A Comparison of Server–Based and Receiver–Based Local Recovery Approaches for Scalable Reliable Multicast*

Sneha Kumar Kasera, Jim Kurose and Don Towsley

CMPSCI Technical Report TR 97-69
December 1997
(Revised October 1998)
An earlier version appeared in IEEE INFOCOM 1998

Abstract

Local recovery approaches for reliable multicast have the potential to provide significant performance gains in terms of reduced bandwidth and delay, and higher system throughput. In this report we examine one server–based and two receiver–based approaches, and compare their performance. The server–based approach makes use of specially designated hosts, called repair servers, co–located with routers inside the network. In the receiver-based approach, only the end hosts (sender and receivers) are involved in error recovery. Using analytical models, we first show that the three local recovery approaches yield significantly higher protocol throughput and lower bandwidth usage than an approach that does not use local recovery. Next, we demonstrate that server–based local recovery yields higher protocol throughput and lower bandwidth usage than receiver–based local recovery when the repair servers have processing power slightly higher than that of a receiver and a few hundred kilobytes of buffer per multicast session.

---

*This work was supported by ARPA under contracts F19628-95-C-0146 and N6601-97-C-8513.

1 Introduction

Reliable multicast is required by many applications such as multicast file transfer, shared whiteboard, distributed interactive simulation and distributed computing. These applications can potentially have several thousands of participants scattered over a wide area network. Even though current multicast networks, such as IP multicast networks, provide efficient routing and delivery of packets to groups of receivers based on multicast group addresses, they are lossy and do not provide the reliability needed by the above applications. Designing scalable approaches and architectures for reliable multicast, that make efficient use of both the network and end–host resources, is a challenging task.

Local recovery approaches for reliable multicast, in which a network entity other than the sender aids in error recovery, have the potential to provide significant performance gains in terms of reduced bandwidth and delay, and higher system throughput. In this report, we examine and compare three different local recovery approaches, one requiring additional processing and caching inside the network and the other two based only on end–to–end means. The first approach, termed *active* server–based local recovery, makes use of specially designated hosts, called *repair servers*, co–located with routers at strategic points inside the network. Each repair server serves the retransmission requests of receivers in its local area by (re)transmitting local repairs. The repair servers themselves recover lost packets from either higher level repair servers or the sender. Co–locating repair servers with routers is functionally equivalent to placing repair services at the routers. Therefore, it is also possible to use *active networking* mechanisms [17] to dynamically place repair services at strategic points inside the network.

In the receiver–based approaches, there are no special repair servers; only the end hosts (sender and receivers) are involved in error recovery. The first receiver–based approach is *self organizing.* Whenever local recovery of a lost packet is attempted, it dynamically selects a receiver within a local geographical area to supply the repair. The second receiver–based approach, called the *designated receiver* approach, assigns an end–receiver the task of supplying repairs in its local area.

We analyze the protocol throughput and the network bandwidth usage of the above local recovery approaches in the face of spatially correlated loss. We first show that local recovery approaches yield significantly higher protocol throughput and lower bandwidth usage than an approach that does not use local recovery. Next we compare the performance of the local recovery approaches. Under the assumption that repair servers have sufficiently high processing and buffering capabilities so that they can never be bottlenecks, we show that active server–based recovery yields higher protocol throughput and lower bandwidth usage than receiver–based recovery. Among the two receiver–based approaches the self organizing one exhibits higher protocol throughput but uses more bandwidth.

Last, we estimate the processing power and buffering needed at repair servers so that they do not become bottlenecks in the repair server approach. We find that a repair server needs a few hundred kilobytes of buffer space and processing power slightly higher than that of a receiver, per multicast session, for reasonable loss probabilities.

The remainder of this report is structured as follows. In the next section we examine the existing work on local recovery for reliable multicast. In Section 3 we present the three local recovery approaches and our system model in Section 4. Section 5 contains the throughput and bandwidth analysis. We compare the performance of the active server–based and the receiver–based approaches in Section 6. In Section 7 we size the repair servers. Conclusions and directions for future work are contained in Section 8.

2 Related Work

Many researchers have recently proposed different approaches for local recovery. Some of these approaches are scalable reliable multicast, SRM (with local recovery enhancements) [5, 6], local group concept, LGC [7], tree–based multicast transport protocol, TMTP [21], LORAX [11], log–based receiver reliable multicast, LBRM [8] and reliable multicast transport protocol, RMTP [12]. These approaches can be broadly classified as follows. LBRM is server–based, SRM with local recovery enhancements is self organizing receiver–based and RMTP, TMTP and LORAX are designated receiver–based with pre-constructed logical hierarchy. LGC is a hybrid of the logical tree–based approach and SRM. We add a new dimension to the earlier work on server-based recovery [8] by considering the placement of repair services inside the network.

Our work is the first to analytically compare the throughput and bandwidth usage of server–based and receiver–based local recovery approaches. Previous throughput analyses have focused on comparing ACK-based versus negative acknowledgment (NAK)-based protocols that do not use local recovery [15, 18, 9] or comparison of logical tree–based local recovery with other non–local recovery approaches [10, 11]. In our work, in addition to throughput, we also analyze the network bandwidth usage of the local recovery protocols. We also estimate the processing and buffering requirements of repair servers. Also all of the previous work mentioned above assumes spatial independence among losses. We use a system model that considers the spatial correlation [20] present in network losses.

Recently, Papadopoulos *et al* [14] have suggested error recovery with the help of modifications at the routers. One concern with this approach is that it requires modification in the “fast path” of routers for efficient operation, in addition to other changes. Our active server–based approach does not require any changes in the fast path of routers.

3 Protocols

We now present a server–based and two receiver–based approaches to local recovery for reliable transmission of data from a sender to multiple receivers. All approaches are receiver–initiated NAK-based approaches. The server–based approach places much of the burden for ensuring error recovery on repair servers placed inside the network. In the self organizing receiver–based approach, the error recovery burden is shared among the sender and the participating receivers. In the designated receiver approach, a designated receiver is assigned the task of supplying repairs in its local area. In this section we assume that mechanisms are available for scoping multicasts within a "local" region. We will list some of the mechanisms for accomplishing this at the end of the Section 3.2.

3.1 A Server–Based Local Recovery Protocol

We now describe a generic server–based reliable multicast local recovery protocol, L1, that assumes the presence of a *repair tree*. This repair tree is the physical multicast routing tree constructed by the routing protocols with the sender as the root, receivers as leaves and repair servers co–located with routers. The receivers recover lost packets from repair servers and repair servers recover lost packets from either upper level repair servers or the sender. Protocol L1 exhibits the following behavior:

- The sender multicasts packets to a multicast address (say A) that is subscribed by all receivers and repair servers.
- On detecting a loss, a receiver, after waiting for a random amount of time, multicasts[1] a NAK to the receivers in its neighborhood and the repair server (for the purpose of NAK suppression among receivers in the neighborhood) and starts a NAK retransmission timer. If the receiver receives a NAK from another receiver for this packet, it suppresses its own NAK and sets the NAK retransmission timer as if it had sent the NAK.
- On detecting a loss, a repair server, after waiting for a random amount of time, multicasts a NAK to the sender (or its upstream repair server) and the other repair servers at the same level in the tree (for the purpose of NAK suppression among repair servers) and starts a NAK retransmission timer. While waiting to send out a NAK for the lost packet, if the repair server receives a NAK for the same packet from another repair server, then it suppresses its own NAK and sets the NAK retransmission timer as if it had sent the NAK.
- If a repair server has the packet for which it received a NAK from a member of its group (of receivers) for which it is responsible, it multicasts the packet to the group. If the repair server does not have the packet, it starts the process of obtaining the packet from the sender (or its upstream repair server) if it has not already done so (as described above).
- The sender, on receiving a NAK from the repair servers, remulticasts the requested packet to all the receivers and repair servers.
- The expiration of the NAK retransmission timer at a repair server (or a receiver) without prior reception of the corresponding packet serves as the detection of a lost packet for the repair server (or the receiver) and a NAK is retransmitted.

We end this section by distinguishing between a log service and a repair service. A log service, as mentioned in [8], provides secondary storage for packets transmitted from the sender. The entire

---

[1] Assume that this multicast address, different from A, is known to the receiver.

data is stored at the log servers for repairs and "late–comers." Log service should be distinguished from the repair service, particularly if the repair servers are considered network resources that will be shared over several multicast sessions. Logging entire multicast sessions at these repair servers may not scale with the number of multicast sessions because of the storage requirement. In our work, we consider storing only the "recent" data at the repair servers for the sole purpose of providing quick repairs.

