Buffer Requirements and Replacement Policies for Multicast Repair Service*

Sneha Kumar Kasera
Bell Labs Research
Lucent Technologies
Holmdel, NJ 07733
kasera@research.bell-labs.com

Jim Kurose
Computer Science Department
University of Massachusetts
Amherst, MA 01003
kurose@cs.umass.edu

Don Towsley
Computer Science Department
University of Massachusetts
Amherst, MA 01003
towsley@cs.umass.edu

ABSTRACT

Server-based local recovery for reliable multicast will perform efficiently only when sufficient processing and buffering resources are available at the servers. In this paper we examine the buffer requirements of servers for reliable multicast. We show how the server buffer requirements depend upon the packet arrival process at the server and the duration of time during which a packet needs to be held in its buffer. This latter quantity depends upon the required number of retransmissions and the length of time between retransmissions of the packet. We find that buffer requirements are very sensitive to small values of the number of retransmissions of a packet within the packet inter-arrival time but increase very slowly with the number of receivers a server is responsible for supplying repairs. We also determine the effect of the number of servers and buffer size at a server on the end-system throughput and network bandwidth usage.

We examine three buffer replacement policies; FIFO, FIFO with minimum buffer holding time or FIFO-MH (that we propose) and LRU. Based on our performance study and noting that it could also be easily implemented, we recommend the use of the simple FIFO buffer replacement policy.

1. INTRODUCTION

*This work was done when the first author was a Ph.D. student at University of Massachusetts Amherst. It was supported by DARPA under Contract No. N66001-97-C-8513 and NSF under Grant No. CDA-9502639. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, of the U.S. Government. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

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. Earlier research [1, 2, 4] has demonstrated the higher performance of server-based local recovery approach in which strategically placed servers, also called repair servers [2], aid in local recovery. However, server-based local recovery will perform efficiently only when sufficient processing and buffering resources are available at the repair servers. Since these repair servers are likely to be used by a large number of reliable multicast sessions, as well as by other applications including web caching and multimedia gateways, it is extremely important to understand their resource requirements. In this paper we examine the buffer requirements of repair servers for reliable multicast.

The most important function of a repair server is to buffer sender's transmissions in order to be able to later retransmit them to repair downstream losses on receiving negative acknowledgments or NAKs[1]. Theoretically, in a NAK-based approach such as ours, a repair server needs to buffer each packet for an infinitely long amount of time so that it can retransmit a packet whenever a NAK for that packet is received. Such perfect local recovery requires an infinite buffer space at the repair servers. Realistically, a multicast session will be allocated a certain number of buffers; when all its buffers are full, a buffer replacement policy will be used to replace old packets occupying the buffer with new packets arriving from the sender. Given the replacement of buffered packets, it is possible that some of the packets will be removed from the buffer at the repair server before they are successfully received downstream. These packets must now be recovered from upstream of the repair server.

Our work in this paper examines the buffer requirements of repair servers; it divides into three parts:

- First, for a simple FIFO (*oldest-first*) buffer replacement policy, we use analytical models to determine the amount of buffer space required at a single repair server to ensure that it can almost always supply a repair when required. We show how the repair server

[1]It has been shown in [8] that positive acknowledgment or ACK-based approaches are not scalable. In our work we consider NAK-based approaches only.

buffer requirements depend upon the packet arrival process at the repair server and the duration of time during which a packet needs to be held in its buffer. This latter quantity depends upon the required number of retransmissions and the length of time between retransmissions of the packet.

- Next, we extend the FIFO buffer analysis of the single repair server case to a topology consisting of multiple repair servers serving different loss domains (a loss domain consists of a repair server and its downstream receivers) to determine how the end-system throughput and network bandwidth consumption will vary with buffer size. In the same context, we study the impact of the number of repair servers in the multicast tree on the mean number of additional retransmissions required from the sender. We show that for homogeneous loss domains it is necessary to populate almost all of the domains with repair servers before a marked reduction is observed in the mean number of additional retransmissions from the sender.
- Third, for the purpose of improving buffer utilization and ensuring graceful performance degradation when only a small number of buffers are available, we propose a new buffer replacement policy. This policy, called FIFO-MH, replaces packets in a FIFO manner only after they have been held for a specified minimum amount of time. We examine the effect of different retransmission request arrival patterns on the performance of FIFO-MH. We also simulate the LRU (*least recently used*) replacement policy. Based on our performance study and noting that it could also be easily implemented, we recommend the use of the simple FIFO buffer replacement policy.

The rest of this paper is structured as follows. In Section 2 we examine the related work on buffer requirements for reliable multicast. In Section 3 we present our network model. In Section 4 we study the buffer requirements of a single repair server when a simple FIFO buffer replacement policy is used. In Section 5 we consider a network consisting of a sender, receiver and multiple repair servers and extend the analysis of Section 4 to determine the dependence of the end-system throughput and network bandwidth consumption on buffer size. In Section 6, we consider more complex buffer management policies for improving buffer utilization. Our conclusions and directions for future work are described in Section 7.

2. RELATED WORK

Most of the earlier performance studies on reliable multicast have focussed upon network bandwidth and end-host resources. There has been very little work on memory requirements for caching packets for reliable multicast. In our earlier work [2], we made the assumption that a repair server can store an infinite number of packets and determined the average buffer occupancy at the repair servers; we did not study any buffer replacement policies. Our work on buffer occupancy has been extended in [7] to determine the buffer occupancy at repair servers for protocols that provide parity encoding or forward error correction service. Our buffer occupancy formalism has also been used to study the activation/deactivation of repair servers in [6].

3. NETWORK MODEL

We adopt the same network model as in [2]. A sender multicasts data to a group of receivers. In addition, there are designated hosts, called repair servers, that are co-located with routers, typically at strategic points (i.e., above lossy links) inside the network. The sender and the repair servers conceptually constitute a *repair tree* rooted at the sender. The repair servers join the multicast group on which data is being transmitted and receive data packets in exactly the same manner as other receivers. These packets are cached for the purpose of supplying future repairs. On detection of loss, a receiver recovers the packet from the closest upper-level repair server. The repair servers in turn recover lost packets from upper-level repair servers. The sender can be considered the highest level repair server.

