Receiver Buffer Requirement for Video Streaming over TCP

Taehyun Kim[a] and Mostafa H. Ammar[b]

[a] Wireless and Mobile Systems Group, Freescale Semiconductor,
Austin, TX 78735, USA
E-mail: taehyun.kim@freescale.com
[b]Networking and Telecommunications Group, College of Computing,
Georgia Institute of Technology, Atlanta, GA 30332, USA
E-mail: ammar@cc.gatech.edu

ABSTRACT

TCP is one of the most widely used transport protocols for video streaming. However, the rate variability of TCP makes it difficult to provide good video quality. To accommodate the variability, video streaming applications require receiver-side buffering. In current practice, however, there are no systematic guidelines for the provisioning of the receiver buffer, and smooth playout is insured through over-provisioning. In this work, we are interested in memory-constrained applications where it is important to determine the right size of receiver buffer in order to insure a prescribed video quality. To that end, we characterize video streaming over TCP in a systematic and quantitative manner. We first model a video streaming system analytically and derive an expression of receiver buffer requirement based on the model. Our analysis shows that the receiver buffer requirement is determined by the network characteristics and desired video quality. Experimental results validate our model and demonstrate that the receiver buffer requirement achieves desired video quality.

Keywords

Media over Networks, Video Streaming, Streaming Video Protocols

1. INTRODUCTION

A video streaming application has to employ a transport layer protocol to transmit packetized video. Since TCP is the dominant protocol in the Internet, it is reasonable to employ TCP for video streaming: recent measurement study has reported that 44% of video streaming flows are actually delivered over TCP.[11] Especially, there are many situations in which video streaming servers are located behind firewalls that permit only pre-specified port numbers. In this scenario, video streaming over TCP is the only choice to get around the firewalls over the well-known port numbers (e.g., HTTP or RTSP). Also, the reliable packet delivery of TCP is important, when error resilience is not implemented in a video codec.

While the use of TCP provides reliable video stream delivery, it is difficult to provide good quality of streaming video over TCP: 1) the sawtooth behavior of additive increase and multiplicative decrease (AIMD) incurs significant data rate variability, and 2) the use of retransmission timeouts may introduce unacceptable end-to-end delay, and the retransmitted data may be delivered too late for display.

These drawbacks of TCP can be mitigated to some extent through the use of receiver-side buffering.[3,5,9] The buffer size has to be large enough to insure that desired video quality can be achieved. In current practice, however, there are no systematic guidelines for the provisioning of the receiver buffer, and smooth playout is insured through *over-provisioning*.

We are interested in memory-constrained applications[4] where it is desirable to determine the *right* receiver buffer size. This paper, therefore, considers the question of how large the receiver buffer should be in order to achieve desired performances for streaming video over TCP. To this end, we characterize video streaming over TCP in a systematic and quantitative manner. Our starting point is an analytic model of a streaming system of CBR video. Based on this model, we quantify the receiver buffer requirement. Experimental results validate our model and demonstrate that the minimum buffering delay can achieve desired video quality.

The remainder of the paper is organized as follows. In Section 2, we present a video streaming model and derive the receiver buffer requirement. Section 3 shows experimental results to validate our model. This paper is concluded in Section 4.

2. MODEL AND ANALYSIS

2.1. Video Streaming Model over TCP

Figure 1 shows a video streaming model which consists of a sender and a receiver. We assume that the sender transmits pre-recorded CBR video over a unicast TCP connection, and the receiver is equipped with a receiver buffer in front of a video decoder. The decoder waits to fill the buffer before displaying video. There are two types of buffering delay caused by the receiver buffer.

- *Initial buffering*: to accommodate initial throughput variability or inter-packet jitters, it is needed to employ initial buffering. While a streaming application achieves more tolerance with larger initial buffering, it increases the startup latency and response time.
- *Rebuffering*: the decrease of throughput within a TCP streaming session might cause the receiver buffer starvation. When this happens, a decoder stops displaying video until it receives enough video packets. Note that rebufferings take place in the middle of video streaming, and therefore the rebuffering delay requirement for a long video stream is determined by the congestion avoidance algorithm of TCP.

In this paper, we assume that the amount of the initial buffering delay and the rebuffering delay is identical, so that the receiver buffer size for initial buffering and rebuffering is the same.

