Burstiness of Output Traffic by a Split Connection TCP Implementation

Toshihiro Shikama†

†Fukui University of Technology, Japan
shikama@fukui-ut.ac.jp

Abstract - This paper studies a TCP split connection implementation that employs separate TCP connections for a wired section and a wireless section, where a node connecting both sections forwards data from one section to another. We assume that transmission errors occur in the wireless section. From the results by the simulations using ns-2, this paper shows that the implementation produces large forward data due to losses of packets over the wireless section. It also shows that the output traffic from the forwarding node becomes bursty due to the large forward data. The effects of the bursts of packets are evaluated for the case where the output traffic from the forwarding node is fed into a node connected to a bottleneck link in the wired section. We found that a large number of packets are queued even when a packet loss rate is relatively small. The problem of bursts by the forwarding node has not been identified yet and its mitigation is left for further study.

Keywords: PEP, TCP Proxy, Split Connection, Bursty Traffic, Reassembling

1 Introduction

Transmission errors have to be considered in IP networks that employ wireless links. Although TCP is mainly employed end-to-end in the Internet, it is well understood that TCP cannot achieve sufficient throughput in the environment where packets are lost due to transmission errors. TCP assumes that losses of packets are derived from a buffer overflow at a forwarding node; it invokes the congestion control to reduce traffic. In the case where losses of packets are caused by transmission errors, the congestion control is performed unnecessarily; the throughput of TCP is lowered.

One approach to mitigate this problem is to improve the congestion control of TCP itself. A typical version of TCP is Westwood [1][2], which estimates an available bandwidth. When it detects a packet loss, it identities the cause of the loss by checking whether the available bandwidth is larger than the actual transmission rate. If this is the case, the congestion control is not invoked.

Another approach is to employ a PEP (Performance Enhancing Proxy) that terminates a TCP connection from a source terminal and establishes another TCP connection to the destination terminal [3]. It forwards receive data from one TCP connection to another one. As the large round-trip delay causes low throughput and the PEP makes the delay short by splitting the TCP connection, the throughput is improved. PEPs are traditionally employed in satellite networks where a propagation delay is large. However, it is also effective to improve throughput of terrestrial radio networks of which transmission rate is increasing rapidly [4].

Although the termination of a TCP connection by the PEP has a significant impact on the characteristics of IP flow, a study on this aspect has not been done so far. This paper focuses on the property of output traffic produced by the PEP and identifies the problem concerning bursts of packets caused by the termination of a TCP connection at the PEP.

The rest of this paper is organized as follows: Section 2 explains the architecture of the split TCP connection and its problem regarding output traffic. Section 3 describes the simulation model and associated assumptions. Section 4 describes simulation results and discussion. Finally our conclusion is presented in Section 6.

2 Output Traffic by a Split TCP Connection Implementation

This paper assumes that a PEP is placed in a node which connects a wireless link to a wired network. In the following description, we introduce the term “a wireless section” to refer to the wireless part. We also call the wired network part “a wired section”. As the PEP splits the round trip delay of an end-to-end TCP connection, an error recovery of a lost segment over the wireless section is done quickly. This leads to the improvement of the total TCP throughput.

Figure 1 illustrates an example of error recovery sequence by the PEP placed between the wireless section and the wired section. When a loss of a packet occurs on the wireless section, an error recovery by a retransmission of associated segment is performed by TCP over the wireless section. The figure assumes that the fast retransmit is invoked after three duplicate ACKs. If a receiving side of TCP in the PEP accepts out-of-order segments after the lost segment, it retains them in its receive buffer to reassemble them. When the lost segment is retransmitted and received correctly, the reassembling of data is completed; the whole data consisting of the lost segment as well as the buffered segments is forwarded to the next TCP connection over the wired section. In this paper, we call this data “forward data”. The size of the forward data might be up to the window size of TCP over the wireless section, a large number of segments might be generated at the same time. In Figure 1, a burst of packets immediately issued by the PEP after the arrival of the retransmitted segment. The burst of packets might be harmful, since it may cause buffer overflows at nodes forwarding these packets on the wired section.

