

# Multipath Live Streaming via TCP: Performance and Benefits

Bing Wang\*, Wei Wei<sup>†</sup>, Zheng Guo\*, Don Towsley<sup>†</sup>

\*Computer Science & Engineering Department,  
University of Connecticut, Storrs, CT, 06269

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

## Abstract

Recently, a number of studies have demonstrated that multipath streaming is a promising technique to overcome the challenges of multimedia streaming over the best-effort Internet. These studies all focus on UDP-based streaming. Motivated by the wide use of TCP for streaming in practice and our earlier results on single-path TCP streaming, we study multipath live streaming via TCP in this paper. We design two simple schemes that use TCP for multipath live streaming and develop analytical models for each of them. We then validate the models using extensive *ns* simulation and Internet experiments. Using these models, we explore the performance of multipath live streaming via TCP over homogeneous and heterogeneous paths and demonstrate that multipath live TCP streaming generally leads to satisfactory performance when the aggregate achievable TCP throughput on the multiple paths is twice the media bitrate, with a few seconds of startup delay. Furthermore, multipath live streaming via TCP over two similar paths can support more than twice the media bitrate supported by single-path TCP streaming over one path.

## I. INTRODUCTION

High-quality multimedia streaming in the best-effort Internet need to overcome a number of challenges, including variable delays, bandwidth constraints and packet losses. Recently, a number of studies have demonstrated that multipath streaming (i.e., streaming over multiple network paths) is a promising technique to overcome these challenges [1], [2], [3], [4], [5], [6], [7], [8]. These studies all assume that UDP is the transport protocol for multimedia streaming. Indeed, TCP is conventionally regarded as inappropriate for multimedia streaming, since its backoff and retransmission mechanisms may lead to long delays which violate the realtime requirement of multimedia streaming.

In this paper, we study an approach defying the conventional wisdom that relies on TCP for multipath streaming. This is motivated by the wide use of TCP for streaming in practice and our earlier results on single-path TCP streaming. TCP streaming is widely supported in commercial streaming products, including Real Media and Windows Media, the two dominant streaming media products. Furthermore, recent measurement studies have shown that a significant fraction (around or above 50%) of commercial streaming traffic uses HTTP/TCP [9], [10], [11]. In our earlier work [12], we analytically study the performance of single-path TCP streaming and found that its performance is generally satisfactory when the achievable TCP throughput is roughly twice the media bitrate, with only a few seconds of startup delay. This result partly explains why TCP streaming has been an attractive option in practice: such a bandwidth requirement can be satisfied even for broadband home users (using cable modem or ADSL technologies) for a large fraction of streaming multimedia clips in the Internet today.

In the scenarios where a single network path does not provide sufficient amount of bandwidth for TCP streaming, a natural solution is to use multiple paths between a server and client. We focus on live streaming in this paper. In particular, we seek to answer the following questions: (1) Under what circumstances can multipath live streaming via TCP provide satisfactory performance? (2) What are the benefits of multipath TCP live streaming over single-path TCP live streaming? (3) What are the impacts of path diversity on multipath TCP live streaming? To answer these questions, we design two simple schemes

for multipath TCP streaming and develop corresponding analytical models for them. The models allow us to systematically investigate the performance of our schemes under various conditions, a task that is difficult when using empirical measurements or simulation alone.

Our paper makes the following main contributions:

- We design two schemes for multipath live streaming via TCP, named *static mpath-streaming* and *dynamic mpath-streaming* respectively. Static mpath-streaming constitutes an interesting baseline. Dynamic mpath-streaming utilizes the bandwidth-probing mechanisms embedded in TCP to stripe packets over multiple paths. It is easy to implement and generally outperforms static mpath-streaming.
- We develop analytic models for static and dynamic mpath-streaming. These models are validated using extensive *ns* simulation and Internet experiments.
- Using these models, we explore the performance of multipath TCP streaming under homogeneous and heterogeneous paths. Our results demonstrate that multipath TCP streaming generally leads to satisfactory performance when the *aggregate* achievable TCP throughput on the multiple paths is twice the media bitrate, with a few seconds of startup delay. Furthermore, multipath live streaming via TCP over two similar paths can support more than twice the media bitrate supported by single-path TCP streaming over one path.

The rest of the paper is organized as follows. Section I-A summarizes related work. Section II describes problem setting. Sections III and IV present static and dynamic mpath-streaming schemes and corresponding models for them. Validation of the models using *ns* simulations and Internet experiments is described in Sections V. A performance study based on the models is presented in Section VI. Finally, Section VII concludes the paper and presents future work.

#### A. Related work

Multipath continuous media streaming is studied in [1], [2], [3], [4], [5], [6], [7], [8], [13]. In particular, [1], [2], [8] demonstrate the benefits of using multiple paths for continuous media streaming. In [3], [4], [5], [6], coding techniques (Multiple Description Coding) are applied to the video streams to improve loss recovery. The study [7] determines the sending rate and the distribution of packets on the multiple paths to minimize loss rate at the receiver. The work [13] proposes a heuristic-based algorithm for multipath video streaming that provides close to optimal performance. All the above studies use UDP as the transport protocol. We study multipath streaming via TCP, which is fundamentally different from UDP-based streaming. To the best of our knowledge, our work is the first study of multipath streaming via TCP.

Using TCP for multimedia streaming eliminates the need for error-recovery at the application-level and automatically provides TCP-friendliness and the capability to pass firewalls. A number of research efforts on TCP-based streaming focus on how to adapt the video bitrate to deal with network bandwidth fluctuations [14], [15], [16], [17]. Our earlier work [12] analytically studies the performance of multimedia streaming using TCP. A recent study [18] analytically determines the proper receiver buffer size to ensure a prescribed video quality for TCP streaming. In this work, we use multiple TCP flows on multiple paths instead of one TCP flow on a single path as in existing studies.

The literature on TCP modeling is extensive. Much of the TCP modeling focuses on TCP performance for file transfers, assuming long-lived flows [19], [20], [21], [22], [23] or short-lived flows [24], [25]. The study of [26] 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. In our earlier study [12], we develop an analytical model for single-path TCP streaming. In this paper, we develop analytical models for multipath TCP streaming.

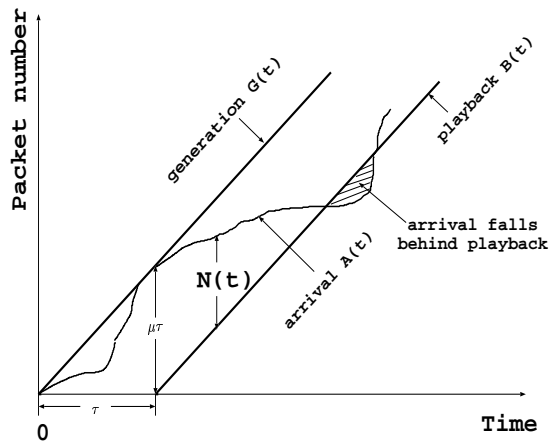


Fig. 1. Live streaming.

