Multimedia Streaming via TCP: An Analytic Performance Study

BING WANG
University of Connecticut
and
JIM KUROSE, PRASHANT SHENOY, and DON TOWSLEY
University of Massachusetts, Amherst

TCP is widely used in commercial multimedia streaming systems, with recent measurement studies indicating that a significant fraction of Internet streaming media is currently delivered over HTTP/TCP. These observations motivate us to develop analytic performance models to systematically investigate the performance of TCP for both live and stored-media streaming. We validate our models via *ns* simulations and experiments conducted over the Internet. Our models provide guidelines indicating the circumstances under which TCP streaming leads to satisfactory performance, showing, for example, that TCP generally provides good streaming performance when the achievable TCP throughput is roughly twice the media bitrate, with only a few seconds of startup delay.

Categories and Subject Descriptors: C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Application (multimedia streaming)*; H.1.0 [**Models and Principles**]: General

General Terms: Performance

Additional Key Words and Phrases: Performance modeling, multimedia streaming

ACM Reference Format:

Wang, B., Kurose, J., Shenoy, P., and Towsley, D. 2008. Multimedia streaming via TCP: An analytic performance study. ACM Trans. Multimedia Comput. Commun. Appl. 4, 2, Article 16 (May 2008), 22 pages. DOI = 10.1145/1352012.1352020 http://doi.acm.org/10.1145/1352012.1352020

1. INTRODUCTION

The rapid deployment of broadband connectivity to the home via cable modem and ADSL technologies has resulted in a significant growth in multimedia streaming. TCP, the most widely used transport protocol in the Internet, is conventionally regarded as inappropriate for media streaming. The primary reason is that the backoff and retransmission mechanisms in TCP can lead to undesirable end-to-end delays that violate the timeliness requirement for streaming media. Due to these limitations, much

The first version of this article appeared in *Proceedings of the ACM Multimedia* [2004].
This work is supported in part by the National Science Foundation under NSF grants ANI-9977635, ANI-9980552, CCR-9984030 and EEC-0313747 001.
Authors' addresses: B. Wang, Computer Science & Engineering Department, University of Connecticut, Storrs, CT 06269; email: bing@engr.uconn.edu; J. Kurose, P. Shenoy, D. Towsley, Department of Computer Science, University of Massachusetts, Amherst, MA, 01003; emails: {kurose, shenoy, towsley}@cs.umass.edu.

of the research over the past decade focused on developing UDP-based streaming protocols, providing mechanisms for TCP-friendliness and loss recovery [Rejaie et al. 1999; Floyd et al. 2000; Boutremans and Le Boudec 2003].

Despite the conventional wisdom that TCP is not desirable for streaming and the large body of literature on UDP-based streaming, TCP is widely used in commercial streaming systems. For instance, Real Media and Windows Media, the two dominant streaming media products, both support TCP streaming. Furthermore, recent measurement studies have shown that a significant fraction of commercial streaming traffic uses TCP [Wang et al. 2001; Sripanidkulchai et al. 2004; van der Merwe et al. 2002]. Wang et al. [2001] reported that 44% of video streaming is delivered using TCP. Sripanidkulchai et al. [2004] studied the live streaming workload from a large content distribution network and showed that roughly 40% to 80% of the live streaming uses HTTP/TCP. Van der Merwe et al. [2002] analyzed 4.5 million session-level logs for two commercial streaming servers over a four month period and found that 72% and 75% of the on-demand and live streaming traffic, respectively, used TCP. Moreover, 27% and 47% of the on-demand and live streaming traffic, respectively, used HTTP. The wide use of HTTP streaming is particularly interesting—HTTP streaming is perhaps the simplest streaming protocol, since no rate adaptation is employed at the application level (HTTP provides no interface for rate adaptation), unlike other TCP streaming approaches [Seelam et al. 2001; Krasic and Walpole 2001; de Cuetos and Ross 2002; de Cuetos et al. 2002]; further, no additional mechanisms are necessary to ensure TCP friendliness or to recover from loss, unlike UDP-based streaming.

In this paper, motivated by the wide use of TCP streaming in commercial systems, we seek to answer the following question: *Under what circumstances can TCP streaming provide satisfactory performance?* To answer this question, we study a baseline streaming scheme which uses TCP directly for streaming. This baseline streaming scheme is similar to HTTP streaming and is henceforth referred to as *direct TCP streaming*. We study the performance of direct TCP streaming using analytical models, which enable us to systematically investigate the performance of TCP streaming under various conditions, a task that is difficult when using empirical measurements or simulation alone. Our paper makes the following main contributions:

—We develop discrete-time Markov models for *live* and *stored-media* streaming using TCP. The models are validated using *ns* simulation and Internet experiments. To the best of our knowledge, our work is the first analytical study of the use of TCP for streaming.

—Using these models, we explore the parameter space (i.e., loss rate, round trip time and timeout value in TCP as well as media playback rate) to provide guidelines as to when direct TCP streaming leads to satisfactory performance. Our results show that direct TCP streaming generally provides good performance when the available network bandwidth, and thus the achievable TCP throughput, is roughly twice the media bitrate, with only a few seconds of startup delay.

Our study has the following important implication. Measurement studies have shown that a large fraction of streaming video clips on the Internet today are encoded at bit rates below 300 Kbps [Li et al. 2005]. Moreover, most broadband connections support download rates of 750 Kbps–1 Mbps. In the situations where the end-end available bandwidth is only constrained by the last-mile access link, our performance study indicates that direct TCP streaming may be adequate for many broadband users.

The rest of the article is organized as follows. In Section 2, we review related work on TCP-based streaming and TCP modeling. Section 3 presents the models for live and stored video streaming using TCP. Validation of the models using *ns* simulations and Internet experiments is described in Sections 4 and 5 respectively. A performance study based on the models is presented in Section 6. Finally, Section 7 concludes the paper.

2. RELATED WORK

TCP-based streaming has several advantages. First, TCP is by definition TCP friendly. Second, reliable transmissions provided by TCP remove the need for loss recovery at higher levels. Furthermore, in practice, streaming content using TCP is more likely to pass through firewalls. A number of existing research efforts that use TCP for streaming [Seelam et al. 2001; Krasic and Walpole 2001; de Cuetos and Ross 2002; de Cuetos et al. 2002] combine *client-side buffering* and *rate adaptation* to deal with the variability in the available TCP throughput. Client-side buffering prefetches data into the client buffer by introducing a startup delay in order to absorb *short-term* fluctuations in the TCP throughput. Rate adaptation adjusts the bitrate (or quality) of the video in order to deal with *long-term* fluctuations. Direct TCP streaming does not deal with long-term fluctuations and only employs client-side buffering. It is consequently much simpler than TCP streaming techniques that employ rate adaptation [Seelam et al. 2001; Krasic and Walpole 2001; de Cuetos and Ross 2002; de Cuetos et al. 2002]. Furthermore, it does not require layered video as in de Cuetos and Ross [2002] and de Cuetos et al. [2002]. In this paper, we focus on the performance of direct TCP streaming. We expect the performance of more sophisticated approaches like Seelam et al. [2001], Krasic and Walpole [2001], de Cuetos and Ross [2002], and de Cuetos et al. [2002] to be better. However, the performance of these approaches and the comparison of different approaches are beyond the scope of this study.

The literature on TCP modeling is extensive, although no existing work has specifically focused on TCP streaming. Much of the TCP modeling focuses on TCP performance for file transfers, assuming long-lived flows [Mathis et al. 1997; Padhye et al. 1998, 1999; Altman et al. 2000; Figueiredo et al. 2002] or short-lived flows [Cardwell et al. 2000; Mellia et al. 2002]. In particular, Padhye et al. [1999] and Figueiredo et al. [2002] use Markov models to capture the congestion control and avoidance mechanisms in TCP to study the steady-state TCP throughput and the autocorrelation structure in TCP traffic respectively. Our work differs from all past work in that we consider real-time requirements when using TCP for streaming. We use the TCP models in Padhye et al. [1999] and Figueiredo et al. [2002] as a baseline to develop Markov models for streaming. The motivation for using Markov models is two fold. First, they capture the detailed congestion control and avoidance mechanisms in TCP. The timeout mechanism, which leads to a drastic decrease in congestion window size, is particularly important for modeling streaming using TCP (see Section 6). Secondly, it is convenient to perform transient analysis using Markov model, which is required for stored video streaming (see Section 3).

An earlier study [Bohacek 2003] provides a model to obtain the probability distribution of TCP congestion window size, and then applies the model to determine a TCP-friendly transmission rate for UDP video flows. Our work, instead, focuses on the performance of TCP-based streaming. A recent work [Kim and Ammar 2006] uses TCP for stored-media streaming and analytically determined the proper receiver buffer size to ensure a prescribed video quality. Our work differs from the above work in that we assume the receiver buffer size is sufficiently large (see Section 3) and focuses on providing guidelines as to when TCP streaming leads to satisfactory performance.

3. MODELS FOR STREAMING USING TCP

In this section, we describe the problem setting and then present discrete-time Markov models for live and stored-media streaming using TCP. For simplicity, in the rest of the paper, we limit our description to video streaming. The same approach can be applied to the streaming of other contents. The key notation introduced in this section is summarized in Table I for easy reference.

