Resilient Multicast using Overlays*

Suman Banerjee, Seungjoon Lee, Bobby Bhattacharjee, Aravind Srinivasan

ABSTRACT

We introduce PRM (Probabilistic Resilient Multicast): a multicast data recovery scheme that improves data delivery ratios while maintaining low end-to-end latencies. PRM has both a proactive and a reactive component; in this paper we describe how PRM can be used to improve the performance of application-layer multicast protocols, especially when there are high packet losses and host failures. Further, using analytic techniques, we show that PRM can guarantee arbitrarily high data delivery ratios and low latency bounds.

As a detailed case study, we show how PRM can be applied to the NICE application-layer multicast protocol. We present detailed simulations of the PRM-enhanced NICE protocol for 10,000 node Internet-like topologies. Simulations show that PRM achieves a high delivery ratio ($> 97\%$) with a low latency bound (600 ms) for environments with high end-to-end network losses (1-5%) and high topology change rates (5 changes per second) while incurring very low overheads ($< 5\%$).

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols; C.2.4 [**Computer-Communication Networks**]: Distributed Systems; C.4 [**Computer Systems Organization**]: Performance of Systems

General Terms

Algorithms, Design, Performance, Experimentation

*The authors are with the Department of Computer Science, University of Maryland, College Park, MD 20742, USA. B. Bhattacharjee and A. Srinivasan are also with the Institute for Advanced Computer Studies, University of Maryland. S. Banerjee, S. Lee, and B. Bhattacharjee were supported in part by NSF award ANI 0092806. A. Srinivasan was supported in part by NSF Award CCR-0208005.

SIGMETRICS'03, June 10–14, 2003, San Diego, California, USA.

Keywords

Overlay multicast, Resilience, Randomized forwarding

1. INTRODUCTION

We present a fast multicast data recovery scheme that achieves high delivery ratios with low overheads. Our technique, called Probabilistic Resilient Multicast (PRM), is especially useful for applications that can benefit from low data losses without requiring perfect reliability. Examples of such applications are real-time audio and video streaming applications where the playback quality at the receivers improves if the delivery ratios can be increased within specific latency bounds. Using terminology defined in prior literature [22] we call this model of data delivery *resilient multicast.*

In this paper we describe PRM in the context of overlay-based multicast [6, 8, 2, 27, 5, 21, 12]. Unlike native multicast where data packets are replicated at routers inside the network, in application-layer multicast data packets are replicated at end hosts. Logically, the end-hosts form an overlay network, and the goal of application-layer multicast is to construct and maintain an efficient overlay for data transmission. The eventual data delivery path in application-layer multicast is an overlay tree. While network-layer multicast makes the most efficient use of network resources, its limited deployment in the Internet makes application-layer multicast a more viable choice for group communication over the wide-area Internet.

A key challenge in constructing a resilient application-layer multicast protocol is to provide fast data recovery when overlay node failures partition the data delivery paths. Overlay nodes are processes on regular end-hosts which are potentially more susceptible to failures than the routers. Each such failure of a non-leaf overlay node causes data outage for nodes downstream until the time the data delivery tree is reconstructed. Losses due to overlay node failures are more significant than regular packet losses in the network and may cause data outage in the order of tens of seconds (e.g. the Narada application-layer multicast protocol [6] sets default timeouts between 30-60 seconds).

PRM uses two simple techniques:

(i) A proactive component called *Randomized forwarding* in which each overlay node chooses a constant number of other overlay nodes uniformly at random and forwards data to each of them with a low probability (e.g. 0.01-0.03). This randomized forwarding technique operates in conjunction with the usual data forwarding mechanisms along the tree edges, and may lead to a small number of duplicate packet

deliveries. Such duplicates are detected and suppressed using sequence numbers. The randomized component incurs very low additional overheads and can guarantee high delivery ratios even under high rates of overlay node failures.

(ii) A reactive mechanism called *Triggered NAKs* to handle data losses due to link errors and network congestion.

Through analysis and detailed simulations we show that these relatively simple techniques provide high resilience guarantees, both in theory and practice. PRM can be used to significantly augment the data delivery ratios of *any* application-layer multicast protocol (e.g. Narada [6], Yoid [8], NICE [2], HMTP [27], Scribe [5], Delaunay Triangulation-based [12], CAN-multicast [21]) while maintaining low latency bounds.

The contributions of this paper are three-fold. First, we propose a simple, low-overhead scheme for resilient multicast. To the best of our knowledge, this work is the first proposed resilient multicast scheme that can be used to augment the performance of application-layer multicast protocols. Second, we present a rigorous analysis to show that a simple randomized approach is sufficient to achieve high data delivery rates within low latency bounds. Finally, we demonstrate how our proposed scheme can be used with an existing application-layer multicast protocol (NICE [2]) to provide a low overhead, low latency and high delivery ratio multicast technique for realistic applications and scenarios.

The rest of this paper is structured as follows. In the next section we present the details of the PRM scheme and analyze its performance in Section 3. In Section 4 we present detailed simulation studies of the the PRM-enhanced NICE protocol. In Section 5 we describe related work and conclude in Section 6. The Appendix gives a brief sketch of the proofs.

2. Probabilistic Resilient Multicast (PRM)

The PRM scheme employs two mechanisms to provide resilience. We describe each of them in turn.

2.1 Randomized Forwarding

In randomized forwarding, each overlay node, with a small probability, proactively sends a few extra transmissions along randomly chosen overlay edges. Such a construction interconnects the data delivery tree with some cross edges and is responsible for fast data recovery in PRM under high failure rates of overlay nodes. Existing approaches for resilient and reliable multicast use either reactive retransmissions (e.g. RMTP [20], STORM [22] Lorax [14]) or proactive error correction codes (e.g. Digital Fountain [3]) and can only recover from packet losses on the overlay links. Therefore the proactive randomized forwarding is a key difference between our approach and other well-known existing approaches.

We explain the specific details of proactive randomized forwarding using the example shown in Figure 1. In the original data delivery tree (Panel 0), each overlay node forwards data to its children along its tree edges. However, due to network losses on overlay links (e.g. $\langle A, D\rangle$ and $\langle B, F\rangle$) or failure of overlay nodes (e.g. C, L and Q) a subset of existing overlay nodes do not receive the packet (e.g. D, F, G, H, J, K and M). We remedy this as follows. When any overlay node receives the first copy of a data packet, it forwards the data along all other tree edges (Panel 1). It also chooses a small number (r) of other overlay nodes and forwards data to each of them with a small probability, β. For example node E chooses to forward data to two other nodes using cross edges F and M. Note that as a consequence of these additional edges some nodes may receive multiple copies of the same packet (e.g. node T in Panel 1 receives the data along the tree edge $\langle B, T\rangle$ and cross edge $\langle P, T\rangle$). Therefore each overlay node needs to detect and suppress such duplicate packets. Each overlay node maintains a small duplicate suppression cache, which temporarily stores the set of data packets received over a small time window. Data packets that miss the latency deadline are dropped. Hence the size of the cache is limited by the latency deadline desired by the application. In practice, the duplicate suppression cache can be implemented using the playback buffer already maintained by streaming media applications. It is easy to see that each node on average sends or receives upto $1 + \beta r$ copies of the same packet. The overhead of this scheme is βr, where we choose β to be a small value (e.g. 0.01) and r to be between 1 and 3. In our analysis we show that if the destinations of these cross edges are chosen uniformly at random, it is possible to guarantee successful reception of packets at each overlay node with a high probability.

Each overlay node periodically discovers a set of random other nodes on the overlay and evaluates the number of losses that it shares with these random nodes. In an overlay construction protocol like Narada [6], each node maintains state information about all other nodes. Therefore, no additional discovery of nodes is necessary in this case. For some other protocols like Yoid [8] and NICE [2] overlay nodes maintain information of only a small subset of other nodes in the topology. Therefore we implement a node discovery mechanism, using a random-walk on the overlay tree. A similar technique has been used in Yoid [8] to discover random overlay group members. The discovering node transmits a *Discover* message with a *time-to-live* (TTL) field to its parent on the tree. The message is randomly forwarded from neighbor to neighbor, without re-tracing its path along the tree and the TTL field is decremented at each hop. The node at which the TTL reaches zero is chosen as the random node.