## II. PROBLEM SETTING

Suppose that a client is connected to a server using  $K$  paths ( $K \geq 2$ ), indexed 1 to  $K$ . These paths are formed using a multipath architecture, e.g., multihoming of the end hosts (i.e., the server or the client). We assume that these multiple paths do not share performance bottlenecks. The client requests a video from the server. Corresponding to the request, the server stripes the video via TCP over the multiple paths to the client. The client allows a startup delay on the order of seconds, 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 is lost 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) [27]. We therefore consider a CBR video with a playback rate of  $\mu$  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*. A late packet typically leads to a glitch during playback. Therefore, we use the *fraction of late packets*, i.e., 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.

In this paper, we focus on live streaming. That is, the server generates video content in real time and is only able to transmit content that has already been generated, as illustrated in Fig. 1. For ease of exposition, we assign a sequence number to each packet. The first packet is given a sequence number of 1 and subsequent packets have sequence numbers incremented by 1. 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.

We next describe the characteristics of live streaming. In Fig. 1,  $G(t)$  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  $t$ . It is the total number of packets that have arrived over all the paths. Note that although packets on the same path are in order, packets arriving at the client may be out of order. Since only packets generated can be transmitted, we have  $A(t) \leq G(t)$ . Let  $B(t)$  denote the number of packets played out by the client by time  $t$ . Suppose that video playback commences at time  $\tau$ . That is, the startup delay is  $\tau$  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  $t$ . This is an important characteristic of live streaming to be used in the models for multipath live streaming in Section IV.

Another characteristic of multipath live streaming is that the average aggregate TCP throughput over the multiple paths does not exceed the video bitrate due to the constraint that only packets generated by the server can be transmitted. We use *achievable TCP throughput* on a path to denote the throughput achieved by an infinite TCP source, i.e., from a server that has infinitely backlogged data to transmit. In live streaming, the TCP throughput on a path differs from the achievable TCP throughput on that path: the achievable TCP throughput can be higher than the TCP throughput.

### III. SCHEMES FOR MULTIPATH TCP LIVE STREAMING

A key decision in multipath TCP live streaming is to determine what packets to send over which path. Before presenting our schemes, we use an example to illustrate that packets must be split carefully among the multiple paths. Let us assume there are two paths between the server and the client, with achievable TCP throughputs 0.5 and 1.5 times of the video bitrate respectively. One scheme for multipath TCP streaming is to send odd packets on the first path and even packets on the other. This scheme, however, will lead to a large fraction of late packets, as indicated by the results in [12]. Intuitively, this is because, on the first path, the achievable TCP throughput equals to the assigned sending rate. Hence, packets are not accumulated into the client's local buffer fast enough to tolerate throughput fluctuations in TCP. The above example implies that a path with higher achievable TCP throughput should be allocated a higher fraction of the video packets. This motivates us to develop two schemes for multipath live streaming via TCP.

Our first scheme assumes that the achievable TCP throughputs on the multiple paths are stable throughout the video transmission and are known beforehand. Let  $T_k$  denote the average achievable TCP throughput on the  $k$ -th path,  $k = 1, \dots, K$ . Then the server sends a fraction,  $\alpha_k = T_k / \sum_{k=1}^K T_k$ , of the video packets on the  $k$ -th path. That is, the number of video packets to be sent on a path is determined in a static manner and is proportional to the achievable TCP throughput on that path. We refer to this scheme as *static mpath-streaming*. Static mpath-streaming is not a practical scheme since it requires an accurate estimate of the achievable TCP throughput beforehand. Furthermore, since packets are allocated in a static manner over the multiple paths, network bandwidths are not used efficiently: the available bandwidth on one path cannot be used to send backlogged packets assigned to another path. However, according to our earlier results [12], static multi-path streaming generally achieves satisfactory streaming performance when the aggregate TCP throughput on the multiple paths is roughly twice the video bitrate, with a few seconds of startup delay. This can be explained as follows. Static mpath-streaming over  $K$  paths can be treated as  $K$  single-path TCP streaming processes since the packets allocation is beforehand and static. Then it is straightforward to show that, when the aggregate achievable TCP throughput on the multiple paths is twice the video bitrate, under static mpath-streaming, the achievable TCP throughput on each path is at least twice the allocated sending rate on that path, which satisfies the condition for satisfactory streaming performance via TCP [12]. We therefore use static mpath-streaming as a baseline scheme for multipath TCP streaming.

Our second scheme does not require explicit knowledge of the achievable TCP throughput on each path. Instead, it utilizes the bandwidth-probing mechanism (i.e., the additive-increase and multiplicative-decrease mechanism) embedded in TCP and works as follows. The server places data generated into a source queue. At time  $t$ , if the TCP sender on a path can send

data, it obtains one packet from the source queue when there is at least one packet inside the queue. We refer to this scheme as *dynamic mpath-streaming* since packets are allocated in a dynamic manner over the multiple paths. Note that, when using dynamic mpath-streaming, a TCP session on a path with higher achievable throughput sends a larger fraction of the packets. Dynamic mpath-streaming is easy to implement. Furthermore, we expect the performance of dynamic mpath-streaming to be better than that of static mpath-streaming. This is because, in dynamic streaming, a TCP source can send a packet whenever it has available bandwidth to send the packet (given that there is packet to send) and hence utilizes the bandwidths on the multiple paths more efficiently.

#### IV. MODELS FOR MULTIPATH TCP LIVE STREAMING

We develop discrete-time Markov models for both static and dynamic mpath-streaming. Our models assume a single TCP flow on each path from the server to the client (e.g., multipaths formed by multihoming at the server or the client). Before presenting our models, we first briefly describe the Markov model for TCP in [21], [23], which is used in our mpath-streaming models.

##### A. Model for TCP

In [21], [23], the behavior of TCP is described by a discrete-time Markov model. The state of a TCP source changes in “rounds”. 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  $X_i^k$  denote the state of the TCP flow on the  $k$ -th path after the  $i$ -th transition,  $i = 0, 1, \dots, k = 1, \dots, K$ . Then  $X_i^k$  is represented as a tuple,  $X_i^k = (W_i^k, C_i^k, L_i^k, E_i^k, R_i^k)$ , where  $W_i^k$  is the window size;  $C_i^k$  models the delayed ACK behavior of TCP ( $C_i^k = 0$  and  $C_i^k = 1$  indicate the first and the second of the two rounds respectively);  $L_i^k$  is the number of packets lost in the previous round;  $E_i^k$  denotes whether the connection is in a timeout state and the value of the back-off exponent;  $R_i^k$  indicates if a packet being sent in the timeout phase is a retransmission ( $R_i^k = 1$ ) or a new packet ( $R_i^k = 0$ ). Let  $S_i^k$  denote the number of packets transmitted successfully by the  $k$ -th TCP at the  $i$ -th transition.  $S_i^k$  is determined by how the congestion window changes at the  $i$ -th transition as detailed in [21], [23].

