Distributed Error Recovery for Continuous Media Data in Wide-Area Multicast

Matthew T. Lucas *Bert J. Dempsey* *Alfred C. Weaver*

Department of Computer Science
University of Virginia
Charlottesville, VA 22903
{matt,bert,weaver}@Virginia.EDU 804-982-2201/2214 (phone/fax)

July 18, 1995

Abstract

This paper proposes new mechanisms to improve the quality of wide-area disseminations of time-sensitive streams, such as packet voice and video, from a single source to multiple receivers. Most current packet-switched networks provide no end-to-end quality-of-service guarantees for packet delay or loss, which can reduce the playback quality at receivers. Buffering at the multicast receivers can be used to minimize the playback degradation due to network delay variations. Addressing packet losses is more problematic. Solutions are constrained by the long latencies that may exist between the source and some of its receivers, the danger of simultaneous messages from a large number of receivers overwhelming network and endsystem resources (*multicast implosion*), and the limited bandwidth available on most wide-area links. Adapting the transmission rate of the multicast source to the available network bandwidth prevents excessive packet loss and protects the network from severe congestion due to any single source. However, rate control does not recover packet losses. To recover packets dropped by the network, the transmission of redundant information in the data stream (*forward error correction*) has been shown effective, but redundancy may consume significant network bandwidth and is inefficient when receivers have heterogeneous error rates. In this paper we develop and evaluate a novel retransmission-based error recovery strategy. Unlike previous reliable multicast protocols, our protocol relies on retransmissions from one receiver to another, thus avoiding the limitations of long latencies and multicast implosion inherent to source-driven retransmissions. Local retransmissions enables low latency recovery, uses wide-area bandwidth sparingly, and isolates the impact of losses to one area of the network. Simulation results presented in this paper demonstrate that this retransmission-based scheme can achieve distributed and effective recovery of packet losses for large receiver sets. The results show further that network resources are efficiently utilized and compare favorably with utilization under forward error correction.

University of Virginia Technical Report CS95-52

1 Introduction

The emergence of network-level multicast capabilities has enabled applications requiring large-scale distributions of time-sensitive data streams such as digital audio and video (*continuous media*) across wide-area packet-switched networks. The success of the experimental multicast network, the MBONE [7, 11], has demonstrated strong user interest in videoconferences and broadcasts over the Internet, and other real-time multicast applications such as distributed simulation [32] are anticipated. Most packet-switched networks, however, have no mechanisms for protecting time-sensitive streams in an end-to-end manner from delay variations (jitter) and packet losses that disrupt timely delivery to receivers. Research efforts to develop efficient quality-of-service mechanisms within the network are currently active [18, 35]. Even if these efforts are successful and become widely accepted, deployment of mature schemes is unlikely to occur in the near future. In this paper we pursue an alternative approach of constructing endsystem protocols to improve the quality of time-sensitive multicasts, requiring only a datagram multicast capability from the underlying network.

Variations in packet delays are one potential threat to the media playback timing at a multicast receiver. Continuous playback can be achieved, however, if packets are buffered at the beginning of the network transmission. Buffering sufficient data for jitter is easily achieved with modern workstations since limited bandwidth in wide-area networks constrains continuous media multicasts to modest data rates. Current MBONE audiovisual streams [11], for example, consume at most a few hundred Kbits/s of bandwidth.

Minimizing the effects of packet losses is more problematic. Some packet loss generally must be tolerated by continuous media applications since timely recovery of lost packets can not be guaranteed. The impact of packet loss on playback quality is variable and depends on the amount of redundancy in the stream, the robustness of the decoding scheme, and the importance of the data lost. Digitized audiovisual streams exhibit a high degree of redundancy, a fact exploited by many encoding algorithms. Highly compressed or low bit-rate encoded streams have little redundancy and will be more sensitive to losses in the network, but limited wide-area bandwidth encourages the aggressive use of compression for multicast audiovisual streams. Also, recursive encodings such as differential PCM for voice [8] and the MPEG video standards [19] may suffer substantial disruptions from a gap in the data since the predictive components at the decoder will lose synchronization. For these reasons even the loss of a single packet may be perceptible during playback.

Error recovery strategies for large-scale time-sensitive multicasts are constrained by considerations of latency, scalability, and limited wide-area bandwidth. A number of reliable multicast protocols have been designed by extending the well-known automatic repeat-request (ARQ) retransmission protocols [3, 14, 31, 33, 2]. These protocols, however, are not designed for time-sensitive data such as continuous media, and their error recovery techniques do not scale well to very large receiver sets. In a time-sensitive distribution, the roundtrip time between a receiver and the source in wide-area transmissions can be sufficiently long that timely recovery through ARQ retransmission is infeasible for that receiver. Even when roundtrip latency is not a barrier, communication between the multicast receivers and the multicast source risks *multicast implosion*, that is, a large number of receivers simultaneously transmitting retransmission requests to the source. Multicast implosion can result in severe network performance degradation and exhaust processing resources at the source. Since the danger of implosion grows with the size of the receiver set, implosion is an important issue for wide-area multicasts, where the potential number of receivers is very large. In addition to the implosion problem, protocol messages for wide-area multicast protocols must carefully control the amount of bandwidth consumed. Wide-area networks are bandwidth-constrained, especially in contrast with current and emerging local area and metropolitan area networks, and, at the same time, there is a clear trend of explosive growth in demand for wide-area bandwidth [26].

Congestion-based losses can be reduced through dynamic rate control of the multicast source. By sending to the source feedback messages indicating network performance, receivers adapt the data rate of the source to the prevailing network capacity [6, 17, 34]. The data rate is adjusted by changing the compression ratio or selectively dropping information, e.g., skipping video frames. While lower data rates reduce the quality of the audiovisual stream, the technique protects the network from being swamped by a single transmitter and provides good quality relative to the current network capacity. To provide a rate control algorithm that avoids the implosion problem for large receiver sets, feedback messages to the source must be time-multiplexed in some manner, e.g., probabilistic polling [6, 34]. Time-multiplexed responses imply a long-duration feedback loop, which yields imperfect knowledge of the network performance at all receivers and limits the rate at which the source can adapt to changing network conditions. Note that rate controlling the source is an orthogonal issue to that of recovering packets dropped by the network since rate control reduces the likelihood of extended periods of heavy packet losses, but it does not eliminate congestion-based losses.

Open-loop error control such as forward error correction (FEC) has been proposed for wide-area time-sensitive multicasts since it allows packet recovery while adding little delay to the end-to-end delivery path [4, 16, 34]. Forward error correction techniques add redundancy to the data stream in order to enable the reconstruction of lost packets from those correctly received. Dramatic improvements in the application error rate have been reported in FEC studies using physical and link layer data replication [24, 25]. The primary drawback to using FEC at the network layer is that redundant packets consume wide-area network bandwidth, even when no losses occur. Note that in large-scale multicasts the total bandwidth lost is a function of the number of links in the multicast routing tree, which can be quite large. Also, in large-scale distributions, receivers in different parts of the network experience heterogeneous error characteristics, making a single fixed level of overcoding either inadequate for some receivers or excessive for others.

In this paper we propose a retransmission scheme for large-scale time-sensitive multicast distributions. In contrast with other multicast retransmission protocols, our protocol uses receiver-to-receiver transmissions to achieve retransmission-based packet recovery. Retransmissions are performed between receivers, independent of the multicast distribution from the source. By enabling distributed recovery through local receiver-to-receiver communication, the scheme achieves low latency recovery, minimizes the impact of error recovery actions on wide-area bandwidth consumption, and provides an effective, scalable solution for large receiver sets. Our approach is supplementary to using rate control of the multicast source for preventive error control and may be combined with forward error correction in hybrid approaches.

