Distributed sender-driven video streaming[*]

Jacob Chakareski and Pascal Frossard

Ecole Polytechnique Fédérale de Lausanne (EPFL)
Signal Processing Institute - LTS4, CH-1015 Lausanne

ABSTRACT

A system for sender-driven video streaming from multiple servers to a single receiver is considered in this paper. The receiver monitors incoming packets on each network path and returns, to the senders, estimates of the available bandwidth on all the network paths. The senders in turn employ this information to compute transmission schedules for packets belonging to the video stream sent to the receiver. An optimization framework is proposed that enables the senders to compute their transmission schedules in a distributed way, and yet to dynamically coordinate them over time such that the resulting video quality at the receiver is maximized. Experimental results demonstrate that the proposed streaming framework provides superior performance over distortion-agnostic transmission schemes that perform proportional packet scheduling based only on the available network bandwidths.

1. INTRODUCTION

Multiparty streaming has drawn considerable attention in recent years. One of the scenarios that fall into this category is distributed streaming, where multiple senders transmit packets over separate network paths to a single receiver, as illustrated in Figure 1. This setting can be encountered for example in Content Delivery Networks (CDNs) where multiple servers may stream multimedia data to a single client or similarly in peer-to-peer (P2P) overlay networks where a client may have access to the same multimedia data at multiple peers in the network. A related concept is the Digital Fountain model [1] where the system tries to minimize the download time of a file at a client by connecting to multiple mirror server sites.

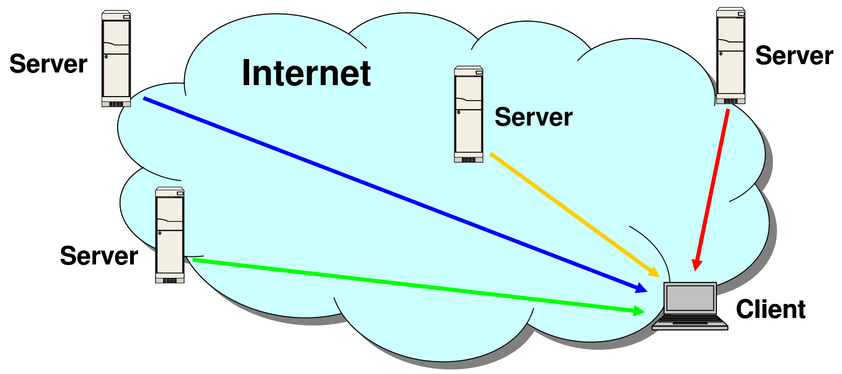

Figure 1. Distributed streaming: multiple senders - single receiver.

The possibility to receive the same data over multiple paths increases the resilience of the media presentation to network outages or congestion onsets. These may occur on some of the paths and thereby prevent timely delivery of some (or all) of the data units if only those paths are used for streaming. On the other hand, if the network paths exhibit good transmission quality it is desirable to spread the transmission of the media data over the multiple paths, i.e., to send different data units over different paths, in order to reduce the start-up delay of the client application and to ensure smooth and continuous play-out. Therefore, some form

[*]This work has been partly supported by the Swiss National Science Foundation, under grant PP-002-68737.

of coordination is required among the senders, with or without assistance from the client, in order to ensure the most efficient delivery of the desired media stream.

To the best of our knowledge, the earliest work that studied the problem of transmission coordination among the multiple senders in distributed streaming is [2]. In this work, the authors propose an algorithm that is run at the client and that performs rate allocation and packet partitioning among the senders. In a follow-up work, the authors combined the previously proposed algorithm with forward error correction for improved error resilience to burst packet loss [3]. Similarly, the works in [4, 5] consider receiver-driven control protocols that synchronize the senders' transmissions in a rate-distortion optimized way. For improved error-resilience, Multiple Description Coding (MDC) is employed at each sender to pre-encode (prior to transmission) a progressively encoded media content that is streamed afterwards to the client. Another related work is [6], where a rate-distortion optimization framework is proposed for packet scheduling in receiver-driven video streaming from multiple servers to a single client. The paper establishes that the gains in performance due to server (path) diversity, relative to a single server (path) case, are dependent on the quality of the network paths in terms of packet loss and delay. The performance of an MDC scheme for video streaming over multiple paths in CDNs is examined in [7, 8]. The authors report a 20-40% reduction in client video distortion, for the considered network conditions and topologies, relative to conventional CDNs that do not employ multiple description encoded video streams.

Differently from the prior work described above, in this paper we propose an optimization framework for synchronizing the transmission schedules of the multiple senders in distributed sender-driven streaming. In particular, instead of computing the transmission schedules for each sender as in receiver-driven approaches, the client only monitors incoming packets on each network path to determine the available bandwidth in the forward directions of the paths. This information is then fed back to the senders and is used in conjunction with the optimization framework to compute appropriate transmission actions at each sender. In essence, the framework enables the senders, based on the feedback information from the client, to compute independently, yet in a coordinated fashion, what their respective transmission policies should be, given the available bandwidth on each network path. It offers the interesting possibility for the system to really optimize the packet selection process: the relative packet importance is a priori known at the sender-driven strategies.