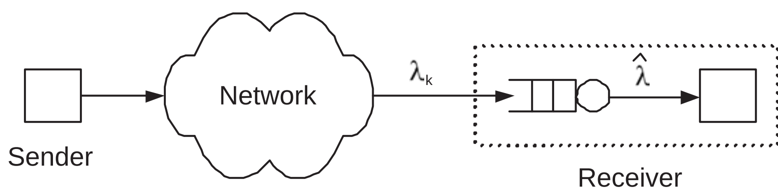


Figure 1. A video streaming model over TCP

Let λ_k be the arrival rate of video packets at round k, and $\widehat{\lambda}$ be the video encoding rate*, where a *round* is defined by a duration between the transmission of packets and the reception of the first acknowledgment (ACK) in a congestion window. It is assumed that a round is equal to the round-trip time (RTT) and independent of the congestion window size.

Figure 2 (a) shows a typical behavior of a TCP flow. We consider the TCP Reno model in this paper, since it is one of the most popular implementations in the current Internet.[7] In this model, the steady state throughput is determined by the congestion window size which is adjusted based on packet losses. A packet loss can be detected by either triple-duplicate ACKs or timeouts, where we denote the former events by *TD* and the latter by *TO*.

Consider a TD period (TDP) in Figure 2 (a). Each TDP starts immediately after triple-duplicate ACKs and increases the congestion window size by $1/b$ until triple-duplicate ACKs are encountered again†. However, when multiple packets are lost and less than three duplicated ACKs are received, a TO period (TOP) begins. In each TOP, a sender stops transmitting data packets for a timeout interval and retransmits non-acknowledged packets. Note that the timeout interval in a TOP increases exponentially until it reaches $64T_0$.

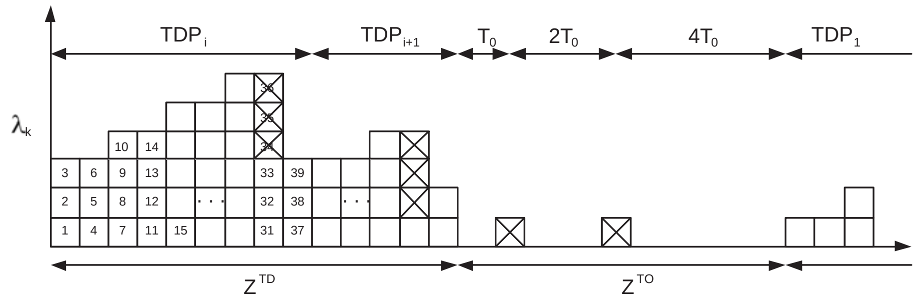
On the other hand, Figure 2 (b) shows the playout characteristic at a receiver, where it is assumed that the video playout rate is two packets worth of data per RTT. We can observe that, if a right size of receiver buffering is employed, a consistent CBR playout can be achieved without any interruption.

In this paper, the performance of a video streaming application is evaluated by the buffer underrun probability and the disruption frequency:

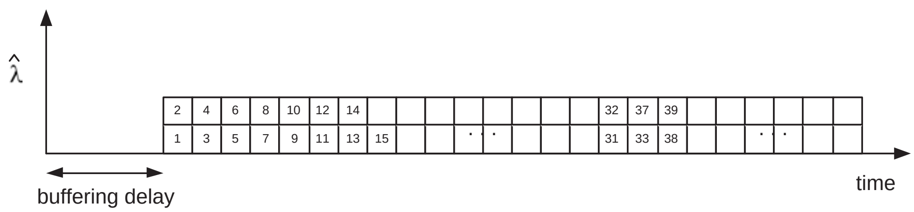
---

*Note that λ_k is a function of time specified by round k, whereas there is no subscript on $\widehat{\lambda}$, since the data rate fed into a video decoder is assumed to be CBR.

†Note that $b = 2$ if delayed acknowledgment is implemented at the receiver. Otherwise $b = 1$.



(a) Packet arrival characteristic at a receiver



(b) Playout characteristic at a decoder

Figure 2. An illustration of receiver buffering

- The *buffer underrun probability* is defined by n/N, where n is the number of buffer underrun events, and N is the number of epochs in a video stream. An *epoch* is defined by $E[Z^{TD} + Z^{TO}]$, where Z^{TO} is the duration of a TOP, and Z^{TD} is the duration between two TOPs. Note that a Z^{TD} consists of one or more TDPs.
- The *disruption frequency*[10] is defined by n/T, where n is the number of buffer underrun events, and T is the duration of a video streaming session. Since T consists of N epochs, a disruption frequency can be expressed by the ratio of the buffer underrun probability to the duration of an epoch.