Note that the repair tree is the same as the physical multicast tree. Therefore, a packet lost at a certain level cannot be received at a lower level. The findings of this paper can also be applied to logically hierarchical repair trees [3, 4, 9], although we do not consider that application here. We assume that the reliable multicast protocol used for error recovery is L1 (or L1-like), as described in [2].

4. BUFFER REQUIREMENTS OF A REPAIR SERVER

In this section, we consider a single repair server that has the responsibility of supplying repairs to its downstream receivers, as shown in Figure 1. The repair server buffers packets that arrive from the sender in order to supply future repairs. When the repair server's buffers are full, a buffer replacement policy is used to replace old packets occupying the buffer with new packets arriving from the upstream sender. It is possible that some of the packets will be removed from the buffer at the repair server before they are successfully received by all of the downstream receivers. These packets must then be recovered from an upstream repair server or from the sender. Additional retransmissions will thus be required from an upper-level repair server or sender. Our goal in this section is to determine the average number of these additional retransmissions as a function of the buffer size at the repair server. We then determine the buffer size required to maintain the average number of such additional retransmissions from upstream repair servers below a threshold.

Before proceeding with our buffer requirement study, we introduce a key parameter, the retransmission interval, that is used in our study. We define *retransmission interval* to mean the following two kinds of time intervals; the time interval from when a packet is first buffered at a repair server until it is first retransmitted and the time interval between successive retransmissions of the packet from the repair server.

In NAK-based protocols, a repair server retransmits a packet on receiving a NAK for that packet from downstream receivers. NAKs can arrive at a repair server at variable time intervals due to the following reasons.

- The round trip times from the repair server to the receivers are variable.
- The time to detect loss at receivers is also variable

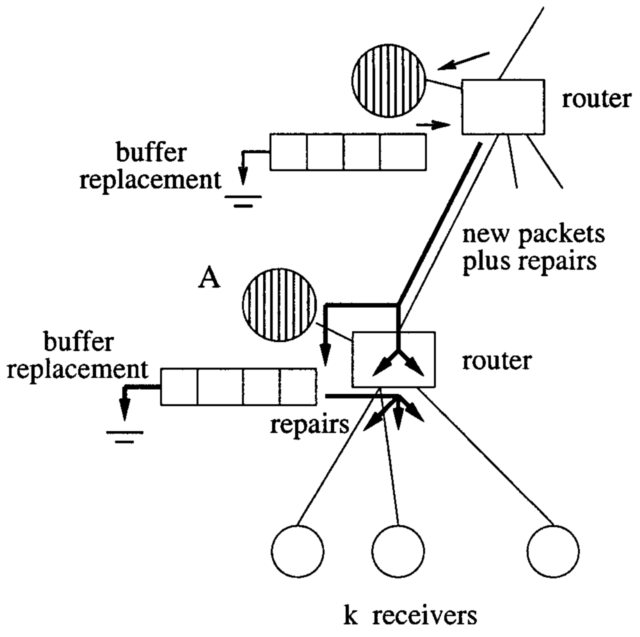

Figure 1: Repair server serving k receivers

(e.g., depends on the inter–packet separation[2]).

- NAKs are sent after a random delay for the purpose of NAK suppression.

If the repair server retransmits a packet whenever and as soon as it receives a NAK for that packet, the retransmission interval is a variable quantity and the repair server has no control over its values. Alternatively, a repair server protocol might batch NAKs for a packet and send out retransmissions of the packet periodically as long as there is a pending NAK for that packet. In the second case, the retransmission interval is a constant and its value can be set by the repair server.

We now determine, through analysis, the mean number of additional retransmissions that an arbitrary packet incurs from an upstream repair server (or the sender) in order to ensure that it is correctly received by all receivers below the repair server. In the analysis, we make the following assumptions. Note that these are modeling assumptions, not protocol requirements. We focus on a single repair server (e.g., A in Figure 1) that supplies repairs to k downstream receivers and itself receives packets from upstream repair servers and/or the original sender[3].

- Packets arrive at the repair server according to a Poisson process with arrival rate $\lambda = 1/\delta$. These also include the retransmissions of packets that were lost upstream of the repair server. These packets are buffered at the repair server on arrival. The repair server can buffer up to B packets. When all of the buffers are occupied, packets are replaced in simple FIFO (or oldest first) order.

- The repair server will periodically retransmit a packet every δ' time units, as long as it receives NAKs for that packet, i.e., the retransmission interval is a constant[4] denoted by δ'.

- The repair server is responsible for supplying repairs to k receivers. Receiver i sees independent loss on the link between the repair server and itself with probability p_i, $i = 1, 2, \ldots, k$.

- A packet, once buffered and subsequently overwritten, is not buffered again at the repair server. The Poisson arrivals described above do not include these packets. There are two reasons for this assumption. First, a scheme that buffers packets that were once buffered and subsequently overwritten by replacing other packets in the repair server buffers will do well only when certain packets are more "desirable" than the others. This might be true in those contexts where there is *locality in reference* (for example, web caches) but unlikely in the reliable multicast case (see the discussion on using *least recently used* buffer replacement policy in Section 6). Second, this assumption greatly simplifies the analysis.

4.1 Analysis

Let the random variable H represent the number of possible retransmissions of an arbitrary packet occupying a buffer at the repair server before it is replaced. This quantity is independent of whether the packet is actually ever retransmitted. As the repair server has B buffers, no retransmissions of the packet are possible if B additional packets arrive subsequent to the packet in time less than δ'. We define the random variable $Q(\tau)$ to denote the number of packet arrivals at the repair server within a time interval of length τ. Therefore,

$$P(H=0) \;=\; P(Q(\delta') \geq B)$$

For $i = 1, 2, 3, \ldots$,

$$P(H=i) \;=\; P(Q(i\delta') < B \leq Q((i+1)\delta'))$$

For $i = 0, 1, 2, \ldots$,

$$P(Q(i\delta') \geq B) \;=\; 1 - \sum_{n=0}^{B-1} \frac{(i\lambda\delta')^n e^{-i\lambda\delta'}}{n!}$$

Let the random variable N denote the number of additional retransmissions required from an upstream repair server or the sender. Let the random variable M denote the number of transmissions required for all k receivers to correctly receive the packet, if the repair server were to supply all the

[2]Losses are usually detected by observing gaps in the sequence numbers of the packets received and hence the time to detect loss also depends upon the inter–packet separation time.