This paper is organized as follows. Section 2 presents our scheme for distributed retransmission-based error recovery in wide-area multicasts. Section 3 is an evaluation of the cost and performance trade-offs of our approach, including comparisons with forward error correction. This investigation is carried out using a sophisticated packet-level simulation of video transmission in a wide-area IP network. Section 4 gives our conclusions and discusses future work.

2 Distributed Retransmission-Based Error Control

In this section we present a novel retransmission-based error recovery strategy for large-scale multicast distributions. Under this scheme a multicast receiver recovers lost packets by requesting a copy of the lost packets from other receivers. Retransmissions between receivers are concurrent with and asynchronous to the on-going transmission from the multicast source. Below we present the scheme and discuss three key design elements: the aggregation of protocol actions for receivers that experience similar network performance, the dissemination of addressing and network performance information among receivers, and a coordinated strategy for efficient recovery of packet losses.

2.1 Selection of Retransmission Agents

In order to control protocol overhead in a scalable fashion, our scheme requires a mechanism by which a subset of the multicast receiver group is designated to coordinate retransmission-based error recovery. These designated receivers are called *retransmission agents*. Each retransmission agent has associated with it a set of *passive receivers*. Passive receivers do not transmit protocol packets, but listen for retransmissions from the retransmission agent with which they are associated. When a retransmission agent detects packet loss, it transmits a retransmission request packet to one or more remote agents in order to recover the lost packets. If the requesting agent receives a timely retransmission of the lost packets, it sends a copy of the recovered data to a local multicast address on which its passive receivers listen. By exploiting the overlap in errors experienced by a set of local receivers, the two-layer recovery reduces the wide-area network bandwidth consumed by the retransmission protocol.

The effectiveness of distributed retransmission depends on developing a mechanism to group together multicast receivers with similar error characteristics. One approach would be to adapt the sender-driven algorithms for destination set-splitting presented in [3] for a reliable multicast protocol. Note that the receiver grouping we require maps well onto the hierarchical structure of contemporary wide-area packet-switched networks. These networks are composed of long-haul links connecting geographically remote campus networks. Each campus network is a set of local area networks that provide fast, reliable, and high-bandwidth communication. By contrast, communication over most wide-area links is relatively slow and prone to congestion-based delays and losses. Consequently, most packet losses in wide-area multicasts occur in the wide-area network environment, and the receivers on a campus will share very similar reception characteristics during a multicast distribution. Receivers may therefore be grouped into similar subgroups by assigning each multicast receiver to a local campus and allocating a multicast address for intra-campus communication between receivers.

A retransmission agent must be selected for each campus group. One approach is for receivers to discover the local group membership using a distributed searching algorithm such as that proposed in [6], followed by an election algorithm to determine the local retransmission agent [13]. Alternatively, the location of the retransmission agent may be configured manually off-line. This option has the advantage of allowing this functionality to be placed on a well-located and powerful node within the campus network. The agent need not be associated with an actual application-level user.

In this paper we explore the feasibility of effective and efficient error recovery between retransmission agents, assuming a mechanism for agent selection is in place. Investigation of agent selection techniques is thus deferred to future work.

2.2 Advertisement Protocol

Each retransmission agent periodically transmits an *advertisement* packet to notify other agents of its location and availability for servicing retransmission requests. Agents use advertisements to build a local database of potential retransmission agents to contact in the event of a loss. The database consists of the address, error independence, and estimated path delay for each agent from which advertisements are received.

Upon receiving an advertisement packet, an agent updates its local estimate of the similarity between its error characteristics and those of the sender of the advertisement. The *error indepen-*

dence value between an agent, R, and a remote agent, S, is computed as the empirical likelihood of S having packets that are lost in transit to R. That is, the error independence value between R and S, $EI(R, S)$, at receiver R is defined as the following ratio:

$$EI(R, S) = \frac{\text{number of data packets received at S but not at R}}{\text{number of data packets not received at R}} \tag{1}$$

This estimator is cumulative over all advertisements received from S by R.

An agent also maintains an estimate of the one-way network latency between itself and the sender of the advertisement. Used by the retransmission protocol to aid in determining the likelihood of a remote agent providing timely retransmissions, this estimate can be coarse, i.e., have a precision on the order of a few tens of milliseconds. Consequently, the network delay of each advertisement is measured using global timestamps at the sender and the receiver of the advertisement. These timestamps are specified to be provided by the Network Time Protocol (NTP) [21]. Implementations of NTP are readily available and widely deployed today [22], and they provide timing precision in excess of our requirements. The network delay estimate is calculated using low-pass filters similar to those for roundtrip time estimation in TCP [15].

Advertisement packets must be sent infrequently in order to conserve wide-area network bandwidth. Our protocol sets the maximum rate at which advertisements can be transmitted, based on the number of retransmission agents and the bandwidth overhead acceptable for a particular data stream. The bandwidth consumed by advertisements is targeted to be a small percentage of the bandwidth of the data stream, e.g., 1-5%, and the advertisement period to be in the range of 500 ms to 5 seconds. [1] One consequence of low-frequency advertisements is the degradation of the path delay estimates [29]. To increase the frequency of samples for the network delay estimator, we specify that an NTP timestamp be carried in retransmission request packets and retransmissions of data.

2.3 Retransmission Protocol

When a retransmission agent detects packet loss, it must contact a remote agent for a retransmission of the lost packets. Since packet reordering is rare in packet-switched networks [28], an out-of-order packet is taken as an indication of loss and triggers the recovery procedure. Using a heuristic called the *server selection algorithm*, a client agent requests a retransmission by sending a request packet to a remote agent, the *server* agent for this request. If the server has the packets, it sends them to the client immediately. Otherwise, the server agent queues the retransmission request and sends the packets at a later time if the server recovers these packets via its own outstanding retransmission

[1] If the period of transmission is every T ms, the actual transmission time is chosen uniformly over $[0.5T, 1.5T]$. This simple mechanism avoids the dangers of synchronized transmission of advertisement packets by all the retransmission agents [12].

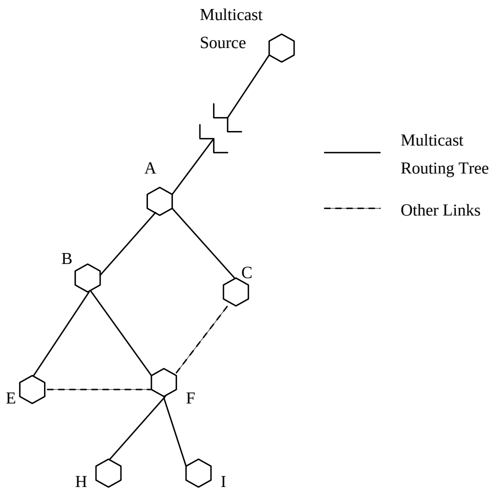

Figure 1: Example of a Wide-Area Network Topology.

requests. All retransmission protocol actions are constrained by the playback deadline of the lost packets, and outstanding retransmission requests are ignored once this deadline passes. However, the time constraints introduced by playback deadlines are generally not problematic since packet buffers at each multicast receiver can provide relatively long holding times [10]. Note that receiver buffering to handle jitter in wide-area distributions is typically in the range of a few hundred milliseconds.

