

Hybrid ARQ for robust video streaming over wireless LANs

Daniel Grobe Sachs*, Igor Kozintsev, and Minerva Yeung
Intel Corporation
sachs@uiuc.edu, {igor.v.kozintsev, minerva.yeung}@intel.com

Douglas L. Jones
Coordinated Sciences Laboratory
University of Illinois at Urbana-Champaign
dl-jones@uiuc.edu

Abstract

In this work, we address the problem of transmission of multimedia data over a wireless Local Area Network (LAN). The wireless LAN is modeled as a packet erasure channel. We present a solution that efficiently combines Automatic Repeat Request (ARQ) and Forward Error Correction coding for sending compressed video and audio over a packet network.

1. Introduction and motivation

As the Internet grows and the bandwidth available to users increases, the nature of the data sets encountered on the Internet is changing rapidly. Streaming media—typically audio, video, or both—forms a large and increasingly important share Internet’s application mix. However, TCP/IP-based techniques, which are designed to deliver the entire source stream reliably but with a potentially large delay, are inappropriate for rich media streams, as late data is no longer useful and need not be transmitted at all.

For this reason, several streaming media architectures, such as Microsoft’s *Windows Media* and Real Networks’ *RealPlayer*, have been introduced to transmit rich media data over Internet networks. Unfortunately, even mechanisms designed exclusively for media streaming work poorly on wireless networks. As wireless networks (including the new IEEE 802.11, HomeRF, Hiperlan and Bluetooth) become more popular, new techniques for transmitting rich media over the wireless networks must be deployed. In this paper, we will propose a technique, called Hybrid Automatic Repeat Request (called Hybrid ARQ or HARQ), which is designed from the outset to perform well on both wireless and wired data networks.

*Daniel is currently at the University of Illinois at Urbana-Champaign; this work was performed during an internship at Intel. This work is partially supported by the NSF under grant number EIA-0072043.

Considerable prior art exists for transmitting streaming multimedia data over wireless networks. One such method is the use of “protocol boosters” [1], which are proxies that can be used inside existing protocols to adapt them for various different stream types and network topologies. These “boosters” are added on an as-needed basis to improve the performance of communications systems for which basic IP protocols are not well suited. For example, for wireless networks these “boosters” can be used to take an existing media stream, encapsulate it in forward error correction, and provide deadline-aware retry requests.

Another method for transmitting multimedia streams over wireless networks is the use of forward error correction to form multiple descriptions of an arbitrary media stream [2]. This technique, called MD-FEC (Multiple description using Forward Error Correction), is well suited for fine-grained scalable (“progressive”) media streams transmitted over a wireless network that allows only packet-level error correction. However, because most existing media streams, including those encoded with standard MPEG audio and video encoders, are not easily scalable, MD-FEC cannot be used to transmit most streams without a costly transcoding process.

For interactive multimedia streams, the the “RESCU” family of protocols introduced in [3] are a viable method for recovering from network transmission errors. These of protocols, designed specifically for the tight deadlines characteristic of interactive video streams, relax the deadlines for error correction and retransmission by using late data to prevent errors from propagating into frames that have not yet been displayed. This family of protocols includes several variants, including variants using both FEC and ARQ as their underlying error-correction method.

2. Problem and proposed approach

For this paper, we assume that multimedia is to be compressed, packetized, and delivered to the receiver over an unreliable channel. In this section, we define our source and channel model, and address the problem of optimizing information delivery over the packet-erasure channel.

2.1. Source and channel models

A diagram of our proposed system is shown in Figure 1. In our system, the source is assumed to produce data at a constant rate R_s packets per second. The maximum time allowed for delivering the data from the sender to the receiver is upperbounded by T_d (seconds). This constraint targets multimedia applications where the data often becomes useless unless delivered by a specific deadline.

The channel is assumed to be a packet erasure channel as shown in Figure 1. In practice, this is implemented using checksums to detect and discard corrupted packets at the link layer. We assume that all channel packets are of the same size, are transmitted with a constant rate, and are randomly "erased" in the channel. The statistics of the erasure channel can be approximated as a Markov-type process in many practical scenarios [4]. Other models (e.g., memoryless) are possible as well. The return channel, if present, can be used to send acknowledgments and other control information back to the transmitter. In this paper we assume that the return channel is obtained by time division multiplexing with the forward link, as is often the case in practice.