[3]In this paper, when we use the term "upstream," it is taken to mean upstream with respect to the repair server, unless otherwise qualified

[4]This assumption is justified when a repair server protocol batches retransmission requests (NAKs) and sends out retransmissions of a packet periodically as long as there is a pending NAK for that packet. The case when δ' is a variable is studied in Section 6

transmissions. Then,

$$
\begin{aligned}
P(N \leq m) &= \sum_{h=0}^{\infty} P(N \leq m | H = h) P(H = h) \\
&= \sum_{h=0}^{\infty} P(M \leq m + 1 + h) P(H = h) \quad (1)
\end{aligned}
$$

where $P(N \leq m|H = h) = P(M \leq m + 1 + h)$ is the conditional probability that the number of retransmissions from upstream is less than or equal to m given $H = h$. We add one to $m + h$ to account for the original transmission of the packet.

The mean number of additional retransmissions from upstream, $E[N]$, is expressed as follows.

$$
\begin{aligned}
E[N] &= \sum_{m=0}^{\infty} (1 - P(N \leq m)) \\
&= \sum_{m=0}^{\infty} (1 - \sum_{h=0}^{\infty} P(M \leq m + 1 + h) P(H = h)) \\
&= \sum_{m=0}^{\infty} (1 - \sum_{h=0}^{\infty} (1 - p_1^{m+1+h}) \ldots \\
&\quad (1 - p_k^{m+1+h}) P(H = h))
\end{aligned}
$$

When $p_1 = p_2 = \ldots = p_k = p$, we have

$$
E[N] = \sum_{m=0}^{\infty} (1 - \sum_{h=0}^{\infty} (1 - p^{m+1+h})^k P(H = h)) \quad (2)
$$

The value $E[N]$ is a measure of the inability of the repair server to supply local repairs. In Section 6, we will look at ways to minimize $E[N]$. Using (2), we can now find the buffer requirements at a repair server as a function of $E[N]$, p, k, λ $(= 1/\delta)$ and δ'. Note that λ and δ' always occur together as a product in the above expressions.

4.2 Numerical Examples

We use (2) to find the minimum buffer space required to maintain $E[N]$ below a certain value for different values of p, R and $\lambda\delta'$. In this section, we choose to maintain $E[N] \leq 0.01$, i.e., on average, one additional retransmission or less will be required for every hundred packets.

We first study the impact of the product $\lambda\delta'$, or the ratio δ/δ', on the buffer requirement. The ratio δ/δ' is a key parameter. Its value can be interpreted as the number of retransmissions of a packet from the time it arrives at the repair server until the time the next packet arrives. If only one retransmission were required to ensure that all downstream receivers receive the packet, and, δ/δ' were greater than 1, then only one buffer would be required at the repair server. The buffer requirement goes up as δ/δ' decreases or as the loss probability p increases, as more retransmissions are needed in this latter case.

Figure 2 shows how the minimum buffer size varies with the ratio δ/δ' for different values of p. Here, k is set to 10. We see that the minimum buffer size is very sensitive to small

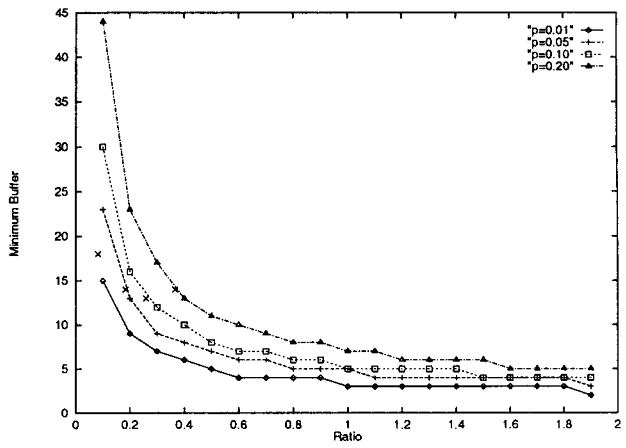

Figure 2: Minimum Buffer Size vs δ/δ'

values of δ/δ'. When $\delta/\delta' = 0.2$, 13 buffers are required to satisfy $E[N] \leq 0.01$. An increase in the arrival rate or in the time between retransmissions of a packet results in decrease in δ/δ', which results in the need for more buffers. At the same time, moderate to high values of δ/δ' result in extremely small buffer requirements. These results confirm our intuition as discussed above.

In order to obtain realistic buffer requirements, it is important to determine realistic values of the ratio δ/δ'. A very rough value for this ratio can be obtained if we assume that the reliable multicast sender transmits packets at a rate, S, that is **TCP–friendly** with respect to the link loss rate, p. Using the formula $S = 1.22/(RTT\sqrt{P})$ (from [5]) for a reliable TCP connection between a sender and a receiver, where S is the sender rate in pkts/sec, RTT is the round trip time from the sender to the receiver and P is the loss probability between the sender and the receiver, if we set $S = 1/\delta$, $RTT = \delta'$ and $P = p$ then we have $\delta/\delta' = \sqrt{p}/1.22$. We use this formula to select values of δ/δ' and find that 13–18 buffers are required to satisfy $E[N] \leq 0.01$, for a wide range of p (from 0.01 – 0.20). The buffer requirements corresponding to the estimated values of δ/δ' are shown by a 'x' in Figure 2 where the leftmost 'x' corresponds to $p = 0.01$ and the rightmost corresponds to $p = 0.20$.

We now study the impact of the number of receivers, k, on the buffer requirement. Figure 3 shows how the minimum buffer size needed to keep $E[N] \leq 0.01$ varies with the number of receivers. Here the arrival rate is 128pkts/sec ($\delta = 7.81$ms) and $\delta' = 40$ms. We observe that the buffer requirement changes very slowly with number of receivers. The same behavior is observed for other values of the δ and δ'. This is because the number of retransmissions required from the repair server, A (Figure 1), for all receivers to correctly receive the packet increases slowly as a function of k ($O(logk)$) [8]. In both Figures 2 and 3 the buffer requirement increases with link loss probability, as expected.

