Streaming Stored Video over AIMD Transport Protocols

Gang Wu
School of ECE
Purdue University
gwu@ecn.purdue.edu

Edwin K. P. Chong
Department of ECE
Colorado State University
echong@engr.colostate.edu

Robert Givan
School of ECE
Purdue University
givan@purdue.edu

Abstract

We study the problem of using proxy servers to stream video stored at a geographically separate location. The separation of the server and the storage introduces a non-negligible delay in retrieving video frames in real time. To ensure network stability, we use an additive-increase and multiplicative-decrease transport protocol to support the streaming process. We develop an effective scheme to achieve high, consistent streaming quality. The heart of the scheme is the control of buffer occupancy at the proxy server. We model the buffer as a bilinear dynamical system with Poisson disturbance and develop three buffer controllers. Our empirical study proves the effectiveness of the streaming scheme. Moreover, we find that the controllers exploiting the buffer model demonstrate performance significantly superior to that of model-free controllers in overcoming the adverse impact of the control delay.

1. Introduction

We study the problem of a proxy server streaming video stored at a distant location to clients and develop an effective scheme to achieve high, consistent visual quality of the streamed video. Recent research has proposed using proxy servers for video delivery to keep manageable the bandwidth required from wide-area networks (WANs) [1]. Figure 1 illustrates such a scenario. The idea is to choose a cutoff size for video frames stored at the central storage and break each frame into two parts: one is of the cutoff size, the other the frame size reduced by the cutoff size. The latter partial frames are pre-fetched by and stored at the proxy server. When a streaming request arrives, the proxy server, in real time, retrieves partial frames stored at the central storage, assembles them with local partial frames, decompresses and re-compresses the restored complete frames to fit the varying bandwidth of the connection from the proxy server to the clients (this procedure is called *transcoding*), and finally transmits the transcoded frames to the client via the local-area network (LAN). Since partial frames fetched from the central storage are no larger than the cutoff size, the requirement of the expensive bandwidth from the WAN is limited. By making use of the limited storage space at the proxy server this way, the savings in the WAN bandwidth can be substantial.

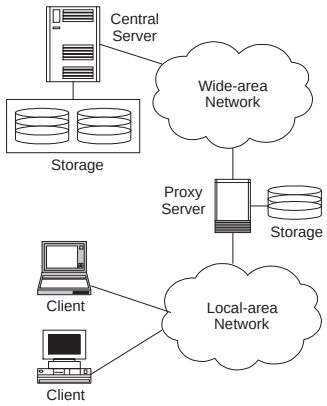

Figure 1. Video delivery using a proxy server.

We wish to develop a streaming scheme executed at the proxy server to achieve high and consistent visual quality in streaming for the aforementioned scenario. (We note that conventional streaming scenarios that involve only locally stored video is a special case of the broader case we consider here.) Overall visual quality is reflected in the throughput and the packet loss rate that the streaming application obtains. The larger the throughput and/or the smaller the loss rate, the better the overall quality. We desire to use the bandwidth offered by the underlying transport protocol to the maximum extent possible, to achieve high throughput, while at the same time maintain a low packet loss rate. Overall quality depends also on "quality consistency," by which we mean smooth changes in the quality of adjacent video frames. Abrupt changes will cause an artifact of

This research is supported by DARPA/ITO under contract F30602-00-2-0552, and by NSF under contracts 9977981-IIS, 0093100-IIS, 0098089-ECS, and 0099137-ANI. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, the National Science Foundation, or the U. S. Government.

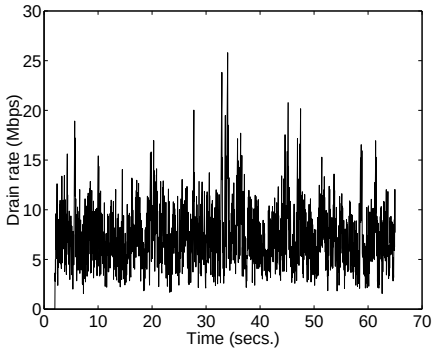

Figure 2. An example rate sequence of RAP.

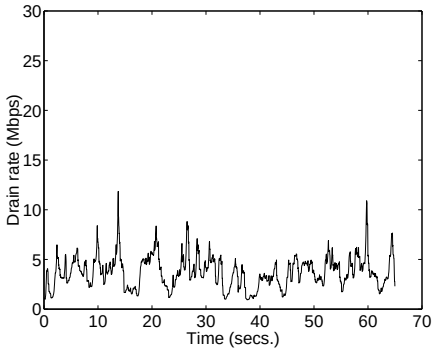

Figure 3. An example rate sequence of TFRC.

"flickering (or blinking)" display of the video, annoying to viewers [2]. Toward this end, we aim to reduce the variance in encoding bit-rates in the transcoding process; encoding bit-rates determine frame sizes, which are intimately related to visual quality. Our approach to accomplishing the above goals is through intelligent buffer control exercised at the proxy server.