With regard to this busty packets from TCP, RFC 2581 specifies that TCP should employ the slow start after a silent period of more than one RTO (Retransmission Time Out) [5]. In the normal case, when an application program issues a send request of a large amount of data, the transmission of seg-

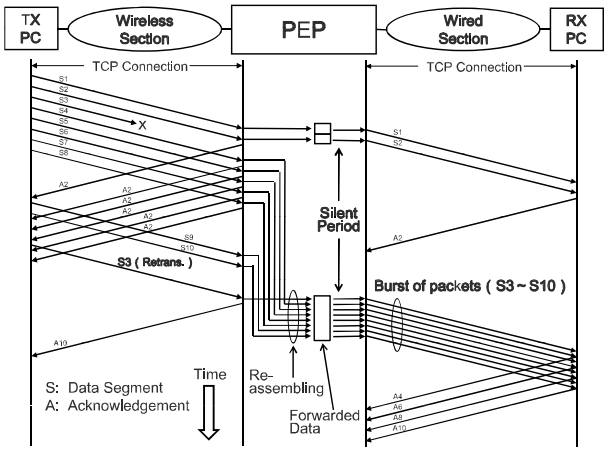

Figure 1: Data forwarding by a PEP and its output traffic

ments begins with the slow start and reaches to the self clocking state after some transient period [6]. The burst of packets occurs during the period of the slow starts only and its effect is considered to be limited.

In Figure 1, there is a silent period over the wired section from the loss of the packet to the reception of the retransmitted packet. However the duration of this period is comparable with one round-trip time of the wireless section, which is much less than RTO period. Therefore the slow-start is probably not invoked in this case; the burst of packets derived from the large forwarding data is likely fed into the wired section.

3 Simulation Models

This paper evaluates output traffic from the PEP by simulations using ns-2 [7]. Figure 2 shows the simulation model for the case where the PEP is employed. NewReno TCP is employed for both the wireless and wired sections. Node n1 represents the PEP that terminates a TCP connection over the wireless section (from node n0 to node n1) and forwards received date to the next TCP connection over the wired section (from node n1 to node n3). Since original ns-2 does not have the function of delivering received date to the upper application layer, we added this function to TCP sink in ns-2. This is indicated as "TCP Sink+" in Figure 2. We measured the distribution of the sizes of forward data inside the PEP and the number of packets included in a burst sent by the PEP.

Figure 3 represents the relationship between forward data inside the PEP and bursts of packets issued by the PEP. Since TCP provides the byte oriented streaming service, TCP segments received by the PEP are reassembled to variable length data, which is forwarded to the next TCP connection. The next TCP that accepts the data divides it into segments. These generated segments are sent over the wired section as packets. A series of packets sent back-to-back at the rate of the interface to node n2 is called a burst. Since the number of packets in a burst is affected by the size of the congestion window of TCP and the number of outstanding segments, this number is not necessarily consistent with the size of the forward data.

The bandwidth of the wireless section is assumed to be 5 Mbit/s, while the PEP is connected to node n2 with the interface rate of 100 Mbit/s and node n2 is connected to destination node n3 with the rate of 6 Mbit/s or 10 Mbit/s, which are assumed to be the bottleneck bandwidth over the wired section. Since there is a difference of packet transmission rate between input and output links in node n2, bursts of packets might be queued in this node. We measured the number of packets in a burst on the 100 Mbit/s interface between nodes n1 and n2.

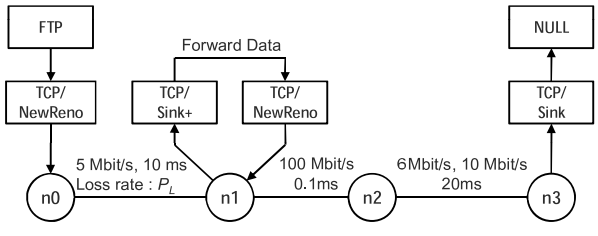

Figure 2: Simulation model for the case with PEP

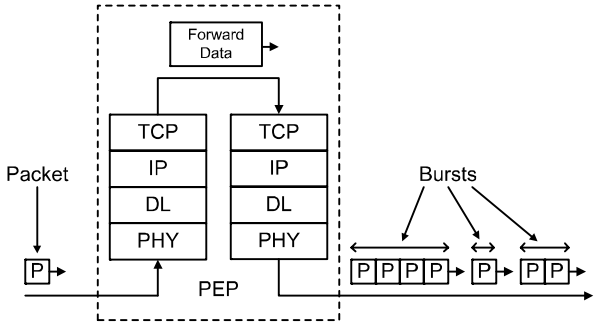

Figure 3: Relation between forward data and bursts of packets

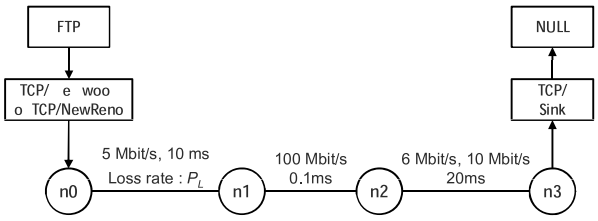

Figure 4: Simulation model for the case without PEP

In order to confirm the improvement of throughput by the PEP we also simulate the case where a PEP is not employed. Figure 4 shows the model of no PEP, where either NewReno TCP or Westwood TCP is employed end-to-end for comparison. Table 1 summarizes the simulation parameters.

