Computing Rate-Distortion Optimized Policies for Streaming Media with Rich Acknowledgments

Jacob Chakareski* and Bernd Girod
Information Systems Laboratory, Department of Electrical Engineering,
Stanford University, Stanford, CA 94305
{cakarz,bgirod}@stanford.edu

Abstract

We consider the problem of rate-distortion optimized streaming of packetized media over the Internet from a server to a client using rich acknowledgments. Instead of separately acknowledging each media packet as it arrives, the client periodically sends to the server a single acknowledgment packet, denoted *rich acknowledgment*, that contains information about all media packets that have arrived at the client by the time the rich acknowledgment is sent. Computing the optimal transmission policy for the server involves estimation of the probability that a single packet will be communicated in error as a function of the expected redundancy (or cost) used to communicate the packet. In this paper, we show how to compute this error-cost function, and thereby optimize the server's transmission policy, in this scenario.

1 Introduction

We consider the problem of streaming packetized media over a lossy packet network from a server to a client using rich feedback from the client. Packets may be lost in the network due to congestion or erasures. Currently, in sender-driven transmission schemes employed for streaming media the client replies with an acknowledgment packet whenever a media packet arrives. The purpose of the acknowledgment packet is to inform the server that the client has received the corresponding media packet and that the server does not need to consider retransmitting that media packet again.

In the present work, we employ an unconventional procedure for communicating to the server the receipt of media packets. Instead of separately acknowledging each media packet as it arrives, we periodically send to the server a single acknowledgment packet, denoted henceforth *rich acknowledgment*, that contains information about all media packets that have arrived at the client by the time the rich acknowledgment is

*On leave from Rice University, Houston, TX 77005.

sent. This information in essence reflects the current state of the client's buffer, i.e., which packets have been received by the client thus far.

All packets sent in either direction are subject to random loss or delay. Therefore the server can never be certain of the state of the client. However, the server is aware of the different deadlines, importances, and dependencies of the various media packets to be transmitted. Using this information the server is able to transmit its media packets based on the rich feedback it receives in a rate-distortion optimized way, that is, minimizing the expected end-to-end distortion subject to a constraint on the expected transmission rate. Such a rate-distortion optimized transmission algorithm, or transmission policy, results in unequal error protection provided to different portions of the media stream.

To compute the rate-distortion optimized transmission policy, we use the optimization algorithm presented in [1]. The core step of this algorithm involves estimation of the probability that a single packet will be communicated in error as a function of the expected redundancy, or cost, used to communicate the packet. The lower convex hull of the set of all expected error-cost pairs is called the error-cost function. How to compute this function, for the scenario of sender-driven streaming with rich acknowledgments, is the focus of this paper.

The concept of rich acknowledgments has been employed, under the name of vector acknowledgments, in the networking community as part of many feedback schemes, such as [2], in order to allow a TCP sender to perform selective retransmission of lost data packets. In addition, vector acknowledgements have been included as an option in the recently proposed Datagram Congestion Control Protocol (DCCP) [3], which implements a congestion-controlled, unreliable flow of datagrams suitable for use by applications such as streaming media, Internet telephony and on-line games. Furthermore, Rate-Distortion Optimized (RaDiO) packet scheduling is one of the latest advances in media streaming [1, 4]. To the best of our knowledge, rich acknowledgments have not been studied in RaDiO streaming. Finally, the initial results of our work are presented in [5].

2 Packet Loss Probabilities

The forward and the backward channel on a network path between a server and a client are characterized as independent time-invariant packet erasure channels with random delay. Hence, they are completely specified with the probabilities of packet loss ϵ_F and ϵ_B, and the probability densities of the transmission delay p_F and p_B, respectively. This means that if the media server sends a packet on the forward channel at time t, then the packet is lost with probability ϵ_F. However, if the packet is not lost, then it arrives at the client at time t', where the forward trip time $FTT = t' - t$ is randomly drawn according to the probability density p_F. Therefore, we let $P\{FTT > \tau\} = \epsilon_F + (1 - \epsilon_F) \int_\tau^\infty p_F(t)dt$ denote the probability that a packet transmitted by the server at time t does not arrive at the client application by time $t + \tau$, whether it is lost in the network or simply delayed by more than τ. Then similarly, $P\{BTT > \tau\} = \epsilon_B + (1 - \epsilon_B) \int_\tau^\infty p_B(t)dt$ denotes the probability that a

rich acknowledgment transmitted by the client at time t does not arrive at the server by time $t + \tau$, whether it is lost in the network or simply delayed by more than τ.