Two features distinguish our streaming problem from previous research. One is the presence of a control delay arising from the physical distance between the proxy and the central storage servers. In the streaming process, it takes the proxy server a non-negligible amount of time to retrieve a particular partial frame from the central storage. This delay complicates the buffer-control problem, which constitutes the core of our solution to quality streaming.

The other feature of our problem is that we choose an *additive-increase multiplicative-decrease (AIMD) transport protocol* to support the streaming application and that we explicitly exploit the characteristics of such a protocol for quality streaming. We employ the Rate Adaptation Protocol (RAP), a well-known AIMD protocol suitable for real-time media delivery [3]. The success of the Internet over the past decade suggests that AIMD behavior is crucial for network stability. In fact, the Transmission Control Protocol (TCP), dominant in the Internet, is a typical AIMD protocol. The down side is that AIMD protocols, including RAP, suffer from jittery transmission rates; see Figure 2. The rate variation in Figures 2 is due both to bursty interfering traffic and to the AIMD behavior of RAP (which reduces the rate by a half when encountering congestion).

The *equation-based transport protocols* [4] can achieve smoother transmission rates than those of AIMD. However, we demonstrate in Figure 3 that in real network scenarios the result is still far from satisfactory. We obtained the shown result using a typical equation-based protocol called the *TCP-Friendly Rate-Control (TFRC) Protocol* [5]. When bursty interfering traffic is present, TFRC is unable to remove jitter at large timescales (e.g., seconds) and may produce flickering artifacts if used in video streaming.

We choose RAP as the underlying protocol based on the following reasons. First, it stabilizes networks, which is critical for healthy network operation. Second, its behavior allows neat mathematical characterization and thus enables a control-theoretic analysis of the problem. Finally, RAP is TCP-friendly; i.e., it shares network bandwidth with TCP connections in a fair way [3], a highly desirable property. We rule out equation-based protocols because they do not produce satisfactorily smooth rates and they have uncertain impact on network stability.

The technical core of our solution to the streaming problem is the control of the buffer occupancy at the proxy server. Two difficulties arise in the buffer control: the control delay and the stochastic wide-band variation of the drain rate determined by the underlying AIMD protocol. Without proper control, the buffer can easily underflow, resulting in loss in throughput. With AIMD protocols, such loss can be significant because, when the buffer is empty, not only is no data transmitted but also the drain rate stops increasing, leading to lower drain rate in the future. The buffer can easily overflow, too, resulting in excess loss of packets. For instance, [15] reports a loss of 25 frames out of 150 in an experiment on real-time video transcoding when a simple drop-tail buffer-management scheme is in place.

Our approach to avoiding buffer underflow or overflow is to maintain the buffer occupancy level at a target value (e.g., half of the buffer size) to achieve high throughput and low packet loss rates. We first develop a straightforward controller that does not assume any model of buffer dynamics. We then formulate the buffer control as a bilinear quadratic optimal control problem with state-dependent Poisson disturbance. Optimal control of a bilinear system with state-dependent Poisson disturbance for the control-delay–free case is studied in [6]; however, no complete solution is given there. The authors of [7] study control of manufacturing systems with state-dependent Poisson disturbance without control delay, and propose dynamic-programming methods as solutions. In [8], the authors model a TCP connection as a system with state-dependent Poisson disturbance to facilitate a fluid simulation of TCP networks; no control is discussed there. Compared with the above pre-

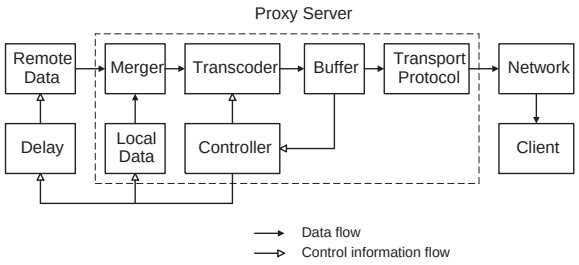

Figure 4. System block diagram.

vious work, our problem involves a control delay, which is addressed for the first time here for such systems.

Our contribution lies in the fact that we are the first to address the problem of buffer control for video-streaming involving geographically separated proxy servers and video content, developing an effective scheme for quality video streaming. A second contribution is that two of the developed buffer controllers are specifically designed for bilinear dynamical systems with Poisson disturbance and with a control delay, and, to our knowledge, our work here is the first attempt towards effectively controlling such systems. A third contribution is that, through our comparative study on model-free and model-based controllers, we demonstrate that, using a proper model, a controller can outperform a model-free controller by a significant margin.

The balance of this report is organized as follows. In Section 2, we formally describe the problem and present our general approach, with the emphasis on formulating the embedded buffer control as an optimization problem. Section 3 contains our main results in solving this buffer-control problem. We first present a model-free controller as a straightforward heuristic solution that serves as a comparison basis for the two model-based controllers we construct subsequently. One model-based controller is in fact the optimal solution to the delay-free case. Based on this optimal solution we then provide a second model-based controller which uses a heuristic to address the optimization problem for the delay system. We have in Section 4 our empirical results demonstrating the effectiveness of our solution scheme. Section 5 concludes the paper.

