RaDiO Edge: Rate-Distortion Optimized Proxy-Driven Streaming From the Network Edge

Jacob Chakareski and Philip A. Chou, *Fellow, IEEE*

***Abstract*—This paper addresses the problem of streaming packetized media over a lossy packet network through an intermediate proxy server to a client, in a rate-distortion optimized way. The proxy, located at the junction of the backbone network and the last hop to the client, coordinates the communication between the media server and the client using hybrid receiver/sender-driven streaming in a rate-distortion optimization framework. The framework enables the proxy to determine at every instant which packets, if any, it should either request from the media server or (re)transmit directly to the client, in order to meet constraints on the average transmission rates on the backbone and the last hop while minimizing the average end-to-end distortion. Performance gains are observed over rate-distortion optimized sender-driven systems for streaming packetized video content. The improvement in performance depends on the quality of the network path both in the backbone network and along the last hop.**

***Index Terms*—Audio coding, channel coding, edge-based streaming, error control, Internet, Markov processes, multimedia communication, optimal control, protocols, proxy servers, rate-distortion, video coding.**

I. Introduction

WE CONSIDER the problem of streaming packetized media over a lossy backbone packet network through an intermediate proxy server to a client, in a rate-distortion optimized way. The proxy is located at the junction of the backbone network and the last hop to the client. The proxy employs a hybrid receiver/sender-driven transmission scheme to communicate simultaneously with the media server and the client. Packets may be lost in the backbone network due to congestion, or on the last hop due to erasures. The scenario under consideration is illustrated in Fig. 1.

Rate-Distortion Optimized (RaDiO) packet scheduling is one of the latest advances in media streaming. In this approach, the media server or the client is equipped with a rate-distortion optimization framework for scheduling packet transmissions such that a constraint on the average transmission rate is met while minimizing at the same time the average end-to-end distortion. In [1]–[3], the authors introduce a framework for distortion-rate optimized scheduling of the transmissions of packetized media and apply it to the scenario of sender-driven streaming. In [4], the authors employ this framework to study the scenario of receiver-driven transmission over best-effort networks. Rate-distortion optimized streaming over lossy packet networks to wireless clients, again in a sender-driven scenario, is studied in [5]–[7]. Finally, the RaDiO framework from [1] and [2] is employed in [8] to study proxy caching in a cost-distortion optimization scenario. All these works demonstrate a substantial improvement in performance over current state-of-the-art solutions.

The present paper uses the above works on RaDiO packet scheduling as a starting point, and realizes additional improvements in performance by performing the scheduling at the edge of the backbone network. Intuitive support for our reasoning can be found in other concepts in media streaming such as proxy caching and monitoring agents, where the ability to store and to locate information at the edge of the backbone network is exploited to provide improved performance. In proxy caching [8]–[17], proxy servers temporarily store the most frequently accessed content from a central media server. The cached content is served afterwards from the proxy servers to newly connecting clients. Proxy caching is in fact an instance of the so-called *edge architecture* approach, where several servers are strategically placed around the Internet so that each client can choose the server that results in shortest round-trip time and least amount of congestion. The edge servers are typically placed at junction points of two or more heterogeneous networks. Many content delivery network (CDN) companies, such as Akamai[1] and Speedera Networks,[2] use this technology to achieve better load balancing and higher throughput, while reducing access latency and bandwidth requirements. Proxy servers can also be successfully used as video transcoders to provide the necessary rate scaling between two or more heterogeneous networks [18]–[23]. In such scenarios, instead of signaling back to the communication source, bandwidth bottleneck problems are dynamically resolved by the proxy during media transmission. In addition to rate scaling, some video transcoding proxies also provide error resilience support as a more rapid and dynamic way of error handling at the edge of different networks [24], [25]. Furthermore, in [26], [27] a scheme is proposed that combines stream scheduling at an origin server and prefix/partial caching at an intermediate proxy to minimize the aggregate transmission rate over the backbone network under a cache capacity constraint. It is shown through simulations that the proposed scheme provides significant improvement in performance relative to a full caching technique.

Manuscript received November 6, 2003; revised May 17, 2005; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor K. Ross.

J. Chakareski was with Stanford University, Palo Alto, CA, and with the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. He is now with Layered Media Inc., Rochelle Park, NJ 07662 USA (e-mail: jakov@jakov.org).

P. A. Chou is with Microsoft Research, Microsoft Corporation, Redmond, WA 98052-6399 USA (e-mail: pachou@microsoft.com).

Digital Object Identifier 10.1109/TNET.2006.886299

[1]Akamai Technologies, Inc. [Online.] Available: http://www.akamai.com

[2]Speedera Networks, Inc. [Online.] Available: http://www.speedera.com

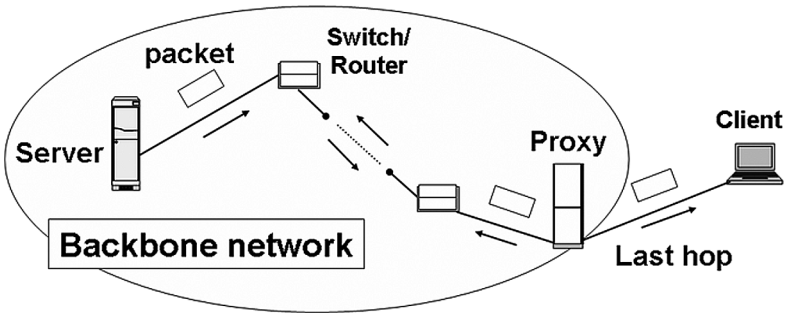


Fig. 1. Streaming on demand to a client through a proxy server.

A related work [28] examines the trade-offs between prefix/partial caching and full caching in the presence of constraints on the backbone network bandwidth and the available cache space. The authors propose streaming strategies that maximize the revenue rate of the service provider given these constraints.

Monitoring agents are network proxies located at the edge of the backbone network that send information back to the media server. Using this knowledge the server can better estimate the network state and adapt the media content sent to the client. *Statistical* feedback from monitoring agents was used in [29]–[31] while both *statistical* and *timely* feedback was used in [32], [33]. Statistical feedback, e.g., conventional RTCP reports [34], contains information such as the mean packet loss rate and the mean and variance of the round-trip time (RTT) over a window of packets. Timely feedback, such as positive and/or negative acknowledgements, indicates individual packets that have or have not arrived at the monitoring agent correctly and on time. In a related body of works [35]–[38], a network proxy in conjunction with timely feedback is employed to improve TCP performance over wireless networks. The presented simulation results show improved performance in terms of average throughput and link utilization. In addition, the proposed techniques allow for a significant reduction of the initial buffering delay and the client buffer size.

In this paper, we introduce a novel streaming system, called *RaDiO Edge*, centered around a proxy server equipped with a RaDiO packet scheduling procedure and a hybrid receiver/sender driven transmission scheme, here called *proxy-driven* transmission. The optimization procedure is based on the Iterative Sensitivity Adjustment (ISA) algorithm introduced in [1]–[3], modified here to suit the settings of the problem under consideration. Using the optimization procedure, rather than streaming the packetized media in a fixed sequence according to presentation time, the proxy can choose a transmission schedule for each data unit that minimizes the expected end-to-end distortion of the entire presentation subject to a transmission rate constraint. Such a rate-distortion optimized transmission schedule, or transmission policy,[3] results in unequal error protection provided to different portions of the media stream. The core step of the optimization procedure involves trading off the expected redundancy (the cost used to communicate the packet) for the probability that a single media packet will be communicated in error. The solution to this resource allocation problem is obtained by minimizing a Lagrangian, taking into account the feedback received from both the media server and the client, and the data units' dependence relationships, delivery deadlines and basic importances.

In summary, the major contributions of the present paper are the introduction of a novel hybrid receiver/sender-driven transmission scheme for proxy-driven streaming from the network edge,[4] and the derivation of an algorithm for computing rate-distortion optimized packet schedules for the scenario of proxy-driven streaming. Parts of this work have been presented in [39] and [40].

We present the major ideas in our paper as follows. In Section II, we define our abstractions of the encoding, packetization, and communication processes. In Section III, we present the proxy-driven transmission scheme employed by the proxy to communicate with the media server and with the client. In Section IV, we show how the entire multimedia presentation can be transmitted in a distortion-rate optimized way, using as a building block an algorithm for distortion-rate optimized transmission of a single media packet. This algorithm is the subject of Section V. In Section VI, we report experimental results. Using simulations the end-to-end distortion-rate performance of our framework for streaming of an entire media presentation is closely examined. Performance improvement is observed over rate-distortion optimized sender-driven systems for streaming video presentations. Finally, in Section VII, we provide concluding remarks.