Why is Randomized Forwarding effective? It is interesting to observe why such a simple, low-overhead randomized forwarding technique is able to increase packet delivery ratios with a high probability, especially when many overlay nodes fail. Consider the example shown in Figure 2, where a large fraction of the nodes have failed in the shaded region. In particular, the root of the sub-tree, node A, has also failed. So if no forwarding is performed along cross edges, the entire shaded sub-tree is partitioned from the data delivery tree. No overlay node in this entire sub-tree would get data packets till the partition is repaired. However using randomized forwarding along cross edges a number of nodes from the unshaded region will have random edges into the shaded region as shown ($\langle M, X\rangle, \langle N, Y\rangle$ and $\langle P, Z\rangle$). The overlay nodes that receive data along such randomly chosen cross edges will subsequently forward data along regular tree edges and any chosen random edges. Since the cross edges are chosen uniformly at random, a large subtree will have a higher probability of cross edges being incident on it. Thus as the size of a partition increases, so does its chance of repair using cross edges.

2.2 Triggered NAKs

This is the reactive component of PRM. We assume that the application source identifies each data unit using monotonically increasing sequence numbers. An overlay node can

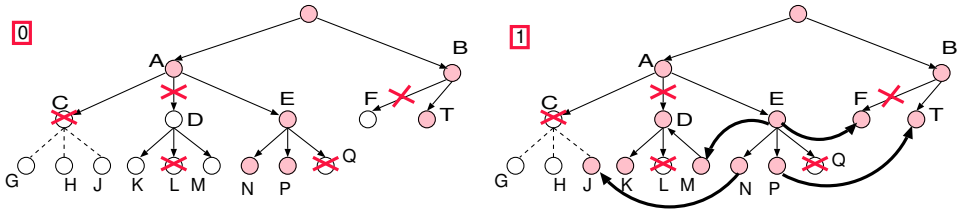

Figure 1: The basic idea of the PRM scheme. The circles represent the overlay nodes. The crosses indicate link and node failures. The arrows indicate the direction of data flow. The curved edges indicate the chosen cross overlay links for randomized forwarding of data.

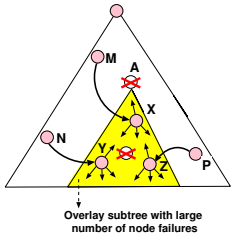

Figure 2: Successful delivery with high probability even under high node failure rate.

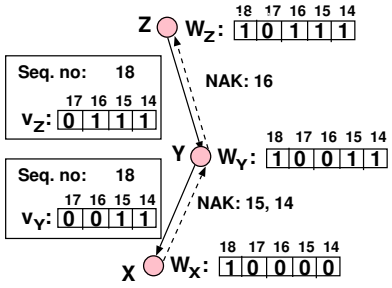

Figure 3: Triggered NAKs to parent on the overlay tree when data unit with sequence number 18 is propagated along the overlay. The length of the bit-mask is 4.

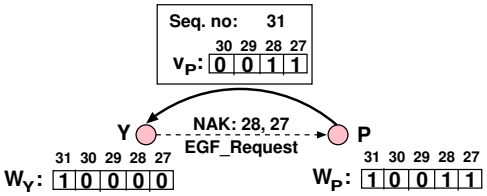

Figure 4: Triggered NAKs to source of random forwarding for data with sequence number 31. The value of Δ for Ephemeral Guaranteed Forwarding is set to 3.

detect missing data using gaps in the sequence numbers. This information is used to trigger NAK-based retransmissions. This technique has been applied for loss repair in RMTP [20].

In our implementation each overlay node, x, piggybacks a bit-mask with each forwarded data packet indicating which of the prior sequence numbers it has correctly received. The recipient of the data packet, y, detects missing packets using the gaps in the received sequence and sends NAKs to x to request the appropriate retransmissions. Note that x can either be a parent of y in the data delivery tree, or a random node forwarding along a cross edge. We illustrate the use of triggered NAKs in Figures 3 and 4.

2.3 Extensions

We describe two extensions to the PRM scheme that further improve the resilience of the data delivery.

Loss Correlation. This is a technique that can be used to improve the randomized forwarding component of PRM. As described in Section 2.1 each overlay node chooses a small number of cross edges completely at random for probabilistic data forwarding on the overlay. In practice, it is possible to increase the utility of these cross edges by choosing them more carefully. In particular if s is the root (and source) of the overlay tree, we want to choose a cross edge between two overlay nodes x and y, if and only if the correlation between the packets lost on the overlay paths $\langle s \longmapsto x \rangle$ and $\langle s \longmapsto y \rangle$ is low. Clearly if these two overlay paths share no underlying physical links, then we expect the losses experienced by x and y to be uncorrelated. However, such a condition is difficult to guarantee for any overlay protocol. Therefore under the loss correlation extension, we let each node x to choose a random edge destination, y, with which has the minimum number of common losses over a limited time window.

Ephemeral Guaranteed Forwarding (EGF). This is an extension to the triggered NAK component and is also useful in increasing the data delivery ratio. Consider the case when node x receives a data packet with sequence number j along a random edge from node y. If on receiving the data packet with sequence number j, x detects a large gap (greater than a threshold Δ) in the sequence number space it is likely that the parent of x has failed. In this case, x can request y to increase the random forwarding probability, β, for the edge $\langle y, x \rangle$ to one for a short duration of time. To do this x sends an *EGF_Request* message to y. Note that the EGF state is soft and expires within a time period T_{EGF}. This is shown by an example in Figure 4. Node Y receives data with sequence number 31 along a random edge from P. Y immediately requests retransmissions for data with sequence numbers 28 and 27. Since $\Delta = 3$, Y also sends the *EGF_Request* message to P. If the tree path of x is repaired before the EGF period expires, x can also send an *EGF_Cancel* message to y to terminate this state.

The EGF mechanism is useful for providing uninterrupted data service when the overlay construction protocol is detecting and repairing a partition in the data delivery tree. In fact, putting such a mechanism in place allows the over-

lay nodes to use larger timeouts to detect failure of overlay peers. This, in turn, reduces the control overheads of the application-layer multicast protocol. In practice, the EGF mechanism can sometimes be overly aggressive and cause false positives leading to a higher amount of data duplication on the data delivery tree. Thus, the improvement in performance is at the cost of additional state, complexity and packet overhead at nodes. Depending on the reliability requirements, applications may choose to enable or disable EGF.

3. EVALUATION OF PRM

A key component of the PRM scheme is the randomized forwarding technique which achieves high delivery ratios in spite of a large number of overlay node failures. In this section we first analytically prove that even a "simplified" randomized forwarding scheme can provide high probability delivery guarantees when only a constant number of random edges are used by each overlay node. We also show that this simplified scheme achieves good latency bounds. Subsequently we present simulation results of the full version of the PRM scheme. The simulation results presented in this section describe the performance of the PRM scheme without modeling an underlying application-layer multicast protocol. Therefore these simulations only model random failures of overlay nodes and random packet losses on overlay links for an overlay tree. We have also implemented PRM over a specific application-layer multicast protocol, NICE [2]. We present detailed performance studies which includes the consequent interactions between PRM and NICE in Section 4.

3.1 Analysis of Simplified PRM

For the tractability of the analysis we consider a simplified version of the PRM scheme (see Figure 13 in the Appendix). A parent node forwards data to all of its children along the overlay tree edges. Each node chooses r such distinct random edges, and data is forwarded independently along each such random edge with probability β. However, data packets received along random edges can be subsequently forwarded only along other random edges. For example, in Figure 13, node K receives data along the random edge $\langle N, K\rangle$. It is allowed to forward this data packet along another random edge (e.g. $\langle K, G\rangle$, but not to its parent D (or children, if it had any). We analyze two different versions of this simplified scheme: (i) the random edges are restricted between nodes on the same level of the tree, and (ii) the random edges can go to arbitrary other nodes. These simplified versions impose stricter data-forwarding rules on the data delivery tree than the complete version described in Section 2; thus, the analytic bounds for the probability of successful data-delivery that we obtain for these schemes serve as lower bounds for our actual proposed scheme. The additional data overhead of the PRM scheme is given by $r\beta$, which is a constant independent of the size of the overlay. We now show that even with such a constant data overhead, the simplified PRM scheme *scales* well; that is, even as the number of nodes in the overlay (n) increases, the guaranteed bounds on reliability and latency hold with arbitrarily high probability.