2. Problem Description and Solution Approach

2.1. System Model

Figure 4 shows a block diagram of the system shown in Figure 1. Within the proxy server, the controller, the merger, the transcoder, the transport protocol RAP, and the buffer are the essential components, and their detailed descriptions are given below. We assume that time t is continuous and that the frame rate of the video is F frames per second.

Controller. In the streaming process, at each time t, the controller makes a decision on the number of the frames to be processed, denoted by $n(t) \in \{0, 1, 2, \ldots\}$. The controller also decides the sizes of the transcoded frames; see the description of the transcoder below. Allowing $n(t)$ to vary with time and, in particular, to take values other than one draws a sharp line between our streaming approach and all previous ones in the literature. Previous schemes only process *one* frame at *discrete* times $t = i/F$, $i = 0, 1, \ldots$. Decision-making on $n(t)$ and on the sizes of the transcoded frames is the focus of our study in this paper, as described in the next subsection.

Merger. The application retrieves $n(t)$ (compressed) partial frames from the central server. After the round-trip delay from the proxy server to the central server, the streaming application receives these partial frames. The delay may vary over time due to varying available bandwidth in the WAN; however, as a first approximation to time-varying delays, here we assume a fixed and known delay h. The application then merges them with (compressed) locally stored partial frames and forms $n(t)$ complete frames.

Transcoder. The application then decompresses and re-compresses the frames into new sizes, and injects them into a buffer that is drained by the underlying transport protocol RAP. Denote by $s_j(t) \in \mathbb{R}$, $j = 1, \ldots, n(t-h)$, the sizes of the newly transcoded frames at time t, if $n(t-h) > 0$. The values of $s_j(t)$ (or the encoding bit-rates) are determined by the controller, discussed in the next subsection.

In this paper, we assume that only the first frame of the transcoded video is intra-coded while all the rest are inter-coded, as in [15]. We further assume that the visual quality of a frame depends on the size of the frame, but not on the content of the frame. This assumption is reasonable in practice and, moreover, with this assumption, we have a much simplified system (yet capturing the essence of the real system), resulting in greater tractability.

Transport Protocol. The transport protocol RAP has two main functions. One is to drain data packets (which make up the video frames) from the buffer mentioned above and send them to the client. The other function is to adapt its transmission rate according to network conditions to ensure stability. The transmission rate is identical to the drain rate of the buffer. Loosely, RAP increases its transmission rate (drain rate) linearly to probe for more available bandwidth until there is a congestion signal from the network (a packet-loss event, as in TCP), when RAP then reduces the current rate by half to alleviate congestion. To focus on the essential AIMD behavior of RAP, we disregard the behavior of RAP during the fast recovery and when time-outs happen.

We formally model the drain-rate process as follows. We assume the drain rate is left continuous and denote it by $v(t) \in \mathbb{R}$ for time t. The drain-rate process $\{v(t) : t \geq 0\}$

evolves as follows:

$$v(t) = at + v(t_i^+), \qquad t \in [t_i^+, t_{i+1}] \tag{1}$$
$$v(t_i^+) = bv(t_i), \qquad i = 0, 1, 2, \ldots, \tag{2}$$

where t_i, $i = 0, 1, 2, \ldots$, is the arrival time of the i-th network-congestion signal (we define $t_0 = 0$ and assume $v(t_0) = 0$), and t_i^+ denotes the time just after t_i. The real number $a > 0$ represents the slope of the linear increase of the drain rate when a congestion signal is absent and is a function of both the packet size and the round-trip time (from the proxy server to the client). The real number $0 < b < 1$ is the portion of the drain rate retained after receiving a congestion signal. We assume a and b ($= 1/2$ in RAP) are both constant.

Denote the number of arrived congestion signals at time t by $N(t)$. We assume the process $\{N(t) : t \geq 0\}$ to be Poisson, following [8]. We assume a constant rate of λ for the process $\{N(t) : t \geq 0\}$. This modeling is appropriate when the traffic load generated by the streaming application constitutes only a small portion of the overall traffic sharing the same path in the network. When the traffic load is sufficiently high, it may have non-negligible impact on the number of congestion events. In this case, it would be appropriate to consider a model where the number of congestion events depends on the proxy traffic (in our ongoing work, we use such a more sophisticated model). The dynamics of $v(t)$ can now be concisely expressed as follows:

$$dv(t) = adt + (b-1)v(t)dN(t). \tag{3}$$

Equation (3) represents a stochastic dynamical system in the Itô sense [11]; that is, (3) is an alternative expression for the following stochastic integral equation:

$$v(t) = v(0) + \int_0^t a\, ds + \int_0^t (b-1)v(s)\, dN(s), \tag{4}$$

where the last term on the right-hand side is a counting integral as defined in [12] (Definition 5.3.1).

