CROSS LAYER TECHNIQUES FOR ADAPTIVE VIDEO STREAMING OVER WIRELSS NETWORKS[1]

Yufeng Shan and Avideh Zakhor

Department of Electrical Engineering and Computer Science,
University of California, Berkeley CA 94720
{yfshan, avz}@eecs.berkeley.edu

ABSTRACT

Wireless multimedia delivery is becoming increasingly more important in today's networks. Unlike wired packet switched networks that suffer from congestion related loss and delay, the wireless networks have to deal with a time varying, error prone, physical channel that in many instances is also severely bandwidth constrained. As such, the solutions needed for wireless video streaming applications are fundamentally different from wired streaming. In this paper, we propose a set of end to end application layer techniques for adaptive video streaming over wireless networks. The adaptation is done both with respect to channel and data. Our approach combines the flexibility and programmability of application layer adaptations, with low delay and bandwidth efficiency of link layer techniques. Socket level simulations are presented to verify the effectiveness of our approach.

1. INTRODUCTION

Streaming media over wireless networks is becoming an increasingly important application, with a large body of literature [1 – 5]. For instance, [1] proposes a two-step adaptive hybrid Automatic Repeat request (ARQ) scheme for transmitting H.263 video sequences over wireless LANs. An Unequal Error Protection (UEP) based error control scheme for transmission of low bit rate MPEG-4 video over wireless channels is proposed in [2], where MPEG-4 bit stream is divided into two classes, and UEP is applied to each class of data. [3] proposes a solution for video transmission over wireless networks by combining the efficiency of DSP chips with the error resilience of MPEG-4 bit stream. The proposed model in [4] covers the complete transmission system.

In wireless environments, the channel conditions change rapidly over time due to noise, interference, multi-path, and the movement of the mobile host. In such a context, transmission control schemes have to dynamically adapt both to the application requirements, and to the channel conditions. In this paper, we propose a set of building blocks for channel and application adaptive wireless streaming applications. In doing so, we exploit two well known principles in wireless video communication: First, has to do with the fact that different parts of a video bit stream are of different importance, and hence need to be protected via Forward Error Correction (FEC) and ARQ to different degrees; second, has to do with adaptivity to channel conditions by dropping unimportant packets. While UEP for video bit streams has been around for a long time, most such techniques are applied at the bit level, rather than packet level. Since most of the today's wired and wireless networks are packet oriented, our approach in this paper is unequal treatment of packets and adaptivity at the packet level. Given this, the main issue that arises is: which network protocol layer this unequal treatment of video, and adaptivity needs to be done at? Is it best to do it at application layer or at link layer? Clearly, there are pros and cons for either case. At lower layers, such as link layer, implementation complexity to adapt to packet importance becomes an issue. In addition, many link layer protocols in today's networks do not support a mechanism for different FEC protection or retransmission policies for differently marked Radio Link Protocol (RLP) packets. This would motivate one to apply unequal FEC and retransmission techniques at the application layer. However, applying FEC at the application layer results in excessive delays, if redundancy is applied across different packets. Also, adaptive retransmissions from application layer have the inherent disadvantage of additional delay. Specifically, the packets at application layer tend to be larger and hence, if existing transport protocols such as TCP are used, unless all RLP packets within the TCP packet are received successfully with fewer than maximum number of retransmiss-ions allowed at the link layer, the entire application layer packet needs to be retransmitted, resulting in excessive delay and waste of bandwidth. In the case of UDP, unless all link layer packets are received successfully, the entire UDP packet is lost.

Based on the above considerations, our proposed building blocks for channel and application adaptive scheme are as follows:

1. Priority based ARQ, together with a scheduling algorithm is applied at the application layer; however, in order to avoid excessive delays, UDP lite is employed at transport layer, so that corrupted radio link layer packets can be passed along to the application layer at the receiver. This way, the receiver can ask the transmitter to re-send only the corrupted RLP packets within an application layer packet. To facilitate the process of specifying the exact RLP packet to be re-sent, the application layer packet is constructed in such a way that an exact integer number of RLP packets, say M, is generated after decomposition of any application layer into RLP packets. This has two advantages: (a) the receiver only sends an index to sender as to which of the M packets were corrupted; (b) no modifications are needed at the link layer.