II. Source and Channel Characterizations

A. Media Source Model

In a streaming media system, the encoded data are packetized into *data units* that are stored afterwards in a file on a media server. All of the data units in the presentation have interdependencies, which can be expressed by a directed acyclic graph. Associated with each data unit l is a size B_l, a decoding time $t_{DTS,l}$, and an importance Δd_l. The size B_l is the size of the data unit in bytes. $t_{DTS,l}$ is the *delivery deadline* (the decoder

[3]In this paper, a *transmission policy* is a finite state algorithm for transmitting data, following the terminology of Markov Decision Processes, in which a *policy* is a mapping of each state (in a finite state machine) to an action, which together with the state and the next observation, determines the next state.

[4]Contrary to all prior schemes where a proxy was involved, it is the proxy here that controls the communication between the sender and the receiver. The purpose of our proxy is not to reduce access latency (as with the caching proxies addressed in the references cited thus far), but to increase throughput by separating the transmission processes on the backbone and the last hop, thereby isolating the packet loss effects.

timestamp in MPEG terminology) by which data unit l must arrive at the client, or be too late to be usefully decoded. Packets containing data units that arrive after the data units' delivery deadlines are discarded. The importance Δd_l is the amount by which the distortion at the client will *decrease* if the data unit arrives on time at the client and is decoded.

B. Packet Loss Probabilities

We consider a system in which the media server communicates to the client indirectly, through a proxy. We refer to the server $\rightarrow$ proxy path as Channel 1 and to the proxy $\rightarrow$ client path as Channel 2. Each channel has a forward and backward direction. We model each direction of each channel as a time-invariant packet erasure channel with random delays. For the forward direction of Channel 1, this means that if the media server inserts a data packet into the network at time t, then the packet is lost with some probability, say ϵ_{F_1}, independently of t. However, if the packet is not lost, then it arrives at the proxy at some later time t', where the forward trip time $\mathrm{FTT}_1 = t' - t$ is randomly drawn according to a probability density p_{F_1}. The backward direction of Channel 1 is similarly characterized by the probability of packet loss ϵ_{B_1} and delay density p_{B_1}. Successive losses and delays, as well as losses and delay in forward and backward channels are assumed to be statistically independent.[5] Then, these induce the probability $\epsilon_{R_1} = 1 - (1 - \epsilon_{F_1})(1 - \epsilon_{B_1})$ of losing a packet in either the forward or backward direction, and the RTT distribution $P\{\mathrm{RTT}_1 > \tau\} = \epsilon_{R_1} + (1 - \epsilon_{R_1}) \int_\tau^\infty p_{R_1}(t)dt$, where $p_{R_1} = p_{F_1} * p_{B_1}$ is the convolution of p_{F_1} and p_{B_1}. Note that $P\{\mathrm{RTT}_1 > \tau\}$ is the probability that a data packet requested by the proxy at time t from the media server does not arrive at the proxy by time $t + \tau$.

Each direction of Channel 2 is also modeled as an independent time-invariant packet erasure channel with random delays. Hence, the forward and the backward directions are characterized by random loss and delay densities ϵ_{F_2}, p_{F_2} and ϵ_{B_2}, p_{B_2}, respectively. We let $P\{\mathrm{FTT}_2 > \tau\} = \epsilon_{F_2} + (1 - \epsilon_{F_2}) \int_\tau^\infty p_{F_2}(t)dt$ denote the probability that a packet transmitted in the forward direction of Channel 2 at time t does not arrive at the client by time $t + \tau$, whether it is lost, or simply delayed by more than τ on the last hop. Finally, these in turn induce the probability ϵ_{R_2} of losing a packet in either the forward or backward direction of Channel 2 and the RTT density p_{R_2}. Note that $P\{\mathrm{RTT}_2 > \tau\}$ is the probability that the proxy does not receive an acknowledgement packet by time $t + \tau$ for a data packet sent to the client at time t.

III. Proxy-Driven Transmission

A multimedia session starts when a client requests a presentation from the media server. The request packet is received by the proxy server and is not forwarded further. The proxy then sends a request to the media server for a rate-distortion preamble for the desired presentation. The preamble contains in effect the size, importance, and deadline information for each data unit. In addition, it also contains the directed acyclic graph describing the dependencies between the data units as explained earlier.

After it has the preamble, the proxy starts communicating simultaneously with the media server and the client using a hybrid receiver/sender-driven transmission scheme. The communication with the media server is receiver-driven, where the proxy sends request packets to the media server, requesting a particular data unit to be transmitted. The media server responds to a request by sending the requested data unit in a data packet to the proxy. The proxy may send requests for a particular data unit periodically if necessary, until the data unit finally arrives at the proxy, or until the proxy gives up. Upon seeing an arriving data packet, the proxy stores the corresponding data unit in its buffer for later (re)transmissions to the client and stops requesting that data unit from the media server. The communication with the client is sender-driven, where the proxy now sends to the client the buffered data unit periodically if necessary, until it receives an acknowledgement packet from the client for that data unit, or until the proxy gives up transmitting the data unit. In this paper, we call this hybrid receiver/sender-driven transmission scheme *proxy-driven*.

IV. Rate-Distortion Optimization Using Iterative Sensitivity Adjustments

Suppose there are L data units in the multimedia presentation. Let π_l be the transmission policy for data unit $l \in \{1, \ldots, L\}$ and let $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_L)$ be the vector of transmission policies for all L data units. A policy for a data unit is a schedule according to which the proxy sends request packets to the media server for this data unit or sends packets with the data unit to the client, in case a packet with the data unit has already arrived at the proxy.[6] Any given policy vector $\boldsymbol{\pi}$ induces, for the media presentation, an expected distortion $D(\boldsymbol{\pi})$ and expected transmission rates $R_1(\boldsymbol{\pi})$ and $R_2(\boldsymbol{\pi})$ on Channels 1 and 2, respectively, in the forward directions. We seek the policy vector $\boldsymbol{\pi}$ that minimizes $D(\boldsymbol{\pi})$ subject to a constraint on a cost-weighted combination of the expected transmission rates, $R(\boldsymbol{\pi}) = \gamma_1 R_1(\boldsymbol{\pi}) + \gamma_2 R_2(\boldsymbol{\pi})$, where γ_1 and γ_2 represent relative costs of transmitting over Channels 1 and 2. The optimal policy vector $\boldsymbol{\pi}$ can be found by minimizing the Lagrangian $D(\boldsymbol{\pi}) + \lambda R(\boldsymbol{\pi})$ for some Lagrange multiplier $\lambda > 0$, thus achieving a point on the lower convex hull of the set of all achievable (D, R) pairs.[7]

We now compute expressions for $R(\boldsymbol{\pi})$ and $D(\boldsymbol{\pi})$. The expected transmission rate $R(\boldsymbol{\pi})$ is the sum of the expected cost-weighted transmission rates for each data unit $l \in \{1, \ldots, L\}$

$$R(\boldsymbol{\pi}) = \sum_l B_l \rho(\pi_l) \tag{1}$$

where B_l is the number of bytes in data unit l and $\rho(\pi_l) = \gamma_1 \rho_1(\pi_l) + \gamma_2 \rho_2(\pi_l)$ is the expected cost of bytes transmitted over Channels 1 and 2 per source byte (under policy π_l), simply

[5]The assumption of iid packet loss and delay is for the purposes of tractability of our analysis, which requires independence of the packet transmissions related to any given data unit. This is justified provided that the retransmissions are infrequent (e.g., the retransmission intervals are larger than the one way delay), as noted for example by Bolot [41].

[6]Note that a request packet for data unit j and a data packet for data unit k can be sent simultaneously by the proxy on Channels 1 and 2, respectively.