##### B. Model for static mpath-streaming

Static mpath-streaming can be treated as multiple independent single-path TCP streaming processes since packets are allocated in a static manner on each path. Each streaming process contains a server, a TCP connection and a client with a buffer. On the  $k$ -th path, the server generates packets at the rate of  $\alpha_k \mu$  packets per second (since a fraction of  $\alpha_k$  is sent on the  $k$ -th path), the client plays back at the same rate starting from time  $\tau$ , and packets arriving earlier than their playback time are stored in the client’s local buffer. Let  $N_{max}^k$  denote the maximum number of early packets for the stream sent on the  $k$ -th path. Then  $N_{max}^k = \alpha_k \mu \tau$  following the observation in Section II. Our model for static mpath-streaming consists of  $K$  independent Markov models, each corresponding to a streaming process on a path, with the length of a round (i.e., RTT) of that path as the time unit. More specifically, the model for the  $k$ -th path is represented as a tuple  $(X_i^k, N_i^k)$ , where  $X_i^k$  is the state of the TCP flow (see Section subsec:model-tcp) and  $N_i^k$  is the number of early packets after the  $i$ -th round,  $N_i^k \leq N_{max}^k$ ,

$k = 1, \dots, K$ . To satisfy the condition that  $N_i^k \leq N_{max}^k$ , the TCP flow on the  $k$ -th path does not send any packet in the  $(i + 1)$ -th if  $N_i^k = N_{max}^k$ . The evolution of  $N_i^k$  follows

$$N_{i+1}^k = \min(N_{max}^k, N_i^k + S_i^k - \alpha_k \mu R_k)$$

where  $R_k$  is the RTT on the  $k$ -th path and  $S_i^k$  is the number of packets transmitted by the TCP on the  $k$ -th in the  $i$ -th round. A detailed description of the state transition probabilities for the Markov model on each path and the time taken for each state transition is in Appendix II.

Let  $f_k$  denote the fraction of late packets on the  $k$ -th path,  $k = 1, \dots, K$ . It is obtained using the technique for single-path TCP streaming in [12]. Then the fraction of late packets under static mpath-streaming is

$$f = \sum_{k=1}^K \alpha_k f_k \quad (1)$$

### C. Model for dynamic mpath-streaming

We develop a model for dynamic mpath-streaming by considering the data production and consumption at the client-side buffer: the multiple TCP connections from the server to the client produce (transmit) packets and store them in the client-side buffer; the client starts to consume (i.e., play back) packets in the buffer from time  $\tau$  at a constant rate of  $\mu$  packets per second. At any time, the number of early packets in the buffer never exceeds  $N_{max}$ ,  $N_{max} = \mu\tau$ , which follows from an earlier observation made in Section II. To satisfy this constraint, a producer stops transmitting packets when there are  $N_{max}$  packets in the buffer. We therefore develop the following model for dynamic mpath-streaming.

Let the tuple  $(X_i^1, \dots, X_i^K, N_i)$  represent the finite-state Markov chain for dynamic mpath-streaming, where  $X_i^k$  is the state of the TCP connection on the  $k$ -th path and  $N_i$  is the number of early packets in the client's local buffer after the  $i$ -th transition,  $k = 1, \dots, K$ . The transition of the Markov chain occurs in the presence of two types of events: (1) when a TCP flow makes a transition, (2) when a packet is consumed (played back) from the client's local buffer. Let  $E_i = C$  denote the event that a packet consumption triggers transition  $i$ . Similarly, let  $E_i = P_k$  denote the event that a transition of the TCP flow on the  $k$ -th path triggers transition  $i$ . Then the evolution of  $N_i$  follows

$$N_{i+1} = \begin{cases} N_i - 1, & E_i = C \\ \min(N_{max}, N_i + S_i^k), & E_i = P_k \end{cases} \quad (2)$$

Recall that  $S_i^k$  is the number of packets transmitted by the TCP flow on the  $k$ -th path at the  $i$ -th transition. In order to satisfy the condition that  $N_i \leq N_{max}$ , a TCP source does not make a transition if  $N_i = N_{max}$ . A detailed description of the state transition probabilities for the Markov model and the time taken for each state transition can be found in [28].

For sufficiently long videos, the fraction of late packets can be approximated by the steady state probability of having no early packets in the client's local buffer while playing back a packet. That is,

$$f = P(N_i < 0 \mid E_i = C) \quad (3)$$

We numerically solve for the stationary distribution of  $N_i$  using TANGRAM-II modeling tool [29] to obtain the fraction of late packets .

Note that the above model for dynamic mpath-streaming is not exact since it does not account for out-of-order packets. Suppose packet  $i$  is lost by a TCP flow and packet  $j$  ( $j > i$ ) is sent successfully by another TCP flow. This will lead to

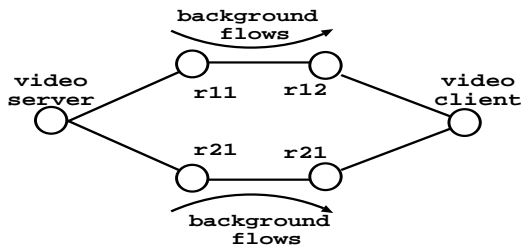


Fig. 2. Validation setting in *ns*: the video server spreads the video over two paths to the client. Packet losses are caused by buffer overflows on the links from router  $r_{11}$  to  $r_{12}$  and router  $r_{21}$  to  $r_{22}$ .

Config.	TCP	HTTP	Prop. Delay (ms)	B.w. (Mbps)	Buffer (pkts)
1	9	40	40	3.7	50
2	9	40	1	3.7	50
3	19	40	40	5.0	50

TABLE I  
CONFIGURATIONS OF THE BOTTLENECK LINK ON A PATH.

out-of-order packets at the receiver, while in our model, packets are consumed as if they are in order. However, since the round trip time of a TCP flow is typically much smaller than the startup delay, the out-of-order packets can be accommodated by the startup delay. We therefore expect our model to provide an accurate approximation for relatively short round-trip times and low loss rates. This is confirmed by our validation through simulation and Internet experiments (see Section V).

## V. MODEL VALIDATION

In this section, we first validate our models for static and dynamic mpath-streaming using *ns* simulations. We then validate the model for dynamic mpath-streaming using experiments over the Internet (static mpath-streaming is not a practical scheme). Two paths are used between the server and client for all the validations.

### A. Model validation using *ns*

The topology used in *ns* is shown in Fig. 2. A video server streams a video to a client via TCP over two paths. On path  $i$ , the link  $(r_{i1}, r_{i2})$  is the bottleneck link, where packets are lost due to buffer overflow,  $i = 1, 2$ . A bottleneck link is also traversed by multiple TCP and HTTP flows (referred to as background flows).

We simulate three different configurations for the bottleneck link on a path, listed in Table I, to create different settings. For each setting, we make 30 runs to obtain 95% confidence intervals. 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

Setting	Dynamic mpath-streaming							Static mpath-streaming						
	$p_1$	$p_2$	$R_1$ (ms)	$R_2$ (ms)	$T_O^1$	$T_O^2$	$\mu$ (pkts ps)	$p_1$	$p_2$	$R_1$ (ms)	$R_2$ (ms)	$T_O^1$	$T_O^2$	$\mu_1 (= \mu_2)$ (pkts ps)
1	0.023	0.023	210	210	1.6	1.6	50	0.019	0.018	210	210	1.6	1.6	25
2	0.037	0.036	150	150	1.7	1.7	50	0.027	0.025	150	150	1.7	1.7	25
3	0.053	0.053	200	200	1.9	1.9	30	0.044	0.045	200	200	1.9	1.9	15

TABLE II  
MODEL VALIDATION IN *ns*: HOMOGENEOUS PATHS.