3.2 Receiver–Based Local Recovery Protocols

We now describe the two receiver–based local recovery protocols. We first describe the self organizing receiver–based protocol for local recovery. This protocol, termed L2, is a generic version of SRM [5] with local recovery enhancements. In L2, loss recovery is performed at two levels. At the first level, a receiver tries to recover packets locally from only within its local neighborhood. We define[2], a receiver's local neighborhood to consist of all the receivers in the subtree (of the multicast routing tree) emanating from its nearest backbone router at the edge of the backbone. If the receiver is unsuccessful in recovering the packet locally, it tries to recover packets from the sender ("global recovery"). Each receiver alternates between local and global recovery until it receives the missing packet. Protocol L2 exhibits the following behavior

- the sender multicasts all packets on address (A) which is subscribed by all of the participating receivers.
- on detecting a loss, a receiver waits a random amount of time and multicasts a local NAK addressed only to the receivers in its neighborhood and starts a local NAK retransmission timer (first–level recovery); if the receiver does not receive the lost packet before the local NAK retransmission timer expires, it waits for a random amount of time and multicasts a global NAK to address A and starts a global NAK retransmission timer (global recovery). The random waiting times and the NAK retransmission timeout values are different for the two levels of recovery. The first level retransmission timeout will be smaller. If the receiver does not recover the lost packet before the global NAK retransmission timer expires, it starts the local recovery again.
- on receiving a local NAK for which it has the associated packet, a receiver multicasts the packet only to its neighborhood; before sending this repair packet, a receiver waits for a certain random amount of time and suppresses its own transmission if another receiver sends out the repair packet in this time. On receiving a global NAK for a packet, a receiver suppresses its own NAK if it is planning to send one for the same packet.
- upon receiving a global NAK, the sender multicasts the packet to A.

The above protocol attempts to ensure that only one NAK and only one repair are generated at either recovery level. Unlike SRM [5], L2 generates *global repairs only from the sender* thereby

---

[2] This definition and other definitions of local neighborhood can be found in [6].

avoiding the risk of global repair explosion. In SRM, receivers and the sender use a random back-off strategy to generate a repair. Here all receivers and the sender " listen" for a repair. If they do not receive the repair within a certain time then they generate a repair if they have it. Note that a repair has to be processed at each node whether it is sent or received. Hence the overhead of processing a global repair under SRM and L2 will be comparable. For local recovery, no particular host is likely to have the repairs and hence, as in SRM, L2 uses a random back-off strategy for generating local repairs. Protocol L2 has been described above in the case of two-level recovery. It could be easily generalized to multiple levels of recovery.

The designated receiver local recovery protocol, termed L3, is similar to protocol H2 described in [11] with some minor differences. The repair tree in L3 (and H2) is a logical tree [10, 11] constructed with end hosts (the sender and the receivers). This logical repair tree is different from the physical multicast routing tree. Instead of special repair server hosts, each non-leaf receiver is assigned the task of providing repair service to its children. In protocol L3, the sender multicasts the original transmissions of a packet to all the receivers but unlike L1, it multicasts retransmissions only to its children in the logical tree. This could be done by using a separate multicast group to which all of the designated receivers and the sender belong. The NAKs from designated receivers are also sent on this group for the purpose of NAK suppression. Each designated receiver recovers lost packets from its parent and supplies repairs to its children in the logical tree. In L3 (unlike H2) a node does not send any periodic acknowledgments to its parent.

Most of the known local recovery mechanisms for supporting server or receiver–based local recovery architectures can be attributed to [12] and [6]. For the purpose of scoping transmissions in a local area one of the the following mechanisms from [6], (*i*) TTL (time to live)–based local recovery, (*ii*) administrative scoping, (*iii*) separate multicast groups, can be used.

The subtree multicast technique, that uses IP encapsulation (see [12]), can be used to direct repairs from a repair server to only the downstream receivers. For informing the receivers about their whereabouts (for example, the IP multicast addresses needed to reach the repair servers) the repair servers can send out slow rate advertisements. This helps in handling new repair server assignments due to routing changes.

4 System Model

We now present the network loss and system model that will be used for the performance analysis of protocols L1, L2 and L3. In [20], Yajnik *et al* have observed that most of the losses take place in the links from the source site to the backbone (we call this the source link) and in the links (called "tail links") from the backbone to the individual sites (termed stub domains in [3]), as shown in Figure 1. The backbone and the individual sites have been observed to be mostly loss free. It has also been noted in [8] that tail links are likely to be bottlenecks for the foreseeable future. Hence in our loss model we consider loss only at the source and tail links. Losses at the source link will be seen by all the receivers whereas losses at a tail link will be seen only by the receivers attached to

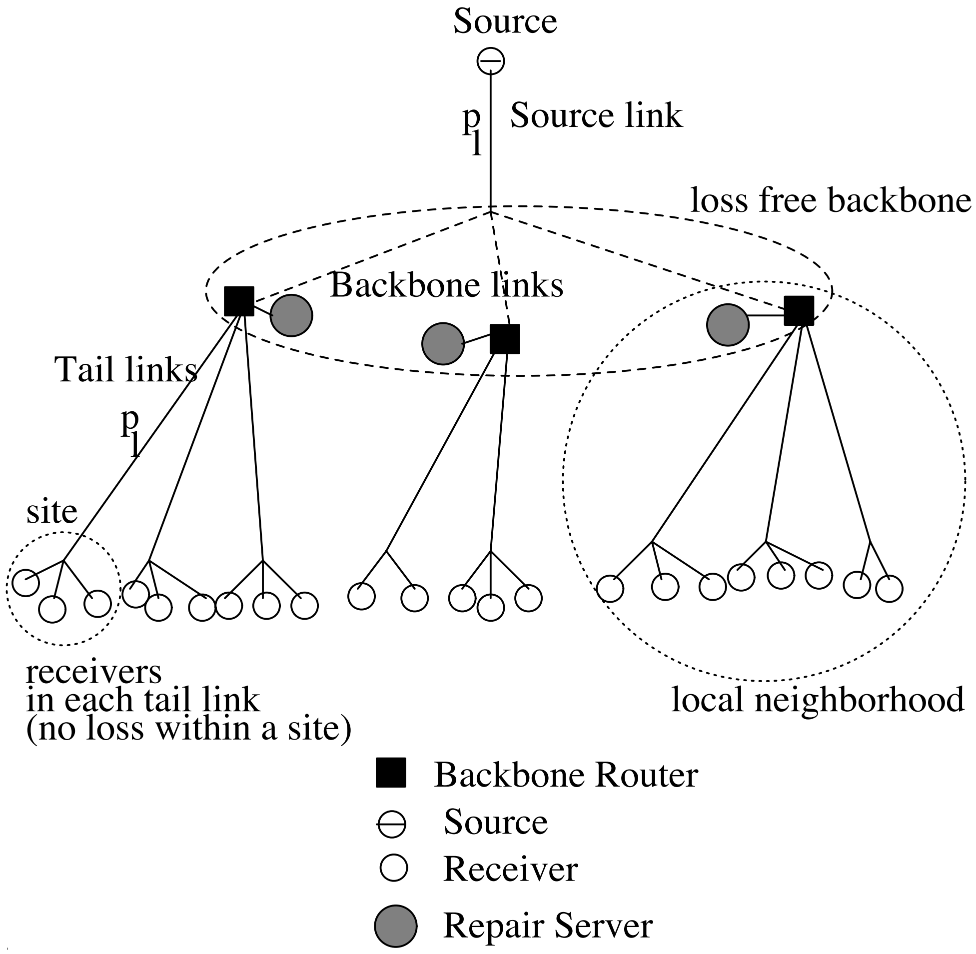

Figure 1: System Model

it. Losses at tail links are assumed to be mutually independent. We also assume that loss events are temporally independent. Let the loss probability in the source link and each tail link be p_l. The end–to–end loss probability as seen by a receiver, p, is equal to $1 - (1 - p_l)^2$. We also assume that NAKs are never lost. This assumption could be relaxed by following the analysis given in [19].

We assume that well–defined local neighborhoods exist and that the mechanisms described at the end of the previous section could be used to address packets to specific neighborhoods. Let there be one source link and T tail links where each tail link has multiple receivers. Let each upstream backbone router, at the edge of the backbone, have k tail links. The number of such backbone routers, W is equal to T/k. We assume that communication between two receivers on different tail links takes place only through a backbone router(s). Direct site-site links (termed stub-stub edges in [3]) are likely to be very rare.

The above loss model has implications on the placement of repair servers inside the network. We argue that the repair servers should be placed with the routers at the edge of the backbone. To be able to take advantage of local recovery, a repair server should be both upstream of and as close to the point of loss as possible. Figure 1 shows the placement of repair servers.

We end this section by noting that for our system model two–level recovery is sufficient for protocol L1. In analyzing L3, we assume that there is a two level hierarchy where one receiver in each neighborhood is assigned the task of supplying repairs within the neighborhood. These designated receivers recover lost packets directly from the sender.

5 Performance Analysis

In this section we analyze the end system processing and network bandwidth requirements of protocols L1, L2 and L3.

Following the approach first suggested in [15], the receive processing cost (time) is determined by computing the total amount of processing involved in *correctly* receiving a randomly chosen packet. This includes the time required to receive those copies of this packet (i.e., the original copy plus any retransmissions) that arrive at the receiver, the time required to send/receive any NAKs associated with this packet and the time needed to handle any timer interrupts associated with this packet. The send processing cost is determined by the total sender processing involved in correctly transmitting a packet correctly to all the receivers. This includes the cost required to process any received NAKs, and process retransmissions that are sent out in response to these NAKs.