The retransmission scheme is, in effect, based on the dynamic construction of a *recovery graph* in the wide-area topology. When a loss occurs, receivers contact each other until an agent with the desired packets is found. The retransmissions from that agent then propagate through the network from one agent to the next in a sequence determined by the original retransmission request messages. Since the server selection algorithm determines the remote site to which an agent sends its retransmission request, the algorithm is crucial to the effectiveness and cost of the recovery procedure. To illustrate its principles we present an example based on the wide-area network topology shown in Figure 1. In the figure each edge of the graph represents a wide-area network link and each node in the graph represents a campus network on which one retransmission agent resides. The edges shown with solid lines in the figure are part of the routing tree for multicast packets flowing from the source, as determined by the network-level multicast routing protocol. Wide-area links that are not part of the routing tree are shown with dashed lines. The example assumes that a multicast packet from the source has been dropped on transmission from campus B to campus F. When the agent at F detects that the packet is missing, the server selection algorithm proceeds as follows.

- *Eliminate agents downstream in the multicast routing tree.*
  The first observation is that packets arrive at H and I but not F only when packet loss occurs inside the campus network at F. Since loss within campus networks is rare, the agent at F should avoid sending retransmission requests to agents at campuses downstream in the multicast routing tree, e.g., H and I.

  In lieu of direct knowledge of the multicast routing tree, the agent at F has the error independence information compiled from advertisements, and low values for $EI(F, H)$ and $EI(F, I)$ will indicate that the agents at H and I are poor candidates for retransmission. Hence the first step in the retransmission server selection algorithm eliminates as a candidate any remote active receiver, R, such that $EI(F, R)$, is below a minimum threshold, which is set at 0.05 in our protocol.

- *Select a nearby agent reachable over a lightly loaded network path.*
  Recovering packets from a near neighbor reduces the total network traffic during packet recovery. Selecting a lightly loaded network path both minimizes the impact of retransmission traffic on the network as a whole and ensures that the probability of losing retransmissions is low. In light of these observations, the second step in the server selection algorithm is to choose the remote agent with the lowest network delay estimate. Here the network delay along a path is taken as an indication of the number of links in the path and the network congestion along the path.

  In the example topology, the agents most likely to have the lowest delay estimates at F are B, E, and C. Note that some of these agents, i.e., E and C, are connected to F via links that are not part of the multicast routing tree. Our retransmission protocol can use such links, where available, to route protocol traffic away from congestion.

- *Detect simple loops in the recovery graph.*
  Since the server selection algorithm biases towards a near neighbor, a clear danger is that two or more sites without the data will send retransmission requests to each other. The retransmission protocol detects two-node loops in the recovery graph by comparing in-coming retransmission requests against a list of the outstanding requests. When a two-node loop is detected, the client agent compares its unicast network-layer address with that of its retransmission server agent. If the address is numerically smaller, the client issues another retransmission request.

  In determining where to transmit this second request, the server selection algorithm considers the three candidate nodes with the lowest path delay metrics, excluding the node that resulted in a loop and any nodes eliminated on the basis of low error independence. From these three candidate nodes, the one with the highest error independence value is selected as the retransmission server for the second recovery attempt. The heuristic thus biases towards the selection of an agent relatively nearby but with a good chance of providing retransmission quickly. In the example topology, if F resolves a two-node loop involving itself and E, the most likely candidates to receive the second retransmission request are A and C, due to their positions upstream in the routing tree.

3 Evaluation of Retransmission Scheme

The performance of our error recovery protocol is investigated using a sophisticated simulation of multicast video transmission in a wide-area network environment. Section 3.1 motivates and describes the simulation model. In Section 3.2 the effectiveness and costs of retransmission-based error control is explored in a simulation environment. Further simulation experiments in Section 3.3 serve to characterize the performance of our protocol over a range of network loss rates. A comparison with a Reed-Solomon forward error correction approach is presented for overcoding levels ranging from 7.5-40%.

3.1 Simulation Environment

To evaluate the performance of our wide-area multicast retransmission scheme, a simulation model for video transmission across a wide-area IP network was developed and implemented with a commercial simulation package [30]. A simulation-based approach was chosen so that the network and application level functionality for different error recovery strategies could be readily constructed and studied over a range of network loss rates. The key components of our simulation model are given below.

- **Topology**
  The network topology for our simulation is shown in Figure 2. It is a modification of SURAnet, a regional network connecting a number of academic and commercial campuses across the southeastern part of the United States [1]. In the figure the physical links of the network are shown as thin lines for 1.5 Mbit/s links (T-1 lines) and thick lines for 3 Mbit/s links (two T-1 lines). Each campus network is modeled as a single 100 Mbits/s backbone link with directly connected 10 Mbits/s segments on which multicast receiver nodes reside.

- **Application Traffic**
  For our simulation experiments the multicast source is placed at campus CTV, and the application multicast traffic is a 500-second trace of MPEG-I encoded video recorded at a resolution of 16 frames per second with 320x240 pixels and 8 bit color [27]. The MPEG encoder

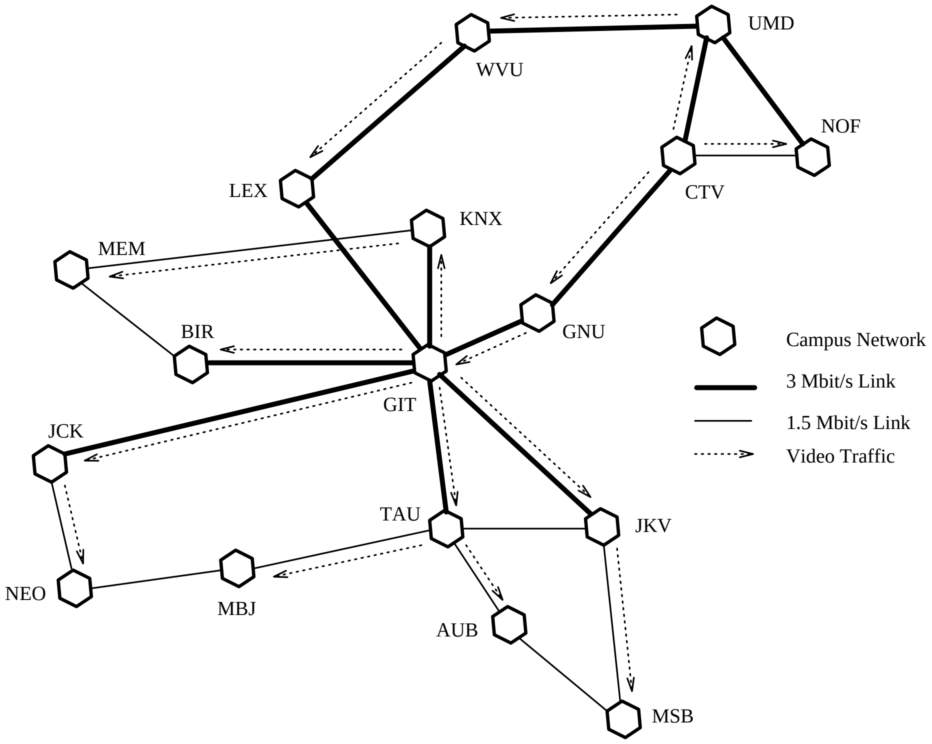

Figure 2: Wide-Area Network Topology Used in Simulation Experiments.

was set to use a frame coding sequence of *IBBPBBPBBPBB*, and the average number of bits per frame is 2.1 times the size of the maximum network packet size of 1024 bytes, though the variance of the bit-rate is high due to the MPEG encoding algorithm. The bandwidth resulting from this video stream averages 220 Kbits/s, which is approximately 15% of a T-1 link, and a peak rate of 530 Kbits/s, 35% of a T-1 link.