2.2. Receiver Buffer Requirement

We investigate the performance when average TCP throughput matches video encoding rate. This is the case when the video encoding rate is determined by the access link bandwidth, and the available bandwidth is limited by the access link capacity. For example, many video streaming websites provide multiple copies with identical content, generated at different data rates. A receiver selects an appropriate stream that matches with the access link capacity.

We assume that the video encoding rate is equal to the average TCP throughput and does not change over time. Hence, the encoding rate is modeled by[7]

$$\widehat{\lambda} = \frac{1}{\sqrt{\frac{2bp}{3}} + \frac{T_0}{R}\min(1, 3\sqrt{\frac{3bp}{8}})p(1+32p^2)} \text{packets/RTT},$$

where p is the packet loss rate of a TCP streaming flow, R is the round-trip time, and T_0 is the retransmission timeout.

Let q_k be the receiver buffer size at round k. Since λ_k packets are received and $\widehat{\lambda}$ packets are drained in a TDP, the buffer size is given by

$$q_k = q_{k-1} + \lambda_k - \widehat{\lambda}, \tag{1}$$

where $k = 1, 2, \ldots, X_i$; and X_i is the number of round where a TD loss is detected. On the other hand, since no packet is delivered to a receiver, the receiver buffer size in a TOP is given by

$$q_k = q_{k-1} - \widehat{\lambda}, \tag{2}$$

where $k = 1, 2, \ldots, \lfloor Z^{TO}/R \rfloor$. Notations in the paper are summarized in Table 1.

Table 1. Notations

q_0	receiver buffer size at round 0
q_k	receiver buffer size at round k
q_{min}	minimum buffer size
p	packet loss rate
R	round-trip time
λ_k	arrival rate of video packets at round k: $\lambda_k = \begin{cases} \frac{W_{i-1}}{2} + \frac{k}{b} - 1 \text{ packets/RTT}, & \text{in TDP } i \\ 0, & \text{otherwise.} \end{cases}$
$\widehat{\lambda}$	video encoding rate
b	number of packets that are acknowledged by an ACK
W_i	congestion window size at the end of TDP i
X_i	number of round where a TD loss is detected
Y_i	number of packets sent in TDP i
α_i	the first packet lost in TDP i
β_i	number of packets sent in the last round
T_0	retransmission timeout
P_u	desired buffer underrun probability

We define the buffer underrun probability by the probability of the minimum buffer size to be non-positive. Since a receiver is either in a TDP or in a TOP, the buffer underrun probability at time t is decomposed into the sum of conditional probabilities, such that

$$P\{q_{min} \leq 0\} = P\{q_{min} \leq 0 | t \in Z^{TD}\} P\{t \in Z^{TD}\} + P\{q_{min} \leq 0 | t \in Z^{TO}\} P\{t \in Z^{TO}\}. \quad (3)$$

From the conditional buffer requirements in (3), we can derive the buffer size requirement under which the probability that the *unconditional* minimum buffer size goes non-positive. Theorem 2.1 below states that the minimum buffer requirement is determined by the network characteristics and desired buffer underrun probability.

THEOREM 2.1. *Given network model characterized by the packet loss rate (p), RTT (R), and retransmission timeout (T_0), the receiver buffer size (q_0) that achieves desired buffer underrun probability (P_u) is given by*

$$q_0 \geq \frac{0.16}{pP_u}[1 + \frac{9.4}{b}(\frac{T_0}{R})^2 \min(1, 3\sqrt{\frac{3bp}{8}})p(1 + 32p^2)].$$

Proof. See Appendix. □

Given receiver buffer size, required buffering *delay* is determined by $d_0 = \frac{q_0}{B(p,R)}$, where $B(p,R)$ is the steady state TCP throughput.[7] Therefore, d_0 corresponds to the time delay for buffered packets to be drained. The duration of an epoch can also be determined,[7] such that an epoch of a TCP flow is given by $E[Z^{TD} + Z^{TO}] = \frac{R(\sqrt{\frac{2b}{3p}}+1)}{\min(1,3\sqrt{\frac{3bp}{8}})} + T_0 \frac{f(p)}{1-p}$, where $f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6$.