Unlike other analyses (e.g. [9, 10, 11, 15, 18]) we also analyze the *bandwidth* requirements of the various protocols. For analyzing bandwidth we focus on the following two bandwidth metrics.

1. *Total bandwidth usage*: the total number of bytes sent per successful packet transmission along all of the links to all the receivers[3].

---

[3] More generally we could have a three dimensional table with the dimensions being type of packet (data-packet

X_p	–	the time to process the transmission of a packet at a sender
X_n	–	the time to process a NAK at a sender
Y_p	–	the time to process a newly received packet at a receiver
Y_t	–	the time to process a timeout at a receiver
Y_n	–	the time to process and transmit/receive a NAK at a receiver
p	–	the end–to–end loss probability at a receiver
p_l	–	the source link or tail link loss probability
T	–	the total no. of tail links
k	–	number of tail links per backbone router/repair server
X^ω,Y^ω	–	the send and receive per packet processing time in protocol $\omega \in$ {L1, L2, L3}

Table 1: Notation

2. *Traffic concentration*: the ratio of average bandwidth usage of the link with the highest average bandwidth usage to the average bandwidth usage over all links in the network. This measure has been used to compare the performances of routing protocols[2, 4].

5.1 Processing Cost Analysis

We now derive expressions for sender and receiver processing requirements for protocols L1, L2 and L3. Table 1 describes the notation used in the analysis. Some of the notation has been reintroduced from [18, 9]. We assume that processing times have general distributions and that they are independent of each other.

5.1.1 Analysis of L1

Notation specific to L1 is described in Table 2. Following the approach described in [18, 9], the mean per packet processing time for a randomly chosen packet at the sender for protocol L1 can be expressed as,

$$E[X^{L1}] = E[M_s]E[X_p] + (E[M_s] - 1)E[X_n] \tag{5.1}$$

The first term corresponds to the mean processing required to transmit a packet. The second term corresponds to the mean NAK processing required at the sender.

The mean processing requirement at a receiver for a randomly chosen packet is

$$E[Y^{L1}] = E[M_R](1 - p_l)E[Y_p] + (E[M_R] - 1)E[Y_n] + E[(M_r - 2)^+]E[Y_t] \tag{5.2}$$

---

or NAK), type of link (source link, tail link or backbone link) and direction on the link (upstream or downstream), to determine the bandwidth associated with the transmission of a packet over a certain link in a certain direction. Then we could sum over the bandwidth used over all links.

M_s – number of transmissions from the sender required for all repair servers to successfully receive a packet
M_r – number of transmissions from a repair server required for a receiver to successfully receive a packet
M_R – number of transmissions from a repair server to all the recvrs for which it is responsible successfully recv a pkt - assuming that no pkts are sent from the sender
Z^{L1} – the per packet processing time at the repair server

Table 2: Notation specific to L1

The first term corresponds to the processing required to correctly receive a packet. The second term corresponds to the mean processing time required to send and receive NAKs[4]. The third term corresponds to the processing required to execute routines due to expiration of NAK retransmission timer. Here, $(x)^+ = \max\{0, x\}$. The mean processing time at a repair server for a randomly chose packet is

$$\begin{aligned} E[Z^{L1}] &= E[M_s](1-p_l)E[Y_p] + (E[M_R]-1)E[X_p] + (E[M_s]-1)E[Y_n] \\ &\quad + (E[M_R]-1)E[Y_n] + E[(M_s-2)^+]E[Y_t] \end{aligned} \tag{5.3}$$

The first two terms correspond to the processing of transmissions and retransmissions, received from the sender and the retransmissions sent out to the receivers respectively. The third term corresponds to processing of NAKs sent to the sender or received from other repair servers. The fourth term corresponds to the processing of NAKs received from the receivers and the fifth term represents the processing cost of the NAK retransmission timeout routine. The terms $E[M_s]$, $E[M_R]$, $E[M_r]$ and $E[(M_r-2)^+]$ are computed as follows:

$P[M_s \leq m] = 1 - p_l^m, m = 1, \ldots$ and $E[M_s] = 1/(1-p_l)$. We know that for any random variable X, such that $X = 1, 2, 3, \ldots$,

$$E[X] = 1 + \sum_{m=1}^{\infty} (1 - P(X \leq m)), \tag{5.4}$$

Now $P(M_R \leq m) = (1 - p_l^m)^k$, hence

$$E[M_R] = 1 + \sum_{m=1}^{\infty} (1 - (1 - p_l^m)^k)$$

Now $P(M_r \leq m) = 1 - p_l^m$, hence $E[M_r] - 1 = p_l/(1-p_l)$ and $E[(M_r-2)^+] = p_l^2/(1-p_l)$.

---

[4] In counting the number of NAKs and the number of timeouts there is an implicit assumption that the lost packet is available at the repair server. For this assumption to hold good a receiver has to cleverly choose NAK retransmission timeout values. Otherwise, the number of NAKs will be more than what we have counted.

M – number of transmissions from the sender for all receivers to receive the packet correctly if there were no local recovery

M_n – number of transmissions from the sender for all receivers in a local neighborhood to receive the packet correctly if there were no local recovery

$M^{'}$ – number of transmissions from the sender to ensure that at least one receiver in each neighborhood receives the packet correctly

$M^{'}_n$ – number of transmissions from the sender to ensure that at least one receiver in a neighborhood receives the packet correctly if there were no local recovery

M_t – number of transmissions (global or local) required for any receiver r to correctly receive a packet

Table 3: Notation specific to L2

5.1.2 Analysis of L2

We now analyze the sender and receiver processing costs of protocol L2. Notation specific to this analysis is described in Table 3. We assume that the required processing times required to send and receive a packet are the same (it has been observed in [9] that these are almost same). We also make the optimistic assumption that only one retransmission is generated per NAK during local recovery in L2. This assumption will result in lower receiver processing costs for L2. The mean processing time at the sender for a randomly chosen packet is

$$\begin{aligned} E[X^{L2}] &= (E[M^{'}] + E[\lfloor (M - M^{'})/2 \rfloor])E[X_p] \\ &\quad (E[M^{'}] + E[\lfloor (M - M^{'})/2 \rfloor] - 1)E[X_n] \end{aligned} \tag{5.5}$$

The first term corresponds to the processing required for transmissions and retransmissions of the packet. This term is made up of two parts. The first part is the expected processing cost of (re)transmissions of the packet so that at least one receiver in every neighborhood receives the packet. The second part is the expected processing cost of additional retransmissions from the sender when receivers cannot locally recover the lost packet even though it is available in the local neighborhood. Recall that a receiver alternates between local and global recovery in that order. Therefore, $\lfloor (M - M^{'})/2 \rfloor$ is the number of additional transmissions required from the sender to ensure that all the receivers correctly receive the packet. The second term corresponds to the processing of global NAKs received by the sender. This number of global NAKs is equal to one less than the number of (re)transmissions from the sender because only one NAK is generated per loss due to NAK suppression[5].

---

[5] Here the assumption is that NAK suppression works perfectly.

The mean processing time at the receiver for a randomly chosen packet is

$$\begin{aligned} E[Y^{L2}] \;=\;& (E[M'] + E[\lfloor (M - M')/2 \rfloor])(1-p_l)^2 E[Y_p] \\ &+ (E[\lceil (M_n - M_n')/2 \rceil])(1/k + (k-1)(1-p_l^2)/k)E[Y_p] \\ &+ (E[M'] + E[\lfloor (M - M')/2 \rfloor] - 1)E[Y_n] \\ &+ (E[M_n'] + E[\lceil (M_n - M_n')/2 \rceil] - 1)E[Y_n] \\ &+ (E[(M_t - 2)^+] + E[(M_n' - 2)^+])E[Y_t] \end{aligned} \tag{5.6}$$

The first two terms correspond to the processing of global and local transmissions at the receiver. The number of local retransmissions is determined by considering a local neighborhood in isolation and determining the number of transmissions needed from the sender to ensure that all the receivers in the neighborhood receive the packet correctly. Then the number of transmissions required to ensure that at least one receiver in the neighborhood receives the packet is found. Half of the difference in these quantities yields the number of local retransmissions. We take the ceiling of this number to account for the fact that a recovery sequence always begins with local recovery. A local transmission is generated within a tail with a probability $1/k$. A receiver receives a local transmission generated in another tail link in its neighborhood with probability $(k-1)(1-p_l)^2/k)$. Here we optimistically assume that only one local repair is generated per local NAK.

