Receiver Playout Buffer Requirement for TCP Video Streaming in the presence of Burst Packet Drops

X. Shen, A. Wonfor, R.V. Penty, I.H. White

Department of Engineering, University of Cambridge, 9 JJ Thomson Avenue, Cambridge, CB3 0FA, UK, xs220@cam.ac.uk

Abstract: This paper studies the buffer size required for streaming video via TCP when burst segment loss is present. The *ns-2* simulation tool is used to identify appropriate buffer sizes which are found to be closely correlated with the maximum delay induced by TCP retransmission. Three types of delay patterns are found under bursty drop conditions and expressions for maximum delay in each case are determined. TCP is found to be feasible for streaming with small bursts of dropped segments. However, large bursts of dropped segments induce multiple time-outs resulting in significant delay.

1. Introduction

There have been increasing demands for delivering video content via IP networks over recent years. Streaming video is popular as it reduces the time before the video starts to play, improving the user experience. Traditionally, User Datagram Protocol (UDP) has been used as the transport protocol of streaming video because of its timely delivery of application data [1] with low timing jitter. However, UDP lacks congestion control which is considered to be essential for maintaining Internet stability [2], does not provide a reliable transport service and hosts behind firewalls are hard to reach by UDP [3]. The success of *Youtube* and the *BBC iPlayer* shows the feasibility of Transmission Control Protocol (TCP) for streaming video. TCP provides reliability, which is essential to prevent video degradation caused by data loss, but lacks a real-time guarantee of packet retransmission as it introduces extra delay if packet drop occurs. This packet drop can trigger fast retransmission [4] and/or congestion control, depending on the number of packets dropped in a burst. However, by using a playout buffer at the receiver, fluctuations of delay can be accommodated and continuous playout can be maintained, despite the TCP induced delay jitter. It is hence important to determine the minimum amount of buffer for streaming video over TCP in the presence of packet drops, in order to create design rules for streaming video application purposes.

TCP induced delay is more severe and complex in the presence of bursty drops. Consecutive TCP segment[1] losses are more likely to happen in the real Internet owing to the way intermediate routers deal with overflow. Single packet drops usually trigger fast retransmission and induce only a small delay. However, burst losses can lead to a combination of fast retransmission and congestion control, and may even introduce a large time out delay. In this paper, the delay patterns of TCP burst segment losses in a constant bitrate (CBR) streaming video flow are studied via *ns-2* [5] simulations. Three delay pattern types are identified and a suitable regime for streaming video is given.

In related work, Kim's analytical study in [6] derives playout buffer size and the probability of buffer under-run, based on TCP throughput modelling in [7]. Work in [8] gives a discrete Markov chain model of TCP's delay and experimentally verifies the model in different loss rate for a set of round trip time (RTT) values.

The work presented in this paper is organised as follows: Section 2 describes the methodology and simulation settings. Section 3 presents results and a discussion. Finally the paper is concluded in Section 4.

2. Simulation

The streaming video system diagram is shown in Figure 1. *n1* and *n2* represent sender and receiver end hosts respectively. *r1* and *r2* are two routers. The link between *r1* and *r2* corresponds to a lossy network where burst drops occur. Server and client applications sit above end hosts *n1* and *n2*. At the receiver side, the client application stores incoming packets at the end of the playout buffer queue, while video frames are fetched from the head of the queue to display at a constant rate. To guarantee a continuous playout, the length of buffer needs to be long enough to absorb any delay fluctuation caused by retransmission. This determines the minimum buffer size to ensure continuous playout.

In the *ns-2* simulator, we have created customised video server and client applications to deliver constant bitrate (CBR) video over Reno TCP which is widely deployed [9]. Constant bitrate video is used to emphasize the effect of delay variation on buffer size. A playout buffer queue and constant rate video playing process are

[1] Here, segment means TCP segment. In this paper, the terms segment and packet are interchangeable.

defined in the video client application within *ns-2*. As the focus of this work is on burst loss performance rather than bandwidth limitation, unconstrained *100Mbps* links are used. Single and burst drops are implemented periodically on the link between routers *r1* and *r2* in Figure 1. Parameters for the simulation are shown in Table 1. Note a large period is used to ensure the effects of adjacent bursts do not overlap.

