Multimedia Streaming via TCP: An Analytic Performance Study

Bing Wang, Jim Kurose, Prashant Shenoy, Don Towsley

Department of Computer Science
University of Massachusetts, Amherst, MA 01003

ABSTRACT

TCP is widely used in commercial media 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 [Network protocols]: Application (video streaming)

General Terms: Performance.

Keywords: Performance modeling, Multimedia streaming.

1. INTRODUCTION

The rapid deployment of broadband connectivity to the home via cable modem and ADSL technologies has resulted in a significant growth in streaming media usage. The conventional wisdom for media streaming is to use UDP, rather than TCP, as the transport protocol. The primary reason for not using TCP 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 of the research over the past decade focused on developing UDP-based streaming protocols, providing mechanisms for TCP-friendliness and loss recovery [1, 2, 3].

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, a recent measurement study has shown that a significant fraction of commercial streaming traffic uses TCP [4]. This study 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, unlike other TCP streaming approaches [5, 6, 7, 8]; 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. Our models 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. This paper makes the following main contributions:

- We develop discrete-time Markov models for *live* and *stored* video 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 using 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 video 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 the video 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 [9]. 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 paper 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. 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 [5, 6, 7, 8] 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 [5, 6, 7, 8]. Furthermore, it does not require layered video as in [7, 8]. In this paper, we focus on the performance of direct TCP streaming. We expect the performance of more sophisticated approaches like [5, 6, 7, 8] 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 [10, 11, 12, 13, 14] or short-lived flows [15, 16]. In particular, [12] and [14] 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 [12, 14] 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 [17] combines TCP modeling and video transmission. The study provides a model to obtain the probability distribution of TCP congestion window size, which is then applied to determine a TCP-friendly transmission rate for non-TCP video flows. Our work differs from the above study in that we study TCP-based streaming instead of determining the TCP-friendly transmission rate for UDP-based streaming.

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 video streaming using TCP. The key notation introduced in this section is summarized in Table 1 for easy reference.

3.1 Problem setting

Consider a client requesting a video from the server. Corresponding to the request, the server streams the video to the client using TCP. Throughout the paper, we assume that the average TCP throughput is no less than the video bitrate. This guarantees that, on average, the throughput provided by TCP satisfies the requirement for streaming the video. However, fluctuations in the instantaneous TCP throughput can still lead to significant late packet arrivals. 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.

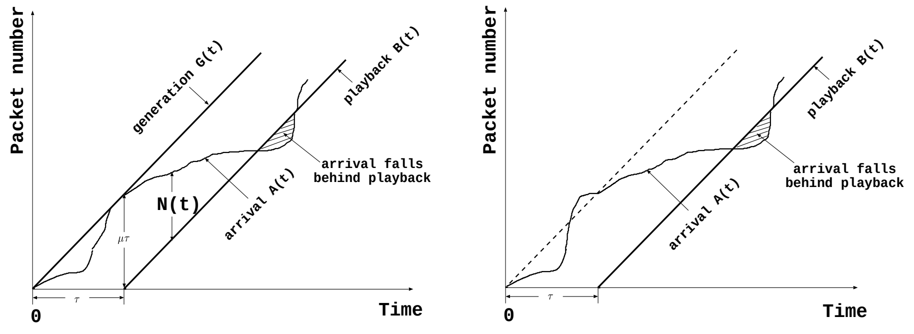
Measurement studies show that most videos streamed over the Internet are constant bit rate (CBR) videos [9]. We therefore consider a CBR video with a playback rate of μ packets per second. For simplicity, all packets are assumed to be of the same size. For analytical tractability, we assume continuous 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*, i.e., the probability that a packet is late, to measure the performance. Strictly speaking, fraction of late packets does not correspond directly to viewing quality, since human perception is tolerant to occasional glitches in the playback. To the best of our knowledge, however, there is no quantitive metric that directly corresponds to viewing quality for videos. We therefore use fraction of late packets as a rough metric of streaming performance.