The third term in (5.6) corresponds to the mean processing time required to send and receive global NAKs. The fourth term is the mean processing time required for sending and receiving local NAKs. It is important to note here that m' local NAKs are generated even before the packet is available in the local neighborhood. The last term is the mean processing time required for timer routine executions. This consists of two parts. The first part is the expected processing cost of the timer routine executions when the packet is transmitted (globally or locally) and not received. The second part is the expected processing cost of the timer routine executions when local recovery attempts are being made in between global recovery attempts even before the packet is available in the local neighborhood. We determine the expressions for $E[M]$, $E[M']$, $E[M_n]$ and $E[M_n']$ as follows:

We use the recursive technique described in [1] to obtain

$$\begin{aligned} P[M \le i] &= \sum_{u=0}^{i-1} \binom{i}{u} p_l^u (1-p_l)^{i-u} (1 - p_l^{i-u})^T \\ P[M_n \le i] &= \sum_{u=0}^{i-1} \binom{i}{u} p_l^u (1-p_l)^{i-u} (1 - p_l^{i-u})^k \\ P[M' \le i] &= \sum_{u=0}^{i-1} \binom{i}{u} p_l^u (1-p_l)^{i-u} (1 - p_l^{(i-u)k})^W \\ P[M_n' \le i] &= \sum_{u=0}^{i-1} \binom{i}{u} p_l^u (1-p_l)^{i-u} (1 - p_l^{(i-u)k}) \end{aligned}$$

$E[M]$, $E[M']$, $E[M_n]$ and $E[M_n']$ can be determined using (5.4) and can be substituted into (5.5)

M_d – number of transmissions from the sender required for all designated receivers to successfully receive a packet

M_l – number of transmissions from a designated receiver to all the recvrs for which it is responsible successfully recv a pkt - assuming that no pkts are sent from the sender

Z^{L3} – the per packet processing time at the designated receiver

Table 4: Notation specific to L3

and (5.6). Now, $P(M_t \leq i) = 1 - p^i$. Hence,

$$E[(M_t - 2)^+] = \sum_{i=2}^{\infty}(1 - P(M_t \leq i)) = p^2/(1-p)$$

$$\begin{aligned} E[(M_n^{'} - 2)^+] &= \sum_{i=2}^{\infty}(1 - P(M_n^{'} \leq i)), \\ &= E[M_n^{'}] - 2 + P(M_n^{'} \leq 1), \\ &= E[M_n^{'}] - 2 + (1 - p_l)(1 - p_l^k) \end{aligned}$$

Further,

$$\begin{aligned} E[\lfloor (M - M^{'})/2 \rfloor] &= (E[M] - E[M^{'}])/2 - P(M - M^{'} \text{is odd})/2 \\ E[\lceil (M_n - M_n^{'})/2 \rceil] &= (E[M_n] - E[M_n^{'}])/2 + P(M_n - M_n^{'} \text{is odd})/2 \end{aligned}$$

Since random variables M and $M^{'}$ (and also M_n and $M_n^{'}$) are not independent it is not easy to obtain simple expressions for the loss probabilities $P(M - M^{'} \text{is odd})/2$ and $P(M_n - M_n^{'} \text{is odd})/2$. Noting that the values of these two probabilities will be significant only when the number of tails is small and loss probabilities are low, we bound $E[\lfloor (M - M^{'})/2 \rfloor]$ and $E[\lceil (M_n - M_n^{'})/2 \rceil]$ by

$$\begin{aligned} E[\lfloor (M - M^{'})/2 \rfloor] &> (E[M] - E[M^{'}])/2 - 1/2 \\ E[\lceil (M_n - M_n^{'})/2 \rceil] &> (E[M_n] - E[M_n^{'}])/2 \end{aligned}$$

5.1.3 Analysis of L3

We now analyze the sender, receiver and designated receiver processing costs of protocol L3. Notation specific to this analysis is described in Table 4. The mean processing requirement at the sender for a randomly chosen packet is

$$E[X^{L3}] = E[M_d]E[X_p] + (E[M_d] - 1)E[X_n] \tag{5.7}$$

The mean receive processing requirement at a leaf receiver that is not located at the same site as the designated receiver is

$$E[Y^{L3}] = E[M_l](1-p)E[Y_p] + (E[M_l - 1)E[Y_n] + p^2/(1-p)E[Y_t] \tag{5.8}$$

The mean receive processing requirement at a leaf receiver located at the same site as the designated receiver is

$$E[Y^{L3}] = E[M_l]E[Y_p] + (E[M_l - 1)E[Y_n] + p^2/(1-p)E[Y_t] \tag{5.9}$$

The mean receiver processing requirement at a designated receiver is

$$\begin{aligned} E[Z^{L3}] &= E[M_d](1-p)E[Y_p] + (E[M_d] - 1)E[Y_n] + p^2/(1-p)E[Y_t] \\ &\quad (E[M_l] - 1)(E[X_n + E[X_p]) \end{aligned} \tag{5.10}$$

It is important to note here that, the number of NAKs sent from a leaf receiver is only one less than the number of transmissions from its parent. This means that if the designated receiver does not have the repair packet, an end-receiver waits for sufficiently long time before sending a NAK, so that the designated receiver is itself able to recover the packet. Otherwise, the receiver is likely to send many more NAKs to its designated receiver than what we have counted in equations (5.8) and (5.10).

Using the recursive technique described in [1],

$$\begin{aligned} P[M_d \leq i] &= \sum_{u=0}^{u=i-1} \binom{i}{u} p_l^u (1-p_l)^{i-u} (1-p_l^{i-u})^W \\ P[M_l \leq i] &= \sum_{u=0}^{u=i-1} \binom{i}{u} p_l^u (1-p_l)^{i-u} (1-p_l^{i-u})^{k-1} \end{aligned}$$

$E[M_d]$ and $E[M_l]$ can now be derived using equation (5.4).

The maximum processing rate at the sender, $\Lambda_s^\omega = 1/E[X^\omega]$ and the maximum processing rate at the receiver, $\Lambda_r^\omega = 1/E[Y^\omega]$, where $\omega \epsilon \{L1, L2, L3\}$. Under the assumption that the repair server is never a bottleneck, the overall protocol throughput for L1 is given by the minimum of the per–packet processing rates at the sender and the receiver.

$$\Lambda_o^{L1} = \min\{\Lambda_s^{L1}, \Lambda_r^{L1}\} \tag{5.11}$$

The overall protocol throughput for L2 is given by the minimum of the per–packet processing rates at the sender and the receiver

$$\Lambda_o^{L2} = \min\{\Lambda_s^{L2}, \Lambda_r^{L2}\} \tag{5.12}$$

The overall protocol throughput for L3 is given by the minimum of the per–packet processing rates at the sender, the receiver and the designated receiver

$$\Lambda_o^{L3} = \min\{\Lambda_s^{L3}, \Lambda_r^{L3}, \Lambda_d^{L3}\} \tag{5.13}$$

Here the maximum processing rate at a designated repair server, $\Lambda_d^{L3} = 1/E[Z^{L3}]$.

5.2 Bandwidth Analysis

To analyze the bandwidth performance we consider the network to be made up of three types of links, a source link, backbone links and tail links. Then the mean total bandwidth usage, denoted by B^{ω}, where $\omega \epsilon \{L1, L2, L3\}$, can be expressed as,

$$E[B^{\omega}] = E[B_s^{\omega}] + E[B_b^{\omega}]W + E[B_t^{\omega}]T \tag{5.14}$$

Here $E[B_s^{\omega}]$, $E[B_b^{\omega}]$ and $E[B_t^{\omega}]$ are the mean bandwidth usage on the source link, a backbone link and a tail link respectively, for protocol ω. For determining these quantities we need to find the mean number of packets flowing in each of the links per successful transmission of a packet from the sender to all the receivers. For simplicity, we assume that the source, backbone and tail links each consists of a single physical link. A packet is counted if it is to be offered to a link.

The traffic concentration, denoted by c^{ω}, can be expressed as,

$$c^{\omega} = \max(E[B_s^{\omega}], E[B_t^{\omega}])/(E[B_s^{\omega}] + E[B_t^{\omega}]T/(T+1)) \tag{5.15}$$

We exclude the backbone links from equation (5.15) because we cannot pin–point the backbone links that are used for distributing packets across the backbone. This exclusion will not affect our results because the backbone links do not become bottlenecks. However, we do get somewhat lower values of traffic concentration because the bandwidth usage in the backbone links tends to reduce the average bandwidth usage over all links. For protocol L1,

$$\begin{aligned}
E[B_s^{L1}] &= E[M_s]E[B_d] + (E[M_s]-1)E[B_n] \\
E[B_b^{L1}] &= E[M_s](1-p_l)E[B_d] + (E[M_s]-1)E[B_n] \\
E[B_t^{L1}] &= E[M_R]E[B_d] + (E[M_r]-1)E[B_n]
\end{aligned}$$

