ENCODER BUFFER CONSTRAINTS FOR VIDEO TRANSMISSION OVER NETWORKS WITH NO QUALITY-OF-SERVICE GUARANTEES

Hayder Radha$^{\$}$ *and Dmitri Loguinov**

***City University of New York**
New York, NY, 10016
csdsl@cs.ccny.cuny.edu

$^{\$}$ **Michigan State University**
East Lansing, MI 48824
radha@egr.msu.edu

Abstract

Real-time transmission of video over networks with no QoS guarantees (e.g., the Internet) is increasingly becoming an important application area in multimedia communications. One of the key challenges in this area is maintaining continuous decoding and playback at the receiver despite severe network impairments such as high packet-loss-ratios, packet-delay-variations, and unbounded roundtrip delays. A popular solution to this problem is to introduce large buffering delays at the receiver to (partially) compensate for the network impairments. This solution, however, is not a viable option for clients with limited resources and for applications requiring minimal delays. In this paper, we advocate a new approach to this challenge. We derive a new set of encoder-buffer constraints taking into consideration network impairments. We show how these new constraints can significantly reduce (or possibly eliminate) underflow and overflow events at the decoder while enabling the receiver to maintain its ideal decoder-buffer size and buffering-delay requirements (needed for video transmission under ideal conditions). We present the results of more than 40 hours of video transmission over the (real) Internet. These results demonstrate the advantages and limitations (asymptotic behavior) of our proposed encoder-buffer constraints.

1. Introduction

Transmission of video over networks with no Quality-of-Service (QoS) guarantees has become a critical technology area for many multimedia applications. The importance of this area is mainly due to the explosion in Internet usage over the past few years. When transmitting video over such networks, one has to design a solution that takes into account packet-loss events and other network-delay impairments. In order to overcome the relatively common packet-loss events, packet-recovery mechanisms, such as negative-acknowledgements (NACKs) with retransmission or a hybrid of Forward-Error-Correction (FEC) and retransmission schemes, are usually employed to recover the lost video in real-time. Overall, for streaming of video over the Internet, retransmission based schemes provide a bandwidth-efficient approach for recovering lost packets [5][6].

One of the objectives of any real-time loss-recovery mechanism (including retransmission based methods) is to provide *timely delivery* of the lost video-data to the receiver. In other words, the lost-data has to arrive prior to the time when the corresponding video is needed for decoding and display at the receiver. This objective is usually hindered because of the excessive roundtrip delay and variation in the delay (i.e., packet-delay-variation – PDV) encountered over the Internet [1][4].

Consequently, packet-loss recovery solutions, in general, and retransmission based schemes in particular employ some type of a transport-layer buffering mechanism to facilitate the timely-delivery of the lost-data to the video-layer. In addition to absorbing network PDVs, extra buffering delays are usually needed to provide enough time for (a) the receiver to send the NACK messages to the source that holds the lost packets, and (b) the source to retransmit the desired data to the receiver. All of these delays, which are needed for detecting and recovering lost data, are normally added to video-buffering delays used by the encoder and decoder. Extra delays at the receiver imply additional resources (e.g., memory) for storing and managing the data in real-time. For many applications, which need to adhere to strict receiver-resources and end-to-end delay requirements, adding extra buffering at the receiver may not be desirable or even feasible in some cases.

In this paper, we present a new approach for dealing with this problem. Here, we advocate the notion of imposing new buffering constraints on the video-encoder to guarantee a minimal level of “resources” for the receiver throughout a real-time video session. To clarify this notion, let’s first consider the ideal network case. Under ideal transmission scenarios (i.e., without packet-losses, packet-delay-variations, etc.), each video stream requires certain (ideal) decoder "resources": a decoder-buffer size ($B^d_{\max}$) and a start-up delay[1] (ΔT). Our objective is to help the receiver avoid increasing its (ideal) buffering and start-up delay requirements while providing it with the ability to compensate for network impairments. This is accomplished by deriving encoder-buffer constraints which are functions of network parameters such as packet-delay-variations – PDVs – and roundtrip time delays – RTTs. When the encoder adheres to the proposed constraints, the decoder can significantly reduce (or eliminate) underflow and overflow events due to network impairments while relying on the ideal buffer-size $B^d_{\max}$ and the ideal start-up delay. We demonstrate the effectiveness and limitations of our proposed encoder-buffer-constraints by presenting the results of more than 40 hours of video streaming tests conducted over the (real) Internet.