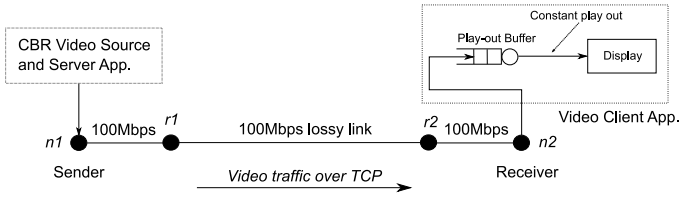


Figure 1: Streaming video system diagram

In the simulation, a set of common round trip times (RTTs) is used. Burst length values up to *20* segments are used. Network traces are recorded at the sender and receiver respectively. TCP segment delay values are computed via log files at the server and client applications respectively. For each simulation run, over *10000* packets are sent to ensure *10* bursts occur.

Notation	*Definition*	*Parameter value in simulation*
r	Video frame rate	30 *fps*
p	TCP segments per frame	1 *segment*
s	TCP segment size	1024 *bytes*
b	Burst loss length	1 ~ 20 *segments*
T	Period of burst drop	1000 *segments*
RTT	Round trip time	20 ~ 200 *ms*, 20 *ms interval*
c	CBR source rate	240 *kbps*

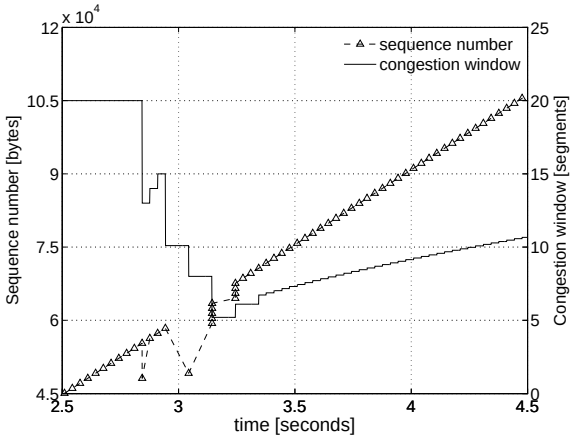
Table 1: Summary of parameters used in the simulations

From the simulations, the correlation between maximum delay and required buffer size is found. Three types of delay patterns are identified. These results are presented and discussed in the next section.

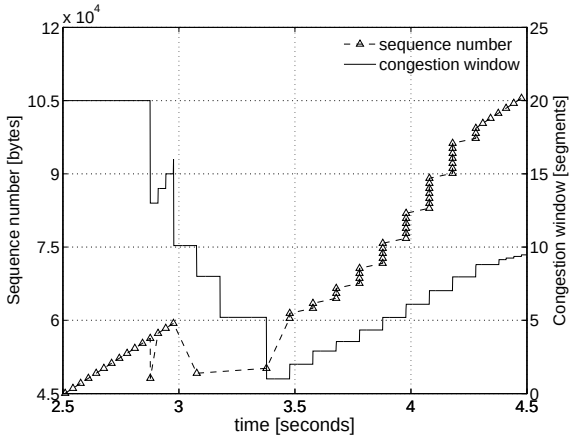
3. Results and Discussions

Three distinct delay patterns are found in addition to the well known fast retransmission mode. We call them: double fast retransmission mode (*DR*) (two consecutive fast retransmissions), double fast retransmission with time out mode (*DRTO*), and time out mode (*TO*). A single packet drop usually only triggers a single fast retransmission. The delay induced is small and the impact on subsequent packets is negligible. Figure 2(a) shows a two segment loss with two fast retransmissions. In this mode, a reduced congestion window prevents transmission of further packets after the second retransmission (at *3.2* seconds in Figure 2(a)) and this causes delay to subsequent packets as well. The maximum delay duration is dominated by the arrival time of the second retransmitted packet as it arrives much later than the first one. In Figure 2(b) three consecutive drops cause two fast retransmissions and one timeout. There is not a third fast retransmission because after the second fast retransmission and its acknowledgement (ACK), no more packets are sent owing to the current congestion window being full and the outstanding unacknowledged third packet. No more ACKs are received before expiry of TCP's retransmission time out timer (RTO). The maximum delay is determined by the sum of the double retransmission delay and the RTO duration. This delay is usually large because RTO is large enough to be tolerant of high RTT variance [10]. In this mode, more packets are affected than in the double retransmission mode.