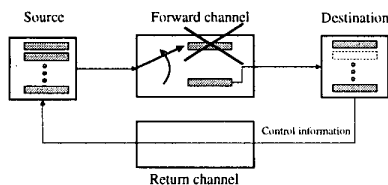


Figure 1. Streaming system diagram

The general problem is to design a coding scheme that maximizes some objective quality criterion between the source and the receiver. In the case of multimedia data, a quality measure such as end-to-end distortion is appropriate; however, such an approach requires both source and channel coding be considered in the optimization. This is referred to as a joint source-channel coding, or JSCC. To simplify matters and maintain compliance with video coding standards such as MPEG, in this paper we will simply consider the maximization of data throughput over the channel subject to a maximum delay constraint.

2.2. Coding for packet erasure channels

Two types of channel coding methods can be used to combat data losses during transmission over a packet erasure channel. The first, FEC, provides additional data that can be used to replace missing packets at the receiver. The second, ARQ, provides for automatic retransmission of missing packets.

Forward error correction is applied to a group of source data packets to produce extra parity packets that are sent along with the data packets. The FEC code is carefully designed to be able to protect the data against channel erasures under most circumstances. FEC therefore provides error resilience by increasing the amount of data to be sent. FEC does not require a return channel and is typically not adaptive to the current state of the channel. Also, FEC techniques do not guarantee that the data will arrive to the receiver without errors. FEC operation is illustrated in the Figure 2 (top diagram). For conventional FEC, a set of data packets is transformed into fixed number of coded packets. If the number of erased packets is less than the decoding threshold for the FEC code, the original data can be extracted intact. One popular class of erasure correction codes, Reed-Solomon codes, have desirable optimality properties and are the focus of this paper. Other classes of erasure correction codes exist, including a family of very fast but suboptimal Tornado codes introduced in [5].

Reed-Solomon codes are an instance of a larger class of linear block codes. They are systematic codes, which mean that the code words contain the original data in unmodified form along with added parity symbols. Reed-Solomon codes can be described in terms of two numbers, (n, k) , where n is the length of the code word, and k is the number of data symbols in that code word. (Therefore, a $(255, 205)$ Reed-Solomon code consists of 205 data words and 50 parity words.) Each symbol is drawn from a finite field of 2^s elements, where s is the number of bits to be represented in each symbol. Typically, 8 bits per symbol are used. The total number of words in the code is equal to $2^s - 1$; however, by replacing some data symbols with known values, we can realize smaller codes. For example, by replacing 105 data symbols in a $(255, 205)$ code with zeros, we create a $(150, 100)$ code.

Although Reed-Solomon codes can be used to correct errors, erasures, or both, particularly efficient decoding algorithms based on Vandermonde matrices [7] exist if only erasures are to be corrected. In this case, each parity symbol can correct one missing data symbol. This means that the original codeword (and, therefore, the original data) can be recovered if at least k of the original n symbols are received intact.

Block codes can be easily used to create packet erasure codes by simply striping the code words across packets, so that each packet contains one symbol from each of a large

number of Reed-Solomon code words. If this is done, a packet erasure will erase one symbol from each code word; this can be corrected using the parity symbols contained in one parity packet. In this way, an (n, k) packet erasure code, where all k data packets can be decoded if at least k packets arrive, can be created.

To calculate the throughput of an FEC channel, let I_e^n denote a random variable that represents the number of packet erasures in a group of n packets. Assuming an (n, k) Reed-Solomon channel code and independent packet erasures with probability P_e the throughput of the FEC coding scheme is equal to:

$$T_{FEC} = \frac{k}{n} \text{Prob}(I_e^n \leq n - k) = \frac{k}{n} \sum_{j=0}^{n-k} P_e^j (1 - P_e)^{(n-j)} \frac{n!}{j!(n-j)!},$$

where $\text{Prob}(I_e^n \leq n - k)$ is the probability of getting $n - k$ or less erasures in n packets.

