

Rate Distortion Optimized Streaming with Multiple Deadlines and Proxy Servers for Low Latency Applications

Eric Thea and Chung-Tse Mar
{ethea, ctmar}@stanford.edu

Collaborator: Mark Kalman

Abstract—In this work, we present a streaming media scheme using proxy servers and multiple deadlines. The proxy server uses its knowledge of channel conditions, packet deadlines and dependencies to request and send data units in a rate distortion optimized way. Because multiple deadlines are considered, the proxy can determine the optimal transmission policy when the decoder employs accelerated retroactive decoding (ARD) to mitigate the effects of late arrivals. We propose an algorithm that efficiently computes this best policy by considering a policy window independently of the deadlines. We expect experiments to demonstrate that the proposed scheme provides a significant performance improvement over the sender driven case and the single deadline case.

I. INTRODUCTION

In this work, we examine a streaming media system where a proxy server acts as an intermediary between a server and a client. Located at the junction of the backbone network and the last hop to the client, this proxy requests data units from the server, caches them upon reception and transfers them to the client. After sending a data unit to the client, the proxy waits for an acknowledgment (ACK) to come back. But a packet sent over the channel between the proxy and the client (channel 2) experiences a random delay or may even be lost. If the time between the data unit is sent and the ACK is received, called round trip time on channel 2 (RTT2), is too large, the proxy can consider retransmitting this data unit. Note that in our formulation, a lost packet is represented by a round trip time equal to infinity. Similarly, data requests might need to be repeated if the round trip time in the backbone network (RTT1), is too large. The round trip times considered in the proxy case are on average shorter than in the sender-driven case, which is the case where the server sends data units directly to the client. Therefore, retransmissions are more efficient in the proxy scenario and performance is improved as shown in [2].

The question remains as when the proxy should transmit or retransmit packets. Transmitting too often would increase the rate unnecessarily and potentially cause congestion whereas not transmitting enough times would hurt the PSNR of the decoded video sequence. The transmission policy is obtained

in a rate distortion optimized way [1]. At discrete transmission opportunities, data units contained in the transmission window \mathcal{W} are eligible for request or transmission. A policy vector $\boldsymbol{\pi} = [\pi_1 \dots \pi_N]$ is formed by expressing the transmission policy π_l for each data unit l . The optimal policy $\boldsymbol{\pi}^*$ is in theory obtained by minimizing over all policies $\boldsymbol{\pi}$ the Lagrangian cost

$$J(\boldsymbol{\pi}) = D(\boldsymbol{\pi}) + \lambda R(\boldsymbol{\pi}) \quad (1)$$

where $D(\boldsymbol{\pi})$ is the expected distortion and $R(\boldsymbol{\pi})$ the expected rate. But an exhaustive search is computationally too expensive because it is exponential with the window size. It is also exponential with the policy length, which is the number of future transmission opportunities examined in order to reach a decision at the present time. The Iterative Sensitivity Adjustment (ISA) algorithm proposed in [1] solves this difficulty. Starting with an initial policy $\boldsymbol{\pi}$, it considers one data unit l at a time and optimizes its policy π_l , the other data unit policies being fixed. When this is done, another data unit is considered and this process is repeated until the total Lagrangian cost $J(\boldsymbol{\pi})$ converges. The policy π_l is optimized for the Lagrangian cost:

$$J_l(\pi_l) = \epsilon(\pi_l) + \lambda' \rho(\pi_l) \quad (2)$$

where $\epsilon(\pi_l)$ is the error probability, $\rho(\pi_l)$ the transmission cost and λ' determines the trade-off between rate and distortion. It is shown in [1] that $\lambda' = \lambda B_l / S_l$ where B_l is the packet size, and the sensitivity S_l characterizes the importance of the packet. A large value for S_l identifies l as an important packet and corresponds to a small λ' . Therefore, larger transmission rates for l are allowed for a constant cost $\lambda' \rho(\pi_l)$.

Work in [1], [2] assign a single deadline to each data unit. Precisely, the proxy considers that a media packet arriving after its decoding time is useless and adapts its policy accordingly. But even if a data unit arrives late, it can still be used by the decoder in an Accelerated Retroactive Decoding (ARD) scheme as shown in [4]. The frame corresponding to this late packet is shown using error concealment with lower quality, but the following frames can still be correctly displayed if this late packet arrives early enough for its dependent packets

to use it. Therefore, for this ARD scheme, work in [1], [2] misestimate the distortion and the packet sensitivity. Recent work in [3] associates multiple deadlines to each data unit l . They correspond to the decoding time of each frame that potentially uses l . With this new scheme, the transmission policy is more efficient and gains in rate or PSNR are achieved.

