Improving Reliable Multicast Using Active Parity Encoding Services (APES)*

Dan Rubenstein[†] Sneha Kasera[‡] Don Towsley[§] Jim Kurose[§]

[†]Dept. of Electrical Engineering
Columbia University
New York, NY
danr@ee.columbia.edu

[‡]Networking Techniques Research Dept.
Bell Labs, Lucent Technologies
Holmdel, NJ
kasera@dnrc.bell-labs.com

[§]Dept. of Computer Science
University of Massachusetts
Amherst, MA
{towsley,kurose}@cs.umass.edu

Abstract

We propose and evaluate novel reliable multicast protocols that combine active repair service (a.k.a. local recovery) and parity encoding (a.k.a. forward error correction or FEC) techniques. We show that, compared to other repair service protocols, our protocols require less buffer inside the network, maintain the low bandwidth requirements of previously proposed repair service / FEC combination protocols, and reduce the amount of FEC processing at repair servers, moving more of this processing to the end-hosts. We also examine repair service / FEC combination protocols in an environment where loss rates differ across domains within the network. We find that repair services are more effective than FEC at reducing bandwidth utilization in such environments. Furthermore, we show that adding FEC to a repair services protocol not only reduces buffer requirements at repair servers, but also reduces bandwidth utilization in domains with high loss, or in domains with large populations of receivers.

Keywords: reliable multicast, forward error correction (FEC), repair services, active services, performance analysis

1 Introduction

Providing reliable multicast delivery in a best-effort network where packet losses occur, such as the Internet, requires a *reliable multicast protocol* that compensates for these losses. A significant amount of previous work exists that explores approaches that allow these protocols to scale to groups containing large numbers of receivers without imposing a large bandwidth or processing burden upon various portions of the network or upon protocol participants. A wide range of approaches was explored, including the use of suppression timers [2, 7], the use of multiple multicast groups [9, 16, 21, 14], allowing receivers to transmit repairs [7, 27, 19, 6, 13].

*An earlier version of this work appeared in Infocom '99.

Repair services and *hybrid parity encoding / automatic repeat request* (FEC/ARQ for short) are two alternative approaches that reduce the bandwidth requirements of reliable multicast protocols. The repair services approach utilizes repair servers, which localize retransmissions to regions of the network where loss occurs [10, 12, 20, 15, 28, 26, 25]. Because each repair server might support numerous sessions in the network, and has limited buffer to store packets for retransmission, it is important to reduce the per-session buffer requirements at each repair server.

FEC/ARQ is an end-to-end approach which uses erasure coding techniques to produce special repair packets [1, 4, 3, 24]. Typically, the set of repair packets that can repair all losses incurred by receivers is smaller than the set of packets that need retransmission [17]. However, end-hosts must perform additional encoding or decoding operations.

Further reduction in bandwidth requirements can be accomplished by designing protocols that utilize both approaches [11, 18]. We refer to such protocols as *Active Parity Encoding Services* protocols, or ***APES*** protocols for short. In this paper, we introduce two novel APES protocols, which, compared to previous APES protocols (such as [11, 18]), maintain similar bandwidth requirements and reduce buffering and FEC processing within the network. The reduction in buffer is achieved by restricting buffering to a small set of encoded repairs instead of buffering original data packets. This set of FEC repair packets can be used to repair many combinations of losses incurred by receivers and because the set is smaller than the set of original data packets, it utilizes less buffer. FEC processing is also reduced at repair servers in the network by moving the majority of such processing to the end-hosts of the network (sender and receivers). Repair server-based approaches to reliable multicast provide an effective and straightforward means to reliable multicast delivery within *overlay multicast networks*, in which special application-level servers or end-systems are used to provide multicast functionality in regions where the network provides only unicast transmission support [5, 8]. APES protocols can reduce the buffering requirements of these application-level participants in overlay environments when the protocol requires data to be stored at intermediate points solely for the purpose of providing repairs.

We perform an analytical study of these novel APES protocols to determine the network bandwidth and repair server buffer requirements as a function of a repair server's likelihood of successfully servicing a repair request. These requirements are compared to those of previous approaches (both non-FEC repair server and previous APES). We find that generating all repairs at the sender

and forwarding them to repair servers allows repair servers to service repair requests with the same likelihood as in previous approaches, but with a smaller buffer. However, this forwarding causes a considerable increase in bandwidth between the sender and the repair server. We find that if repairs are generated and buffered at the repair server, then repair servers can service repair requests with the same likelihood using roughly the same amount of of network bandwidth as in previous approaches. However, the difference in buffer requirements is a function of the rate at which the sender transmits data, with the new approach yielding a larger savings as the transmission rate increases.

Finally, we determine how various loss characteristics within the network impact the bandwidth savings achieved by using an APES protocol instead of a protocol that uses repair service approaches without FEC, or only an end-to-end FEC/ARQ approach. For this study, we base our network models on studies that indicate that receiver loss rates vary across different regions of the network [29, 30]. We find that in such networks, repair services provide higher bandwidth savings than end-to-end FEC/ARQ. This is because the heterogeneous loss behavior results in a large amount of repair traffic to all receivers in the end-to-end FEC/ARQ case. However, due to the use of FEC in conjunction with repair services, our APES protocols utilize noticeably less bandwidth than non-FEC repair server protocols in high loss regions, or in regions in which a repair server services a large number of receivers. Additionally, our APES protocols also reduce buffer requirements.

The remainder of the paper proceeds as follows. The network topology and APES protocols are presented in Section 2. The protocols' performance is analyzed in Section 3 in which repair servers are used in a flat, non-hierarchical fashion. Section 4 extends our results to networks in which repair servers can be connected to one another in a hierarchical fashion.[1] Section 5 examines the impact that the networking environment has on the benefit of adding FEC/ARQ or repair services. Finally, we suggest directions for future work and conclude in Section 6. Due to lack of space, we omit discussion of some important implementation issues. These are available in [23].

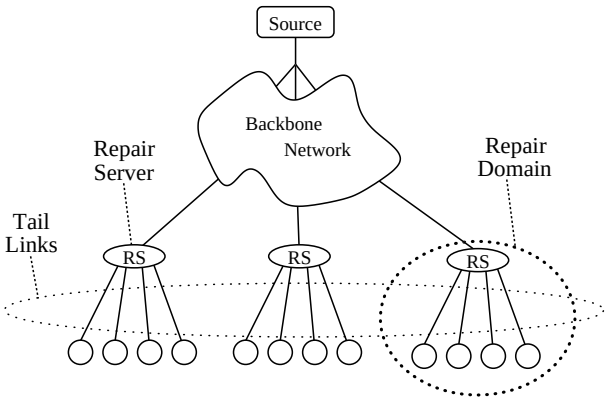


Figure 1: The network model.

2 APES

Our network model consists of a multicast tree that contains a sender at the root of the tree, and receivers at the leaves of the tree.[2] Receivers are topologically partitioned into N ***repair domains***, $D_1, \cdots, D_N$. We define the ***domain size***, $|D_i|$ to be the number of receivers in domain i. For now, we consider a one level hierarchy of repair servers, where each repair server is located at a point in the tree between the sender and a single domain. This allows the repair server to receive all packets that are multicast from the sender to receivers within its domain. See Figure 1 for a pictorial representation.