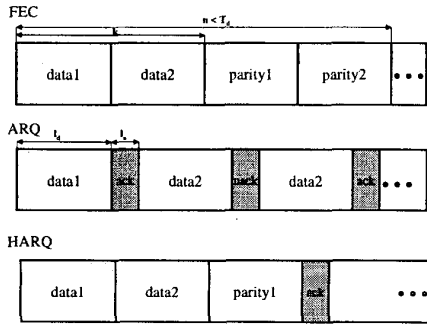


Figure 2. FEC, ARQ and Hybrid ARQ coding schemes

An alternative channel coding method is called an automatic repeat request (ARQ) illustrated in Figure 2 (diagram in the middle). For conventional “stop-and-wait” ARQ, if a transmission error is detected in the receiver (i.e., if the packet is erased), the receiver requests the packet retransmission by sending a negative acknowledgment signal on the return channel. If no error is detected the receiver sends a positive acknowledgment signal to the transmitter. To alert the transmitter of the error, ARQ requires a two-way communication channel to be present. In the case when the return channel uses the same physical medium as the forward channel, ARQ, like forward error correction, effectively expands the data bandwidth with retransmissions and communication of control information. The difference between the FEC and ARQ is that ARQ is inherently channel adaptive, as only lost packets are retransmitted, while the introduction of FEC adds overhead even if the channel

is clean. However, on poor channels ARQ may introduce significant delays due to the roundtrip propagation time of the retransmission request and its response. These delays significantly limit the applicability of ARQ to multimedia communications.

To estimate the throughput of a simple ARQ scheme described above we assume that l_d is the duration (in the channel) of a data packet, and l_a is the duration of an acknowledgment, and we assume that no erasures occurs on the return channel. We also relax the maximum delay constraint for simplicity.¹ The throughput is equal to:

$$T_{ARQ} = \frac{l_d}{l_a + l_d} (1 - P_e),$$

assuming independent packet erasures. (This simplification provides an upper bound to the actual throughput.)

2.3. Proposed Hybrid ARQ method

In this section, we propose a way to combine the two error control methods to improve the performance of multimedia communications over packet erasure channels. Our scheme is inspired by a similar solution proposed in [6] for rate-compatible punctured convolutional (RCPC) codes. The idea is illustrated in Figure 2 (bottom diagram). We start by generating FEC packets similar to the “pure” FEC coding scheme above. The Hybrid ARQ method then works as follows. Initially, the first k data packets are sent to the receiver. Then transmitter starts sending parity packets until one of the following two events occurs: either an acknowledgment from the receiver arrives, or the total number of parity packets $n - k$ is exhausted. Once at least k packets are received intact, the receiver sends an acknowledgment and the transmitter continues with the next k data packets. Notice that the algorithm does not break down even when acknowledgments are lost, as the transmitter simply assumes that more parity is needed. Note that this description provided is only intended to illustrate the proposed HARQ scheme; Actual implementations can vary. One particular implementation of the HARQ scheme for the IEEE 802.11 wireless LAN (WLAN) is described in the next section.

To compare the expected throughput of the proposed HARQ with the FEC and ARQ methods, we can compute the throughput of the hybrid system under the same assumptions. Let E denote a random variable representing the total number of packets sent in a *successful* transmission of k data packets. Assuming no erasures in the return channel, independent erasures in the forward channel, the throughput of the described HARQ system can be estimated as:

$$T_{HARQ} = \sum_{e=k}^n \frac{kl_d}{el_d + l_a} \text{Prob}(E = e),$$

¹Deriving the exact formulas requires the use of queuing theory beyond the scope of this paper.

where

$$Prob(E = e) = (1 - P_e)^k P_e^{e-k} \frac{(e-1)!}{(e-k)!(k-1)!}$$

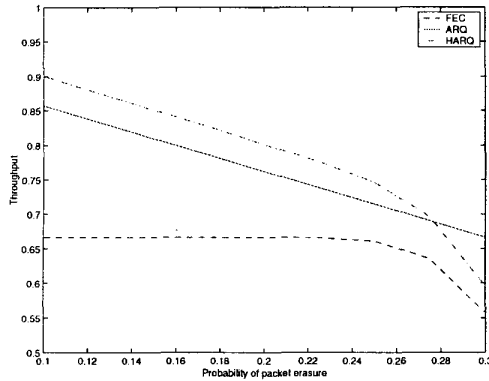


Figure 3. Throughput for FEC, ARQ and HARQ schemes for $\frac{l_d}{l_a} = 20$, $n = 150$ and $k = 100$.