Background and Main Results

The overlay data delivery path is a tree with the source node R as the root, at level 0. R does not fail. Every other overlay node does not fail with some probability and each overlay link has a packet loss probability; all these failure events are independent. The total number of nodes is n. We will make the following assumptions about the overlay tree and the failure probabilities.

(A1) The tree has some depth d; for $i = 0, 1, \ldots, d-1$, all nodes at level i have some D_i children. We will later take D_0 to be a sufficiently large constant (which depends on the success probability we need). The only other requirement is that $D_i \geq 2$ for $i \geq 1$.

(A2) Call a node "good" if it did not fail, and if it received the data from its parent (i.e., without a link loss). Then, for any node u, let S_u be the expected number of good children of u, conditional on u being good. There are constants $\alpha > 0$ and $\lambda > 1$ such that for any node u at any level $i \leq d-1$, $S_u \geq D_i\alpha \geq \lambda$.

(A3) Let A_i be the number of overlay nodes in level i that are expected to survive. There is some constant $\gamma > 0$ such that for all i, $A_i \geq \gamma \cdot D_0 D_1 \cdots D_{i-1}$; in other words, some constant fraction (which is γ here) of the overlay nodes at any level i are expected to survive.

In the Appendix we discuss how some of these assumptions can be relaxed, and how the remaining are necessary. Our two main results are as follows. Both of them show that almost all surviving overlay nodes get the data with high probability, even if the overhead $r\beta$ is only some *constant* that is independent of n; thus, the protocols scale with system-size. Furthermore, the first result shows that when the random forwarding is done within the same level, we achieve data delivery with high probability with low latency: every surviving overlay node at any level i gets the data using $O(i)$ overlay hops with high probability. Thus, every surviving overlay node receives the data within a constant factor bound of its end-to-end latency overlay from the root.

THEOREM 3.1. *Consider the case where the random forwarding is only done within the same level. Suppose we are given an arbitrary constant (confidence parameter) $\delta \in (0,1)$. Then, there is a constant threshold t_0 that is $O(1/\delta)$ and another constant $D_0(\delta)$ such that if $r\beta \geq t_0$ and $D \geq D_0(\delta)$, then the following holds with probability at least $1-\delta$: For every level i, at least an $(1-\delta)$–fraction of the overlay nodes in that level which survive, receive the data from the root; also, they do so within $O(i)$ steps.*

The next result deals with the case where random forwarding is done to arbitrary overlay nodes in the tree; here, the worst-case waiting time is $O(\log n)$, with high probability.

THEOREM 3.2. *Consider the case where the random forwarding is done to arbitrary other overlay nodes in the tree. Suppose we are given an arbitrary constant (confidence parameter) $\delta \in (0,1)$. Then, there is a constant threshold t_0 that is $O(1/\delta)$ and another constant $D_0(\delta)$ such that if $r\beta \geq t_0$ and $D \geq D_0(\delta)$, then the following holds with probability at least $1-\delta$: At least an $(1-\delta)$–fraction of the overlay nodes which survive, receive the data from the root; also, they do so within $O(\log n)$ steps.*

Note that the two theorems involve (slightly) different schemes and latency bounds; their proofs are sketched in

the Appendix, and require a careful probabilistic analysis. One of the main delicate issues related to the randomized forwarding crops up when nearly all the required overlay nodes have received the data: from this point on, much of the random forwarding is likely to go to the nodes that have already received the data. In particular, as sketched in the Appendix, if $r\beta$ does not grow linearly with $1/\delta$, with high probability we will not reach the number of surviving overlay nodes that we aim to. Thus, the communication overheads of our scheme are optimal. In essence, we show that the randomized forwarding percolates through the network in two distinct epochs: the first, when the number of overlay nodes reached yet is "not too large" – the rate of growth of this number is fast enough here; and the second, when the rate of growth is (much) smaller as indicated a few sentences above. We are able to show that with a suitably-chosen value of $r\beta$ and a careful analysis, the second epoch succeeds with high probability.

The linear growth of the overheads, $r\beta$ with $1/\delta$ holds only for the simplified, restricted version of the PRM scheme. Note that in the restricted version, data received along cross edges are not forwarded on any tree edge. The full version of PRM does not have such a restriction and therefore incurs significantly lesser overheads. We examine the different aspects of the full version of PRM next.

3.2 Simulations in an Idealized Environment

The above theorems show that even the simplified version of PRM can maintain good reliability properties with high probability. Now we show that the full version of PRM is also performs quite well in practice. In these idealized simulations, we assume that there exists some application-layer multicast protocol which constructs and maintains a data delivery tree. When data packets are sent on this tree, a subset of the overlay nodes, chosen at random, fail simultaneously. The failed subset is chosen independently for each data packet sent. Additionally data packets also experience network layer losses on overlay links. We present a more realistic simulation study in Section 4 where we explore the vagaries of implementing the PRM scheme over a specific application-layer multicast protocol.

In this study, we performed simulation studies comparing three different schemes:

BE: This is the best-effort multicast scheme and has no reliability mechanisms and therefore serves as a baseline for comparison.

HHR: This is a reliable multicast scheme, where each overlay link employs a hop-by-hop reliability mechanism using negative acknowledgments (NAKs). A downstream overlay node sends NAKs to its immediate upstream overlay node indicating packet losses. On receiving these NAKs the upstream overlay node retransmits the packet. This scheme, therefore, hides all packet losses on the overlay links.

FEC-y: This is an idealized equivalent of an FEC-based reliability scheme with a stretch factor of $1+y$. The overhead of this scheme, therefore, is y. In this idealized scenario we assume that if x fraction of packets are lost on the end-to-end overlay path, a receiver recovers $\min(1, x(1+y))$ fraction of the packets. In practice the delivery ratio of an FEC-based scheme depends on the size of the encoding. However, for low overhead ranges the above approximation provides a reasonable upper-bound. We perform more detailed comparisons of PRM with FEC-based schemes in Section 4.

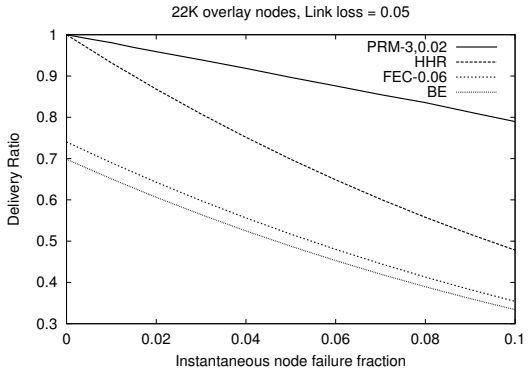

Figure 5: Delivery ratio of different schemes as the fraction of instantaneously failed overlay nodes is varied.

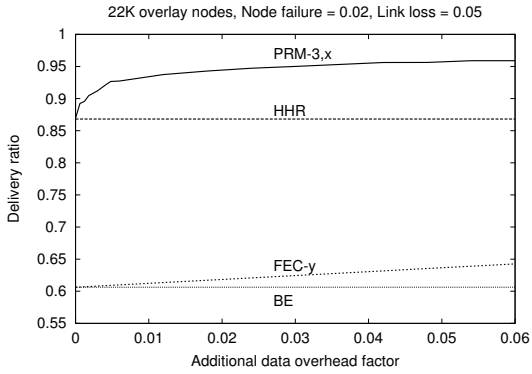

Figure 6: Delivery ratio of different schemes as the data overhead is varied.

PRM-r,β: This is the PRM scheme. r indicates the number of random nodes chosen by each node, and β is the forwarding probability for each of these random edges. This scheme as defined in Section 2 employs the hop-by-hop reliability of overlay links using the triggered NAK technique. PRM incurs $r\beta$ additional data overheads above the best-effort scheme.