APES protocols send FEC-based repairs in place of retransmissions. The sender groups data packets into blocks of size k (henceforth k is referred to as the ***block size***), and feeds them into an encoder to produce repair packets. The number of repair packets that can be produced is sufficiently large that, for our purposes, it is safe to assume that the sender can produce an unlimited supply. Repair packets generated from a block of k data packets are said to belong to the block from which they were generated. Any entity (i.e. repair server or receiver) with a decoder can retrieve the k data packets for a block once it receives any combination of k data and repair packets that belong to that block. FEC-based protocols perform reliable delivery by ensuring that each receiver gets k distinct packets per block. The fact that receivers can lose different data packets but use the same set of repairs to recover from losses reduces the number of required repair transmissions [17].

[1]This section does not appear in the Infocom version of this paper.

[2]In practice, what we call a receiver is likely to be a local area network (LAN) that can contain several receiving participants.

To ensure that each of its downstream receivers gets at least k distinct packets per block, a repair server must receive k distinct packets per block from the sender. We refer to the first k packets received at the repair server per block (of block size k) as ***source packets***, which can consist of any combination of data and repair packets belonging to that block. The various protocols we examine here use different combinations of buffering, forwarding, and FEC coding at the repair server to ensure reliable delivery of k distinct packets to downstream receivers. We now consider what the repair server does in each protocol in order to ensure that a single block of packets is reliably transmitted to the receivers in its domain. Multiple blocks are delivered by applying the described protocols to each block of data being transferred. Due to lack of space, an exploration into the design of feedback mechanisms are not presented here, but are discussed in [23].

The Store-Data-Build-Repairs Protocol (SDBR): This protocol is similar to previous proposals [11, 18]. Once a repair server reliably obtains k source packets, it reproduces (via FEC decoding) the k original data packets, which it subsequently buffers. Whenever an additional repair is required by one or more receivers, the repair server can generate a distinct repair using its FEC encoder.

The Build-Repairs-Store-Repairs Protocol (BRSR): A repair server decides in advance on a fixed number, b, of repairs per block to generate via FEC encoding. Here, the repair server does not buffer the source packets, but merely supplies them as they arrive to the FEC encoder. We refer to this coding process as *on-the-fly encoding*.[3] For that block, only the b packets that are generated are buffered at the repair server. These b packets occupy buffer space upon arrival of the first source packet, and cannot be used for repair until the kth source packet has been fed into the encoder. If all receivers lose less than b source packets, then they can reconstruct the original data by reliably receiving a subset of repairs residing in the repair server's buffer. *Reliability is guaranteed by having the repair server reliably transmit needed repairs, instead of generating new repairs for each loss.* We shall see that this results in a reduction in buffer size and also in a negligible increase in bandwidth compared to the bandwidth used by SDBR. If any receiver loses more than b source packets, it must obtain additional repairs from the source, since the repair server is unable to generate additional repairs.

[3]Coding packages with on-the-fly encoding capabilities, as well as several other enhancements, and supporting documentation are available for download at **http://www-net.cs.umass.edu/~drubenst/software/software.html#fec.**

Table 1: High level comparison of the three proposed APES protocols.

Repair Server...	SDBR	BRSR	GRSR
Encodes?	Yes	Yes	No
Decodes?	Yes	No	No
Buffers	Data	Repairs	Repairs
Fwds repairs fr. src	Never	Sometimes	Always

The Get-Repairs-Store-Repairs Protocol (GRSR): Under this protocol, the repair server does not require FEC encoding capabilities. Instead, it requests b repair packets from the sender, which are stored upon receipt. These packets are constructed at the sender. Once the repair server obtains the b packets, the protocol behaves identically to BRSR for the remainder of the transmission of the block.

The value for b can be chosen to be between 0 and the size of the block, k, ($0 \leq b \leq k$). In Section 3, we examine how the choice of b impacts bandwidth and buffer requirements.

Table 1 presents a high-level comparison of the three proposed APES protocols. SDBR is the most efficient in terms of bandwidth. However, since it must store all within a block to perform FEC encoding, its per block buffer requirements are identical to that of a non-FEC repair server protocol, henceforth referred to as an ***RS protocol***. It also performs decoding at repair servers.[4] BRSR and GRSR buffer only repairs. Since receivers reliably receive these repairs, there is never a need to buffer more than k repairs per block. This means that these protocols need never buffer more packets per block than SDBR. They also do not require decoding functionality at repair servers. BRSR and GRSR differ in that BRSR requires additional FEC processing at repair servers to perform encoding, while the repair servers in GRSR perform no encoding. As a result, GRSR requires extra bandwidth to deliver repairs between the sender and repair server that BRSR instead generates with its encoder.

We assume, for BRSR and GRSR, that the repair server restricts buffering to the b repairs. Other policies (such as adding additional repair transmissions from the source to the buffer when receivers require more than b repairs), do not lead to considerable bandwidth savings, and can significantly increase the amount of buffer utilized. In the next section, we consider a single

[4]The decoding is not a requirement of the protocol. However, previous work assumes the capability is available. If decoding is not performed, then SDBR will require receivers to perform the recursive decoding algorithm described in [23].

hierarchical level of repair servers. We show how to extend these results to networks that contain hierarchies of repair servers in the subsequent section.

3 Buffer/Bandwidth Performance

As a consequence of our model, each repair domain can be analyzed separately. Let us assume that a block size of k is used, a repair server has r receivers downstream, and each receiver loses any packet sent to it with a probability p. To simplify presentation, we do not concern ourselves with how the repair server reliably obtains k source packets. Hence, we assume that there is no loss between the sender and the repair server. This assumption is relaxed in Section 4 when we consider sessions that utilize repair servers connected within a hierarchy. In this section, we consider four performance metrics: the expected number of packet transmissions from the repair server to its downstream receivers, the expected number of repairs received by the repair server from the sender, the expected number of packets that must be buffered per block at a repair server, and the repair server's expected buffer utilization at any moment in time, which we refer to as the ***buffer size***.

We begin by analyzing BRSR and GRSR. We index the k source packets that arrive at the repair server from 1 to k. Additional repairs are assigned distinct indices larger than k. We assume that the repair server multicasts all packet transmissions. If instead repairs are unicast, then the use of FEC provides no savings in bandwidth beyond what can be achieved by retransmitting original data packets. However, the bandwidth costs and buffer savings respectively examined in subsections 3.2 and 3.3 still hold. Our analysis here also assumes that a receiver that loses m of k packets in a block *requires* the repair server to reliably transmit packets $k+1$ through $k+m$. In practice, a receiver that requires m repairs and loses some of the repairs numbered $k+1$ through $k+m$ could effectively use any repair numbered $k+m', m' > m$ in place of a lost repair. In the latter, the number of packets that a repair server must forward is reduced. Thus, our assumption gives a conservative upper bound on the number of transmissions from a repair server.