The rest of the paper proceeds as follows. In Section 2 we define our abstraction of the source of media data units, while in Section 3 we define a statistical model for the packet delay and loss exhibited on a network path. The specifics of the communication protocols employed between the senders and the client in the scenario under consideration are described in Section 4. Then, in Section 5 we show how the entire media presentation can be transmitted in a rate-distortion optimized way over the multiple network paths, using as a building block an algorithm for rate-distortion optimized transmission of a single media packet. This algorithm is the subject of Section 6. Finally, in Section 7 we examine the performance of the proposed streaming system via simulation experiments and compare its performance to that of a conventional distortion-agnostic system for distributed sender-driven streaming that performs packet scheduling based only on the available network bandwidths on the network paths. The papers ends with concluding remarks provided in Section 8.

2. MEDIA SOURCE MODEL

In a streaming media system, the encoded data are packetized into *data units* and are stored in a file on a media server. All the data units in the presentation have interdependencies, which can be expressed by a directed acyclic graph. Each node of the graph corresponds to a data unit, where an edge of the graph directed from data unit l' to data unit l implies that data unit l' can be decoded only if data unit l is first decoded.

Associated with each data unit l is a size B_l, a decoding time $t_{DTS,l}$, a set of data units $\mathcal{N}_c^{(l)}$ and an importance $\Delta d_l^{(l_1)}$. Specifically, the size B_l is the size of the data unit in bytes. $t_{DTS,l}$ is the *delivery deadline* by which data unit l must arrive at the client to be usefully decoded. Packets containing data units that arrive after the data units' delivery deadlines are discarded. Finally, $\mathcal{N}_c^{(l)} = \{1, \dots, l\}$ is the set of data units that the receiver considers for error concealment in case data unit l is not decodable by the receiver on time. Finally, $\Delta d_l^{(l_1)}$, for $l_1 \in \mathcal{N}_c^{(l)}$, is the reduction in reconstruction error (distortion) for the media presentation, when data unit l is not decodable but is concealed with data unit l_1 that is received and decoded on time.

3. PACKET DELAY AND PACKET LOSS PROBABILITIES

The forward and backward directions on the network path between a sender (server) and the receiver are modeled 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 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 an 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. Finally, these induce a probability $\epsilon_R = 1 - (1 - \epsilon_F)(1 - \epsilon_B)$ of losing a packet either on the forward or backward channel, and a round trip time distribution $P\{RTT > \tau\} = \epsilon_R + (1 - \epsilon_R) \int_\tau^\infty p_R(t)dt$, where $p_R = p_F * p_B$ is the convolution of p_F and p_B. Note that $P\{RTT > \tau\}$ is the probability that the server does not receive an acknowledgement packet by time $t + \tau$ for a data packet transmitted by the server at time t. In the following, we use a superscript m, for $m = 1, \ldots, M$, to distinguish the packet delay characterizations associated with the individual network paths.

4. DISTRIBUTED SENDER-DRIVEN MEDIA COMMUNICATION

In this section, we describe in more detail the communication protocols employed in the streaming scenario under consideration. Let there be M senders (servers) transmitting data units of a media presentation on M independent paths. The receiving client in turn monitors the forward-trip time (FTT) of arriving packets and, based on these quantities, estimates the available bandwidth in the forward direction for each path. In particular, let $FTT_1^k, FTT_2^k, \ldots, FTT_P^k$ be the transmission delays experienced by the packets received by the client on path k in the last ΔT seconds. Then, the most recent estimate for the bandwidth (data rate) available on path k is computed by the client as $\widetilde{R}_k = (1/P) \sum_{j=1}^P (B_j / FTT_j^k)$. This is simply the average of the P most recent estimates of the available bandwidth on the path associated with the corresponding received packets, where B_j is the size of packet j in bits (or bytes). At the end of each estimation period ΔT the client returns to every sender a special acknowledgement packet[†] that contains the latest bandwidth estimates $\widetilde{R}_1, \ldots, \widetilde{R}_M$ for all paths. The senders, equipped with the optimization framework described in the next section, use this information to compute their appropriate transmission schedules for streaming media packets to the client.

A transmission policy basically represents the actions performed on a given data unit, at each transmission opportunity. Let $t_0, t_1, \ldots, t_{N-1}$ be a window of N transmission opportunities at which the senders can transmit packets with the data unit to the client, prior to its delivery deadline $t_N = t_{DTS}$. The transmission policy π consists then of a binary vector of actions a_i^m, for $i = 0, \ldots, N - 1$, and $m = 1, \ldots, M$; we denote by $a_i^m = 1$ the transmission of a packet with the data unit on path m at transmission opportunity t_i, and $a_i^m = 0$ signifies the converse. The arrival of a packet with the data unit at the client is immediately acknowledged to all senders by sending acknowledgement packets in the backward direction on all M paths. Note that sender m will only send a packet with the data unit for $a_i^m = 1$ if no acknowledgement arrives at the sender by t_i to report the correct reception of the data unit due to earlier transmissions.