2. Encoder Buffer Constraints as Functions of Network Parameters

First, we present the necessary notations, and briefly cover the ideal encoder-buffer constraints as presented in previous works [2][6][7]. Let Δ be the end-to-end delay (i.e. including both encoder and decoder buffers, and the channel delay δ_c) in units of time. For a given video coding system, Δ is a constant number that is applicable to all pictures entering the encoder-decoder buffer model. To simplify the discrete-time expressions, it is assumed that the end-to-end buffering delay $\Delta T = \Delta - \delta_c$ is an integer-multiple of the frame duration T. Therefore, $\Delta N = (\Delta - \delta_c) / T$ represents the buffers' delay in terms of the number of video pictures.

Let $r(i)$ be the data rate[2] at the output of the encoder-buffer during frame-interval i. The (ideal) encoder-buffer lower bound (*ELB*) and upper bound (*EUB*) at time index n depend on the rate-function $r(i)$, the delay ΔN, and the encoder and decoder buffer sizes [2][7]:

$$ELB\left(n\,;r,\Delta N,B^d_{\max}\right) \le B^e(n) \le EUB\left(n\,;r,\Delta N,B^e_{\max}\right), \quad (1)$$

where $B^d_{\max}$ and $B^e_{\max}$ are the maximum decoder and encoder buffer sizes, respectively.

[1] Throughout this paper, we consider decoder buffer delays as "resources". A delay is a "resource" since it can be used to compensate for network jitter and/or recover lost packets (e.g., through retransmission).

[2] Here we use "data rate" in a generic manner, and therefore it could signify "bit", "byte", or even "packet" rate. More importantly, $r(i)$ represents the total amount of data transmitted during period i.

In addition to packet losses, we take into consideration three network-delay parameters to derive the proposed encoder-buffer constraints:

1. Positive packet-delay-variation (PDV$^+$). This parameter measures the amount of *timing-expansion* that some sets of packets may experience through the network [4]. This leads to the arrival of packets later than their ideal arrival time at the decoder buffer.
2. Roundtrip delay (RTT). This parameter measures the amount of time needed for recovering a lost packet (or a set of lost packets) through retransmission.
3. Negative packet-delay-variation (PDV$^-$). This parameter measures the amount of *timing-compression* that some groups of packets may experience through the network [4]. This leads to the arrival of packets earlier than their ideal arrival time at the decoder buffer.

It is important to note that the above three parameters could vary in real-time. Therefore, depending on the application, the encoder can update these parameters in real-time (e.g., based on some feedback from the receiver in a point-to-point application), or the encoder can assume some "reasonable" values for these parameters if the video is coded off-line (i.e., not in real-time). As will be shown in our Internet test results, it is quite feasible for the encoder to use a range of network-parameter values that could reduce underflow events significantly.

The positive PDV and roundtrip delay impact the encoder upper bounds while the negative PDV impacts the encoder lower bounds. Due to space limitations, for the remainder of this paper, we only show the constraints on the encoder-upper-bounds which are needed for avoiding underflow events. We also show the corresponding simulation results. The encoder-lower-bound constraints are very similar in nature, and, therefore, will not be shown in this paper. In addition, the encoder-upper-bound constraints influence the ability of the receiver in recovering lost packets over the network. Consequently, the upper-bound constraints are significantly more critical than the lower-bound constraints from a network packet-loss recovery point-of-view.

2.1 Encoder Buffer Constraints as Functions of Positive PDV and Roundtrip Delays

Let the positive PDV value selected by the encoder be T_L (to compensate for *late*-arriving packets at the decoder). And let the roundtrip value be T_R (needed for retransmission based recovery). Moreover, let $N_L = \lceil T_L / T \rceil$ and $N_R = \lceil T_R / T \rceil$.

The encoder-buffer constraints can be influenced by the retransmission based strategy used for recovering lost-packets. We only consider two general cases. Under the first case, the decoder declares an unreceived-packet "missing" (and consequently requests retransmission) immediately after the expected arrival time of that packet. In other words, under this scenario, the receiver ignores packet-delay-variations, and only concentrates on recovering (potentially) lost packets. Therefore, in this case, retransmitted packets can represent either truly lost packets or duplicates of packets that do arrive after the retransmission request is sent. Although this strategy can generate duplicate packets, it generally increases the probability of receiving all needed packets on time. Under this strategy, the encoder buffer constraints can be shown to be:

$$B^e(n) \le EUB(n) - \left\{ \sum_{j=n+1}^{n+\Delta N} \delta r'(j) + \sum_{j=n+\Delta N - N_R + 1}^{n+\Delta N} r'(j) \right\}, \qquad (2)$$

where *EUB* is the encoder-upper-bound in the ideal case, and $r'(j)$ is the adjusted (lower) encoder output bitrate taking into consideration retransmitted packets due to actual losses and duplicate packets. Therefore, $\delta r'(j) = r(j) - r'(j)$ is the amount of decrease in the bitrate.