To simplify presentation, we define $\gamma_j^k(p)$ to be the probability of losing exactly j of k packets.

$$\gamma_j^k(p) \quad = \quad \binom{k}{j} p^j (1-p)^{k-j}$$

Define ϕ_i to be the probability that a receiver requires transmission of packet i within the block. For $i \leq k$, $\phi_i = 1$. For $i > k$, this is the probability that fewer than $2k - i + 1$ of the initial k packets are received by a receiver. If exactly $2k - i + 1$ of the initial k packets are received, then it can recover the block by reliably obtaining the $i - k - 1$ repairs numbered $k + 1$ through $i - 1$. Thus,

$$\phi_i = \left\{ \begin{array}{ll} 1, & i \leq k \\ 1 - \sum_{j=0}^{i-k-1} \gamma_j^k(p), & k < i \leq 2k \\ 0, & i > 2k \end{array} \right\}$$

Define $q_i(j)$ to be the probability that at least one receiver needs more than j transmissions of packet i,

$$q_i(j) = \left\{ \begin{array}{ll} 1, & i \leq k, j = 0 \\ 0, & i \leq k, j > 0 \text{ or } i > 2k \\ 1 - (1 - \phi_i p^j)^r, & k < i \leq 2k, j \geq 0 \end{array} \right\}$$

3.1 Bandwidth: Repair Server to Receivers

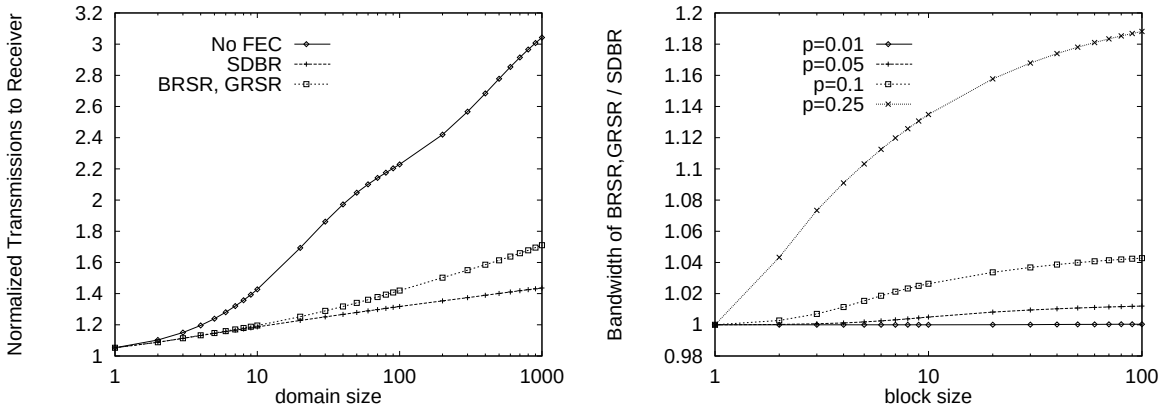
We now consider the expected bandwidth required between the repair server and its receivers (data plus repairs). Let T_{SDBR}, T_{BRSR}, and T_{GRSR} be random variables that denote the number of packets that are transmitted by the repair server using SDBR, BRSR, and GRSR, respectively. The analysis presented in [17] that gives the bandwidth requirements between a sender and a set of receivers over a star topology lends itself directly to the bandwidth computation for SDBR in our network model:

$$E[T_{SDBR}] = \sum_{j=0}^{\infty} 1 - \left[1 - \sum_{m=0}^{k-1} \gamma_m^{k+j}(1-p) \right]^r \quad (1)$$

We now compute upper bounds on $E[T_{BRSR}]$ and $E[T_{GRSR}]$. Let τ_i be a random variable that equals the number of times that packet i is transmitted. For $i \leq k$, we have $E[\tau_i] = 1$, since the

packet is always transmitted at most once. For $k < i \leq 2k$, a packet is transmitted as many times as it is needed by some receiver. Hence, (recalling that our computations yield conservative upper bounds) we have

$$\begin{aligned} E[\tau_i] &\leq \sum_{j=0}^{\infty} q_i(j). \\ E[T_{BRSR}] = E[T_{GRSR}] &\leq k + \sum_{i=k+1}^{2k} E[\tau_i]. \end{aligned} \tag{2}$$



(a) Varying domain size

(b) Varying block size

Figure 2: A bandwidth comparison of protocols.

Figure 2(a) presents the expected number of transmissions (normalized per packet) for a block size of 10 and receivers with loss rates of 5% ($p = .05$), as a function of the domain size. We observe a clear reduction in bandwidth due to the introduction of FEC, and further observe that BRSR and GRSR use essentially the same bandwidth as SDBR until the domain size grows very large. We emphasize that the bandwidth from the repair server to receivers is unaffected by the choice of b, the number of repairs that a repair server buffers per block.

Figure 2(b) gives an upper bound to the ratio of the expected number of per packet transmissions for BRSR and GRSR over the expected number of per packet transmissions for SDBR as a function of block size. Here, there are 8 receivers in the repair domain. Curves are presented for various loss rates. We observe that, for loss rates at or below 10%, SDBR is only 4% more bandwidth efficient than BRSR and GRSR. The difference is significant only for high loss rates.

Thus, for reasonable loss rates, BRSR and GRSR do not use substantially more bandwidth than SDBR between the repair server and receivers.

Note that because SDBR always sends the minimal number of distinct repairs to provide reliability, it provides a lower bound on the expected bandwidth for BRSR and GRSR. The small difference in bandwidth required by SDBR when compared to the upper bounds of BRSR and GRSR indicates that the upper bound is tight. To summarize, *SDBR requires less bandwidth between the repair server and receivers than BRSR and GRSR. However, for domain sizes and loss rates that one might expect in practice, the difference in bandwidth is negligible.*

3.2 Bandwidth: Sender to Repair Server

Recall that for BRSR and GRSR, if receivers lose more than b source packets, then additional packets must be obtained from the source by the repair server. Let A be a random variable equal to the number of additional transmissions the sender must make to a repair server. The approach used to derive (2) yields the following:

$$E[A] \;=\; \sum_{i=k+b+1}^{2k} E[\tau_i].$$

The expected number of repairs that the sender must reliably send to a particular repair server under BRSR is bounded from above by $E[A]$. For GRSR, where the b buffered repairs are transmitted from the sender as well, the expected number of repairs is bounded from above by $b + E[A]$.

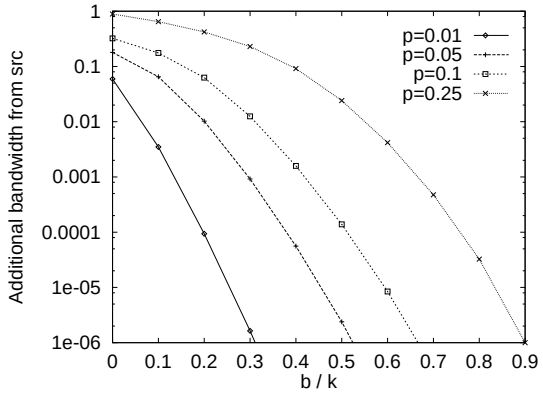


Figure 3: Additional packets from insufficient buffering, normalized to block size.

We now focus on how the choice of b affects per-block buffer and sender-to-repair server bandwidth requirements. The graph in Figure 3 illustrates the expected additional bandwidth normalized per packet (i.e., $E[A]/k$) as a function of the normalized initial buffer size (i.e., b/k) for various loss rates. For this figure, the block size is 10 and the domain size is 8. This illustrates that the number of times that the repair server must retrieve a repair from the sender is extremely small when b is chosen sufficiently large. From this, we conclude that for sufficiently large b, the additional bandwidth needed from the sender is negligible. We see a similar trend as we vary the block size or the domain size (see [23]).