Here, B_d is a random variable representing the bandwidth required for a data-packet transmission or retransmission and B_n is another random variable representing the bandwidth associated with a NAK over any of the above three types of links. We assume that B_d and B_n have general distributions and are independent of each other. For protocol L2,

$$\begin{aligned}
E[B_s^{L2}] &= (E[M'] + E[\lfloor (M-M')/2 \rfloor])E[B_d] \\
&\quad + (E[M'] + E[\lfloor (M-M')/2 \rfloor] - 1)E[B_n]
\end{aligned}$$

$$\begin{aligned}
E[B_b^{L2}] &= (E[M'] + E[\lfloor (M-M')/2 \rfloor])(1-p_l)E[B_d] \\
&\quad + (E[M'] + E[\lfloor (M-M')/2 \rfloor] - 1)E[B_n]
\end{aligned}$$

$$\begin{aligned}
E[B_t^{L2}] &= (E[M'] + E[\lfloor (M-M')/2 \rfloor])(1-p_l)E[B_d] \\
&\quad + (E[\lceil (M_n - M_n')/2 \rceil])(1/k + (k-1)(1-p_l)/k)E[B_d] \\
&\quad + (E[M'] + E[\lfloor (M-M')/2 \rfloor] - 1)E[B_n] \\
&\quad + (E[M_n'] + E[\lceil (M_n - M_n')/2 \rceil] - 1)E[B_n]
\end{aligned}$$

For protocol L3,

$$\begin{aligned} E[B_s^{L3}] &= E[M_d]E[B_d] + (E[M_d]-1)E[B_n] \\ E[B_b^{L3}] &= E[M_d](1-p_l)E[B_d] + (E[M_d]-1)E[B_n] \end{aligned}$$

Now the tails leading to sites with designated receivers have more traffic flowing through them in comparison to tails leading to sites without any designated receivers. For tails leading to sites with designated receivers,

$$\begin{aligned} E[B_{t_1}^{L3}] &= E[M_d](1-p_l)E[B_d] + (E[M_d]-1)E[B_n] \\ &\quad + (E[M_l]-1)E[B_d] + (E[M_l-1)E[B_n] \end{aligned}$$

For tails not leading to sites with designated receivers,

$$E[B_{t_2}^{L3}] = E[M_l](1-p_l)E[B_d] + (E[M_l-1)E[B_n]$$

Therefore, for protocol L3, the contribution of the tails links to the bandwidth usage is $E[B_{t_1}^{L3}]W + E[B_{t_2}^{L3}] * (T-W)$.

6 Throughput and Bandwidth Comparisons

In this section we first compare the throughput and bandwidth usage of protocols L1, L2 and L3 with a protocol N2 [18] that does not use local recovery[6] to establish the benefits of local recovery. N2 is a receiver oriented protocol that uses global NAK suppression and is shown to be the best of three global recovery schemes (proposed in [18]). Next we compare the performances of L1, L2 and L3. In computing L1's throughput we assume that the repair server has sufficient processing power and is never a bottleneck. The required processing power for this to be true is determined in the next section.

In order to compute the send and receive processing costs we use the measurements reported in [9] for processing time associated with sending/receiving a data packet and a NAK packet, as well as the timeout processing times. We use $E[X_p] = E[Y_p] = 500\mu$secs, $E[X_n] = E[Y_n] = 85\mu$secs and $E[Y_t] = 32\mu$secs. Here a data packet is of size 1024 bytes and a NAK packet is of size 32 bytes. The number of tails per neighborhood (as defined in Section 3.2), k, is set to 8. For bandwidth comparisons, we choose $E[B_d] = 1024$ and $E[B_n] = 32$.

Figures 2 and 3 show the throughput increase and bandwidth reduction due to local recovery. The performance of L1 is significantly higher than N2. Further, this behavior becomes more pronounced as the number of tails, T, and the loss probability, p increase. L2 also performs much better than

---

[6] N2 is analyzed for our system model not presented here for brevity.

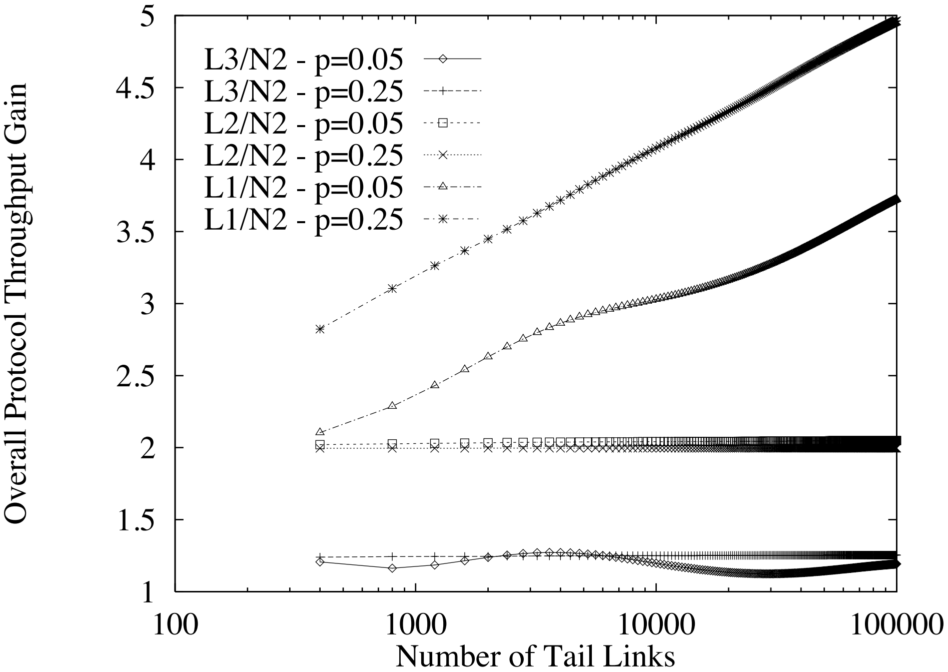

Figure 2: Throughput Gain

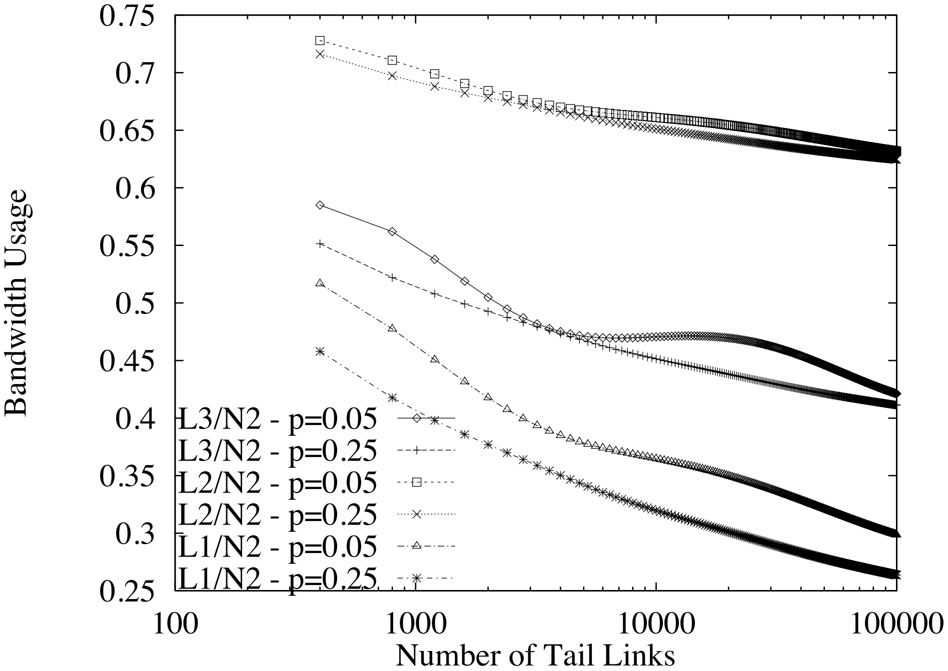

Figure 3: Bandwidth Reduction

N2. However, the ratio of throughput obtained under L2 and N2 remains equal to 2 and does not change significantly with increase in T or p. This is because the bottleneck node (receiver for $p = 0.05$ and sender for $p = 0.25$) under L2 does only half the amount of processing in comparison to the bottleneck node (sender) under protocol N2 due to L2's alternate local and global recovery. The throughput obtained under L3 is determined by the the designated receiver's throughput (or by the sender throughput for very high p and T) and is only marginally higher than that obtained under N2.

The throughput and bandwidth comparison of protocols L1, L2 and L3 is exhibited in Figures 4 and 5. This behavior could also be inferred from Figures 2 and 3. Protocol L1 has the highest throughput and lowest bandwidth usage in comparison to L2 and L3. The L1 sender is responsible for correctly transmitting packets only over the source link. An L1 receiver does not receive any redundant packets from the sender. Under L2 the sender is responsible for making sure that at least one receiver in each neighborhood receives the packet going through the source and tail links. Since all packets are multicast to all receivers, the receivers receive some redundant retransmissions from the sender. For these reasons the sender and the receivers in the case of L2 do much more processing in comparison to the sender and receivers under L1, leading to higher sender and receiver throughput under L1. This results in higher overall protocol throughput under L1. As more data packets and NAKs are multicast to the entire network under L2, L1 uses much less bandwidth than L2.