Table I. Key Notation

Notation	Definition
μ	Playback rate of the video (packets per second)
τ	Startup delay (seconds)
X_i	State of the TCP source in the i-th round
S_i	Number of packets transmitted successfully by TCP in the i-th round
R	Round trip time (seconds)
L	Length of the video (measured in rounds)
f	Fraction of late packets
N_i	Number of early packets in the i-th round
N_i^l	Number of late packets in the i-th round
Y_i^l	State of the model for live streaming in the i-th round
Y_i^s	State of the model for stored-media streaming in the i-th round
P_i	Probability of having at least one late packet in the i-th round

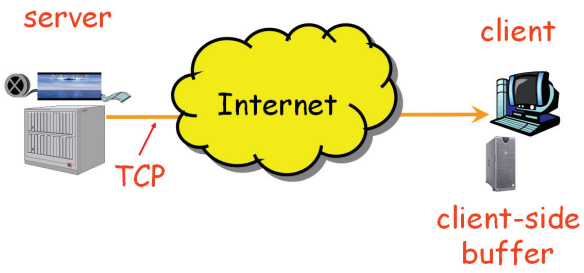


Fig. 1. Problem setting: the server streams the video to the client using TCP. At the client, all packets arriving earlier than their playback times are stored in the client-side buffer.

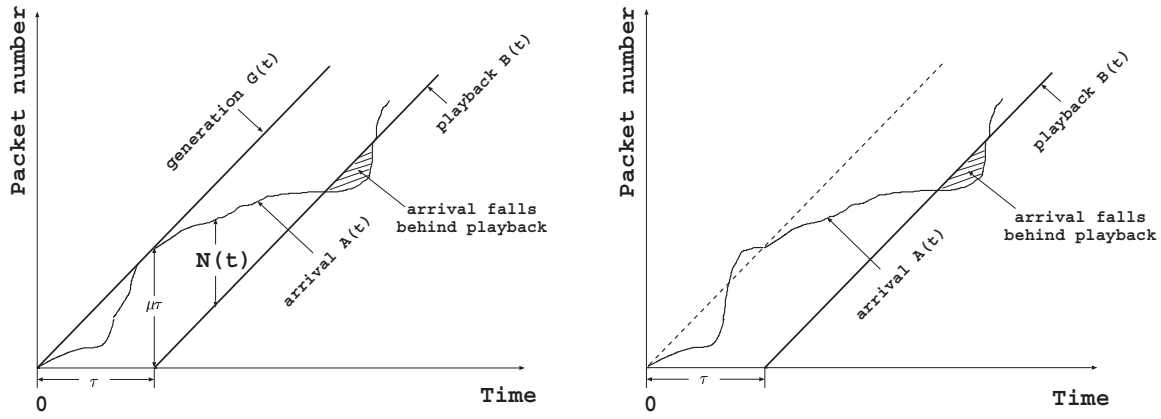
3.1 Problem Setting

The problem setting is illustrated in Figure 1. Consider a client requesting a video from the server. Corresponding to the request, the server streams the video to the client using TCP. The client allows a startup delay on the order of seconds, which is a common practice in commercial streaming products. All packets arriving earlier than their playback times are stored in the client's local buffer. This local buffer is assumed to be sufficiently large so that no packet loss is caused by buffer overflow at the client side. This assumption is reasonable since modern machines are equipped with a large amount of storage.

Measurement studies show that most videos streamed over the Internet are constant bit rate (CBR) [Li et al. 2005]. We therefore consider a CBR video with a playback rate of μ packets per second.[1] For simplicity, all packets are assumed to be of the same size. For analytical tractability, we assume continuous playback at the client. A packet arriving later than its playback time is referred to as a *late packet*. We assume that a late packet leads to a glitch during playback and use the *fraction of late packets*, that is, the probability that a packet is late, to measure performance. Strictly speaking, the fraction of late packets does not correspond directly to viewing quality, since human perception tolerates occasional glitches in the playback. However, there is no quantitative metric that directly corresponds to the viewing quality of a video perceived by a human. We therefore use fraction of late packets as a rough metric of streaming performance.

We study two forms of streaming—live and stored-media streaming. In live streaming, the server generates video content in real time and is only able to transmit the content that has already been

[1]Variable bit rate (VBR) coding is significantly more efficient than CBR coding. Therefore, it is important to model VBR streaming when more VBR videos are streamed over the Internet. However, such a model is beyond the scope of this paper.



(a) Live streaming. (b) Stored-media streaming.

Fig. 2. Illustration of live and stored-media streaming using TCP.

generated. For a stored video, the server is assumed to transmit the video as fast as allowed by the achievable TCP throughput in order to fully utilize the available network bandwidth. Next, we discuss the characteristics of live and stored-media streaming. For ease of exposition, each packet is associated with a sequence number. The first packet is given a sequence number of 1 and subsequent packets have sequence numbers incremented by 1.

Live streaming is illustrated in Figure 2(a). Without loss of generality, we assume that the first packet is generated at time 0 and sent immediately to the client. Later packets are generated at a constant rate equal to the playback rate of the video. In Figure 2(a), $G(t)$ represents the number of packets generated at the server by time t. Then $G(t) = \mu t$. At the client side, let $A(t)$ denote the number of packets arriving at the client by time t. Since the TCP transmission is constrained by the generation rate at the server, we have $A(t) \leq G(t)$. Let $B(t)$ denote the number of packets played by the client by time t. The playback of the video commences at time τ. That is, the startup delay is τ seconds. Then $B(t) = \mu(t - \tau)$, $t \geq \tau$. Observe that $G(t) - B(t) = \mu\tau$. A packet arriving earlier than its playback time is referred to as an *early packet*. At time t, let the number of early packets be $N(t)$. Then $N(t) = A(t) - B(t)$. A negative value of $N(t)$ indicates that the packet arrival is behind the playback by $-N(t)$ packets. Since $A(t) \leq G(t)$ and $G(t) - B(t) = \mu\tau$, we have $N(t) \leq G(t) - B(t) = \mu\tau$. That is, there are at most $\mu\tau$ early packets in live streaming at any time t, as shown in Figure 2(a).

Store-media streaming is illustrated in Figure 2(b). As shown in the figure, the packet transmission is only limited by the achievable TCP throughput. Therefore, stored-media streaming differs from live streaming in that the number of early packets at time t, $N(t)$, can be larger than $\mu\tau$. This difference directly leads to different analytical models for live and stored-media streaming, as detailed in Section 3.3.

As described above, a negative value of $N(t)$ indicates that late packets occur at time t. We need to model $N(t)$ during the playback of the video in order to obtain the fraction of late packets. For this purpose, we extend the model of TCP in Padhye et al. [1999] and Figueiredo et al. [2002] to incorporate the specific characteristics of live and stored-media streaming. In Section 3.3.1, we construct a Markov model for live streaming where the number of early packets is one component in the model. In Section 3.3.2, we provide a transient analysis technique for stored-media streaming. Before describing the models for live and stored-media streaming, we first briefly describe the model for TCP.

3.2 Model of TCP

TCP is a window-based protocol using several mechanisms to regulate its sending rate in response to network congestion. Timeout and congestion avoidance are two mechanisms that have significant impact on the throughput. For completeness, we give a brief description of these two mechanisms; a more detailed description can be found in Stevens [1994]. For every packet sent by the source, TCP starts a retransmission timer and waits for an acknowledgment from the receiver. The retransmission timer expires (timesout) when the source does not receive an ACK for the corresponding packet and there are no triple duplicate ACKs. When a timeout occurs, the packet is retransmitted and the window size is reduced to one. Furthermore, the retransmission timer value for this retransmitted packet is set to twice the previous timer value. This exponential backoff behavior continues until the retransmitted packet is successfully acknowledged. In congestion avoidance, the window size increases by one packet when all packets in the current window are acknowledged. Most versions of TCP, such as TCP Reno and TCP Sack, reduce the window size by half when triple duplicate ACKs are received. If a timeout occurs before receiving triple duplicate ACKs, the window size is reduced to one.

In Padhye et al. [1999] and Figueiredo et al. [2002], the behavior of TCP is described by a discrete-time Markov model, where each time unit is the length of a "round." A round starts with the back-to-back transmission of W packets, where W is the current size of the TCP congestion window. Once all packets in the congestion window are sent, no more packets are sent until ACKs for some or all of these W packets are received. The reception of the ACKs marks the end of the current round and the beginning of the next round. The length of a round is assumed to be a round trip time (RTT). Packet losses in different rounds are assumed to be independent and packet losses in the same round are correlated: if a packet is lost, all remaining packets until the end of the round are lost. Furthermore, the effect of lost ACKs is regarded as negligible.

Let $\{X_i\}_{i=1}^{\infty}$ be a discrete-time Markov model for the TCP source, where X_i is the state of the model in the i-th round. Following the notation in Figueiredo et al. [2002], X_i is a tuple: $X_i = (W_i, C_i, L_i, E_i, R_i)$, where W_i is the window size in the i-th round; C_i models the delayed ACK behavior of TCP ($C_i = 0$ and $C_i = 1$ indicate the first and the second of the two rounds respectively); L_i is the number of packets lost in the $(i-1)$th round; E_i denotes whether the connection is in a timeout state and the value of the back-off exponent in the i-th round; R_i indicates if a packet being sent in the timeout phase is a retransmission ($R_i = 1$) or a new packet ($R_i = 0$). Let S_i denote the number of packets transmitted successfully by TCP in the i-th round. Then S_i is determined by X_i and X_{i+1}. For instance, when there is no packet loss from state $X_i = (w, c, l, e, r)$ to $X_{i+1} = (w', c', l', e', r')$, we have $S_i = w$, the window size in the i-th round. A detailed description of S_i can be found in Padhye et al. [1999] and Figueiredo et al. [2002].

3.3 Models for Live and Stored-Media Streaming

We now present discrete-time Markov models for live and stored-media streaming. Each time unit corresponds to the length of a round, which is assumed to be a RTT of length R seconds. We consider a media whose length is L rounds. The playback rate of the video is μR packets per round, which is assumed to be an integer for simplicity of notation.