- **Routing**
  An underlying assumption of our retransmission scheme is that the IP network provides a multicast capability for IP datagrams. To provide multicast distribution of application traffic and advertisement packets, our simulation routes packets using source-based multicast trees constructed with a Reverse Path Forwarding algorithm [9]. The unicast routing algorithm is based on the shortest number of hops. All multicast trees and unicast routes are statically defined, and the multicast routing tree for the application video traffic is indicated in Figure 2 by the dotted arrows, assuming multicast receivers are present at all 17 campuses.

- **Background Traffic**
  The background traffic introduced by each campus into the wide-area network is modeled using Poisson arrivals with a geometrically distributed burst size. In the network topology in

Figure 2, the background load at campuses GIT, TAU, BIR, WVU, and CTV is chosen to be 2 Mbit/s, and the load at all other campuses is 1 Mbit/s.

Each burst of background packets is routed onto the wide-area links in proportion to their capacity. Each burst of background traffic is randomly routed through the wide-area network such that all packets in a burst follow an identical route. At each hop in the random route, the burst will be removed from the wide-area network with a 50% probability. The random routing of background traffic is intended to simulate the load balancing behavior found in current Internet wide-area routing protocols [20].

- **Delays**
  The end-to-end packet delays in the network are primarily composed of queuing and transmission delay at the wide-area router on each campus. Transmission delay is based on the out-going link capacity and packet size, while propagation delay is ignored. Queueing delay results from congestion due to application, background, and protocol traffic. The packet queue size in a router is 32 Kbytes for T-1 and 64 Kbytes for dual T-1 links. These values were chosen as realistic defaults for the bandwidth of the links. They imply a worst case queueing delay of 333 ms on each wide-area link. Within a campus the transmission delay over each network segment is fixed at 1 ms.

- **Losses**
  Packet losses occur in the network routers via two mechanisms. First, the output queues associated with each wide-area link has finite capacity and may overflow. Secondly, a discrete loss-load error model provides a parameterized error rate for each packet queue. Under this model, a packet entering an output queue will be the first packet in a burst loss with probability ρ if the queue is less than two-thirds full. If the queue is greater than two-thirds full, this probability increases by an order of magnitude. Each burst loss is geometrically distributed with mean μ. In this way the model captures the load-dependent probability that the processing capacity within a router will not be available to forward a set of packets.

- **Receiver Operation**
  Multicast receivers buffer one second of video data at the beginning of the network transmission. After the initial buffers are filled, video frames are removed at the generation rate of the source, i.e., at 16 frames per second. When measuring application performance, only the frames that are complete and are available in the receiver buffer at their playback deadline are designated as successfully played out. The inter-frame dependencies between MPEG-encoded frames are not considered in measuring application performance in order to prevent distortion of our study by MPEG-specific factors.

3.2 Analysis of Retransmission Protocol

In this section we select simulation parameter values that define a base configuration for our simulation environment. This configuration is used to evaluate the performance of the video applications at the multicast receivers and the network costs of our retransmission protocol.

3.2.1 Base Configuration

Multicast receivers are located on all 17 campus networks in the wide-area topology, and the receivers on each campus have a single local retransmission agent. The advertisement rate at each retransmission agent is once per second, which results in a network traffic load of about 3% of the multicast video stream.

Characterizing loss is difficult, and Internet traffic studies [5, 23, 28] have documented that losses vary greatly depending on the network path, time of day, and other factors. For the base configuration study, we assign the error intensity value $\rho = 0.006$ and set the mean number of packets in a burst loss to be $\mu = 3$. With these parameter values, the packet loss rates over individual wide-area links are between 1% and 6% in the simulation. These error rates are high enough to exercise the retransmission protocol and are within the range of measured loss rates for peak-usage communications in the Internet. We study the performance of our protocol over a range of error rates in Section 3.3.3.

3.2.2 Application Performance

Under the base configuration for the simulation parameters, the effectiveness of our retransmission protocol is shown in Figure 3. The horizontal axis in the figure identifies a campus, and the vertical axis shows the application performance at that campus, as measured by the percentage of played frames. Recall that all the packets in a frame must be available at the playback deadline in order for the application to play a frame.

The light rectangles in the figure show the percentage of played frames when no error recovery is used while the dark rectangles indicate the improvement afforded by our retransmission scheme. In this transmission scenario retransmissions enable all receivers to achieve a high playback percentage. Receivers fartherest from the multicast source experience the highest loss rates and the greatest performance gains from the use of retransmission. The improvement at AUB, for example, is from 87% frames played under no protocol to 97.5% played under the retransmission scheme.

3.2.3 Network Cost

The amount of wide-area network bandwidth consumed by protocol traffic is the primary consideration in computing the network cost of error recovery strategies. The impact of additional traffic

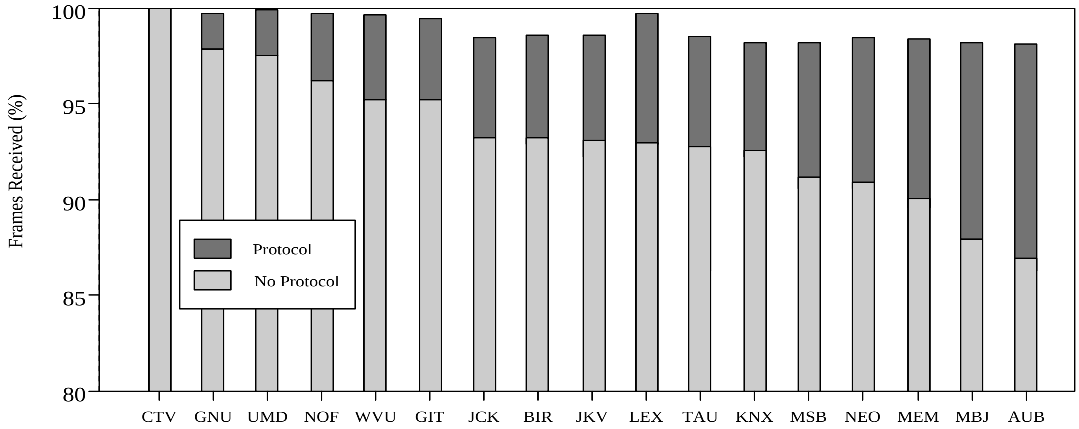

Figure 3: Retransmission Protocol Effectiveness.

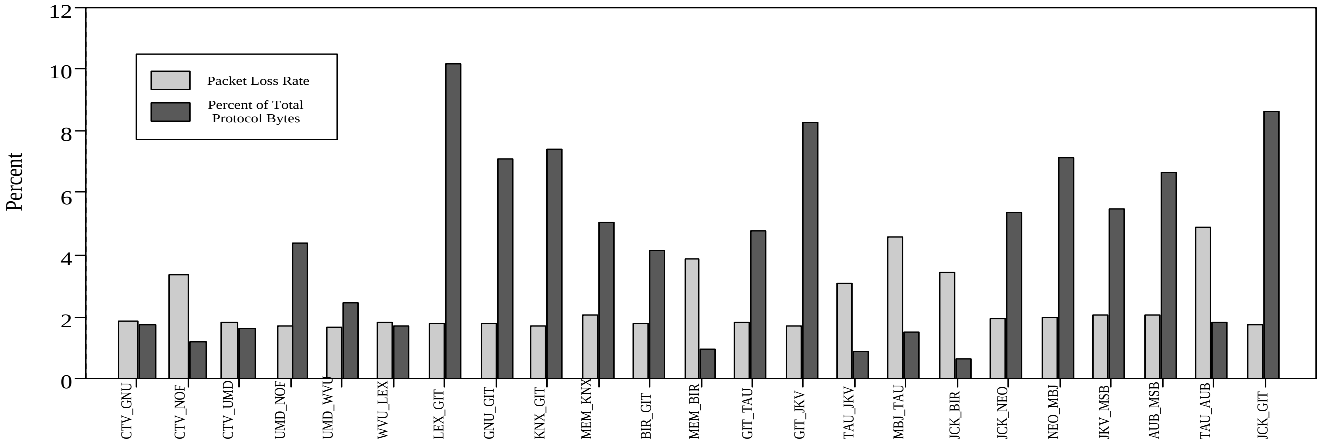

Figure 4: Retransmission Protocol Load Distribution and Loss Rates Per Link.

on a network link, however, is a function of the congestion on that link. Hence both the amount and the placement of protocol traffic in the network are of interest.