4 Simulation Results and Discussion

4.1 Throughput

Figure 5 shows the relation between the packet loss rate and the throughput. In this figure NOPEP (W) represents the no PEP case where Westwood TCP is employed end-to-end, while NOPEP (N) represents the no PEP case where NewReno TCP is employed end-to-end. Throughput generally decreases as the packet loss rate becomes large. However, the decrease of throughput is suppressed significantly by the PEP. In the cases of no PEP, where an error recovery is per-

Table 1: Simulation parameters

Bandwidth of the wireless section B_R	5 Mbit/s
Delay of the wireless section	10 ms
TCP for the wireless section	NewReno, Westwood
TCP window size for the wireless section	128 KB
Generation of packet losses	random
Packet loss rate P_L	0.0001, 0.0002, 0.0005, 0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1
Bandwidth of the wired section B_W	6 Mbit/s, 10 Mbit/s
TCP for the wired section	NewReno
TCP window size for the wired section	256 KB
Delay of the wireless section	20 ms
The number of TCP connections N_{TCP}	1, 5
Interface rate from the PEP to the wired section	100 Mbit/s
The number of buffers at each node	unlimited
Simulation time	1500 sec
The number of simulation runs	12

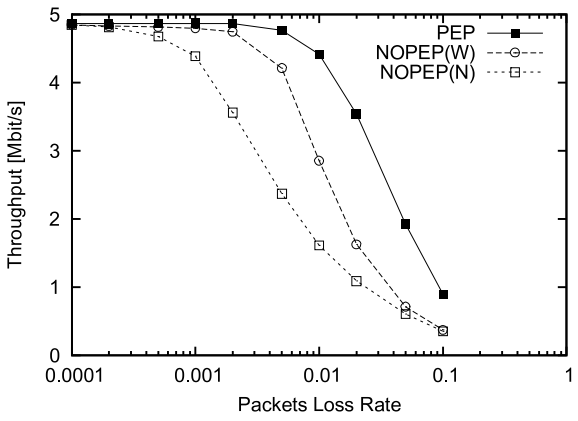

Figure 5: Packet loss rate vs. throughput

formed end-to-end, it is clear that Westwood TCP improves the throughput compared with NewReno TCP.

4.2 The distributions of the forward data sizes and the number of packets in a burst

Figure 6 shows the distributions of both the forward data sizes and the number of packets in a burst, where the number of TCP connections N_{TCP} is 1 and the packet loss rate P_L is changed to 0.0001, 0.001, 0.01, and 0.1. As the original size of forward data is measured in bytes, the values of data size indicated in the figure are divided by MSS (Maximum Segment Size: 1460 bytes) to compare the burst size. In this figure rectangle boxes represent the distribution of the forward data sizes, while black dots show the distribution of the number of packets in a burst.

The maximum size of forward data is about 90, which is consistent with the TCP window size (128 kB) of the wireless section. We can observe the trend that the distribution of the forward data size and the number of packets in a burst are well consistent. This means that the PEP generates bursts of packets corresponding to the forward data sizes. In the cases of small packet loss rates, although the frequency of bursts is generally small, there is a trend that large size bursts are generated.

Figure 7 shows the same distributions as Figure 6, where the number of TCP connections N_{TCP} is changed to 5. Similar trends are observed as the case of the single TCP connection. However, when the packet loss rate is small ($P_L = 0.0001, 0.001$), the distribution of the number of packets in a burst is different from that of the forward data sizes. Frequency of bursts of which size is larger than 1 becomes small as compared with the distribution of the forward data sizes. The reason is as follows: as the number of TCP connections increases, the number of outstanding segments waiting for ACKs on the wired section becomes large. In this case, a new segment can be sent only after a new ACK arrives and the window is updated and opened by the ACK. This means that the self clocking state likely continues in the cases of small packet loss rates.

4.3 The distribution of inter-arrival times of forward data

Figure 8 shows the distribution of inter-arrival times of forward data in the case where the number of TCP connections N_{TCP} is 1 and the packet loss rate P_L is 0.001. In this small packet loss case, the inter-arrival times take the values of less than 200ms. Since the minimum value of the RTO time is 200ms and retransmissions by the RTO timer needs more than 200ms, the distribution shows that the recovery of lost segments is done by the fast retransmit and not by a timeout.