We compare the data delivery ratio achieved by the different schemes for varying instantaneous node failure rates and the additional overheads incurred. We examine the data delivery ratio of leaf nodes, defined as the ratio of the number of data packets successfully received to the number of data packets sent by the source. This is lower than the delivery ratio over all nodes, although the difference is small. Since these are not packet-level simulations, we defer the results of delivery latency to the next section.

Resilience against node failures. In Figure 5 we plot the delivery ratio of the different schemes as the number of *simultaneous* node failures on the overlay tree of size 22,000 is varied. The loss probability on each overlay link was 5% for this experiment. We can observe that the delivery ratio of the PRM scheme degrades gracefully with increase in the

number of simultaneous node failures in the overlay tree. For a very high failure rate scenario when 5% of the overlay nodes simultaneously fail (i.e. 1100 nodes) for each packet sent, PRM-3,0.02 incurs 6% additional data overheads and achieves about 90% successful data delivery to nodes, while the HHR scheme achieves about 70% successful delivery. The best-effort scheme with no reliability mechanism performs poorly. Note that the FEC-based scheme does not provide any significant benefits when the stretch factor of the code is low.

Additional Data Overheads. In Figure 6 we show how the data delivery ratio of the PRM scheme depends on the additional data overheads incurred. In this experiment we simulated the situation when the link loss probability was 0.05 and 2% of the overlay nodes (i.e. 440 overlay nodes) fail simultaneously for each data packet sent. While usual multicast groups typically will not see such a large number of simultaneous failures, a goal of this experimentation is to understand the resilience that can be achieved by the PRM scheme despite its low overheads. For this experiment we varied the β parameter of PRM to control the additional overheads of the scheme. Even with less than 1% overheads the PRM scheme is able to achieve a delivery ratio greater than 95%. FEC-0.06 achieves very little improvement over the best-effort scheme since its performance is marginal when only low data overheads are permissible. By introducing significantly higher overheads the data delivery ratio of the FEC scheme can be improved.

4. SIMULATION EXPERIMENTS

The PRM scheme can be applied to any overlay-based multicast data delivery protocol. In our detailed simulation study we implemented PRM over the NICE application-layer multicast protocol [2]. We used NICE because of three reasons: (1) the authors in [2] show that the NICE application-layer multicast protocol achieves good delivery ratios for a best-effort scheme; (2) NICE is a scalable protocol and therefore we could perform detailed packet-level simulations for large overlay topologies; (3) the source-code for NICE is publicly available.

We have studied the performance of the PRM-enhanced NICE protocol using detailed simulations. We compare the four schemes as described in Section 3.2, all of which use NICE as the underlying multicast protocol. We elaborate further on the details of the last two schemes here:

FEC-(d, r): This is an enhanced version of the NICE protocol in which the source uses a forward error correction mechanism to recover from packet losses. In this scheme, the source takes a set of d data packets and encodes them into a set of $d+r$ packets and sends this encoded data stream to the multicast group. A receiving member can recover the d data packets if it receives *any* d of the $r+d$ encoded packets [1]. Additional data overheads of this scheme are r/d.

PRM-(r, β): This is our proposed PRM enhancements implemented on the basic NICE application-layer multicast protocol. The meanings of r and β are the same as in Section 3.2. For all these experiments we implemented the loss correlation extensions to PRM. We enable EGF for only one specific experiment (described later) due to its higher overheads. Our results will show that EGF is useful for very dynamic scenarios, at the cost of higher data overheads.

[1]The Digital Fountain technique [3], uses Tornado codes that require the receiver to correctly receive $(1+\epsilon)d$ packets to recover the d data packets, where ϵ is a small constant.

Choosing FEC parameters. Since the FEC-based schemes need to send $d+r$ packets instead of d packets we use a higher data rate at the source (i.e. a data rate of $(d+r)/d$ times the data rate used by the other schemes). The resilience of an FEC-based scheme can be increased by increasing the overheads parameter, r. Also for the same amount of additional data overheads, resilience against network losses of the FEC-based schemes improve if we choose higher values of d and r. For example, FEC-(128,128) has better data recovery performance than FEC-(16,16) even though both have 100% overhead. This improved reliability comes at the cost of increased delivery latencies. Therefore, the maximum value of d and r depends on the latency deadline. We have experimented with a range of such choices upto the maximum possible value that will allow correct reception at receivers within the latency deadline. However, we observed that in presence of failures of overlay nodes increasing d and r does not always improve the resilience properties. This is because choosing higher values of d and r leads to increased latencies in data recovery. However when the group change rate is high the data delivery paths break before the FEC-based recovery can complete, and the achieved data delivery ratio is low.

4.1 Simulation Scenarios

In all these experiments we model the scenario of a source node multicasting streaming media to a group. The source sends CBR traffic at the rate of 16 packets per second. For all these experiments we chose a latency deadline of upto 8 seconds. As a consequence the size of the the packet buffer for NAK-based retransmissions is 128. The packet buffer will be larger for longer deadlines.

We performed detailed experiments with a wide-range of parameters to study different aspects of the PRM scheme. The network topologies were generated using the Transit-Stub graph model, using the GT-ITM topology generator [4]. All topologies in these simulations had $10,000$ routers with an average node degree between 3 and 4. End-hosts were attached to a set of routers, chosen uniformly at random, from among the stub-domain nodes. The number of such hosts in the multicast group were varied between 8 and 4096 for different experiments. Each physical link on the topology was modeled to have losses. Inter-domain links had 0.5-0.6% loss rates, while intra-domain links was about 0.1% loss rates. We also model bursty losses as follows: if a packet is lost on a physical link we increase the loss probability for subsequent packets received within a short time window. The average propagation and queueing latency on each physical link was between 2-10 ms. In all our experiments we use a *HeartBeat* period of 5 seconds for NICE and its extensions as was described by the authors in [2].

We have simulated a wide-range of topologies, group sizes, member join-leave patterns, and protocol parameters. In the experiments, all departures of end-hosts from the multicast group were modeled as "ungraceful leaves." This is equivalent to a host failure, where the departing member is unable to send a *Leave* message to the group.

In the experiments reported in these section, we first let a set of end-hosts join the multicast group and stabilize into

an appropriate multicast data delivery tree. Subsequently a traffic source end-host starts sending data group and end-hosts continuously join and leave the multicast group. The join and the leave rate for members are chosen to be equal so that the average size of the group remained nearly constant. The instants of group membership changes were drawn from an exponential distribution with a pre-defined mean, which varied between experiments. We studied the various data delivery properties of our proposed scheme over this dynamically changing phase of the experiment.

4.2 Simulation Results

We have studied the three metrics of interest: data delivery ratio, delivery latency and data overheads. The data overheads in PRM are because of duplication due to randomized forwarding, and due to redundant encoding in FEC-based schemes. We also examine the additional control overheads due to NAKs, random member discovery etc.

Delivery Ratio. In Figure 7 we show the delivery ratio of the different schemes as the frequency of changes to group membership is varied. The average size of the group was 512. The average loss rate for physical links for this experiment was 0.5%, which corresponds to between 2-5% end-to-end losses on each overlay link.

We plot the data delivery ratios as the group change rate is varied between 0 and 10 changes per second. Note that even 5 changes per second implies that 512 (which is also the size of the group) membership changes happen in less than two minutes! While such a high change rate is drastic, it is not improbable for very large distribution groups in the Internet. The PRM scheme is able to recover from a vast majority of these losses through the use of randomized forwarding mechanism. The delivery ratio for PRM-(3,0.01) is $> 97\%$ for a group membership change rate of 5 per second and $> 80\%$ for a group membership change rate of 10 per second. Additionally PRM incurs a very low (3%) additional data overhead.

The data delivery ratio for the best-effort protocol falls significantly (to about 0.35 for change rate of 10 group changes per second) with increase in the change rate to the overlay. In [2], the authors had shown that the NICE application-layer multicast protocol achieves good delivery ratios for a best-effort scheme, and is comparable to other best-effort application-layer multicast protocols, e.g. Narada [6]. Therefore, we believe that PRM-enhancements can significantly augment the data delivery ratios of all such protocols. An FEC-based scheme is typically able to recover from all network losses. However changes in the overlay data delivery path significantly impacts the performance of an FEC-based scheme. Note that the performance of the FEC-based scheme degrades with increasing frequency of group changes and even falls below the simple best-effort scheme for high group change rates.