The throughput for different coding schemes is shown in Figure 3 for the case of $\frac{l_d}{l_a} = 20$, $n = 150$ and $k = 100$. Those parameters are chosen to be close to the parameters of a real system described in the next sections. Notice that the throughput of the proposed HARQ method is better than the one of the ARQ system because of sending less ACK's (the crossover point is due to the ARQ bound becoming less accurate in noisier channel condition). Both ARQ and HARQ also outperform FEC in this setup, mostly due to their adaptivity to the channel.

3. Hybrid ARQ Test Bed

To evaluate the effectiveness of the Hybrid ARQ architecture, we developed a wireless test bed consisting of several 11Mbit/sec 802.11b "Wi-Fi" access points and wireless network PC cards. The streaming media server is attached to the wireless access point via a wired network. The clients are Intel-architecture PC laptops, which receive the streaming media data via the wireless network cards.

3.1. The 802.11b WLAN

The 802.11 standard along with the 'b' annex define a 2.4GHz spread-spectrum wireless LAN capable of operation at bit rates ranging from 1Mbit/sec (using a spread-spectrum BPSK modulation) up to 11Mbit/s (using a coded QPSK modulation called Complementary Code Keying or CCK.) It uses the same logical link layer as other 802-series networks (including the 802.3 wired Ethernet standard), and uses compatible 48-bit hardware Ethernet addresses to simplify routing between wired and wireless networks. Like

in wired Ethernet, corrupted packets are dropped at the link layer; therefore, the wireless link appears as a packet loss network to applications running on the network.

However, significant differences between the properties of wired and wireless networks demand a very different media access control (MAC) layer for 802.11 wireless networks. Using radio transceivers for the network physical layer is complicated by two important problems: the inability of radio transceivers to detect collisions as they transmit, and the potential for devices outside the network to interfere with network transmissions. Communication is also hampered by the hidden node problem; two widely spaced nodes on the network may be unable to communicate with each other directly, and yet still interfere with transmissions to an intermediate point.

To address these limitations, a complex MAC that includes retransmissions of corrupt packets and collision avoidance is used. This complex MAC is required for adequate TCP performance, as TCP responds to packet losses by throttling the transmission rates. Therefore, packet losses not caused by congestion must be hidden by the MAC layer to allow TCP and TCP-friendly congestion control algorithms to choose an appropriate transmission rate.

Unfortunately, the retransmissions used to render the wireless network TCP-friendly introduce significant delay and can actually delay multimedia packets beyond their deadline. Furthermore, our measurements indicate that, largely due to this complex MAC, 802.11b has a large per-packet overhead. This means that acknowledgments, which carry no source information, take a large fraction of the amount of network time taken by long data packets. However, while forward error correction can eliminate the overhead of acknowledgement packets without compromising reliability, it wastes network bandwidth by transmitting extra parity which is not required to reconstruct the stream at the receiver.

Because of the high cost of acknowledgments, a compromise scheme between pure FEC and pure ARQ systems is desirable. The hybrid ARQ scheme, outlined above, is such a scheme.

3.2. Implementation details

The physical network used for our testing was an otherwise unloaded network based on Cisco Aironet 340 series hardware. We found that the network was reliable at short range, occasionally dropping single packets. However, at longer ranges, fades would render the channel unreliable (and sometimes unusable) for seconds at a time. In our testing, we also found that the uplink from the PC card to the access point was somewhat less reliable than the downlink from the access point to the PC card.

The software and protocols were designed to transmit MPEG-encoded data from a fixed server, through the wire-

less access point and the 802.11b wireless network, and to a mobile receiver. The server and client use standard MPEG streams at arbitrary bit rates. The server hardware consisted of a dual Pentium III Xeon 500MHz desktop server attached to the 802.11b access point, and the clients were Pentium III 500MHz laptops. In our implementation, the server reads the MPEG data from a file on its hard disk drive. The packet erasure correction is based on the fast Reed-Solomon erasure correction codes described in [7]. MPEG decoding and display is done using the Intel Media Processing Library.²

The test bed divides the MPEG data into groups of 100 1024-byte packets. For each group of 100 packets, 50 parity packets are then created, in effect creating a (150, 100) Reed-Solomon code using 8-bit symbols. A sequence number and a checksum are added to each transmitted packet, to allow proper identification of each packet at the receiver and to ensure proper operation of the erasure-correction code.