4.4 The correlation between the inter-arrival times and the forward data sizes

It can be expected that a strong correlation may exist between the sizes and inter-arrival times of forward data. Figure

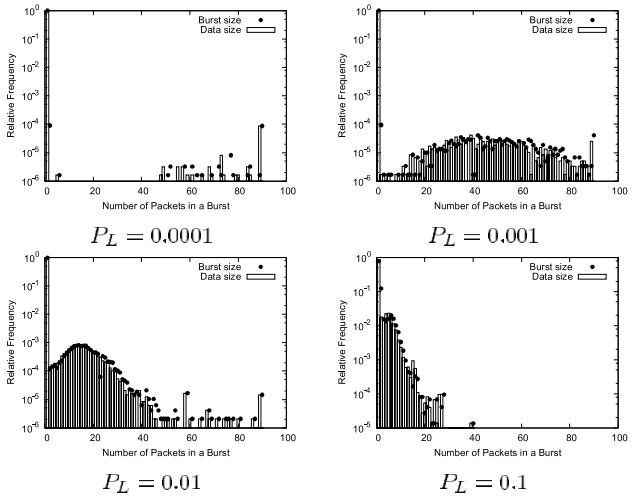

Figure 6: Distribution of the forward data sizes and the number of packets in a burst, where N_{TCP} is 1.

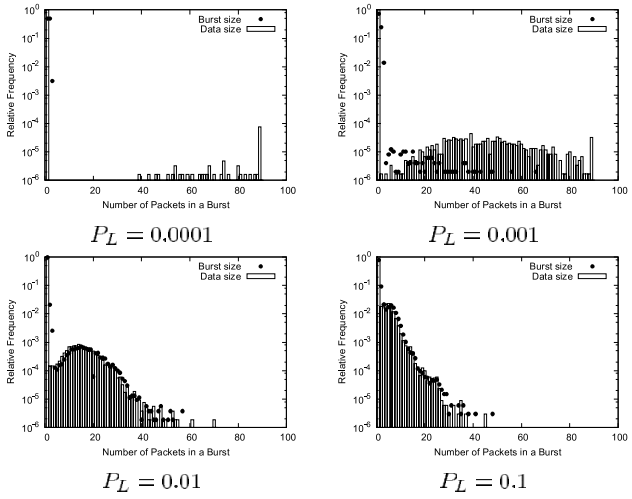

Figure 7: Distribution of the forward data sizes and the number of packets in a burst, where N_{TCP} is 5.

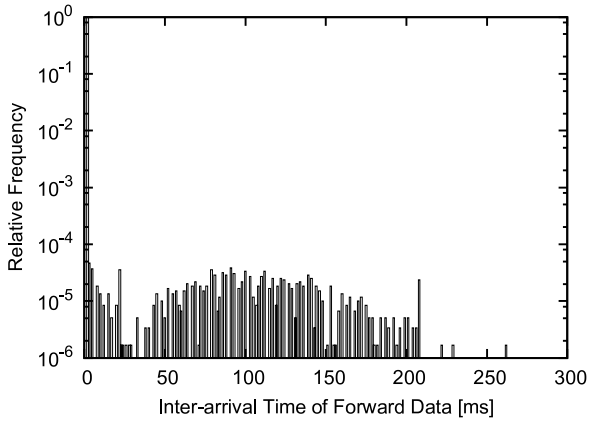

Figure 8: Distribution of inter-arrival times of forward data, where P_L is 0.001 and N_{TCP} is 1.

9 shows the scatter diagram of the pair of these two values, where the number of TCP connections N_{TCP} is 1 and the packet loss rate P_L is 0.001. We can observe the strong correlation between the pair of two values; as the inter-arrival time of forward data becomes large, the forward data size also increases. In this condition, a lost segment is mainly recovered by the fast retransmit; the number of segments received correctly after a lost segment is approximately proportional to the interval needed to recover the lost segment. This is considered to be the reason why the strong correlation exists between the sizes and inter-arrival times of forward data.

Figure 10 also shows the same scatter diagram as Figure 9, except that the packet loss rate P_L is 0.01. When the inter-arrival times are less than 120 ms, the strong correlation can be observed, however the dispersion of plotted data is larger than the case of Figure 9. The correlation cannot be observed for the inter-arrival times larger than 220ms. This is the effect of retransmissions by timeouts. When the packet loss rate becomes large, the lost segments cannot be recovered by the fast retransmit only, the possibility of retransmissions by timeouts increases.