Figure 4 presents the relative distribution of protocol traffic and congestion in the network. For each link in the wide-area network, the lightly shaded histograms give the packet loss percentage, and the dark histograms give the percentage of the total traffic on a link that is due to our protocol. The protocol traffic includes advertisements, retransmission requests, and retransmitted data packets. The data graphically illustrates the tendency of the retransmission server selection algorithm to focus retransmission protocol traffic along network paths with the least congestion, which is indicated here by the packet loss rate on each link. As seen in Figure 4, network links with higher loss rates, e.g., MBJ_TAU and MEM_BIR, carry smaller percentages of the overall protocol traffic than links with lower loss rates, e.g., LEX_GIT.

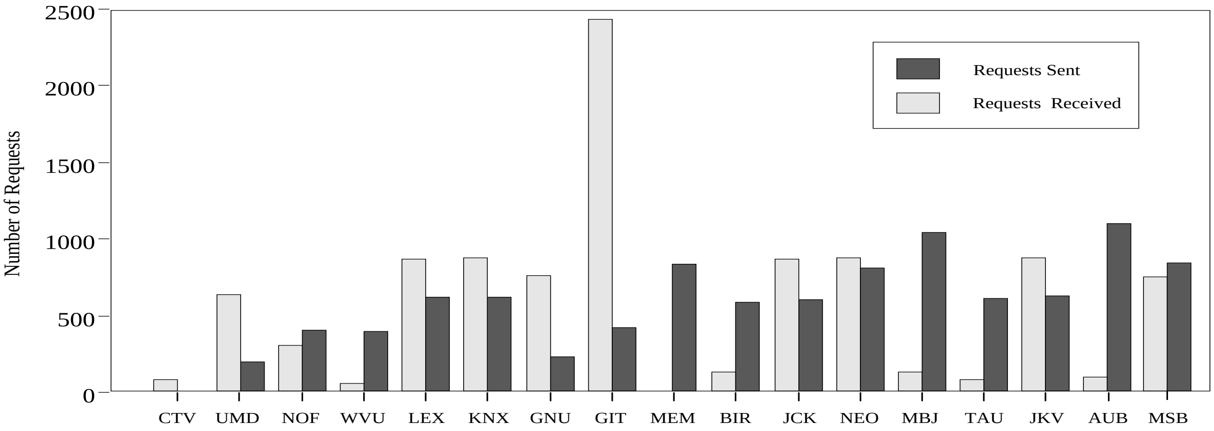

Figure 5: Distribution of Retransmission Requests.

3.2.4 Distribution of Protocol Processing

Figure 5 shows the number of retransmission requests sent (dark bars) and received (light bars) by the retransmission agent at each campus. The number of retransmission requests transmitted indicates the end-to-end error rate at each campus, and the data shows that these rates vary greatly. Due to differences in the bandwidth of wide-area links, even campuses with a similar distance from the multicast source, e.g., UMD and NOF, experience significantly different error rates.

The data for the number of requests received reveals the dependency of our algorithm on network topology. The retransmission agent at GIT receives approximately three times as many requests as any other agent, due to the fact that seven campuses are directly connected to GIT. Six of these campuses are downstream from GIT in the multicast routing tree, and thus the retransmission server selection algorithm, which biases towards nearby agents upstream in the multicast tree, results in these sites sending most of their requests at GIT. Since in general the number of links fanning out from any one site is small, the concentration of retransmission processing such as that at GIT rarely presents a problem. In cases where processing loads impose a burden, multiple retransmission agents may be needed.

3.3 Comparison with Forward Error Correction

In this section we compare the performance of distributed retransmission with forward error correction. In Section 3.3.1 and Section 3.3.2 we consider the application performance and network cost of different levels of FEC overcoding and distributed retransmission under the base configuration of the simulation. From the data in these sections, two levels of FEC overcoding are selected as good candidates for comparison with distributed retransmission, and Section 3.3.3 presents a performance evaluation of these three error recovery strategies over a range of network error intensities.

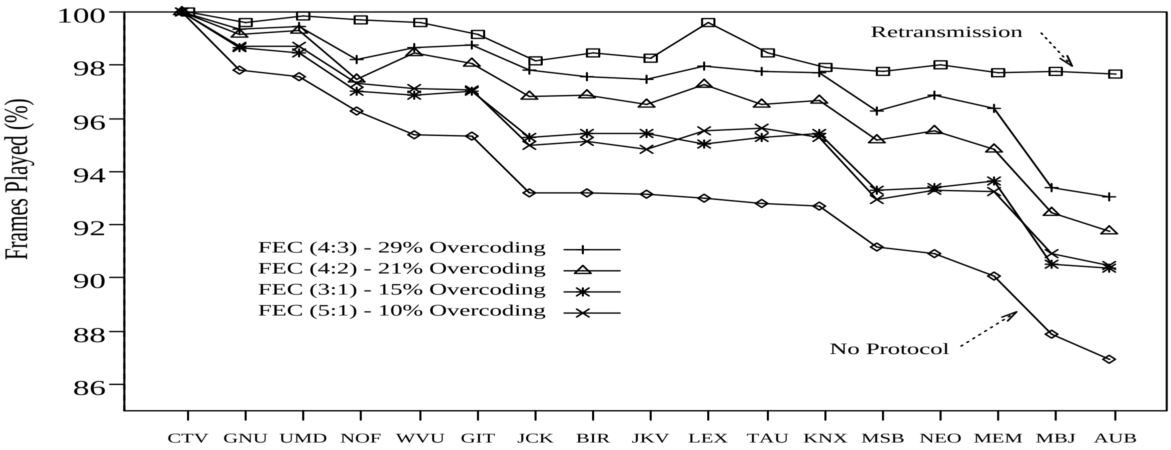

Figure 6: Performance for Different Amounts of FEC Overcoding.

The forward error correction technique used here is modeled on Reed-Solomon codes in which h redundant packets are added to n data packets [4]. If any n of the $n + h$ packets are received, then the original data can be recovered. In our experiments H redundant packets are added to each group of packets from N frames, referred to as a FEC (N:H) level of overcoding. Recall that each frame is composed of, on average, slightly more than two data packets. Thus, for a FEC (2:1) approximately one data packet is added for every four packets, e.g., approximately 20% overcoding. Using our FEC model, a frame can be played successfully if either its data packets are received correctly, or if the missing packets can be reconstructed using the FEC redundancy.

3.3.1 Application Performance

Figure 6 plots the application performance at each campus for no error recovery, distributed retransmission and FEC overcoding at 10% (5:1), 15% (3:1), 21% (4:2) and 29% (4:3). The simulation experiment for each error recovery scheme was performed under the base configuration. Over all campuses, the FEC (5:1) scheme and the FEC (3:1) scheme improves the application performance by 30% over the performance when there is no error recovery protocol. The FEC (4:2) and (4:3) schemes improve the application performance by 49% and 61%, respectively. The retransmission scheme provides the best performance, increasing application performance approximately 80%.