[7]Since $\lambda\gamma_1$ and $\lambda\gamma_2$ may be regarded as separate Lagrange multipliers, this also achieves a point on the lower convex hull of the set of all achievable (D, R_1, R_2) triples.

called the *expected cost*. The expected distortion $D(\boldsymbol{\pi})$ is somewhat more complicated to express, but it can be expressed in terms of the probability $\epsilon(\pi_l)$ that data unit l does not arrive at the client on time (under policy π_l), called the *expected error*. Specifically, let I_l be the indicator random variable that is 1 if data unit l arrives at the client on time, and is 0 otherwise. Then $\prod_{l' \preceq l} I_{l'}$ is 1 if data unit l is *decodable* by the client on time, and is 0 otherwise. Here, $l' \preceq l$ means that l depends directly or indirectly on l'. If data unit l is decodable by the client on time, then the reconstruction error is reduced by the quantity Δd_l; otherwise the reconstruction error is not reduced. Hence, the total reduction in reconstruction error for the presentation is $\sum_l \Delta d_l \prod_{l' \preceq l} I_{l'}$. Subtracting this quantity from the reconstruction error for the presentation if no data units are received, and taking an expected value, gives for the expected distortion

$$D(\boldsymbol{\pi}) = D_0 - \sum_l \Delta d_l \prod_{l' \preceq l} (1 - \epsilon(\pi_{l'})) \tag{2}$$

where D_0 is the expected reconstruction error for the presentation if no data units are received and $\Delta d_l \prod_{l' \prec l}(1 - \epsilon(\pi_{l'}))$ is the expected reduction in reconstruction error due to data unit l.

Note that the formulation in (2) assumes that the distortion contribution of each data unit is incremental, which is not necessarily true in general, in particular when error concealment is employed at the decoder. Still, the assumption approximates well real systems with error concealment for packet loss rates that are not excessive so that the distortion contribution of each lost packet is additive, as shown in [42]. Furthermore, for the purpose of our simulation experiments that will be presented later on we have concealed every missing video frame at the receiver with a uniform gray-level image.

Finding a policy vector $\boldsymbol{\pi}$ that minimizes the expected Lagrangian $J(\boldsymbol{\pi}) = D(\boldsymbol{\pi}) + \lambda R(\boldsymbol{\pi})$, for $\lambda > 0$, is difficult since the terms involving the individual policies π_l in $J(\boldsymbol{\pi})$ are not independent. Specifically, the expression for the expected distortion cannot be split into a sum of terms each depending on a single π_l. Therefore, we employ an iterative descent algorithm, called the Iterative Sensitivity Adjustment (ISA), in which we minimize the objective function $J(\pi_1, \ldots, \pi_L)$ one variable at a time while keeping the other variables constant, until convergence [1]–[3]. It can be shown that the optimal individual policies at iteration n, for $n = 1, 2, \ldots$, are given by

$$\pi_l^{(n)} = \arg\min_{\pi_l} S_l^{(n)} \epsilon(\pi_l) + \lambda B_l \rho(\pi_l) \tag{3}$$

where $S_l^{(n)} = \sum_{l' \succeq l} \Delta d_{l'} \prod_{l'' \preceq l' : l'' \neq l} (1 - \epsilon(\pi_{l''}^{(n)}))$ is the *sensitivity* to losing data unit l, i.e., the amount by which the expected distortion will increase if data unit l cannot be recovered at the client, given the current transmission policies for the other data units. Another interpretation of $S_l^{(n)}$ is as the partial derivative of (2) with respect to $\epsilon_l = \epsilon(\pi_l)$, evaluated at $\boldsymbol{\pi}^{(n)}$.

The minimization (3) is now simple, since each data unit l can be considered in isolation. Indeed the optimal transmission policy $\pi_l \in \Pi$ for data unit l minimizes the "per data unit" Lagrangian $J(\pi) = \epsilon(\pi) + \lambda' \rho(\pi)$, where $\lambda' = \lambda B_l / S_l^{(n)}$ and Π is the family of policies. Thus, to minimize (3) for any l and λ', it suffices to know the lower convex hull of the set of points in the so-called "error-cost" plane, $\{(\rho(\pi), \epsilon(\pi)) : \pi \in \Pi\}$, which we call the expected *error-cost* function. The error-cost function can be considered as a normalized distortion-rate function pertaining to the transmission of a single dimensionless data unit, and it depends only on the transmission scenario and the channel characteristics. In the next section, we show how to find the optimal policy π^* for the family of transmission policies Π corresponding to proxy-driven transmission.

Note that as the space of transmission policies and the domains of the functions $D(\boldsymbol{\pi})$ and $R(\boldsymbol{\pi})$ are not necessarily convex, the optimal solution for the vector of transmission policies $\boldsymbol{\pi}^*$ provided by the ISA algorithm is not necessarily globally optimal. In addition, the ISA algorithm is an iterative descent algorithm, and may find only a local minimum rather than a global minimum.

V. Finding the Optimal Policy

For requesting/transmitting a data unit, we assume that there are N discrete transmission opportunities $t_0, t_1, \ldots, t_{N-1}$ prior to the data unit's delivery deadline t_{DTS} at which the proxy server is allowed either to request the data unit from the media server or to (re)transmit it to the client, once the data unit has arrived at the proxy. The proxy need not transmit a packet at every transmission opportunity.

At each transmission opportunity t_i, $i = 0, 1, \ldots, N-1$, the proxy takes an action a_i, where $a_i = 1$ if the proxy sends a packet and $a_i = 0$ otherwise. Then, at the next transmission opportunity t_{i+1}, the proxy makes an observation o_i, where o_i summarizes the packets that arrived at the proxy during the interval $(t_i, t_{i+1}]$. In particular, $o_i = \emptyset$ means that no packets arrived, $o_i = \text{DAT}$ means that at least one data packet arrived from the sender (but no acknowledgement packets arrived from the client), and $o_i = \text{ACK}$ means that at least one acknowledgement packet arrived from the client, during the interval $(t_i, t_{i+1}]$. The history, or the sequence of action–observation pairs $(a_0, o_0) \circ (a_1, o_1) \circ \cdots \circ (a_i, o_i)$ leading up to time t_{i+1}, determines the state q_{i+1} at time t_{i+1}. Therefore, a state represents uniquely this sequence of action–observation pairs. If the final observation o_i is an ACK, then q_{i+1} is a final state. In addition, any state at time $t_N = t_{DTS}$ is a final state.

The proxy server computes the optimal policy π^* at every transmission opportunity t_i, where $\{(\rho(\pi), \epsilon(\pi)) : \pi \in \Pi\}$ is calculated conditioned on q_i and all the policies under consideration are consistent with the history $(a_0, o_0) \circ (a_1, o_1) \circ \cdots \circ (a_{i-1}, o_{i-1})$ leading up to state q_i at time t_i. Then, a_i is set to the first action $\pi^*(q_i)$ of π^*, and the procedure is repeated at each successive transmission opportunity until a final state is reached. Note that it would be sufficient to determine π^* only once at time t_0, except for the fact that λ' may be adjusted (by the ISA algorithm from Section IV) at subsequent transmission opportunities to take into account feedback from transmission of other data units.

Two cases of transmission history can be distinguished: 1) no previous DATs, i.e., $o_0 = o_1 = \cdots = o_{i-1} = \emptyset$, and 2) with previous DATs. In the following, we show how π^* can be determined for each of them.

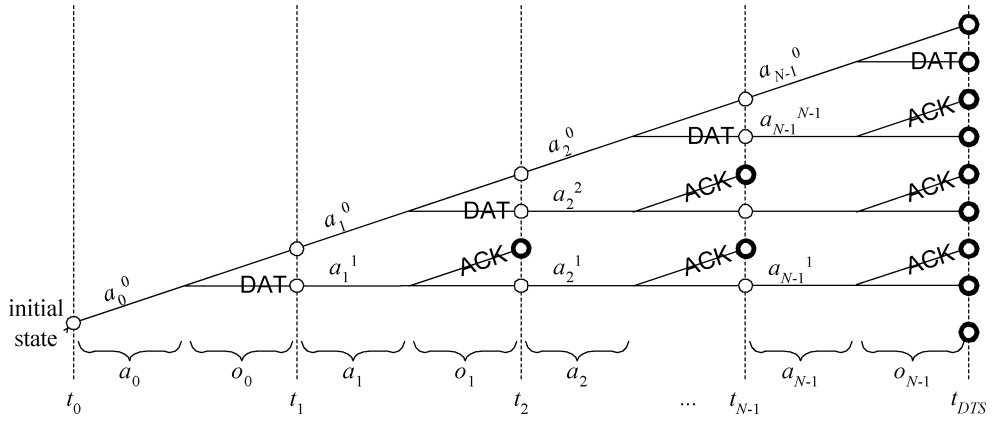


Fig. 2. The decision tree of a transmission policy for a data unit when no previous DATs have arrived at the proxy.