It is very interesting to note in Figures 4 and 5 that even though L2 achieves a higher throughput than L3, L2 also uses more bandwidth. This is because the responsibility of providing repairs is concentrated at the sender and the designated receivers under L3 even though less packets flow on the links. The sender in L3 has to continue sending retransmissions until the designated receiver in each local neighborhood correctly receives the packet. In the case of L2, the sender sends retransmissions only until at least one receiver in each local neighborhood receives the packet correctly (the additional transmissions due to failure of local recovery do not contribute much). Therefore the sender under L3 sends more retransmissions. Also, the designated receiver under L3 does much more processing than a receiver under L2. The bottleneck under L3 moves between sender and the designated receiver with change in loss probability and number of tail links. Thus L3 has lower throughput in comparison to L2. Due to the fact that retransmissions from the sender are sent only to the designated receivers, less bandwidth is used under L3.

We now look at the traffic concentration under protocols L1, L2 and L3. Figure 6 shows how the traffic concentration obtained under protocols L1, L2 and L3 varies with the loss probability for 10000 tails. It can be seen that the traffic concentration under L3 increases with loss probability. This is because traffic is concentrated on the tail links leading to the designated receivers (and the source link for high p and T). For $p = 0.25$ the traffic concentration under L3 is 250% higher than that under L1 and L2. The traffic concentration under L1 and L2 is always almost 1.

For a performance comparison that takes into account the processing costs at the repair servers, let us assume that the repair servers are the same kind of machines as the sender or or a receiver. Hence we consider $E[X_p]$, $E[Y_p]$, $E[X_n]$, $E[Y_n]$, $E[Y_t]$ for a repair server to be the same as that for

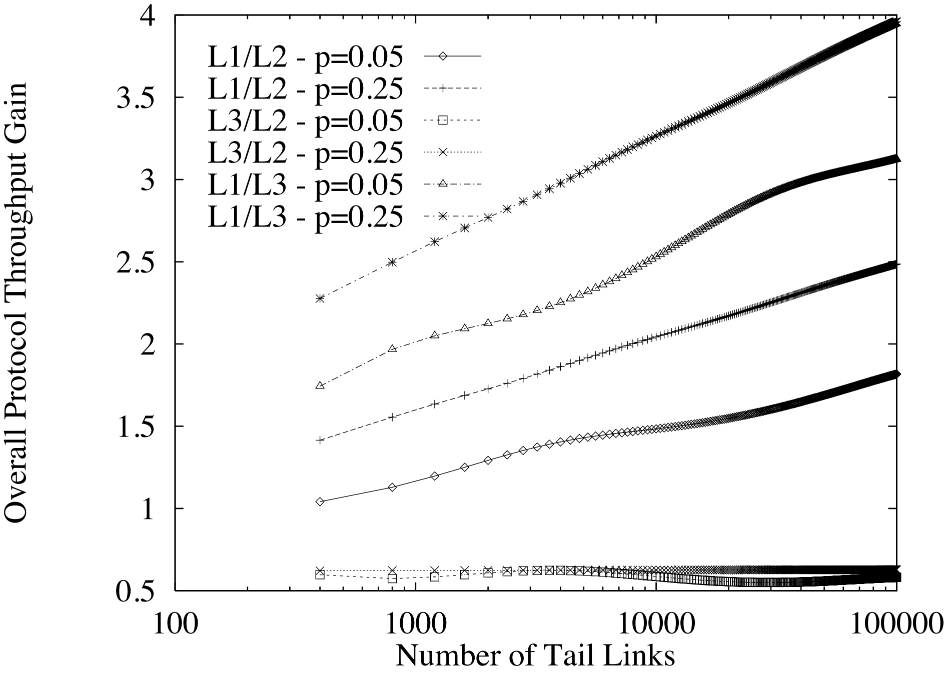

Figure 4: Throughput Gain

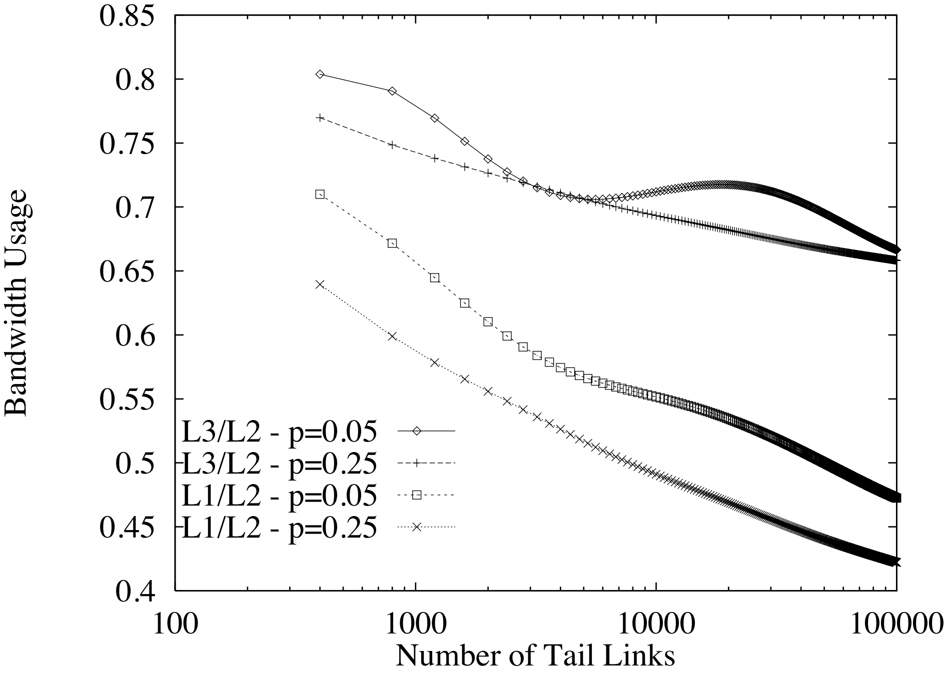

Figure 5: Bandwidth Reduction

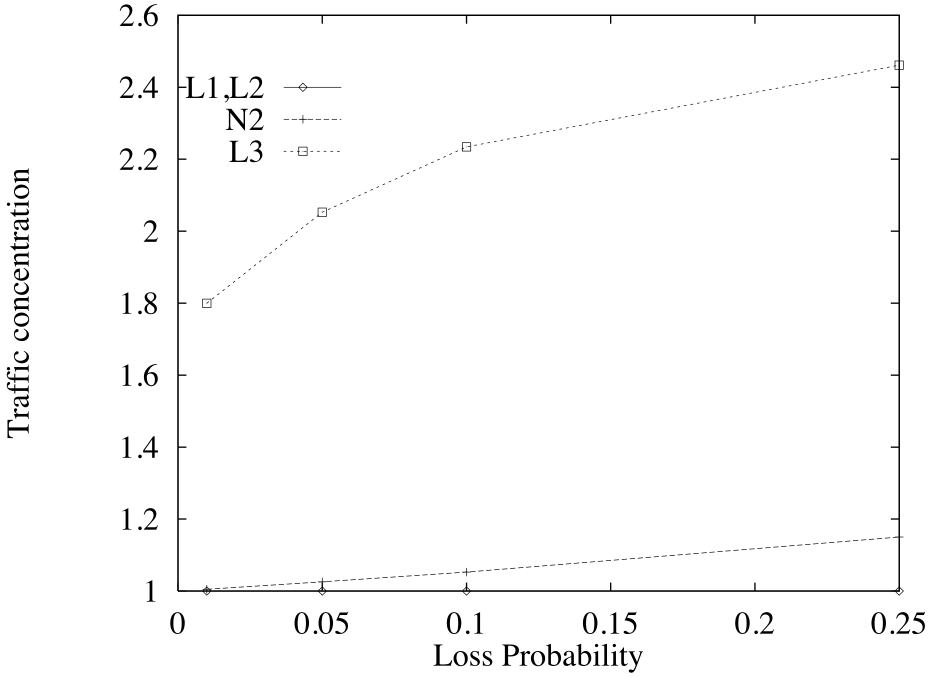

Figure 6: Traffic Concentration

a receiver or sender. We then find the mean total per packet processing costs at all nodes (sender, repair servers, receivers) under protocol L1. We also find the mean total per packet processing costs at all nodes (sender, receivers) under L2 and L3. For the case of a single receiver per tail circuit, Figure 7 shows how the mean total per packet processing ratio varies with T and p. It can be seen that for $p = 0.05$, the mean total per packet processing cost under L1 is more than that of L3. This is because for low loss probabilities the repair server does not need to supply many repairs but it still does the extra work of receiving all of the packets. This behavior is also observed while comparing L1 and L2 for $p = 0.05$ and less than thousand tails. With an increase in the number of tails and loss probability the mean total per packet processing cost under L2 and L3 starts increasing relative to that under L1. The mean total per packet processing cost under L3 is less than L2. Noting the fact that the receiver processing cost under L1 is the least, adding more receivers per tail will further reduce, relatively, the mean total per processing cost under L1.