We study two forms of streaming — live and stored video streaming. In live streaming, the server generates video content in real time and is only able to transmit the content that has already been generated. The transmission is therefore constrained by the generation rate of the video at the application level. Hence we refer to this form of streaming as *constrained streaming*. 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. We refer to this form of streaming as *unconstrained streaming* since the application does not impose any constraint on the transmission. Next, we discuss the characteristics of constrained and unconstrained streaming. For ease of exposition, each packet is associated with a sequence number starting from 1.

Constrained streaming is illustrated in Fig. 1(a). Without loss of generality, we assume that the first packet is generated at time 0. Later packets are generated at a constant rate equal to the playback rate of the video. In the figure, $G(t)$ 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 $A(t) \le G(t)$. 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 $B(t) = \mu(t - \tau)$, $t \ge \tau$. 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 $A(t) \le G(t)$ and $G(t) - B(t) = \mu\tau$, we have $N(t) \le G(t) - B(t) = \mu\tau$. That is, there are at most $\mu\tau$ early packets in constrained streaming at any time t, as shown in Fig. 1(a).

Unconstrained streaming is illustrated in Fig. 1(b). As shown in the figure, the packet transmission is only limited by the achievable TCP throughput and no constraint is imposed from the application level. Therefore, the number of early packets at time t, $N(t)$, can be larger than $\mu\tau$.

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 for TCP in [12, 14] to incorporate the specific characteristics of constrained and unconstrained streaming. In Section 3.3.1, we construct a Markov model for constrained 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 unconstrained streaming. Before describing



(a) Constrained streaming.

(b) Unconstrained streaming.

Figure 1: Video streaming using TCP: constrained and unconstrained streaming.

Notation	Definition
μ	Playback rate of the video (packets per second)
τ	Startup delay (seconds)
X_i	State of the TCP source in the ith round
S_i	Number of packets transmitted successfully by TCP in the ith 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 ith round
N_i^l	Number of late packets in the ith round
Y_i^c	State of the model for constrained streaming in the ith round
Y_i^u	State of the model for unconstrained streaming in the ith round
P_i	Probability of having at least one late packet in the ith round

Table 1: Key notation.

the models for constrained and unconstrained streaming, we first briefly describe the model for TCP.

3.2 Model for TCP

TCP is a window-based protocol with several mechanisms used 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. For every packet sent by the source, TCP starts a retransmission timer and waits for an acknowledgment from the receiver. The retransmission timer expires (times-out) when the ACK for the corresponding packet is lost and there are no triple duplicate ACKs. When 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 be 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. In most versions of TCP, such as TCP Reno and TCP Sack, the window size is reduced by half when triple duplicate ACKs are received. If timeout occurs before receiving triple duplicate ACKs, the window size is reduced to one.

In [12, 14], 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 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 ith round. Following the notation in [14], 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 ith 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 ith 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 ith 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 ith round. A detailed description of S_i can be found in [12, 14, 20].

3.3 Models for constrained and unconstrained streaming

We now present discrete-time Markov models for constrained and unconstrained 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 video whose length is L rounds. The playback rate of the video is μR packets per round.

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 ith round, which is a discrete-time version of $N(t)$ introduced earlier (see Section 3.1) and $N_i = N(iR)$. For simplicity of notation, we assume the number of packets played back in a round, μR, to be an integer. Let N_i^l be the number of late packets in the ith round. Then $N_i^l \in \{0, 1, \ldots, \mu R\}$, where $N_i^l = 0$ indicates that no packet is late in the ith round. Let the expected number of late packets in the

ith round be $E[N_i^l]$. Then

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

where $P(N_i^l = k)$ is the probability of having k late packets in the ith round. The fraction of late packets is

$$f = \frac{\sum_{i=1}^{L} E[N_i^l]}{\mu R L} \tag{2}$$

where the numerator and denominator correspond, respectively, to the expected number of late packets throughout the playback of the video and the total number of packets in the video.