Buffer. Denote by $\bar{u}(t) \in \mathbb{R}$ the number of bits injected into the buffer at time t. Then, we have

$$\bar{u}(t) = \begin{cases} \sum_{j=1}^{n(t-h)} s_j(t) & \text{if } n(t-h) > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

Writing the buffer-occupancy level (called the *buffer level* henceforth) at time t as $q(t) \in \mathbb{R}$, we model the buffer dynamics using a first-order differential equation:

$$\frac{d}{dt}q(t) = \bar{u}(t) - v(t). \tag{6}$$

As in many other works on buffer control (e.g., [9], [10]), we ignored buffer boundaries for tractability of the problem.

Combining (3) and (6), we have the following system:

$$\begin{bmatrix} dq(t) \\ dv(t) \\ dw(t) \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} q(t) \\ v(t) \\ w(t) \end{bmatrix} dt + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \bar{u}(t)dt + \begin{bmatrix} 0 & 0 & 0 \\ 0 & b-1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} q(t) \\ v(t) \\ w(t) \end{bmatrix} dN(t), \tag{7}$$

where the variable $w(t)$ is created to make the equation homogeneous. The initial condition is:

$$q(0) = q_0 \quad v(0) = 0 \quad w(0) = a, \quad t \in [-h, 0),$$

where q_0 and a, $t \in [-h, 0)$ are all known. The input to the buffer, $\bar{u}(t)$, for $t \geq h$ are free for us to decide to achieve some control objectives, described below. We assume the history of $\bar{u}(t)$, for $t < h$, is known.

We write the above dynamical equation (7) in a concise form as follows:

$$d\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t)dt + \mathbf{B}\bar{u}(t)dt + \mathbf{C}\mathbf{x}(t)dN(t), \tag{8}$$

where $\mathbf{x}(t) = [q(t), v(t), w(t)]' \in \mathbb{R}^3$ (the prime denotes transposition); and **A**, **B**, and **C** are corresponding constant matrices with appropriate dimensions. The initial condition is implicitly given by the previously specified initial condition, and we do not write it down explicitly here.

2.2. The Control Problem and the Solution Approach

In this subsection, we describe the two control decisions made by the streaming application at time t: deciding the number of frames to be processed and determining the target sizes of the transcoded frames. The streaming application can use the system observations and the model of the buffer dynamics described earlier to assist decision-making. The decision-making also needs to take into account the retrieval delay of the remote partial frames; i.e., the desired number of remote partial frames will arrive only at $t + h$.

Our first objective is to transcode frames into a constant size to obtain quality consistency. In general, to achieve the same visual quality, different types of frames require different numbers of bits. However, in our setting, all (but the first frame) are of the same type (inter-coded frames), and thus the same frame size will guarantee the same visual quality. In implementation, we directly put the first frame into the buffer without transcoding it. We apply the control starting from the second frame. We therefore exclude the first frame from the following discussion. We prioritize consistent quality over other objectives because we believe a jittery display of the video results in more dissatisfaction than video presented with constant quality, even if the average throughput of the latter is smaller.

A natural way to reach the first objective is to set $s_j(t) = S$ (a constant) for all t and j. We choose S to be the expected throughput of RAP, denoted by R, divided by the frame rate F. The rationale for setting $S = R/F$ is that as time t progresses, the total number of transcoded frames will equal Ft, which is as desired. We assume that the drain rate of RAP under the controlled buffer is a stationary process and that R is known. In our experiments, we estimate R by averaging over a history of the past drain rates.

Our second objective is to choose an appropriate $n(t)$ at each t to maximize the usage of the bandwidth provided by RAP while keeping the packet loss rate to the minimum, subject to the constraint on the frame sizes. This objective is made precise as follows. Define $T \in \mathbb{R}$ to be the finite time period we consider and $u(t) \in \mathbb{R}^+$ to be the desired rate calculated by our controller at time t. Due to the retrieval delay, the actual input rate into the buffer at time t is then $u(t-h)$. We term $u(t)$ as the "control" of the system and call $u(t-h)$ the "input" to the buffer. Since we have fixed the frame size to meet our first objective, we have

$$u(t) = \bar{u}(t+h) = n(t)S. \tag{9}$$

Notice that (9) constrains $u(t)$ to an integral multiple of S. Then, our second objective is to choose a sequence of $n(t)$ (or, equivalently, a sequence of $u(t)$) for the time period of $[0, T]$ to minimize the following objective function for a given initial condition $\mathbf{x}(0)$:

$$V(\mathbf{x}(0), u(t)) = E\left\{\int_0^T \left[(q(t)-Q)^2 + \alpha u^2(t)\right] dt\right\}, \tag{10}$$

where Q is the target buffer level, $\alpha > 0$ is a weighting coefficient, and E denotes expectation. The first term on the right-hand side reflects our wish to bring the buffer level to the target size to avoid buffer underflow or overflow. The second term is a penalty on using excess bandwidth in the WAN; recall that $u(t)/S$ partial frames will be fetched from the central storage via the WAN in real time.