In summary, we find that the buffer requirements are very sensitive to small values of the number of retransmissions of a packet within the packet inter–arrival time. When this

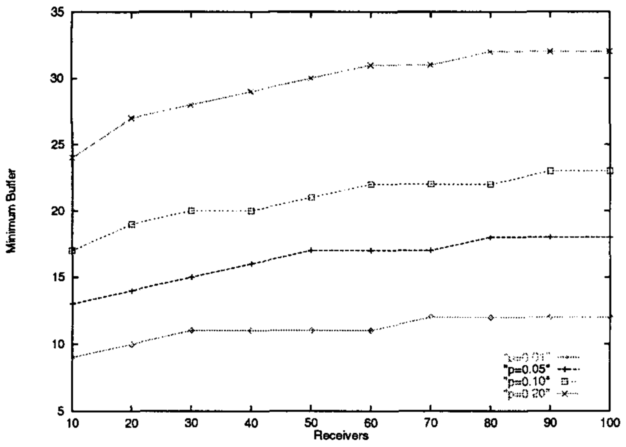

Figure 3: Minimum Buffer Size vs Number of Receivers

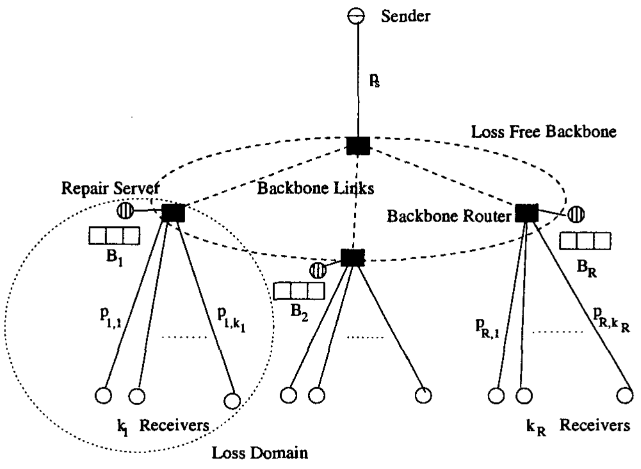

Figure 4: Repair server serving k receivers

value is large, very few buffers are required at the repair server to meet the demands of local recovery. We also find that the buffer requirements increase very slowly as the number of receivers increases.

5. DEPENDENCE OF OVERALL PERFORMANCE ON BUFFER SIZE

In the last section we found the mean number of additional retransmissions required, due to finite buffer space at a repair server, from an upper–level repair server (or sender) in the repair tree. In this section we consider the entire repair tree as described in Section 2. We determine, through analysis, the mean additional retransmissions required from the sender as a function of buffer space at the repair servers. This metric determines the effect of buffer size on the end–system throughput and network bandwidth usage. To compute this metric, we make the following assumptions.

- Our system model (see Figure 4) is similar to the one used in [2]. There are R repair servers. Repair server i has B_i buffers and is responsible for supplying repairs to its k_i downstream receivers, where $i = 1, 2, \ldots R$. We consider loss only in the links from the source to the backbone and from the backbone to the receivers[5]. The loss seen by receiver j in loss domain i on the link(s) from the backbone to itself is denoted by $p_{i,j}$. The loss on the link(s) from the sender to the backbone is represented by p_s. Loss events are assumed to be temporally independent. Each loss domain in Figure 4 is the same as the single repair server system studied in Section 3.
- As before, packets arrive at each repair server according to a Poisson process with arrival rate $\lambda = 1/\delta$. These arrivals also include the retransmissions of packets that were lost on the link(s) from the source to the backbone. Packets are buffered at the repair server on arrival. Repair server i can buffer up to B_i packets. When all of the buffers at a repair server are occupied, packets are replaced in simple FIFO (or oldest first) order. A repair server retransmits a packet with a fixed time interval δ' as long as it receives NAKs for that packet. A packet once buffered and subsequently overwritten is not buffered again at a repair server.

5.1 Analysis

We focus on an arbitrary packet sent by the sender. Let the random variable A_i denote the number of additional transmissions from the sender due to finite buffers at repair server i, and the random variable A denote the total number of additional retransmissions from the sender due to finite buffers at all the repair servers. Then,

$$\begin{aligned} P(A \le m) &= P(\max(A_1, A_2, \ldots, A_R) \le m) \\ &= \prod_{i=1}^{R} P(A_i \le m) \qquad (3) \end{aligned}$$

Using (1) from the previous section, we have

$$\begin{aligned} P(A_i \le m) &= \sum_{h_i=0}^{\infty} (1 - p_{i,1}^{m+1+h_i}) \ldots \\ & \quad (1 - p_{i,k_i}^{m+1+h_i}) P(H_i = h_i) \qquad (4) \end{aligned}$$

where H_i is the random variable denoting the number of possible retransmissions of the packet from repair server i before it is replaced.

The mean number of total additional retransmissions from the sender due to finite buffer space at the repair servers can be determined by using (3) and (4) in the following expression.

$$E[A] = (1/(1 - p_s))(\sum_{m=0}^{\infty}(1 - P(A \le m))) \qquad (5)$$

In equation (5), the factor $1/(1 - p_s)$ is used to account for the loss in the link(s) from the sender to the backbone.

5.2 Numerical Examples

We use (5) to examine how the overall performance depends on buffer size at repair servers. We assume that $k_i = k$, $B_i = B$ and $p_{i,j} = p$ for $i = 1, 2, \ldots, R$ and $j = 1, 2, \ldots, k$.

[5]Each site below a tail link (defined in Chapter 3) is assumed to contain only one receiver.