In order to obtain $P(N_i^l = k)$, we introduce N_i^b to be

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

N_i^b can be thought of as the number of packets by which the arrival falls behind the playback of the video in the ith round. Expression (3) follows directly from the definition of N_i and N_i^b. We obtain $P(N_i^l = k)$ as

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

Note that while the number of late packets N_i^l in the ith 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 above, 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 constrained and unconstrained streaming, with a focus on deriving N_i from the models.

3.3.1 *Constrained streaming*

Constrained streaming can be modeled as a producer-consumer problem. The producer produces packets according to TCP and stores them in a buffer. The consumer starts to consume the packets in the buffer from time τ at a constant rate of μR packets per round. At any time, the number of packets in the buffer is no more than N_{max}, $N_{max} = \mu\tau$. This is from an earlier observation that $N_i \leq N_{max}$ ($i = 1, 2, \ldots, L$) due to the constraint of the video generation rate (see 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 constrained streaming.

Let $\{Y_i^c\}_{i=1}^L$ be a discrete-time Markov model for constrained streaming, where Y_i^c is the state of the model in the ith round. Y_i^c 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 ith round respectively. The evolution of N_i follows

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

where S_i is the number of packets transmitted successfully by TCP in the ith round, which is determined by X_i and X_{i+1}. 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 $\{Y_i^c\}_{i=1}^L$ and the time taken for each state transition can be found in [20].

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 [18]. We then compute the stationary distribution of N_i^l using (3) and (4). Finally, the fraction of late packets is computed from (2).

3.3.2 *Unconstrained streaming*

Unconstrained streaming can also be modeled as a producer-consumer problem. It differs from constrained streaming in that the number of packets in the buffer can be more than N_{max}. Therefore, it appears that solving unconstrained streaming is simpler than solving constrained streaming. This is not true for the following reason. In unconstrained streaming, the fraction of late packets depends heavily on the position of the round. This is because, under the assumption that the average TCP throughput is higher than the video bitrate, as the length of the video goes to infinity, the number of early packets approaches infinity and, hence, the fraction of late packets approaches 0. The fraction of late packets in the steady state (when the video is regarded as infinitely long) is thus trivial (equal to 0). To obtain the fraction of late packets over a finite video, we therefore resort to transient analysis, which is, in general, much more complex than steady state analysis.

We develop the following model for unconstrained streaming. Let $\{Y_i^u\}_{i=1}^L$ be a discrete-time Markov model for unconstrained streaming, where Y_i^u is the state of the model in the ith round. Here Y_i^u only contains the state of the TCP source in the ith round, that is, $Y_i^u = X_i$. The number of early packets in the ith 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 [19] for references on reward models). We associate an impulse reward of $\rho_{yy'}$ to a transition from state $Y_i^u = y$ to state $Y_{i+1}^u = y'$, 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 ith round as N_i'. The TANGRAM-II modeling tool [18] provides a functionality to solve for the transient distribution of N_i' based on the algorithm in [19].

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 ith round when the transmission and playback both start at time 0. Recall that N_i is the number of early packets in the ith 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 can be found in [20].

To compute the fraction of late packets, we first solve for the transient distribution of N_i using the TANGRAM-II modeling tool [18]. Next, the transient distribution of N_i^l is calculated using (3) and (4). Finally, the fraction of late packets is computed from (2).

For convenience, let P_i denote the probability that the ith round has at least one late packet. Then

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

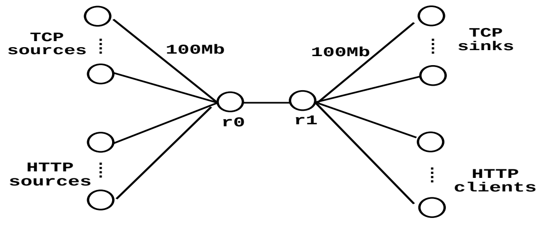


Figure 2: Validation setting in *ns*: packet losses are caused by buffer overflow on the link from router r_0 to r_1.