In Figure 8 we compare the delivery ratio of the different schemes as the average packet loss rate on the physical links of the topology are varied. In this experiment, changes to the overlay topology (including both joins and leaves) occur with a mean of one change per second. In contrast to the other schemes, which suffer between 20% to 55% losses, the PRM-(3,0.01) scheme achieves near perfect data delivery under all data loss rates.

In Figure 9 we show how the delivery ratio achieved by

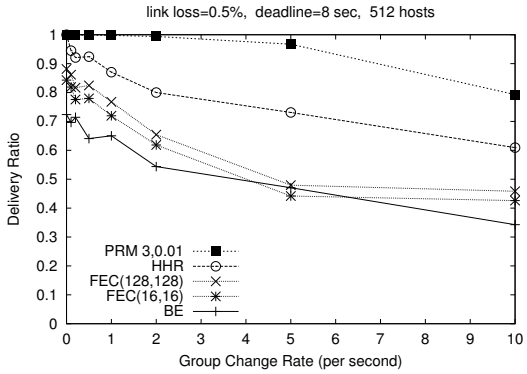

Figure 7: Delivery ratio with varying rate of changes to the group.

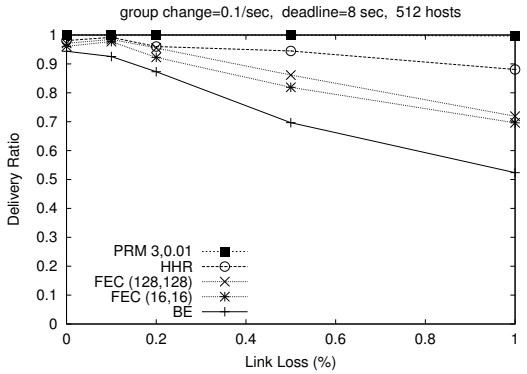

Figure 8: Delivery ratio with varying network loss rates.

the PRM scheme evolves over time in comparison to the best-effort protocol. In this experiment, the group change rate was one per second, i.e. the entire group can potentially change in less than 10 minutes. In the best-effort protocol, every data packet was not received by 20-40% of the group members. In contrast, the losses experienced in the PRM scheme was minimal. For the same experiment, we plot the cumulative distribution of the maximum data outage period of the overlay nodes in Figure 10. Most overlay nodes had no significant data outage period in PRM and more than 98% of the overlay nodes had a maximum outage period less than 5 seconds. This is a significant improvement over the best-effort protocol where more than 20% of the overlay nodes experience a maximum data outage of more than 30 seconds.

Delivery Latency. In Figure 11 we show the distribution of latency experienced by the data packets at the different overlay nodes. In this experiment, the average group membership change rate was 0.1 per second and the average loss probability at the physical links was 0.5%. Note that the latency of data packets is the lowest for the best-effort NICE protocol. This is because the best-effort scheme incurs

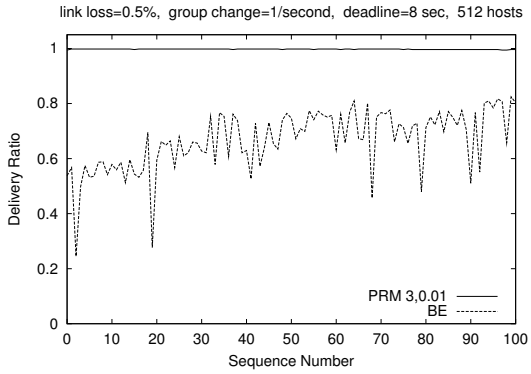

Figure 9: Time evolution of the delivery ratio for a sequence of data packets.

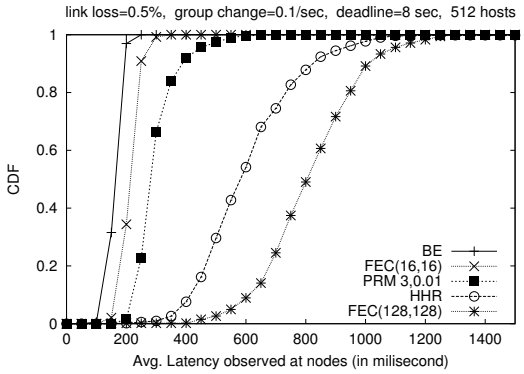

Figure 11: Cumulative distribution of average latency for packets received at the different nodes.

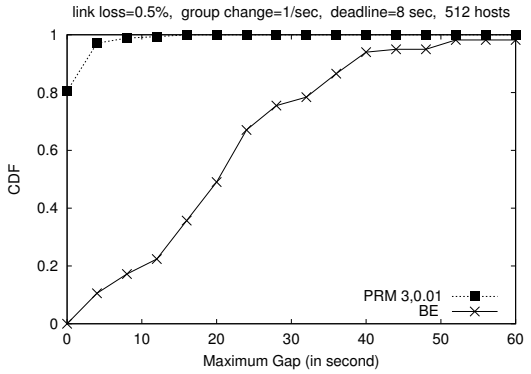

Figure 10: Cumulative distribution of the maximum time gap over which an overlay node lost all data packets.

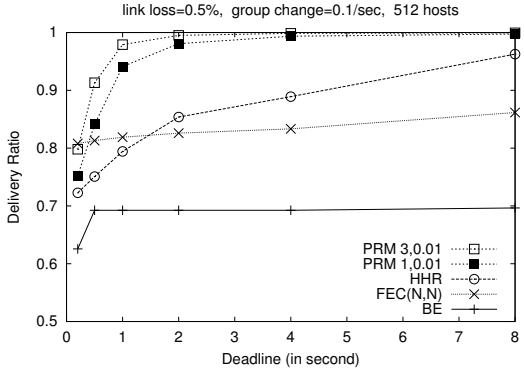

Figure 12: Delivery ratio achieved with varying deadline within which the data packets are to be successfully delivered.

no additional delay due to timeout-based retransmissions or delivery using alternate longer overlay paths. FEC-(16,16) incurs a slightly higher latency. This is because, to recover a lost data packet in the FEC-based scheme has to wait for additional (encoded) packets to arrive. FEC-(128,128) can decode data only after sufficiently many packets are received, and as a result, it incurs highest latency. The HHR scheme also suffers from high latency since it is purely reactive; in our simulations of HHR, packet losses due to node failure are often detected after the tree is recovered by underlying multicast protocol. The PRM scheme is a combination of proactive and reactive schemes and therefore incurs significantly lower latency than the HHR scheme. However data packets delivered using PRM still incur higher latency than the simple best-effort delivery. This is because many of the data packets that are lost on the shortest best-effort path are successfully delivered either using Triggered NAK-based retransmissions or randomized forwarding using a longer overlay path. More than 90% of the overlay nodes receive data packets within 500 ms, which is 2.5 times the worst case overlay latency on the topology.

In Figure 12 we show the effect of imposing a deadline for packet delivery. The deadline specifies a time upper bound within which a packet must reach an overlay node to be be useful to the application. For a deadline of x seconds, we allow different overlay nodes to have slightly different additional slacks in the time upper bound within which packets must be delivered. This slack is to account for the minimum latency that data packets encounter on the shortest path from the source to that overlay node. The maximum slack for any overlay node was less than 200 ms. The best-effort NICE protocol makes a single attempt for packet delivery and therefore achieves almost identical delivery ratio for all deadlines. The performance of the HHR scheme increases gradually with increase in the deadline due to its dependence on timeouts. In contrast, for short deadlines, the PRM scheme achieves rapid improvements due to its proactive component and further gradual improvements for longer deadlines due to its reactive component. For the FEC-based scheme we used $d = r$ and chose the value of d based on the deadline imposed. It achieved between 80-87% delivery ratios in these experiments.

scheme changes/sec, deadline (sec)	Delivery Ratio 80%	85%	90%	95%	99%
FEC, 0.1, 0.5	88-100	-	-	-	-
FEC, 0.1, 2.0	62-75	-	-	-	-
FEC, 0.1, 8.0	50-62	75-87	-	-	-
FEC, 0.1, 64.0	37-50	50-62	75-87	87-100	-
PRM, 1, 0.2	9-12	18-21	21-24	30-60	-
PRM, 1, 0.5	0-1	1-3	3-6	9-15	30-60
PRM, 1, 2.0	0-1	0-1	0-1	0-1	3-9
PRM, 1, 8.0	0-1	0-1	0-1	0-1	1-3

Table 1: Comparison of additional data overheads (in %) required for PRM and FEC-based schemes to meet different delivery ratios for specific group change rates and latency bounds. We do not report results for the cases when the overheads exceed 100% (marked by -).