3.3 Expected Buffer Size

We refer to the size of the buffer being utilized by the protocol at any given time as the ***buffer size***. The expected buffer size is an increasing function of both the number of packets that must be buffered and the amount of time that each packet is buffered. We define B_{RS} to be the buffer size of an RS protocol, and B_{BRSR} to be the buffer size of BRSR. We now compute $E[B_{RS}]$ and $E[B_{BRSR}]$. Later, we discuss how these values compare with the buffer size of GRSR, $E[B_{GRSR}]$.

Our analysis of the BRSR and RS protocols considers a repair server to which data packets arrive with arrival rate, Λ. We also assume that there is a fixed round trip time (RTT), R, between each receiver and its upstream repair server, which includes the time that it takes for a receiver to detect a packet loss, send feedback to the sender, and have the sender respond with retransmissions. When a repair server cannot provide sufficient repairs for a block (because its buffer does not contain sufficient information to produce repairs for that block), we say that a ***repair server miss*** has occurred for that block. For the RS case, a repair server miss is defined per packet (i.e., a block size of 1). We measure the repair server's effectiveness in terms of the ***repair server miss probability***, $\mathcal{P}_k$, which is defined to be the probability that at least one repair server miss occurs for a block of size k. Given k, we can compare expected buffer sizes by choosing some ϵ_k and determining the minimum expected buffer size where $\mathcal{P}_k < \epsilon_k$ holds.

A difficulty in comparing buffer sizes for differing block sizes is that repair server misses are defined in terms of the block size. A similar difficulty arises when comparing buffer sizes for an APES approach with an RS approach. However, we can take advantage of the fact that we

assume losses in the network are described by a Bernoulli process so that, for the RS protocol, $\mathcal{P}_1 < \epsilon_1 \Leftrightarrow \mathcal{P}_k < \epsilon_k$ whenever $\epsilon_k = 1 - (1 - \epsilon_1)^k$. We perform a fair comparison over different block sizes, k_1 and k_2 by selecting a value, $\epsilon_1 < 1$. From this, we compute ϵ_{k_1} and ϵ_{k_2} using the above equivalence relation.

The buffer requirements that would allow a repair server to have a miss probability lower than an application chosen bound, ϵ_1, were computed in [10] for an RS protocol. The repair server maintains each original data packet in its buffer for a fixed number of transmissions, N, which allows each receiver at least N attempts at receiving the packet. The value of N is chosen so that $\mathcal{P}_1 = 1 - (1 - p^N)^r < \epsilon_1$, where p is the loss probability from the repair server to each receiver, and r is the domain size. Using Little's Law, the expected buffer size can be computed:

$$E[B_{RS}] \quad = \quad \Lambda R(N-1).$$

The term, ΛR indicates the number of packets that a sender is expected to send between receiver retransmission requests. We refer to this value as the ***retransmission factor***.

For protocol BRSR, we compute an upper bound on the time that each packet is maintained in the buffer, such that the repair server miss probability, $\mathcal{P}_k$, is less than an application chosen bound, ϵ_k, for a block size of k. A repair server miss occurs when either b repairs are insufficient for some receiver, or when one of the b repairs is released from the buffer before it is received by some receiver that required it. Define N_i to be the number of times that a receiver can request transmission of packet i (waiting one RTT between transmission requests) before the repair server drops packet i from its buffer. We have $N_i = 0$ for $i \leq k, i > k + b$. We now compute values of N_i for $k < i \leq k + b$. Define $\psi(j)$ to be the probability that a receiver loses j source packets and makes a request for a repair that is not in the buffer. This will occur if $j > b$, or if both $j \leq b$ and some packet $k + m$ is not received after N_{k+m} transmissions, $1 \leq m \leq j$,

$$\psi(j) \quad = \quad \left\{ \begin{array}{ll} \gamma_j^k(p) \left[1 - \prod_{m=1}^{j}(1 - p^{N_{k+m}})\right] & 1 \leq j \leq b \\ \gamma_j^k(p) & j > b \end{array} \right\}$$

Define ψ to be the probability that a receiver requires a packet that causes the repair server to request an additional packet from the sender.

$$\psi \;=\; \sum_{j=1}^{k} \psi(j)$$

The repair server miss probability, $\mathcal{P}_k$, equals the probability that a repair server needs to request an additional repair from the sender. This equals the probability that at least one downstream receiver requires such a packet.

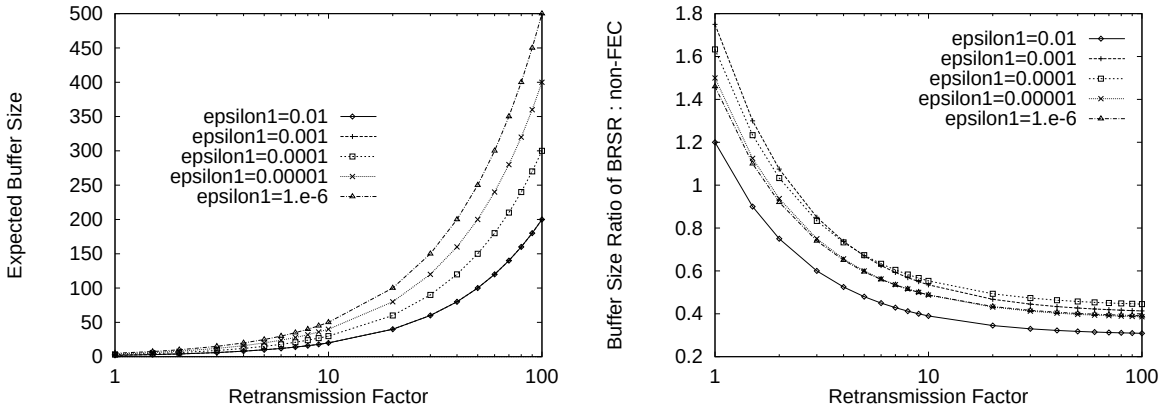
$$\mathcal{P}_k \;=\; 1 - (1-\psi)^r$$

Thus, it is sufficient for a repair server to choose b and the set of $\{N_i\}$ sufficiently large so that $\mathcal{P}_k < \epsilon_k$.

We now determine the expected buffer size. Each repair packet i, $k < i \leq k+b$, begins to occupy buffer space once the first source packet for the block arrives at the repair server. To keep our computations conservative (such that they provide an upper bound on the expected buffer size), we assume that no feedback is sent from receivers until the kth source packet has been transmitted by the repair server, and has been given ample time (e.g., half a RTT) to be received.[5] Afterwards, the ith repair packet is held for a period of time that allows receivers N_i attempts at retrieving it. The rate at which the ith packet arrives (or is constructed) at the repair server is Λ/k. Buffer space is used by the ith packet during the period of time it is being constructed (i.e., the $k-1$ time steps that occur between the arrival of the first and kth source packet), plus enough time to allow N_i retransmissions to receivers. The expected amount of time that packet i resides in the buffer is $(k-1)/\Lambda + RN_i$. It follows from Little's Law that the expected amount of buffer being used to hold packet i over all blocks is $(k-1)/k + \Lambda R N_i/k$. The total expected buffer size is obtained by summing over the b values of i for which packets can reside in the buffer:

$$E[B_{BRSR}] \;=\; \frac{b(k-1)}{k} + \frac{\Lambda R}{k} \sum_{i=k+1}^{k+b} N_i.$$

[5]Note that receivers could potentially send feedback sooner. For example, a loss of the first source packet could be detected long before the kth source packet arrives.