2. Furthermore, to avoid excessive delay introduced due to FEC coding across application layer packets, FEC is implemented within an application packet, but not at bit level,

[1] This work is sponsored by AFOSR Award FDF49620-99-1-0062

rather at RLP packet level. Specifically, video data is first divided into four classes based on their importance; an UEP algorithm described in Section 3 is applied to distribute FEC bits among the four classes; finally packetization at application layer is done in such a way that (a) a given application packet contains data and FEC for only one class, (b) after the application layer packet is decomposed into RLP packets, each RLP packet corresponds to either data or FEC. At the receiver, once the UDP lite passes on the RLP layer packets to the application layer, the application layer can decide whether or not retransmissions are needed on any portion of the application layer based on its awareness of the particular FEC used for that class.

The above proposed scheme has the advantage that all the processing is done end to end at the application layer; yet, FEC and ARQ granularity are at the RLP packet size, thus reducing delay, overhead and waste of bandwidth.

This paper is organized as follows: A packetization scheme for the application layer is discussed in Section 2; in Section 3, class-based UEP is discussed; priority-base ARQ scheme is discussed in Section 4; simulations are presented in Section 5, followed by conclusions in Section 6.

2. APPLICATION LAYER PACKETIZATION SCHEME

The basic idea of our packetization scheme is to construct the application layer packet in such a way, that it is decomposed exactly into an integer number of equal sized RLP packets.[2] At the receiver side, the UDP lite protocol passes the corrupted data to the application. The application layer at the receiver side can easily identify which RLP packet is lost by analyzing the received application packet, and hence request for retransmission. At the RLP layer at the sender, the application packet with all the header information is divided into M data blocks shown as "B" in Figure 1, with their size equal to the payload of RLP packet. Our proposed packetization algorithm at the sender calculates the number of bytes that should be included in the payload of the application packet using the following:

$$payload+sizeof(Header) = M*sizeof(RLP\ payload) \quad (1)$$

where M is an integer, and p*ayload* is the payload of the application packet as shown in Figure 2.

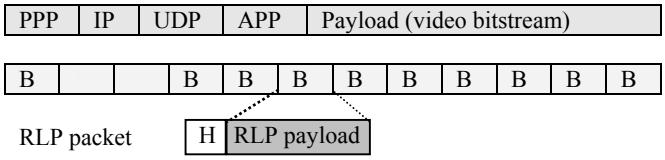


Figure 1: *Packetization scheme in the application layer*

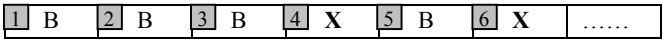


Figure 2: *Payload structure of the application packet*

Every data block of payload has a 4 bits sequence number to indicate the position of the block in the application packet payload. The receiver may receive a packet with corrupted or lost blocks in it, as shown in Figure 2, where the fourth block and the sixth block are lost. The application analyzes the received application packet, identifies lost blocks and based on its knowledge of FEC level for that class of data decides whether any retransmissions are needed. If so, the 4 bits sequence number in Figure 2 is used to indicate the RLP packet payload.

Due to UDP lite properties, one RLP layer packet loss corresponds to one data block loss in the application packet. The receiver can thus estimate the link layer packet loss rate by analyzing the lost blocks in the application packets. This information can then be fed back to the sender to determine optimal FEC levels for a given channel condition. We argue that this is a more accurate method to estimate the channel condition and loss rate at the RLP layer than the traditional application layer estimation techniques. This is because traditional schemes observe only one application packet loss regardless whether one or more RLP packets are lost. Suppose decomposition of header information in application layer onto RLP packet results in p RLP packets, and decomposition of the video data results in q RLP packets. After N application packets, using our proposed scheme, the RLP packet loss rate, P_{BLER}, can be estimated at the receiver by:

$$P_{BLER} = \frac{\sum_{i=0}^{i=N-1} L_i}{N \times (p+q)} \quad (2)$$

where L_i is the number of lost RLP packets within ith application packet. Since the loss of any header information in the RLP layer may cause the entire application packet to be lost, in this case, we use the following equation to estimate the L_i.

$$L_i = \frac{L_{i-1} + L_{i+1}}{2} \quad (3)$$

Finally, successive application packets loss means a long burst loss in the channel; so for this situation, we choose to estimate the L_i as follows:

$$L_i = p + q \quad (4)$$

Whenever an application packet arrives at the receiver side, the receiver estimates L_i and P_{BLER} based on Equations (2)– (4), and feeds this information back to the sender. The sender uses this information to determine how much FEC should be added to each class of data.