The data also shows the correlation between the distance from the multicast source and the effectiveness of forward error correction. Since the error rate at a receiving site increases with each additional network link in the data path, receivers farther from the multicast source recover much fewer frames, in absolute numbers, than receivers close to the multicast source. By contrast, distributed retransmission improves performance relatively uniformly across the multicast receiver set and is hence especially beneficial to receivers at far points in the network, e.g., at campus AUB.

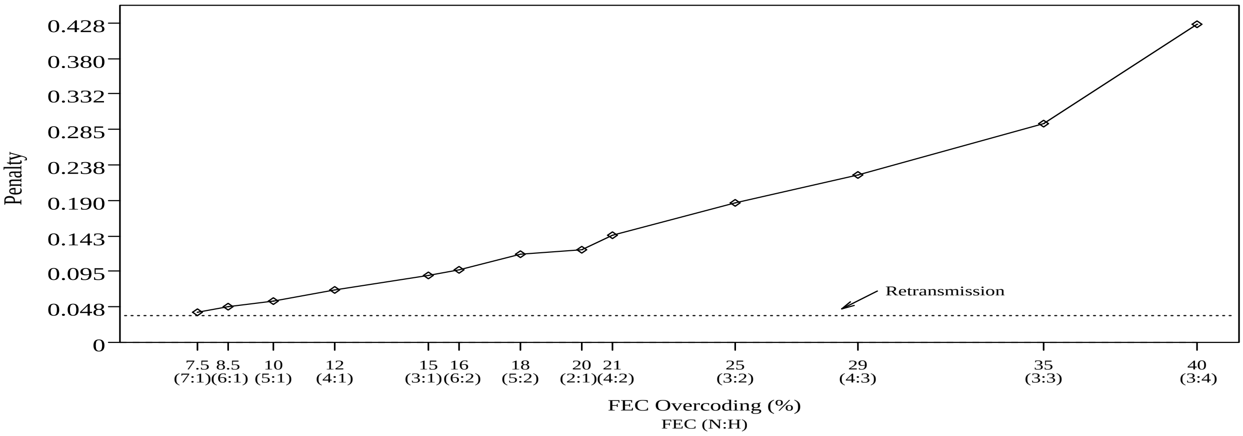

Figure 7: Penalty Metric for Different Amounts of FEC Overcoding.

At some point bandwidth constraints in the wide-area links limit the effectiveness of increasing the amount of FEC overcoding. Figure 6 shows the FEC schemes exhibit a noticeable decrease in their error recovery performance at NOF, relative to that at UMD. Both UMD and NOF are a single hop from the source at CTV, but the CTV-NOF link is a T-1 link with a 3% drop rate in the no-protocol scenario. The loss rate on CTV-NOF increases to approximately 4% under 20% FEC overcoding. This data suggests that the significant increases in the link drop rate bounds the error recovery performance of FEC. In contrast, the retransmission scheme avoids the CTV-NOF link by utilizing the NOF-UMD link for all retransmission traffic and provides excellent application performance. Note that the retransmission scheme does increase the error rate on the CTV-NOF link slightly due to advertisement packets

3.3.2 Network Cost

We now investigate the network costs associated with FEC and retransmission error recovery. In order to compare the cost of different recovery strategies, we define a penalty metric that measures the impact of error recovery overhead on the network. For each link, a penalty value is calculated according to two factors: (1) the congestion of the link, as measured by its packet loss rate, and (2) the amount of protocol traffic carried on the link, as measured by the ratio of protocol bytes to total traffic bytes on the link. The penalty for a link is found by multiplying (1) and (2), and the network-wide penalty is the sum over all link penalties. A higher penalty is therefore incurred for adding protocol overhead on congested links as opposed to uncongested links.

Figure 7 plots the network penalty metric for FEC overcoding levels ranging from 7.5-40%. The penalty metric increases linearly for the range of overcoding levels up to 35% overcoding and more rapidly thereafter. This data suggests that, for the base configuration, the error rate grows linearly for application data rates up to those created by 35% overcoding. The dotted line in Figure 7 represents the penalty metric calculated for the retransmission scheme, which equals that of 7.5% overcoding. The retransmission scheme actually adds 10% more traffic to the network than 7.5% FEC overcoding. The penalty of the two schemes are equal, however, since the retransmission protocol, whenever possible, steers its traffic towards uncongested links.

3.3.3 Sensitivity to Error Rates

In this section we compare FEC and distributed retransmission over different error intensities and burst loss sizes. For our analysis we consider FEC(5:1) 10% overcoding and FEC(4:2) 21% overcoding. The choice of FEC(5:1) was motivated by the similar penalty between retransmission in the previous section while FEC(4:2) provides a comparison for a much higher amount of overcoding.

Recall from Section 3.1 that ρ and μ characterizes the load/loss error model at each network packet switch. The intensity of burst losses is determined by ρ, and the mean number of packets in a burst loss is μ where burst size is modeled as a geometrically distributed random variable. To investigate error recovery performance, we consider isolated losses, e.g., $\mu = 1$, and bursty losses, e.g., $\mu = 3$, and for each case the value of ρ is varied from 0.0 to 0.024 to consider a range from very light to quite heavy loss rates. All other simulation parameters remain as in the base configuration.

3.3.4 Isolated Losses

Here we compare the protocol performance and network cost of FEC and distributed retransmission for the isolated loss model, e.g., $\mu = 1$. In this case, in the no-protocol scenario, the link drop rates are between 0-2.7% for $\rho = 0$, and between 2.4-4.8% for $\rho = 0.024$.

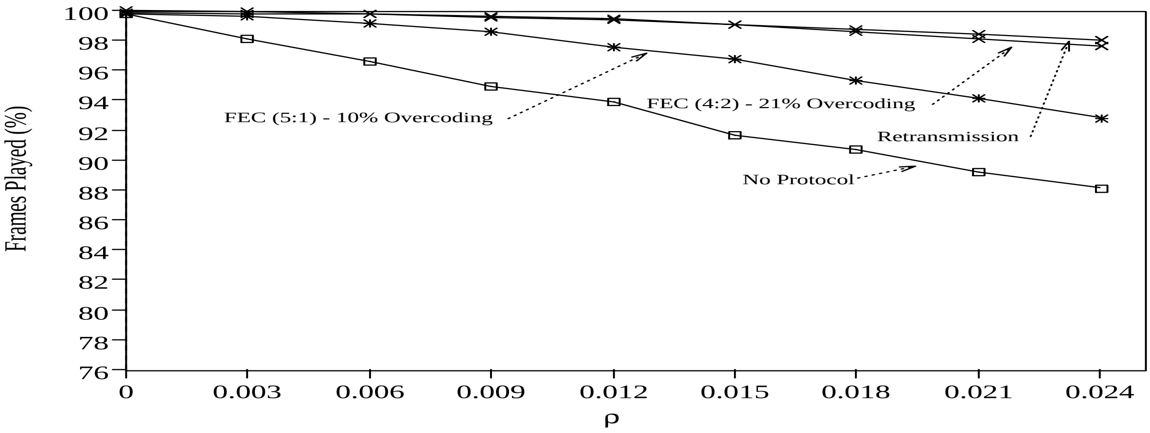

Figure 8: Performance for FEC(4:2), FEC(5:1), and Retransmission with $\mu = 1$.