3. EVALUATION

In this section, we present experimental results by which playout disruption characteristics are evaluated. The experimental results including simulation scripts are available in the companion website.[2]

3.1. Experimental Setup

TCP throughput dynamics are generated over a single bottleneck topology. The number of TCP streaming flows is set to 5. All access links have sufficient capacity so that any packet drop occurs at the bottleneck link: the access links have 100Mbps capacity and 1ms delay, whereas the bottleneck link has 10Mbps capacity and 60ms delay.

We run `ns-2` simulations[6] over this topology. To model the TCP throughput dynamics, we use the throughput experienced between streaming senders and receivers: the throughput is measured by counting the number of packets delivered from a sender to a receiver. All data packets are 1200 bytes long. The queue management algorithm running on intermediate routers is the random early detection (RED).

To construct dynamic network characteristics, competing traffic (or cross traffic) is generated by triggering persistent FTP flows 10 seconds prior to TCP streaming sessions. The number of cross traffic flows is varying to investigate the effect of the packet loss rate on the performance of TCP streaming. Unless otherwise specified, following sets of configurations are examined, each of which generates 10 traces using different random seeds. In all configurations, the duration of simulation time is set to 600 seconds.

- *Configuration 1*: the number of competing FTP flows is assumed to be 3 that leads video streaming flows to have 140.3ms RTT, 179.0ms T_0, 0.44% packet loss rate, and 1.24Mbps throughput.
- *Configuration 2*: the number of competing flows is set to 6. Measured characteristics of video streaming flows are 141.3ms RTT, 182.3ms T_0, 0.79% packet loss rate, and 902.1kbps throughput.
- *Configuration 3*: the number of competing flows is 9. Measurement results are 142.4ms RTT, 181.6ms T_0, 1.2% packet loss rate, and 715.0kbps throughput.

It should be noted that measured round-trip delays are different in each configuration, because of the queuing delays in the intermediate routers as well as link delays.

To estimate the packet loss rate in each configuration, we employ the TCP throughput equation.[7] The equation provides an analytic relationship between the packet loss rate, RTT, T_0, and TCP throughput. However, as the relationship is too complicated to yield a closed form of a packet loss rate as a function of throughput and RTT, we develop an iterative algorithm in Figure 3 based on the bisection method.[8] Since the TCP throughput equation is continuous and an estimated throughput must lie in the packet loss rate of [0, 1], the existence of a root is guaranteed by the intermediate value theorem. Also the estimated packet loss rate is unique, since the estimated throughput is monotonically decreasing as the packet loss rate increases.

procedure ComputeLossRate (R, B)
    p_l = 0, p_h = 1
    while $|p_h - p_l| > \epsilon$ **do**
        $p = (p_l + p_h)/2$
        $\widehat{B} = \frac{1}{R\sqrt{\frac{2bp}{3}} + T_0 \min(1, 3\sqrt{\frac{3bp}{8}})p(1+32p^2)}$
        if $\widehat{B} < B$
            $p_h = p$
        else
            $p_l = p$
    enddo
endprocedure

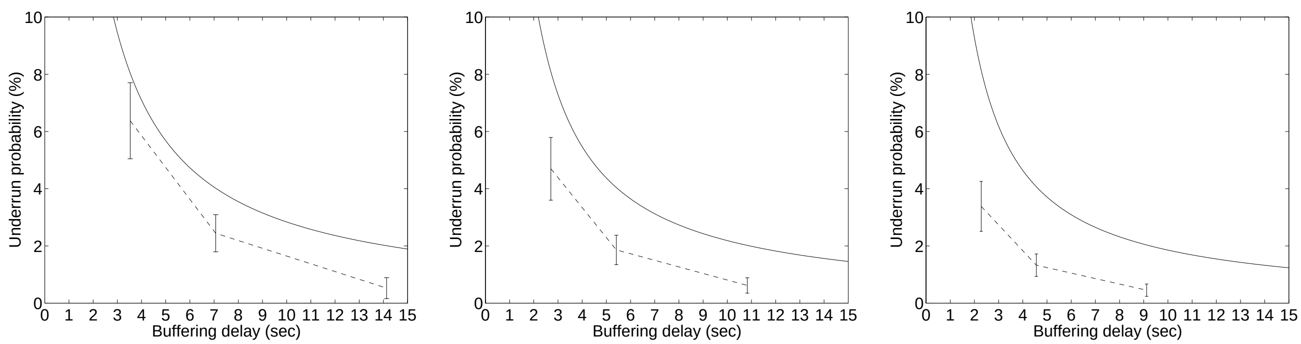
Figure 3. Packet loss rate estimation algorithm

The performance of TCP streaming experiments is evaluated by the buffer underrun probability and the disruption frequency as defined in Section 2.1. Note that the number of epochs is given by the simulation time (600 seconds) divided by an epoch, and the disruption frequency is the buffer underrun probability divided by an epoch.