The problem of minimizing (10) is in essence a buffer-control problem, with the goal to drive the buffer level to the target value. Solving the buffer-control problem is the heart of our approach. Owing to both the delay involved and the special constraint on the control (see (9)), to our knowledge this optimization problem does not lend itself to any known exact solution methods. We therefore resort to heuristic solutions in the following section.

3. Solutions to the Buffer-control Problem

3.1. A Model-free Controller

We first present a straightforward heuristic solution to optimizing (10). This controller does not make decisions with any regard to the cost function in (10). In fact, the controller does not exploit any model of the buffer dynamics. We construct this controller for two purposes: to have a simple solution to the buffer-control problem, and to use it as a comparison basis for the two controllers we will develop next, which explicitly exploit the buffer model.

The controller computes the control as follows:

$$\tilde{u}(t) = \gamma(q(t)-Q) + v(t) \tag{11}$$

$$n(t) = \max\{\lfloor \tilde{u}(t)/S \rfloor, 0\} \tag{12}$$

$$u(t) = n(t)S, \tag{13}$$

where $\gamma < 0$ is the gain of the error term on the buffer level and $v(t)$ is the drain rate as before. The term $\gamma(q(t)-Q)$ in (11) is to drive the deviation of the buffer level to zero, while the $v(t)$ term is for the controller to track the varying drain rate. The value of $v(t)$ can be measured simply by monitoring the speed at which the packets are disappearing from the buffer. This model-free controller resembles the well-known proportional-derivative controllers in classical control theory. The variable $\tilde{u}(t) \in \mathbb{R}$ in (11) denotes the control that we would apply if the constraints on $u(t)$ in (9) did not exist. Equations (12) and (13) are to ensure that $u(t)$ is nonnegative and a multiple of S.

3.2. Zero-delay Optimal Controller

In this subsection and the next, we present two controllers constructed based on an optimal-control analysis of the buffer-control problem. To apply optimal-control analysis, we consider the control $\tilde{u}(t) \in \mathbb{R}$ instead of $u(t)$ here; i.e., $\tilde{u}(t)$ can take any real value, and we assume that it is directly applied to the system without further manipulation. We also set the retrieval delay to zero and re-define $\mathbf{x}(t) = [q(t)-Q, v(t), w(t)]'$. With zero delay, the dynamical equation governing the system becomes

$$d\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t)dt + \mathbf{B}\tilde{u}(t)dt + \mathbf{C}\mathbf{x}(t)dN(t). \tag{14}$$

Our choice of control $\tilde{u}(t)$, $t \in [0, T]$ here is based on finding a "state-feedback" map $\mu : [0, T] \times \mathbb{R}^3 \mapsto \mathbb{R}$ and apply $\tilde{u}(t) = \mu(t, \mathbf{x}(t))$ to the system. The sequence of maps $\pi = \{\mu(t, \mathbf{x}(t)), t \in [0, T]\}$ is called a *policy*. Define the instantaneous cost

$$L(t, \mathbf{x}(t), \tilde{u}(t)) = \mathbf{x}(t)'\mathbf{M}\mathbf{x}(t) + \alpha\tilde{u}^2(t),$$

where $\mathbf{M}$ is a positive-semidefinite matrix of appropriate dimensions and can easily be constructed from the integrand in (10). Then, our goal is to find a policy π to minimize the following cost function:

$$V_T^\pi(\mathbf{x}(0)) = E\left\{\int_0^T L(t, \mathbf{x}(t), \tilde{u}(t))dt\right\}. \tag{15}$$

Define $\mathbf{I}$ to be an identity matrix of appropriate dimensions. Following the spirit in [6], we have the following optimal control law (the proof is omitted due to space limitations).

Proposition 1 *For the system given in (14), the optimal control policy that minimizes the cost function in (15) is*

$$\tilde{u}(t) = -(1/\alpha)\boldsymbol{B}'\boldsymbol{P}(t)\boldsymbol{x}(t), \tag{16}$$

where $\boldsymbol{P}(t)$ is assumed symmetric without loss of generality and is the solution to the following Riccati equation

$$\frac{d}{dt}\boldsymbol{P}(t) + \boldsymbol{A}'\boldsymbol{P}(t) + \boldsymbol{P}(t)\boldsymbol{A} + \lambda\boldsymbol{D}'\boldsymbol{P}(t)\boldsymbol{D} + \boldsymbol{M} - \lambda\boldsymbol{P}(t) - (1/\alpha)\boldsymbol{P}(t)\boldsymbol{B}\boldsymbol{B}'\boldsymbol{P}(t) = \boldsymbol{0}, \tag{17}$$

with the terminal condition $\boldsymbol{P}(T) = \boldsymbol{0}$ and where $\boldsymbol{D} = \boldsymbol{I} + \boldsymbol{C}$.

Based on Proposition 1, the real control $u(t)$ (conforming to (9)) we apply to the system is then decided using (12) and (13). The control decision $u(t)$ is made based on the modeling of the buffer, including the behavior of the underlying RAP. However, $u(t)$ is computed without any awareness of the delay h. Nevertheless, we apply such calculated $u(t)$ to the delay system as a heuristic. One would expect its performance to be better than that of the model-free controller; in fact, our experiments shown later confirm this.