Under the second retransmission strategy, the decoder declares an unreceived-packet "missing" after waiting some time-period following the expected arrival time of that packet [6]. This waiting time-period equals to the selected positive PDV value T_L. In this case, retransmission of duplicate-packets is virtually eliminated. It can be shown that this strategy gives the following upper-bound:

$$B^e(n) \le EUB(n) - \left\{ \sum_{j=n+1}^{n+\Delta N} \delta r''(j) + \sum_{j=n+\Delta N - (N_L + N_R) + 1}^{n+\Delta N} r''(j) \right\}. \qquad (3)$$

This expression can be simplified as follows. It is common for streaming applications to use bitrates $r(j)$ taking into consideration retransmissions due to packet losses (but not due to duplicates). Since the above expression results in (virtually) duplicate-free retransmissions, it is reasonable to use the approximation $r''(j) \approx r(j)$. Consequently, (3) can be expressed using the original bitrates $r(j)$:

$$B^e(n) \;\le\; \min\left(\sum_{j=n+1}^{n+\Delta N - (N_R + N_L)} r(j) \;,\; B^e_{\max} \right). \qquad (4)$$

The above expression is based on the following two conditions:

$$\Delta N > N_R + N_L, \text{ and} \qquad (5)$$

$$\sum_{j=n+1}^{n+\Delta N - (N_R + N_L)} r(j) \ge \max\left\{ \left(\sum_{j=n+1}^{n+\Delta N} r(j) - B^d_{\max} \right), 0 \right\}. \qquad (6)$$

If the network positive-PDV and RTT are bounded by T_L and T_R, respectively, then the above constraints in (4) – (6) ensure that the decoder buffer will always have enough delay for detecting lost packets and for recovering these packets while maintaining the original end-to-end delay ΔT and the ideal buffer-size $B^d_{\max}$. Consequently, when the encoder adheres to the constraint in (4), any packet, that enters the video decoder buffer without encountering delay-variation or loss, spends a minimum of ($T_T = T_L + T_R$) in the decoder buffer. This naturally leads to lowering the encoder-upper-bound as shown in the example of Figure 1. Therefore, the advantages of our proposed approach come at the expense of constraining the compressed picture sizes which could lead to degradation in video quality. However, since underflow events can result in far more severe degradations, the level of penalty paid by the above constraints can be easily justified when transmitting video over networks with no QoS guarantees.

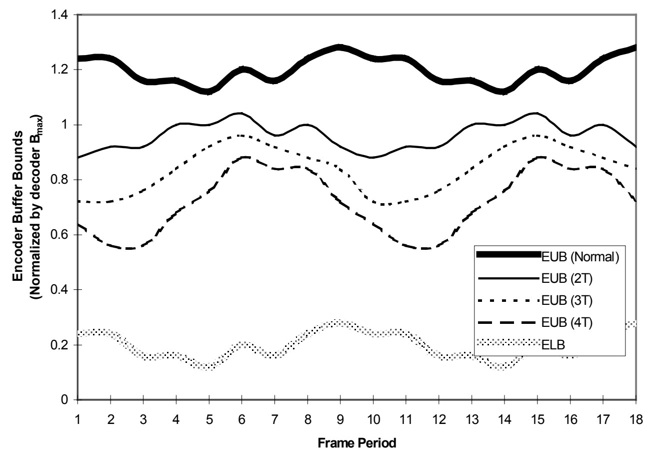

Figure 1: An example of encoder buffer bounds due to data-loss constraints. The normal Encoder Upper Bound (EUB) has been reduced by the new network constraints. The example shows cases when $N_R + N_L$ = 2, 3, and 4. In all cases, ΔN=10.