Setting	Dynamic mpath-streaming							Static mpath-streaming							
	$p_1$	$p_2$	$R_1$ (ms)	$R_2$ (ms)	$T_O^1$	$T_O^2$	$\mu$ (pkts ps)	$p_1$	$p_2$	$R_1$ (ms)	$R_2$ (ms)	$T_O^1$	$T_O^2$	$\mu_1$ (pkts ps)	$\mu_2$ (pkts ps)
1-2	0.023	0.036	210	150	1.6	1.7	50	0.019	0.025	210	150	1.6	1.7	25	25
1-3	0.023	0.053	210	200	1.6	1.9	40	0.019	0.052	210	200	1.6	1.9	26	14
2-3	0.036	0.053	150	200	1.7	1.9	40	0.025	0.052	150	200	1.7	1.9	26	14

TABLE III  
MODEL VALIDATION IN *ns*: HETEROGENEOUS PATHS.

if one of the following two conditions is satisfied:  $f_m$  falls within the confidence interval obtained from the simulation, or  $0.1 \leq f_m/f_s \leq 10$ . 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).

Our validation is in two settings: when the configurations of the two paths are the same and when they differ, referred to as *homogeneous paths* and *heterogeneous paths* respectively. For the video packets streamed on the  $i$ -th path via TCP, let  $p_i$  denote the average loss rate;  $R_i$  the round trip time and  $R_{TO}^i$  the value of the first retransmission timer of the TCP. We further define  $T_O^i = R_{TO}^i/R_i$ . For static mpath-streaming, let  $\mu_i$  denote the sending rate on the  $i$ -th path,  $\mu_1 + \mu_2 = \mu$ . In all cases, the video streams uses TCP Reno. We are interested in the steady-state behavior of multipath streaming and set the video length to 10000 seconds. We now describe the validation results.

1) *Homogeneous paths*: We consider 3 different settings with homogeneous paths, one for each configuration of the bottleneck links listed in Table I. The parameters of the video stream using dynamic and static mpath-streaming are listed in Table II. They are averaged over 30 simulation runs. As expected, the various parameters on the two paths are similar. The playback rate of the video is 50 or 30 packets per second and each packet is 1500 bytes. Therefore, the bandwidth of the video is 600 or 360 Kbps. In static mpath-streaming, the video is split evenly across the two paths. We observe that, in the same setting, the RTTs and first timeout values are similar under dynamic and static mpath-streaming while the loss rates under dynamic mpath-streaming are higher than those under static mpath-streaming. This might be due to more aggressive transmission in dynamic mpath-streaming. However, we expect the difference in loss rates between static and dynamic mpath-streaming to be less dramatic when the links are shared by more flows as is usual in practice. 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, as shown in Fig. 3; the results for other settings being similar [28]. From Fig. 3, we observe a good match between the model and the simulation for both static and dynamic mpath-streaming. Furthermore, as expected, the performance of static mpath-streaming is inferior to that of dynamic streaming.

2) *Heterogeneous paths*: We consider 3 different settings with heterogeneous paths by pairing two different configurations for the bottleneck links listed in Table I. The parameters of the video stream using dynamic and static mpath-streaming are listed in Table III. The playback rate of the video is either 50 or 40 packets per second. We next describe the validation for Setting 1-3 (i.e., using configuration 1 and 3 in Table I for the bottlenecks of the two paths); the results for the other settings being similar [28]. Fig. 4 depicts the fraction of late packets from the model and simulation for Setting 1-3. We observe similar results as in homogeneous paths: the model and the simulation results match well; dynamic mpath-streaming outperforms static mpath-streaming.

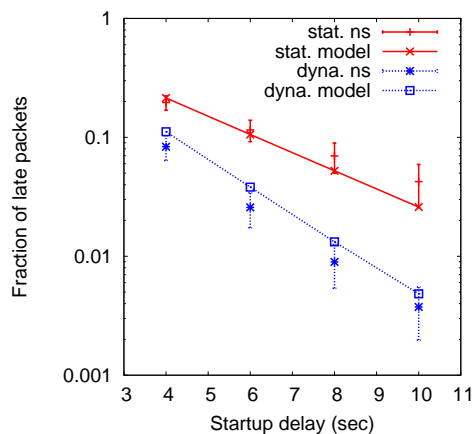


Fig. 3. Homogeneous paths: validation results for Setting 2.

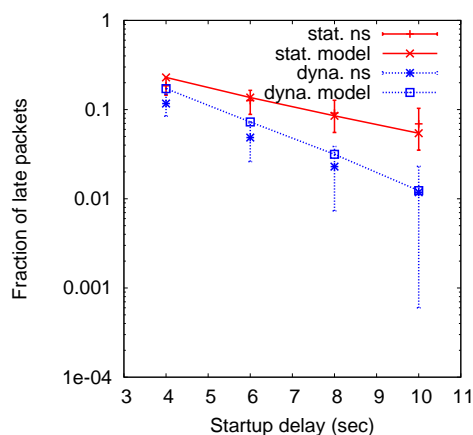


Fig. 4. Heterogeneous paths: validation results for Setting 1-3.

### B. Model Validation Using Experiments over the Internet

We now validate the model for dynamic mpath-streaming through experiments conducted over the Internet. In each experiment, we use *tcpdump* [30] to capture the packet timestamps on each path. The average loss rate, average RTT and timeout value of each TCP flow are estimated from the *tcpdump* traces. We use Linux-based machines for all experiments.

With no access to multihomed machines, we emulate multipath streaming by streaming from a server to two clients and then combining the traces at the two clients. We place the video server inside University of Connecticut and use nodes in planetlab [31] as clients. Multipath streaming via two homogeneous paths is emulated by streaming from the server to two nodes that are connected through ADSL in San Francisco, California. Multipath streaming via heterogeneous paths is emulated by streaming from the server to one node in San Francisco, California and another in Hefei, China. The playback rate of the video is 25 or 50 packets per second (for the homogeneous paths) and 100 packets per second (for the heterogeneous paths). Each packet consists of 1448 bytes. Therefore, the bandwidth of the video varies from 300 Kbps to 1.2 Mbps. We conducted 10 experiments from July 21 to July 27, 2006 at randomly chosen times; each experiment lasted for 3000 seconds. Fig. 5 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

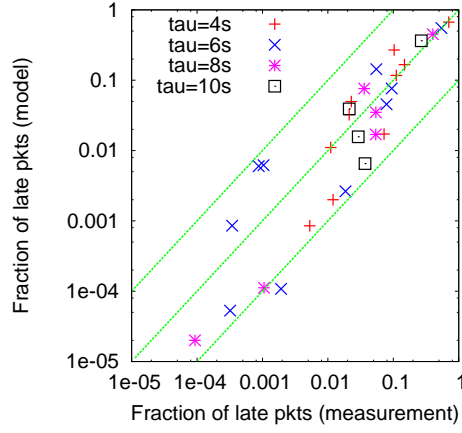


Fig. 5. Validation for dynamic mpath-streaming using experiments over the Internet.