A. No Previous DATs

Fig. 2 shows the tree of action–observation pairs associated with a policy π for this case of transmission history. Final states in Fig. 2 are indicated by double circles. Starting from the initial state at time t_0, the proxy follows the upper path (the backbone of the tree), taking actions $a_0^0, a_1^0, a_2^0, \ldots$, i.e., sending request packets (or not) to the server, until it receives a data packet. Once the first data packet is received, say during the interval $(t_i, t_{i+1}]$, i.e., $o_i = \mathrm{DAT}$, the proxy begins to follow the corresponding horizontal branch from time t_{i+1}, taking actions $a_{i+1}^{i+1}, a_{i+2}^{i+1}, a_{i+3}^{i+1}, \ldots$, i.e., sending data packets (or not) to the client, until it receives an acknowledgement packet. The decision tree of any particular policy π can also be represented as an upper triangular matrix of actions:

$$\pi = \begin{bmatrix} a_0^0 & a_1^0 & a_2^0 & \cdots & a_{N-1}^0 \\ & a_1^1 & a_2^1 & \cdots & a_{N-1}^1 \\ & & a_2^2 & \cdots & a_{N-1}^2 \\ & & & \ddots & \vdots \\ & & & & a_{N-1}^{N-1} \end{bmatrix}.$$

Here, the first row $\boldsymbol{a}^0 = [a_0^0, a_1^0, a_2^0, \ldots, a_{N-1}^0]$ is the sequence of actions to take until the first DAT is received and each subsequent row $\boldsymbol{a}^{i+1} = [a_{i+1}^{i+1}, a_{i+2}^{i+1}, \ldots, a_{N-1}^{i+1}]$ is the sequence of actions to take until the first ACK is received.

Each such policy π induces a stochastic finite state automaton. Label the state preceding action a_i^k by q_i^k. Then the transition probabilities along the backbone are given by $P_\pi(q_{i+1}^0|q_i^0) = A(i)$ and $P_\pi(q_{i+1}^{i+1}|q_i^0) = 1 - A(i)$, where

$$A(i) = \prod_{j \le i : a_j^0 = 1} P\{\mathrm{RTT}_1 > t_{i+1} - t_j | \mathrm{RTT}_1 > t_i - t_j\}.$$

Note that these transition probabilities depend on $\boldsymbol{a}^0$ only. Recall that the transmission actions $\boldsymbol{a}^0$ exclusively refer to sending request packets to the server.

Using the transition probabilities, the expected error $\epsilon(\pi|q_i^0)$ of policy π starting from the backbone state q_i^0 can be computed recursively for $i = N-1, N-2, \ldots, 0$ as

$$\begin{aligned} \epsilon\left(\pi|q_N^0\right) &= 1 \\ \epsilon\left(\pi|q_i^0\right) &= A(i)\epsilon\left(\pi|q_{i+1}^0\right) + (1 - A(i))\,\epsilon\left(\pi|q_{i+1}^{i+1}\right) \end{aligned}$$

where

$$\epsilon\left(\pi|q_{i+1}^{i+1}\right) = \prod_{j > i : a_j^{i+1} = 1} P\{\mathrm{FTT}_2 > t_{DTS} - t_j\} \qquad (4)$$

is the expected error of policy π starting from branch state q_{i+1}^{i+1}. Note that (4) depends only on $\boldsymbol{a}^{i+1}$. Recall that the transmission actions $\boldsymbol{a}^{i+1}$ exclusively refer to sending data packets to the client.

Similarly, the expected cost $\rho(\pi|q_i^0)$ of policy π starting from the backbone state q_i^0 can be computed recursively. As introduced in Section IV, let γ_2 be the cost per source byte of transmitting a data packet from the proxy to the client downstream on Channel 2, and let γ_1 be the cost per source byte of transmitting a data packet from the server to the proxy downstream on Channel 1, so that $\gamma_1(1 - \epsilon_{B_1})$ is the expected cost per source byte of transmitting a request packet from the proxy to the server upstream on Channel 1. Then the expected cost of policy π starting from backbone state q_i^0 can be computed recursively for $i = N-1, N-2, \ldots, 0$ as

$$\begin{aligned} \rho\left(\pi|q_N^0\right) &= 0 \\ \rho\left(\pi|q_i^0\right) &= \gamma_1\left(1 - \epsilon_{B_1}\right) a_i^0 + A(i)\rho\left(\pi|q_{i+1}^0\right) \\ &\quad + (1 - A(i))\,\rho\left(\pi|q_{i+1}^{i+1}\right) \end{aligned}$$

where

$$\rho\left(\pi|q_{i+1}^{i+1}\right) = \gamma_2 \sum_{j > i : a_j^{i+1} = 1} \prod_{k \le j : a_k^{i+1} = 1} P\{\mathrm{RTT}_2 > t_j - t_k\} \qquad (5)$$

is the expected cost of policy π starting from branch state q_{i+1}^{i+1}. Note that (5) depends only on $\boldsymbol{a}^{i+1}$.

Finally, the expected Lagrangian of policy π starting from state q can be expressed $J(\pi|q) = \epsilon(\pi|q) + \lambda'\rho(\pi|q)$.

By expanding the above recursive expressions, the expected error, cost, and Lagrangian of policy π can also be expressed nonrecursively as

$$\epsilon\left(\pi|q_0^0\right) = \sum_{i=0}^{N-1}\left(\left(\prod_{j=0}^{i-1}A(j)\right)(1-A(i))\,\epsilon\left(\pi|q_{i+1}^{i+1}\right)\right) + \prod_{j=0}^{N-1}A(j)$$

$$\rho\left(\pi|q_0^0\right) = \sum_{i=0}^{N-1}\left(\left(\prod_{j=0}^{i-1}A(j)\right)(1-A(i))\,\rho\left(\pi|q_{i+1}^{i+1}\right)\right) + \sum_{i=0}^{N-1}\left(\left(\prod_{j=0}^{i-1}A(j)\right)\gamma_1\left(1-\epsilon_{B_1}\right)a_i^0\right)$$

$$J\left(\pi|q_0^0\right) = \sum_{i=0}^{N-1}\left(\left(\prod_{j=0}^{i-1}A(j)\right)(1-A(i))\,J\left(\pi|q_{i+1}^{i+1}\right)\right) + \prod_{j=0}^{N-1}A(j) + \lambda'\sum_{i=0}^{N-1}\left(\prod_{j=0}^{i-1}A(j)\right)\gamma_1(1-\epsilon_{B_1})a_i^0.$$

Note that the above expression for the expected cost can be easily factored into the form $\gamma_1\rho_1(\pi)+\gamma_2\rho_2(\pi)$ that was introduced in Section IV.

Thus, minimizing the Lagrangian over all policies π can be expressed

$$\min_{\pi} J\left(\pi|q_0^0\right) = \min_{\boldsymbol{a}^0}\left\{\sum_{i=0}^{N-1}\left(\left(\prod_{j=0}^{i-1}A(j)\right)(1-A(i))\,J^*\left(q_{i+1}^{i+1}\right)\right) + \prod_{j=0}^{N-1}A(j) + \lambda'\sum_{i=0}^{N-1}\left(\prod_{j=0}^{i-1}A(j)\right)\gamma_1(1-\epsilon_{B_1})\,a_i^0\right\},$$

where

$$J^*\left(q_{i+1}^{i+1}\right) = \min_{\boldsymbol{a}^{i+1}} J\left(\pi|q_{i+1}^{i+1}\right).$$

That is, minimizing the Lagrangian over π can be accomplished by first independently minimizing the Lagrangian $J(\pi|q_{i+1}^{i+1})$ over the sequence of actions $\boldsymbol{a}^{i+1}$, for $i=0,1,\ldots,N-1$, and then minimizing $J(\pi|q_0^0)$ over $\boldsymbol{a}^0$. In this way the optimal policy π^* can be computed for each λ'. Note that at times t_i, for $i>0$, the proxy needs to recompute only the rows $\boldsymbol{a}^{i+1}$ of the matrix representation of π^* and the entries $j\geq i$ of the first row $\boldsymbol{a}^0$.

B. With Previous DATs

Here, the data unit has already arrived at the proxy. Assume that the first data packet is received at the proxy by time t_i, for $i=1,2,\ldots,N-1$. Then, at every t_j, for $j\geq i$, the proxy needs to determine only the elements $a_j^i, a_{j+1}^i,\ldots,a_{N-1}^i$ of the row $\boldsymbol{a}^i$ that minimize the Lagrangian $J(\pi|q_j^i)$.