3. CLASS-BASED UNEQUAL ERROR PROTECTION

The encoded MPEG-4 data in a bitstream includes interleaved header, shape, motion, and texture information. We classify the MPEG-4 bitstream according to the importance of data and dependency of frames as follows:

Class 0: header information.
Class 1: I and P frames with scene change.
Class 2: Shape and motion information of P frames.
Class 3: Texture information of P frames.

UEP has been proved to be an efficient way to protect different classes of data with different amounts of redundancy [2],[4]. In our scheme, a block-based Reed-Solomon (RS) code is applied as FEC over different classes of data. The RS encoder chooses k video data blocks and generates *(n-k)* parity blocks. The video

[2] In doing so, we assume that the mobile host can obtain RLP packet size via negotiations with the base station at the beginning of the streaming session.

and parity blocks are packetized into one application packet as payload in such a way that once application packet is decomposed into RLP packets, each RLP packet either corresponds to data or parity. Also each application packet corresponds to only one class of data. Every data block has its own block sequence number as shown in Figure 2. The block sequence number is useful at the receiver side in that provides the RS decoder the position of the lost block. The RS decoder can then recover up to *(n-k)* lost blocks with this position information instead of recovering *(n-k)/2* lost blocks without the position information. Given a target application packet loss rate P_{loss}, the estimated P_{BLER} from receiver, and fixed n, the upper bound on k can be computed using:

$$P_{loss} = \sum_{i=n-k+1}^{n} \binom{n}{i} P_{BLER}{}^{i} (1 - P_{BLER})^{n-i} \quad (5)$$

The UEP can be achieved by selecting different k for different classes. The bandwidth allocated to FEC, B_{FEC}, is shared among all the data classes as shown below:

$$B_{FEC} = \sum_{i=0}^{i=C-1} B_{FECi} \quad (6)$$

where C is the total number of classes, B_{FECi} is the bandwidth allocated to class i, and B_{FEC} is the total bandwidth allocated to FEC. We use a simple method to allocate FEC bandwidth to different classes at the sender side as shown in Figure 3. For simplicity, we assume that all data classes have a known, predefined, required loss rate, specified by the user. We start by allocating enough bandwidth to the most important class until the desired packet loss rate is achieved. We then successively move on to the less important classes until FEC budget is exhausted.

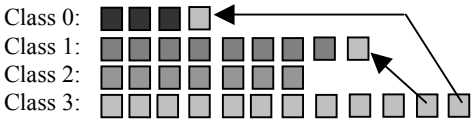


Figure 3: *The FEC bandwidth allocation scheme*

Table 1 shows the PSNR of a video sequence described in detail in Section 6, using an UEP scheme as described above, and equal error protection (EEP) scheme. Block loss rate, P_{BLER}, varies from 3% to 12%. 10% of total bandwidth is allocated to FEC. As expected, UEP performs considerably better than EEP.

BLER	3%	6%	9%	12%
EEP	27.985	22.356	19.816	17.320
UEP	33.956	30.865	29.137	27.540

Table 1: *PSNR using EEP vs UEP*

4. PRIORITY-BASED ARQ (P-ARQ)

P-ARQ is a hybrid-ARQ scheme taking into account different priorities of different data classes. Based on our packetization scheme, several blocks in the received application packet shown in Figure 2 may be lost. If the RS decoder cannot recover the errors, P-ARQ algorithm is run to decide whether or not to send an ARQ request. There are two steps for the P-ARQ algorithm to decide whether to retransmit a lost block. Both steps ensure that the retransmitted packets are not "late" by the time they arrive at the receiver. In the first step, the receiver checks:

$$RTT < T_{buff} \quad (7)$$

to decide whether to send an ARQ request packet. In the above equation RTT is the round trip time, and T_{buff} is the display deadline of the frame with the lost block. If the condition in Equation (7) is satisfied, an ARQ request packet is sent back with the information specifying location of the lost blocks and T_{buff}. In the second step, the sender analyzes the ARQ request packet and checks the following conditions in order to decide whether a packet is to be re-sent:

$$\eta > 0 \quad (8)$$

$$RTT + T_{sender} < T_{buff} \quad (9)$$

η is the remaining number of retransmissions for the lost block, and T_{sender} is the time difference between the time the sender receives the ARQ request until it re-sends out the lost block. The parameter η is initially set based on the class of data block, and is decremented whenever the data block is retransmitted. Specifically, initially η is set to 0,1,2,3, for classes 3,2,1,0 respectively. For an ARQ request, if the conditions in Equations (8) and (9) are satisfied, the lost block is eligible for retransmission.

Suppose the number of lost blocks in an application layer packet is R. Since there is a level of redundancy among the RLP packets in an application layer packet, in many situations the sender does not need to re-send all these R blocks. Specifically, it suffices for the sender to send *[R-(n-k)]* packets with an additional level of *n/k* FEC redundancy, i.e,

$$T = (R - (n-k)) \times \frac{n}{k} \quad (10)$$

After selecting blocks to be retransmitted, the sender packetizes them together with other video data to make an application packet.

The receiver sends a P-ARQ request packet only after it receives an entire application packet. The delay induced by the application layer feedback is larger than that induced by RLP layer feedback. Specifically, if the i-th RLP packet among the total of *(p+q)* packets is lost, induced delay is given by:

$$t_{delay} = \frac{(p + q - i) \times sizeof(RLP)}{B} \quad (11)$$

where *sizeof(RLP)* is the RLP packet size in bits, and B is the channel bandwidth in bits/s.

5. SIMULATIONS AND DISCUSSION

We evaluate our proposed scheme using simulations on a socket-based test-bed. We have encoded the video sequence "forman" in CIF format, at 7.5 frames per second, with MPEG-4 encoder using one I frame followed by 14 P frames in one GOP. We have chosen quantization step size of 8 and video packet length of 960 bits. Packet-based RS code is used as FEC. Every application packet is mapped onto 12 RLP packets at link layer. The network bandwidth is 256Kbps.We assume the round trip time is 300ms, and the reverse channel bandwidth is always available for the negative acknowledgements (NAK) and ARQ requests.

5.1 Comparisons against traditional schemes

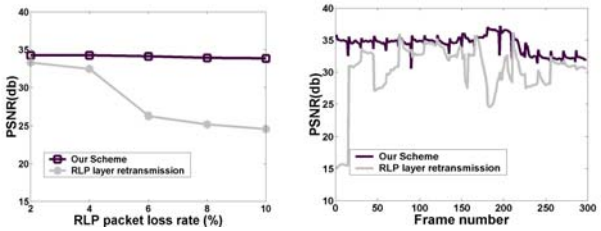
In this section, we compare our scheme against traditional schemes, where the network does not pass the corrupted data along to the application layer. In this case, one RLP packet loss may cause the loss of entire application packet, resulting in the retransmission of the entire application packet. We show the effectiveness of our scheme in bandwidth utilization assuming (a) no retransmission at RLP layer, and (b) maximum of one retransmission for every application packet. Every application packet consists of 10 blocks of data in its payload. Table 2 shows the retransmission bandwidth utilization of our scheme versus the traditional scheme under different channel conditions, with RLP layer packet loss rate ranging from 3% to 12%. Each figure in the table indicates the ratio between number of retransmitted blocks in traditional scheme and the number of retransmitted blocks in our scheme. As seen our scheme reduces the retransmission bandwidth by a factor of 3 to 5.