(a) Expected buffer size, using an RS protocol

(b) Ratio of expected buffers of BRSR to RS.

Figure 4: Expected Buffer Size comparison

Figure 4 demonstrates how the retransmission factor (ΛR) affects the expected buffer size for various values of ϵ_1. The x-axis indicates the retransmission number, and the various curves represent the values of ϵ_1. In this figure, the loss rate is .05, and the domain size is 8. Figure 4(a) gives the expected amount of buffer for an RS protocol. We see that for a small retransmission factor, very little buffer is required. The buffer size increases linearly with the retransmission factor.

Figure 4(b) gives the ratio of the expected buffer of BRSR with a block size of 10 to that of the RS protocol. We see that for a small retransmission factor, the expected buffer of the RS protocol is actually less than that of BRSR. This is due to the fact that as the data rate slows, the RS protocol can make more retransmissions of a given packet before BRSR (and similarly, SDBR) receives k source packets and is able to complete its building of repairs. Thus, very low rate data transfers via SDBR or BRSR make inefficient use of the buffer. We see in Figure 4(b) that as the retransmission factor increases, BRSR's buffer increases at a slower rate, and buffer size is considerably smaller for high rate data flows.

Figure 5 gives the expected buffer size for BRSR as a function of the retransmission factor for a variety of block sizes. A small block size results in a smaller buffer size for a small retransmission factor, because large block sizes must store repairs that are in the process of being built for longer periods of time. Their increase in efficiency at repairing loss makes them more effective as the retransmission factor increases.

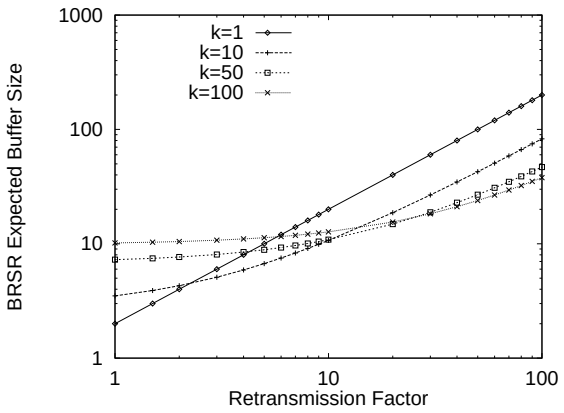


Figure 5: Expected BRSR buffer for various block sizes.

We have seen that the retransmission factor is a critical factor in determining how effective BRSR can be at reducing buffer requirements. Several additional factors lead us to believe that our computations produce an upper bound on the buffer size used by BRSR. For instance, if packet transmissions from the sender are bursty, or if receivers transmit feedback in bursts (e.g., the bit-vector approach in the RMTP protocol [20]), buffer size at the repair server for RS protocols will increase to retain the packets at the start of the burst. However, bursts will have no impact on buffer size in BRSR so long as each burst is subsumed within a block.

Finally, we discuss the buffer requirements of GRSR. We note that the buffer requirements differ in that repair packets are not cached in advance of receiving the kth source packet. We conservatively assume that all repairs arrive immediately after the last source packet for the block.[6] Little's law gives the expected buffer size to be:

$$E[B_{GRSR}] \;=\; \frac{\Lambda R}{k} \sum_{i=k+1}^{k+b} N_i.$$

The ratio of expected buffer size of GRSR to an RS protocol remains constant as the retransmission factor varies. Since the values of N and $\{N_i\}$ are chosen independent of the retransmission factor, it turns out that

$$E[B_{GRSR}]/E[B_{RS}] \;=\; \lim_{R\Lambda\to\infty} E[B_{BRSR}]/E[B_{RS}].$$

[6]If packets arrive out of order at the repair server (i.e., repairs arrive before source packets), the repair server can send the repairs as source packets and use the late source packets as repairs.

The ratio of expected buffer size of GRSR to an RS protocol can be viewed as the asymptotic values of the curves in Figure 4(b) as the retransmission factor increases.

In summary, we have analyzed the buffer and bandwidth requirements for various APES protocols. For domain sizes and loss rates that one would expect in practice, bandwidth requirements of BRSR are similar to those of SDBR, and BRSR also uses considerably less buffer than SDBR. The bandwidth requirements of GRSR on links between the repair server and receivers are identical to those of BRSR. However, because encoding is not performed at the repair server, additional bandwidth is used between the sender and repair servers to transmit all repairs which are to be buffered. GRSR uses a fraction of the buffer that an RS repair services protocol uses to meet a fixed miss probability. BRSR uses more buffer to meet a fixed miss probability for slow data rates. However, as the data rate increases, the buffer requirements of BRSR approach those of GRSR, and are considerably less than the buffer requirements of RS repair server protocols and of SDBR.

4 Hierarchies of repair servers

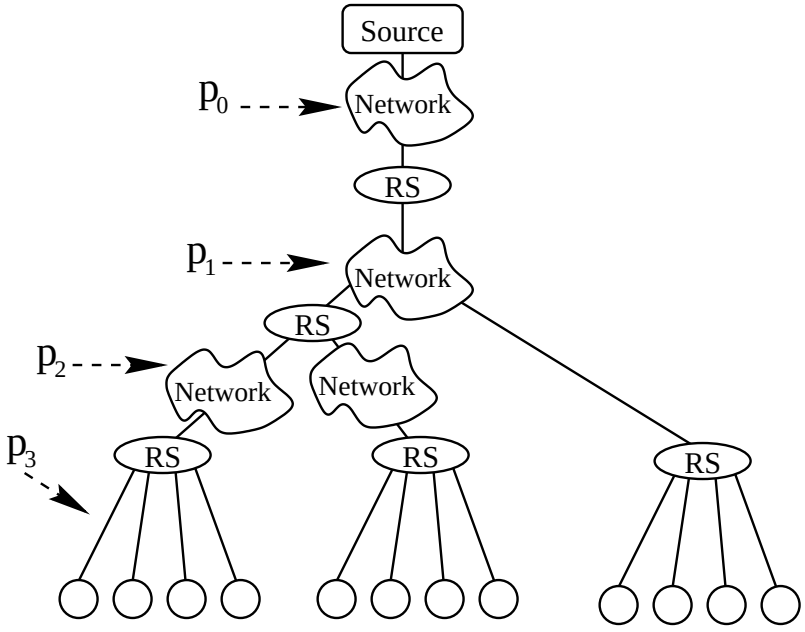


Figure 6: A hierarchy of repair servers.

We now consider a network scenario where each receiver is serviced by a hierarchy of repair

servers. We say that a repair server or receiver has a height of i if there are $i-1$ repair servers on its upstream path to the source, with the source having height 0. We denote the probability of loss between the entity at height i and the entity at height $i+1$ as p_i. A sample hierarchical network is shown in Figure 6. The repair domain on the right is serviced by two repair servers, so receivers in this domain have a height of three, while the two domains on the left are serviced by three repair servers, so receivers in these domains heights of four. Loss rates are indicated for the leftmost domain.