Group Size	Control Overheads (pkts/sec) BE	PRM	Delivery ratio BE	PRM
128	2.9	4.0	0.68	0.99
256	3.3	4.4	0.58	0.99
512	3.4	4.7	0.60	0.99+
1024	4.1	5.5	0.51	0.98
2048	5.8	7.4	0.41	0.97
4096	10.1	13.5	0.40	0.97

Table 2: Comparison of best-effort and PRM-enhanced NICE protocols with varying group sizes for group change rate of 0.2% per second.

Additional Data Overheads. In Table 1, we compare the overheads of PRM and FEC-based schemes to achieve different delivery ratios. The table shows the additional data overheads for both schemes under different parameters (e.g. latency bounds, group change rate, etc.). The FEC-based schemes perform poorly when the frequency of changes on the overlay is high. Hence, we used an order of magnitude lower group change rates (0.1 changes/sec) for the FEC-based schemes than what we used for PRM (1 change/sec).

The table shows that PRM incurs very low additional data overheads to achieve relatively high delivery ratios within low latency bounds. For example for a group change rate of one per second and data latency bound of 0.5 seconds, the PRM scheme incurs 3-6% additional data overheads to achieve data delivery ratio of 90%. In fact for most of the scenarios shown in the table, PRM requires overheads less than 10% to meet the desired deadline and delivery ratios. PRM requires higher overheads for only the very stringent deadline of 200 ms and to achieve 99% delivery ratio for a 500 ms deadline. As is evident from the table, FEC-based schemes require far higher overheads even for much lower group change rates (0.1 per second).

Scalability. In Table 2 we show the effect of multicast group size on control overheads and delivery ratio. In the original best-effort NICE application-layer multicast, the control overheads at different overlay nodes increase logarithmically with increase in group size. The control overheads for the PRM-enhanced NICE is higher due to the additional messages, which include random node *Discover* messages, NAKs, etc. However this overhead, less than 3.5 extra control packets per second, at any overlay node is negligible in comparison to data rates that will be used by the applications (e.g. media streaming). We also observe that the data delivery ratio of the PRM-enhanced NICE protocol remains fairly constant across various group sizes.

Loss Correlation and EGF. Due to space constraints, we briefly describe other experiments to demonstrate the benefits of the two proposed extensions to the basic PRM scheme. We simulated some specific pathological network conditions that led to highly correlated losses between large groups of members; here use of the loss correlation technique improved data delivery rates by upto 12%.

EGF is beneficial under high frequency of group changes. For the 10 group changes per second experiment with 512 overlay nodes, EGF can improve the data delivery ratio from about 80% (see Figure 7) to 93%. Note that under these circumstances 512 changes (i.e. same as the size of the group) to the group happen in less than a minute. However, it also increases duplicate packets on the topology by nearly 10%.

5. RELATED WORK

A large number of research proposals have addressed reliable delivery for multicast data, most notably in the context of network-layer multicast. A comparative survey of these protocols is given in [13] and [24]. In SRM [7] receivers send NAKs to the source to indicate missing data packets. Each such NAK is multicast to the entire group and is used to suppress NAKs from other receivers that did not get the same packet. In this approach, however, a few receivers behind a lossy link can incur a high NAK overhead on the entire multicast group.

Tree-based protocols provide another alternative solution for reliable and resilient multicast. In this approach the receivers are organized into an acknowledgment tree structure with the source as the root. This structure is scalable because the acknowledgments are aggregated along the tree in a bottom-up fashion and also allows local recovery and repair of data losses. Protocols like RMTP [20], TMTP [26], STORM [22], LVMR [15] and Lorax [14] construct this structure using TTL-scoped network-layer multicast as a primitive. In contrast, LMS [19] uses an additional mechanism, called directed subcast, to construct its data recovery structure. Our work differs from of all these above approaches in two key aspects. First, unlike all these protocols that employ network-layer multicast service for data distribution our scheme is based upon an application-layer multicast delivery service. To the best of our knowledge the PRM scheme is the first application-layer multicast based scheme that addresses resilience. Second, all the network-layer multicast based schemes described employ completely reactive mechanisms for providing data reliability and therefore incurs moderate or high delivery latencies. As we show in this paper, proactive mechanisms, e.g. randomized forwarding, can be used to significantly improve resilience for applications that require low latency data delivery.

PRM is not the only proactive approach to provide improved reliability performance for multicast data. There exists some well-known forward error correcting code based approaches that are also proactive in nature. For example, Huitema [11] had proposed the use of packet level FECs for reliable multicast. Nonnenmacher et. al. [18] studied

Scheme	Data delivery	Recovery mechanism	Overheads	Recovery latency
SRM [7]	Network multicast	Reactive NAKs with global scope	High (for high network losses)	High
STORM [22] Lorax [14]	Network multicast	Reactive NAKs on ack tree	Low	Moderate
LMS [19]	Network multicast and directed subcast	Reactive NAKs on ack-tree	Low	Moderate
RMTP [20] LVMR [15]	Network multicast	Reactive/periodic ACKs with local scope	Low	Moderate
TMTP [26]	Network multicast	Reactive NAKs and periodic ACKs with local scope	Low	Moderate
Parity-based [18] (APES [23])	Network multicast (and directed subcast)	Reactive NAKs and FEC-based repairs	Moderate	Moderate
FEC-based [11, 18, 3, 16]	Network multicast or App-layer multicast [2]	Proactive FECs	High	Low
PRM	App-layer multicast	Proactive randomized forwarding and reactive NAKs	Low	Low

Table 3: Comparison of different reliability/resilience mechanisms for multicast data.

and demonstrated that additional benefits can be achieved when an FEC-based technique is combined with automatic retransmission requests. APES uses a related approach for data recovery [23]. Digital Fountain [3] and RPB/RBS [16] are two other efficient FEC-based approaches that provide significantly improved performance. All these FEC based approaches can recover from network losses. However, they alone are not sufficient for resilient multicast data delivery when overlays are used. Overlay nodes are processes on regular end-hosts and are more prone to failures than network routers. FEC-based approaches are not sufficient to recover from losses due to temporary losses on the data path, especially when low-latency delivery is required. The PRM scheme differs from all these other schemes by providing a proactive component that allows the receivers to recover from losses due to overlay node failures. In Table 3 we summarize the characteristics of all these schemes.

A different randomized error recovery scheme called RRMP is proposed in [25]. RRMP differs from PRM in that nodes detecting losses send randomized repair request (reactively) while PRM uses proactive randomized forwarding. Gossiping [9] and PRM both use randomized forwarding. However, randomized forwarding in PRM only provides additional resilience to data delivery along multicast tree, while gossiping only uses communication with random peers. Also related is dispersity routing scheme [17], where additional messages are sent using different routes. The difference is that PRM forwards data packets through randomized routes instead of a fixed set of paths between source and destination as in dispersity routing.

6. CONCLUSIONS

In this paper, we have shown how relatively simple mechanisms can be used to provide highly resilient data delivery over application-level multicast distribution trees. We have identified randomized forwarding as a key mechanism to mask data delivery failures due to failed overlay nodes, and have shown how even a very low overhead randomized forwarding is sufficient for handling rapid and massive changes to the distribution group. Our results are especially interesting since previously studied error-recovery techniques, such as FEC, alone do not provide adequate data recovery especially when overlay node failure rates are high. We have analytically shown why a randomized forwarding approach is able to achieve high data delivery ratios with low latencies.

Since PRM achieves very high data delivery ratios, we believe that it can easily be extended to provide perfect reliability as well. Even a naive extension in which each group member unicasts NAKs back to the source is likely to be sufficient. This is an area for future work.