Let f denote the fraction of late packets during the playback of the video. Our goal is to derive models for determining f as a function of various system parameters (including the loss rate, RTT, the retransmission timer in the TCP flow and the video playback rate). Let N_i denote the number of early packets in the i-th round, which is a discrete-time version of $N(t)$ introduced earlier (see Section 3.1) with $N_i = N(iR)$. A negative value of N_i indicates that the packet arrival is behind the playback by $-N_i$. Let N_i^l be the number of late packets in the i-th round. Then $N_i^l \in \{0, 1, \ldots, \mu R\}$, where $N_i^l = 0$

indicates that no packet is late in the i-th round. Let the expected number of late packets in the i-th round be $E[N_i^l]$. Then the fraction of late packets is

$$f = \frac{\sum_{i=1}^{L} E\left[N_i^l\right]}{\mu R L}, \tag{1}$$

where the numerator and denominator correspond, respectively, to the expected number of late packets throughout the playback of the media and the total number of packets in the video. We obtain $E[N_i^l]$ as

$$E\left[N_i^l\right] = \sum_{k=1}^{\mu R} k P\left(N_i^l = k\right), \tag{2}$$

where $P(N_i^l = k)$ is the probability of having k late packets in the i-th round. In order to obtain $P(N_i^l = k)$, we introduce N_i^b, defined as the number of packets by which the arrival falls behind the playback of the video in the i-th round. Then by the definition of N_i and N_i^b, we have

$$N_i^b = \begin{cases} 0, & N_i \geq 0 \\ -N_i, & N_i < 0. \end{cases} \tag{3}$$

We then obtain $P(N_i^l = k)$ as

$$P\left(N_i^l = k\right) = \begin{cases} P\left(N_i^b = k\right), & 0 \leq k < \mu R \\ P\left(N_i^b \geq \mu R\right), & k = \mu R. \end{cases} \tag{4}$$

Note that while the number of late packets N_i^l in the i-th round is at most μR, N_i^b can be larger than μR. When $N_i^b \geq \mu R$, we have $N_i^l = \mu R$. Therefore, $P(N_i^l = \mu R) = P(N_i^b \geq \mu R)$.

To summarize the preceding, the fraction of late packets can be obtained from N_i, $i = 1, 2, \ldots, L$, by applying (1), (2), (3) and (4). Next we describe the models for live and stored-media streaming, with a focus on deriving the distribution of N_i from the models.

3.3.1 *Live Streaming.* We develop a model of live streaming by considering the data production and consumption at the client-side buffer. More specifically, we model live streaming as a producer-consumer problem: the TCP connection from the server to the client is the producer, the client is the consumer, and the client-side buffer is the buffer shared by the producer and consumer. The producer produces packets according to TCP and stores them in the client-side buffer. The consumer starts to consume (i.e., play back) and remove the packets in the buffer from time τ at a constant rate of μR packets per round. For modeling convenience, we assume that packets that are past their playback deadlines are removed from the buffer. That is, only early packets are stored in the buffer. Then, at any time, the number of packets in the buffer never exceeds N_{max}, $N_{max} = \mu\tau$, which follows from an earlier observation made in Section 3.1. To satisfy this constraint, the producer stops producing packets when there are N_{max} packets in the buffer. We therefore use the following model for live streaming.

Let $\{Y_i^l\}_{i=1}^{L}$ be a discrete-time Markov model for live streaming, where Y_i^l is the state of the model in the i-th round. Y_i^l is a tuple represented as (X_i, N_i), where X_i and N_i are the state of the TCP source and the number of early packets in the i-th round respectively. The evolution of N_i follows

$$N_{i+1} = \min(N_{max}, N_i + S_i - \mu R).$$

Recall that S_i is the number of packets transmitted successfully by TCP in the i-th round, which is determined by X_i and X_{i+1} (see Section 3.2). In order to satisfy the condition that $N_i \leq N_{max}$ for

$i = 1, 2, \ldots, L$, the TCP source does not send out any packet in the $(i+1)$th round if $N_i = N_{max}$. A detailed description of the state transition probabilities for the Markov chain $\{Y_i^l\}_{i=1}^L$ and the time taken for each state transition can be found in Appendix A.

We consider videos of lengths significantly larger than the RTT. In this case, the fraction of late packets can be approximated by the limiting case, where the length of the video, L, approaches infinity. That is, the fraction of late packets can be approximated by the steady state probability

$$\lim_{L\to\infty} \frac{\sum_{i=1}^{L} E[N_i^l]}{\mu R L} = \lim_{i\to\infty} \frac{E[N_i^l]}{\mu R}.$$

We solve for the stationary distribution of N_i using the steady state analysis in the TANGRAM-II modeling tool [de Souza e Silva and Leao 2000]. We then compute the stationary distribution of N_i^l using (3) and (4). Finally, the fraction of late packets is computed from (1).

3.3.2 *Stored-Media Streaming.* Stored-media streaming can also be modeled as a producer-consumer problem, where the TCP connection is the producer, the client is the consumer, and the client-side buffer is shared by the producer and consumer. However, in stored-media streaming, the number of packets in the buffer can exceed N_{max}. Furthermore, when the average TCP throughput is higher than the video bitrate, the number of early packets increases with the length of the video, and hence the fraction of late packets in the steady state (when the video is regarded as infinitely long) is trivial (equal to 0). To obtain the fraction of late packets over a finite video, we therefore resort to a transient analysis.

We develop the following model for stored-media streaming. Let $\{Y_i^s\}_{i=1}^L$ be a discrete-time Markov model for stored-media streaming, where Y_i^s is the state of the model in the i-th round. Here $Y_i^s = X_i$. The number of early packets in the i-th round, N_i, is excluded from the state space to reduce the size of the state space, and thus the computational overhead. We introduce an *impulse reward* into the model to obtain the transient distribution of N_i. An impulse reward associated with a state transition is a generic means to define measure of interest (see [de Souza e Silva and Gail 1998] for references on reward models). We associate an impulse reward of $\rho_{yy'}$ to a transition from state $Y_i^s = y$ to state $Y_{i+1}^s = y'$, defined to be the difference between the number of packets received and played back during this transition. Denote the accumulation of this impulse reward up to the i-th round as N_i'. The TANGRAM-II modeling tool [de Souza e Silva and Leao 2000] provides a functionality to solve for the transient distribution of N_i' based on the algorithm in [de Souza e Silva and Gail 1998].

We then obtain the transient distribution of N_i from that of N_i' as follows. Observe that N_i' is the total number of early packets in the i-th round when the transmission and playback both start at time 0. Recall that N_i is the number of early packets in the i-th round when the playback starts at time τ instead of 0. We therefore have the following relationship between N_i and N_i'

$$N_i = N_i' + \mu\tau.$$

This relationship allows us to obtain the transient distribution of N_i from that of N_i'. The detailed description of the impulse reward in our model can be found in Appendix B. To compute the fraction of late packets, we first solve for the transient distribution of N_i using the TANGRAM-II modeling tool [de Souza e Silva and Leao 2000]. Next, the transient distribution of N_i^l is calculated using (3) and (4). Finally, the fraction of late packets is computed from (1).

We are also interested in obtaining the probability that no late packet occurs during the playback of the video, denoted as α. That is

$$\alpha = P(N_1 \geq 0, N_2 \geq 0, \ldots, N_L \geq 0)$$

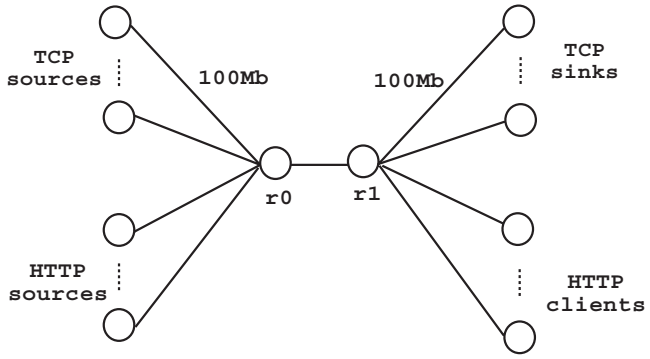


Fig. 3. Validation setting in *ns*: packet losses are caused by buffer overflow on the link from router r_0 to r_1.

This is a difficult quantity to compute exactly since it involves the joint probability distribution of N_1, $N_2, \ldots, N_L$. For convenience, let P_i denote the probability that the i-th round has at least one late packet. That is,

$$P_i = P(N_i < 0) = P(N_i' < -\mu\tau). \tag{5}$$

Therefore, P_i can be obtained directly from the transient distribution of N_i' using the TANGRAM-II modeling tool. We obtain the following lower bound on α; the proof is in Appendix C.

THEOREM 1. $\alpha \geq \Pi_{i=1}^{L} P(N_i \geq 0) = \Pi_{i=1}^{L}(1 - P_i)$.

4. MODEL VALIDATION USING *NS* SIMULATIONS

In this section, we validate our models for live and stored-media streaming using *ns* simulations. The topology is shown in Figure 3. Multiple TCP and HTTP sources are connected to router r_0 and their corresponding sinks are connected to router r_1. Each HTTP source contains 16 connections. The HTTP traffic is generated using empirical data provided by *ns*. The bandwidth and queue length of a link from a source/sink to its corresponding router are 100 Mbps and 1000 packets, respectively. The propagation delay of the link from a source/sink to its corresponding router is uniformly distributed in [10, 20] ms. Note that the delay and bandwidth settings of the access links (links from the sources to router r_0 and from router r_1 to the sinks) are not intended to mimic access links in practice. Rather, they are chosen so that the flows have different Round Trip Times (RTTs) and the bottleneck is the link between routers r_0 and r_1.