3 Media Communication using Rich Acks

A media session starts when a client requests a presentation from the media server. Once the request packet is received, the media server starts sending packets with data units from the presentation at discrete transmission opportunities $t_i, t_{i+1}, \ldots$ Note that the transmission decisions regarding when and how often each of the data units is sent to the client are completely determined by the server's transmission policy. The client periodically monitors the status of its buffer at every t_i and returns this information to the server via a single acknowledgment packet, i.e., a rich acknowledgment. The buffer information basically informs the server what data units have arrived at the client by the time (t_i) the rich acknowledgment is transmitted. Based on this information and the optimization framework presented in the next section, the server then dynamically decides at every transmission opportunity t_i what is at that moment the best transmission policy for every data unit in the presentation.

Note that in practice the server computes a sliding window of data units $\mathcal{W}_i$ at every t_i. Only the data units from $\mathcal{W}_i$ are considered for transmission at t_i. Therefore, at every t_i the client needs to send to the server information on the arrival status only for the data units in $\mathcal{W}_i$.

4 Rate-Distortion Optimization

The optimization algorithm has been explained elsewhere [1] and due to space constraints is not explained here. Its core step is finding a point on the lower convex hull of a set of points in the so-called "error-cost" plane, $\{(\rho(\pi), \epsilon(\pi)) : \pi \in \Pi\}$, where π is a *transmission policy* or protocol or algorithm for communicating a single data unit, Π is a family of transmission policies corresponding to the scenario at hand, $\epsilon(\pi)$ is the *expected error* or the probability that a data unit does not arrive at the client on time under policy π, and $\rho(\pi)$ is the *expected cost* or the expected number of transmitted bytes per source byte on the forward channel under policy π. The point of interest corresponds to the policy π^* minimizing $\epsilon(\pi) + \lambda\rho(\pi)$, for a given Lagrange multiplier $\lambda > 0$. In the next section, we show how to find π^* for the family of transmission policies Π corresponding to sender-driven streaming with rich acknowledgments.

5 Computing the Optimal Transmission Policy

For transmitting a single data unit on the forward channel, 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 server considers transmitting a packet for the data unit. The server need not transmit a packet at every transmission opportunity. The server does not transmit any packets after a rich acknowledgment is received confirming the

receipt of the data unit at the client. In addition, as explained in Section 3, the client sends a rich acknowledgment at every t_i notifying the server about the state of its buffer, i.e., which data units have arrived at the client by t_i. Note that at the same time this also informs the server which data units have not arrived at the client by t_i. Therefore, the information provided by a received rich acknowledgment is richer than that provided by a received conventional acknowledgment.

At each transmission opportunity t_i, $i = 0, 1, \ldots, N-1$, the server takes an action a_i, where $a_i = 1$ if the server sends a packet and $a_i = 0$ otherwise. Then, at the next transmission opportunity t_{i+1}, the server makes an observation o_i, where o_i is the set of rich acknowledgments received by the server in the interval $(t_i, t_{i+1}]$. For example, $o_i = \{NAK(t_1), ACK(t_3)\}$ means that during the interval $(t_i, t_{i+1}]$, the rich acknowledgment sent at t_1 arrived at the server informing that the data unit has not arrived at the client by t_1 ($NAK(t_1)$) and that the rich acknowledgment sent at t_3 arrived at the server informing that the data unit has arrived at the client by t_3 ($ACK(t_3)$). Note that for the purposes of our algorithm it is irrelevant for the server to distinguish the transmission times of the rich acknowledgments received within $(t_i, t_{i+1}]$ confirming that the data unit has arrived at the client by their respective transmission times. As explained above, once the server receives a confirmation that the data unit has arrived at the client, it stops sending any packets afterwards, regardless of the transmission times of the rich acknowledgments that brought that confirmation and regardless of any number of NAKs that might also arrive during $(t_i, t_{i+1}]$. Therefore, we drop the timing notation on these rich acknowledgments and simply use ACK to denote the event that at least one rich acknowledgment has been received confirming the receipt of the data unit due to previous transmissions. And thus, we also denote the event $o_i = \{NAK(t_1), ACK\}$ simply as $o_i = ACK$. Finally, we denote the event $o_i = \{NAK(t_1), NAK(t_3)\}$ simply as $o_i = NAK(t_3)$ since not receiving the data unit by t_3 implies that the data unit was certainly not received by t_1. In other words, we denote the events o_i, when multiple NAKs are received as an observation, using only the most recently sent NAK.