Our detailed packet-level simulations show that the mechanisms described in this paper are immediately useful in realistic scenarios involving streaming media applications. The low bandwidth, storage and processing overheads of PRM makes it attractive for both low and high bandwidth streaming applications. Further, the very high recovery ratios —often in excess of 97% under adverse conditions— will allow PRM to be used with more efficient media encodings (which are usually less tolerant of data loss). We believe the techniques we have described will become standard and essential components of streaming media applications implemented over application-layer overlays.

Acknowledgements. We thank the SIGMETRICS 2003 referees for their helpful comments.

7. REFERENCES

[1] K. Azuma. Weighted sums of certain dependent random variables. *Tohoku Mathematical Journal*, 19:357–367, 1967.

[2] Although FEC-based schemes can be implemented over application-layer multicast, as this paper shows, it alone is not sufficient to achieve high delivery ratios even under moderate frequency of membership changes on the overlay.

[2] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *Proc. ACM Sigcomm*, Aug. 2002.

[3] J. Byers, M. Luby, and M. Mitzenmacher. A digital fountain approach to asynchronous reliable multicast. *IEEE Journal on Selected Areas in Communications*, 20(8), Oct. 2002.

[4] K. Calvert, E. Zegura, and S. Bhattacharjee. How to Model an Internetwork. In *Proc. of Infocom*, 1996.

[5] M. Castro, P. Druschel, A.-M. Kermarrec, and A. Rowstron. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in communications*, 20(8), Oct. 2002. To appear.

[6] Y.-H. Chu, S. G. Rao, and H. Zhang. A Case for End System Multicast. In *Proc. ACM SIGMETRICS*, 2000.

[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), Dec. 1997.

[8] P. Francis. Yoid: Extending the Multicast Internet Architecture, 1999. White paper http://www.aciri.org/yoid/.

[9] S. M. Hedetniemi, T. Hedetniemi, and A. L. Liestman. A Survey of Gossiping and Broadcasting in Communication Networks. *NETWORKS*, 18, 1988.

[10] W. Hoeffding. Probability inequalities for sums of bounded random variables. *American Statistical Association Journal*, 58:13–30, 1963.

[11] C. Huitema. The case for packet level FEC. In *Proc. 5th International Workshop on Protocols for High Speed Networks*, Oct. 1996.

[12] J. Leibeherr and M. Nahas. Application-layer Multicast with Delaunay Triangulations. In *IEEE Globecom*, Nov. 2001.

[13] B. Levine and J. Garcia-Luna-Aceves. A comparison of reliable multicast protocols. *Multimedia Systems Journal*, 6(5), Aug. 1998.

[14] B. Levine, D. Lavo, and J. Garcia-Luna-Aceves. The case for concurrent reliable multicasting using shared ack trees. In *Proc. ACM Multimedia*, Nov. 1996.

[15] X. Li, S. Paul, P. Pancha, and M. Ammar. Layered video multicast with retransmissions (LVRM): Evaluation of error recovery schemes. In *Proc. NOSSDAV*, 1997.

[16] A. Mahanti, D. Eager, M. Vernon, and D. Sundaram-Stukel. Scalable on-demand media streaming with packet loss recovery. In *ACM Sigcomm*, Aug. 2001.

[17] N. F. Maxemchuk. Dispersity routing. In *Proc. ICC*, pages 41.10–41.13, 1975.

[18] J. Nonnenmacher, E. Biersack, and D. Towsley. Parity-based loss recovery for reliable multicast transmission. *IEEE/ACM Transactions on Networking*, 6(4), Aug. 1998.

[19] C. Papadopoulos, G. Parulkar, and G. Varghese. An error control scheme for large-scale multicast applications. In *Proc. Infocom*, 1998.

[20] S. Paul, K. Sabnani, J. Lin, and S. Bhattacharyya. Reliable multicast transport protocol (rmtp). *IEEE Journal on Selected Areas in Communications*, 15(3), Apr. 1997.

[21] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level multicast using content-addressable networks. In *Proc. of 3rd International Workshop on Networked Group Communication*, Nov. 2001.

[22] X. Rex Xu, A. Myers, H. Zhang, and R. Yavatkar. Resilient multicast support for continuous-media applications. In *Proceedings of NOSSDAV*, 1997.

[23] D. Rubenstein, S. Kasera, D. Towsley, and J. Kurose. Improving reliable multicast using active parity encoding services (APES). In *Proc. Infocom*, 1999.

[24] D. Towsley, J. Kurose, and S. Pingali. A comparison of sender-initiated and receiver-initiated reliable multicast protocols. *IEEE Journal on Selected Areas on Communication*, 15(3), Apr. 1997.

[25] Z. Xiao and K. Birman. A randomized error recovery algorithm for reliable multicast. In *Proc. Infocom*, 2001.

[26] R. Yavatkar, J. Griffioen, and M. Sudan. A reliable dissemination protocol for interactive collaborative applications. In *ACM Multimedia*, Nov. 1995.

[27] B. Zhang, S. Jamin, and L. Zhang. Host multicast: A framework for delivering multicast to end users. In *Proc. IEEE Infocom*, June 2002.

Appendix: Proof sketches (Theorems 3.1, 3.2)

We just give proof sketches, due to space constraints. We first define the following notation.

(i) $\mathcal{L}_i$: The set of nodes that belong to level i.

(ii) $\mathcal{A}_i$: The set of nodes in level i that have not failed. Let A_i denote the expected size of $\mathcal{A}_i$.

(iii) $\mathcal{X}_i$: The set of nodes in level i of the tree to which the data transmitted by their parents on the tree reach successfully. In Figure 13, the node A is able to successfully send data to its children C and E, and $\mathcal{X}_i = \{C, E, T\}$. Note that the node C has failed, but still $C \in \mathcal{X}_i$, since the data reaches C successfully from the parent. Similarly, in the figure $\mathcal{X}_{i+1} = \{N, P, Q\}$. Also, let X_i denote $E[|\mathcal{X}_i|]$.

(iv) $\mathcal{Y}_i$: The subset of $\mathcal{X}_i$ which consists of nodes that have not failed. Since the node C has failed, in Figure 13, $\mathcal{Y}_i = \{E, T\}$. Similarly, $\mathcal{Y}_{i+1} = \{N, P\}$. Let Y_i be $E[|\mathcal{Y}_i|]$.

(v) S_u: For any node u at level i, let S_u be the expected number of children of u that lie in $\mathcal{Y}_{i+1}$, conditional on u lying in $\mathcal{Y}_i$. Let $v_1, \ldots, v_t$ be the children of u. Let e_j denote the edge $\langle u, v_j \rangle$, $p(w)$ the probability that the node w has not failed, and $q(e)$ the probability of not having a packet loss on the overlay link e. Then, $S_u = \sum_{j=1}^{t} q(e_j) \cdot p(v_j)$.

(vi) $\mathcal{N}_{i,j}$: is defined recursively as shown in Figure 14. $\mathcal{N}_{i,0} = \mathcal{Y}_i$. $\mathcal{N}_{i,j}$ is defined as the set of nodes in level i that do not belong to $\mathcal{N}_{i,k}, \forall k < j$ and successfully receive the data from the nodes in $\mathcal{N}_{i,j-1}$ using a single random edge. Thus in Figure 14, $\mathcal{N}_{i,0} = \{A_0, A_1, A_2\}$, $\mathcal{N}_{i,1} = \{A_3, A_4, A_5\}$, and so on. Clearly, if $k \neq l, \mathcal{N}_{i,k} \cap \mathcal{N}_{i,l} = \emptyset$. Let $N_{i,j} = E[|\mathcal{N}_{i,j}|]$.