One of the TCP flows is used to stream video, and is referred to as the *video stream*; the rest of the TCP flows are for FTP. For the video stream, let D denote the round trip propagation delay; p the average loss rate; R the RTT (R is the sum of D and the queuing delay on the round trip) and R_{TO} the value of the first retransmission timer. We further define $T_O = R_{TO}/R$. Since R_{TO} is based on the average and the variance of round trip times, T_O reflects the variation of the RTTs. For simplicity, T_O is rounded to the closest integer. In all of the results, the video stream uses TCP Reno. For live and stored-media streaming, the video length is 7000 and 80 seconds respectively. The choice of such video lengths for live and stored-media streaming is for the convenience of validation. We vary the video length in Section 6.1. In particular, we show that the model for live streaming is accurate for a wide range of media lengths. We also show that, in stored-media streaming, it is sufficient to model a relatively short video and we provide a method to obtain the fraction of late packets for longer videos.

The link from router r_0 and r_1 forms a bottleneck link where packet losses occur due to buffer overflow. We create different settings by varying the bandwidth, buffer size and the propagation delay of the

Table II. Live Streaming: Five Settings for Model Validation in *ns*

	# of Sources		Link from r_0 to r_1			Parameters of Video Stream			
Setting	TCP	HTTP	Prop. Delay (ms)	B.w. (Mbps)	Buffer (pkts)	D (ms)	Loss Rate	R (ms)	T_O
1	7	40	5	3.7	100	50	0.004	285	3
2	10	40	40	3.7	50	120	0.019	210	2
3	6	15	5	5	80	50	0.007	165	5
4	8	30	1	5	50	42	0.014	113	4
5	5	20	1	5	30	22	0.014	64	6

Table III. Live Streaming: Selected Runs for the Five Settings for Model Validation in *ns*

Setting	Range of Loss Rate	# of Selected Runs	T (pkts ps)	μ (pkts ps)	T/μ
1	[0.003, 0.005]	37	49.7	25	1.99
2	[0.017, 0.021]	40	31.7	25	1.27
3	[0.006, 0.008]	32	64.9	50	1.30
4	[0.013, 0.014]	20	68.1	25	2.7
5	[0.012, 0.016]	22	105.5	50	2.1

bottleneck link as well as the number of flows (TCP and HTTP) traversing the bottleneck link. For each setting, we run multiple simulations to obtain a confidence interval. For a fixed setting, we find the values of R and T_O among different runs to be close. However, due to the randomness in the background traffic, the loss (packet drop) rate for the video stream in different runs may vary significantly, especially in stored-media streaming, where the video length, and hence, the simulation run is short. We thus face the problem of validating a model with a given loss rate, p, against multiple simulation runs with varying loss rates. Since our goal is to validate our model for a given value of p, we select simulation runs with loss rate close to p. In particular, we select the runs with loss rate in the range of $(1 \pm \epsilon)p$, where $\epsilon < 0.15$, for model validation. The 95% confidence intervals for the simulations are obtained from the selected runs.

In each setting, we obtain the fraction of late packets from the model and the simulation, denoted as f_m and f_s respectively. We say the model and the simulation match well if one of the following two conditions is satisfied: f_m falls within the confidence interval obtained from the simulation, or $\frac{1}{5} \leq \frac{f_m}{f_s} \leq 5$. The reason for the second condition is as follows. We use the fraction of late packets to roughly measure the viewing quality. When f_m and f_s lie within the same order of magnitude of each other, we regard that they correspond to similar viewing quality (the quality is classified as either satisfactory or unsatisfactory since our ultimate goal is to determine the conditions under which TCP provides satisfactory streaming performance).

4.1 Validation for Live Streaming

We validate the live-streaming model using the five settings listed in Table II. In these settings, the number of TCP sources varies from 5 to 10. The number of HTTP sources varies from 15 to 40. The buffer size of router r_0 is 30, 50 or 100 packets. The bandwidth of the link from r_0 to r_1 is 3.7 or 5 Mbps. The propagation delay from r_0 to r_1 is 1, 5 or 40 ms. A TCP flow is associated with a CBR source for video streaming. The playback rate of the video is 25 or 50 packets per second and each packet is 1500 bytes. Therefore, the bandwidth of the video is 300 or 600 Kbps. The various parameters for the media stream are listed in Table II: the round trip propagation delay ranges from 22 or 120 ms; the loss rate ranges from 0.004 to 0.019; R ranges from 64 to 285 ms and T_O ranges from 2 to 6. Table III lists the selected range of loss rate and the number of selected runs in each setting. Let T represent the available TCP throughput for the video stream. Then when $T \geq \mu$, T/μ represents how much the achievable TCP

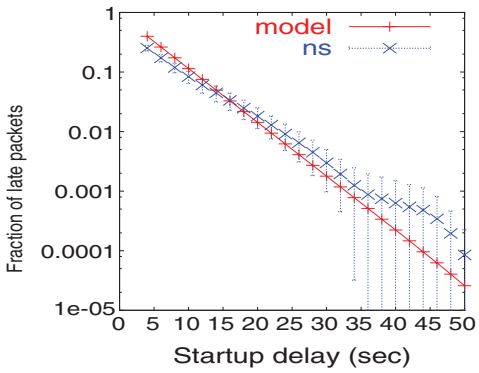


Fig. 4. Live streaming (Setting 2): fraction of late packets versus the startup delay for a 7000-second video.

Table IV. Stored-Media Streaming: Five Settings for Model Validation in *ns*

Setting	# of Sources		Link from r_0 to r_1			Parameters of Video Stream			
	TCP	HTTP	Prop. Delay (ms)	B.w. (Mbps)	Buffer (pkts)	D (ms)	Loss Rate	R (ms)	T_O
1	9	40	40	3.7	50	120	0.022	220	2
2	5	30	40	3.7	50	120	0.006	195	2
3	9	40	5	5	100	50	0.015	162	3
4	5	30	5	5	100	50	0.014	110	3
5	6	20	1	5	30	22	0.029	59	6

Table V. Stored-Media Streaming: Selected Runs for the Five Settings for Model Validation in *ns*

Setting	Range of Loss Rate	# of Selected Runs	T (pkts ps)	μ (pkts ps)	T/μ
1	[0.020, 0.024]	431	30.8	28	1.1
2	[0.005, 0.007]	364	66.5	60	1.1
3	[0.013, 0.017]	482	46.1	42	1.1
4	[0.012, 0.016]	554	71.4	65	1.1
5	[0.026, 0.033]	562	67.1	61	1.1

throughput is higher than the video playback rate. In each setting, the fraction of late packets predicted by the model is compared to that from the simulation. We next describe the validation for Setting 2 in detail; the results for other settings being similar [Wang et al. 2004].

We generate 60 simulation runs in Setting 2, each run lasting for 7000 seconds. The video length is 7000 seconds. The average loss rate of all the runs is 0.019. We use $p = 0.019$ in the model and select runs with loss rates in the range of 0.017 to 0.021 for the reason given earlier. Among the selected 40 runs, the values of R and T_O are close with averages of 210 ms and 2 respectively. These values are used in the model to obtain the fraction of late packets. Figure 4 depicts the fraction of late packets versus the startup delay predicted by the model and obtained from the simulation. We observe a good match between the model and the simulation.

4.2 Validation for Stored-Media Streaming

We validate the model for stored-media streaming using five settings listed in Table IV. A TCP flow is used for store-media streaming. The various parameters of this video stream (including p, R, and T_O) are estimated and listed in Table IV. Table V lists the selected range of loss rate and the number of selected runs in each setting. The playback rate of the video is chosen so that the achievable TCP

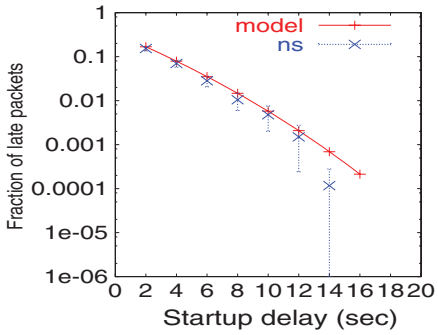


Fig. 5. Stored-media streaming (Setting 4): fraction of late packets versus the startup delay for a playback rate of 65 packets per second.

throughput is 1.1 times the video playback rate. For each setting, we compare the results obtained from the model to those obtained from the simulation. We next describe the validation for Setting 4 in detail; the results for other settings are similar [Wang et al. 2004].

We generate 1000 simulation runs in Setting 4, each lasting for 100 seconds. We assume the length of the video to be 80 seconds, corresponding to approximately the initial 80 seconds of a simulation run. The average loss rate of the video stream in the 1000 runs is 0.014. We use $p = 0.014$ in the model and select the runs with loss rates between 0.012 to 0.016. There are a total of 554 such runs. For the selected runs, the average RTT and T_O are 110 ms and 3 respectively; the average TCP throughput is 71.4 packets per second. We set the playback rate of the video to be 65 packets per second. That is, the available TCP throughput is 10% higher than the video playback rate. Figure 5 depicts the fraction of late packets versus startup delays. The results predicted by the model and those obtained from the simulation are shown in the figure. Again, we observe a good match between the model and the simulation. We now look at the probability that no late loss occurs during the playback of the video. For a startup delay of 6 seconds, at playback rates of 51, 55 and 60 packets per second, the probabilities of experiencing no late packet throughout the 80-second video are 0.98, 0.96 and 0.91 respectively from the simulation. The lower bounds on these probabilities given by Theorem 1 are 0.82, 0.42, and 0.01 respectively. The lower bounds are not very tight compared to the simulation results. This is likely due to the independence assumption used in deriving the bound.

5. MODEL VALIDATION AGAINST INTERNET EXPERIMENTS

In this section, we validate the models for live and stored-media streaming through experiments conducted over the Internet. In each experiment, we use *tcpdump* (http://www.tcpdump.org) to capture the packet timestamps. The average loss rate p, average RTT R and T_O of this TCP flow are estimated from the *tcpdump* traces. We use Linux-based machines for all experiments.