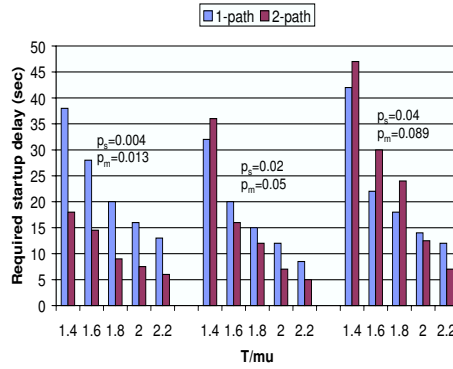


Fig. 6. Performance comparison of single-path and multipath TCP streaming under Condition 1,  $T_m^1 = T_m^2 = T_s/2$ ,  $\mu_m = \mu_s = 25$  packets per second,  $T_O = 4$ .

respectively 10 times higher and lower than that from the measurements. All but one scatterplot points fall within the upper and lower 45 degree lines, indicating a match between the model and the Internet experiments. For startup delay of 10 seconds, the fraction of late packets is 0 for 6 experiments (therefore are not in the plot). In these 6 experiments, the model predictions are also 0 for 5 of them and  $10^{-4}$  for one. We speculate that the discrepancies is due to an insufficient number of samples in the data trace.

## VI. PERFORMANCE AND BENEFITS OF MULTIPATH LIVE STREAMING VIA TCP

In this section, we compare the performances of multipath and single-path TCP live streaming. In doing so, we identify the benefits of using multiple paths over a single path and provide guidelines as to when multipath TCP live streaming leads to satisfactory performance. 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. This is because people can usually tolerate several seconds of startup delay and studies show that video quality drops when the packet loss rate exceeds  $10^{-4}$  (e.g., [32]). Unless otherwise specified, multipath TCP streaming in the following refers to dynamic mpath-streaming (since it is more practical and performs better than static mpath-streaming). The results for single-path TCP live streaming are derived from the model in [12].

We first compare the performances of multipath and single-path TCP live streaming, assuming homogeneous paths in

multipath streaming for simplicity. We then explore the impact of path diversity on multipath TCP live streaming. We use two paths (i.e.,  $K = 2$ ) for multipath TCP streaming throughout this section.

#### A. Multipath versus single-path live streaming via TCP

For a fair comparison, we compare multipath and single-path TCP live streaming in the following manner. For single-path streaming, let  $T_s$  denote the achievable TCP throughput on this single path, which is the TCP throughput for an infinite source (see Section II). For multipath streaming, let  $T_m^k$  denote the achievable TCP throughput on the  $k$ -th path,  $k = 1, 2$ . Let  $T_m$  denote the aggregate achievable TCP throughput over the two paths,  $T_m = T_m^1 + T_m^2$ . Let  $\mu_m$  and  $\mu_s$  denote the playback rate of multipath and single-path TCP streaming respectively. Then we compare the performances of multipath and single-path TCP streaming under the same ratio of the (aggregate) achievable TCP throughput over the video playback rate, i.e., when  $T_m/\mu_m = T_s/\mu_s$ . We set this requirement for the following reason. The achievable TCP throughput on a path reflects the available bandwidth of that path. The performance of TCP streaming improves as  $T_m/\mu_m$  or  $T_s/\mu_s$  increases (for a higher ratio, packets accumulate in the client's local buffer faster relative to the video playback rate). For ease of exposition, we refer to this ratio simply as  $T/\mu$ .

While comparing multipath and single-path streaming, we provide answers to the following two important questions:

- If a video bitrate can be satisfied by a single path, can two paths, each with half of the achievable TCP throughput of the single path, support the same video bitrate? When the access link is the bottleneck of a path, this question transforms to: can a high bandwidth access link be replaced by two lower bandwidth links while maintaining the same streaming performance?
- If a video bitrate is satisfied by a single path, can two such paths support videos with twice the bitrate? When the access link is the bottleneck of a path, these questions transform to: if a user subscribe to two access networks of similar bandwidth (e.g., ADSLs from two different providers), can he view videos with bitrates twice as those that can be supported by a single access network?

The first question corresponds to the condition that the same video is played in multipath and single-path streaming, and the two paths in multipath streaming each has achievable TCP throughput half that in single-path streaming. That is,  $T_m^1 = T_m^2 = T_s/2, \mu_m = \mu_s$ . The second question corresponds to the condition that the two paths in multipath streaming have the same achievable TCP throughput as that in single-path streaming while multipath streaming transmits a video of twice bitrate as that in single-path streaming. That is,  $T_m^1 = T_m^2 = T_s, \mu_m = 2\mu_s$ . Both conditions satisfy  $T_m/\mu_m = T_s/\mu_s$ . We next compare the performances of multipath and single-path streaming under these two conditions.

1) *Condition 1* ( $T_m^1 = T_m^2 = T_s/2, \mu_m = \mu_s$ ): For a TCP flow, its achievable TCP throughput is determined by three parameters: loss rate, RTT and timeout value (the ratio of first timeout value over RTT). When comparing multipath and single-path streaming, we first fix the parameters of the single-path streaming and then obtain the parameters of multipath streaming correspondingly. Since the two paths in multipath streaming each has achievable TCP throughput half that in single-path streaming, we consider two ways to obtain the parameters of a path in multipath streaming: using a longer RTT (assuming the path has a longer queuing delay and processing time) or higher loss rate (assuming the path is more lossy). We only describe the results when each path in multipath streaming has a higher loss; the results when using a longer RTT are similar.

For single-path streaming, we vary the loss rate in the range from 0.004 to 0.04. The timeout value,  $T_O$  is varied from 1 to 4, based on several measurements in [20] and our measurements. The RTT is in the range of 40 ms to 300 ms based on measurement results that the median RTTs between two sites on the same coast and the two coasts in the US are 50 and 100

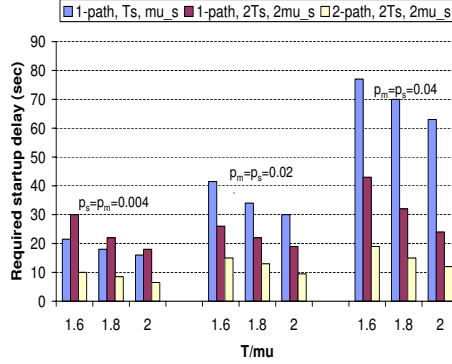


Fig. 7. Performance comparison of single-path and multipath TCP streaming under Condition 2 ( $T_m^1 = T_m^2 = T_s, \mu_m = 2\mu_s$ ),  $T_O = 4$ , RTT is 300 ms.

ms respectively [33]. In the following, we vary the value of  $T/\mu$  from 1.4 to 2.2 to identify the minimum value of  $T/\mu$  that leads to satisfactory performance under multipath streaming. Since the achievable TCP throughput in a round (i.e., a RTT) is determined by the loss rate and timeout value, we vary  $T/\mu$  in single-path streaming by fixing the loss rate and timeout value, and varying either the RTT or video playback rate,  $\mu$ . Let  $p_s$  and  $p_m$  denote respectively the loss rate under single-path and multipath streaming. Then  $p_m$  is determined correspondingly from  $p_s$ , as to be described later.