We first show that the node-failures are not much more than expected. Let $\exp(x)$ denote e^x from now on. Using the Hoeffding bound [10], we can show that

$$Pr[\exists i : \ |\mathcal{A}_i| \leq 0.5\gamma \cdot D_0 D_1 \cdots D_{i-1}] \leq \sum_{i=1}^{d} \exp(-\frac{\gamma}{8} \cdot 2^{i-1} D_0); \quad (1)$$

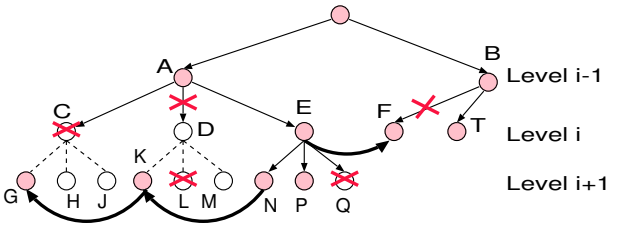

Figure 13: Simplified PRM on a tree. The random edges are restricted to occur between nodes on the same level. The crosses in the figure mark link losses or node failures.

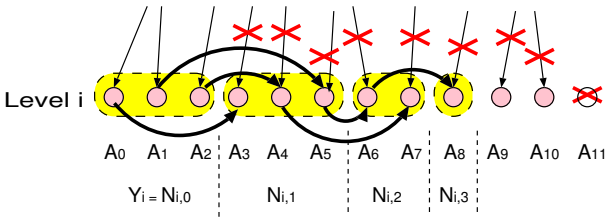

Figure 14: Defining $\mathcal{N}_{i,j}$. The crosses in the figure mark link losses or node failures.

the sum in the r.h.s. is smaller than, say, $\delta^2/3$ if D_0 is more than some constant times $\log(1/\delta)$.

We next show that all the sets $\mathcal{Y}_i$ are large enough, with high probability. By assumption (A2), there is a constant $\theta \in (0,1)$ such that $\lambda(1-\theta) > 1$; fix such a θ. Define the following events for $1 \le i \le d$: event $\mathcal{E}_i$ is that "$|\mathcal{Y}_i| \ge \prod_{j=0}^{i-1}(\alpha D_j(1-\theta))$". The Hoeffding bound can again be used to show that

$$\Pr[\exists i : \overline{\mathcal{E}}_i] \le \sum_{i=1}^{d-1} \exp(-(D_0\alpha\theta^2/2)\cdot(\lambda(1-\theta))^{i-1}), \quad (2)$$

where $\overline{\mathcal{E}}_i$ denotes the complement of $\mathcal{E}_i$. Since $\lambda(1-\theta) > 1$, the sum in the r.h.s. is smaller than $\delta^2/3$ if D_0 is more than some constant times $\log(1/\delta)$.

Now that we have shown that all the $\mathcal{Y}_i$ are large enough with high probability, we are ready to show that the randomized forwarding reaches all the surviving nodes that did not receive the data from their parent – and does so fast – with high probability. To do this in a way that captures the scenarios of Theorems 3.1 and 3.2, we consider the following general setting. We have a (large) set $\mathcal{U}$ of nodes, and a subset $\mathcal{G} \subseteq \mathcal{U}$. Letting $U \doteq |\mathcal{U}|$, we also have $G \doteq |\mathcal{G}| = (1-\epsilon)U$, for some constant $\epsilon \in (0,1)$. Here, $\mathcal{U}$ represents some total set of nodes, and $\mathcal{G}$ (the "good" nodes) represents those that have not failed. We also have some $\mathcal{U}_0 \subseteq G$ such that $|\mathcal{U}_0| \ge U\phi$ for some parameter $\phi \in (0,1)$; this parameter can be quite small. $\mathcal{U}_0$ corresponds to the nodes that have received the data from their parent; we aim to see how the random forwarding initiated by the nodes in $\mathcal{U}_0$, "expands" to cover almost all of $\mathcal{G}$. Recall that every node that receives the data, chooses a random set of r other nodes and forwards data to each of them independently with probability β. Analogous to the sets $\mathcal{N}_{i,j}$, we define sets $\mathcal{U}_1 \subseteq \mathcal{G}$, $\mathcal{U}_2 \subseteq \mathcal{G}, \ldots$ as follows: $\mathcal{U}_t$ is the set of nodes in $\mathcal{G}$ which do not belong to $\mathcal{U}_k, \forall k < t$ and which successfully receive the data from some node in $\mathcal{U}_{t-1}$ using a single random edge. Let $\mathcal{M}_t$ denote $\mathcal{U}_0 \cup \mathcal{U}_1 \cup \cdots \cup \mathcal{U}_t$; define $U_t = |\mathcal{U}_t|$ and $M_t = |\mathcal{M}_t|$. Now, our main goal is to show that with high probability (say, at least $1-\delta^2/3$), there is a value $t = O(\log(1/\phi) + \log(1/\delta))$ such that $M_t \ge (1-\delta)G$; i.e., that in $O(\log(1/\phi) + \log(1/\delta))$ steps of random forwarding, essentially all the surviving nodes get the data. We only provide the essence of the proof here.

The dynamics of the evolution of the U_t is as follows: $E[U_{t+1} \mid (M_t = a,\ U_t = b)]$ equals $(U(1-\epsilon) - b)$ times

$$(1 - (1 - \frac{r\beta}{U-1})^a) \ge (1 - \exp(-ar\beta/U)); \quad (3)$$

the initial condition is the deterministic one that $M_0 = U_0$. By Azuma's martingale inequality [1], we are able to show that if U is large enough, then with high probability, say $1 - 1/U^3$, all U_t are at least $(1 - O(\sqrt{(\log U)/U}))$ times their mean; thus, we are able to treat the U_t as *deterministic*, given by (3). The evolution of the U_i can be divided into two epochs: when $M_t \le U(1-\epsilon)/2$ and when $M_t > U(1-\epsilon)/2$. In the first epoch, the U_t values increase geometrically, and in $O(\log(U/U_0)) = O(\log(1/\phi))$ steps, we get $M_t \ge U(1-\epsilon)/2$ (with high probability). However, now that M_t is quite large, many of the randomly forwarded edges are likely to fall in M_t: i.e., progress slows down. In particular, suppose we have almost reached our goal of covering $U(1-\epsilon)\cdot(1-\delta)$ nodes; at this point, we have $U(1-\epsilon) - M_t \approx U(1-\epsilon)\delta$. Since the U_t closely track their means, we only get something like

$$U_{t+1}/U \ge (1-\epsilon)\delta \cdot (1 - \exp(-r\beta\cdot(U_t/U)). \quad (4)$$

Thus, if U_t/U becomes small, the successive values U_{t+i}/U can get much smaller, very fast. This is where we take $r\beta$ large enough. Suppose we take $r\beta \ge 4\ln 2/(\delta(1-\epsilon)) = O(1/\delta)$; let $x_0 = \delta(1-\epsilon)/4$. It can be shown that for all $x \ge x_0$, $(1-\epsilon)\delta\cdot(1-\exp(-r\beta x)) \ge 2x_0$. Using this, we can show that with high probability, U_t/U always remains above x_0. We subdivide the second epoch into phases: there is one phase for each integer $0 \le j \le O(\log(1/\delta))$, which captures the time period when we have $U(1-\epsilon)\cdot(1-2^{j+1}\delta) \le M_t < U(1-\epsilon)\cdot(1-2^j\delta)$. By analyzing the phases using the idea that $U_t/U \ge x_0$ holds always with high probability, we can show that the second epoch will be completed within $O(\log(1/\delta))$ steps with high probability.

Taking $\mathcal{U}$ to be the set of nodes at any given level i yields Theorem 3.1; Theorem 3.2 follows by taking $\mathcal{U}$ to be the set of all nodes in the tree.

On the necessity of our assumptions: Our assumption in (A1) that each $D_i \ge 2$ is not strictly required: it is sufficient if this condition holds, say, at least once every constant number of steps as we go down the tree. In (A2), a condition such as "$\lambda > 1$" is necessary: if $E[S_u] < 1$ for many nodes u, then with high probability, the data will stop at some finite level (before reaching the leaf level). Condition (A3) is necessary for meaningful operation: in practice, one will require $\gamma \ge 0.9$, say. It is more involved to prove that $r\beta = \Omega(1/\delta)$ is necessary. We just mention here that we can show the following: there is a constant $c > 0$ such that if $r\beta \le c/\delta$, then with high probability, we will cover at most $N(1-\epsilon)\cdot(1-2\delta)$ nodes, falling short of our goal.