5. RATE-DISTORTION OPTIMIZATION

Suppose that there are L data units in the multimedia session. 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. Any given policy vector $\boldsymbol{\pi}$ induces for the multimedia session an expected distortion $D(\boldsymbol{\pi})$ and a vector of expected transmission rates $R(\boldsymbol{\pi}) = [R_0(\boldsymbol{\pi})\, R_1(\boldsymbol{\pi}) \ldots R_M(\boldsymbol{\pi})]$ on the forward channels of the network paths. We seek

[†]Alternatively, these bandwidth estimates may be piggy-backed on regular acknowledgement packets.

the policy vector $\boldsymbol{\pi}$ that minimizes the expected distortion $D(\boldsymbol{\pi})$ such that the expected transmission rates on the forward channels do not exceed the available bandwidth on these channels, $\{\widetilde{R_m}\}$, respectively, i.e.,

$$\boldsymbol{\pi}^* = \arg\min_{\boldsymbol{\pi}} D(\boldsymbol{\pi}), \qquad \text{s.t. } R_m(\boldsymbol{\pi}) \leq \widetilde{R}_m \;,\; m = 1, \ldots, M. \tag{1}$$

Using the method of Lagrange multipliers, we reformulate (1) as an unconstrained optimization problem. That is, we seek the policy vector $\boldsymbol{\pi}$ that minimizes the expected Lagrangian $J(\boldsymbol{\pi}) = D(\boldsymbol{\pi}) + \sum_{m=1}^{M} \lambda_m R_m(\boldsymbol{\pi})$ for some vector of positive Lagrange multipliers $\boldsymbol{\lambda} = [\lambda_1 \, \lambda_2 \, \ldots \, \lambda_M]$, and thus achieves a point on the lower convex hull of the set of all achievable distortion-rate pairs $(D(\boldsymbol{\pi}), R(\boldsymbol{\pi}))$[‡].

In the following, we explain how $D(\boldsymbol{\pi})$ and $R(\boldsymbol{\pi})$ can be computed. The expected transmission rate $R_m(\boldsymbol{\pi})$ on path m is the sum of the expected transmission rates on this path for each data unit $l \in \{1, \ldots, L\}$ in the presentation:

$$R_m(\boldsymbol{\pi}) = \sum_{l} B_l \rho_m(\pi_l), \tag{2}$$

where B_l is the size of data unit l in bytes and $\rho_m(\pi_l)$ is the *expected cost* per byte, or the expected number of transmitted bytes per source byte under policy π_l on path m. The expected distortion $D(\boldsymbol{\pi})$ is somewhat more complicated to express, but it can be expressed in terms of the *expected error*, or the probability $\epsilon(\pi_l)$ that data unit l does not arrive at the client on time (under policy π_l). We borrow the expression for $D(\boldsymbol{\pi})$ from [9] and refer the reader to this work for more details on the derivation

$$\begin{aligned} D(\boldsymbol{\pi}) \;&=\; D_0 - \sum_{l} \sum_{l_1 \in \mathcal{N}_c^{(l)}} \Delta d_l^{(l_1)} \prod_{j \in \mathcal{A}(l_1)} (1 - \epsilon(\pi_j)) \times \\ & \prod_{l_2 \in \mathcal{C}(l,l_1)} \left(1 - \prod_{l_3 \in \mathcal{A}(l_2) \setminus \mathcal{A}(l_1)} (1 - \epsilon(\pi_{l_3})) \right) \end{aligned} \tag{3}$$

where D_0 is the expected reconstruction error for the presentation if no data units are received. $\mathcal{A}(l_1)$ is the set of ancestors of l_1, including l_1. $\mathcal{C}(l, l_1)$ is the set of data units $j \in \mathcal{N}_c^{(l)} : j > l_1$ that are not mutual descendants, i.e., for $j, k \in \mathcal{C}(l, l_1) : j \notin \mathcal{D}(k), k \notin \mathcal{D}(j)$, where $\mathcal{D}(j)$ is the set of descendants of data unit j. Finally, "$\setminus$" denotes the operator "set difference".

Finding a policy vector $\boldsymbol{\pi}$ that minimizes the expected Lagrangian $J(\boldsymbol{\pi})$ is difficult since the terms involving the individual policies π_l in $J(\boldsymbol{\pi})$ are not independent. Therefore, we employ an iterative descent algorithm, called 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 [9, 10]. It can be shown that the optimal individual policies at iteration n, for $n = 1, 2, \ldots$, are given by

$$\pi_l^{(n)} = \arg\min_{\pi_l} S_l^{(n)} \epsilon(\pi_l) + B_l \sum_{m=1}^{M} \lambda_m \rho_m(\pi_l), \tag{4}$$

where $S_l^{(n)} = \sum_{l_1 \,:\, l \in N_c^{(l_1)}} \left(S_{l,l_1}^{+(n)} - S_{l,l_1}^{-(n)} \right) = S_l^{+(n)} - S_l^{-(n)}$ can be regarded as 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. note that the expressions for $S_{l,l_1}^{+(n)}$ and $S_{l,l_1}^{-(n)}$ can be obtained from (3) by grouping terms.