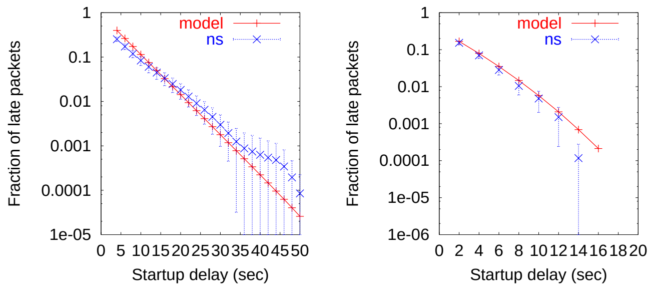
4. MODEL VALIDATION USING *NS* SIMULATIONS

In this section, we validate our models for constrained and unconstrained streaming using *ns* simulations. The topology is shown in Fig. 2. Multiple TCP and HTTP sources are connected to router r_0 and their corresponding sinks 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.

One of the TCP flows is used to stream video, and is referred to as the *video stream*. For this video stream, let D denote the round trip propagation delay; p denote the average loss rate; R denote the RTT and R_{TO} denote the value of the first retransmission timer. For simplicity, R_{TO} is rounded to be a multiple of R. 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 constrained and unconstrained streaming, we assume the video length to be 7000 and 80 seconds respectively. We vary the video length in Section 6.1. In particular, we show that the model for constrained streaming is accurate for a wide range of video lengths. We also show that, in unconstrained 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 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 found the values of R and T_O among different runs are 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 unconstrained 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 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 $\epsilon < 15\%$, 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 have a good match if f_m falls within the confident interval from the simulation or $\frac{1}{5} \le \frac{f_m}{f_s} \le 5$. The reason for the second "loose" criterion can be explained as follows. We use the fraction of late packets to roughly measure the viewing quality. When f_m and f_s satisfy the above criterion, they correspond to similar viewing experience. For instance, fraction of late packets as 0.1 and 0.5 both correspond to bad viewing experience; fraction of late packets as 10^{-4} and 5×10^{-4} both correspond to good viewing experience, etc.



(a) Constrained streaming. (b) Unconstrained streaming.

Figure 3: Model validation using *ns*.

4.1 Validation for constrained streaming

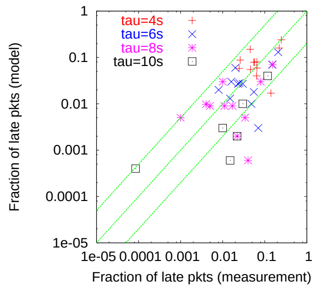
We validate the model for constrained streaming in four settings [20]. In the interest of space, we only describe the results for one setting. In this setting, 10 TCP sources and 40 HTTP sources are connected to router r_0. The video stream has a playback rate of 25 packets per second. The round trip propagation delay of this video stream, D, is 120 ms. We generate 60 simulation runs, each run lasting for 7000 seconds. The video length is 7000 seconds. The average loss rate of all the runs is 1.9%. We use $p = 1.9\%$ in the model and select runs with loss rates in the range of 1.7% to 2.1%. Among the selected 39 runs, the values of R and T_O are close with the average of 210 ms and 2 respectively. These values are used in the model to obtain the fraction of late packets. Fig. 3(a) depicts the fraction of late packets versus the startup delay predicted by the model and obtained from the simulation. As shown, we observe a good match between the model and the simulation.