Figure 8 and Figure 9 show the performance and penalty, respectively, for FEC(5:1) 10% and

FEC(4:2) 21% overcodings as well as retransmission. As seen in Figure 8, the performance of FEC(4:2) and retransmission are very comparable. Both techniques provide excellent error recovery over the full range of ρ, improving the application performance 85% over the no-protocol scenario. The FEC(5:1) 10% overcoding provides good error recovery as well, with 40% improvement. Although FEC(4:2) and retransmission provide comparable error recovery, Figure 9 shows that the penalty metric for FEC(4:2) is roughly 2.5 to 16 times greater than that for retransmission over the range of ρ shown. FEC(5:1) has a slightly higher network cost compared to retransmission under light error rates and a comparable network cost under heavy error rates.

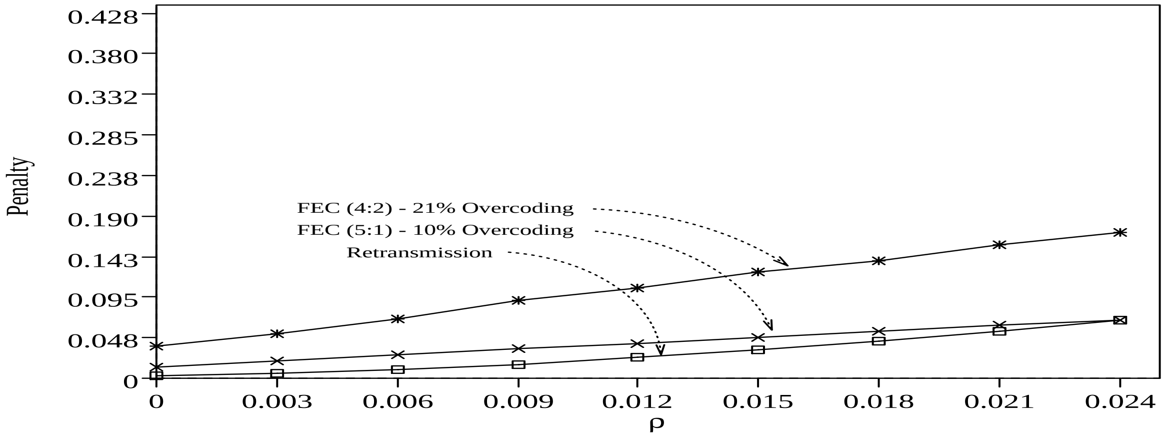

Figure 9: Penalty for FEC(4:2), FEC(5:1), and Retransmission with $\mu = 1$.

3.3.5 Bursty Losses

In this section we compare the protocol performance and network cost of the two FEC schemes and distributed retransmission for bursty losses, e.g., $\mu = 3$. In this case, without protocol traffic, the link drop rates are between 0-2.8% for $\rho = 0$ and between 6.9-10.1% for $\rho = 0.024$. Note that when $\rho = 0.006$, this represents the base configuration of the simulation.

Figure 10 shows the recovery performance for bursty losses. Here the FEC(4:2) overcoding does not perform as well as retransmission, e.g., at $\rho = 0.009$ retransmission improves application performance 80% and FEC(4:2) only 47%. Figure 11 reveals that the network costs for retransmission at low error rates are small in the bursty loss scenario. However, the network penalty increases rapidly as error rates rise, due largely to the increased amount of retransmitted data. As a result, the penalty for retransmission is similar to 10% overcoding at $\rho = 0.009$, but closer to the penalty for 21% overcoding at $\rho = 0.024$. By contrast, the FEC schemes have penalties that grow linearly over the ρ values shown.

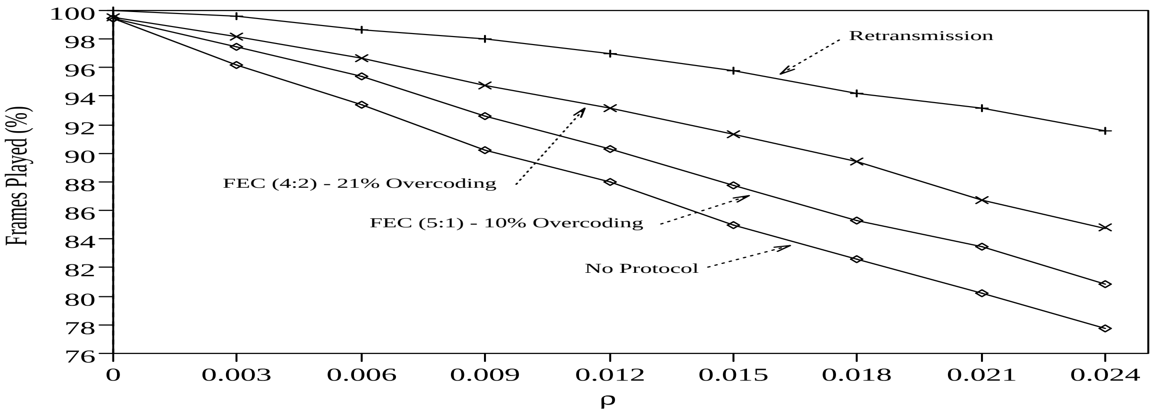

Figure 10: Performance for FEC(4:2), FEC(5:1), and Retransmission with $\mu = 3$.

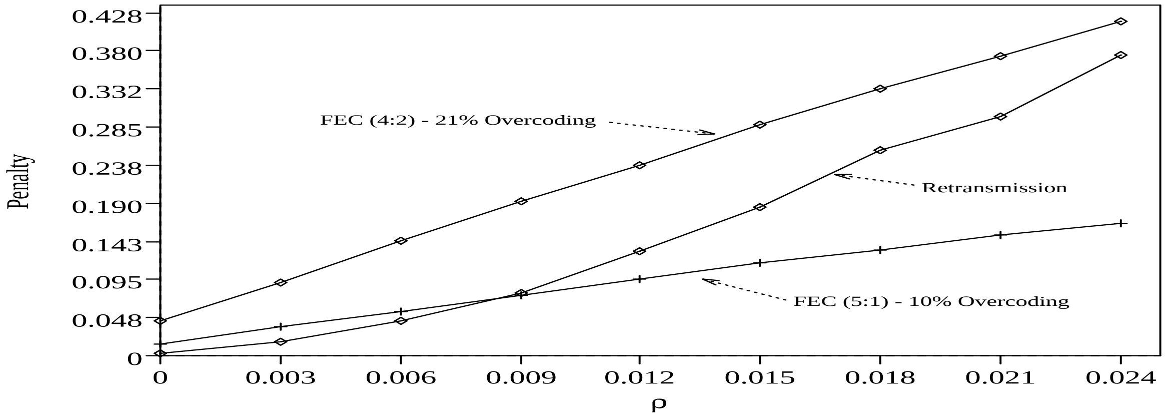

Figure 11: Penalty for FEC(4:2), FEC(5:1), and Retransmission with $\mu = 3$.

4 Conclusions and Future Work

A new error recovery protocol for multicast data distribution has been proposed in which retransmission is used to recover packet losses. Unlike traditional ARQ, this protocol distributes retransmission functionality among a set of retransmission agents, which are designated members of the multicast receiver group. In this paper we have applied this approach to time-sensitive data distributions, i.e., packet video, whereby retransmission agents attempt to recover missing data within the deadline determined by the application. This distributed error recovery approach has several advantages:

- **Recovery Performance**
  The simulation experiments in this paper suggest that the distributed retransmission scheme can improve application performance in large-scale multicasts significantly. The results here also indicate that retransmission performs well under a wide range of network error conditions.

- **Scalability**
  Protocol processing and network bandwidth overhead scales well in both the number of multicast receivers and network error rates. Our receiver-oriented approach distributes the error recovery processing among the retransmission agents throughout the network. The protocol conserves network bandwidth by having retransmission agents emit a single, local request when a loss is detected, thereby recovering the loss for an entire subgroup of (passive) receivers.