The minimization (4) 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 $\epsilon(\pi_l) + \sum_{m=1}^{M} \lambda'_m \rho_m(\pi_l)$, where $\lambda'_m = \lambda_m B_l / S_l^{(n)}$.

[‡]Equivalently, the set of all achievable distortion-rate $(D, R_1, \ldots, R_M)$ (M+1)-tuples.

Finally, finding the appropriate choice of the Lagrange multipliers λ_m is performed as follows. We initially select $\lambda_m = \lambda$, for $m = 1, \ldots, M$, and some $\lambda > 0$. Then, we repeatedly re-run the ISA optimization algorithm till convergence, while adjusting one of the Lagrange multipliers every time the optimization algorithm converges. Adjusting a Lagrange multiplier for a given rate constraint is usually done in an iterative fashion using fast convex search techniques such as the bisection search technique [11–13] or the Bezier curve based technique proposed in [14]. The procedure outlined above is repeated until we properly adjust all Lagrange multipliers.

6. COMPUTING THE OPTIMAL POLICY π^*

It is the duty of sender m, $m = 1, \ldots, M$, to recompute the optimal policy π^* for a data unit at every t_i and then execute a_i^m from π^*. Note that it would be sufficient to determine π^* only once at time t_0, except for the fact that the Lagrange multipliers λ'_m in (4) may be adjusted by the optimization procedure from Section 5 at each transmission opportunity t_i in order to take into account feedbacks (acknowledgements) observed by the senders due to transmissions of packets related to data units sent prior to t_i.

In the following, we provide expressions for the expected error-cost for the family of transmission policies Π corresponding to sender-driven distributed streaming. They have been adopted from [9], where they are derived for the scenario of sender-driven streaming with path diversity: single server - single client communicating over multiple network paths. As described earlier, $\epsilon(\pi)$ is the probability that a data unit is not delivered on time given the transmission actions in π. Furthermore, the expected cost on path m at transmission opportunity t_i is zero if $a_i^m = 0$, and otherwise it is equal to the probability that no acknowledgements arrive at sender m by t_i, as a result of previous transmissions of the data unit. Hence, we can write

$$\begin{aligned}
\epsilon(\pi) &= \prod_{\substack{j<i,\, p\,: \\ a_{jp}=1}} P\{FTT^p > t_{DTS} - t_j | FTT^p + BTT^m > t_i - t_j\} && (5)\\
&\quad \times \prod_{\substack{j\geq i,\, p\,: \\ a_{jp}=1}} P\{FTT^p > t_{DTS} - t_j\},\\
\rho_m(\pi) &= \sum_{\substack{j\geq i\,: \\ a_{jm}=1}} \prod_{\substack{l<i,\, p\,: \\ a_{lp}=1}} P\{FTT^p + BTT^m > t_j - t_l | FTT^p + BTT^m > t_i - t_l\} && (6)\\
&\quad \times \prod_{\substack{i\leq l<j,\, p\,: \\ a_{lp}=1}} P\{FTT^p + BTT^m > t_j - t_l\},
\end{aligned}$$

where the conditional probabilities in (5) and (6) can be computed using Bayes' rule [15] and the fact that $P\{x > a, x + y > b\} = P\{x > a\}$, for x, y nonnegative and $a > b$. To this end, note that the probability densities of the sums $FTT^p + BTT^m$ can be obtained as the convolution of the corresponding densities for FTT^p and BTT^m.

It has to be noted that the search for the optimal transmission policy, as described before, may lead to systems that does not scale very well in practice; indeed, the overall computational complexity, as well as the computational complexity in each server, increase with the number of senders. This is a price to pay for having a distributed streaming strategy, which is fully sender-driven. However, several alternatives can be considered to reduce the computational complexity of the system. First, a priori packet classification [16] can considerably reduce the complexity of the search for the best transmission policy. Then, one can propose an a priori selection of subsets of packets to be sent by each of the servers, to reduce the computational load of the overall system. Finally, fast search methods based on efficient reduction of the policy search space can further decrease the computational complexity [17].

7. EXPERIMENTAL RESULTS

Here, we examine the performance of the proposed packet scheduling framework for distributed video streaming. The video content employed in the simulation experiments are the test video sequences Foreman and Mother & Daugther in QCIF size encoded at 10 fps using an H.264 codec [18]. Each sequence is encoded with a constant quantization level at an average luminance (Y) PSNR of about 36 dB and a Group of Pictures (GOP) size of 20 frames, where each GOP consists of an I frame followed by 19 consecutive P frames.