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}, as illustrated in Figure 1. Therefore, a state represents uniquely this sequence of action-observation pairs. If the final observation o_i includes an ACK, then q_{i+1} is a final state. In addition, any state at time $t_N = t_{DTS}$ is a final state. Final states in Figure 1 are indicated by double circles.

The action a_i taken at a non-final state q_i determines the transition probability $P(q_{i+1}|q_i, a_i)$ to the next state q_{i+1}. Formally, a policy π is a mapping $q \mapsto a$ from non-final states to actions. Thus any policy π induces a Markov tree with transition probabilities between states $P_\pi(q_{i+1}|q_i) \equiv P(q_{i+1}|q_i, \pi(q_i))$, and consequently also induces a probability distribution on final states. Therefore, Π is a collection of all possible trees in the state space described in Figure 1. Let q_F be a final state with history $(a_0, o_0) \circ (a_1, o_1) \circ \cdots \circ (a_{F-1}, o_{F-1})$, and let $q_{i+1} = q_i \circ (a_i, o_i)$, $i = 1, \ldots, F-1$, be the sequence of states leading up to q_F. Then q_F has probability $P_\pi(q_F) = \prod_{i=0}^{F-1} P_\pi(q_{i+1}|q_i)$, transmission cost $\rho_\pi(q_F) = \sum_{i=0}^{F-1} a_i$, and error $\epsilon_\pi(q_F) = 0$ if o_{F-1} contains an ACK and otherwise $\epsilon_\pi(q_F)$ is equal to the probability that none of

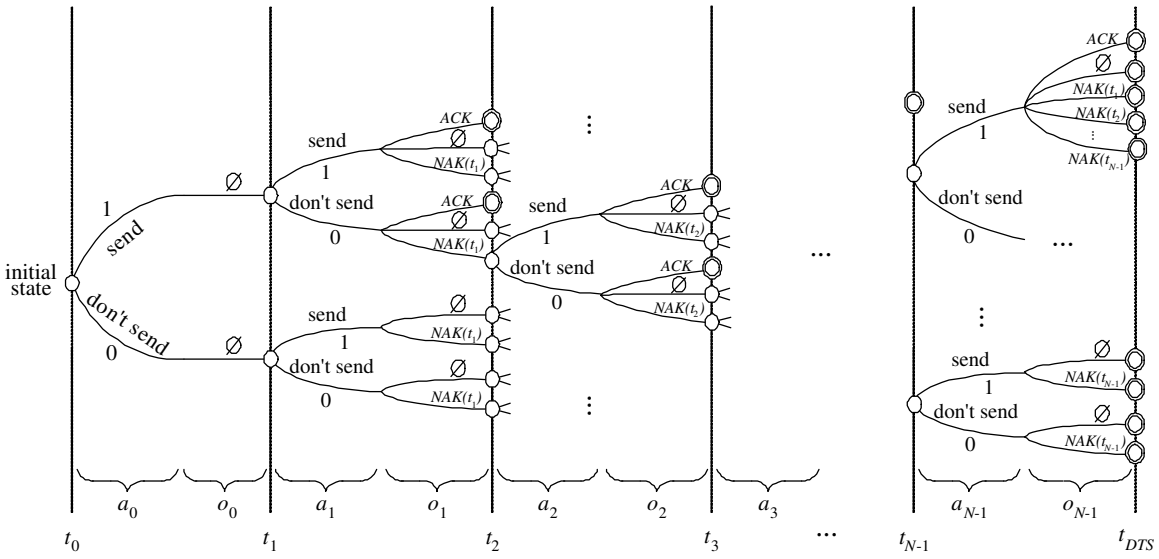

Figure 1: State space for a Markov decision process.

the transmitted packets arrives at the client on time (t_{DTS}), given q_F. For example, if q_F is the second state from the top at time t_{DTS} in Figure 1, then a packet with the data unit is sent at every transmission opportunity $t_0, t_1, \ldots, t_{N-1}$ and no rich acknowledgments are received. In that case, $\epsilon_\pi(q_F) = \prod_{i=0}^{N-1} P\{FTT > t_{DTS} - t_i\}$.