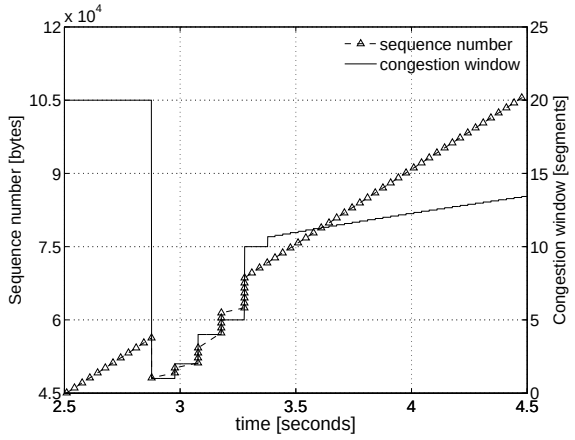
When the burst of dropped packets plus the RTT is long enough that fewer than 3 duplicate ACKs are received before the RTO expiry, the first lost packet times out. This is time-out mode phase I as shown in Figure 2(c). TCP is back to slow start and the congestion window size falls to 1 segment. Notably, in this phase the maximum delay is smaller than the double fast retransmission with time-out mode and fewer packets are delayed. However, in the phase II time out mode, the burst of dropped packets is so long that even the retransmission of the first packet is lost. The RTO timer thus increases exponentially and many packets are delayed a considerable duration, as illustrated in Figure 2(d).



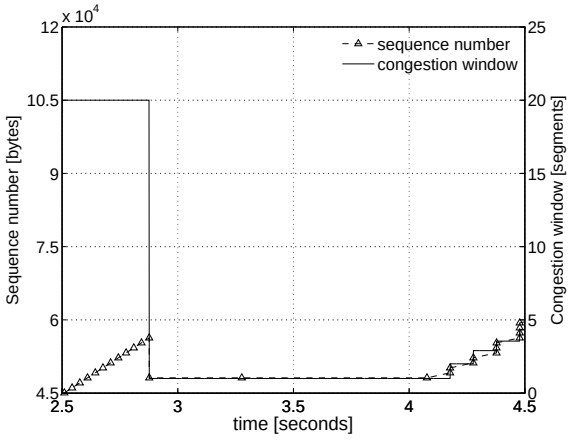
(a) double fast retransmission mode, 2 drops



(b) double fast retransmission with timeout, 3 drops



(c) phase I time-out mode, 4 drops



(d) phase II time-out mode, 11 drops

Figure 2: Delay patterns of burst drop, *100ms* RTT

The expressions of maximum delay are given in Equations (1)–(4), where (1) is described in [8] and (2)-(4) are derived from burst patterns we found. In the equations, $d_{\max}$ is the maximum segment delay time, t_p is the TCP inter-segment time, t_{RTO} is the TCP retransmission time out time, s is the delay mode state variable, *FR* (fast retransmission), *DR* (double fast retransmission), *DRTO* (double fast retransmission with time out), and *TO* (time out) corresponds to 4 delay modes, and i represents the total count that RTO timer expires in phase *II* of *TO* mode.

$$d_{\max} = \begin{cases} \frac{1}{2}RTT + 3t_p + RTT, & \text{if } s \in FR \quad (1) \\ \frac{1}{2}RTT + 2(3t_p + RTT), & \text{if } s \in DR \quad (2) \\ \frac{1}{2}RTT + 2(3t_p + RTT) + t_{RTO}, & \text{if } s \in DRTO \quad (3) \\ \frac{1}{2}RTT + \sum_i t_{RTO_i}, & \text{if } s \in TO \quad (4) \end{cases}$$

The maximum delay plot is shown in Figure 3. For all RTTs, the delay becomes significantly large after a long burst length; this is due to the exponential increase of the RTO value. For a burst length of 3 segments, a local-maximum area is observed, being due to the time out of the third lost packet. Interestingly, long RTTs can tolerate larger bursts than short RTT flows. This is because long RTTs lead to large RTO values [10], and therefore they start to experience time-outs later than short RTT flows. Increasing RTT allows longer burst-drop lengths while still in DRTO mode, explaining the enhanced local maximum region at *200ms* RTT in Figure 3.