3.3. Expectation Controller

The optimal control obtained in (16) is linear in the state feedback. However, when the control delay is present, no linear state-feedback solution exists, and no other analytical solution has been attained. (This can be seen from the fact that a delay-free linear system with state-dependent noise and with partial observation does not admit an optimal linear control policy and that no other optimal policy is available analytically; see, e.g., [13].) Therefore, in this subsection, we develop a controller by taking into account the delay in a heuristic way.

We observe that, at time t, if the future state $\mathbf{x}(t+h)$ were known so that we could use it as the state feedback and apply the policy given in Proposition 1, then the cost function in (15) would indeed be minimized. Hence, a natural extension of the optimal policy in Proposition 1 to the case with a control delay is to find an estimate of $\mathbf{x}(t+h)$. As a heuristic, we use the expected value of $\mathbf{x}(t+h)$ given the system information up to time t as an estimate.

The idea is formalized as follows. With the control delay h, the system dynamical equation takes the form:

$$d\mathbf{x}(t) = \mathbf{A}\mathbf{x}(t)dt + \mathbf{B}\tilde{u}(t-h)dt + \mathbf{C}\mathbf{x}(t)dN(t). \tag{18}$$

Furthermore, minimizing the original cost function (15) is now equivalent to minimizing the following cost function

$$V_T^{\pi}(\mathbf{x}(0)) = E\left\{\int_h^T L(t, \mathbf{x}(t), \tilde{u}(t-h))dt\right\}. \tag{19}$$

From Proposition 1, we have the optimal control policy

$$\tilde{u}(t-h) = -(1/\alpha)\mathbf{B}'\mathbf{P}(t)\mathbf{x}(t), \quad h \le t \le T. \tag{20}$$

Therefore, we have

$$\tilde{u}(t) = -(1/\alpha)\mathbf{B}'\mathbf{P}(t+h)\mathbf{x}(t+h), \quad 0 \le t \le T-h. \tag{21}$$

We note that $\tilde{u}(t)$, for $t > T-h$, has no impact on the system due to the delay h.

However, we do not know the future state $\mathbf{x}(t+h)$ at time t. Hence, as mentioned before we compute the following estimate and use it in (21) in place of $\mathbf{x}(t+h)$

$$\hat{\mathbf{x}}(t+h) = E\{\mathbf{x}(t+h)|\mathbf{H}(t)\}, \tag{22}$$

where the set $\mathbf{H}(t) = \{\mathbf{x}(s) : s \le t\} \bigcup \{u(s) : s < t\}$ denotes the information we have up to time t.

The estimate $\hat{\mathbf{x}}(t+h)$ can be readily calculated (see [12] and [14]). Let

$$\hat{\mathbf{x}}(t+r) = E\{\mathbf{x}(t+r)|\mathbf{H}(t)\}, r \ge 0.$$

Then, we have

$$\frac{d}{dr}\hat{\mathbf{x}}(t+r) = \mathbf{A}\hat{\mathbf{x}}(t+r) + \mathbf{B}\tilde{u}(t-h+r) + \lambda\mathbf{C}\hat{\mathbf{x}}(t+r). \tag{23}$$

We can compute $\hat{\mathbf{x}}(t+h)$, using (23) and $\hat{\mathbf{x}}(t) = \mathbf{x}(t)$.

The control $u(t)$ is then calculated from $\tilde{u}(t)$ using equations similar to (12) to (13). Because the expectation controller uses extra information (the delay h) in decision-making, it outperforms the other two controllers when the delay h is present, as we show in the next section.

4. Empirical Study

4.1. Evaluation Setup

Our empirical study is carried out on a simulated network shown in Figure 5 using the network simulator *ns2*. As shown, four source-destination pairs, numbered (S_i, D_i), $i = 1, \ldots, 4$, share the same link between the routers R_1 and R_2. Traversing from S_1 to D_1 is a real-traffic trace captured by the National Laboratory for Applied Network Research (http://www.nlanr.net), representing bursty traffic often seen in real networks and simulating interfering traffic *not responding* to congestion. Two File-Transfer-Protocol (FTP) applications transmit data from S_2 to D_2 and from S_3 to D_3, respectively. The FTP applications use TCP as their underlying transport protocol, thereby simulating interfering traffic *responding* to congestion.

Source S_4 employs the scheme developed in this paper to stream video (assumed stored elsewhere) to D_4. We set the frame rate $F = 30$ and assume that in transcoding the streaming application uses the Fine-Grain-Scalability

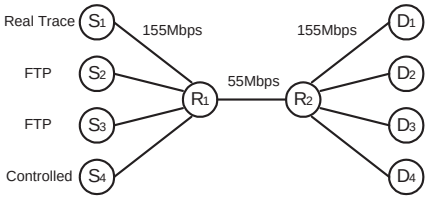

Figure 5. Simulate network.