4.2 Validation for unconstrained streaming

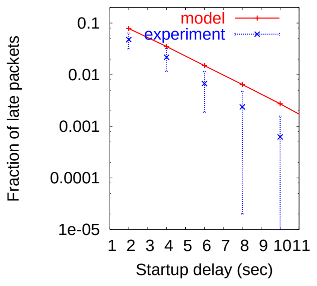
We validate the model for unconstrained streaming in four settings [20] and only describe one setting in detail. In this setting, 5 TCP sources and 30 HTTP sources are connected to router r_0. We generate 1000 simulation runs. Each run lasts for 200 seconds. We assume the length of the video to be 80 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 1.4%. We use $p = 1.4\%$ in the model and select the runs with loss rate between 1.2% to 1.6%. 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. Fig. 3(b) depicts the fraction of late packets versus startup delays. Both the results predicted by the model and measured from the simulation are shown in the figure. Again, we observe a good match between the model and the simulation.

5. MODEL VALIDATION USING EXPERIMENTS OVER THE INTERNET

In this section, we validate the models for constrained and unconstrained streaming using experiments conducted over the Internet. In each experiment, we stream a video using TCP from one site to another site and use *tcpdump* [21] to capture the packet timestamps. The average loss rate p, average RTT R and T_O of this TCP

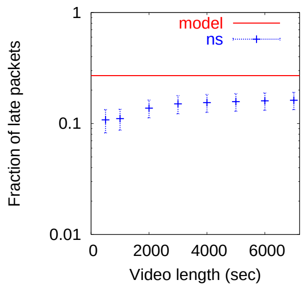


(a) Constrained streaming.

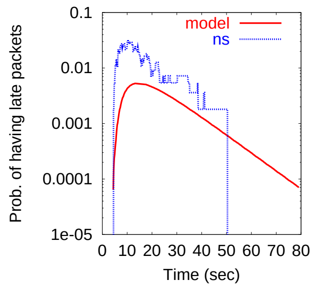


(b) Unconstrained streaming.

Figure 4: Model validation using experiments over the Internet.



(a) Constrained streaming.



(b) Unconstrained streaming.

Figure 5: Effect of video length on performance.

flow are estimated from the *tcpdump* traces. We use Linux-based machines for all the experiments.

5.1 Validation for constrained streaming

We first focus on constrained streaming. A CBR video is transmitted using TCP from University of Southern California (USC) to a client in a resident house in Amherst, Massachusetts. The playback rate of the video is 40 or 50 packets per second and each packet consists of 1448 bytes. That is, the bandwidth of the video is approximately 480 or 600 Kbps. We conducted 22 experiments from February 19 to March 7, 2003 at randomly chosen times; each experiment lasted for one hour. 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 [20]. 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 obtained a total of 12 segments from the experiments. The startup delay varies between 4 to 10 seconds. Fig. 4(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 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 7 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 7 bad matches are due to insufficient number of samples in the data segment.

5.2 Validation for unconstrained streaming

We next compare model prediction to measurements taken over the Internet for unconstrained 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 connection is too low to benefit from parallel TCP connections, we chose a high-bandwidth university path. 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 $p = 3.1\%$ in the model and select 266 segments having loss rate between 2.7% and 3.5%. 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. Fig. 4(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 constrained and unconstrained 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 0.4% to 4%. 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 [22]. 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 [11] and our measurements.

In the following, we first explore how the performance of constrained and unconstrained 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.

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 constrained streaming. The startup delay is set to 6 seconds and the length of the video ranges from 500 to 7000 seconds. Fig. 5(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 fraction of late packets for different video lengths from the simulation is similar and closer to the prediction from the model than for shorter videos. Throughout this section, we assume the video for constrained 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 the video length in unconstrained 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 ith round has at least one late packet (see Section 3.3.2). Fig. 5(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

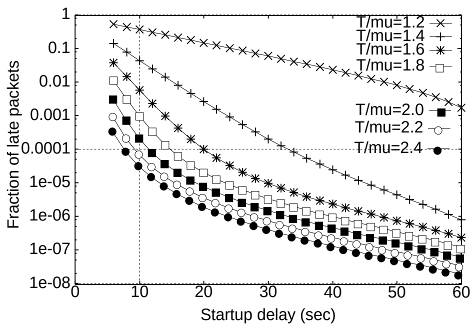


Figure 6: The fraction of late packets versus the startup delay for $p = 2\%$, $T_O = 4$ and $\mu = 25$ packets per second.