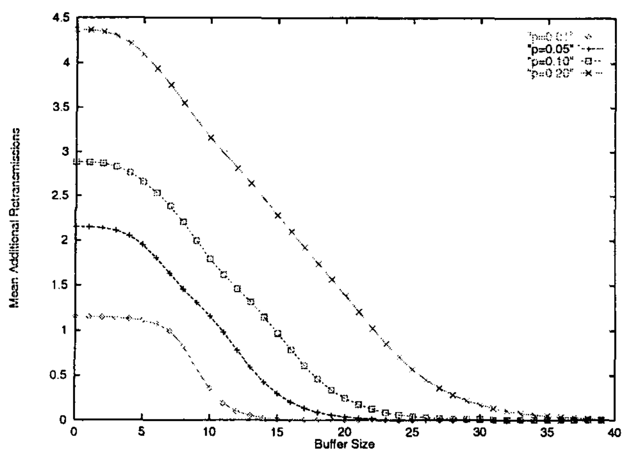

Figure 5: Mean Additional Retransmissions vs Buffer

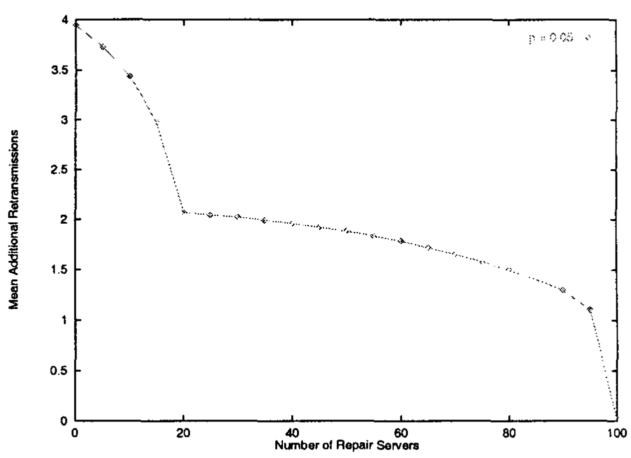

Figure 6: Mean Additional Retransmissions vs Number of Repair Servers

Figure 5 shows the dependence of the mean additional retransmissions from the sender on the buffer size at the repair servers. Here, the number of repair servers, $R = 100$, the mean arrival rate $\lambda = 1/\delta = 128$pkts/sec, $\delta' = 40$ms, and number of receivers per repair server, $k = 10$. The loss probability between the sender and the backbone, p_s is set to 0.05. When $p = 0.05$, 23 buffers are required at each repair server to ensure that less than 0.01 additional retransmissions are sent from the sender. Beyond a minimum number of buffers, the number of additional retransmissions from the sender falls quickly as the buffer size at each repair server increases. This is because the mean number of additional retransmissions from the sender depends upon factors such as $(1 - p^{cB})$ where c is a constant.

We also use (5) to study the dependence of the mean additional retransmissions required from the sender on the number of repair servers. Since we are interested in looking at the repair server population, we make the best case assumption that a repair server, if there is one in a domain, has enough buffers so that buffering is not a constraint. We set the buffer size to infinity in (5) for loss domains that have repair servers and to zero for the loss domains that do not have repair servers.

Figure 6 shows the dependence of the mean additional retransmissions from the sender on the number of repair servers. Here, there are 100 loss domains, with 20% of the loss domains experiencing 25% loss and 80% of the loss domains experiencing 1% loss at each link from backbone to a receiver. We populate the higher loss domains with repair servers first. In other words, the first 20 repair servers are placed in the high loss domains. We observe that the mean additional number of sender retransmissions can be reduced by about 50% (from 4 to 2) by populating only the high loss domains with repair servers. A more interesting observation is that almost *all* of the remaining 80% loss domains must be populated with repair servers before we see any additional marked reduction in the mean number of retransmissions from the sender. This is because for homogeneous loss domains, the mean number of additional transmissions from the sender varies as $log(R - r)$ where R is the maximum number of loss domains and r is the number of loss domains that have a repair server. A similar behavior is observed for other values of the system parameters as well. These results suggest the obvious fact that repair servers should be placed in high loss domains first. Interestingly, they also suggest that when loss domains are homogeneous, almost all domains need to have a repair server before we see any marked performance improvement in end-system throughput and network bandwidth usage due to local recovery.

6. RETRANSMISSION BUFFER REPLACEMENT POLICIES

In the previous two sections we studied the impact that finite buffers have on performance when old packets occupying the buffer at the repair server are replaced by newly arriving packets in a FIFO manner. The problem with this policy is that when there are insufficient buffers, it is possible for a packet buffered at a repair server to be replaced before it gets a chance to be sent as a repair. In such a situation, buffering a packet at a repair server serves no purpose, since lost packets must then be recovered from the upper-level repair server. In this section we propose a new buffer management policy, FIFO-MH, with the goal of making the best use of the finite buffer space, however small, at the repair servers. We study how this policy compares with a simple FIFO buffer replacement policy. We also compare the performance of the simple FIFO policy and LRU (least recently used) buffer replacement policy. We focus on a single repair server and use the mean number of additional retransmissions due to finite buffer space at the repair server, from an upper-level repair server or sender, as a measure for studying and comparing the performance of the different buffer replacement policies.

6.1 FIFO with Minimum Holding Time

We propose a new buffer management policy, called FIFO with minimum holding time (or FIFO-MH in short), in which packets are replaced in a FIFO manner but each packet is held in its buffer for a specified minimum amount of time. This minimum amount of time is called the ***buffer holding time.*** Our heuristic for choosing a buffer holding time is that *a packet should be buffered for at least one* *retransmission interval before it can be replaced.* In other words, the buffer holding time should be at least one retransmission interval (as defined in Section 4). This heuristic is based on the intuition that if a packet is buffered for some finite time, it is better to keep it for at least one retransmission interval, since this is the minimum amount of time needed to determine if a retransmission of that packet will be needed.

In order to use the above heuristic when the retransmission interval is variable, the repair server must maintain a running estimate of the retransmission interval. At any time, the buffer holding time is set to the most current estimate of the retransmission interval. More generally, our FIFO–MH policy can be described as follows.