5.1 Validation for Live Streaming

We validate the model for live streaming through two sets of experiments over the Internet. In each set of experiments, we choose two servers for a client, one physically close and the other physically far away from the client. The first set of experiments are conducted from University of Massachusetts (UMass) or University of Southern California (USC) to a client in a residence (using cable modem for the Internet connection) in Amherst, Massachusetts. The second set of experiments are conducted from Stanford University or Brown University to a client (using ADSL for the Internet connection) at University of California, Berkeley (UCB). All three hosts used in this set of experiments are from PlanetLab

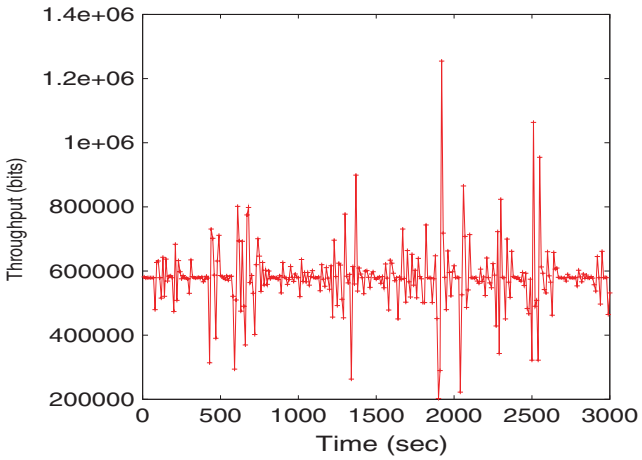
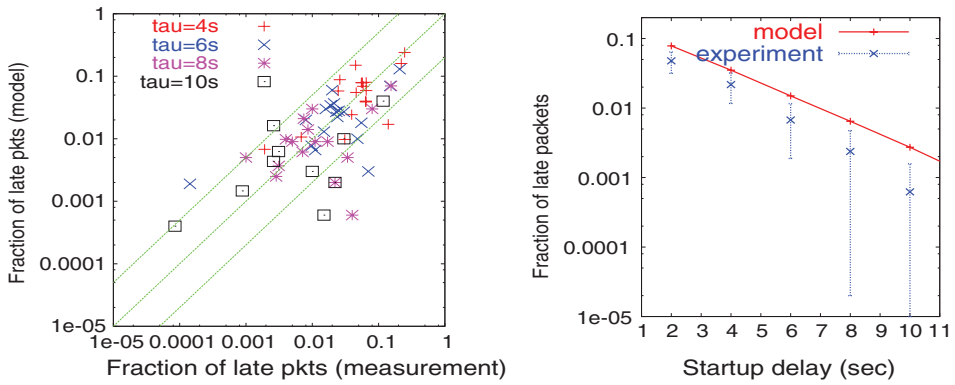


Fig. 6. The TCP throughput of an experiment from USC to the client in the residence at Amherst, MA.

(http://www.planet-lab.org). In each experiment, we stream a CBR video using TCP from the server to the client. The playback rate of the video is 25, 40, or 50 packets per second and each packet consists of 1448 bytes. That is, the bandwidth of the video is approximately 300, 480 or 600 Kbps. Each experiment lasts for one hour. The first set of experiments are conducted from February 19 to March 7, 2003, at randomly chosen times. The second set of experiments are conducted from November 14 to November 19, 2006.

For each experiment, we plot the time series of the TCP throughput, where each point is the average throughput over a 10-second interval. Based on the throughput series, we choose segments of length 500 to 1000 seconds that exhibit variations in throughput, implying the occurrence of congestion. The segments are chosen by visual inspection although more rigorous methods can be used [Bendat and Peirsol 1986]. Each segment is treated as a separate video. The loss rate, RTT and T_O are obtained from the data segment and used in the model. We use one trace to illustrate our procedure. Figure 6 plots the TCP throughput averaged over every 10 seconds for one experiment. We choose the first, second and third 1000 seconds of the trace as three separate 1000-second videos to validate the model against the measurements.

We obtained 12 and 7 segments from these two sets of experiments respectively. In the first set of experiments, the loss rate varies from 0.0001 to 0.003; the round trip time from UMass to the resident home varies from 120 ms to 300 ms; the round trip time from USC to the resident home (at Amherst, MA) varies from 250 ms to 330 ms. In the second set of experiments, the loss rate varies from 0.0009 to 0.003; the round trip time from Stanford University to UCB varies from 260 ms to 350 ms; the round trip time from Brown University to the client (at UCB) varies from 400 ms to 430 ms. Figure 7(a) presents a scatterplot showing the fraction of late packets for various startup delays obtained from the measurements versus that predicted by the model. The startup delay is between 4 to 10 seconds. The 45 degree line starting at the origin represents a hypothetical perfect match between the measurements and the model. Along the upper and lower 45 degree lines, the fraction of late packets from the model is respectively 5 times higher and lower than that from the measurements. All but 9 scatterplot points fall within the upper and lower 45 degree lines, indicating a match between the model and the Internet experiments. We speculate that the 9 bad matches are due to an insufficient number of samples in the data segment.



(a) Live streaming. (b) Stored-media streaming.

Fig. 7. Model validation using experiments over the Internet.

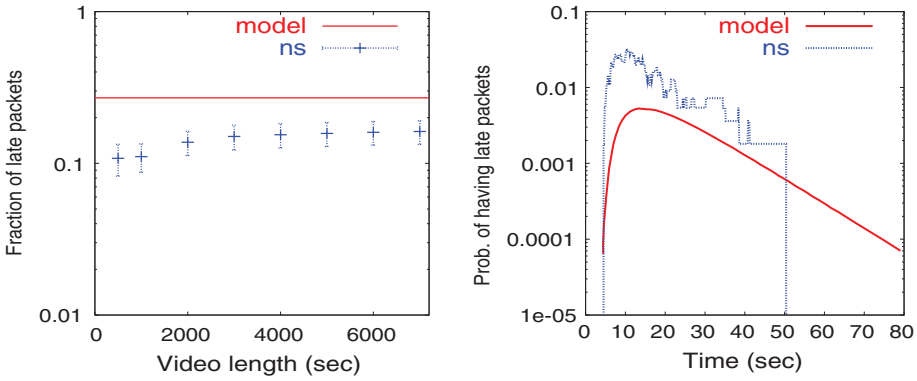
5.2 Validation for Stored-Media Streaming

We next compare model predictions against measurements taken over the Internet for stored-media streaming. In each experiment, we run 8 parallel TCP connections to obtain a group of runs with similar TCP parameters (loss rate, RTT and T_O). Since the bandwidth for a cable modem or ADSL connection is too low to benefit from parallel TCP connections, we chose a high-bandwidth university path. Paths connecting universities in the US are over Internet2 (http://www.internet2.edu/) and are very lightly used. We therefore chose a path between a university at the US and a university in Europe. In particular, the server is at UMass and the client is at Universita' dell'Aquila, Italy. Each experiment lasts for 1 hour. We then divide the trace for each TCP flow into multiple segments, each of 100 seconds. Each 100-second segment is treated as a 100-second video. We use $p = 0.031$ in the model and select 266 segments having loss rate between 0.027 and 0.035 (in the range of $(1 \pm \epsilon)p$, where $\epsilon < 0.15$, for the same reasons as those in Section 4). For the selected segments, the RTT is 300 ms and $T_O = 1$. The average throughput is 15.2 packets per second. We set the playback rate of the video to be 14 packets per second. Correspondingly, the available TCP throughput is 9% higher than the playback rate of the video. Figure 7(b) plots the fraction of late packets for various startup delays. The fraction of late packets predicted by the model is slightly higher than that from the measurements. This might be because, at the beginning of the video streaming, the window size is always one in the model while it may be larger than one in the measurement data segment.

6. EXPLORING THE PARAMETER SPACE

In this section, we vary the model parameters in live and stored-media streaming to study the impact of these parameters on performance. In doing so, we provide guidelines as to when TCP streaming leads to satisfactory performance.

We set the values of the parameters in TCP (i.e., loss rate, R and T_O) to represent a wide range of scenarios. The loss rate is varied in the range of 0.004 to 0.04. Previous work shows that the median RTT between two sites on the same coast in the US is 50 ms, while the median RTT between west-coast and east-coast sites is 100 ms [Huffaker et al. 2001]. Consequently, we vary R in the range of 40 ms to 300 ms. We vary T_O from 1 to 4, based on several measurements from Linux machines in Padhye et al. [1998] and our measurements. In the following, we first explore how the performance of live and stored-media streaming varies with the length of the video. We then identify the conditions under which TCP streaming provides a satisfactory viewing experience. At the end, we summarize the key results.



(a) Live streaming. (b) Stored-media streaming.

Fig. 8. Effect of video length on performance.

6.1 Effect of Video Length on Performance

We first use the setting in Section 4.1 to illustrate the effect of the video length on the performance in live streaming. The startup delay is set to 6 seconds and the length of the video ranges from 500 to 7000 seconds. Figure 8(a) plots the fraction of late packets versus the video length from the model and the *ns* simulation. The model provides a good prediction for various video lengths. For videos longer than 2000 seconds, the fractions of late packets for various video lengths from the simulation are similar. Throughout this section, we assume the video for live streaming is sufficiently long so that stationary analysis can be used to obtain the fraction of late packets.