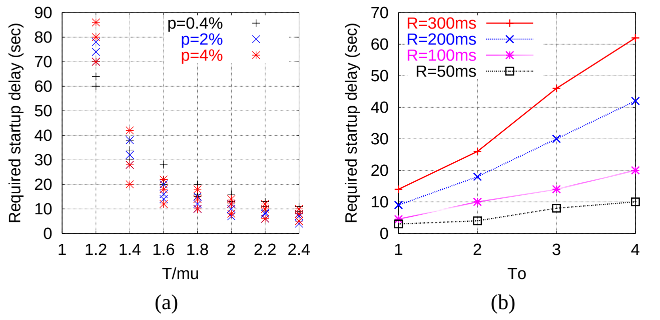


Figure 7: Constrained streaming: the required startup delay such that $f \leq 10^{-4}$ (a) when $\mu = 25$ packets per second, $T_O \geq 2$; (b) when $p = 4\%$ and $T/\mu = 2$.

video, increases to a peak value and then decreases over time. This can be explained as follows. At the beginning of the 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 Fig. 5(b), the probability of having 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 packet 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 $f = l f_l / L$. Throughout this section, we use videos of 80 seconds for unconstrained 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 drops when the packet loss rate exceeds 10^{-4} (e.g., [23]). Consequently, we assume that the performance of direct TCP streaming is satisfactory when the fraction of late packets is below 10^{-4} for a startup delay of around 10 seconds.

Denote the achievable TCP throughput as T packets per second. Then T/μ represents how much the achievable TCP throughput is higher than the video playback rate. The performance of TCP streaming improves as T/μ increases [20]. This is intuitive since packets are accumulated in the client's local buffer faster relative to the playback rate of the video as T/μ increases. We choose $p = 0.4\%, 2\%$ or 4%, corresponding to low, medium and high loss rates respectively, and choose $T_O = 1, 2, 3$ or 4. Let T_R denote the achievable TCP throughput in one RTT. Then T_R is determined by p and T_O, and $T = T_R/R$. Since T_R is fixed once p and T_O are fixed, the value of $T/\mu = T_R/(\mu R)$ is varied by varying the product of μ and R. We next explore quantitively the impact of T/μ on the performance of TCP streaming.

6.2.1 *Constrained streaming*

We first fix the playback rate of the video, μ, to be 25 packets per second and vary the value of RTT such that T/μ ranges from 1.2 to 2.4. We observe a diminishing gain by increasing T/μ on the performance: the performance improves dramatically as T/μ increases from 1.2 to 1.6 and less dramatically as T/μ increases from 1.6 to 2.4. One example is shown in Fig. 6, where $p = 2\%$ and $T_O = 4$. This diminishing gain indicates that, to achieve a low fraction of late packets, the required startup delay is very long 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. Fig. 7(a) shows the required startup delay such that the fraction of late packets, f, is below 10^{-4} as a function of T/μ for various loss rates and $T_O \geq 2$ (the required startup delay when $T_O = 1$ is much lower for the same loss rate and T/μ). We observe that under various settings, the performance becomes satisfactory when T/μ is roughly 2.

We next set the value of RTT to 50, 100, 200 or 300 ms and vary the playback rate of the video such that T/μ ranges from 1.2 to 2.4. We again observe a dramatic performance gain when T/μ increases from 1.2 to 1.6 and less dramatic gain afterwards. Next, we investigate the required startup delay such that the fraction of late packets is below 10^{-4} when $T/\mu = 2$. Fig. 7(b) shows the required startup delay when $p = 4\%$. 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 ($p = 4\%$) and high T_O values ($T_O \geq 3$). However, for a long RTT, high loss rate and timeout value, the required startup delay is in tens of seconds, as shown in Fig.7(b).