Figure 11 shows the time changes of cwnd, where the packet loss rate P_L is 0.001 and the number of TCP connections N_{TCP} is 1. This figure indicates that the fast retransmit is employed for retransmissions. Figure 12 shows the same cwnd changes, where the packet loss rate P_L is 0.01. In this figure, retransmissions by both the fast retransmit and timeout are observed. The observation of these figures of cwnd is consistent with the scatter diagrams shown in Figures 9 and 10.

Figure 13 shows the scatter diagram of the same condition as Figure 9, except that the number of TCP connections N_{TCP} is changed to 5. Since an inter-arrival time of forward data on one TCP connection is fragmented by the arrivals of forward data from other 4 TCP connections, the correlation between the sizes and inter-arrival times of forward data becomes small. However, plotted circles tend to be distributed around the upper side of the correlation line observed in Figure 9 because of the fragmentation of an inter-arrival time.

The queueing analysis for bursty traffic has been studied such as $M^{[X]}/G/1$ and $G^{[X]}/G/1$ [8]. It is expected that we can apply such results to evaluate queueing effects of the bursty traffic at a node forwarding the traffic. However, the analysis generally assumes that the inter-arrival times and the burst sizes are independent (no correlation). The strong correlation observed in Figures 9 and 10 indicates that we cannot apply the analytical approach to evaluate queueing effects at the forwarding node.

4.5 The distribution of queue length at the bottleneck node in the wired section

The effect of the bursts of packets is typically observed as large queueing at a bottleneck node forwarding the bursts. Figures 14 and 15 show the distributions of queue length at the bottleneck node n2, where the number of TCP connections N_{TCP} is 1 and 5, respectively. Each figure shows the cases where the packet loss rate P_L is changed to 0.0001, 0.001, 0.01, and 0.1. In Figure 14, although the bandwidth of the wired section B_W is 1.2 times larger than that of the wireless section B_R, the large queue length is observed even in the case where the packet loss rate P_L is small (0.0001 and 0.001). When the number of TCP connections becomes 5, the maximum queue length becomes slightly larger compared

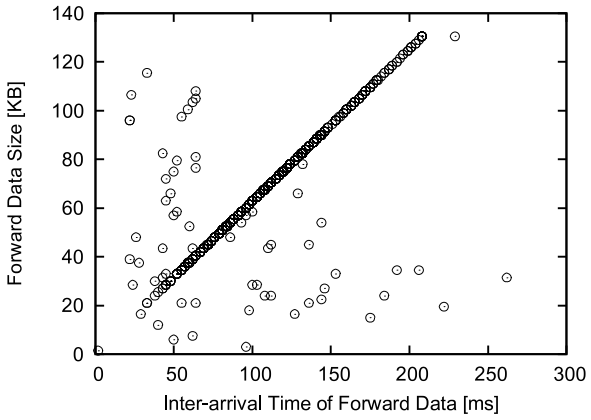

Figure 9: Scatter diagram of the sizes and inter-arrival times of forward data, where P_L is 0.001 and N_{TCP} is 1.

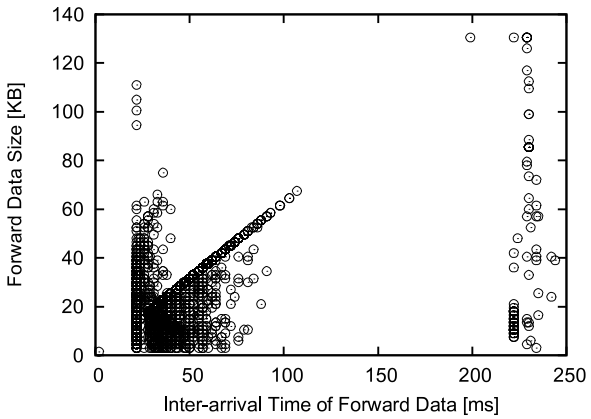

Figure 10: Scatter diagram of the sizes and inter-arrival times of forward data, where P_L is 0.01 and N_{TCP} is 1.

with the cases of the single TCP connection. However, there is not significant difference on this value.

4.6 The average of queue length at the bottleneck node in the wired section

Figures 16 and 17 show the average queue length and its standard deviation, where the bandwidth of the wired section B_W is 6 Mbit/s and the number of TCP connections N_{TCP} is 1 and 5, respectively. These figures also indicate 95% confidence interval for each plotted value.