In summary, we see that local recovery leads to higher protocol throughput and lower bandwidth usage. While comparing three local recovery protocols, if repair servers are not bottlenecks then L1 exhibits highest throughput and uses least bandwidth. Among L2 and L3, L2 has higher higher throughput but L3 has lower bandwidth usage. Interestingly, if we consider the repair servers to have the same processing power as sender or receivers then there is an overall reduction in processing costs in the repair server based approach with increasing loss probabilities, number of tail links and number of receiver per tail link.

The sender throughput under L3 is only slightly more than the designated receiver throughput. In

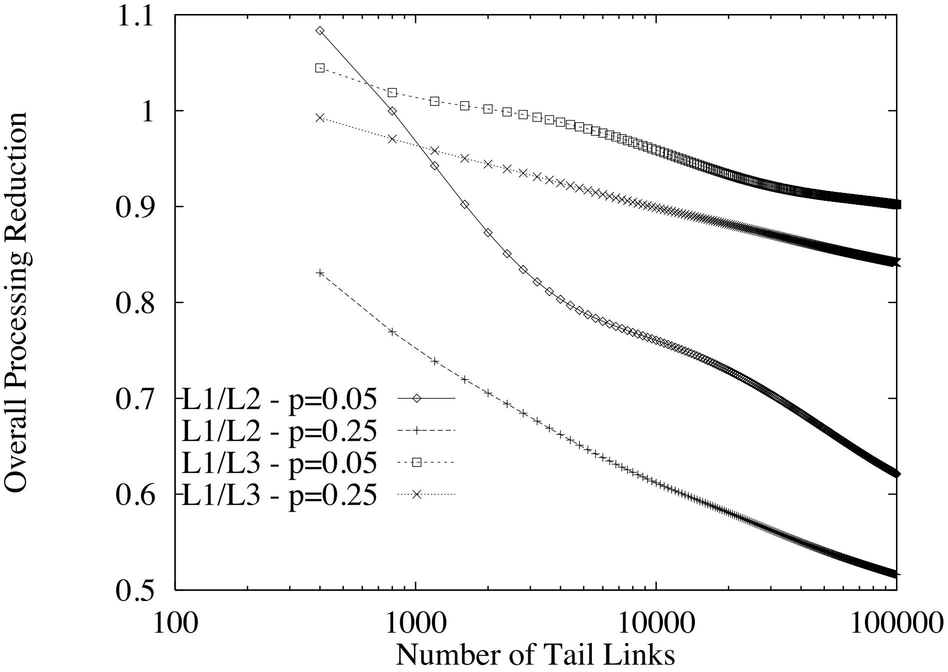

Figure 7: Mean Total Per Packet Processing Ratio

fact for higher values of $p(\geq 0.25)$ and large values of T the sender throughput is lower than the designated receive throughput. If protocol L3 is re-designed such that all the transmissions and retransmissions were multicast to every receiver and not just the designated receiver, then there will be less burden on the designated receivers to supply the repairs. Still, the throughput under L3 will improve only slightly (the retransmission processing cost at the sender still remaining the same). At the same time the bandwidth usage and mean total per packet processing will increase a lot more.

This also suggests that using faster machines as designated receivers will not help much because the sender throughput will dominate. L2 will still have a higher protocol throughput than L3. One way to improve the throughput without increasing bandwidth is to construct a logical tree [11] with a small and constant branching factor. Instead of two levels (sender sending retransmissions to a large number of designated receivers and designated receivers supplying repairs to other receivers), this tree would have several levels depending upon the number of tails links. The performance of this approach is very sensitive to the branching factor of the logical tree. We extend the analysis in [11] using our loss model to find that the logical tree approach provides excellent throughput and bandwidth performance when the branching factor is very small. However, for a large number of tails, a smaller branching factor increases the depth of the tree resulting in longer recovery paths. This can potentially lead to high delays. A meaningful comparison of the logical tree–based approach of [11] with the local recovery approaches examined in this report is possible only by modeling the delay behavior in addition to modeling the throughput and bandwidth usage.

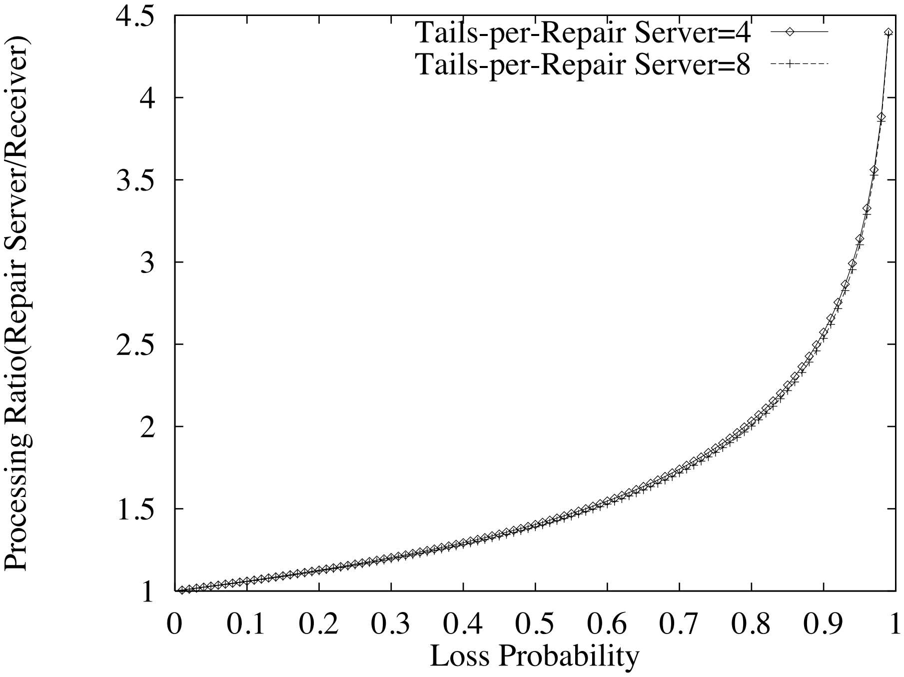

Figure 8: Ratio of Repair Server to Receiver Processing Cost

7 Sizing the Repair Servers in L1

In the previous section the throughput of protocol L1 was obtained under the assumption that the repair server has sufficient processing power and is never a bottleneck. In this section we determine the processing and buffer requirements of a repair server so that it does not become a bottleneck.

7.1 Processing Power

A repair server must perform receive–side processing of packets and NAKs, and send–side processing of repairs. We now determine the processing power required at a repair server, per multicast session, so that it does not become the bottleneck. Since a receiver performs more processing than the sender under L1, we determine the processing power of the repair server in terms of that receiver (using equations (5.2) and (5.3)). If a repair server can match the receiver throughput then it will never be a bottleneck in a multicast session that uses protocol L1.

In Figure 8 we plot the ratio of mean repair server processing cost (assuming that it has the same processing power as a receiver) and the mean receiver processing cost for varying end-to-end loss probability, p. A repair server needs a processing power that is no more than 1.16 times that of a receiver for a number of tail links per repair server equal to 8 and for loss probabilities less than 0.25. The reduction in tail links per server does not change this requirement because even though the repair server has to supply fewer repairs, the processing cost at a receiver is also reduced because it now receives fewer retransmissions of a packet.

Thus a repair server need only be a little faster than a receiver to avoid becoming a bottleneck itself for reasonable loss probabilities. This result is good for one multicast session. If a repair server is to handle K multicast sessions, then it has to be K time faster than a receiver.

7.2 Buffer Requirements

A repair server has to buffer a set of packets in order to be able to retransmit them due to losses in the tail links. Theoretically, each packet should be held for an infinitely long time to ensure perfectly reliable local recovery from a repair server. Realistically, it is possible to use a finite size buffer such that the probability of local recovery failure from the repair server is extremely small. In the rare event of local recovery failure, the lost packet should be retrieved from the sender. In this section, we use simple analysis to determine the expected buffer requirement at each repair server as a function of the probability of local recovery failure.

We model a repair server as an infinite capacity queue with an infinite number of servers, assuming that the repair server has sufficient processing power. Hence, a packet is immediately assigned a server on its arrival at the queue. The server holding time of each packet is determined by the number of retransmissions from the repair server to the receivers (the repair server is responsible for) and the time delay between retransmissions. The number of retransmissions from the repair server to the receivers is determined by the probability of imperfect recovery. A packet occupies a buffer for the duration of the holding time.

If n is the number of retransmissions of a packet from the repair server, and, λ is the mean packet arrival rate, and, the delay between the repair server and the receivers is d then, from Little's law, the expected buffer occupancy at a repair server is equal to

$$\lambda n d$$

The number n is determined as follows. Let N be the number of retransmissions of a packet from the repair server. If ϵ is the probability of local recovery failure that can be tolerated, then n is the minimum value of m ($m \geq 0$) that satisfies the inequality, $P(N > m) < \epsilon$. $P(P(N > m)$ is determined by using the following equation.

$$P(N > m) = 1 - P(N \leq m) \quad = \quad 1 - (1 - p_l^{m+1})^k$$

Recall from Section 4 that p_l is the tail link loss probability and k is the number of tail links a repair server is responsible for.