To account for previous transmissions, if any, of the data unit on the last hop due to actions $a_i^i, a_{i+1}^i,\ldots,a_{j-1}^i$, we adjust the expressions for the expected error and cost, given in (4) and (5), respectively, as follows:

$$\epsilon\left(\pi|q_j^i\right) = \prod_{l\geq i:a_l^i=1} P\{\mathrm{FTT}_2 > t_{DTS}-t_l \,|\, \mathrm{RTT}_2 > t_j-t_l\} \tag{6}$$

$$\rho\left(\pi|q_j^i\right) = \gamma_2\sum_{l>j:a_l^i=1}\ \prod_{k\leq l:a_k^i=1} P\{\mathrm{RTT}_2 > t_l-t_k \,|\, \mathrm{RTT}_2 > t_j-t_k\}. \tag{7}$$

Note that the conditional probabilities in (6) and (7) become unconditional for transmission actions $a_j^i, a_{j+1}^i,\ldots,a_{N-1}^i$. Finally, as stated above, we are interested in the sequence of actions $[a_j^i, a_{j+1}^i,\ldots,a_{N-1}^i]^*$ that minimizes the Lagrangian $J(\pi|q_j^i)=\epsilon(\pi|q_j^i)+\lambda'\rho(\pi|q_j^i)$, i.e.,

$$\left[a_j^i, a_{j+1}^i,\ldots,a_{N-1}^i\right]^* = \arg\min_{a_j^i,a_{j+1}^i,\ldots,a_{N-1}^i} J\left(\pi|q_j^i\right).$$

C. Computational Complexity

In this section, we describe briefly the computational requirements per data unit of the optimization framework for proxy-driven streaming presented in this paper. As argued in [43], the computational complexity of a conventional RaDiO system, for sender-driven or receiver-driven streaming, is on the order of 2^N operations per data unit, where N is the number of transmission opportunities associated with a data unit, as explained earlier.

Now, in the first case of transmission history discussed in Section V-A, a proxy-driven system computes a matrix of transmission actions π. The actions from the first row of π correspond to receiver-driven streaming that the proxy performs with the media server requesting transmission of data units from the server. Therefore, the computational complexity of finding the optimal set of actions for the first row of π is 2^N. Next, for each of the subsequent rows of π the proxy performs sender-driven streaming with the client, in which the proxy transmits data units from its buffer to the client. Therefore, the computational complexity of finding the optimal set of actions for rows $2,3,\ldots,N$ of π is 2^n, for $n=N-1,N-2,\ldots,1$, respectively. Note that due to the design of the algorithm employed for proxy-driven packet scheduling, the optimal actions associated with each row of π can be computed independently, as explained in Section V-A. Finally, the overall computational complexity of a proxy-driven system when computing the optimal transmission schedule π^* for this case of transmission history is simply the sum of the corresponding complexities associated with each row of π, i.e., $\sum_{n=1}^{N}2^n = 2^{N+1}$. Hence, relative to a conventional RaDiO system employing the same number of transmission opportunities N, we observe that twice as much computational complexity is required to perform the packet scheduling at an intermediate proxy server.

On the other hand, for the second case of transmission history discussed in Section V-B, the proxy-driven system essentially performs sender-driven streaming of a data unit to the client.

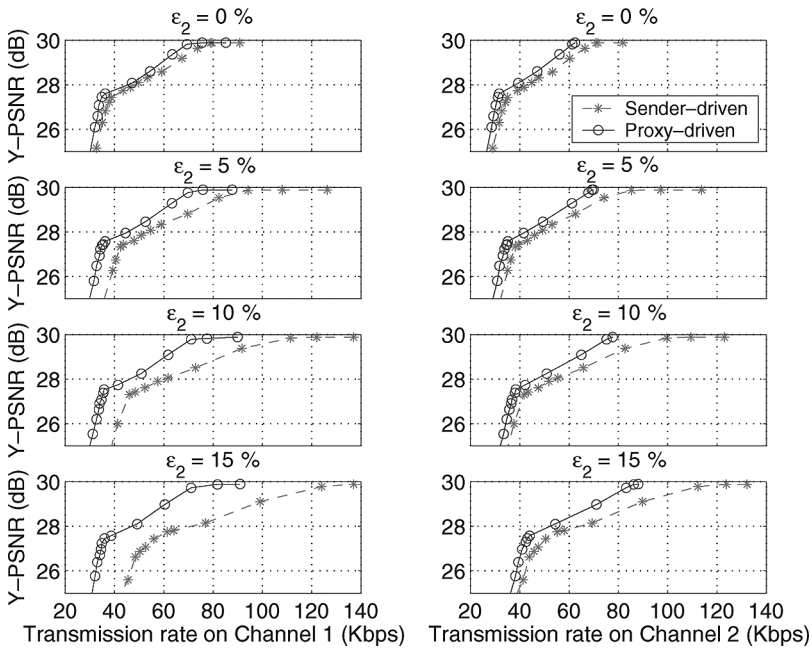


Fig. 3. Proxy-driven versus sender-driven performance for *Foreman*: Y-PSNR (dB) versus transmission rate (kb/s) in the forward direction on Channel 1, and on Channel 2, for $\epsilon_2 = 0\%$, 5%, 10%, and 15%, and $\epsilon_1 = 10\%$.

Therefore, the associated computational complexity is on the order of 2^n, where n is the number of transmission opportunities left before the delivery deadline for the data unit. Hence, in this case there is no increase in complexity relative to conventional RaDiO streaming.

Finally, it should be mentioned that the computational requirements for proxy-driven streaming described above increase linearly with the number of streams that the proxy is required to handle using the proposed optimization framework. Therefore, that may put a limit on the number of streams that can be simultaneously processed, depending on the available computational power at the proxy server.

VI. Experimental Results

Here, we investigate the (D, R_1, R_2) distortion-rate performance for streaming packetized video contents using different algorithms. The videos are two-layer SNR scalable representations of the sequences *Foreman* and *Mother and Daughter*, henceforth denoted *MaD*. Using an H.263+ [44] codec, 130 frames of QCIF *Foreman* have been encoded into a base layer and an enhancement layer with corresponding rates of 32 and 64 kb/s. Similarly, 130 frames of QCIF *MaD* have been encoded into two layers with rates 32 and 69 kb/s, respectively. For both videos, the frame rate is 10 fps and the size of the Group of Pictures (GOP) is 10 frames, consisting of an I frame followed by nine consecutive P frames. Two RaDiO streaming systems are employed in the experiments. *Sender-driven* is a system that performs RaDiO scheduling of the packet transmissions at the media server [1]–[3]. *Proxy-driven* is the system presented in this work, which also performs RaDiO packet scheduling, but at the network edge using a proxy server. The Lagrange multiplier λ is fixed for the entire presentation for both systems. Performance is measured in terms of the luminance peak signal-to-noise ratio (Y-PSNR) in dB of the mean squared error distortion, averaged over the duration of the video clip, as a function of the average transmission rates in kilobits per second (kb/s) on Channels 1 and 2. In the experiments, we use $T = 100$ ms as the time interval between transmission opportunities and 600 ms for the playback delay. The proxy-driven system is designed with $\gamma_1 = \gamma_2 = 1$.

Channel 1 is specified as follows. Packets transmitted on this channel are dropped at random, with a drop rate $\epsilon_{F_1} = \epsilon_{B_1} = \epsilon_1$. Those packets that are not dropped receive a random delay, where for the forward and backward delay densities p_{F_1} and p_{B_1} we use identical shifted Gamma distributions with parameters (n_1, α_1) and right shift κ_1, where $n_1 = 1$ node, $1/\alpha_1 = 25$ ms, and $\kappa_1 = 25$ ms for a mean delay of $\kappa_1 + n_1/\alpha_1 = 50$ ms and standard deviation $\sqrt{n_1}/\alpha_1 = 25$ ms. Channel 2 is similarly specified with $\epsilon_{F_2} = \epsilon_{B_2} = \epsilon_2$, $n_2 = 1$ node, $1/\alpha_2 = 5$ ms, and $\kappa_2 = 5$ ms for a mean delay of $\kappa_2 + n_2/\alpha_2 = 10$ ms and standard deviation $\sqrt{n_2}/\alpha_2 = 5$ ms. Note that for *Sender-driven* the communication channel between the server and the client represents a concatenation of Channels 1 and 2. Therefore, the delay distribution for the concatenation is computed numerically as the convolution of the delay distributions for Channels 1 and 2.