(FGS) coding mechanism [17], developed for Motion-Picture-Expert-Group-4 (MPEG-4) video. An FGS encoder encodes a stream into a base-layer stream and an enhancement stream. The enhancement stream can be cut off anywhere to achieve exactly the target bit-rate. Therefore, unlike many other encoding schemes (e.g., MPEG-2 Test-Model-5 encoder [16]), an FGS encoder can eliminate any discrepancy between the desired frame size and the actual encoded frame size. The buffer at the source S_4 is of size 250 packets, each packet of the maximum size of 512 bytes. The target queue size Q is 125 packets.

We apply the two model-based controllers in a "receding-horizon" way, common in the optimal-control literature (see, e.g., [18]). At each decision-making time, we compute $\mathbf{P}(t)$ by setting the horizon $T = 300$ time steps (each time step is $1/F \approx 33$ ms) and use only $\mathbf{P}(0)$ in calculating the current control. In fact, $\mathbf{P}(t)$ is computed only once, and we repeatedly use $\mathbf{P}(0)$ in every decision-making epoch.

4.2. Simulation Results

In this subsection, we first present the results of our streaming scheme in terms of quality consistency of the streamed video. We then study the impact of the control delay on the performance of the buffer controllers used in the streaming scheme.

Quality Consistency. Figure 6 shows the frame-size sequence generated by the model-free controller. The control delay is zero in this experiment. This smooth result is achieved despite the bursty behavior of RAP as shown previously in Figure 2. The results of the zero-delay optimal controller and the expectation controller are almost identical to the one illustrated in Figure 6 and are hence omitted.

Figure 7 shows the frame sizes generated by a conventional streaming application, where a frame is produced every $1/F \approx 33$ ms at the bit-rate specified by the underlying TFRC protocol. (The behavior of TFRC in this experiment has been previously shown in Figure 3.) The control delay is also set to zero. Despite the supposedly "smooth" transmission rates generated by TFRC, it is apparent that the resulting quality inconsistency is beyond tolerable.

Comparing Figure 6 with Figure 7, we see a marked reduction in the variance of the frame sizes by using our scheme, leading to significant improvement in quality consistency. Our experiments with non-zero control delays reveal similar results and therefore we omit the plots here.

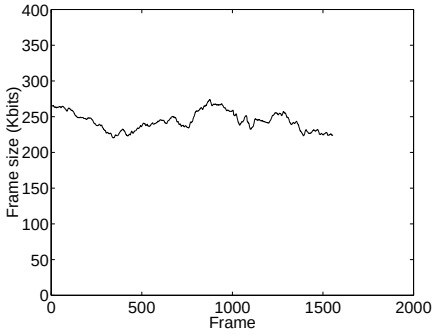

Figure 6. Frame sizes associated with the model-free controller.

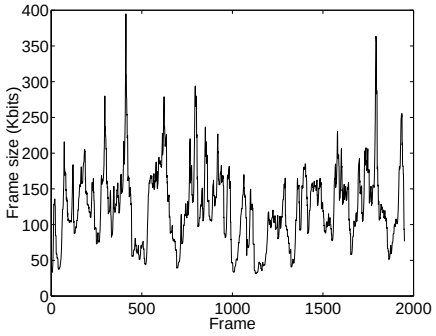

Figure 7. Frame sizes associated with a conventional streaming scheme using TFRC.

Impact of Control Delays. Figure 8 shows the utilization values achieved versus the control delay. We define utilization as the ratio of the number of packets actually transmitted by RAP to the number of the packets that would be transmitted during the same time period if the buffer never empties. As expected, the larger the control delay, the more severe the negative impact of the delay. It is clear that the expectation controller is more robust to the control delay than the zero-delay optimal controller. The reason is that the former exploits more information of the system than the latter—the former predicts the future state in the expectation sense. Moreover, these two model-based controllers both achieve utilizations larger than the model-free controller does, since the model-free controller does not exploit any system model. We see that the expectation controller has around 10% improvement over the model-free controller when the control delay is 15 time steps (at the upper-end of typical delays).

Figure 9 illustrates the packet loss rates associated with the three controllers. We define the packet loss rate as the number of the packets lost at the source due to buffer overflow divided by the total number of packets generated by the streaming application. Figure 9 demonstrates the negative

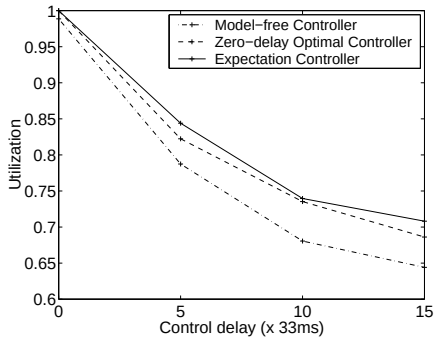

Figure 8. Utilization versus control delay.