We next use the setting in Section 4.2 to investigate how the fraction of late packets varies with video length in stored-media streaming. The startup delay is set to 4 seconds. The playback rate is 51 packets per second. Correspondingly, the available TCP throughput is 40% higher than the video playback rate. We obtain P_i $(i = 1, \ldots, L)$, the probability that the i-th round has at least one late packet (see Section 3.3.2). Figure 8(b) plots P_i over the length of the video from the model and the simulation. In the figure, the fraction of late packets is low at the beginning of the video, increases to a maximum and then decreases over time. This can be explained as follows. At the beginning of playback, the probability of having a late packet in a round is low due to the packets accumulated in the client local buffer during the startup delay. Subsequently, packets are played out while at the same time being accumulated in the client buffer. The number of early packets in the buffer increases with time since, on average, the achievable throughput is higher than the playback rate of the video. Therefore, the probability of having late packets in a round reaches a peak value and then decreases over time as the number of early packets in the buffer increases. In Figure 8(b), the probability of having a late packet in the 730th round (i.e., 80th second) decreases to 10^{-4}. This indicates that, after 730 rounds, the fraction of late packets is approximately inversely proportional to the length of the video, since the probability of having late packets after 730 rounds is close to 0. This is confirmed by the simulation results. In general, to obtain the fraction of late packets, f, for a video of L rounds, it is sufficient to obtain the fraction of late packets in the initial l rounds of the video, denoted as f_l, such that P_l is close to 0. Then $f = lf_l/L$. Throughout this section, we use videos of 80 seconds for stored-media streaming.

6.2 Conditions for Satisfactory Performance

In general, viewing quality is satisfactory when the fraction of late packets is low for a short startup delay. People can usually tolerate a few seconds of startup delay. Studies show that the video quality

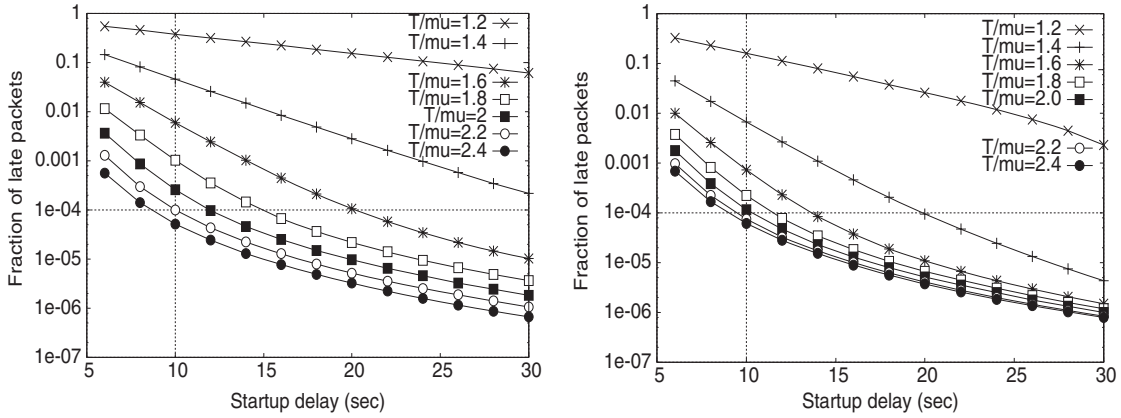


Fig. 9. Diminishing gain from increasing T/μ on performance in live streaming, $p = 0.02$, $T_O = 4$. (a) Vary T/μ by fixing play back rate to 25 packets per second and varying RTT. (b) Vary T/μ by fixing RTT to 100 ms and varying play back rate.

drops when the packet loss rate exceeds 10^{-4} [Verscheure et al. 1998]. Consequently, we assume that the performance of TCP streaming is satisfactory when the fraction of late packets is below 10^{-4} for a startup delay of around 10 seconds.[2]

Recall that we denote the achievable TCP throughput as T packets per second. When $T/\mu \geq 1$, T/μ represents how much greater the achievable TCP throughput is than the video playback rate. The performance of TCP streaming improves as T/μ increases [Wang et al. 2004]. This is intuitive since packets accumulate in the client's local buffer faster relative to the playback rate of the video as T/μ increases. We next vary the value of T/μ in order to identify the minimum value of T/μ that leads to satisfactory performance. For convenience, we define T_R to be the achievable TCP throughput in one RTT. Then $T = T_R/R$. Since $T/\mu = T_R/(\mu R)$ and T_R is determined by p and T_O, we vary the value of T/μ by fixing p and T_O and varying either the playback rate μ or the RTT R. In particular, we fix p to be 0.004, 0.02 or 0.04, corresponding to low, medium and high loss rates respectively, and fix T_O to be 1, 2, 3 or 4.

6.2.1 *Live Streaming.* For fixed values of p and T_O, we increase T/μ from 1.2 to 2.4 by either fixing RTT and decreasing the video playback rate or fixing the playback rate and decreasing RTT. We observe a diminishing gain from increasing T/μ on performance: performance improves dramatically as T/μ increases from 1.2 to 1.6 and less dramatically afterwards. Two examples are shown in Figures 9(a) and (b) respectively. In both figures, $p = 0.02$ and $T_O = 4$. In Figure 9(a), the playback rate of the media is fixed to 25 packets per second and RTT is varied such that T/μ ranges from 1.2 to 2.4. In Figure 9(b), the RTT is fixed to 100 ms and the playback rate is varied. This diminishing gain indicates that, to achieve a low fraction of late packets, the required startup delay is large when T/μ is only slightly higher than 1 and reduces quickly as T/μ increases. However, the reduction becomes less dramatic for large values of T/μ.

We now explore the required startup delay such that the fraction of late packets, f, lies below 10^{-4}. In Figure 10(a), the playback rate is fixed to 25 packets per second and the RTT is varied such that T/μ ranges from 1.2 to 2.4 for various loss rates and $T_O = 4$ (the required startup delay for lower values

[2]Some people may be willing to tolerate a longer startup delay in stored-media streaming than that in live streaming. However, our focus is on on-demand streaming for both forms of streaming where the startup delay is a few seconds.

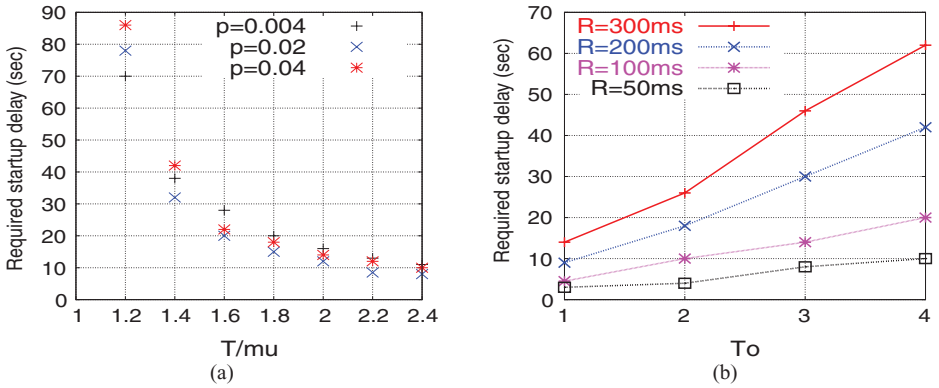


Fig. 10. Live streaming: the required startup delay such that the fraction of late packets $f \leq 10^{-4}$. (a) Fixing playback rate to 25 packets per second and varying RTT such that T/μ ranges from 1.2 to 2.4, $T_O = 4$; (b) fixing RTT to 50, 100, 200 or 300 ms and varying play back rate such that $T/\mu = 2$, $p = 0.04$.

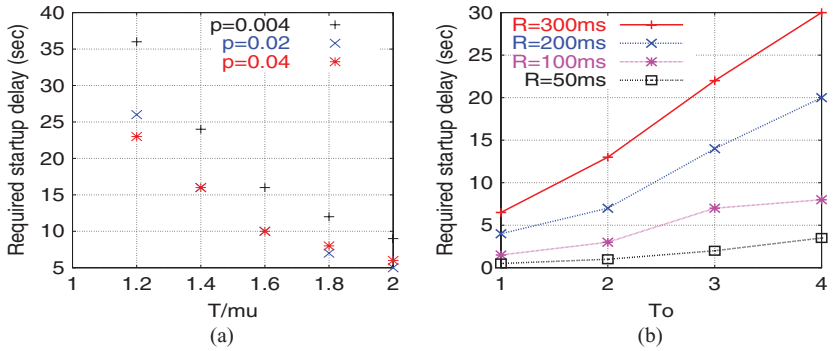


Fig. 11. Stored-media streaming: the required startup delay such that the fraction of late packets $f \leq 10^{-4}$ (a) fixing playback rate to 25 packets per second and varying RTT such that T/μ ranges from 1.2 to 2, $T_O = 4$; (b) fixing RTT to 50, 100, 200 or 300 ms and varying play back rate such that $T/\mu = 2$, $p = 0.04$.

of T_O is lower for the same loss rate and T/μ). We observe that under various settings, performance becomes satisfactory when T/μ is roughly 2. In Figure 10(b), the value of RTT is 50, 100, 200 or 300 ms and the playback rate is varied such that $T/\mu = 2$. In this figure, $p = 0.04$; the required startup delay for lower loss rates is lower (figures omitted). When $R = 50$ ms (corresponding roughly to two sites on the same coast in the US), the required startup delay is no more than 10 seconds under all settings. When $R = 100$ ms (corresponding roughly to two sites on the two coasts in the US), the required startup delay is no more than 10 seconds under all settings except for very high loss rate ($p = 0.04$) and high T_O values ($T_O \geq 3$). However, for a large RTT, high loss rate and timeout value, the required startup delay is in tens of seconds.