We first fix the video playback rate and vary the RTT so that  $T/\mu$  varies from 1.4 to 2.2. Fig. 6 plots the required startup delay such that the fraction of late packets lies below  $10^{-4}$  under single-path and multipath streaming,  $\mu_s = \mu_m = 25$  packets per second,  $T_O = 4$  (required startup delay for lower values of  $T_O$  are lower). In single-path streaming, the loss rate,  $p_s$ , is 0.004, 0.02 and 0.04, representing low, medium and high loss rates. Correspondingly, the loss rate,  $p_m$ , in multipath streaming is 0.013, 0.05, and 0.089, obtained from the achievable TCP throughput formula in [20], so that the achievable TCP throughput on each path in multipath streaming is half that in single-path streaming. We observe that the required startup delays in multipath streaming are significantly lower than those in single-path streaming when the loss rate is low. For medium loss rate, multipath streaming still requires lower startup delay in most cases. Under high loss rate, multipath streaming requires lower startup delays when  $T/\mu \geq 2$ . Furthermore, we observe that, for all the cases, multipath TCP live streaming provides satisfactory performance when the aggregate achievable TCP throughput is twice the video bitrate.

We next fix the RTT and vary the video playback rate so that  $T/\mu$  varies from 1.4 to 2.2. The RTT in single-path streaming is set to 100 ms, 200 ms or 300 ms. We observe a lower required startup delay under multipath streaming for most cases. We also observe that, for a large RTT, high loss rate and timeout value (RTT is at least 200 ms,  $p_m \geq 0.05$  and  $T_O = 4$ ), multipath streaming requires tens of seconds of startup delay for a fraction of late packets below  $10^{-4}$ .

In summary, the required startup delay such that the fraction of late packets lies below  $10^{-4}$  in multipath streaming is general smaller than that in single-path streaming under Condition 1 ( $T_m^1 = T_m^2 = T_s/2, \mu_m = \mu_s$ ). We therefore conclude that a single path can be replaced by two paths, each with half of the achievable TCP throughput of the single path, to support the same video bitrate.

2) *Condition 2* ( $T_m^1 = T_m^2 = T_s, \mu_m = 2\mu_s$ ): Under this condition, each path in multipath streaming is the same as that in single-path streaming while the playback rate is doubled. In this setting, the performance of static mpath-streaming is the same as that of single-path streaming (in static mpath-streaming, the fraction of late packets obtained on the two paths are the same, equal to that in single-path streaming). Since dynamic mpath-streaming outperforms static mpath-streaming, we expect

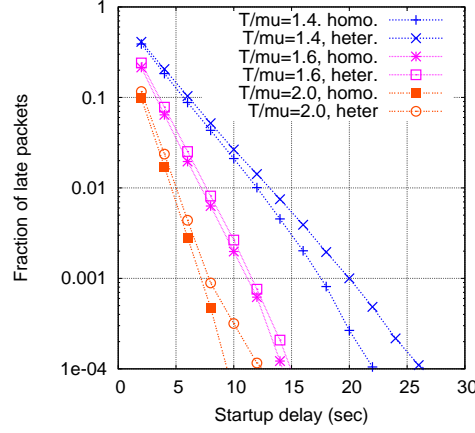


Fig. 8. Impact of path diversity: two paths have different loss rates,  $R_m = 300$  ms,  $p_m = 0.02$ ,  $p_m^1 = 0.04$ ,  $p_m^2 = 0.012$ .

that dynamic mpath-streaming to outperform single-path TCP streaming. This is indeed the case. One example is shown in Fig. 7 where  $p_m = p_s = 0.004, 0.02$  or  $0.04$ ,  $T_O = 4$  and the RTT is 300 ms. The video playback rate is varies such that  $T/\mu$  varies from 1.6 to 2. We observe that the startup delays required so that the fraction of late packets lies below  $10^{-4}$  are significantly lower in multipath streaming (the third bar) than those in single-path streaming (the first bar). The above results indicate that two equivalent paths can support videos with twice (even more than twice) the bitrate than that supported by a single path.

For the sake of comparison, we also plot the required startup delays for single-path streaming under the same achievable TCP throughput ( $2T_s$ ) and playback rate ( $2\mu_s$ ) as those in multipath streaming (the second bar). This is obtained by using a loss rate lower than  $p_s$ , calculated using the formula for achievable TCP throughput in [20]. We observe that using two paths provides a better performance than that using a higher-bandwidth single path. This demonstrates that using multiple paths can be more beneficial than increasing the bandwidth of a single path. When the access link is the bottleneck of a path, this indicates that subscribing to two providers can be better than choosing a provider that doubles the current achievable TCP throughput.

### B. Impact of path diversity

To explore the impact of path diversity, we compare the performances of multipath streaming with homogeneous and heterogeneous paths. For homogeneous paths, let  $p_m$  and  $R_m$  denote respectively the loss rate and RTT on both paths. For heterogeneous paths, let  $p_m^k$  and  $R_m^k$  denote respectively the loss rate and RTT on the  $k$ -th path,  $k = 1, 2$ . We consider two types of heterogenous paths: (1) the paths have different RTTs. In this case, we set  $p_m^1 = p_m^2 = p_m$ ,  $R_m^1 = \gamma R_m$ ,  $R_m^2 = 1/(2 - 1/\gamma)$  so that the achievable TCP throughputs under homogeneous and heterogenous paths are the same; (2) the paths have different loss rates. In this case, we set  $R_m^1 = R_m^2 = R_m$ ,  $p_m^1 = \gamma p_m$ , and  $p_m^2$  is set using the formula for the achievable TCP throughput in [20] to obtain the same aggregate achievable TCP throughput as that in homogeneous paths. We set  $p_m = 0.004, 0.02$  or  $0.04$ ,  $R_m = 300$  ms, and  $\gamma = 1.5$  or  $2.0$ . For all the settings, we observe close performance between homogeneous and heterogenous paths. Fig. 8 shows the result when  $p_m = 0.02$ ,  $R_m = 300$  ms,  $\gamma = 2$ . The above results indicate that the performance of multipath streaming is not dramatically affected by path diversity. Rather, the performance is more closely related to the aggregate achievable TCP throughput over the multiple paths.