When the number of TCP connections N_{TCP} is 1, the average queue length takes the maximum value at around the packet loss rates of 0.001 and 0.002. The standard deviation also becomes the largest when the packet loss rate is around 0.001, and its value is about 2 times larger than the average value. This means that the variation of the queue length is large. When the number of TCP connections N_{TCP} is 5, the same trend is observed, but the standard deviation is relatively small compared with the average value.

Figures 18 and 19 show the same values as Figures 16 and 17, where the bandwidth of the wired section B_W is increased to 10 Mbit/s. When we compare Figures 18 and Figures 16 where the number of TCP connections is 1, the average queue length in Figure 18 is decreased to about a quarter of the values in Figures 16, while the standard deviation is decreased to a half. Although the average queue length becomes small, the variation of queue length relative to the average value tends to increase.

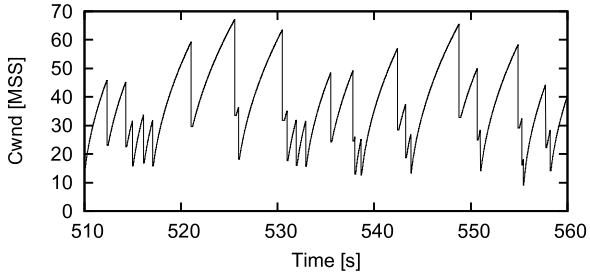

Figure 11: Example of cwnd change, where P_L is 0.001 and N_{TCP} is 1.

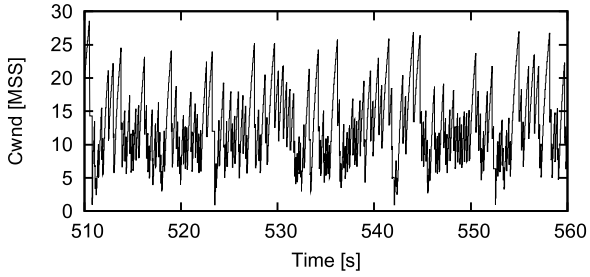

Figure 12: Example of cwnd change, where P_L is 0.01 and N_{TCP} is 1.

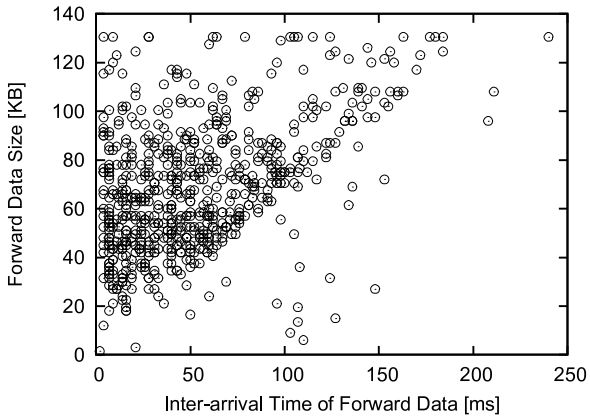

Figure 13: Scatter diagram of the sizes and inter-arrival times of forward data, where P_L is 0.01 and N_{TCP} is 1.

If the PEP is not employed and the error recoveries are performed end-to-end, the queueing at the bottleneck node never occurs. Although the results of no PEP case are not plotted in the figures, the queue length always takes value 0. It is clear that a split TCP implementation causes significant queueing at a bottleneck node because of the bursts of packets issued by the PEP. Mitigation of this problem is left for further study.

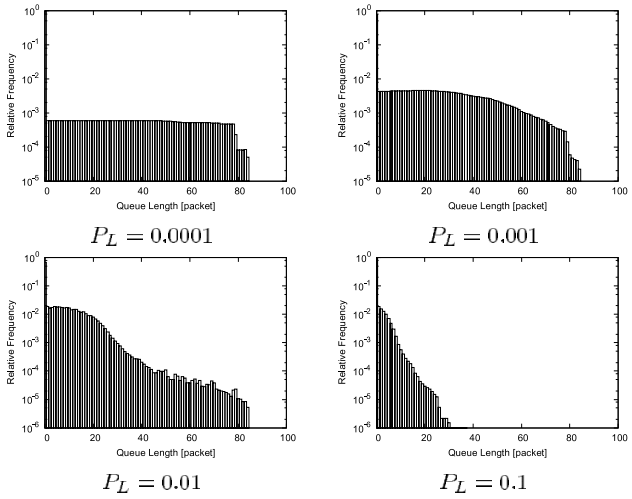

Figure 14: Distribution of the queue length, where N_{TCP} is 1 and B_W is= 6 Mbit/s.