We note that within this hierarchical model, protocol SDBR operates at level i in the hierarchy by first retrieving the k data packets (acting as a receiver) and then forwarding these k data packets to the next level (acting as a sender). The bandwidth and buffer requirements at each level in the hierarchy can therefore be computed separately. For the GRSR protocol, the number of repair packets that must be delivered to the repair server at height i is $\max_{j \leq i} b_i$, where b_i is the number of additional repairs requested by the repair server at height i. As in the case of the the SDBR protocol, repair servers in the middle of the hierarchy behave as receivers when receiving packets from the upstream repair server and as senders when delivering packets to the downstream repair server. Bandwidth and buffer requirements can therefore be computed separately for each level of the hierarchy in the GRSR protocol.

In the case of the BRSR protocol, if each repair server acts as a receiver, then it must decode repair packets and hence store the k decoded repairs. To keep buffer requirements low at the repair server, we require that a repair server does not perform decoding upon received repair packets, but instead applies on-the-fly encoding directly to the packets it receives from its parent server. We note that these incoming repair packets may themselves be generated from repair packets that were generated at a server at a lesser height within the hierarchy. Hence, repairs are being recursively generated from combinations of repair and data packets. The decoder must therefore be modified to perform *recursive decodes*: by feeding in the set of repair packets that were generated and transmitted by the repair server at level i, the receiver retrieves the repair packets that were sent by the repair server at level $i-1$. This process is repeated until the receiver receives the set of repair packets sent by the sender, and one more additional run through the decoder retrieves the set of data packets. This recursive decoding can be implemented by repeatedly running an unmodified decoder, or by modifying the decoder so that it performs decoding over several iterations. We note

that this extension to eh BRSR protocol impacts the amount of decoding that must be done at the receiver. Under such a coding/decoding process, the bandwidth and buffer requirements can again be computed separately at each level in the hierarchy. In addition, the cost of encoding at each level in the hierarchy is similar to the cost of encoding at the sender in the non-hierarchical model.

For the remainder of this section, we focus on the decoding cost at the receiver since it is the only computation that does not follow directly from the results in the previous section. It has been observed in [22, 4] that the decoding time of a decoder is roughly proportional to kd, where d is the number of packets that must be retrieved via decoding, and k is the block size used to encode the packets. We assume that the same block size is used in the encoding at each level of the hierarchy, and thus each repair server requires exactly k packets per block before it completes encoding. Let d_i be a random variable equal to the number of packets that were lost by a repair server at height $i+1$ from the transmission of k packets by the upstream repair server at height i. This repair server must obtain d_i repairs from its upstream repair server.

A receiver at height h must perform h recursive decodes to obtain the original data packets sent by the source. The ith decoding iteration (for the initial iteration, $i = 1$) begins with a set of k packets equivalent to those that the entity at height $h - i + 1$, takes an amount of time proportional to kd_{h-i}, and produces the k packets that were initially transmitted by the entity at height $h - i$. After the hth decoding iteration, the original source packets are retrieved. The expression for the total decoding time, $\mathcal{D}$, is

$$\mathcal{D} = k \sum_{i=0}^{h-1} d_i. \tag{3}$$

Since the loss rate between the entity at height i and height $i + 1$ is p_i, it follows that $E[d_i] = c_d k p_i$, where c_d is a constant associated with the decoder. Thus,

$$E[\mathcal{D}] = c_d k^2 \sum_{i=0}^{h-1} p_i. \tag{4}$$

Protocol SDBR would only require a decoding time equal to $c_d k^2 p_{h-1}$, but each repair server would have to decode incoming repairs, and would not be able to do so until receiving all k packets for a block. If repair servers are not used, but FEC is sent directly from the source and the links

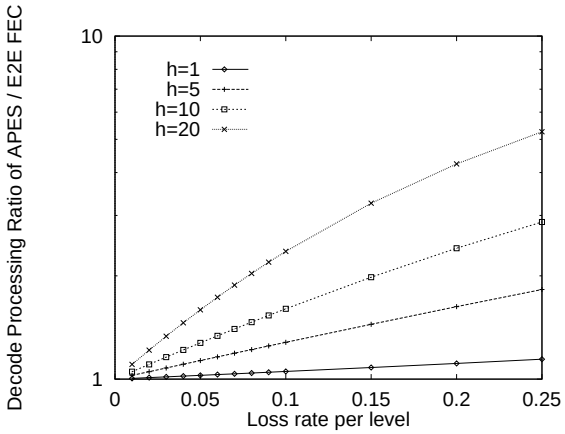


Figure 7: The ratio of decode processing between a receiver in a network that uses Protocol BRSR with that required by GRSR or of an end-to-end FEC protocol. The increase in processing is due to recursive decodes.

maintain identical loss rates, then the probability of losing a packet is $1 - \prod_{i=0}^{h-1}(1-p_i)$, and so the expected amount of decoding processing is $k^2(1 - \prod_{i=0}^{h-1}(1-p_i))$. The ratio of expected decode processing at a receiver between Protocol BRSR and Protocol GRSR is given in Figure 7. Note that since all repairs are generated at the source for the GRSR protocol, the receiver's decoding processing is equal to what is required for an end-to-end, FEC based protocol.

The various plots are for receivers at differing heights, the x-axis represents the loss rate at each level of the hierarchy, (i.e. $p_i = p_j = p$ for all i, j). The end-to-end loss rate between source and receiver is $1 - (1-p)^h$. We see that for values of p up to .1 and hierarchies containing heights as high as 10, the decoding cost in the Protocols BRSR is less than double that of the end-to-end FEC protocol.

When the loss rates on all links are identical, then the decoding ratio between Protocols BRSR and SDBR equals $h-1$, where h is the height of the receivers within the domain being considered. This ratio is constant regardless of the loss rate.

5 APES-favorable environments

We now consider the reduction in network bandwidth that results from using APES protocols. We also reveal a network setting which frequently occurs in real networks that we believe causes the largest variation in the amount of savings in bandwidth that is obtained by using an APES approach. This setting was not considered in [18].

We extend our network model to consider receivers with different loss rates. Define $\mathcal{R}_{i,j}$ to be the jth receiver in domain i, $1 \leq j \leq |D_i|, 1 \leq i \leq N$. We define $P_{i,j}$ to be the loss probability observed by receiver $\mathcal{R}_{i,j}$. Without loss of generality, we assume that receivers are enumerated within each domain, i, so that for any $1 \leq j < j' \leq |D_i|, P_{i,j} \leq P_{i,j'}$, i.e., receivers are ordered within each domain by increasing loss rate.

We define a multicast session to be ***intra-homogeneous*** if, for any i and for all $j, j', P_{i,j} = P_{i,j'}$, i.e., all receivers in the same domain have the same loss rate. A session that is not intra-homogeneous is ***intra-heterogeneous***. We define a session to be ***inter-homogeneous*** if for any two domains, D_i and $D_{i'}$, we have that $|D_i| = |D_{i'}|$ and for each $j \leq |D_i|, P_{i,j} = P_{i',j}$. If a session is not inter-homogeneous, it is defined to be ***inter-heterogeneous***.

We also assume that the sender always multicasts repairs, and the repair server always subcasts repairs. For simplicity, we again consider a network where all losses occur on the tail links, which lie between each receiver and its repair server (again, the reader is referred to Figure 1). This means that receiver losses are independent from one another, and that there are no losses within the backbone network (i.e., no losses between the sender and the repair server).