We can now express the expected cost and error for the Markov tree induced by policy π: $\rho(\pi) = E_\pi \rho_\pi(q_F) = \sum_{q_F} P_\pi(q_F)\rho_\pi(q_F)$, $\epsilon(\pi) = E_\pi \epsilon_\pi(q_F) = \sum_{q_F} P_\pi(q_F)\epsilon_\pi(q_F)$. As stated earlier we are interested in the policy minimizing the expected Lagrangian cost

$$J(\pi) \equiv \epsilon(\pi) + \lambda' \rho(\pi) = \sum_{q_F} P_\pi(q_F) J_\pi(q_F), \tag{1}$$

where $J_\pi(q_F) \equiv \epsilon_\pi(q_F) + \lambda' \rho_\pi(q_F)$. Let

$$J_\pi(q_i) = \begin{cases} \epsilon_\pi(q_F) + \lambda' \rho_\pi(q_F) \\ \qquad \text{if } q_i \text{ is final } (i = F), \\ \sum_{q_{i+1}} P(q_{i+1}|q_i, \pi(q_i)) J_\pi(q_{i+1}) \\ \qquad \text{otherwise} \end{cases}$$

be the expected Lagrangian of all paths through q_i (under π). Then let

$$J^*(q_i) = \begin{cases} \epsilon_\pi(q_F) + \lambda' \rho_\pi(q_F) \\ \qquad \text{if } q_i \text{ is final } (i = F), \\ \min_{a_i} \sum_{q_{i+1}} P(q_{i+1}|q_i, a_i) J^*(q_{i+1}) \\ \qquad \text{otherwise.} \end{cases} \tag{2}$$

By induction, $J^*(q_i) \leq J_\pi(q_i)$ for all q_i and all π, with equality if $\pi = \pi^*$, where

$$\pi^*(q_i) = \arg\min_{a_i} \sum_{q_{i+1}} P(q_{i+1}|q_i, a_i) J^*(q_{i+1}) \tag{3}$$

for all non-final states q_i. Thus the optimal policy (minimizing (1)) can be computed efficiently using (2) and (3).

Next, we provide the actual expressions for $\epsilon_\pi(q_F)$, $\rho_\pi(q_F)$ and $P(q_{i+1}|q_i, a_i)$ used in the equations above. As given before, the transmission cost $\rho_\pi(q_F) = \sum_{i=0}^{F-1} a_i$. Now, if no NAKs have been received along the path that leads to q_F in the state space from Figure 1, then the transmission error is $\epsilon_\pi(q_F) = \prod_{i\,:\,a_i=1} P\{FTT > t_{DTS} - t_i\}$. On the other hand, if at least one NAK has been received, then the transmission error is

$$\epsilon_\pi(q_F) = \prod_{i\,:\,i<j\,,\,a_i=1} P\{FTT > t_{DTS} - t_i | FTT > t_j - t_i\} \prod_{i\,:\,j\leq i\,,\,a_i=1} P\{FTT > t_{DTS} - t_i\},$$

where $j \in \{1, \ldots, N-1\}$ is the index of the most recently sent NAK that has arrived at the server, i.e., $NAK(t_j)$. Finally, we need to differentiate 4 possible cases for the transition probability $P(q_{i+1}|q_i, a_i)$. As mentioned earlier, $(a_0, o_0)\circ(a_1, o_1)\circ\cdots\circ(a_i, o_i)$ is the history, or the sequence of action-observation pairs that leads to state q_{i+1} at time t_{i+1}. Then, we can have

(a) no NAK received along the path that leads to q_i, i.e., $NAK \notin \bigcup_{j=0}^{i-1} o_j$ and no NAK received during $(t_i, t_{i+1}]$, i.e., $NAK \notin o_i$

$$P(q_{i+1}|q_i, a_i) = \prod_{l=1}^{i} P\{BTT > t_{i+1} - t_l | BTT > t_i - t_l\}, \tag{4}$$

(b) no NAK received along the path that leads to q_i, i.e., $NAK \notin \bigcup_{j=0}^{i-1} o_j$ and at least one NAK received during $(t_i, t_{i+1}]$, i.e., $NAK \in o_i$

$$\begin{aligned} P(q_{i+1}|q_i, a_i) &= P\{t_i - t_k < BTT \leq t_{i+1} - t_k | BTT > t_i - t_k\} \\ &\times \prod_{l>k}^{i} P\{BTT > t_{i+1} - t_l | BTT > t_i - t_l\} \prod_{l\,:\,l<k\,,\,a_l=1} P\{FTT > t_k - t_l\}, \end{aligned} \tag{5}$$