3.2. Buffer Underrun Probability

In the first experiment, we measure the buffer underrun probability in each TCP streaming flow. We investigate 50 TCP streaming flows, since each configuration contains 10 traces, each of which contains 5 TCP streaming flows.

Figure 4 (a) shows the buffer underrun characteristics of configuration 1. The solid line specifies the minimum buffering delay requirements in Theorem 2.1 to achieve desired buffer underrun probability. Each error bar corresponds to the measured buffer underrun probability of 50 TCP streaming flows with 99% confidence interval. When RTT, T_0, and the packet loss rate are 140.3ms, 179.0ms and 0.44%, buffering delays targeting desired buffer underrun probabilities of 8%, 4%, and 2% are 3.53, 7.06, and 14.13 seconds respectively. Experimental results show that measured buffer underrun probabilities are tightly bounded by the solid line of 99% confidence level. Observe that the range of the confidence interval is reduced as buffering delay is increased, since the variance of measured buffer underrun probabilities is decreased.



(a) Configuration 1 (b) Configuration 2 (c) Configuration 3

Figure 4. Buffer underrun probability characteristics

Note that the buffering delay characteristic is a non-linear curve. For example, when buffering delay is increased from 3 seconds to 9 seconds in Figure 4 (a), desired buffer underrun probability is reduced by 6.31%. However, when buffering delay is increased from 9 seconds to 15 seconds, the probability is reduced by only 1.26%. Therefore, a system designer can find a point of marginal return using the non-linear characteristics.

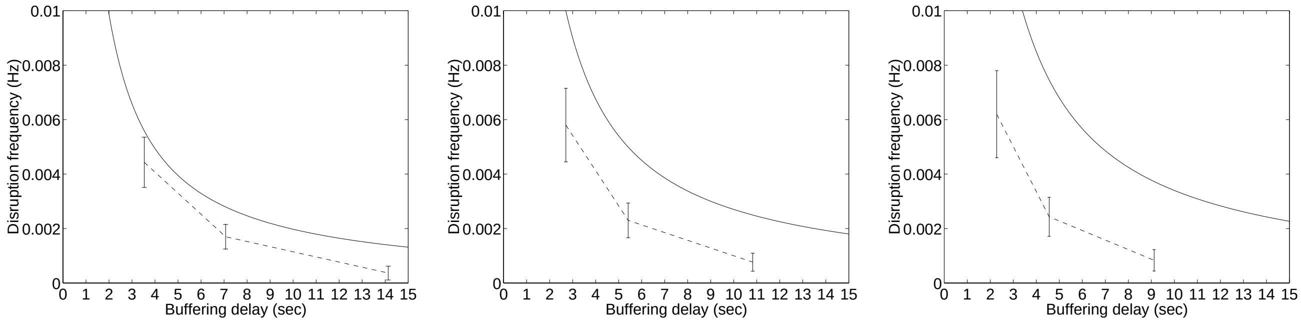
Figure 4 (b) shows the buffer underrun probability of configuration 2. Required buffering delays targeting $P_u = 8\%$, 4%, and 2% are 2.70, 5.41, and 10.82 seconds respectively. Compared with Figure 4 (a), the 99% confidence intervals are loosely bounded by the buffering delay requirement curve. This is because, as the packet loss rate is increased, the $o(1/p)$ term in (21) gets more significant, and the measured buffer underrun probability deviates more from the desired buffer underrun probability curve.

Figure 4 (c) shows the buffer underrun probability of configuration 3. Required buffering delays for $P_u = 8\%$, 4%, and 2% are 2.28, 4.56, and 9.12 seconds respectively. Experimental results demonstrate that desired buffer underrun probability becomes more conservative, as the packet loss rate is increased.

3.3. Disruption Frequency

In this experiment, we investigate disruption frequency characteristics which was defined in Section 2.1. Measured disruption frequency is defined by the number of buffer underrun events during the 600 second simulation time.

Figure 5 shows disruption frequency characteristics for configuration 1, 2, and 3. The error bar specifies the 99% confidence interval, and the solid line shows desired disruption frequency which is defined by the ratio of desired buffer underrun probability to the duration of an epoch. Hence, Figure 5 exhibits the same characteristics as Figure 4: 99% confidence interval of measured disruption frequencies is tightly bounded by desired disruption frequency when the packet loss rate is small. However, as the packet loss rate is increased, desired disruption frequency becomes a conservative bound.