	3%	6%	9%	12%
RS(10,9)	4.37	3.65	3.19	2.74
RS(10,8)	4.62	3.90	3.68	3.11

Table 2: *The retransmission bandwidth of our scheme versus traditional scheme*

5.2 Comparisons against RLP layer retransmission scheme

Our proposed scheme in this paper can adaptively transmit different classes of data according to the RLP packet loss rate, receiver buffer size and priority of classes. Specifically, the sender selectively drops lowest priority of data in poor channel conditions when the display deadline cannot be met. Alternatively, consider an RLP Layer Retransmission (RLR) scheme in which the RLP layer does not adapt itself to the network conditions, and keeps retransmitting up to a maximum number of allowed retransmissions. The following simulations show that our scheme outperforms RLR scheme. In our scheme we assume application layer FEC adaptation to channel conditions, application layer retransmissions based on data classes, and no RLP level retransmissions. In RLR, we assume a maximum of 3 RLP layer retransmissions, RS(10,9) for FEC at application layer, and no adaptation in the application layer.

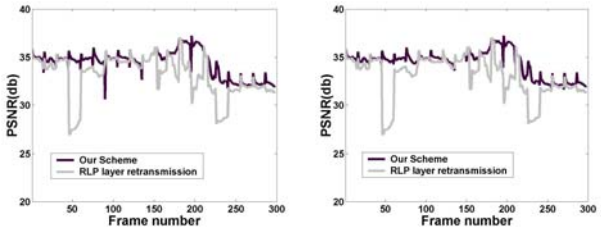


(a) (b)

Figure 4: *The video quality of our scheme versus RLR. (a) Average PSNR of the test video sequence at different RLP loss rates. (b) PSNR as a function of frame number at RLP loss rate 6%.*

Figure 4 shows the video quality using our scheme versus RLR under different packet loss conditions. In our scheme, since lowest priority packets are dropped in poor channel condition, only a few frames are affected. However with the RLR, high priority data have the same loss rate as low priority data, resulting in severe performance degradations. Figure 4(a) shows the PSNR as a function of RLP packet loss rate, and Figure 4 (b) shows the PSNR as a function of frame number with loss rate of 6%. The dips in RLR scheme in Figure 4 (b) are due to errors in I frames propagating across the GOP. The dips in our scheme are due to data corresponding to last frames in a GOP being dropped.

Figure 5 compares two scheduling schemes at application layer for 4% RLP packet loss rate. In Figure 5 (a), the number of dropped packets within a GOP equals to the number of retransmitted packets. As seen, this results in the data corresponding to later P frames in a GOP being dropped. This is due to our simplistic scheduling algorithm in which the data for successive frames are sent sequentially. An alternative would be to send higher priority classes of later P frame before lower priority classes of earlier P frames. In Figure 5(b), if the number of retransmitted RLP packets for GOP i is N_i, we drop $N_i/3$ packets in GOP i, and $N_i/3$ packets in each of GOP $i+1$ and $i+2$. Since the wireless channel is bursty, this distributes the loss among different GOPs, resulting in smaller PSNR variations across frames.



(a) (b)

Figure 5: *The video quality of our scheme versus RLR with active drop packets*

6. CONCLUSION

In this paper, we have proposed application and channel adaptive scheme for video transmission over wireless networks. We have demonstrated the effectiveness of the application layer adaptivity combined with the RLP layer granularity. Future work involves use of Fine Grain Scalable (FGS) video in conjunction with some of the schemes proposed in the paper.

7. REFERENCES

[1] D. Qiao and K. G. Shin, "A Two-Step Adaptive Error Recovery Scheme for Video Transmission over Wireless Networks," IEEE Infocom 2000, pp. 1698 – 1704, 2000

[2] J. Cai, et al "An FEC-Based Error Control Scheme for Wireless MPEG-4 Video Transmission," IEEE WCNC 2000, vol. 3, pp. 1243 – 1247, 2000.

[3] M. Budagavi, W.R. Heinzelman, J. Webb, R. Talluri, "Wireless MPEG-4 video communication on DSP chips",IEEE Signal Processing Magazine, vol: 17 no 1 , pp 36-53 Jan. 2000.

[4] K. Stuhlmüller, N. Färber, M. Link, and B. Girod, "Analysis of Video Transmission over Lossy Channels", IEEE Journal on Selected Areas in Communications, Vol. 18, No 6. June 2000

[5] H. Zheng, J. Boyce, "Packet coding schemes for MPEG video over Internet and wireless networks", IEEE WCNC, pp.191-195, vol.1. 3 Sept. 2000.