6.2.2 *Unconstrained streaming*

We again first fix the playback rate of the video to 25 packets per second and vary the value of RTT such that T/μ ranges from 1.2 to 2.4. The results are similar as in constrained streaming: diminishing performance gains are observed when increasing T/μ and the performance becomes satisfactory when the achievable TCP throughput is twice the video bitrate. Fig. 8(a) shows the required startup delay such that the fraction of late packets is below 10^{-4} as a function of T/μ for various loss rates and $T_O = 4$ (the required startup delay for lower T_O values is lower). We observe that the required startup delay is bounded within 10 seconds when T/μ increases to 2.

We next set the value of RTT to 50, 100, 200 or 300 ms and vary the playback rate of the video such that T/μ ranges from 1.2 to 2. We obtain the required startup delay such that the fraction of late packets is below 10^{-4} when $T/\mu = 2$. Fig. 8(b) shows

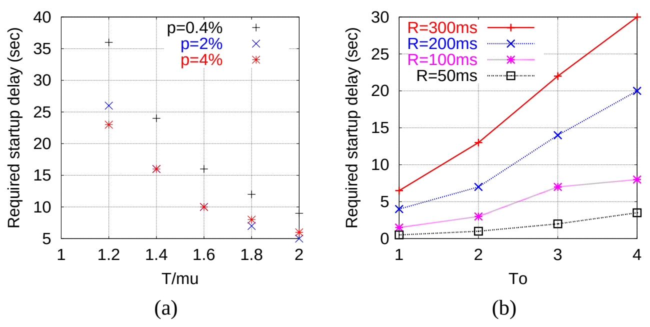


Figure 8: Unconstrained streaming: the required startup delay such that $f \leq 10^{-4}$ (a) when $\mu = 25$ packets per second, $T_O = 4$; (b) when $p = 4\%$ and $T/\mu = 2$.

the required startup when $p = 4\%$ for various values of RTT. The required startup delay for lower loss rates is lower (figures omitted). Under relatively short RTT, 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, as shown in Fig.8(b).

6.3 Summary of results

The key results from our exploration of parameter space are:

- The fraction of late packets when the video is beyond a certain length is similar in constrained streaming while it decreases with the video length in unconstrained streaming (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 paper, we developed discrete-time Markov models for constrained and unconstrained streaming that correspond to live and stored video streaming, respectively. 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 constrained and unconstrained 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 media bitrate, with only a few seconds of startup delay. Note that our model can be easily extended to the setting 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.

8. ACKNOWLEDGMENT

We thank E. de Souza e Silva and D. R. Figueiredo for many helps with numerically solving 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 and the anonymous reviewers for their insightful comments. This research was supported in part by the National Science Foundation under NSF grants ANI-9977635, ANI-9980552, CCR-9984030 and EEC-0313747 001.

9. REFERENCES

[1] R. Rejaie, M. Handley, and D. Estrin, “Quality adaptation for congestion controlled video playback over the Internet,” in *SIGCOMM*, pp. 189–200, September 1999.

[2] S. Floyd, M. Handley, J. Padhye, and J. Widmer, “Equation-based congestion control for unicast applications,” in *SIGCOMM 2000*, (Stockholm, Sweden), pp. 43–56, August 2000.

[3] C. Boutremans and J. Y. Le Boudec, “Adaptive joint playout buffer and FEC adjustment for Internet telephony,” in *Proceedings of IEEE INFOCOM‘2003*, (San-Francisco, CA), April 2003.

[4] J. van der Merwe, S. Sen, and C. Kalmanek, “Streaming video traffic: Characterization and network impact,” in *Proceedings of the Seventh International Web Content Caching and Distribution Workshop*, August 2002.