## VII. CONCLUSIONS AND FUTURE WORK

In this paper, we developed two schemes, static and dynamic mpath-streaming, for multipath live streaming via TCP and analytical models for each of them. We validated the models using extensive *ns* simulation and Internet experiments. Using these models, we explored the performance of multipath TCP live streaming under homogeneous and heterogeneous paths. Our results demonstrate that multipath TCP live streaming generally leads to satisfactory performance when the aggregate achievable TCP throughput over the multiple paths is twice the media bit rate, with a few seconds of startup delay. Furthermore, multipath live streaming via TCP over two similar paths can support more than twice the media bitrate supported by single-path TCP streaming over one path. As future work, we are pursuing in the following directions: (1) explore the performance of multipath stored-video streaming via TCP; (2) investigate the performance of multipath streaming via TCP under other multipath architectures, e.g., through multiple servers streaming video to a client or through an overlay network.

### APPENDIX I

#### MODEL FOR STATIC MPATH-STREAMING

Our model for static mpath-streaming consists of  $K$  independent Markov models, each corresponding to a streaming process on a path, with the length of a round (i.e., RTT) of that path as the time unit. More specifically, the model for the  $k$ -th path is represented as a tuple  $(X_i^k, N_i^k)$ , where  $X_i^k$  is the state of the TCP flow (see Section IV-A) and  $N_i^k$  is the number of early packets after the  $i$ -th round,  $N_i^k \leq N_{max}^k = \alpha_k \mu \tau$ ,  $k = 1, \dots, K$ .

We now describe in detail the state transition probabilities for the Markov model on the  $k$ -path and the time taken for each state transition. Let  $p_{w,c,l,e,r,n;w',c',l',e',r',n'}^k = P(W_{i+1}^k = w', C_{i+1}^k = c', L_{i+1}^k = l', E_{i+1}^k = e', R_{i+1}^k = r', N_{i+1}^k = n' \mid W_i^k = w, C_i^k = c, L_i^k = l, E_i^k = e, R_i^k = r, N_i^k = n)$  be the probability associated with the state transition from state  $(W_i^k = w, C_i^k = c, L_i^k = l, E_i^k = e, R_i^k = r, N_i^k = n)$  to state  $(W_{i+1}^k = w', C_{i+1}^k = c', L_{i+1}^k = l', E_{i+1}^k = e', R_{i+1}^k = r', N_{i+1}^k = n')$ . Let  $r_{w,c,l,e,r,n;w',c',l',e',r',n'}^k$  be the time taken for this state transition. Let  $p_k$  and  $R_k$  denote respectively the loss probability and the RTT on the  $k$ -th path. Let  $R_{TO}^k$  be the value of the first retransmission timer. Denote the maximum window size as  $W_{max}$ . The state transition probabilities and the times taken for the transitions are listed in Table IV. 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.

### APPENDIX II

#### MODEL FOR DYNAMIC MPATH-STREAMING

Let the tuple  $(X_i^1, \dots, X_i^K, N_i)$  represent the finite-state Markov chain for dynamic mpath-streaming, where  $X_i^k$  is the state of the TCP connection on the  $k$ -th path and  $N_i$  is the number of early packets in the client's local buffer after the  $i$ -th transition,  $N_i \leq N_{max} = \mu \tau$ . The transition of the Markov chain occurs in the presence of two types of events: (1) when a TCP flow makes a transition, (2) when a packet is consumed (played back) from the client's local buffer. Let  $E_i = C$  denote the event that a packet consumption triggers transition  $i$ . Similarly, let  $E_i = P_k$  denote the event that a transition of the TCP flow on the  $k$ -th path triggers transition  $i$ .

We now describe in detail the state transition probabilities for the Markov model and the time taken for each state transition. If  $E_i = P_k$ , that is, a transition of the TCP flow on the  $k$ -th path triggers transition  $i$ , let  $p_{w,c,l,e,r,n;w',c',l',e',r',n'}^k = P(W_{i+1}^k = w', C_{i+1}^k = c', L_{i+1}^k = l', E_{i+1}^k = e', R_{i+1}^k = r', N_{i+1}^k = n' \mid W_i^k = w, C_i^k = c, L_i^k = l, E_i^k = e, R_i^k = r, N_i^k = n)$  be the probability associated with the state transition from state  $(X_i^1, \dots, X_i^K, N_i)$  to state  $(X_{i+1}^1, \dots, X_{i+1}^K, N_{i+1})$ , where

$X_{i+1}^j = X_i^j$ ,  $j = 1, \dots, K$ ,  $j \neq k$ , and  $X_i^k = (w, c, l, e, r)$ ,  $X_{i+1}^k = (w', c', l', e', r')$ ;  $r_{w,c,l,e,r,n;w',c',l',e',r',n'}$  be the time taken for this state transition. Then we have  $p_{w,c,l,e,r,n;w',c',l',e',r',n'}$  and  $r_{w,c,l,e,r,n;w',c',l',e',r',n'}$  the same as those listed in Table IV except that  $N_{max}^k$  is replaced by  $N_{max}$ . If  $E_i = C$ , that is, a packet consumption triggers transition  $i$ , let  $r_{n;n'}^c$  be the time taken for this state transition (i.e., from state  $(X_i^1, \dots, X_i^K, N_i)$  to state  $(X_{i+1}^1, \dots, X_{i+1}^K, N_{i+1})$ ), where  $X_{i+1}^k = X_i^k$ ,  $k = 1, \dots, K$ , and  $N_i = n$ ,  $N_{i+1} = n' = n - 1$ . Then  $r_{n;n-1}^c = 1/\mu$ .