First, we compare the efficiencies of proxy-driven and sender-driven systems in terms of the transmission rates in the forward directions of Channels 1 and 2, as the packet loss rate increases on Channel 2 and remains fixed on Channel 1. Fig. 3 shows the Y-PSNR of the *Foreman* sequence as a function of the transmission rates on Channels 1 and 2, respectively, when $\epsilon_2 = 0\%$, 5%, 10%, and 15%, while $\epsilon_1 = 10\%$. It can be seen that the proxy-driven system outperforms the sender-driven system in all cases over the whole range of rates. Further, Fig. 3 (left) shows that as Channel 2 degrades, the transmission rate on Channel 1 of the proxy-driven system remains approximately constant (for any given Y-PSNR), while the transmission rate on Channel 1 of the sender-driven system increases. Thus, the

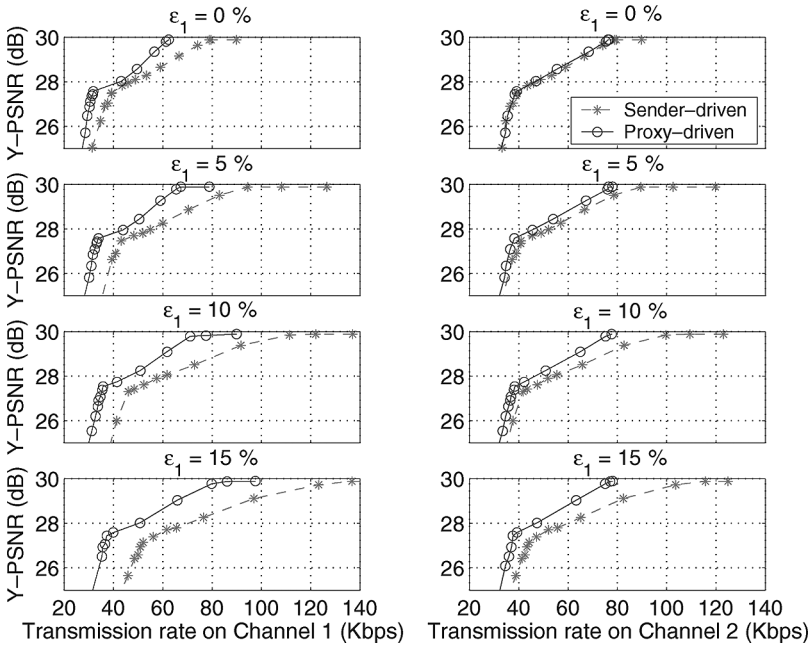


Fig. 4. Proxy-driven versus sender-driven performance for *Foreman*: Y-PSNR (dB) versus transmission rate (kb/s) in the forward direction on Channel 1, and on Channel 2, for $\epsilon_1 = 0\%$, 5%, 10%, and 15%, and $\epsilon_2 = 10\%$.

gap between them, which is essentially zero when $\epsilon_2 = 0\%$, increases as Channel 2 degrades. This is because as Channel 2 degrades, the sender-driven system must increasingly retransmit information over Channel 1, while the proxy-driven system does not need to retransmit information over Channel 1 to accommodate information lost on Channel 2. On the other hand, Fig. 3 (right) shows that the transmission rate on Channel 2 increases for both proxy-driven and sender-driven systems as Channel 2 degrades. However, even when $\epsilon_2 = 0\%$, there is a positive gap in transmission rate between the two systems. This is because the sender-driven system suffers an additional loss of acknowledgements in the backward direction of Channel 1. This shows up as additional rate in the forward direction of Channel 2 due to retransmissions. Indeed, the ratio of transmission rates forming this performance gap appears to remain approximately constant as Channel 2 degrades, indicating that this gap is due to losses in Channel 1 rather than in Channel 2.

Next we compare the efficiencies of proxy-driven and sender-driven systems, as packet loss increases on Channel 1 and remains fixed on Channel 2. Fig. 4 shows the Y-PSNR of the *Foreman* sequence as a function of the transmission rates on Channels 1 and 2, respectively, when $\epsilon_1 = 0\%$, 5%, 10%, and 15%, while $\epsilon_2 = 10\%$. As in the previous figure, it can be seen that the proxy-driven system outperforms the sender-driven system in all cases over the whole range of rates. Further, Fig. 4 (right) shows that as Channel 1 degrades, the transmission rate on Channel 2 of the proxy-driven system remains approximately constant (for any given Y-PSNR), while the transmission rate on Channel 2 of the sender-driven system increases. Thus, the gap between them, which is essentially zero when $\epsilon_1 = 0\%$, increases as Channel 1 degrades. This is because as Channel 1 degrades, the sender-driven system must increasingly retransmit information over Channels 1 and 2, while the proxy-driven system does not need to retransmit information over Channel 2 to accommodate information lost on Channel 1. On the other hand, Fig. 4 (left) shows that the transmission rate on Channel 1 increases for both proxy-driven and sender-driven systems as Channel 1 degrades. However, even when $\epsilon_1 = 0\%$, there is a positive gap in transmission rate between the two systems. This is because the sender-driven system suffers additional loss in both the forward and backward directions of Channel 2. This shows up as additional rate in the forward direction of Channel 1 due to retransmissions.

We find that the performance of both proxy-driven and sender-driven systems on both Channels 1 and 2 can be accurately predicted from the distortion-rate function of the source and the values of ϵ_1 and ϵ_2. When the number of retransmission opportunities before the playout deadline is sufficiently large, both the proxy-driven and sender-driven systems transmit close to channel capacity. As is well known [45], the capacity of an erasure channel with erasure probability ϵ is $1-\epsilon$. Thus, an optimal system transmits $1/(1-\epsilon)$ channel packets for every data unit. In particular, since our sender-driven system continues to transmit packets until it receives an acknowledgement, which it receives with probability $(1-\epsilon_{F1})(1-\epsilon_{F2})(1-\epsilon_{B2})(1-\epsilon_{B1})$ for each transmitted packet, the sender-driven system transmits on average $1/[(1-\epsilon_{F1})(1-\epsilon_{F2})(1-\epsilon_{B2})(1-\epsilon_{B1})]$ packets per data unit over Channel 1. Only fraction $(1-\epsilon_{F1})$ of these survive to cross Channel 2, so the average rate on Channel 2 is $1/[(1-\epsilon_{F2})(1-\epsilon_{B2})(1-\epsilon_{B1})]$ packets per data unit. Similarly, the proxy-driven system transmits on average $1/[(1-\epsilon_{B1})(1-\epsilon_{F1})]$ requests per data unit in the backward direction of Channel 1. Only fraction $(1-\epsilon_{B1})$ of these reach the server, so the proxy-driven system transmits on average $1/(1-\epsilon_{F1})$ packets per data unit in the forward direction of Channel 1. Likewise, the proxy-driven system transmits on average $1/[(1-\epsilon_{F2})(1-\epsilon_{B2})]$ packets per data unit in the forward direction of Channel 2. These factors are summarized

TABLE I
NUMBER OF PACKETS (PER DATA UNIT) TRANSMITTED OVER CHANNELS 1 AND 2 FOR SENDER-DRIVEN AND PROXY-DRIVEN SYSTEMS

Packets per DU	Channel F1	Channel F2
Sender-driven	$1/[(1-\epsilon_{F1})(1-\epsilon_{F2})(1-\epsilon_{B2})(1-\epsilon_{B1})]$	$1/[(1-\epsilon_{F2})(1-\epsilon_{B2})(1-\epsilon_{B1})]$
Proxy-driven	$1/(1-\epsilon_{F1})$	$1/[(1-\epsilon_{F2})(1-\epsilon_{B2})]$
Ratio	$1/[(1-\epsilon_{F2})(1-\epsilon_{B2})(1-\epsilon_{B1})]$	$1/(1-\epsilon_{B1})$

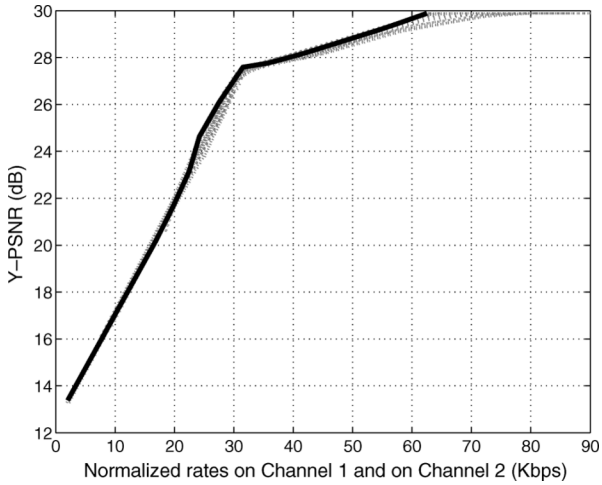


Fig. 5. Y-PSNR (dB) of the *Foreman* sequence versus normalized rate (kb/s) on Channels 1 and 2.