[5] N. Seelam, P. Sethi, and W. chi Feng, “A hysteresis based approach for quality, frame rate, and buffer management for video streaming using TCP,” in *Proc. of the Management of Multimedia Networks and Services 2001*, 2001.

[6] C. Krasic and J. Walpole, “Priority-progress streaming for quality-adaptive multimedia,” in *ACM Multimedia Doctoral Symposium 2001*, (Ottawa, Canada), October 2001.

[7] P. de Cuetos and K. W. Ross, “Adaptive rate control for streaming stored fine-grained scalable video,” in *Proc. of NOSSDAV*, May 2002.

[8] P. de Cuetos, P. Guillotel, K. W. Ross, and D. Thoreau, “Implementation of adaptive streaming of stored MPEG-4 FGS video over TCP,” in *International Conference on Multimedia and Expo (ICME02)*, August 2002.

[9] M. Li, M. Claypool, R. Kinicki, and J. Nichols, “Characteristics of streaming media stored on the Internet,” Tech. Rep. WPI-CS-TR-03-18, CS Department, Worcester Polytechnic Institute, May 2003.

[10] M. Mathis, J. Semke, and J. Mahdavi, “The macroscopic behavior of the TCP congestion avoidance algorithm,” *Computer Communications Review*, vol. 27, no. 3, 1997.

[11] J. Padhye, V. Firoiu, D. Towsley, and J. Krusoe, “Modeling TCP throughput: A simple model and its empirical validation,” in *Proc. ACM SIGCOMM*, (Vancouver, CA), pp. 303–314, 1998.

[12] J. Padhye, V. Firoiu, and D. Towsley, “A stochastic model of TCP Reno congestion avoidance and control,” Tech. Rep. 99-02, Department of Computer Science, University of Massachusetts, Amherst, 1999.

[13] E. Altman, K. Avrachenkov, and C. Barakat, “A stochastic model of TCP/IP with stationary random losses,” in *SIGCOMM*, pp. 231–242, 2000.

[14] D. R. Figueiredo, B. Liu, V. Misra, and D. Towsley, “On the autocorrelation structure of TCP traffic,” *Computer Networks Journal Special Issue on Advances in Modeling and Engineering of Long-Range Dependent Traffic*, 2002.

[15] N. Cardwell, S. Savage, and T. Anderson, “Modeling TCP latency,” in *INFOCOM (3)*, pp. 1742–1751, 2000.

[16] M. Mellia, I. Stoica, and H. Zhang, “TCP model for short lived flows,” *IEEE Communication Letters*, vol. 6, February 2002.

[17] S. Bohacek, “A stochastic model of TCP and fair video transmission,” in *Proc. IEEE INFOCOM*, 2003.

[18] E. de Souza e Silva and R. M. M. Leao, “The TANGRAM-II environment,” in *Proc. of the 11th Int. Conf. on modeling tools and techniques for computer and communication system performance evaluation (TOOLs 2000)*, May 2000.

[19] E. de Souza e Silva and H. R. Gail, “An algorithm to calculate transient distribution of cumulative rate and impulse based reward,” *Stochastic models*, vol. 14, no. 3, pp. 509–536, 1998.

[20] B. Wang, J. Kurose, P. Shenoy, and D. Towsley, “Multimedia streaming via TCP: An analytic performance study,” Tech. Rep. 04-21, Department of Computer Science, University of Massachusetts, Amherst, 2004.

[21] “tcpdump.” http://www.tcpdump.org/.

[22] B. Huffaker, M. Fomenkov, D. Moore, and K. Claffy, “Macroscopic analyses of the infrastructure: Measurement and visualization of Internet connectivity and performance,” in *A Workshop on passive and active measurements*, (Amsterdam), April 2001.

[23] O. Verscheure, P. Frossard, and M. Hamdi, “MPEG-2 video services over packet networks: Joint effect of encoding rate and data loss on user-oriented QoS,” in *Proc. of NOSSDAV*, July 1998.