(a) Configuration 1 (b) Configuration 2 (c) Configuration 3

Figure 5. Disruption frequency characteristics

4. CONCLUSION

In this paper, we consider video streaming over TCP. While the use of TCP provides the reliable video stream delivery, the bursty nature of TCP requires buffering at a receiver for smooth video playout. Since it is desirable to determine the right size of receiver buffer in memory-constrained applications, we quantify the buffering requirement to achieve desired buffer underrun probability by analytically modeling the CBR video streaming. Our analysis shows that the receiver buffer requirement is determined by the network characteristics and desired buffer underrun probability (or disruption frequency). Experimental results validate our model and analysis.

APPENDIX: Proof of Theorem 2.1

In a TDP, since the packet arrival rate is greater than zero, and it is increased linearly until it encounters triple duplicate ACKs, the receiver buffer size at round k in (1) is given by

$$\begin{aligned} q_k &= q_0 + \sum_{j=1}^{k}(\lambda_j - \widehat{\lambda}) \\ &= q_0 + \frac{k^2}{2b} + \frac{(W_{i-1} - 2\widehat{\lambda} - 1)k}{2}. \end{aligned} \tag{4}$$

On the other hand, the receiver buffer size in a TOP in (2) is given by[‡]

$$q_k = q_0 - k\widehat{\lambda}. \tag{5}$$

To prove Theorem 2.1, the following relationships[1] are used[§]:

$$W_i = \frac{W_{i-1}}{2} + \frac{X_i}{b} - 1 \tag{6}$$

$$Y_i = \alpha_i + W_i - 1 \tag{7}$$

$$Y_i = \frac{X_i}{2}(\frac{W_{i-1}}{2} + W_i) + \beta_i. \tag{8}$$

Note that the expressions in (6) and (8) are different from the original equations[7] by a constant term[¶]. However, for small values of p, TCP throughput in a TDP can still be expressed by

$$B_{TDP}(p, R) = \frac{1}{R}\sqrt{\frac{3}{2bp}} + o(\frac{1}{\sqrt{p}}). \tag{9}$$

[‡]We assume that a round in Z^{TO} is equal to RTT, although no ACK packet is received during Z^{TO}.

[§]Although three duplicated packets are lost and not delivered to a receiver in a TDP, the data reception rate can be approximated by the data transmission rate for small p.

[¶]The relationships can be verified from the ith TDP in Figure 2 (a): the parameters given by $W_{i-1} = W_i = 6$, $X_i = 8$, $Y_i = 39$, $\alpha_i = 34$, $\beta_i = 3$, and $b = 2$ satisfy (6), (7), and (8).

To achieve a desired buffer underrun probability, we need to consider the minimum buffer size in (4) and (5). Using the Markovian inequality, the buffer underrun probability at time t in a TDP is given by

$$\begin{aligned} P\{q_{min} \le 0 | t \in Z^{TD}\} &= P\{q_0 \le \frac{b}{8}(W_{i-1} - 2\widehat{\lambda} - 1)^2\} \\ &\le \frac{E\{\frac{b}{8}(W_{i-1} - 2\widehat{\lambda} - 1)^2\}}{q_0} \\ &= \frac{b}{8q_0}(E\{W^2\} - 4\widehat{\lambda}E\{W\} + 4\widehat{\lambda}^2 + 4\widehat{\lambda} - 2E\{W\} + 1), \end{aligned} \tag{10}$$

where $E\{W^2\}$ stands for the average of W_{i-1}^2 and $E\{W\}$ for the average of W_{i-1}.

Equation (10) can be solved using (6), (7), and (8). From (6), it follows that

$$E\{X\} = \frac{b}{2}E\{W\} + b. \tag{11}$$

Observe that squaring (6) leads to $W_i^2 = \frac{W_{i-1}^2}{4} + \frac{W_{i-1}X_i}{b} + \frac{X_i^2}{b^2} - \frac{2X_i}{b} - W_{i-1} + 1$. Hence, the average of X_i^2 is given by

$$E\{X^2\} = \frac{3b^2}{4}E\{W^2\} - \frac{b^2}{2}E^2\{W\} + b^2E\{W\} + b^2. \tag{12}$$