1. *A packet occupying a buffer is marked at the time of its first retransmission.*
2. *When a new packet arrives and there are no free buffers, if the oldest packet in the repair server buffer is marked or has been in the buffer for more than the buffer holding time, then the oldest packet is replaced by the new packet otherwise, the new packet is discarded.*

We study the FIFO–MH policy through simulation under the assumption that packets arrive at the repair server according to a Poisson process and for the cases of constant retransmission intervals, retransmission intervals from known distributions, and retransmission intervals based on measurements taken from the Internet. Our simulation model is identical to the analytical model used in Section 4. The only differences are that the buffer replacement policy now is FIFO–MH instead of FIFO and the retransmission interval, δ', is not necessarily a constant. To compare the performance of FIFO–MH with FIFO, we also simulate the FIFO policy for the above cases.

6.1.1 Constant Retransmission Interval

When the retransmission interval is constant the buffer holding time time under FIFO–MH is simply set to the value of the retransmission interval which is either known to the repair server or can be easily estimated by the repair server. Figure 7 shows the benefit of using FIFO–MH over FIFO for three different arrival rates, when the retransmission interval is 40ms, $p = 0.05$ and $k = 10$. We observe that the mean number of additional retransmissions under FIFO–MH is less than that under FIFO. When the buffer size is 5 and arrival rate is 128pkts/sec, the mean additional retransmissions under FIFO–MH is 55% of the mean additional retransmissions under FIFO.

If we increase the arrival rate or reduce δ, and, increase B while keeping $B\delta$ fixed, we find that FIFO–MH performs better for higher values of the arrival rate and larger number of buffers. Figure 7 shows the higher performance of FIFO–MH in comparison to simple FIFO when the arrival rate is increased to 256pkts/sec and to 512pkts/sec. When the buffer size is 10 and the arrival rate is 256pkts/sec the mean number of additional retransmissions under FIFO–MH is 45% of the mean number of additional retransmissions under FIFO. This is because, under FIFO–MH, when

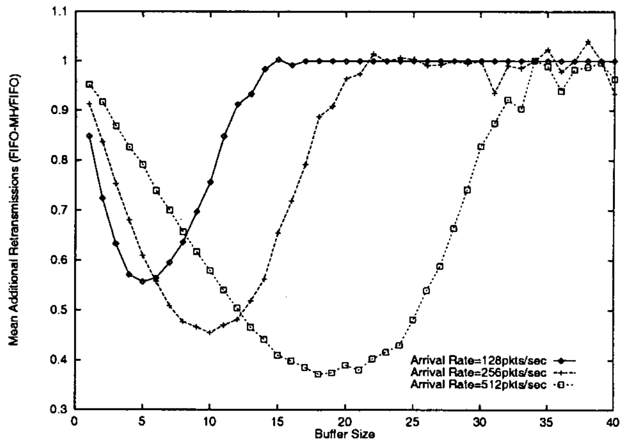

Figure 7: Mean Additional Retransmissions Ratio vs Buffer Size

the arrival rate is not very high and the buffer size is small, a packet might be held in its buffer for more than one retransmission interval (but less than two retransmission intervals). This longer holding time wastes the buffer for that amount of time. When the arrival rate and the buffer size are doubled the holding time beyond one retransmission interval is reduced.

Figure 7 also shows that the difference in the mean number of additional retransmissions under FIFO–MH and FIFO reduces as the number of buffers increases. This is because when the number of buffers is large enough to allow each packet to be held for at least one retransmission interval, there is no difference between FIFO and FIFO–MH.

Note that holding the packet in the buffer for at least one retransmission interval is not necessarily optimal. For example, when the number of buffers is large enough that each packet is always present in its buffer for at least one retransmission interval before being replaced and when the probability of the receivers requiring more than one retransmission from the repair server is high, then higher performance might be obtained if the buffer holding time is two or more retransmission intervals. The problem of deciding when to increase the buffer holding time to two or more retransmission intervals is an interesting topic for future work.

6.1.2 Variable Retransmission Interval

We now consider two cases when the retransmission interval is variable. In the first case, the retransmission interval has a known distribution. In the second case the distribution is derived empirically from round trip time measurements.

- **Retransmission Interval with Exponential Variation**

We represent the retransmission interval as the sum of a constant delay and an exponentially distributed random variable. Let Y be a random variable denoting the retransmission interval, c be the constant minimum delay and Z be an exponentially distributed random variable. We have $Y = c + Z$. Here c models the round trip propagation delay.

We first determine what statistic (including mean, mode,

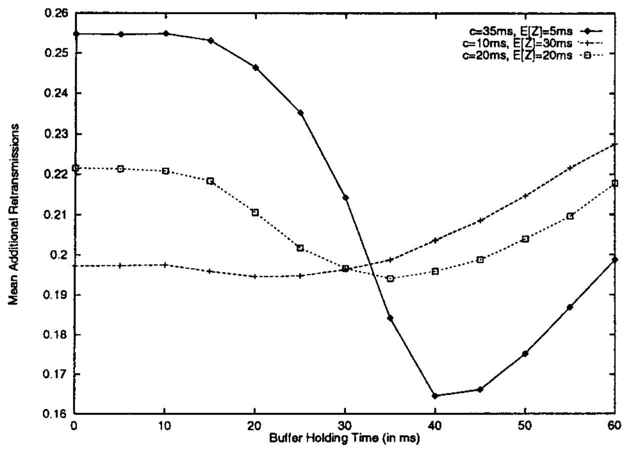

Figure 8: Exponentially distributed Z

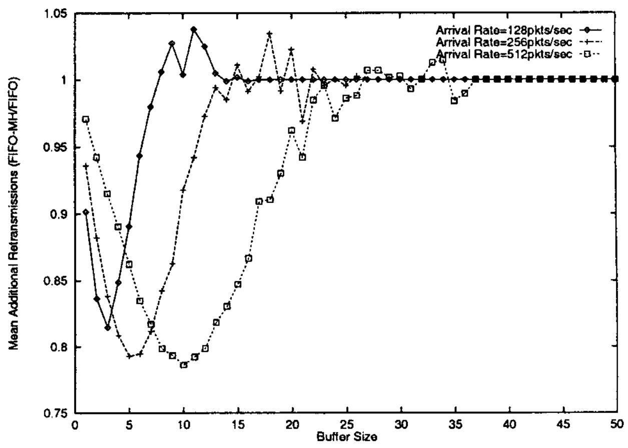

Figure 9: FIFO vs FIFO–MH

median) of Y best allows the repair server to determine a good buffer holding time under FIFO–MH. The repair server computes and estimate of this statistic and sets the buffer holding time to this estimate. We then compare the performance of FIFO–MH and FIFO. Note that we are not proposing any repair server algorithms for estimation of the retransmission interval.