Upon the receipt of each packet, the decoder checks the identity of the received packet. The receiver maintains spaces for each of the 150 possible packets in each group; if one of the packets from the current group is received, it is placed into the correct space. Once any 100 packets are received in a group, an acknowledgment is sent to the transmitter. When packets from a new group are detected, the packets from the previous group are sent to a Reed-Solomon decoder if at least 100 packets have been received; successful decoding is guaranteed.

If, due to severe fading, fewer than 100 packets are received out of the 150 sent, the Reed-Solomon decoder does not attempt to decode the packet group. Missing packets are dropped, along with all MPEG data until the next PACK header.

We found that the hybrid ARQ algorithm allowed the system to maintain the video stream during fades, even if the fade prevented data from being transmitted from the mobile to the access point. Both the pure FEC and the hybrid ARQ scheme resulted in fewer video breakups than the use of a simple, ARQ-based TCP connection; in addition, the hybrid ARQ algorithm effectively prevented the transmission of unnecessary FEC packets when channel conditions were good.

4. Discussion and Future Work

In this paper, we have presented a combination of traditional ARQ and FEC techniques which combines the reliability advantages of FEC with the bandwidth-saving properties of ARQ. Several extensions of this hybrid ARQ scheme are possible. One relevant extension is the combination of hybrid ARQ with IP multicasting. Hybrid ARQ is particularly well suited to the transmission of a stream to a small number of hosts on the same wireless network, as parity

²More information about the Media Processing Library is available at <http://www.intel.com/research/mrl/research/mp/>.

packets sent by the transmitter are applicable to any receiver that is missing a packet. The explosion of requests that can occur when various receivers erasure different packets can therefore be avoided; the transmitter can simply continue to transmitting additional parity packets until all receivers report they can reconstruct the original data.

For interactive video transmission, where the delays introduced by the Hybrid ARQ method's buffering and decoding process are significant, Hybrid ARQ could be combined with the "RESCU" [3] protocol to extend deadlines past the display of individual frames. In this application, the Hybrid ARQ protocol brings the same advantages to the "RESCU" framework as it does to conventional media frameworks.

Another possible extension of the hybrid ARQ scheme is to respond to changing line conditions by using feedback to choose an appropriate transmission rates using joint source-channel coding techniques. If acknowledgments are not received or indicate failure even after all FEC packets are exhausted, the 802.11b data rate can be reduced; conversely, if the acknowledgments indicate that the system is transmitting all packets successfully, the 802.11b rate can be increased to reduce the network utilization of the stream, or allow a higher-rate stream to be transmitted. Congestion could also be handled using joint source-channel techniques along with scalable source encodings to optimize received stream quality while maintaining TCP friendliness of the end-to-end system.

References

- [1] D.C. Feldmeier, A.J. McAuley, J.M. Smith, D. Bakin, W.S. Marcus, T. Raleigh, "Protocol Boosters", IEEE JSAC, Special Issue on Protocol Architectures for 21st Century, vol 16, no. 3, pp. 437-444, April 1998.
- [2] R. Puri and K. Ramchandran, "Multiple description coding using forward error correction codes" *Proc. of 33rd Asilomar Conference on Signals and Systems* Pacific Grove, CA, Oct. 1999.
- [3] I. Rhee, and S. Joshi, "Error Recovery using FEC and Retransmission for Interactive Video Transmission," July 1998, Technical Report TR-98-12.
- [4] H. Zheng, J. Boyce, Bell-Labs, Lucent Technologies Packet Coding Schemes for MPEG Video over Internet and Wireless Networks *IEEE Wireless Communication and Networking conference 2000*, Chicago, IL.
- [5] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, V. Stemann, "Practical Loss-Resilient Codes" 29th STOC'97.
- [6] J. Hagenauer, "Rate-compatible punctured convolutional codes and their applications," *IEEE Transactions on Communication*, vol. 36, pp. 389-400, April 1988.
- [7] L. Rizzo, "On the feasibility of software FEC", DEIT Technical Report LR-970131. Available as <http://www.iet.unipi.it/luigi/softfec.ps>