II. MATHEMATICAL ANALYSIS

Let us consider a data unit l and its associated deadlines. Our analysis here extends the formulation in [2] to use multiple deadlines. Starting from a known initial state q_0 , the proxy makes transmission decisions according to its policy π_l until it reaches a final state. These final states are the states for which an ACK has been received and the states that reach the last transmission slot defined by the policy length. Surprisingly, this definition of final states does not depend on the deadlines, and we will use this property to compute the Lagrangian cost recursively.

Let us look at the example where in the initial state at time α_0 , the proxy does not have the data unit l . There are 2 possible actions for the proxy: requesting this data unit from the server or not requesting it. If a request is sent, there are 2 possible values for q_1 , the proxy state at the next transmission opportunity α_1 : the data unit will either arrive at the proxy or not arrive. If no request has been sent, no data unit will arrive by α_1 so q_1 is known with probability 1. Depending on the proxy state q_1 , i.e. on the observation made by time α_1 , different actions can be taken: send the data unit if it is available, request the data unit if it is not available or do nothing. This action/observation process is carried on until a final state is reached. As mentioned in [2], a policy π_l determines what action to take at each state q_i and therefore induces a Markov chain with transition probabilities $P_{\pi_l}(q_{i+1}|q_i) \triangleq P(q_{i+1}|q_i, \pi(q_i))$. We can deduce that for a policy π_l and a sequence of states q_0, q_1, \dots, q_f , the probability of the final state is $P_{\pi_l}(q_f) = \prod_{i=0}^{f-1} P_{\pi_l}(q_{i+1}|q_i)$. Note that this does not depend on any deadline t_i . The set of policies considered can indeed be extended to inefficient policies that transmit even after the last deadline, and state probabilities can still be computed for these scenarios. But these policies induce a rate cost while failing to reduce the distortion, so they will be discarded by the optimization process.

Let us call the number of requests M and the corresponding request times τ_j . Let us call the number of times the data unit l is sent N and the corresponding sending times t_j . Assume also that the data unit (DAT) is received between the 2 consecutive transmission opportunities α_k and α_{k+1} . Defining the forward trip time on channel 2 (FFT2) as the times between which the data unit is sent by the proxy and received by the client, we can compute the error probability for the state \tilde{q}_f located at the deadline t_i :

$$\epsilon_{\pi_l}(\tilde{q}_f, t_i) = \begin{cases} \prod_{j=0}^{M-1} \frac{P(RTT1 > \alpha_k - \tau_j) - P(RTT1 > \alpha_{k+1} - \tau_j)}{P(RTT1 > \alpha_0 - \tau_j)} \\ \times \prod_{j=0}^{N-1} P(FFT2 > t_i - t_j | RTT2 > t_i - t_j) & \text{if } M \neq 0 \\ \prod_{j=0}^{N-1} P(FFT2 > t_i - t_j | RTT2 > t_i - t_j) & \text{if } M = 0 \end{cases}$$

Let us define $\rho_{\pi_l}(q_f)$ as the transmission cost at state q_f .

$$\rho_{\pi_l}(q_f) = \sum_{i=0}^{f-1} cost(a_i)$$

where

$$cost(a_i) = \begin{cases} 1 & \text{if a data unit is sent at time } \alpha_i \\ P(RTT1 < \infty) & \text{if a request is sent at time } \alpha_i \\ 0 & \text{if nothing is sent at time } \alpha_i \end{cases}$$

The multiple deadline framework developed in [3] associates a set of deadlines $\{t_i\}$ to a data unit l , where i indices the frames that potentially use l . We define the cost function for a policy π_l as:

$$J_l(\pi_l) = \rho(\pi_l) + \sum_{i \in \mathcal{W}} \nu_{t_i} \epsilon(\pi_l, t_i) \quad (3)$$

where $\epsilon(\pi_l, t_i)$ is the probability that, using this policy, the data unit does not arrive before its deadline t_i , $\rho(\pi_l)$ is the expected number of transmissions, and \mathcal{W} is the transmission window. The ν_{t_i} 's are Lagrangian multipliers dependent on the sensitivity and hence on t_i .