There are $M = 2$ senders streaming the video content to a client over two independent network paths. The network bandwidth in the forward direction of each path is randomly varying between a lower and an upper bound. In the simulations, random fluctuations of the available bandwidth occur every two seconds, while the time period ΔT employed at the client for bandwidth estimation is set to one second. The packet delay densities are assumed to be exponential functions and are inherently tied to the available bandwidth on the network paths. In particular, from the M/M/1 model [19] that is frequently used to model network queues, we know that the mean of the corresponding exponential distribution for the network delay experienced by a packet is $\mu = PackSize/BW$, where $PackSize$ is the average packet size and BW is the available bandwidth. In our simulations, we employ an average packet size of 500 bytes to simulate a random packet delay from a given exponential distribution based on the available bandwidth. Similarly, the senders employ the most recent estimates of the available bandwidth on the network paths to compute the appropriate packet delay probabilities in Section 6. Finally, in the experiments we use $T = 100$ ms as the time interval between transmission opportunities, while the playback delay of the client application is set to one second.

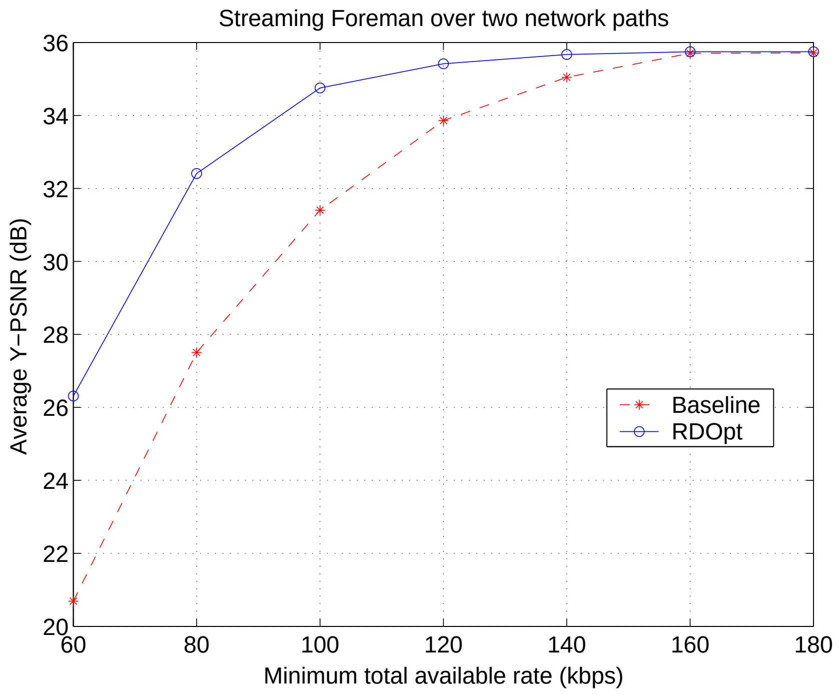

Figure 2. Rate-distortion performance for bandwidth adaptation of Foreman.

In the first set of experiments, we employ the optimization framework for bandwidth adaptation. In particular, the packet loss rates on the network paths are assumed to be zero, i.e., $\epsilon_F = \epsilon_B = 0$, and we study how the framework performs in adjusting the streaming rate of the video content to the bandwidth variations of the underlying network. Performance is measured in terms of the average PSNR in dB of the luminance (Y) component of the reconstructed video signal at the receiver. The range in which the network bandwidth is randomly varying, is given as $[BW_{min}, BW_{min} + 20]$ for one of the paths, and $[BW_{min} + 20, BW_{min} + 40]$ for the other network path, where BW_{min} is measured in kbps. Hence, the paths are asymmetric in terms of available bandwidth. In the simulations, we change BW_{min} across a certain range, and for each of its values we record the corresponding Y-PSNR performance of the proposed system, henceforth denoted *RDOpt*. For comparison purposes, in the experiments we have also examined the performance of a baseline system, denoted *Baseline*, that performs proportional packet scheduling based only on the available network bandwidths. In particular, the two senders split the packets of the video stream in proportion to the bandwidth estimates provided by the client. Packet dropping decisions in *Baseline* are performed randomly without taking into account their specific distortion importance. In other words, *Baseline* is distortion-agnostic.

In Figure 2 we show the performances of the two systems for streaming Foreman as a function of the minimum transmission bandwidth available on both paths during a session, i.e., $2 * BW_{min} + 20$. It can be seen from the figure that *RDOpt* provides an improved performance over the baseline system almost over the whole range of values considered for the minimum overall bandwidth that is available. The gains in performance are especially significant in the lower half of the bandwidth range. For example, at 60 kbps minimum overall bandwidth there is a difference of 4.5 dB in performance between the two systems. The improvement is due to the fact that the optimized system takes into account the distortion importance of the individual packets while adapting the video stream to the available bandwidth. In particular, by selecting the most important packets for transmission for the given available data rate on each path, *RDOpt* ensures the best possible reconstruction quality of the video presentation at the receiver. On the other hand, *Baseline* performs bandwidth adaptation without treating the various packets preferentially, as it is distortion-agnostic. Therefore, some more important packets may be dropped at the expense of others, less important ones, which would ultimately lead to a degradation in video quality at the client. Finally, it can be seen from Figure 2 that the two systems under comparison perform alike for a sufficiently large minimum total bandwidth. This is expected as here there is sufficient bandwidth available throughout the session to ensure timely deliver of all packets to the receiver in both systems. In other words, no packets need to be dropped anymore due to a mismatch in the network bandwidth variations and the temporally varying source encoding rate.