(c) at least one NAK received along the path that leads to q_i, i.e., $NAK \in \bigcup_{j=0}^{i-1} o_j$ and no NAK received during $(t_i, t_{i+1}]$, i.e., $NAK \notin o_i$

$$P(q_{i+1}|q_i, a_i) = \prod_{l>j}^{i} P\{BTT > t_{i+1} - t_l | BTT > t_i - t_l\}, \tag{6}$$

(d) at least one NAK received along the path that leads to q_i, i.e., $NAK \in \bigcup_{j=0}^{i-1} o_j$ and at least one NAK received during $(t_i, t_{i+1}]$, i.e., $NAK \in o_i$

$$\begin{aligned} P(q_{i+1}|q_i, a_i) &= P\{t_i - t_k < BTT \leq t_{i+1} - t_k | BTT > t_i - t_k\} \\ &\times \prod_{l>k}^{i} P\{BTT > t_{i+1} - t_l | BTT > t_i - t_l\} \\ &\times \prod_{l\,:\,l<k\,,\,a_l=1} P\{FTT > t_k - t_l | FTT > t_j - t_l\}, \end{aligned} \tag{7}$$

where j in (6) and (7) is the index of the most recently sent NAK received up to t_i, i.e., $NAK(t_j) \in \bigcup_{l=0}^{i-1} o_l$. Similarly, k in (5) and (7) is the index of the most recently sent NAK received during $(t_i, t_{i+1}]$, i.e., $NAK(t_k) \in o_i$. Note that in (7), it necessarily holds $j < k$ and $P\{FTT > t_k - t_l | FTT > t_j - t_l\} = P\{FTT > t_k - t_l\}$ for $j \leq l$.

6 Experimental Results

Here, we investigate the end-to-end distortion-rate performance for streaming packetized video content using different algorithms. The video content is a two-layer SNR scalable encoding of the image sequence *Foreman*. Using an H.263+ [6] codec 130 frames of QCIF *Foreman* have been encoded into a base layer and an enhancement layer with corresponding rates of 32 Kbps each. 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 9 consecutive P frames. Two RaDiO streaming systems are employed in the experiments. *Conv. ACK* is a system that performs RaDiO scheduling using conventional acknowledgments [4]. *Rich ACK* is the system presented in this work, which also performs RaDiO packet scheduling, but using rich acknowledgments. 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 end-to-end distortion, averaged over the duration of the video clip, as a function of the average transmission rate (Kbps) on the forward channel. In the experiments we use $T = 100$ ms as the time interval between transmission opportunities and 600 ms for the playback delay.

The forward and backward channels between the server and the client are specified as follows. Packets transmitted on these channels are dropped at random, with drop rates ϵ_F and ϵ_B, respectively. Those packets that are not dropped receive a random delay, where for the forward and backward delay densities p_F and p_B we use identical shifted Gamma distributions with parameters (n, α) and right shift κ, where $n = 1$ node, $1/\alpha = 25$ ms, and $\kappa = 20$ ms for a mean delay of $\kappa + n/\alpha = 50$ ms and standard deviation $\sqrt{n}/\alpha \approx 25$ ms. The distortion-rate performance of the two streaming systems was obtained as an average over 300 simulation runs in order to obtain statistically meaningful results.

First, we compare the efficiencies of the two streaming systems in terms of the transmission rate on the forward channel, as the packet loss rate increases on the backward channel and remains fixed on the forward channel. Figure 2 shows the Y-PSNR for the *Foreman* sequence as a function of the transmission rate on the forward channel, when $\epsilon_B = 0\%$, 5%, 10%, and 15%, while $\epsilon_F = 10\%$. It can be seen that as the backward channel degrades, the transmission rate on the forward channel of *Rich ACK* remains approximately constant (for any given Y-PSNR), while the transmission rate of *Conv. ACK* increases. Thus the gap between them, which is essentially zero when $\epsilon_B = 0\%$, increases as the packet loss rate on the backward channel increases. This is because as the backward channel degrades, *Conv. ACK* increasingly and unnecessarily retransmits information over the forward channel, due to loss of acknowledgment packets on the backward channel. Note that *Rich ACK* avoids this by acknowledging