3. Simulation Results

We evaluated the impact of our proposed approach on reducing underflow events by conducting rather extensive sets of experiments of video streaming over the Internet. All tests were performed using dial-up analog modems through a popular nationwide Internet Service Provider (ISP) with several million active subscribers. The path from the server to the client often contained a large number of end-to-end hops (including backbone routers) and spanned several Autonomous Systems. This setup not only reflects the current use of streaming applications by Internet users, but also allows us to test the proposed scheme over paths with large round-trip delays (RTT) and randomly varying delay jitter (i.e., over paths that are traditionally considered "difficult" for real-time streaming). In this paper, we report the results of two sets of tests, each conducted during different times of the day (i.e., in the morning, afternoon, and at night). Each set consists of more than 20 hours of streaming compressed MPEG-4 video from a server attached to Internet backbone through a T1 circuit and located in Briarcliff Manor, NY. The video stream used in the experiments was coded at very low bitrates (10 – 14 kbit/sec), and the frame-rate used throughout the experiments was five frames-per-second (i.e., $T = 200$ ms).

The first set of tests was conducted by streaming video to dial-up clients that used ISP's access points located in Los Angeles, CA. In these tests, the cross-country path contained 17 hops. The other set of tests was conducted by using a dial-up access point in White Plains, NY, and in this setup, the end-to-end path consisted of 12 hops. We refer to these two sets of experiments as the NY-LA and NY-NY tests, respectively. For each set, we evaluated the impact of employing the encoder buffer constraints as expressed in (3) and (4) by using a range of values for the sum (N_L+N_R). This encoder-constraint range of values corresponds to a minimum decoding-delay guarantee (i.e., $T_T = T_L+T_R$) with values in the range of 500 milliseconds to 2 seconds. The ideal end-to-end delay for the tested streams was in the range of 3–4 seconds. (For very low bitrate coding used in typical Internet streaming applications, these end-to-end delays are quite common.) For recovering lost packets, we employed the retransmission based algorithm described in [6]. This algorithm is consistent with the second packet-loss-recovery strategy covered in the previous section.

As can be seen from Figure 2, even for the 500 ms minimum-decoding-delay constraint, the proposed approach could assist the receiver in recovering about 75% and 90% of the lost packets on-time (i.e., avoiding underflow events) for the NY-LA and NY-NY scenarios, respectively. The other key observation here is that the level of constraints imposed can help to a certain limit (about 1–1.25 seconds of minimum decoding-delay). Beyond this limit, there is not much improvement in reducing underflow events can be anticipated. This phenomenon is due to the fact that although the majority (90-95%) of packet loss events occurred when round-trip delays were below 1 second (i.e., retransmissions were successful within the allocated decoding time guarantee of T_T), the remaining packet loss events occurred during severe congestion within the network, which was accompanied by extremely high RTTs (up to tens of seconds) and a high probability of loss among *retransmitted* packets. Consequently, regardless of the amount of video-buffering delay, very few packets lost during severe congestion were recovered and even fewer were recovered *before* their decoding deadlines. These observations lead to a conclusion that if a packet is to be recovered, it is likely that the packet will be recovered within the first 1200-1500 ms from the time of the first retransmission request. If the packet is not recovered within this timeframe, it is likely it will not be recovered at all.

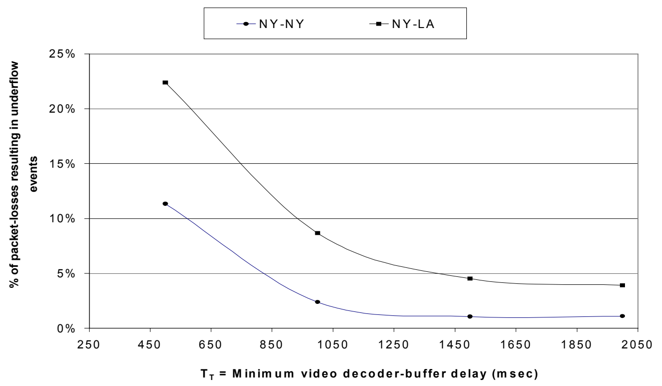

Figure 2: Results of video streaming experiments over the Internet illustrating the impact of imposing encoder-buffer constraints on reducing underflow events due to packet losses. The proposed encoder buffer constraints are expressed in terms of minimum decoder-buffer delay constraints as shown in the figure.

Even though in the experiments presented in this paper we used static values of T_T, future real-time protocols ideally will provide feedback to the server about the current values of the RTT and delay jitter. In its simplest form, the feedback mechanism will carry the most recent estimates of the round-trip delay T_R and delay jitter T_L in each NACK packet, as well as in special keep-alive packets, which should be sent by the client to the server at regular intervals. More sophisticated feedback protocols (such as the ones needed for congestion control) can be effortlessly augmented to carry these two values in each control packet in addition to other feedback information.