- **Network Cost**
  The retransmission scheme uses bandwidth sparingly in comparison to forward error correction techniques, as illustrated in the simulation experiments. Conservation of bandwidth is achieved through transitive recovery among agents. Moreover, impact on wide-area network performance is minimized by steering protocol traffic away from congested links in the network where possible.

- **Localized Cost**
  Distributed retransmission has the advantage of only adding protocol overhead where losses occur in the network. In wide-area multicasts, receivers may well experience very different loss rates. Under distributed retransmission, receivers in areas of the network that experience little or no losses are isolated from the error recovery overhead incurred by receivers elsewhere. This is in contrast to the fixed cost and performance level for all receivers that results from using source-based forward error correction.

The primary drawback of the proposed retransmission-based approach is the complexity costs associated with setting up and managing the distributed retransmission protocol. Dynamic division of the multicast receiver group into subgroups with the desired aggregation properties is a particularly difficult problem. Further, electing and synchronizing retransmission agents is also challenging since it requires timely and controlled dissemination of protocol control information. We believe that the advantages of the proposed distributed architecture justify these complexity costs, and this architecture integrates well with other multicast protocol functionality such as rate control, synchronization, and group management.

We are currently investigating protocols for the dynamic subgrouping of the multicast receiver set. Also, although the retransmission server selection algorithm used in this study is simple and effective, we plan to investigate more sophisticated schemes to achieve better application performance with lower network costs. Finally, our future work includes exploring the adaptation of our distributed retransmission scheme to fully reliable multicast data distributions and the integration of our scheme with receiver-based rate control of the multicast source.

Acknowledgment: The authors are grateful to Sudhir Srinivasan for his expert advice in regard to the SES simulation package.

References

[1] SURAnet Backbone Map, October 1994. ftp.sura.net:pub/maps/SURAnet/SURA.backbone.3.ps.

[2] Xpress Transport Protocol Specification, Version 4.0, 1994. http://www.ca.sandia.gov/xtp/xtp.html.

[3] M. Ammar and L. Wu. Improving the Performance of Point to Multi-Point ARQ Protocols through Destination Set Splitting. *IEEE INFOCOM '92*, pages 262–271, May 1992.

[4] E. Biersack. Performance Evaluation of Forward Error Correction in ATM Networks. *ACM SIGCOMM '92*, 22(4):248–258, August 1992.

[5] J. Bolot. End-to-End Packet Delay and Loss Behavior in the Internet. *ACM SIGCOMM '93*, 23(4):289–298, September 1993.

[6] J. Bolot, T. Turletti, and I. Wakeman. Scalable Feedback Control for Multicast Video Distribution in the Internet. *ACM SIGCOMM '94*, 24(4):58–67, September 1994.

[7] S. Casner and S. Deering. First IETF Internet Audiocast. *Computer Communication Review*, 22(3):92–97, July 1992.

[8] CCITT. Recommendation G.727 - 5-, 4-, 3-, 2-Bits/Sample Embedded ADPCM, July 1990.

[9] S. Deering. *Multicast Routing in a Datagram Internetwork.* PhD thesis, Department of Computer Science, Stanford University, 1991.

[10] B. Dempsey. *Retransmission-Based Error Control for Continuous Media Traffic in Packet-Switched Networks.* PhD thesis, Department of Computer Science, University of Virginia, May 1994.

[11] H. Eriksson. MBONE: The Multicast Backbone. *Communications of the ACM*, 37(8):54–60, August 1994.

[12] S. Floyd and V. Jacobson. The Synchronization of Periodic Routing Messages. *ACM SIGCOMM '93*, 23(4):33–45, September 1993.

[13] H. Garcia-Molina. Elections in Distributed Computing Systems. *IEEE Transactions on Computers*, C-31(1), January 1982.

[14] I. Gopal and J. Jaffe. Point-to-Multipoint Communication over Broadcast Links. *IEEE Transactions on Communications*, COM-33(9):1034–1044, September 1984.

[15] V. Jacobson. Congestion Avoidance and Control. *Computer Communication Review*, 18(4):314–329, August 1988.

[16] M. Jones, S-A. Sorenson, and S.Wilbur. Protocol Design for Large Group Multicasting: The Message Distribution Protocol. *Computer Communications*, 14(5):287–297, June 1991.

[17] H. Kanakia, P. Mishra, and A. Reibman. An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport. *ACM SIGCOMM '93*, 23(4):20–31, September 1993.

[18] J. Kurose. Open Issues and Challenges in Providing Quality of Service Guarantees in High-Speed Networks. *ACM Computer Communication Review*, 23(1):6–15, January 1993.

[19] D. LeGall. MPEG: A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 34(4), April 1991.

[20] K. Lougheed and Y. Rekhter. A Border Gateway Protocol 3 (BGP-3), October 1991. RFC-1267, (Cisco Systems and T.J. Watson Research Center, IBM).

[21] D. Mills. Network Time Protocol (Version 3): specification, implementation, and analysis. *DARPA Network Working Group*, RFC-1305, March 1992.

[22] D. Mills. Improved Algorithms for Synchronizing Computer Network Clocks. *ACM SIGCOMM '94*, 24(4):317–327, September 1994.

[23] A. Mukherjee. On the Dynamics and Significance of Low Frequency Components of Internet Load. Technical Report CIS-92-83, University of Pennsylvania, December 1992.

[24] N. Oguz and E. Ayanoglu. A Simulation Study of Two-Level Forward Error Correction for Lost Packet Recovery in B-ISDN/ATM. *Proceedings of ICC '93*, pages 1843–1846, May 1993.

[25] H. Ohta and T. Kitami. A Cell Loss Recovery Method using FEC in ATM Networks. *IEEE Journal on Selected Areas in Communications*, 9(9):1471–1483, December 1991.

[26] V. Paxton. Growth Trends in Wide-Area TCP Connections. *IEEE Network*, 8(4):8–17, July 1994.

[27] O. Rose. Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems. Technical Report 101, Institute of Computer Science Research Report Series, University of Wuerzburg, February 1995.

[28] A. Sanghi, A. Agrawala, and B. Jain. Experimental Assessment of End-to-End Behavior on the Internet. *IEEE INFOCOM '93*, pages 867–874, March 1993.

[29] D. Sanghi and *et. al.* A TCP Instrumentation and Its Use in Evaluating Roundtrip-Time Estimators. *Internetworking: Research and Experience*, 1(2):77–99, 1990.

[30] SES. SES Workbench Release 2.1, February 1992. Scientific and Engineering Software.

[31] N. Shacham and D. Towsley. Resequencing Delay and Buffer Occupancy in Selective Repeat ARQ with Multiple Receivers. *IEEE Transactions on Communications*, 39(6):928–937, June 1991.

[32] S. Srinivasan and B. de Supinski. Multicasting in DIS: A Unified Solution. *ELECSIM '95*, April-June 1995. *available at* ftp://ftp.cs.virginia.edu/pub/techreports/CS-95-17.ps.Z.

[33] D. Towsley. An Analysis of a Point-to-Multipoint Channel Using a Go-Back-N Error Control Protocol. *IEEE Transactions on Communications*, COM-33(3):282–285, March 1985.

[34] R. Yavatkar and Leelanivas Manoj. Optimistic Strategies for Large-Scale Dissemination of Multimedia Information. *ACM Multimedia 1993*, pages 1–8, August 1993.

[35] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A New Resource ReSerVation Protocol. *IEEE Network*, 7(5):8–18, September 1993.