An examination of inter-homogeneous domains appears in [18]. Loss studies over the MBone [29, 30] suggest that loss rates differ in different regions of the Internet. Additionally, the domain size can vary considerably over the set of domains in a session. Consequently, we expect most multicast sessions to be inter-heterogeneous. We now examine an inter-heterogeneous, intra-homogeneous session, in which there are two types of domains. There are n_i domains of type i, where each domain of type i consists of r_i receivers, each having a loss rate of p_i, $i = 1, 2$. In this domain, we consider two protocols.

e-e: The e-e protocol is a hybrid, end-to-end FEC/ARQ protocol that does not make use of repair servers. The protocol details and its analysis can be found in [17].

SDBR: SDBR is described in Section 2. By setting the block size to one, this protocol performs identically to an RS repair server protocol.

Our measure of bandwidth is the number of packets (data plus repairs) that are sent per block of data. We define the random variable, N_i^S, to equal the number of packets multicast from a repair

server to downstream receivers for a block in SDBR. We extend (1) so that it applies to the various types of domains $(i = 1, 2)$:

$$E[N_i^S] = \sum_{j=0}^{\infty} 1 - \left[1 - \sum_{m=0}^{k-1} \gamma_m^{k+j}(1 - p_i)\right]^{r_i}$$

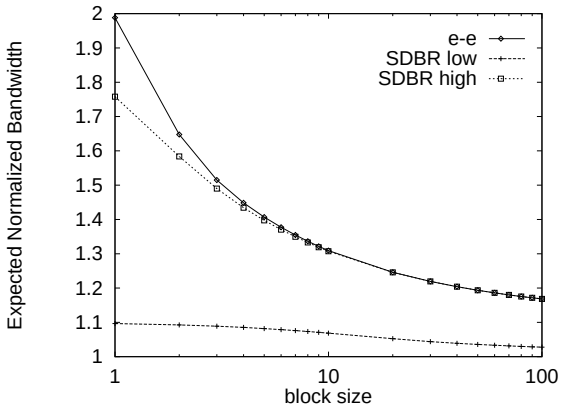
We define N^E to be a random variable that equals the number of (data and repair) transmissions in the e-e protocol that a sender must transmit to all downstream receivers (over all domains). Then

$$E[N^E] = \sum_{j=0}^{\infty} 1 - \prod_{i=1}^{2} \left[1 - \sum_{m=0}^{k-1} \gamma_m^{k+j}(1 - p_i)\right]^{r_i n_i}$$

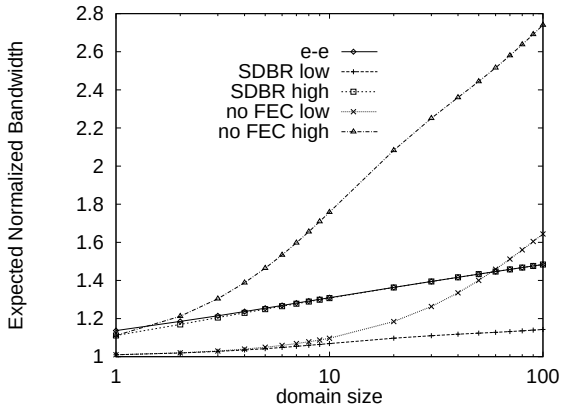
We also define F_i to a random variable that equals the number of transmissions in a domain of type i by a repair server using a protocol that does not implement FEC. $E[F_i]$ equals $E[N_i^S]$ when $k = 1$, and can be written more succinctly as

$$E[F_i] = \sum_{j=0}^{\infty} 1 - (1 - {p_i}^j)^{r_i}$$

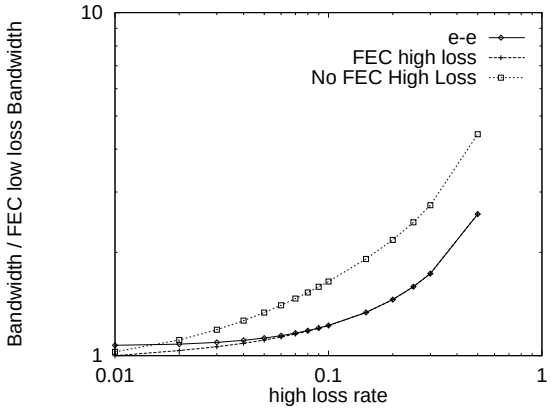
Figure 8 compares the number of expected transmissions from a repair server to receivers in its repair domain using an APES approach to non-APES approaches as (a) the block size, k, varies, (b) the domain size varies, and (c) the high loss rate varies along the x-axis. As a default, the block size, k, equals ten, the domain size of each domain, r_i, is ten, and there are ten domains of type one with loss rates from repair server to receiver of $p_1 = .01$, and a single domain of type two with loss rates of $p_2 = .1$. In Figures 8(a) and 8(b), the block size, k, and the domain size vary along the x-axis, respectively. Figure 8(a) plots curves, where the y-axis equals the normalized, expected cost of reliably transmitting a data packet. The curves labeled **e-e**, **SDBR low**, and **SDBR high** plot values of $E[N^E]/k$, $E[N_1^S]/k$, and $E[N_2^S]/k$, respectively. Note that values for $E[F_1]$ and $E[F_2]$ are given by respective plots **SDBR low** and **SDBR high** when the block size, k, equals one. Figure 8(b) includes additional plots for $E[F_1]$ and $E[F_2]$, respectively labeled **no FEC low** and **no FEC high**. In Figure 8(c), the loss rate in the type two, high loss, domain is varied along the x-axis, while the y-axis equals the normalized, expected cost of reliably transmitting a data packet



(a) Varying block size



(b) Varying domain size



(c) Varying high loss rate

Figure 8: Expected number of packets transmitted to receivers in inter-heterogeneous, intra-homogeneous networks varying (a) block size, (b) repair domain size, and (c) high loss rate.

in multiples of this cost in a low loss (type one) domain. Plots labeled **e-e**, **FEC high loss**, and **No FEC high loss** respectively plot values for $E[N^E]/E[N_1^S]$, $E[N_2^S]/E[N_1^S]$, and $kE[F_2]/E[N_1^S]$.

From Figures 8(a) and 8(b), we see that the bandwidth used by the e-e protocol throughout the network is almost identical to what is required in the high loss domain, which means low loss domains incur significant additional bandwidth. In Figure 8(c), we see that in an intra-homogeneous, intra-heterogeneous network, there is a small savings in bandwidth due to the use of repair services. As we increase the inter-heterogeneity, adding FEC reduces bandwidth further in protocols that make use of repair services, particularly in high loss domains. This behavior is observed in Figure 8(a), as well as in Figure 8(b) by comparing SDBR and the RS protocol for either the high

or low loss domain. We also observed little variation in bandwidth usage as the ratio of the number of low loss domains to the number of high loss domains, n_1/n_2, increases. The plot has been omitted due to lack of variation along x-axis.

We conclude from these observations that *the bandwidth used throughout the network in an end-to-end FEC approach is dominated by the bandwidth required by domains with high loss, whereas in a network with repair servers, this bandwidth consumption can be limited to the domains where it is required. An additional savings in bandwidth comes as a result of using FEC, particularly in high loss or large domains.* We expect the improvement due to FEC to be less if the loss between repair server and receivers has a high level of spatial correlation. However, our analysis in Section 3 showed that using FEC still improves performance by reducing buffering requirements.