Note that squaring (6) after manipulating the $\frac{W_{i-1}}{2}$ term yields

$$W_i^2 - W_iW_{i-1} + \frac{W_{i-1}^2}{4} = \frac{X_i^2}{b^2} - \frac{2X_i}{b} + 1. \tag{13}$$

From (11), (12), and (13), the correlation of congestion window sizes between adjacent TDPs is given by

$$E\{W_iW_{i-1}\} = \frac{1}{2}(E\{W^2\} + E^2\{W\}). \tag{14}$$

We consider (7) and (8) to derive $E\{W^2\}$. Since α_i is the first packet lost in a TDP, α_i can be assumed to have a geometric distribution with the probability p. Hence, it follows that $E\{\alpha_i\} = \frac{1}{p}$ and $E\{\alpha_i^2\} = \frac{2-p}{p^2}$. With this assumption, squaring (7) leads to

$$E\{Y^2\} = \frac{2-p}{p^2} + E\{W^2\} + 1 + \frac{2}{p}E\{W\} - \frac{2}{p} - 2E\{W\}. \tag{15}$$

In the same way, $E\{Y^2\}$ can also be obtained by (8) and (14). Since β_i is the number of packets in the last round, it can be assumed to have a uniform distribution in $[1, W_i]$. Therefore, squaring (8) yields

$$\begin{aligned} E\{Y^2\} &= \frac{E\{X^2\}}{4}(\frac{5E\{W^2\}}{4} + E\{W_iW_{i-1}\}) + E\{\beta\}E\{X\}\frac{3E\{W\}}{2} + E\{\beta^2\} \\ &= \frac{b^2}{4}(\frac{3}{4}E\{W^2\} - \frac{1}{2}E^2\{W\} + E\{W\} + 1)(\frac{7}{4}E\{W^2\} + \frac{1}{2}E^2\{W\}) + \frac{3b}{4}E^2\{W\}(\frac{1}{2}E\{W\} + 1) \\ &+ \frac{2E\{W^2\} + 3E\{W\} + 1}{6}. \end{aligned} \tag{16}$$

Since the average window size is given[7] by $E\{W\} = \sqrt{\frac{8}{3bp}} + o(\frac{1}{\sqrt{p}})$, we assume $E\{W^2\} = O(\frac{1}{p})$. By equating the relationships in (15) and (16), we can derive a relationship, such that

$$\frac{21b^2}{64}E^2\{W^2\} - \frac{b}{3p}E\{W^2\} - \frac{22}{9p^2} = o(\frac{1}{p^2}).$$

Hence,

$$E\{W^2\} = \frac{32 + 8\sqrt{478}}{63bp} + o(\frac{1}{p}). \tag{17}$$

Note that the long-term average of λ_k is equal to $\widehat{\lambda}$. Therefore, TCP throughput in (9) can also be applied to $\widehat{\lambda}$, such that $\widehat{\lambda} = \sqrt{\frac{3}{2bp}} + o(\frac{1}{\sqrt{p}})$. From (17), the buffer underrun probability in a TDP in (10) is bounded by

$$\begin{aligned} P\{q_{min} \leq 0 | t \in Z^{TD}\} &\leq \frac{b}{8q_0}[\frac{32 + 8\sqrt{478}}{63bp} - 4\sqrt{\frac{3}{2bp}}\sqrt{\frac{8}{3bp}} + 4(\sqrt{\frac{3}{2bp}})^2] + o(\frac{1}{p}) \\ &= \frac{0.16}{q_0 p} + o(\frac{1}{p}). \end{aligned} \tag{18}$$

To derive an expression of the buffer underrun probability in a TOP, we consider (5) and apply the Markovian inequality. Since the minimum buffer size in a TOP is given at $k = \lfloor Z^{TO}/R \rfloor$, we have

$$\begin{aligned} P\{q_{min} \leq 0 | t \in Z^{TO}\} &= P\{q_0 \leq \lfloor \frac{Z^{TO}}{R} \rfloor \widehat{\lambda}\} \\ &\leq \frac{E\{Z^{TO}\}}{q_0 R}\widehat{\lambda}, \end{aligned} \tag{19}$$

Since the average duration of a TOP is described by $E\{Z^{TO}\} = T_0 \frac{f(p)}{1-p}$, where $f(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6$, (19) leads to

$$P\{q_{min} \leq 0 | t \in Z^{TO}\} \leq \frac{T_0}{q_0 R}\sqrt{\frac{3}{2bp}} + o(\frac{1}{\sqrt{p}}). \tag{20}$$