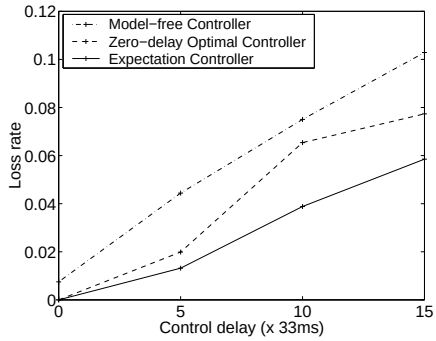

Figure 9. Packet loss rate versus control delay.

impact on the packet loss rate resulting from the control delay: larger control delays incur more packet losses. It is to be expected that a model-based controller has lower packet loss than in a model-free controller. The figure shows that the expectation controller experiences only about half (or even less) of the loss suffered by the model-free controller.

Figures 8 and 9 together clearly show that in the presence of a non-negligible control delay, exploiting a model of the system can increase the robustness of a controller against the impact of the control delay and therefore result in substantial performance improvement.

5. Conclusions

We have studied a problem of streaming stored video that includes geographically separated proxy servers and video data, and have developed a scheme effectively achieving high, consistent streaming quality. Lying in the heart of the scheme is a controller that intelligently manages the buffer occupancy to attain high utilization while maintaining low packet loss rates. We offer three choices of the buffer controller, and we have found that the more information a controller uses in decision-making, the more robust it is against the adverse effects caused by the control delay.

Our streaming scheme and buffer-control techniques have a broad range of applications. For instance, they can readily be applied to voice/audio streaming. Moreover, the buffer controllers are also applicable to delivering data files when the files and the delivery server are physically far apart and when an AIMD transport protocol is used.

Although the performance of our schemes is promising, one important problem suggests further study. We intra-code only the first frame. In reality, frames are often required to be intra-coded periodically to combat propagation of coding error and packet loss. Periodic intra-coded frames add extra dynamics to the system and thus impose more complexity on solving the buffer-control problem.

References

[1] Z.-L. Zhang, Y. Wang, D. H. C. Du, and D. Su, "Video staging: A proxy-server-based approach to end-to-end video delivery over wide-area networks," *IEEE/ACM Trans. Networking*, vol. 8, no. 4, Aug. 2000, pp. 429–442.

[2] H. Song and C.-C. J. Kuo, "Rate control for low-bit-rate video via variable-encoding frame rates," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 4, Apr. 2001, pp. 512–521.

[3] R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet," in *Proc. IEEE INFOCOM*, 1999, New York, vol. 3, pp. 1337–1345.

[4] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," *ACM Computer Communication Review*, vol. 30, no. 4, Oct. 2000, pp. 43–56.

[5] M. Handley, J. Padhye, S. Floyd, and J. Widmer, "TCP friendly rate control (TFRC): Protocol specification," *Internet Draft*, http://www.icir.org/tfrc (work in progress, current Apr. 2002).

[6] W. M. Wonham, "Random differential equations in control theory," *Probabilistic Methods in Applied Mathematics, Vol. 2*, edited by A. T. Bharucha-Reid, Academic Press, New York, 1970.

[7] J. J. Westman, F. B. Hanson, "Non-linear state dynamics: computational methods and manufacturing application," *International Journal of Control*, vol. 73, no. 6, Apr. 2000, pp. 464–480.

[8] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proc. SIGCOMM*, Stockolm, Sweden, 2000.

[9] E. Altman and T. Basar, "Multiuser rate-based flow control," *IEEE Trans. Commun.*, vol. 46, no. 7, Jul. 1998, pp. 940–949.

[10] A. Kolarov and G. Ramamurthy, "A control-theoretic approach to the design of an explicit rate controller for ABR service," *IEEE/ACM Trans. Networking*, vol. 7, no. 5, Oct. 1999, pp. 741–753.

[11] K. Ito, "Stochastic differential equations in a differential manifold," *Nagoya Mathematical Journal*, vol. 1, 1950, pp. 35–47.

[12] D. L. Snyder and M. I. Miller, *Random Point Processes in Time and Space*, Springer-Verlag, New York, 1991.

[13] R. H. Kwong and N. Meijer, "Optimal control of partially observable linear systems with state and control dependent noise," in *Proc. 37th IEEE Conf. Decision and Control*, Orlando, FL, 1982, vol. 3, pp. 1322–1323.

[14] S. I. Marcus, "Modeling and analysis of stochastic differential equations driven by point processes," *IEEE/ACM Trans. Information Theory*, vol. IT-24, no. 2, Mar. 1978, pp. 164–172.

[15] Q. Zhang, W. Zhu, and Y.-Q. Zhang, "Resource allocation for multimedia streaming over the Internet," *IEEE Trans. Multimedia*, vol. 3, no. 3, Sept. 2001, pp. 339–355.

[16] T. Chiang and Y.-Q. Zhang, "A new rate control scheme using quadratic rate distortion model," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 7, no. 1, Feb. 1997, pp. 246–250.

[17] W. Li, "Overview of Fine Granularity Scalability in MPEG-4 video standard," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 11, no. 3, Mar. 2001, pp. 301–317.

[18] D. Q. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *IEEE Trans. Auto. Contr.*, vol. 35, no. 7, Jul. 1990, pp. 814–824.