#### REFERENCES

- [1] L. Golubchik, J. Lui, T. Tung, A. Chow, W. Lee, G. Franceschinis, and C. Anglano, "Multi-path continuous media streaming: What are the benefits?," *Performance Evaluation*, pp. 429–449, 2002.
- [2] B. Abdouni, B. Cheng, A. L. Chow, L. Golubchik, and J. Liu, "Picture-perfect streaming over the internet: is there hope?," *IEEE Communications Magazine*, pp. 72–79, August 2004.
- [3] J. Apostolopoulos, T. Wong, W. Tan, and S. Wee, "On multiple description streaming with content delivery networks," in *IEEE INFOCOM*, 2002.
- [4] J. G. Apostolopoulos and M. D. Trott, "Path diversity for enhanced media streaming," *IEEE Communications Magazine*, August 2004.
- [5] Y. J. Liang, E. G. Steinbach, and B. Girod, "Multi-stream voice over ip using packet path diversity," in *IEEE Fourth Workshop on Multimedia Signal Processing*, (Cannes, France), 2001.
- [6] Y. J. Liang, E. G. Steinbach, and B. Girod, "Real-time voice communication over the internet using packet path diversity," in *ACM Multimedia*, (Ottawa, Canada), September/October 2001.
- [7] T. P. Nguyen and Z. Avidel, "Multiple sender distributed video streaming," *IEEE Transaction on Multimedia, Special Issue on Streaming Media*, vol. 6, pp. 315–326, April 2004.
- [8] B. Ribeiro, E. de Souza e Silva, and D. Towsley, "On the efficiency of path diversity for continuous media applications," Tech. Rep. 05-19, Department of Computer Science, University of Massachusetts, Amherst, 2005.
- [9] Y. Wang, M. Claypool, and Z. Zuo in *ACM SIGCOMM Internet Measurement Workshop*, (San Francisco, CA), November 2001.
- [10] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An analysis of live streaming workloads on the Internet," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, (Taormina, Sicily, Italy), pp. 41–54, 2004.
- [11] 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.
- [12] B. Wang, J. Kurose, P. Shenoy, and D. Towsley, "Multimedia streaming via TCP: An analytic performance study," in *Proc. ACM Multimedia*, October 2004.
- [13] D. Jurca and P. Frossard, "Packet selection and scheduling for multipath video streaming," *IEEE Transactions on Multimedia*, February 2006.
- [14] 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.
- [15] C. Krasic and J. Walpole, "Priority-progress streaming for quality-adaptive multimedia," in *ACM Multimedia Doctoral Symposium 2001*, (Ottawa, Canada), October 2001.
- [16] P. de Cuetos and K. W. Ross, "Adaptive rate control for streaming stored fine-grained scalable video," in *Proc. of NOSSDAV*, May 2002.
- [17] P. de Cuetos, P. Guillot, 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.
- [18] T. Kim and M. Ammar, "Receiver buffer requirements for video streaming over tcp," in *Proceedings of Visual Communications and Image Processing Conference*, (San Jose, CA), January 2006.
- [19] M. Mathis, J. Semke, and J. Mahdavi, "The macroscopic behavior of the TCP congestion avoidance algorithm," *Computer Communications Review*, vol. 27, no. 3, 1997.
- [20] 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.
- [21] 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.
- [22] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of TCP/IP with stationary random losses," in *SIGCOMM*, pp. 231–242, 2000.
- [23] 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.
- [24] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," in *INFOCOM (3)*, pp. 1742–1751, 2000.
- [25] M. Mellia, I. Stoica, and H. Zhang, "TCP model for short lived flows," *IEEE Communication Letters*, vol. 6, February 2002.
- [26] S. Bohacek, "A stochastic model of TCP and fair video transmission," in *Proc. IEEE INFOCOM*, 2003.
- [27] 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.
- [28] B. Wang, W. Wei, Z. Guo, and D. Towsley, "Multipath multimedia streaming via TCP: An analytic performance study," tech. rep., Computer Science and Engineering Department, University of Connecticut, 2006.
- [29] 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.
- [30] "tcpdump." <http://www.tcpdump.org/>.
- [31] *Planetlab*. <http://www.planet-lab.org/>.
- [32] 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.
- [33] 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.

$p_{w,0,0,0,0,n;w,1,0,0,n'}^k$	$= (1 - p_k)^w,$	$1 \leq w \leq W_{max}$
$p_{w,1,0,0,0,n;w+1,0,0,n'}^k$	$= (1 - p_k)^w,$	$n' = \min(N_{max}^k, n + w)$ $1 \leq w \leq W_{max},$
$p_{w,1,0,0,0,n;w,0,0,n'}^k$	$= (1 - p_k)^w,$	$n' = \min(N_{max}^k, n + w)$ $w = W_{max},$
$r_{w,0,0,0,0,n;w,1,0,0,n'}^k$	$= R_k,$	$n' = \min(N_{max}^k, n + w)$ $1 \leq w \leq W_{max}$
$r_{w,1,0,0,0,n;w+1,0,0,n'}^k$	$= R_k,$	$1 \leq w \leq W_{max}$
$r_{w,1,0,0,0,n;w,0,0,n'}^k$	$= R_k,$	$1 \leq w \leq W_{max}$
$p_{w,c,0,0,0,n;w-l,0,l,0,0,n'}^k$	$= p_k(1 - p_k)^{w-l},$	$2 \leq w \leq W_{max},$ $c = 0, 1, 1 \leq l \leq w,$ $n' = \min(N_{max}^k, n + w - l)$
$p_{w,c,0,0,0,n;1,0,0,1,1,n}^k$	$= p_k,$	$2 \leq w \leq W_{max}, c = 0, 1$
$r_{w,c,0,0,0,n;w-l,0,l,0,0,n'}^k$	$= R_k,$	$2 \leq w \leq W_{max},$ $c = 0, 1, 1 \leq l \leq w,$
$r_{w,c,0,0,0,n;1,0,0,1,1,n}^k$	$= R_{TO}^k,$	$2 \leq w \leq W_{max}, c = 0, 1$
$p_{1,0,l,0,0,n;1,0,0,1,1,n'}^k$	$= 1,$	$n' = \min(N_{max}^k, n + 1 - p_k)$
$p_{2,0,l,0,0,n;1,0,0,1,1,n'}^k$	$= 1,$	$n' = \min(N_{max}^k,$ $n + p_k(1 - p_k) + 2(1 - p_k)^2)$
$p_{w,0,l,0,0,n;1,0,0,1,1,n'}^k$	$= \sum_{i=1}^2 p_k(1 - p_k)^i,$	$3 \leq w \leq W_{max}$ $n' = \min(N_{max}^k,$ $n + \frac{\sum_{i=0}^2 ip_k(1 - p_k)^i}{\sum_{i=0}^2 p_k(1 - p_k)^i})$
$p_{w,0,l,0,0;[(w+l)/2],0,0,0,0}^k$	$= \sum_{i=3}^{w-1} p_k(1 - p_k)^i + (1 - p_k)^w,$	$3 \leq w \leq W_{max},$ $n' = \min(N_{max}^k,$ $n + \frac{\sum_{i=3}^{w-1} ip_k(1 - p_k)^i + w(1 - p_k)^w}{\sum_{i=3}^{w-1} p_k(1 - p_k)^i + (1 - p_k)^w})$
$r_{w,0,l,0,0,n;1,0,0,1,1,n'}^k$	$= R_{TO}^k - R_k,$	$1 \leq w \leq W_{max}$
$r_{w,0,l,0,0;[(w+l)/2],0,0,0,0}^k$	$= R_k,$	$3 \leq w \leq W_{max}$
$p_{1,0,0,i,r,n;1,0,0,\min\{i+1,7\},1,n}^k$	$= p_k,$	$1 \leq i \leq 7, r = 0, 1$
$p_{1,0,0,i,1,n;1,0,0,i,0,n+1}^k$	$= 1 - p_k,$	$1 \leq i \leq 7$
$p_{1,0,0,i,0,n;2,0,0,0,0,n+1}^k$	$= 1 - p_k,$	$1 \leq i \leq 7$
$r_{1,0,0,i,r,n;1,0,0,\min\{i+1,7\},1,n}^k$	$= 2^{(i-1)} R_{TO}^k,$	$1 \leq i \leq 7, r = 0, 1$
$r_{1,0,0,i,1,n;1,0,0,i,0,n+1}^k$	$= R_k,$	$1 \leq i \leq 7$
$r_{1,0,0,i,0,n;2,0,0,0,0,n+1}^k$	$= R_k,$	$1 \leq i \leq 7$
$r_{w,c,l,e,r,n;w,c,l,e,r,n'}^k$	$= R_k,$	$n' = n - \mu R_k$

TABLE IV

STATIC MPATH-STREAMING: DEFINITION OF THE STATE TRANSITION PROBABILITIES AND THE TIMES TAKEN FOR THE TRANSITIONS.