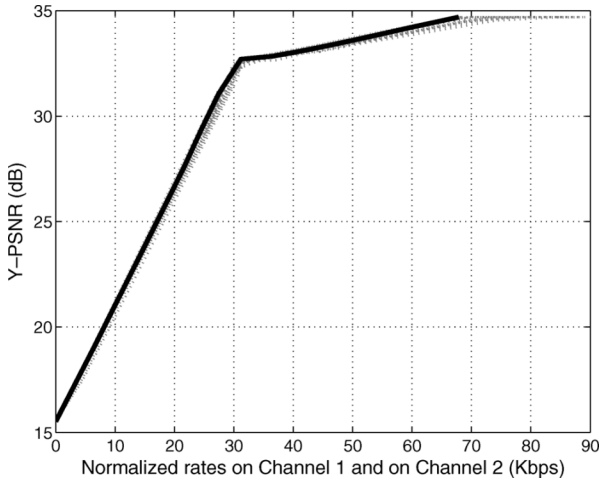


Fig. 6. Y-PSNR (dB) of the *MaD* sequence versus normalized rate (kb/s) on Channels 1 and 2.

in Table I. To confirm that these factors accurately predict performance, we normalize the transmission rates for all the graphs in Figs. 3 and 4 by the appropriate factors in the table (for example, we divide all the proxy-driven rates for Channel 1 by $1/(1-\epsilon_{F1})$) and we plot them (in gray) in Fig. 5, along with the distortion-rate function of the source (in bold). It can be seen that all graphs lie essentially on top of the distortion-rate function for the source, meaning that we can effectively synthesize the graphs in Figs. 3 and 4 by multiplying the rate-distortion function by the factors in Table I. Fig. 6 shows even tighter correspondence for the *Mother and Daughter* sequence.

VII. CONCLUSION

A system for RaDiO streaming to clients over lossy packet networks has been presented. Rather than streaming the data directly from a media server to a client, we propose to have a mediator, a proxy server, between the media server and the client. The proxy is located at the edge of the backbone network and coordinates the streaming process in a RaDiO framework for packet scheduling. In experiments, we employ our system for streaming video sequences and compare its performance to that of a sender-driven RaDiO system. The results demonstrate that the proposed proxy-driven system performs favorably over a large range of conditions. In particular, if the backbone is a more expensive resource than the last hop (e.g., as in WiFi home networking or high speed local area networking), then the proxy-driven system outperforms the sender-driven system by a factor of $1/[(1-\epsilon_{F2})(1-\epsilon_{B2})(1-\epsilon_{B1})]$. Thus, the gain is due to losses anywhere in the network except in the forward direction of the last hop, which penalizes both systems equally. On the other hand, if the last hop is a more expensive resource than the backbone (e.g., as in cellular telephony), then the proxy-driven system outperforms the sender-driven system by a factor of $1/(1-\epsilon_{B1})$. In this case the gain is entirely due to losses in the backward direction in the backbone. As this loss can be made negligible by rich acknowledgements [46], [47], we conclude that proxy-driven streaming is most useful in scenarios where communication near the edge is cheap relative to communication through the backbone.

REFERENCES

[1] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 390–404, Apr. 2006.

[2] ——, "Rate-distortion optimized streaming of packetized media," Microsoft Research, Redmond, WA, Tech. Rep. MSR-TR-2001-35, Feb. 2001.

[3] P. A. Chou and Z. Miao, "Rate-distortion optimized sender-driven streaming over best-effort networks," in *Proc. IEEE Workshop on Multimedia Signal Processing*, Cannes, France, Oct. 2001, pp. 587–592.

[4] P. A. Chou and A. Sehgal, "Rate-distortion optimized receiver-driven streaming over best-effort networks," presented at the Int. Packet Video Workshop, Pittsburg, PA, Apr. 2002.

[5] J. Chakareski, P. Chou, and B. Aazhang, "Computing rate-distortion optimized policies for streaming media to wireless clients," in *Proc. Data Compression Conf. (DCC'2002)*, Snowbird, UT, Apr. 2002, pp. 53–62.

[6] J. Chakareski and P. Chou, "Application layer error correction coding for rate-distortion optimized streaming to wireless clients," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'02)*, Orlando, FL, May 2002, vol. 3, pp. 2513–2516.

[7] ——, "Application layer error correction coding for rate-distortion optimized streaming to wireless clients," *IEEE Trans. Commun.*, vol. 52, no. 10, pp. 1675–1687, Oct. 2004.

[8] A. Sehgal and P. A. Chou, "Cost-distortion optimized caching of streaming media," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'02)*, Orlando, FL, May 2002, vol. 2, pp. 1973–1976.

[9] Z. Miao and A. Ortega, "Scalable proxy caching of video under storage constraints," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 7, pp. 1315–1327, Sep. 2002.

[10] J. Rexford and D. Towsley, "Smoothing variable-bit-rate video in an Internetwork," *IEEE/ACM Trans. Netw.*, vol. 7, no. 2, pp. 202–215, Apr. 1999.

[11] R. Rejaie, H. Yu, M. Handely, and D. Estrin, "Multimedia proxy caching mechanism for quality adaptive streaming applications in the Internet," in *Proc. IEEE INFOCOM 2000*, Tel-Aviv, Israel, Mar. 2000, vol. 2, pp. 980–989.

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

[13] K.-L. Wu, P. Yu, and J. Wolf, "Segment-based proxy caching of multimedia streams," in *Proc. ACM Int. Conf. World Wide Web*, Hong Kong, Apr. 2001, pp. 36–44.

[14] R. Rejaie and J. Kangasharju, "Mocha: a quality adaptive multimedia proxy cache for Internet streaming," in *Proc. ACM Int. Workshop on Network and Operating Systems Support for Digital Audio and Video*, Port Jefferson, NY, Jun. 2001, pp. 3–10.

[15] B. Wang, S. Sen, M. Adler, and D. Towsley, "Optimal proxy cache allocation for efficient streaming media distribution," in *Proc. IEEE INFOCOM 2002*, New York, Jun. 2002, vol. 3, pp. 1726–1735.

[16] Z. Antoniou and I. Stavrakakis, "An efficient deadline-credit-based transport scheme for prerecorded semisoft continuous media applications," *IEEE/ACM Trans. Netw.*, vol. 10, no. 5, pp. 630–643, Oct. 2002.

[17] F. Yu, Q. Zhang, W. Zhu, and Y.-Q. Zhang, "QoS-adaptive proxy caching for multimedia streaming over the Internet," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 3, pp. 257–269, Mar. 2003.

[18] H. Sun, W. Kwok, and J. Zdepski, "Architectures for MPEG compressed bitstream scaling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 2, pp. 191–199, Apr. 1996.

[19] P. A. A. Assuncao and M. Ghanbari, "Post-processing of MPEG-2 coded video for transmission at lower bit rates," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP'96)*, Atlanta, GA, May 1996, vol. 4, pp. 1998–2001.

[20] G. Keesman, R. Hellinghuizen, F. Hoeksema, and G. Heidman, "Transcoding of MPEG bitstreams," *Signal Process.: Image Commun.*, vol. 8, no. 6, pp. 481–500, Sep. 1996.

[21] N. Bjork and C. Christopoulos, "Transcoder architectures for video coding," *IEEE Trans. Consumer Electron.*, vol. 44, no. 1, pp. 88–98, Feb. 1998.

[22] T. Shanableh and M. Ghanbari, "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," *IEEE Trans. Multimedia*, vol. 2, no. 2, pp. 101–110, Jun. 2000.

[23] T. Warabino, S. Ota, D. Morikawa, M. Ohashi, H. Nakamura, H. Iwashita, and F. Watanabe, "Video transcoding proxy for 3G wireless mobile Internet access," *IEEE Commun. Mag.*, vol. 38, no. 10, pp. 66–71, Oct. 2000.