Numerical Example

We choose d = 20ms (the approximate round trip delay from UMass, Amherst to its backbone router) and $k = 8$. The mean packet arrival rate is equated to the overall throughput of protocol L1, i.e., $\lambda = \Lambda_o^{L1}$. Figure 9 shows the dependence of expected buffer occupancy (in kbytes) on ϵ for two different end–to–end loss probabilities, $p = 0.05, 0.25$ (recall that $p = 1 - (1 - p_l^2))$. The

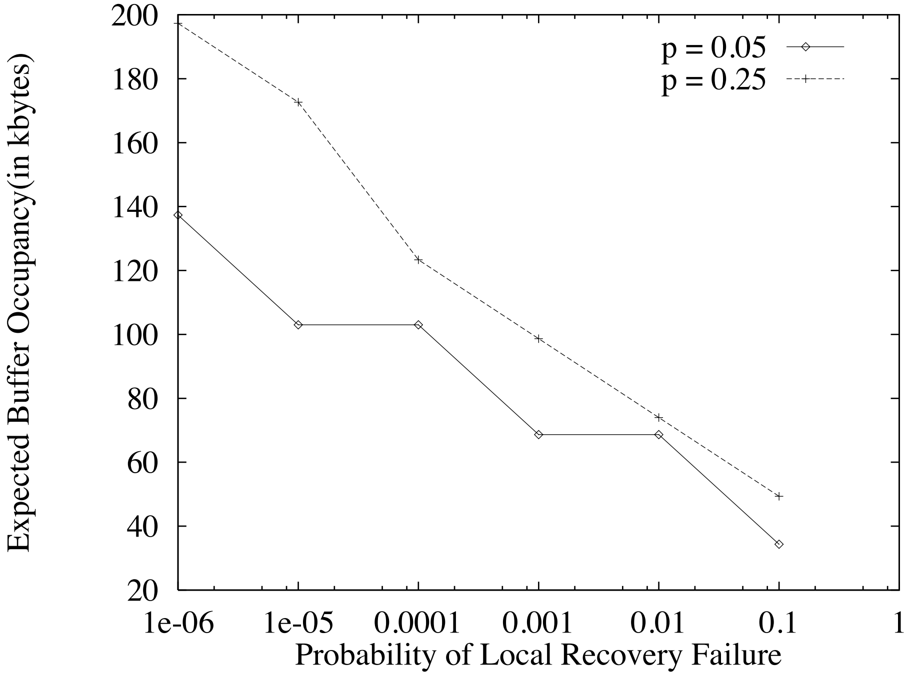

Figure 9: Buffer Requirement at a Repair Server, $d = 20ms$ and $l = 8$

expected buffer occupancy at a repair server is 198 kbytes and 137 kbytes, for $p = 0.25$ and $p = 0.05$ respectively when $\epsilon = 10^{-6}$.

Thus a repair server requires a few 100 kilobytes of buffer space per multicast session for reasonable loss probabilities.

8 Conclusions

In this report we have investigated one server–based and two receiver–based local recovery approaches for scalable reliable multicast. In the server–based approach, designated hosts, placed inside the network, are used as repair servers. In the receiver–based approaches, error recovery is done involving only the participating receivers and the sender. We analyzed and compared protocols developed for these approaches. Using analytical models, we demonstrated the performance gains in using the server–approach in terms of protocol throughput, network bandwidth and overall processing costs. The performance gains increase as the size of the network and the loss probability increase making the server–based approach more scalable with respect to these parameters. We also estimated the processing power and buffer capacity required at the repair servers per multicast session for achieving the performance gains.

A server–based approach could be deployed to improve performance in Intranets where services requiring reliable multicast are offered. An Intranet network provider could place the repair servers at appropriate locations. Depending on the processing power and buffer space that can be provided on the repair servers, the network provider could restrict the use of repair servers for only certain number of multicast sessions (by assigning separate multicast addresses and limiting the use of repair servers only for those addresses). More multicast sessions could be allowed with the increase in processing power and buffer space at the repair servers.

As far as the Internet is concerned, wide scale *static* deployment of repair servers with sufficient processing power and buffer capacity may not be easy. It is possible that receiver–based approaches, even with a lower performance, may be more attractive. Given the performance improvements of the server–based approach, it is worth studying how to make it more dynamic and flexible. In this report, we have considered the mean total per packet processing cost at all nodes (sender, receivers and repair servers) as a performance metric for taking into account the processing cost of repair servers. We need to find better cost metrics to take into account the processing and buffering resources used inside the network. We also need to study the performance degradation due to insufficient processing and buffering resources at the repair servers.

We argued that the receiver–based approach that builds logical trees is likely to incur higher delays. Hence there is an immediate need to analyze the delay of this approach (and also L1 and L2 for comparison) and study the tradeoff between delay, throughput and bandwidth usage. Our system model could be extended to take into account temporal correlation in the network loss[20] and to also consider more complex topologies with heterogeneity. It is easy to extend the analysis of L1 and L3 to any complex topology. The analysis of L2 is slightly more tricky. Recently there has been an increasing interest in reliable multicast approaches using forward error correction (FEC) [13, 16]. These approaches are based on end-to-end recovery from the sender. They have the potential to reduce network bandwidth usage. It would be interesting to compare the bandwidth usage of local recovery approaches with those that use FEC.

References

[1] P. Bhagawat, P.P. Mishra and S.K. Tripathi, *Effect of Topology in Performance of Reliable Multicast Communication.* Proceedings of IEEE Infocom, June 1994.

[2] T. Billhartz, J.B. Cain, E. Farrey-Goudreau, D. Fieg and S. Batsell, *Performance and Resource Cost Comparisons for the CBT and PIM Multicast Routing Protocols in DIS Environments.* Proceedings of IEEE Infocom, 1996.

[3] K. Calvert, M. Doar and E. Zugera, *Modeling Internet Topology.* IEEE Communications Magazine, June 1997.

[4] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu and L. Wei, *An Architecture for Wide Area Multicast Routing.* Proceedings of ACM SIGCOMM, pages 126-135, August 1994.

[5] S. Floyd, V. Jacobson, S. McCanne, C. Liu and L. Zhang, *A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing.* Proceedings of ACM SIGCOMM, pages 342-356, August 1995.

[6] S. Floyd, V. Jacobson, S. McCanne, C. Liu and L. Zhang, *A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing.* A later version of the ACM SIGCOMM paper, November 1995.

[7] M. Hofmann, *Enabling Group Communication in Global Networks.* Proceedings of Global Networking'97, Calgary, Canada, November 1996.

[8] H.W. Holbrook, S.K. Singhal and D.R. Cheriton, *Log-Based Receiver Reliable Multicast for Distributed Interactive Simulation.* Proceedings of ACM SIGCOMM, pages 328-341, August 1995.

[9] Sneha K. Kasera, Jim Kurose and Don Towsley *Scalable Reliable Multicast Using Multiple Multicast Groups.* Proceedings of ACM Sigmetrics Conference, June 1997.

[10] B.N. Levine and J. Garcia-Luna-Aceves, *A Comparison of Known Classes of Reliable Multicast Protocols.* Proceedings of IEEE ICCC, November 1996.

[11] B.N. Levine, D.B. Lavo and J. Garcia-Luna-Aceves, *The Case for Reliable Concurrent Multicasting Using Shared Ack Trees.* Proceedings of ACM Multimedia, November 1996.

[12] J.C. Lin and S. Paul, *RMTP: A Reliable Multicast Transport Protocol.* Proceedings of IEEE Infocom, 1995.

[13] J. Nonnenmacher, E. Biersack and D. Towsley, *Parity–Based Loss Recovery for Reliable Multicast Transmission.* Proceedings of ACM SIGCOMM, August 1997.

[14] C. Papadopoulos, G. Parulkar and G. Varghese, *An Error Control Scheme for Large–Scale Multicast Applications.* Proceedings of IEEE Infocom, March 1998.

[15] S. Pingali, D. Towsley and J. Kurose, *A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols.* Proceedings of ACM Sigmetrics, 1994.

[16] L. Rizzo and L. Vicisano, *A Reliable Multicast Data Distribution Protocol Based on Software FEC Techniques.* Proceedings of the Fourth IEEE HPCS'97 Workshop, Chalkidiki, Greece, June 1997.

[17] D. Tennenhouse, J. Smith, W. Sincoskie, D. Wetherall and G. Minden, *A Survey of Active Network Research.* IEEE Communications Magazine, January 1997.

[18] D. Towsley, J. Kurose and S. Pingali, *A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols.* IEEE JSAC, April 1997.

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

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

[21] R. Yavatkar, J. Griffioen and M. Sudan, *A Reliable Dissemination Protocol for Interactive Collaborative Applications.* Proceeding of ACM Multimedia, November 1995.