The measurement of the first control parameter, T_R, should be performed by the client by timing the duration between the time of requesting a retransmission and the time of the arrival of the corresponding retransmitted packet [3]. During periods of low packet loss, the client may use simulated retransmission requests to sample the round-trip delay as further described in [3]. Generally, the samples of the RTT should be smoothed to produce a more stable estimate, and such smoothed value should be used to derive T_R. Furthermore, the client may use an estimate of the RTO (retransmission timeout) instead of the current estimate of the RTT. The use of the RTO provides better protection against premature retransmission requests, but typically requires larger values of T_T. For more discussion, see [3].

Real-time measurement of the second control parameter, T_L, consist of timing inter-packet delay dynamics of arriving packets and similarly applying a smoothing function to produce a stable estimate. A thorough discussion of the algorithms for selecting the optimal values of the estimated RTT (i.e., T_R) and delay jitter (i.e., T_L) that work best with the scheme proposed in this paper are topics for future work and lie beyond the scope of this paper.

In addition to requesting that the encoder change the decoding delays of frames in a video stream based on the feedback from the client, a real-time streaming application may utilize, for example, various FEC-based error control methods, error-resilient encoding, or error concealment. These approaches are supplemental to ours and provide the grounds to fall back to in case one method is unable to completely recover lost or corrupted frames. Consequently, we consider these methods to be fairly orthogonal to ours, and the study of how these methods can be put together for an optimal overall performance is also beyond the scope of this paper.

4. Conclusion

In this paper, we presented a new approach for dealing with the problem of packet loss recovery and delay-jitter encountered when transmitting real-time video over networks with no QoS guarantees. Here, we advocate the notion of imposing new buffering constraints on the video-encoder to guarantee a minimal level of delay "resources" for the receiver throughout a real-time video session. These "resources" represent minimum decoding delays that enable the receiver to recover lost packets and to absorb packets experiencing "time expansion" over the network (i.e., arriving later than expected). We showed new encoder buffer constraints that take into account round-trip delays over the network (for NACK based packet-loss recovery) and Packet-Delay-Variation (PDV or delay jitter).

Our extensive simulation results showed that this approach could be effective in recovering lost packets and minimizing underflow events. We also showed that the amount of constraints (or equivalently the amount of decoding delay) needed can be bounded to relatively small values for packet loss recovery over the Internet. Based on our experience, large buffering delays may not be necessary due to a "thresholding phenomena" that occurs during network congestions. One of our key observations is that beyond a certain level of constraints/decoding-delays, the underflow events cannot be reduced when excessive network congestions occur. During such congestions the roundtrip delay between the client and server is very high, and consequently, regardless of the constraints/decoding-delays used one cannot recover lost packets through retransmission.

One of the issues not addressed in this study is the impact of this approach on the overall video quality of the sequence. It is clear that imposing buffer constraints on video sequences imply some degradation in quality. The tradeoff between (a) encoder-buffer-constraints that enable packet loss recovery and (b) the degradation in video quality due to these constraints represent an interesting problem for future work.

References

[1] M.Allman and V.Paxson, "On estimating end-to-end network path properties," Proc. ACM SIGCOMM'99 Conf., Cambridge, Mass., Sept 1999, vol. 29, no. 4, pp 263-274, October 1999.

[2] T.V.Lakshman, A.Ortega and A.R.Reibman, "VBR video: Tradeoffs and potentials," Proceedings of the IEEE, vol. 86, no. 5, pp. 952-973, May 1998.

[3] D. Loguinov and H. Radha, "On Retransmission Schemes for Real-time Streaming in the Internet," IEEE INFOCOM, 2001.

[4] V.Paxson, "End-to-end Internet packet dynamics," Proc. ACM SIGCOMM'97 Conf., Cannes, France, Sept 1997, vol. 27, no. 4, pp 139-52, October 1997.

[5] C.Perkins and O.Hudson, "Options for repair of streaming media," IETF Network Working Group, RFC: 2354, June 1998.

[6] H.Radha, Y.Chen, K.Parthasarathy and R.Cohen, "Scalable Internet video using MPEG-4," Signal Processing: *Image Communication*, vol. 15, pp. 95-126, September 1999.

[7] A.R.Reibman and B.G.Haskell, "Constraints on variable bit-rate video for ATM networks," IEEE Trans. Circuits System and Video Technol., vol. 2, pp. 361-372, December 1992.