6.2.2 *Stored-Media Streaming.* In stored-media streaming, when increasing T/μ from 1.2 to 2.4, similar results as those in live streaming are observed; performance gains diminish when increasing T/μ and the performance becomes satisfactory once the achievable TCP throughput is twice the video bitrate. Figure 11(a) depicts the required startup delay such that the fraction of late packets lies below 10^{-4} when fixing the playback rate and varying the RTT, where $T_O = 4$ (the required startup delay for lower T_O values is lower). We observe that the required startup delay is less than 10 seconds when

T/μ increases to 2. Figure 11(b) depicts the required startup delay when fixing the value of RTT to 50, 100, 200 or 300 ms and varying the playback rate of the video such that $T/\mu = 2$, where $p = 0.04$ (the required startup delay for lower loss rates is lower). Under relatively small RTTs, i.e., $R = 50$ or 100 ms, the required startup delay is within 10 seconds for all the settings. However, for a long RTT, high loss rate and timeout value, the required startup delay is in tens of seconds.

6.3 Summary of Results

The key results from our exploration of the parameter space are:

—When TCP throughput is no less than the video bitrate, in live streaming, the fraction of late packets is insensitive to video length once the video is sufficiently long; in stored-media streaming, the fraction of late packets decreases with video length (after an initial increasing trend at the beginning of the playback).

—The performance of TCP streaming improves as the value of T/μ increases. Furthermore, increasing T/μ beyond a point yields diminishing performance gain.

—The performance of TCP streaming is not solely determined by T/μ but is sensitive to the values of the various parameters in the models. However, the performance is generally good when the achievable TCP throughput is roughly twice the video bitrate, when allowing a few seconds of startup delay.

—For large RTTs, high loss rates and timeout values, to achieve a low fraction of late packets, either a long startup delay or a large T/μ (greater than 2) is required.

7. CONCLUSIONS

In this article, we developed discrete-time Markov models for live and stored-media streaming. Our validation using *ns* and Internet experiments showed that the performance predicted by the models are accurate. Using the models, we studied the effect of various parameters on the performance of live and stored-media streaming. In doing so, we provided guidelines as to when direct TCP streaming renders satisfactory performance, showing, for example, that TCP generally provides good streaming performance when the achievable TCP throughput is roughly twice the video bitrate, with only a few seconds of startup delay. Note that our model can be easily extended to the settings where loss rate varies during the playout of the video by incorporating the various loss rate values into the Markov models. Last, we use fraction of loss rate as the performance metric throughout the paper. Performance study using more complicated and user-oriented metrics is left as future work.

APPENDIXES

A. MODEL FOR LIVE STREAMING

In live streaming, the state of the model in the i-th round, Y_i^l, is represented as (X_i, N_i), where X_i and N_i are the state of the TCP source and the number of early packets in the i-th round respectively. As in [Padhye et al. 1999; Figueiredo et al. 2002], X_i, is represented as a tuple: $X_i = (W_i, C_i, L_i, E_i, R_i)$. W_i is the window size for round i. C_i models the delayed ACK behavior of TCP; $C_i = 0$ and $C_i = 1$ indicate that the first and the second of the two rounds respectively. L_i is the number of packets lost in the $(i-1)$th round. E_i denotes whether the connection is in a timeout state and the value of the back-off exponent in round i. R_i indicates a packet being sent in the timeout phase is whether a retransmission ($R_i = 1$) or a new packet ($R_i = 0$).

Let $p_{w,c,l,e,r,n;w',c',l',e',r',n'} = P(W_{i+1} = w', C_{i+1} = c', L_{i+1} = l', E_{i+1} = e', R_{i+1} = r', N_{i+1} = n' \mid W_i = w, C_i = c, L_i = l, E_i = e, R_i = r, N_i = n)$ be the probability associated with the state transition from state $(W_i = w, C_i = c, L_i = l, E_i = e, R_i = r, N_i = n)$ to state $(W_{i+1} = w', C_{i+1} = c', L_{i+1} = l',$

Table VI. Live Streaming: Definition of the State Transition Probabilities and the Times Taken for the Transitions

$p_{w,0,0,0,0,n;w,1,0,0,n'}$	=	$(1-p)^w$,	$1 \le w \le W_{max}$ $n' = \min(N_{max}, n+w)$
$p_{w,1,0,0,0,n;w+1,0,0,0,n'}$	=	$(1-p)^w$,	$1 \le w \le W_{max}$, $n' = \min(N_{max}, n+w)$
$p_{w,1,0,0,0,n;w,0,0,0,n'}$	=	$(1-p)^w$,	$w = W_{max}$, $n' = \min(N_{max}, n+w)$
$r_{w,0,0,0,0,n;w,1,0,0,n'}$	=	R,	$1 \le w \le W_{max}$
$r_{w,1,0,0,0,n;w+1,0,0,0,n'}$	=	R,	$1 \le w \le W_{max}$
$r_{w,1,0,0,0,n;w,0,0,0,n'}$	=	R,	$1 \le w \le W_{max}$
$p_{w,c,0,0,0,n;w-l,0,l,0,0,n'}$	=	$p(1-p)^{w-l}$,	$2 \le w \le W_{max}$, $c = 0, 1, 1 \le l \le w$, $n' = \min(N_{max}, n+w-l)$
$p_{w,c,0,0,0,n;1,0,0,1,1,n}$	=	p,	$2 \le w \le W_{max}, c = 0, 1$
$r_{w,c,0,0,0,n;w-l,0,l,0,0,n'}$	=	R,	$2 \le w \le W_{max}$, $c = 0, 1, 1 \le l \le w$,
$r_{w,c,0,0,0,n;1,0,0,1,1,n}$	=	R_{TO},	$2 \le w \le W_{max}, c = 0, 1$
$p_{1,0,l,0,0,n;1,0,0,1,1,n'}$	=	1,	$n' = \min(N_{max}, n+1-p)$
$p_{2,0,l,0,0,n;1,0,0,1,1,n'}$	=	1,	$n' = \min(N_{max}$, $n + p(1-p) + 2(1-p)^2)$
$p_{w,0,l,0,0,n;1,0,0,1,1,n'}$	=	$\sum_{i=1}^{2} p(1-p)^i$,	$3 \le w \le W_{max}$ $n' = \min(N_{max}$, $n + \frac{\sum_{i=0}^{2} ip(1-p)^i}{\sum_{i=0}^{2} p(1-p)^i})$
$p_{w,0,l,0,0;\lfloor (w+l)/2 \rfloor,0,0,0,0}$	=	$\sum_{i=3}^{w-1} p(1-p)^i + (1-p)^w$,	$3 \le w \le W_{max}$, $n' = \min(N_{max}$, $n + \frac{\sum_{i=3}^{w-1} ip(1-p)^i + w(1-p)^w}{\sum_{i=3}^{w-1} p(1-p)^i + (1-p)^w})$
$r_{w,0,l,0,0,n;1,0,0,1,1,n'}$	=	$R_{TO} - R$,	$1 \le w \le W_{max}$
$r_{w,0,l,0,0;\lfloor (w+l)/2 \rfloor,0,0,0,0}$	=	R,	$3 \le w \le W_{max}$
$p_{1,0,0,i,r,n;1,0,0,\min\{i+1,7\},1,n}$	=	p,	$1 \le i \le 7, r = 0, 1$
$p_{1,0,0,i,1,n;1,0,0,i,0,n+1}$	=	$1-p$,	$1 \le i \le 7$
$p_{1,0,0,i,0,n;2,0,0,0,0,n+1}$	=	$1-p$,	$1 \le i \le 7$
$r_{1,0,0,i,r,n;1,0,0,\min\{i+1,7\},1,n}$	=	$2^{(i-1)}R_{TO}$,	$1 \le i \le 7, r = 0, 1$
$r_{1,0,0,i,1,n;1,0,0,i,0,n+1}$	=	R,	$1 \le i \le 7$
$r_{1,0,0,i,0,n;2,0,0,0,0,n+1}$	=	R,	$1 \le i \le 7$
$r_{w,c,l,e,r,n;w,c,l,e,r,n'}$	=	R,	$n' = n - \mu R$

$E_{i+1} = e', R_{i+1} = r', N_{i+1} = n'$). Let $r_{w,c,l,e,r,n;w',c',l',e',r',n'}$ be the time taken for this state transition. Let R_{TO} be the value of the first retransmission timer. It is rounded as a multiple of the RTT R. Denote the maximum window size as W_{max}. The state transition probabilities and the times taken for the transitions are listed in Table VI. In the table, there are 5 groups of p's and r's corresponding respectively to situations (1) no packets are lost in a round; (2) one or more packets are lost in a round; (3) one or more packets are lost in a short round; (4) exponential back-off; (5) packet playback.