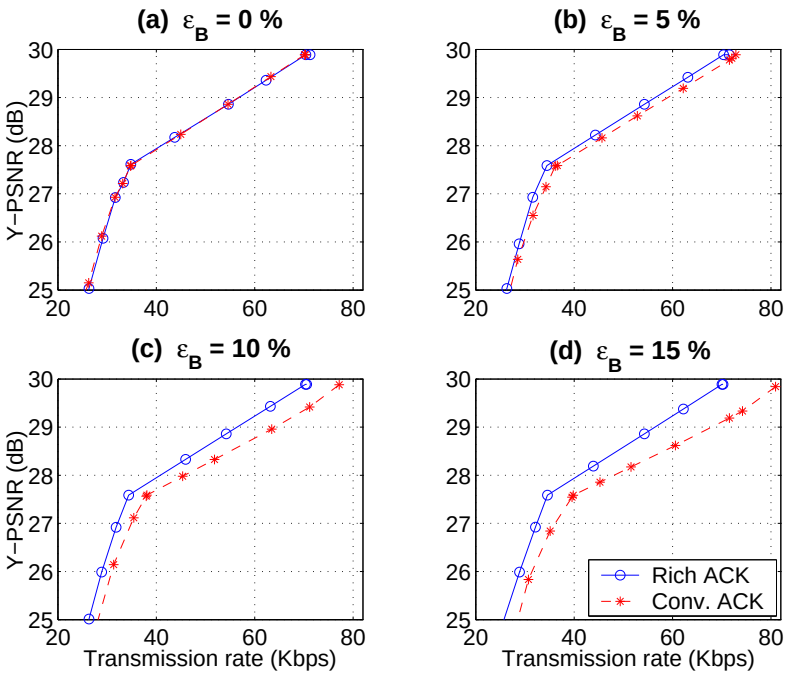

Figure 2: Y-PSNR (dB) vs. Transmission rate (Kbps) for *Foreman*.

every arriving media packet multiple times by consecutive rich acknowledgments.

We find that the performance of both *Rich ACK* and *Conv. ACK* can be accurately predicted from the distortion-rate function of the source and the values of ϵ_F and ϵ_B. When the number of retransmission opportunities before the playout deadline is sufficiently large, both systems transmit close to channel capacity. As is well known [7], 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 *Conv. ACK* continues to transmit packets until it receives an acknowledgment, which it receives with probability $(1-\epsilon_F)(1-\epsilon_B)$ for each transmitted packet, then *Conv. ACK* transmits on average $1/[(1-\epsilon_F)(1-\epsilon_B)]$ packets per data unit over the forward channel. On the other hand, *Rich ACK* transmits on average only $1/[(1-\epsilon_F)(1-\epsilon_B^K)]$ packets per data unit on the forward channel, where K is the number of rich acknowledgments sent after the data unit arrives at the client. Note that for the range of values for ϵ_B considered here, this can be approximately written as $1/(1-\epsilon_F)$ even for very small values of K. Therefore, rich acknowledgments essentially cancel out the effect of packet loss on the backward channel.

To confirm that these factors accurately predict performance, we normalize the transmission rates on the forward channel for all the graphs in Figure 2 with $1/[(1-\epsilon_F)(1-\epsilon_B)]$ for *Conv. ACK* and with $1/(1-\epsilon_F)$ for *Rich ACK* and we plot them (in gray) in Figure 3, 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 Figure 2 by multiplying the rate-distortion function by the appropriate factors for *Conv. ACK* and *Rich ACK*.

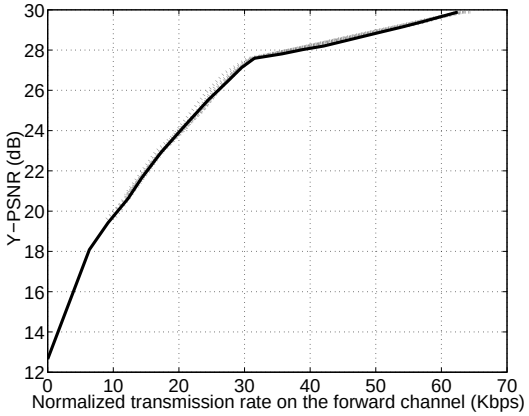

Figure 3: Y-PSNR (dB) vs. Normalized transmission rate (Kbps) for *Foreman*.