6 Conclusion

We considered novel protocols that combine repair services approaches and hybrid FEC/ARQ approaches. We have examined and compared performance of protocols that combine these approaches, which we call APES protocols. We described several APES protocols that maintain this high bandwidth efficiency while reducing buffer and FEC processing requirements. We thoroughly explored how the bandwidth, buffer, and FEC processing requirements vary among the various versions of the protocols. We also describe the kinds of networking environments in which such protocols improve the bandwidth efficiency of reliable multicast beyond what either approach is able to accomplish separately.

We now turn our attention to spatially correlated loss. Spatial correlation was examined in [18] for SDBR in intra-heterogeneous, inter-homogeneous environments. The techniques presented there can easily be applied to examine the effects of spatial correlation on inter-heterogeneous environments, and on BRSR and GRSR. We also point out that a domain size of one is equivalent to assuming that all losses within a domain are 100% correlated. Thus, the bandwidth utilized in a within a domain size of r with partially correlated loss lies somewhere between the results plotted for a domain size of r and those plotted for a domain size of one. While correlated loss will limit the savings in bandwidth obtained by using FEC, but will increase the savings in buffer to maintain

the same repair server miss probability.

References

[1] J. Blomer, M. Kalfane, R. Karp, M. Karpinski, M. Luby, and D. Zuckerman. An XOR-Based Erasure-Resilient Coding Scheme. Technical report, International Computer Sciences Institute, ICSI TR-95-048, August 1995.

[2] J. Bolot, T. Turletti, and I. Wakeman. Scalable Feedback Control for Multicast Video Distribution in the Internet. In *Proceedings of ACM SIGCOMM'94*, pages 58–67, London, U.K., August 1994.

[3] J. Byers, M. Luby, and M. Mitzenmacher. Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speed Up Downloads. In *Proceedings of IEEE INFOCOM'99*, New York, NY, March 1999.

[4] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A Digital Fountain Approach to Reliable Distribution of Bulk Data. In *Proceedings of SIGCOMM'98*, Vancouver, Canada, September 1998.

[5] Y. Chawathe, S. McCanne, and E. Brewer. RMX: Reliable Multicast for Heterogeneous Networks. In *Proceedings of IEEE INFOCOM'00*, Tel-Aviv, Israel, March 2000.

[6] D. DeLucia and K. Obraczka. Multicast Feedback Suppression Using Representatives. In *Proceedings of INFOCOM'97*, Kobe, Japan, March 1997.

[7] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang. A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing. *IEEE/ACM Transactions on Networking*, 5(6):784–803, December 1997.

[8] J. Jannotti, D. Gifford, K. Johnson, M. Kaashoek, and J. O'Toole. Overcast: Reliable Multicasting with an Overlay Network. In *Proceedings of USENIX*, San Diego, CA, October 2000.

[9] S. Kasera, J. Kurose, and D. Towsley. Scalable Reliable Multicast Using Multiple Multicast Groups. In *Proceedings of ACM SIGMETRICS'97*, Seattle, WA, June 1997.

[10] S. Kasera, J. Kurose, and D. Towsley. A Comparison of Server-Based and Receiver-Based Local Recovery Approaches for Scalable Reliable Multicast. In *Proceedings of INFOCOM'98*, San Francisco, CA, March 1998.

[11] R. Kermode. Scoped Hybrid Automatic Repeat Request with Forward Error Correction (SHARQFEC). In *Proceedings of SIGCOMM'98*, Vancouver, Canada, September 1998.

[12] L.H. Lehman, S.J. Garland, and D.L. Tennenhouse. Active Reliable Multicast. In *Proceedings of INFOCOM'98*, San Francisco, CA, March 1998.

[13] B. N. Levine, S. Paul, and J.J. Garcia-Luna-Aceves. Organizing multicast receivers deterministically according to packet-loss correlation. In *Proceedings of ACM Multimedia'98*, Bristol, U.K., September 1998.

[14] X. Li, S. Paul, and M. Ammar. Layered Video Multicast with Retransmissions (LVMR): Evaluation of Hierarchical Rate Control. In *Proceedings of IEEE INFOCOM'99*, New York, NY, March 1999.

[15] N.F. Maxmchuk, K. Padmanabhan, and S. Lo. A Cooperative Packet Recovery Protocol for Multicast Video. In *Proceedings of ICNP'97*, Atlanta, GA, October 1997.

[16] J. Nonnenmacher and E. Biersack. Scalable Feedbackfor Large Groups. *Transactions on Networking*, June 1999.

[17] J. Nonnenmacher, E. Biersack, and D. Towsley. Parity-Based Loss Recovery for Reliable Multicast Tranmission. *Transactions on Networking*, August 1998.

[18] J. Nonnenmacher, M. Lacher, M. Jung, E. Biersack, and G. Carle. How Bad is Reliable Multicast without Local Recovery. In *Proceedings of INFOCOM'98*, San Francisco, CA, March 1998.

[19] C. Pappadopoulos, G. Parulkar, and G. Varghese. An Error Control Scheme for Large-Scale Multicast Applications. In *Proceedings of INFOCOM'98*, San Francisco, CA, March 1998.

[20] S. Paul, K.K. Sabnani, J.C. Lin, and S.Bhattacharyya. Reliable Multicast Transport Protocol (RMTP). *IEEE JSAC*, 15(3):407–421, April 1997.

[21] I. Rhee, S. Joshi, M. Lee, S. Muthukrishnan, and V. Ozdemir. Layered Multicast Recovery. In *Proceedings of IEEE INFOCOM'00*, Tel-Aviv, Israel, March 2000.

[22] L. Rizzo. Effective Erasure Codes for Reliable Computer Communication Protocols. *Computer Communication Review*, April 1997.

[23] D. Rubenstein, S. Kasera, D. Towsley, and J. Kurose. Improving Reliable Multicast Using Active Parity Encoding Services (APES). Technical report, University of Massachusetts, CMPSCI 98-79, July 1998.

[24] D. Rubenstein, J. Kurose, and D. Towsley. A Study of Proactive Hybrid FEC/ARQ and Scalable Feedback Techniques for Reliable, Real-time Multicast. *Computer Communications*, 24(5-6), March 2001.

[25] D. Rubenstein, N.F. Maxemchuk, and D. Shur. A Centralized Approach to Network Repair Service for Multicast Streaming Media. In *Proceedings of NOSSDAV'00*, Chapel Hill, NC, June 2000.

[26] K. Sripanidkulchai, A.C. Meyers, and H. Zhang. A Third-Party Value-Added Network Service Approach to Reliable Multicast. In *Proceedings of ACM SIGMETRICS'99*, Atlanta, GA, May 1999.

[27] D. Towsley, J. Kurose, and S. Pingali. A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols. *IEEE JSAC*, 15(3):398–406, April 1997.

[28] R.X. Xu, A.C. Meyers, and H. Zhang. Resilient Multicast Support for Continuous Media Applications. In *Proceedings of NOSSDAV'97*, St. Louis, MO, July 1997.

[29] M. Yajnik, J. Kurose, and D. Towsley. Packet Loss Correlation in the MBone Multicast Network. In *Proceedings of the IEEE Global Internet Conference*, London, UK, November 1996.

[30] M. Yajnik, S.B. Moon, J. Kurose, and D. Towsley. Measurement and Modeling of the Temporal Dependence in Packet Loss. In *Proceedings of IEEE INFOCOM'99*, New York, NY, March 1999.