Table VII. Stored-Media Streaming: Definition of Impulse Reward

$\rho_{w,0,0,0,0;w,1,0,0}$	=	$w - \mu R$,	$1 \le w \le W_{max}$
$\rho_{w,1,0,0,0;w+1,0,0,0}$	=	$w - \mu R$,	$1 \le w \le W_{max}$
$\rho_{w,1,0,0,0;w,0,0,0}$	=	$w - \mu R$,	$1 \le w \le W_{max}$
$\rho_{w,c,0,0,0;w-l,0,l,0,0}$	=	$w - l - \mu R$,	$2 \le w \le W_{max}$, $c = 0, 1, 1 \le l \le w$,
$\rho_{w,c,0,0,0;1,0,0,1,1}$	=	$-\mu R$,	$2 \le w \le W_{max}, c = 0, 1$
$\rho_{1,0,l,0,0;1,0,0,1,1}$	=	$1 - p - \mu R_{TO}$,	
$\rho_{2,0,l,0,0;1,0,0,1,1}$	=	$p(1-p) + 2(1-p)^2 - \mu R_{TO}$,	
$\rho_{w,0,l,0,0;1,0,0,1,1}$	=	$\frac{\sum_{i=0}^{2} ip(1-p)^i}{\sum_{i=0}^{2} p(1-p)^i} - \mu R_{TO}$,	$1 \le w \le W_{max}$
$\rho_{w,0,l,0,0;\lfloor (w+l)/2 \rfloor,0,0,0,0}$	=	$\frac{\sum_{i=3}^{w-1} ip(1-p)^i + w(1-p)^w}{\sum_{i=3}^{w-1} p(1-p)^i + (1-p)^w} - \mu R$,	$3 \le w \le W_{max}$
$\rho_{1,0,0,i,r;1,0,0,\min\{i+1,7\},1}$	=	$-\mu 2^{(i-1)} R_{TO}$,	$1 \le i \le 7, r = 0, 1$
$\rho_{1,0,0,i,1;1,0,0,i,0}$	=	$1 - \mu R$,	$1 \le i \le 7$
$\rho_{1,0,0,i,0;2,0,0,0,0}$	=	$-\mu R$,	$1 \le i \le 7$

B. MODEL FOR STORED-MEDIA STREAMING

In stored-media streaming, the state of the model in the i-th round $Y_i^s = X_i$, where $X_i = (W_i, C_i, L_i, E_i, R_i)$. The impulse reward $\rho_{w,c,l,e,r;w',c',l',e',r'}$ is associated with a transition from state $(W_i = w, C_i = c, L_i = l, E_i = e, R_i = r)$ to state $(W_{i+1} = w', C_{i+1} = c', L_{i+1} = l', E_{i+1} = e', R_{i+1} = r')$. This impulse reward is defined in Table VII. In the table, there are 4 groups of ρ's corresponding respectively to situations (1) no packets are lost in a round; (2) one or more packets are lost in a round; (3) one or more packets are lost in a short round; (4) exponential back-off.

C. PROOF OF THEOREM 1

PROOF. Let A_i be the total number of packets reaching the client up to the i-th round. Let B_i be the total number of packets played back by the client up to the i-th round. Then A_i and B_i are respectively the discrete-time version of $A(t)$ and $B(t)$ introduced in Section 3.1. It is clear that $N_i = A_i - B_i$ and

$$B_i = \begin{cases} \mu(iR - \tau), & \text{if } iR \ge \tau \\ 0, & \text{o.w.} \end{cases}$$

Recall that S_k is the number of packets sent out successfully by TCP in the k-th round (defined in Section 3.2). Then $A_i = \sum_{k=1}^{i} S_k$. Since the time unit in the model is the length of a round, we have

$$\begin{aligned} \alpha &= P(N_1 \ge 0, N_2 \ge 0, \ldots, N_L \ge 0) \\ &= P(A_1 \ge B_1, A_2 \ge B_2, \cdots, A_L \ge B_L). \end{aligned}$$

It is clear that A_i is a nondecreasing function of S_i. If S_i's are associated, then by Esary et al. [1967], we have

$$\alpha \ge \Pi_{i=1}^{L} P(A_i \ge B_i) = \Pi_{i=1}^{L} (1 - P_i),$$

where $P_i = P(N_i < 0) = P(A_i < B_i)$. We next sketch a proof that S_i's are associated by only considering the congestion control behavior in TCP. The proof when also considering time out behavior is similar.

We first define random variables χ_i as

$$\chi_i = \begin{cases} 1, & \text{congestion occurs in round } i \\ 0, & \text{o.w.} \end{cases}$$

Then by the mechanism of TCP, we have

$$S_i = \max(S_{i-1} + 1, W_{max})(1 - \chi_i) + S_{i-1}\chi_i/2.$$

It is observed that packet losses in TCP follow a Poisson process [Misra et al. 1999]. We therefore assume χ_i's are independent. Then by Property P_4 in Esary et al. [1967], S_i's are associated. □

ACKNOWLEDGMENTS

We thank E. de Souza e Silva and D. R. Figueiredo for their help with the numerical solution of Markov models. We thank L. Golubchik, C. Papadopulous and F. L. Presti for providing us accounts for the Internet experiments. We also thank W. Gong, P. Key, S. Sen and W. Wei for helpful discussions. Last, we thank Z. Guo for helping us conduct several Internet experiments.

REFERENCES

ALTMAN, E., AVRACHENKOV, K., AND BARAKAT, C. 2000. A stochastic model of TCP/IP with stationary random losses. In *Proceedings of the SIGCOMM*. 231–242.

BENDAT, J. S. AND PEIRSOL, A. G. 1986. *Random Data Analysis and Measurement Procedures*. John Wiley & Sons.

BOHACEK, S. 2003. A stochastic model of TCP and fair video transmission. In *Proceedings of IEEE INFOCOM*.

BOUTREMANS, C. AND LE BOUDEC, J. Y. 2003. Adaptive joint playout buffer and FEC adjustment for Internet telephony. In *Proceedings of IEEE INFOCOM'*. San-Francisco, CA.

CARDWELL, N., SAVAGE, S., AND ANDERSON, T. 2000. Modeling TCP latency. In *Proceedings of INFOCOM*. Vol, 3, 1742–1751.

DE CUETOS, P., GUILLOTEL, P., ROSS, K. W., AND THOREAU, D. 2002. Implementation of adaptive streaming of stored MPEG-4 FGS video over TCP. In *International Conference on Multimedia and Expo (ICME02)*.

DE CUETOS, P. AND ROSS, K. W. 2002. Adaptive rate control for streaming stored fine-grained scalable video. In *Proceedings of the ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*.

DE SOUZA E SILVA, E. AND GAIL, H. R. 1998. An algorithm to calculate transient distribution of cumulative rate and impulse based reward. *Stochastic Models 14*, 3, 509–536.

DE SOUZA E SILVA, E. AND LEAO, R. M. M. 2000. The TANGRAM-II environment. In *Proceedings of the 11th International Conference on Modeling Tools and Techniques for Computer and Communication System Performance Evaluation (TOOLs '00)*.

ESARY, J. D., PROSCHAN, F., AND WALKUP, D. W. 1967. Association of random variables, with applications. *Annals of Mathe. Stat. 38*, 5 (October), 1446–1474.

FIGUEIREDO, D. R., LIU, B., MISRA, V., AND TOWSLEY, D. 2002. On the autocorrelation structure of TCP traffic. *Comput. Netw. J. (Special Issue on Advances in Modeling and Engineering of Long-Range Dependent Traffic)*.

FLOYD, S., HANDLEY, M., PADHYE, J., AND WIDMER, J. 2000. Equation-based congestion control for unicast applications. In *Proceedings of SIGCOMM*. Stockholm, Sweden, 43–56.

HUFFAKER, B., FOMENKOV, M., MOORE, D., AND CLAFFY, K. 2001. Macroscopic analyses of the infrastructure: Measurement and visualization of Internet connectivity and performance. In *A Workshop on Passive and Active Measurements*. Amsterdam, The Netherlands.

KIM, T. AND AMMAR, M. 2006. Receiver buffer requirements for video streaming over TCP. In *Proceedings of Visual Communications and Image Processing Conference*. San Jose, CA.

KRASIC, C. AND WALPOLE, J. 2001. Priority-progress streaming for quality-adaptive multimedia. In *Proceedings of the ACM Multimedia Doctoral Symposium*. Ottawa, Canada.

LI, M., CLAYPOOL, M., KINICKI, R., AND NICHOLS, J. 2005. Characteristics of streaming media stored on the Web. *ACM Trans. Internet Tech. 5*, 5.

MATHIS, M., SEMKE, J., AND MAHDAVI, J. 1997. The macroscopic behavior of the TCP congestion avoidance algorithm. *Comput. Commun. Rev. 27*, 3.

MELLIA, M., STOICA, I., AND ZHANG, H. 2002. TCP model for short lived flows. *IEEE Comm. Lett. 6*, 2.

MISRA, V., GONG, W., AND TOWSLEY, D. 1999. Stochastic differential equation modeling and analysis of TCP-windowsize behavior. In *Proceedings of the PERFORMANCE*. Istanbul, Turkey.

PADHYE, J., FIROIU, V., AND TOWSLEY, D. 1999. A stochastic model of TCP Reno congestion avoidance and control. Tech. Rep. 99-02, Department of Computer Science, University of Massachusetts, Amherst.

PADHYE, J., FIROIU, V., TOWSLEY, D., AND KRUSOE, J. 1998. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of the ACM SIGCOMM*. Vancouver, CA, 303–314.

REJAIE, R., HANDLEY, M., AND ESTRIN, D. 1999. Quality adaptation for congestion controlled video playback over the Internet. In *Proceedings of the SIGCOMM*. 189–200.

SEELAM, N., SETHI, P., AND CHI FENG, W. 2001. A hysteresis based approach for quality, frame rate, and buffer management for video streaming using TCP. In *Proceedings of the Management of Multimedia Networks and Services*.

SRIPANIDKULCHAI, K., MAGGS, B., AND ZHANG, H. 2004. An analysis of live streaming workloads on the Internet. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*. Taormina, Sicily, Italy, 41–54.

STEVENS, W. 1994. *TCP/IP Illustrated, Vol. 1*. Addison-Wesley.

VAN DER MERWE, J., SEN, S., AND KALMANEK, C. 2002. Streaming video traffic: Characterization and network impact. In *Proceedings of the Seventh International Web Content Caching and Distribution Workshop*.

VERSCHEURE, O., FROSSARD, P., AND HAMDI, M. 1998. MPEG-2 video services over packet networks: Joint effect of encoding rate and data loss on user-oriented QoS. In *Proceedings of the ACM International Workshop on Network and Operating Systems Support for Digital Audio and Video*.

WANG, B., KUROSE, J., SHENOY, P., AND TOWSLEY, D. 2004. Multimedia streaming via TCP: An analytic performance study. Tech. rep. 04-21, Department of Computer Science, University of Massachusetts, Amherst.

WANG, Y., CLAYPOOL, M., AND ZUO, Z. 2001. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*. San Francisco, CA.

Received May 2006; revised December 2006; accepted January 2007