Note that $J_l(\pi_l)$ can be rewritten as a summation over the final states q_f :

$$J_l(\pi_l) = \sum_{q_f} P_{\pi_l}(q_f) J_{\pi_l}(q_f) \quad (4)$$

where $J_{\pi_l}(q_f) = \rho_{\pi_l}(q_f) + \sum_{i \in \mathcal{W}} \nu_{t_i} \epsilon_{\pi_l}(q_f, t_i)$.

In the case when the state q_f is reached by the deadline t_i , $\epsilon_{\pi_l}(q_f, t_i)$ can be interpreted naturally as the error probability at state q_f . In the case when the state q_f is reached after the deadline t_i , $\epsilon_{\pi_l}(q_f, t_i)$ is equal to the error probability at state \tilde{q}_f where \tilde{q}_f is the ancestor of q_f located at time t_i . We can now recursively compute the expected Lagrangian cost for a path through q_i :

$$J_{\pi_l}(q_i) = \begin{cases} \sum_{i \in \mathcal{W}} \nu_{t_i} \epsilon_{\pi_l}(q_f) + \rho_{\pi_l}(q_f) & \text{if } i = f \text{ (final state)} \\ \sum_{q_{i+1}} P(q_{i+1}|q_i, \pi_l(q_i)) J_{\pi_l}(q_{i+1}) & \text{otherwise} \end{cases}$$

Note that $J_l(\pi_l) = J_{\pi_l}(q_0)$ so the optimal policy can be computed efficiently using dynamic programming.

Since the client does not see the request process and is only affected by data unit receptions, the framework developed in [3] for the state dependent distortion still holds. We invite the reader to refer to this paper for more details.

III. EXPERIMENTAL RESULTS

The test sequence is *Mother-Daughter* with a two layer representation (base and enhancement layers). The error concealment scheme is a nearest frame substitution. In order to model each channel, we have to characterize the delay and the loss. A shifted Γ distribution is used to represent the delay. Its probability density function is therefore parameterized by three values: κ the amount of shift, the mean, and the standard deviation. Moreover, we fix the loss probability ϵ to be a constant. We also assume independent losses and delays between the packets (there are no error bursts for example). We did not actually implement the recursive algorithm described above, but used instead a full search method over the appropriate space to determine the optimal policy for each data unit.

We conducted a first set of experiments using a proxy server scenario with 2 schemes: multiple deadlines (with ARD) and single deadline (with ARD). The parameters for channel 1 are kept constant, $\epsilon_1 = 10\%$, $mean = 50ms$, $standard\ deviation = 23ms$, $\kappa_1 = 10ms$, as are $mean = 10ms$, $standard\ deviation = 5ms$, $\kappa_2 = 5ms$ for channel 2. But the loss probability for channel 2 varies for each experiment: $\epsilon_2 \in \{1\%, 5\%, 10\%\}$. The policy length is equal to 3. Transmission opportunities are scheduled every 50 ms, and the end-to-end delay d is fixed to 300 ms. We did 20 to 30 experiments to generate the outcomes for each rate. The results are shown on Fig. 1.

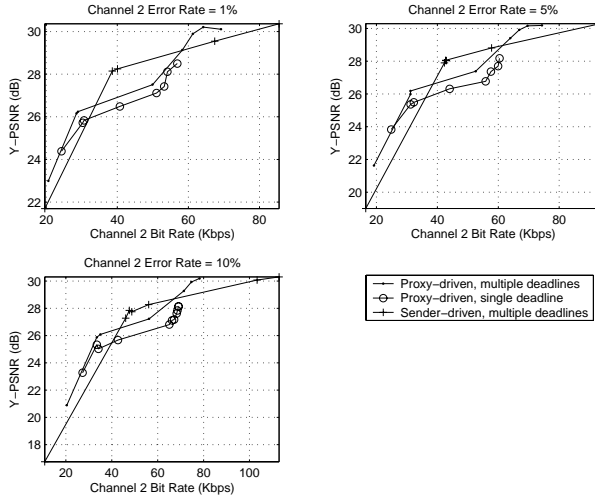


Fig. 1. Comparisons between the different schemes for different error probabilities on channel 2.