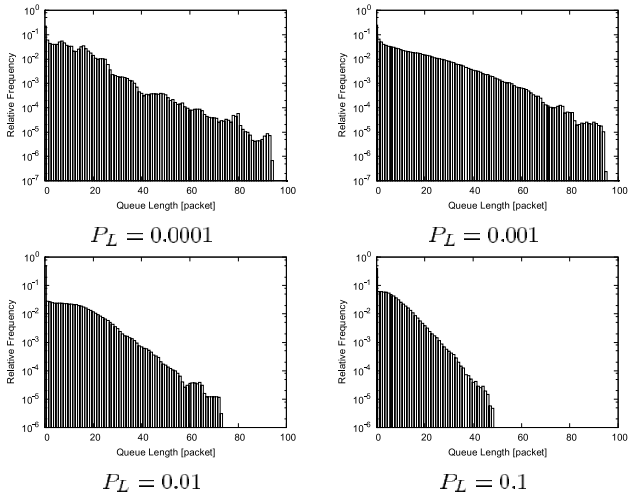

Figure 15: Distribution of the queue length, where N_{TCP} is 5 and B_W is 6 Mbit/s.

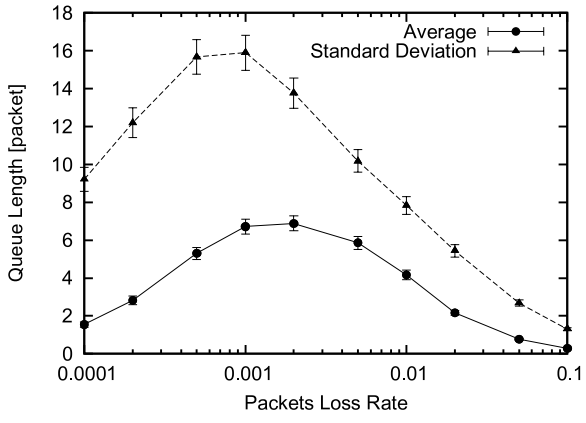

Figure 16: Average and standard deviation of queue length, where N_{TCP} is 1 and B_W is 6 Mbit/s.

5 Possible approaches to mitigate the burstiness of the PEP output

As described in section 4, the output traffic by the PEP becomes bursty in cases where packets are lost by transmission errors. A further study is needed to mitigate this problem. Our goal is not only to reduce the burst sizes on the output link but also to suppress the large queueing at the bottleneck node in the wired section. There are three possible approaches considered.

The first approach is to utilize the "maxburst" parameter. There are TCP implementation and its variant supporting the

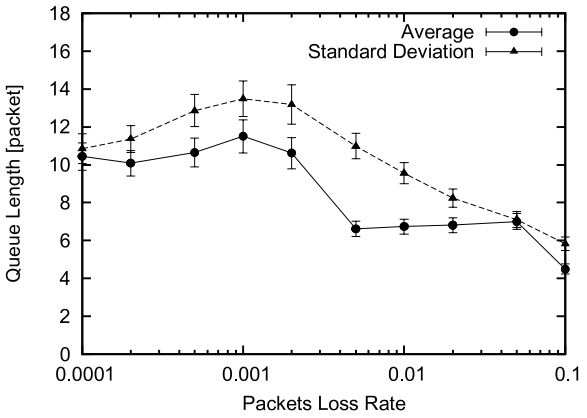

Figure 17: Average and standard deviation of queue length, where N_{TCP} is 5 and B_W is 6 Mbit/s.

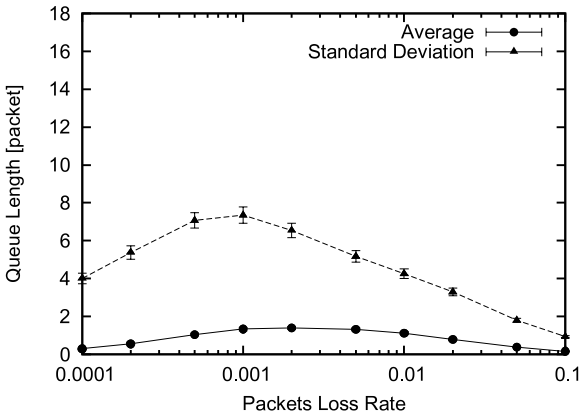

Figure 18: Average and standard deviation of queue length, where N_{TCP} is 1 and B_W is 10 Mbit/s.