Next, we study the transmission rates of acknowledgments for both systems which simply comprise the amount of acknowledgment data sent to the client during a streaming session. In the case of *Conv. ACK* each acknowledgment packet contains only a header and no payload. On the other hand, a rich acknowledgment packet sent at transmission opportunity t_i consists of both a header and a payload of size w_i bits, where w_i is the number of data units in the transmission window of the server at t_i, as explained in Section 3. In our experiments, the header size in both systems is set to 320 bits. In Figure 4, we show the transmission rates of *Conv. ACK* and *Rich ACK* on the backward channel as a function of the transmission rate on the forward channel. It can be seen that for transmission rates greater than 40 Kbps, the rate of acknowledgments for *Conv. ACK* becomes larger than the corresponding rate for *Rich ACK*. For example, for transmission rates of 80 Kbps, *Rich ACK* saves 50 % of the bandwidth on the backward channel. Furthermore, note that the range of transmission rates $\geq$ 40 Kbps is also the range over which *Rich ACK* provides the most significant improvement in Y-PSNR performance, as shown in Figure 2.

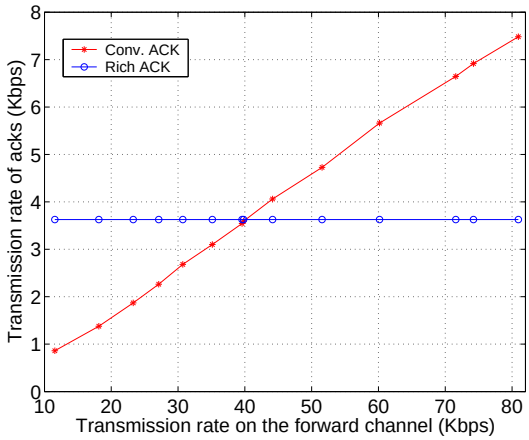

Figure 4: Rate of acks (Kbps) vs. Transmission rate (Kbps) for *Foreman*.

7 Conclusions

A framework has been presented for computing rate-distortion optimized transmission policies for streaming packetized media with rich acknowledgments. The computation of the optimal policies is done using a Markov decision tree with finite horizon N, associated with the transmission scenario under consideration. The R-D optimization framework is employed for streaming video content and its performance is compared with that of a sender-driven R-D optimized system that employs conventional acknowledgments. The results demonstrate that the proposed framework performs favorably over a large range of conditions considered for the feedback (backward) channel. The gains in performance increase as the packet loss rate on the backward channel increases. An additional advantage of streaming with rich acknowledgments is that it provides significant savings in transmission bandwidth on the backward channel. Finally, it was also shown that the performance of the proposed system is quite insensitive to variations in the quality of the feedback channel. This can be exploited to reduce the frequency of ack packets in order to conserve bandwidth or energy, which can be extremely useful for two classes of networks: networks with asymmetric links where the relative cost of sending an ack is higher and wireless sensor networks that have stringent power constraints.

References

[1] J. Chakareski and B. Girod, "Rate-distortion optimized packet scheduling and routing for media streaming with path diversity," in *Proc. Data Compression Conference*, Snowbird, UT, Mar. 2003, IEEE Computer Society, pp. 203–212.

[2] H.-S. Wilson So, Y. Xia, and J. Walrand, "A robust acknowledgement scheme for unreliable flows," in *Proc. Conf. on Computer Communications (INFOCOM)*, New York, NY, USA, June 2002, IEEE, vol. 3, pp. 1500–1509.

[3] E. Kohler, M. Handley, S. Floyd, and J. Padhye, "Datagram Congestion Control Protocol (DCCP)," Tech. Rep., IETF, http://www.ietf.org/internet-drafts/draft-ietf-dccp-spec-05.txt, Oct. 2003, Internet-Draft.

[4] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. Multimedia*, 2001, submitted.

[5] J. Chakareski and B. Girod, "Rate-distortion optimized video streaming with rich acknowledgments," in *Proc. Visual Communications and Image Processing*, San Jose, CA, Jan. 2004, SPIE, vol. 5310, to appear.

[6] Telecom. Standardization Sector of ITU, "Video coding for low bitrate communication," *ITU-T Recommendation H.263 Version 2*, Feb. 1998.

[7] T. Cover and J. Thomas, *Elements of Information Theory*, Wiley, 1991.