Now we derive the unconditional probability of buffer underrun from the conditional probabilities. Since W_i is a regenerative process over the period of $Z^{TD} + Z^{TO}$, we have

$$\begin{aligned} P\{t \in Z^{TD}\} &= \frac{E\{Z^{TD}\}}{E\{Z^{TD}\} + E\{Z^{TO}\}}, \\ P\{t \in Z^{TO}\} &= \frac{E\{Z^{TO}\}}{E\{Z^{TD}\} + E\{Z^{TO}\}}, \end{aligned}$$

where $E\{Z^{TD}\} = E\{n\}(E\{X\} + 1)R$, $E\{X\} = \sqrt{\frac{2b}{3p}} + o(\frac{1}{\sqrt{p}})$, and $E\{n\}$ is the average number of TDPs in Z^{TD}. Consider the probability of TO loss indication Q, which is given[7] by $Q = \frac{1}{E\{n\}} \approx \min(1, 3\sqrt{\frac{3bp}{8}})$. From (18) and (20), the buffer underrun probability is thus

$$\begin{aligned} P\{q_{min} \leq 0\} &= \frac{P\{q_{min} \leq 0 | t \in Z^{TD}\}(E\{X\} + 1) + QP\{q_{min} \leq 0 | t \in Z^{TO}\}T_0 \frac{f(p)}{1-p}}{(E\{X\} + 1)R + QT_0 \frac{f(p)}{1-p}} \\ &\leq \frac{\frac{0.16}{q_0 p}R\sqrt{\frac{2b}{3p}} + \frac{T_0}{q_0 R}\sqrt{\frac{3}{2bp}}\min(1, 3\sqrt{\frac{3bp}{8}})T_0 \frac{f(p)}{1-p}}{R\sqrt{\frac{2b}{3p}} + \min(1, 3\sqrt{\frac{3bp}{8}})T_0 \frac{f(p)}{1-p}} + o(\frac{1}{p}) \\ &\leq \frac{0.16}{q_0 p}[1 + \frac{9.4}{b}(\frac{T_0}{R})^2 \min(1, 3\sqrt{\frac{3bp}{8}})p(1 + 32p^2)] + o(\frac{1}{p}). \end{aligned} \tag{21}$$

Therefore, given desired buffer underrun probability, such that $P\{q_{min} \leq 0\} \leq P_u$, required buffer size is given by

$$q_0 \geq \frac{0.16}{pP_u}[1 + \frac{9.4}{b}(\frac{T_0}{R})^2 \min(1, 3\sqrt{\frac{3bp}{8}})p(1 + 32p^2)].$$

REFERENCES

1. Z. Chen, T. Bu, M. Ammar, and D. Towsley, “Comments on modeling TCP Reno performance: a simple model and its empirical validation,” *to appear in IEEE/ACM Trans. Networking*.
2. Companion web site, `http://www.cc.gatech.edu/computing/Telecomm/people/Phd/tkim/tcp_streaming.html`, 2005.
3. P. de Cuetos and K. W. Ross, “Adaptive rate control for streaming stored Fine-Grained Scalable video,” *NOSSDAV 2002*, Miami, FL, May 2002.
4. Freescale application processors, `http://www.freescale.com/webapp/sps/site/overview.jsp?nodeId=01J4Fs29733642`.
5. C. Krasic, K. Li, and J. Walpole, “The case for streaming multimedia with TCP,” *iDMS 2001*, Lancaster, UK, Sept. 2001.
6. `ns-2` network simulator, `http://www.isi.edu/nsnam/ns/`, 2001.
7. J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, “Modeling TCP reno performance: a simple model and its empirical validation,” *IEEE/ACM Trans. Networking*, vol. 8, no. 2, pp. 133-145, Apr. 2000.
8. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical recipes in C*, Cambridge university press, 1988.
9. D. Saparilla and K. W. Ross, “Streaming stored continuous media over fair-share bandwidth,” *NOSSDAV 2000*, Chapel Hill, NC, June 2000.
10. W. Tan, W. Cui, and J. G. Apostolopoulos, “Playback buffer equalization for streaming media using stateless transport prioritization,” *Packet Video 2003*, Nantes, Fr, Apr. 2003.
11. Y. Wang, M. Claypool, and Z. Zuo, “An empirical study of realvideo performance across the Internet,” *ACM SIGCOMM IMW 2001*, San Francisco, CA, Nov. 2001.