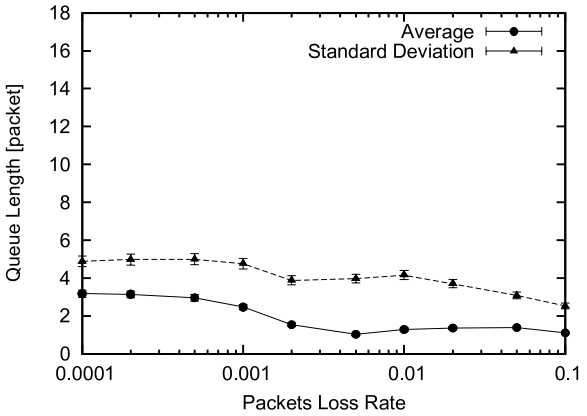

Figure 19: Average and standard deviation of queue length, where N_{TCP} is 5 and B_W is 10 Mbit/s.

"maxburst" parameter, which limits the maximum number of packets sent back-to-back on the output link when a transmission of a large size data is requested while the size of the transmission window is large enough [9], [10]. The second approach is to introduce a kind of traffic shaping between two TCP connections. The third approach is to change the PEP architecture, where an out-of-order segment is forwarded without being reassembled by the PEP.

The first approach is simple, but its effectiveness on suppressing the large queueing at the bottleneck link is questionable. Although the second approach is straightforward, it hard to define the suitable shaping rate. In this approach we also have to consider the coordination of the TCP flow control between two TCP connections to avoid buffer overflows in the PEP. Since the TCP implementation in current ns-2 does not include the dynamical change of the available window at the receiver, we have conducted the simulations on condition that throughput of the wired section is always larger than the wireless section. Further modification of ns-2 is required to evaluate the coordination. The third approach is interesting since bursts of packets never occur. However it seems that the implementation becomes very complex and hard.

We are now performing studies and performance evaluations concerning these approaches.

6 Conclusion

This paper has investigated the architecture of the split connection TCP implementation. We have focused on the output traffic where the PEP that forwards receive data from one TCP connection on the wireless section to another TCP connection on the wired section. When losses of packets are caused by transmission errors, we have observed that the output traffic from the PEP becomes bursty due to the reassembling function done by the receiving side of TCP covering the wireless section. We also have found that the burstiness becomes significant at packet loss rates that are relatively small, and the effects of the bursts become small as the packet loss rate increases. We have observed the strong correlation between the inter arrival times of bursts and burst sizes. This means that analytical approaches cannot be applied to evaluate the queueing effects at a node forwarding the bursty traffic. We simulated the case where the output traffic from the PEP is forwarded by a node that is connected to a bottleneck link in the wired section. We have found that a large number of packets are queued when the packet loss rate is relatively small.

All these observations concerning the PEP have not been identified yet. As the rate of wireless links is increasing, the PEP architecture will be important to attain high TCP throughput. Accordingly, effects of the bursty traffic by the PEP become also significant. Further studies and evaluations are needed to mitigate this problem.

Acknowledgment

This work has been supported by KAKENHI (20500079) and Special Research Grant in Aids of Fukui University of Technology.

REFERENCES

[1] Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, "TCP westwood: Bandwidth estimation for enhanced transport over wireless links," ACM MOBICOM 2001, pp.287-297, July 2001.

[2] L. A. Grieco and S. Mascolo, "Performance evaluation and comparison of Westwood+, New Reno and Vegas TCP congestion control," ACM Computer Communication Review, Vol. 34(2) , April 2004.

[3] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations," IETF, RFC3135, June 2001.

[4] M. Meyer, J. Sachs, and M. Holzke, "Performance evaluation of a TCP proxy in WCDMA networks," IEEE Wireless Communications, vol.10, no.5, pp.70-79, Oct. 2003.

[5] M. Allman, V. Paxson, and W. Tevens, "TCP Congestion Control," RFC 2581, April 1999.

[6] W. R. Stevens, "TCP/IP Illustrated, Volume 1: The Protocols," Addison-Wesley, 1994.

[7] Network Simulator - ns (version 2), http://www.isi.edu/nsnam/ns/, 2009.

[8] L. Kleinrock, "Queueing Systems, vol. 2," John Wiley and Sons, 1976.

[9] J. Iyengar, E. Blanton, and M. Allman, "TCP Burst Mitigation Through Congestion Window Limiting," draft-iyengar-burst-mitigation-00.txt, Nov. 2005.

[10] R. Stewart, L. Ong, I. Arias-Rodriguez, K. Poon, A. Caro, and M. Tuexen, "Stream Control Transmission Protocol (SCTP) Specification Errata and Issues," RFC 4460, April 2006.