Because of the small number of seeds used and the small policy length, the results discussed here should be considered

as trends that need to be confirmed by additional experiments. We notice a step in these curves that indicates a rate threshold above which the PSNR increases. This might be due to a more extensive use of the enhancement layer at high rates whereas for low bit rates, this layer is dropped. The sender driven case performs better between 30 to 60 Kbps. But this difference decreases as the channel 2 error rate increases. Indeed, requests over channel 1 impose additional delays because the proxy needs to wait for the packet. This delay can be compensated only if there are many transmissions over channel 2 that use the cached version of the packet at the proxy.

We conducted a second set of experiments where we compared the proxy scenario to the sender driven case. Multiple deadlines were used for both. The same channel parameters from the first experiment were used except for the loss probability on channel 2, which is fixed at 5%. The end-to-end delay d takes two values: 200 or 300ms. Fig.2 shows that for a low probability of error on channel 2, the sender driven case performs better in the low latency case. The explanation is the same as above: less time spent over the backbone network in the sender driven case compared to the proxy case.

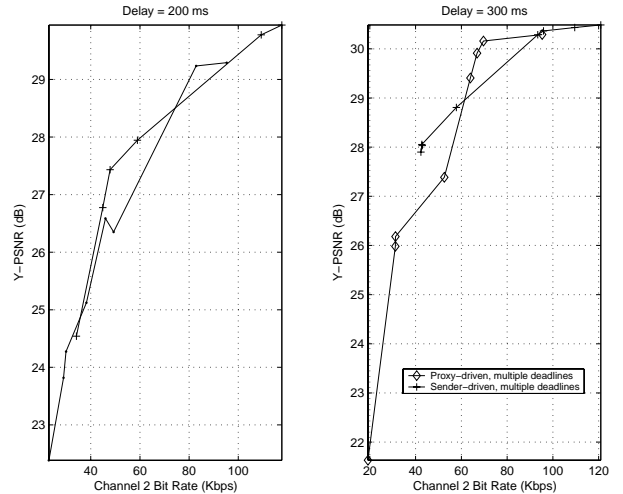


Fig. 2. Proxy and sender driven scenarios with multiple deadlines for different latencies.

Our last experiment uses the previous data at $d=300ms$ to estimate the standard deviation of the PSNR at a given rate. Each PSNR measurement can be considered as a random variable and a low standard deviation would suggest that there are no large variations in visual quality for the decoded video sequence. Because multiple deadlines make better use of ARD, it performs better than the single deadline scheme as can be seen on Fig.3.

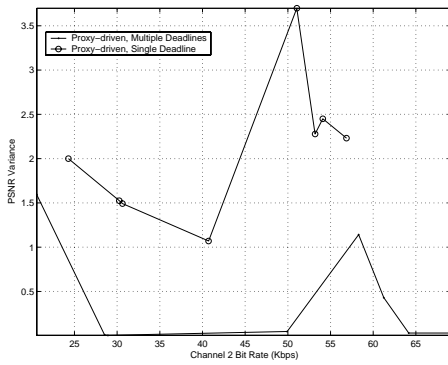


Fig. 3. PSNR standard deviation comparisons.

IV. CONCLUSION

In this work, we have applied the multiple deadlines scheme to the hybrid sender/receiver driven scenario, where a proxy server acts as an intermediary between a server and a client. We have rederived the mathematical analysis necessary to apply the ISA algorithm to find the optimal transmission policy. We have also described the different experiments we conducted and explained the results. Additional experiments are needed to clearly evaluate this scheme.

ACKNOWLEDGMENT

The authors wish to thank Mark Kalman and Jacob Chakareski for discussions during the course of this project.

REFERENCES

- [1] P.A.Chou and Z. Miao. Rate-Distortion Optimized Streaming of Packetized Media. Microsoft Research, Technical Report, MSR-TR-2001-35, February 2001.
- [2] J. Chakareski, P.A. Chou, and B. Girod. Computing rate-distortion optimized policies for hybrid receiver/sender driven streaming of multimedia. *Proc. Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, Nov. 2002. Invited Paper.
- [3] M. Kalman and B. Girod. Rate-Distortion Optimized Video Streaming with Multiple Deadlines for Low Latency Applications, *Proc. ACM Multimedia 2003*, Berkeley, CA, Nov. 2003. Submitted.
- [4] M. Ghanbari. Postprocessing of late cells for packet video. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(6):669-678, December 1996.

V. APPENDIX

Eric Thea

Paper review

Theoretical analysis

Feedback on coding

Debugging

Report and presentation

Chung-Tse Mar

Paper review

Feedback on theoretical analysis

Coding and debugging

Feedback on report and presentation