[24] G. de los Reyes, A. R. Reibman, S.-F. Chang, and J. C.-I. Chuang, "Error-resilient transcoding for video over wireless channels," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 6, pp. 1063–1074, Jun. 2000.

[25] S. Dogan, A. Cellatoglu, M. Uyguroglu, A. Sadka, and A. Kondoz, "Error-resilient video transcoding for robust Internetwork communications using GPRS," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 453–464, Jun. 2002.

[26] O. Verscheure, C. Venkatramani, P. Frossard, and L. Amini, "Joint server scheduling and proxy caching for video delivery," *Comput. Commun.*, vol. 25, no. 4, pp. 413–423, Mar. 2002.

[27] C. Venkatramani, O. Verscheure, P. Frossard, and K.-W. Lee, "Optimal proxy management for multimedia streaming in content distribution networks," in *Proc. ACM Int. Workshop on Network and Operating Systems Support for Digital Audio and Video*, Miami, Florida, May 2002, pp. 147–154.

[28] J. Kangasharju, F. Hartanto, M. Reisslein, and K. Ross, "Distributing layered encoded video through caches," *IEEE Trans. Comput.*, vol. 51, no. 6, pp. 622–636, Jun. 2002.

[29] T. Yoshimura, T. Kawahara, T. Ohya, and M. Etoh, "QoS control architecture with RTP monitoring agent for mobile multimedia streaming," presented at the IPSJ Symp. Multimedia, Distributed, Cooperative and Mobile Systems (DICOMO), Jun. 2001.

[30] T. Yoshimura, T. Ohya, T. Kawahara, and M. Etoh, "Rate and robustness control with RTP monitoring agent for mobile multimedia streaming," in *Proc. IEEE Int. Conf. Communications*, New York, Apr. 2002, vol. 4, pp. 2513–2517.

[31] T. Yoshimura, Y. Yonemoto, T. Ohya, M. Etoh, and S. Wee, "Mobile streaming media CDN enabled by dynamic SMIL," in *Proc. ACM Int. Conf. World Wide Web*, Honolulu, HI, May 2002, pp. 651–661.

[32] G. Cheung and T. Yoshimura, "Streaming agent: a network proxy for media streaming in 3G wireless networks," presented at the IEEE Int. Packet Video Workshop, Pittsburgh, PA, Apr. 2002.

[33] G. Cheung, W.-T. Tan, and T. Yoshimura, "Rate-distortion optimized application-level retransmission using streaming agent for video streaming over 3G wireless network," in *Proc. IEEE Int. Conf. Image Processing*, Rochester, NY, Sep. 2002, vol. 1, pp. 529–532.

[34] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," IETF, RFC 1889, Jan. 1998 [Online]. Available: http://www.ietf.org/rfc/rfc1889.txt

[35] H. Balakrishnan, S. Seshan, and R. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *ACM Wireless Netw.*, vol. 1, no. 5, pp. 469–481, Dec. 1995.

[36] S. Hadjiefthymiades, S. Papayiannis, and L. Merakos, "Using path prediction to improve TCP performance in wireless/mobile communications," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 54–61, Aug. 2002.

[37] L. Huang, U. Horn, F. Hartung, and M. Kampmann, "Proxy-based TCP-friendly streaming over mobile networks," in *Proc. ACM Int. Workshop on Wireless Mobile Multimedia*, Atlanta, GA, Sep. 2002, pp. 17–24.

[38] J.-H. Hu, G. Feng, and K. Yeung, "Hierarchical cache design for enhancing TCP over heterogeneous networks with wired and wireless links," *IEEE Trans. Wireless Commun.*, vol. 2, no. 2, pp. 205–217, Mar. 2003.

[39] J. Chakareski, P. Chou, and B. Girod, "Rate-distortion optimized streaming from the edge of the network," in *Proc. IEEE Workshop on Multimedia Signal Processing*, St. Thomas, U.S. Virgin Islands, Dec. 2002, pp. 49–52.

[40] ——, "Computing rate-distortion optimized policies for hybrid receiver/sender driven streaming of multimedia," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, CA, Nov. 2002, vol. 2, pp. 1310–1314.

[41] J.-C. Bolot, "End-to-end packet delay and loss behavior in the Internet," in *Proc. ACM Data Communication Conf.*, Ithaca, NY, Sep. 1993, pp. 289–298.

[42] A. Reibman and V. Vaishampayan, "Quality monitoring for compressed video subjected to packet loss," in *Proc. IEEE Int. Conf. Multimedia and Exhibition*, Baltimore, MD, Jul. 2003, vol. 1, pp. 17–20.

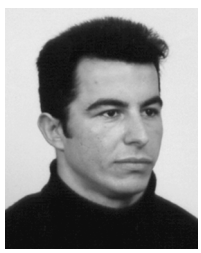
[43] J. Chakareski, J. Apostolopoulos, and B. Girod, "Low-complexity rate-distortion optimized video streaming," in *Proc. IEEE Int. Conf. Image Processing*, Singapore, Oct. 2004, vol. 3, pp. 2055–2058.

[44] *Video Coding for Low Bitrate Communication*, ITU-T Recommendation H.263 Version 2, Telecom. Standardization Sector of ITU, Feb. 1998.

[45] T. Cover and J. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[46] J. Chakareski and B. Girod, "Rate-distortion optimized video streaming with rich acknowledgments," in *Proc. SPIE Conf. Visual Communications and Image Processing*, San Jose, CA, Jan. 2004, vol. 5308, pp. 661–668.

[47] J. Chakareski and B. Girod, "Computing rate-distortion optimized policies for media streaming with rich acknowledgments," in *Proc. IEEE Data Compression Conf.*, Snowbird, UT, Mar. 2004, pp. 202–211.



Jacob Chakareski received the B.S. degree from Ss. Cyril and Methodius University, Skopje, Macedonia, in 1996, the M.S. degree from Worcester Polytechnic Institute, Worcester, MA, in 1999, and the Ph.D. degree from Rice University, Houston, TX, in 2005, all in electrical and computer engineering. He performed his doctoral thesis research at Stanford University, Palo Alto, CA, from 2002 to 2005.

In 2005–2006, he was a Postdoctoral Researcher with the Swiss Federal Institute of Technology (EPFL) in Lausanne. Presently, he is a Senior Engineer with Layered Media Inc., Rochelle Park, NJ, where he works on real-time video communication. He has held research positions with Microsoft and Hewlett-Packard. He has co-authored over 50 international publications and has two pending patent applications. His field of interests are networked

media systems, distributed computation and control, wireless communications, and Macedonian history.

Dr. Chakareski received the Best Student Paper Award at the IS&T/SPIE VCIP 2004 Conference.

Philip A. Chou (S'82–M'87–SM'00–F'04) received the B.S.E. degree from Princeton University, Princeton, NJ, in 1980, and the M.S. degree from the University of California, Berkeley, in 1983, both in electrical engineering and computer science, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1988.

From 1988 to 1990, he was a Member of Technical Staff at AT&T Bell Laboratories, Murray Hill, NJ. From 1990 to 1996, he was a Member of Research Staff at the Xerox Palo Alto Research Center, Palo Alto, CA. In 1997, he was Manager of the compression group at VXtreme, Mountain View, CA, before it was acquired by Microsoft in 1997. Since 1998, he has been a Senior Researcher with Microsoft Research, Redmond, WA, where he currently manages the Communication and Collaboration Systems research group. He also served as a consulting Associate Professor at Stanford University in 1994–1995, and has been an Affiliate Professor at the University of Washington since 1998. His research interests are data compression, information theory, communications, and pattern recognition, with applications to video, images, audio, speech, and documents.

Dr. Chou served as an Associate Editor in source coding for the IEEE TRANSACTIONS ON INFORMATION THEORY from 1998 to 2001, and served as a Guest Associate Editor for special issues in the IEEE TRANSACTIONS ON IMAGE PROCESSING and the IEEE TRANSACTIONS ON MULTIMEDIA in 1996 and 2004, respectively. From 1998 to 2004, he was a member of the IEEE Signal Processing Society's Image and Multidimensional Signal Processing technical committee (IMDSP TC). He is a Fellow of the IEEE, a member of Phi Beta Kappa, Tau Beta Pi, Sigma Xi, and the IEEE Computer, Information Theory, Signal Processing, and Communications societies, and was an active member of the MPEG committee. He is the recipient, with Anshul Seghal, of the 2002 ICME Best Paper Award, and is the recipient, with Tom Lookabaugh, of the 1993 Signal Processing Society Paper Award.