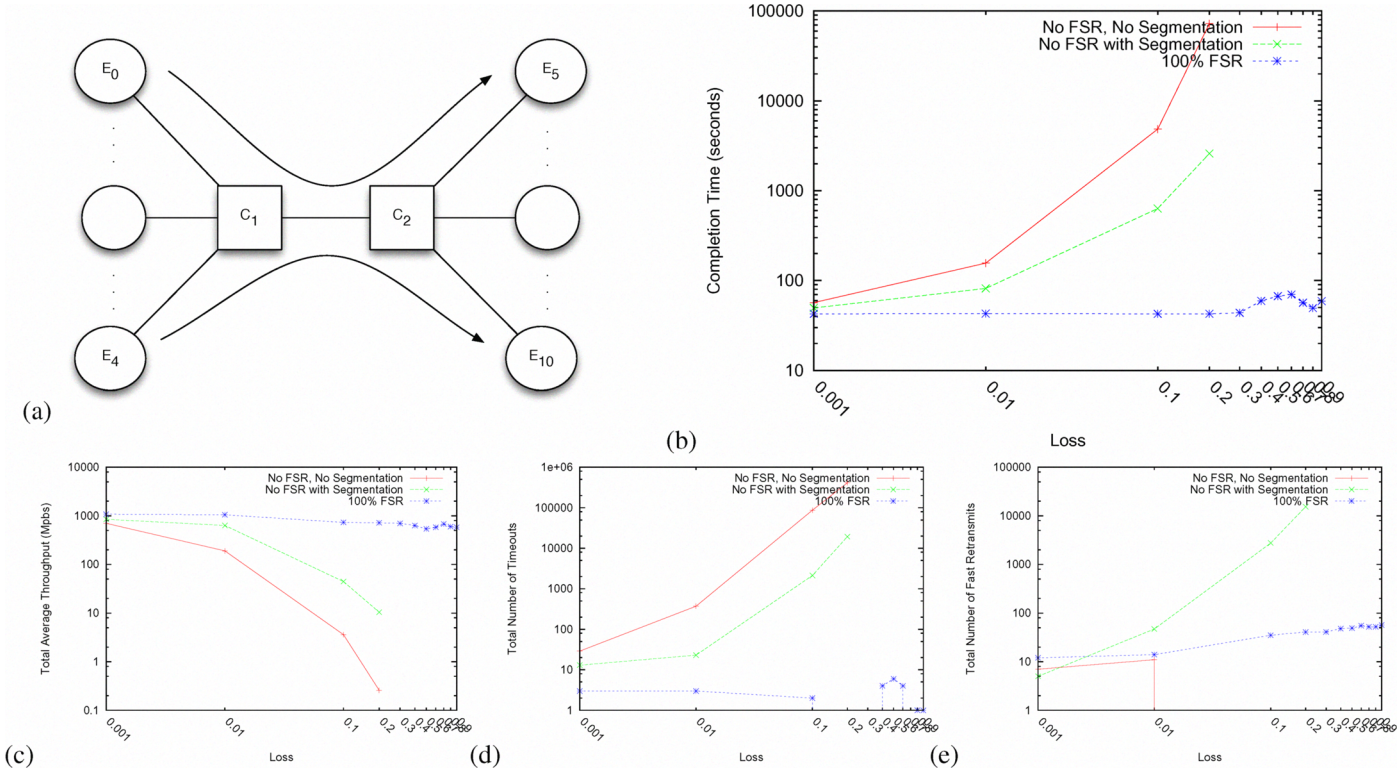

Fig. 2. Comparison of TCP performance with and without FSR versus burst loss probability (0.001-0.9). (a) Simulation network. (b) Completion time marks when all flows have finished sending their 100MB file. (c) Total average throughput across all TCP flows. (d) Total number of timeouts experienced across all TCP flows. (e) Total number of fast retransmissions experienced across all TCP flows.

dropped depends on the current size of the burst (including redundant data). The burst is reordered at the egress node using packets from the original data segments and the redundant data segments. The reordered burst is then sent to the upper layer.

In the simulations we compare 100% FSR, 0% FSR with burst segmentation, and 0% FSR with no segmentation. The loss values on the x-axis of each graph show the probability of a random contention occurring at the core node. For simulations with no segmentation, this loss value indicates the percentage of bursts completely dropped instead of segmented.

In Fig. 2(b), we compare the completion time of the simulations sending data over FTP. We can see a significant improvement using 100% FSR. Up to three orders of magnitude compared to 0% FSR without segmentation even at high loss. We can see the completion time does not increase with 100% FSR even as the probability of a random contention increases. This is because when a random contention occurs, 1% to 100% of the burst is dropped, or 50% on average. Therefore, as the probability of a random contention increases, the effective loss approaches 50% of the burst. The 100% redundant data can cover this loss so completion time does not suffer. TCP must resend the lost data for the bursts that do not contain redundant data though.

In Fig. 2(c) we see the total average throughput across all TCP flows. We can see that with redundant data TCP achieves much higher throughput. This can be explained by looking at the number of timeouts and fast retransmissions in Fig. 2(d) and Fig. 2(e). 100% FSR TCP flows have very few timeouts because of the redundant data TCP throughput does not suffer, while we can see the number of timeouts increase significantly for no FSR. There are some fast retransmissions because of segmentation, but these do not cause TCP to significantly reduce its send rate. We hardly see any fast retransmissions for no segmentation because the entire burst is being dropped in the case of a contention.

IV. Conclusion

We have proposed a new proactive loss recovery mechanism called forward segment redundancy (FSR), that minimizes segment loss during burst contentions in the core and also recovers from segment loss in the forward direction using redundant segments placed in each data burst. In FSR mechanism, redundant segments are added to each burst assembled at the ingress node before transmission to the destination. The segment losses due to contentions are minimized using a modified burst segmentation mechanism, and most lost segments are recovered in the forward direction using redundant segments. Using 100% FSR, we observed up to three orders of magnitude improvement in TCP performance over OBS networks. An interesting area of future work is to determine the optimal FSR value for a given network. Development of analytical loss models for FSR will also be useful.

References

[1] C. Qiao et. al, "Optical burst switching (OBS) - a new paradigm for an optical Internet," *Journal of High Speed Networks*, 1999.
[2] V. M. Vokkarane et. al., "Burst segmentation: An approach for reducing packet loss in optical burst switched networks," *Optical Networks*, 2003.
[3] Q. Zhang et. al., "Analysis of TCP over optical burst-switched networks with burst retransmission," *IEEE Globecom*, Nov. 2005.
[4] H. Overby, "The network layer packet redundancy scheme: A novel approach to reduce packet loss in OPS networks," in *ONDM*, May. 2006.
[5] X. Yu et. al, "TCP implementations and false time out detection in OBS networks," in *IEEE INFOCOM*, Mar. 2004.
[6] "OBS-NS simulator: http://wine.icu.ac.kr/õbsns/index.php," .