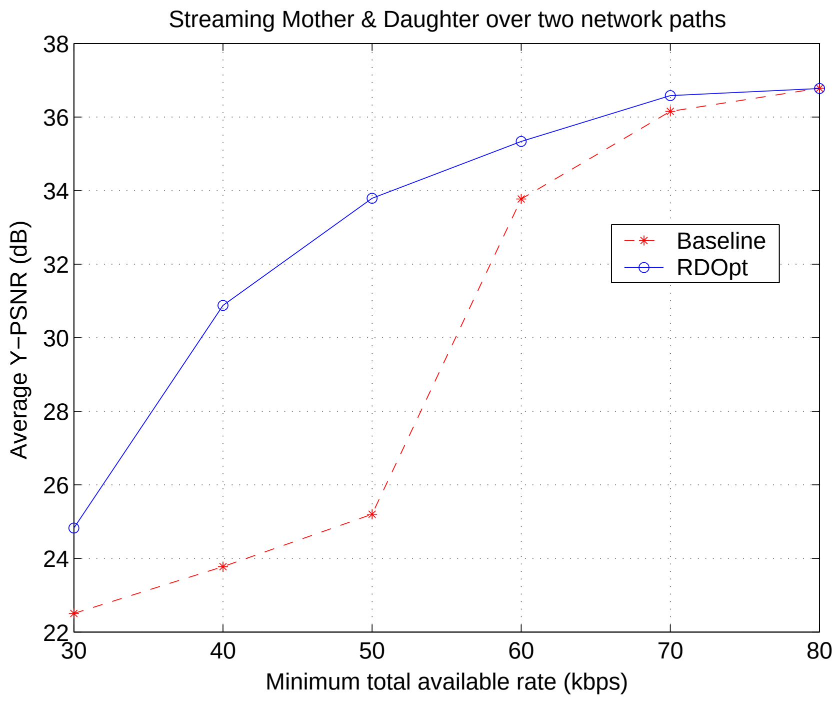

Figure 3. Rate-distortion performance for bandwidth adaptation of Mother & Daughter.

A similar relative performance of the two systems is observed for streaming Mother & Daughter, as shown in Figure 3. It can be seen that again *RDOpt* outperforms *Baseline* almost over the whole range of bandwidth values under consideration, with gains reaching up to 6-8 dB in the lower half of the bandwidth range. Finally, when the overall available bandwidth reaches a value at which no bandwidth adaptation is needed throughout the session, the two systems perform alike, as illustrated by their performances for 80 kbps minimum overall bandwidth.

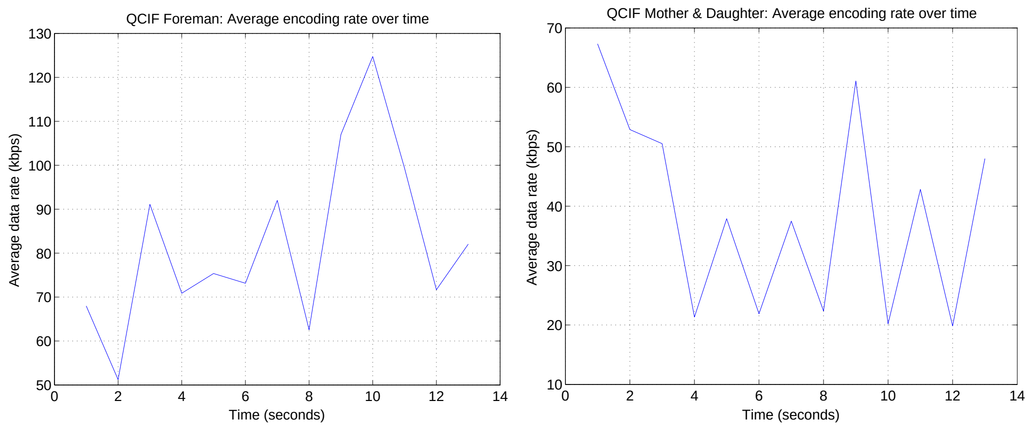

Figure 4. Average encoding rate (kbps) over time for (left) Foreman, (right) Mother & Daughter.