In this work the playout buffer length is truncated at 10 seconds, causing video problems only for large burst-drops, with their associated multiple time-out delays. For moderate bursts of fewer than 10 dropped packets,

only a minimal buffer is required as shown in Figure 4. There is a strong correlation between the end-to-end delay and required buffer size, as can be seen in Figures 3 and 4.

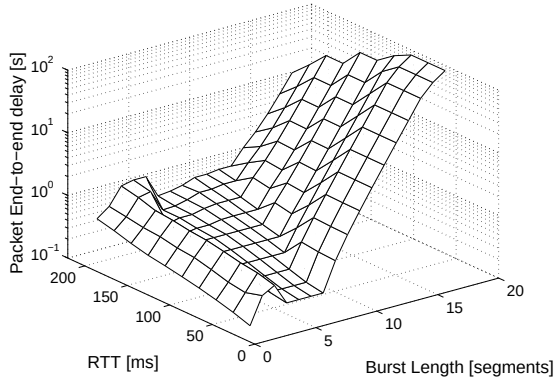


Figure 3: Maximum packet end-to-end delay

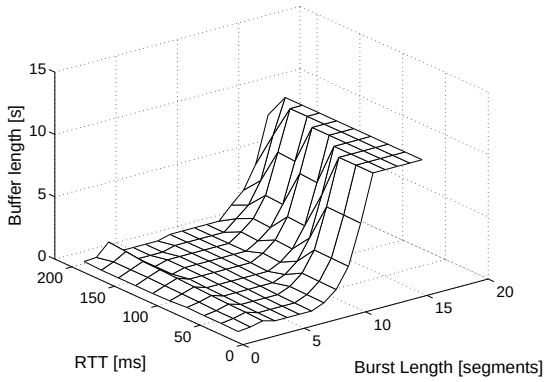


Figure 4: Minimal required playout buffer

4. Conclusions and future work.

In this paper we study the relationship between packet drop under bursty conditions and the size of playout buffer required in streaming video applications using TCP. The minimum buffer size is found to be closely correlated with the maximum TCP delay. Three distinct delay modes are identified and maximum delay expressions are derived for each mode. When small burst lengths occur, TCP can provide a reliable transport service for video streaming since its delay fluctuation can be absorbed by using playout buffer. However, large bursts that cause multiple TCP time-outs cause excessive delays which would result in delay fluctuations in excess of the buffer size.

References

[1] J. B. Postel, "RFC 768 - User Datagram Protocol," 1980.

[2] M. Handley, S. Floyd, J. Padhye, and J. Widmer, "RFC3448 - TCP Friendly Rate Control (TFRC): Protocol Specifications," 2003.

[3] P. Mehra and A. Zakhor, "Tcp-Based Video Streaming Using Receiver-Driven Bandwidth Sharing," in *Proceedings of the International Packet Video Workshop*, 2003.

[4] R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, 1st ed.: Addison-Wesley, 1994.

[5] "Vint Network Simulator version 2, http://www.isi.edu/nsnam/ns/."

[6] T. Kim and M. H. Ammar, "Receiver Buffer Requirement for Video Streaming over TCP," in *Proceedings of Visual Communications and Image Processing 2006*. vol. 6077 San Jose, CA, USA 2006.

[7] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling TCP Reno performance: a simple model and its empirical validation," *IEEE/ACM Transactions on Networking (TON)* vol. 8, pp. 133-145, April 2000.

[8] B. Eli, B. S. Abdul, R. Dan, and S. Henning, "The Delay-Friendliness of TCP," in *Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems* Annapolis, MD, USA: ACM, 2008.

[9] G. Hasegawa, K. Kurata, and M. Murata, "Analysis and improvement of fairness between TCP Reno and Vegas for deployment of TCP Vegas to the Internet," in *Proceedings of the 2000 International Conference on Network Protocols*, 2000, p. 177.

[10] V. Jacobson, "Congestion Avoidance and Control," *Computer Communication Review*, vol. 18, pp. 314-329, Aug 1988.