In order to determine a suitable statistic of Y that can be used as the buffer holding time, we compute the mean number of additional retransmissions under FIFO–MH by setting the buffer holding time to constant values that are taken from the range of values of Y.

Figure 8 shows how the mean number of additional retransmissions varies as the buffer holding time increases. Here, the arrival rate is 128pkts/sec, $B = 5$, $p = 0.05$ and $k = 10$. $E[Z]$ and c are so chosen that $E[Y]$ is always 40. We observe from the figure that we can operate FIFO–MH at any buffer holding time as long as it is greater than or equal to c but not much higher than $c + E[Z]$. The same behavior is observed for other values of the arrival rate, B, p, k and $E[Y]$. Some of the candidates for the buffer holding time are mode(Y), median(Y) and $E[Y]$. We observe that median(Y) is most suitable for all the three curves shown in Figure 8 (median(Y) = 38.5, mode(Y) = 35, $E[Y] = 40$, when $c = 35$ and $E[Z] = 5$; median(Y) = 34, mode(Y) = 20, $E[Y] = 40$ when $c = 20$ and $E[Z] = 20$; median(Y) = 31, mode(Y) = 5, $E[Y] = 40$ when $c = 5$ and $E[Z] = 35$).

We now compare the performance of FIFO–MH and FIFO by setting the buffer holding time to median(Y) under FIFO–MH. Figure 9 shows how the ratio of mean number of additional retransmissions under FIFO–MH to that under FIFO varies with the number of buffers when $c = 20$ms, $E[Z] = 20$ms and the buffer holding time is median(Y) $\approx$ 34ms. We observe that the mean number of additional retransmissions under FIFO–MH can at best be about 80% of the mean number of additional retransmissions under FIFO even for high data rates. This reduction in mean number of additional retransmissions is much less than the reduction observed when the retransmission interval is a constant (and set to 40 which is same as $E[Y]$).

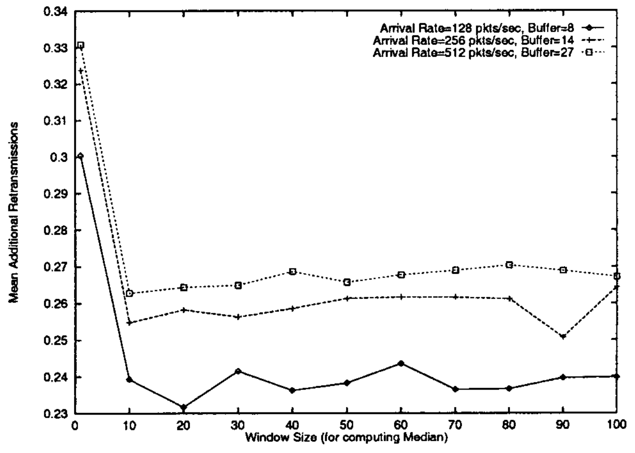

Figure 10: Mean Additional Retransmissions vs Window Size

- **Retransmission Interval from Round Trip Time Traces**

In order to obtain retransmission intervals that reflect the variation in round trip times from repair servers to receivers, we collected round trip time from traces of Internet delay. We used these traces in conjunction with an idealized NAK suppression mechanism at the receivers to generate retransmission intervals. In our NAK suppression mechanism, among the receivers that lose a packet, only the receiver with the smallest round trip time from the repair server sends a NAK for that packet to the repair server. In this paper, we do not worry about how such a NAK suppression can be achieved.

We collected round trip times from a host at UMass (repair server), to three hosts (receivers) at MIT, CMU and Georgia Tech by running simultaneous *ping* commands. Each trace consists of over 12,000 samples. The results in this section are based on these measurements.

Using these traces, we simulate the FIFO–MH policy. In the simulation, we use a sliding window estimate of the median of the retransmission intervals. In order to find a right size for the sliding window we find the mean number of ad-

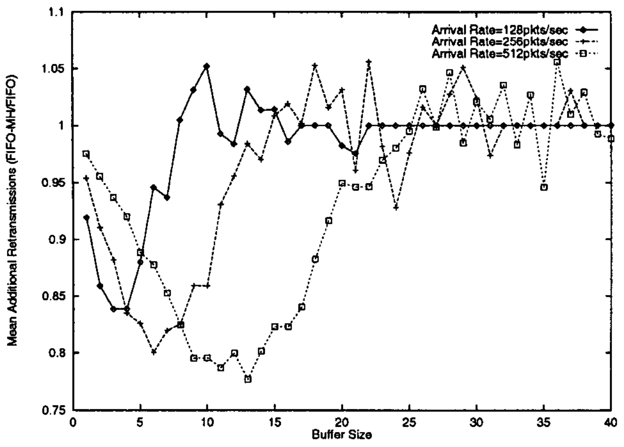

Figure 11: FIFO vs FIFO–MH

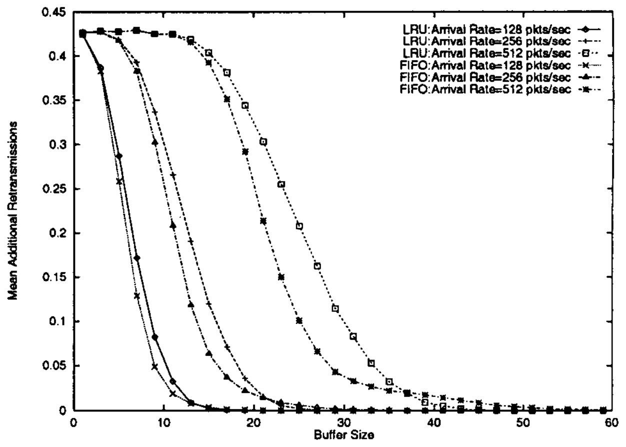

Figure 12: FIFO vs LRU, constant retransmission interval

ditional retransmissions for different window sizes and for different values of packet arrival rates and buffer sizes under FIFO–MH. Figure 10 shows a sample of our experimental results. We observe that a window of size of 10 is appropriate for our traces.