A useful evidence for the interpretation of the experimental results described thus far is the source encoding rate of the video content used in the experiments. In particular, in Figure 4 (left) and Figure 4 (right) we show the encoding rates over time of the two sequences Foreman and Mother & Daughter, respectively. It can be seen from the figures that their encoding rates over time, averaged over one second segments, can reach up to 130 kbps and 70 kbps, respectively. Therefore, it is only above such data rates that no bandwidth adaptation via packet dropping needs to be performed when these videos are streamed to a receiver. Hence, the identical performances of *RDOpt* and *Baseline* in the very upper end of the bandwidth range under consideration in Figure 2 and Figure 3. Finally, it should be noted in this regards that extra available bandwidth, above the peak encoding rate of a video content, is needed if full Y-PSNR performance at the receiver is desired. Due to the random bandwidth variations, the senders may sometimes overestimate the available bandwidth at present. This in turn will cause them to transmit at rates which may contribute to long packet delays experienced at the network queues, especially if the rates (the available bandwidths on the network paths) at which the queues are drained are not sufficiently large. Hence, in such situations, even if all video packets are transmitted by the senders, some may still arrive late to be decoded and displayed on time at the client.

Next, we examine the performances of *RDOpt* and *Baseline* when adapting to packet loss. In particular, there is sufficient bandwidth on the network paths to accommodate streaming the video content without packet dropping for bandwidth adaptation. However, packet losses occur and some packets are lost during transmission. Therefore, a streaming system needs to decide then whether it would retransmit (old) previously transmitted packets or it would send new packets that have not been sent before. This trade-off is necessitated by the fact that the available bandwidth is sufficient only to support streaming the video content without retransmissions. In the experiments, we vary the packet loss rate experienced on the forward channels from the senders to the receiver and record the resulting Y-PSNR (dB) performances of the two systems under examination. The network paths still exhibit random bandwidth variations, however, the minimum overall bandwidth on all paths is selected such that it can support sending once all packets from the video presentation.

In Figure 5, we show the Y-PSNR improvement of *RDOpt* over *Baseline* in dB for performing packet loss adaptation, as a function of the average packet loss rate (%) on the network paths. In particular, in Figure 5 (left) we can see that roughly 1 dB gain is obtained for streaming Foreman, when the choice between transmission and (re)transmission is performed in an optimized way. The gains in performance increase almost linearly with the packet loss rate, so that, at a 20% packet loss ratio, an improvement of close to 3.5 dB is measured. We observe a similar situation when streaming Mother & Daughter in the presence of packet loss, as shown in Figure 5 (right). Note, however, that the gains in this case and their relative increase as

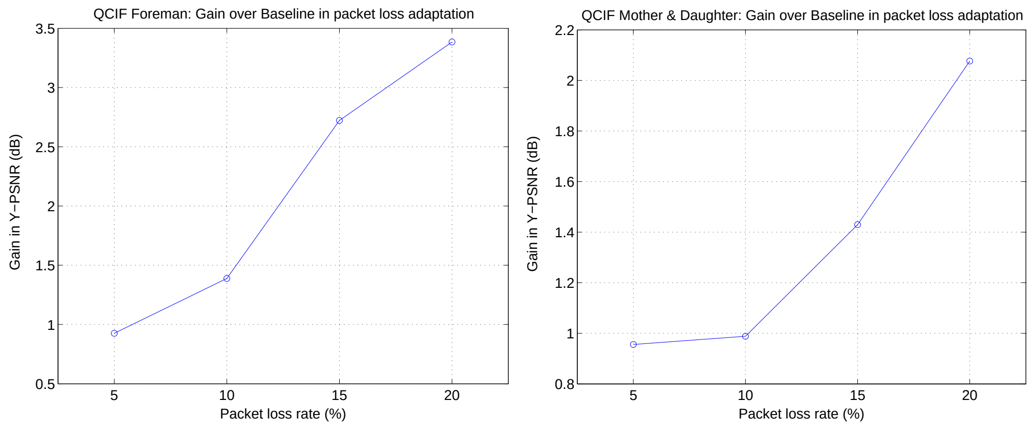

Figure 5. Y-PSNR gain (dB) of *RDOpt* over *Baseline* vs. Packet loss rate (%) for (left) Foreman, (right) Mother & Daughter.

a function of the packet loss rate are not as significant as the corresponding ones for the case of Foreman. That is because Mother & Daughter is a sequence that exhibits low motion and scene complexity relative to Foreman. Therefore, performing error concealment in order to account for missing video packets works much better in the case of the former sequence, as noted in several prior works, e.g., [20]. This in turn renders the choice between transmissions of new packets and retransmissions of old packets less important for the resulting end-to-end performance: hence we see less performance difference between *RDOpt* and *Baseline* in this case.

8. CONCLUSIONS

We have presented a system for rate-distortion optimized packet scheduling in distributed video streaming. The system consists of an optimization framework for scheduling the packet transmissions at the individual senders, combined with a bandwidth estimation technique that is performed at the receiving client. Using the framework and the bandwidth estimates provided by the client, the sender can independently, but still in coordination, decide what are the most importance packets to transmit on each path given their available bandwidths. In the experimental evaluation of the proposed system, it is established that significant performance gains are observed over a conventional distortion-agnostic system for distributed streaming. This is true as long as bandwidth adaptation of the video content needs to be performed due to the disparity between the available network bandwidth and the encoding rate of the video content, both of which are varying over time. Finally, we are currently working on analyzing the stability of such a distributed streaming strategy, as it may represent an important issue in networks with very heterogeneous delay channels.

REFERENCES

1. J. Byers, M. Luby, and M. Mitzenmacher, “Accessing multiple mirror sites in paralel: using tornado codes to speed up downloads,” in *Proc. Conf. on Computer Communications (INFOCOM)*, vol. 1. New York City, USA: IEEE, Mar. 1999, pp. 275–283.
2. T. Nguyen and A. Zakhor, “Distributed video streaming over internet,” in *Proc. Multimedia Computing and Networking*, vol. 4673. San Jose, CA: SPIE, Jan. 2002, pp. 186–195.
3. ——, “Distributed video streaming with forward error correction,” in *Proc. Int'l Packet Video Workshop*, Pittsburg, PA, Apr. 2002.
4. A. Majumdar, R. Puri, and K. Ramchandran, “Distributed multimedia transmission from multiple servers,” in *Proc. Int'l Conf. Image Processing*, vol. 3. Rochester, NY, USA: IEEE, Sept. 2002, pp. 177–180.
5. J. Kim, R. M. Mersereau, and Y. Altunbasak, “Network-adaptive video streaming using multiple description coding and path diversity,” in *Proc. Int'l Conf. Multimedia and Exhibition*, vol. 2. Baltimore, MD, USA: IEEE, July 2003, pp. 653–656.

6. J. Chakareski and B. Girod, “Server diversity in rate-distortion optimized streaming of multimedia,” in *Proc. Int’l Conf. Image Processing*, vol. 3. Barcelona, Spain: IEEE, Sept. 2003, pp. 645–648.
7. J. Apostolopoulos, T. Wong, W.-T. Tan, and S. Wee, “On multiple description streaming with content delivery networks,” in *Proc. Infocom*, vol. 3. New York City, NY, USA: IEEE, June 2002, pp. 1736–1745.
8. J. Apostolopoulos, W.-T. Tan, and S. Wee, “Performance of a multiple description streaming media content delivery network,” in *Proc. Int’l Conf. Image Processing*, vol. 2. Rochester, NY, USA: IEEE, Sept. 2002, pp. 189–192.
9. 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: IEEE Computer Society, Mar. 2003, pp. 203–212.
10. P. A. Chou and Z. Miao, “Rate-distortion optimized streaming of packetized media,” Microsoft Research, Redmond, WA, Tech. Rep. MSR-TR-2001-35, Feb. 2001.
11. Y. Shoham and A. Gersho, “Efficient bit allocation for an arbitrary set of quantizers,” *IEEE Trans. Acoustics Speech and Signal Processing*, vol. 36, no. 1, pp. 1445–1453, Sept. 1988.
12. K. Ramchandran and M. Vetterli, “Best wavelet packet bases in a rate-distortion sense,” *IEEE Trans. Image Processing*, vol. 2, no. 2, pp. 160–175, Apr. 1993.
13. G. M. Schuster, G. Melnikov, and A. K. Katsaggelos, “A review of the minimum maximum criterion for optimal bit allocation among dependent quantizers,” *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 3–17, Mar. 1999.
14. G. M. Schuster and A. K. Katsaggelos, “An optimal quadtree-based motion estimation and motion-compensated interpolation scheme for video compression,” *IEEE Trans. Image Processing*, vol. 7, no. 11, pp. 1505–1523, Nov. 1998.
15. A. Papoulis and S. Unnikrishna Pillai, *Probability, Random Variables and Stochastic Processes.* New York: McGraw-Hill Inc., 2001, 4-th ed.
16. J. Chakareski and P. Frossard, “Low-complexity adaptive streaming via optimized a priori media pruning,” in *Proc. Workshop on Multimedia Signal Processing.* Shanghai, China: IEEE, Oct./Nov. 2005, to appear.
17. A. Sehgal, A. Jagmohan, O. Verscheure, and P. Frossard, “Fast distortion-buffer optimized streaming of multimedia,” in *Proc. Int’l Conf. Image Processing.* Genova, Italy: IEEE, Sept. 2005.
18. Telecom. Standardization Sector of ITU, “Video coding for low bitrate communication,” *Draft ITU-T Recommendation H.264*, Mar. 2003.
19. D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Englewood Cliffs, N.J.: Prentice Hall, 1992.
20. J. Chakareski, J. Apostolopoulos, S. Wee, W.-T. Tan, and B. Girod, “R-D hint tracks for low-complexity R-D optimized video streaming,” in *Proc. Int’l Conf. Multimedia and Exhibition*, vol. 2. Taipei, Taiwan: IEEE, June 2004, pp. 1387–1390.