Using a window size of 10, we find the running estimate of the median and use that as the buffer holding time under FIFO–MH and then compare the performance of FIFO–MH with FIFO. Figure 11 shows how the ratio of the mean number of additional retransmissions under FIFO–MH to that under FIFO varies as we increase the buffer size. We observe that the mean number of additional retransmissions under FIFO–MH is about 80% of the mean additional retransmissions under FIFO. This result is similar to the result when the retransmission interval has exponential variation.

In summary, the FIFO–MH buffer replacement policy can be used instead of simple FIFO to prevent premature overwriting of buffers and reduce the mean number of additional retransmissions only when the retransmission interval is constant. If the retransmission intervals are highly variable over a large range of values, then FIFO–MH does not significantly improve performance over FIFO. This is because a good buffer holding time cannot be determined in such a case. When the retransmission interval estimate is large, a large number of the packets will be discarded when the buffers are occupied by packets that are not lost downstream and are not required to be retransmitted. On the other hand, when the retransmission interval estimate is small, the packets (including those that are lost downstream and are required to be retransmitted) occupying the buffers will get replaced prematurely.

6.2 LRU policy

In the LRU policy, when the repair server buffers are all occupied, a new packet, on arrival at the repair server, replaces the packet that was least recently accessed. The LRU policy gives priority to packets that are retransmitted over those that are holding the buffers but are not "in demand." LRU is useful when the following two conditions are met.

- Packets are held in the buffer for sufficiently long time such that those packets whose retransmissions have not been requested can be assumed to have been successfully received and can be safely replaced.
- Lost packets require several retransmissions before they are successfully received by all receivers.

Figure 12 compares the performance of LRU with FIFO when the retransmission interval is a constant. In this figure $k = 10$, $p = 0.05$ and the retransmission interval is 40ms. We observe that when the number of buffers is small, the mean number of additional retransmissions under LRU is equal to or greater than the mean number of additional retransmissions under FIFO. LRU begins to perform better when the number of buffers is increase beyond a certain value. When the arrival rate is 512pkts/sec, LRU incurs lower mean number of additional retransmissions in comparison to FIFO when the buffer size is greater than 37. A similar behavior is observed, as shown in Figure 13, when we generate the retransmission intervals from our round trip time traces. From our observations we conclude that LRU cannot help in reducing mean additional retransmissions or improving buffer utilization over FIFO when the number of buffers is small.

7. CONCLUSIONS

In this paper, we have studied the buffer requirements and buffer replacement policies at a repair server for providing reliable multicast. We used analytical models to show how repair server buffer requirements depend upon the packet arrival rate at the repair server, the required number of retransmissions, and the duration between retransmissions of a packet. We found that buffer requirements are very sensitive to small values of the number of retransmissions of a packet within the packet inter–arrival time but increase very slowly with the number of receivers a repair server is responsible for supplying repairs.

We determined the effect of the number of repair servers and buffer size at a repair server on the end–system throughput and network bandwidth usage. We examined three buffer

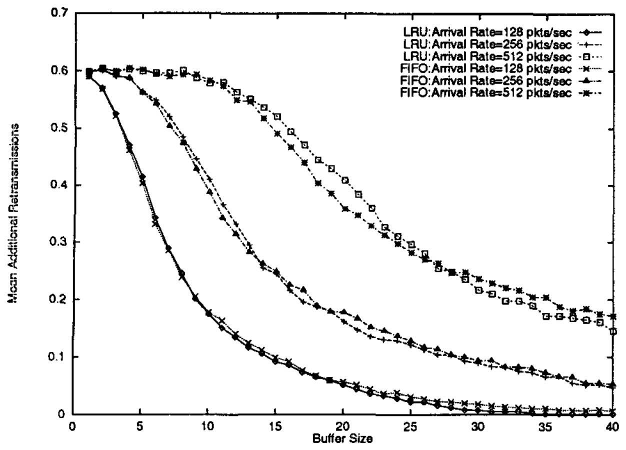

Figure 13: FIFO vs LRU, retransmission interval from measurements

replacement policies; FIFO, FIFO–MH (that we proposed) and LRU. We showed that a simple FIFO buffer replacement policy performs nearly as well as FIFO–MH and even better than LRU when the number of buffers is small.

Our work can proceed in several directions. We have studied the buffer requirements of one multicast session in isolation. An extremely important future work will be to study the buffer requirements when several multicast sessions share the buffer resources at the repair server. It would also be interesting to extend our buffer study to other applications such as web caching to determine the memory requirements at web caches. In our work we have found how the buffer requirements depend on the various system parameters such as upstream retransmissions, loss probability, packet arrival rate, retransmission interval and number of receivers. We would like to use the knowledge of these dependencies for studying the placement of repair servers in the multicast tree.

8. REFERENCES

[1] H. Holbrook, S. Singhal, and D. Cheriton. Log-based receiver reliable multicast for distribute interactive simulation. *Proceedings of ACM SIGCOMM Conference*, pages 328–341, August 1995.

[2] S. K. Kasera, J. Kurose, and D. Towsley. A comparison of server-based and receiver-based local recovery approaches for scalable reliable multicast. *Proceedings of IEEE Infocom*, March 1998.

[3] B. Levine and J. Garcia-Luna-Aceves. A comparison of known classes of reliable multicast protocols. *Proceedings of IEEE ICCC*, November 1996.

[4] J. Lin and S. Paul. A reliable multicast transport protocol. *Proceedings of IEEE Infocom*, March 1996.

[5] J. Mahdavi and S. Floyd. Tcp-friendly unicast rate-based flow control. *http://www.psc.edu/networking/papers/tcp_friendly.html*, January 1997.

[6] P. Osland, S. K. Kasera, J. Kurose, and D. Towsley. Dynamic activation and deactivation of repair servers in a multicast tree. *Proceedings of NIK'99*, November 1999.

[7] D. Rubenstein, S. K. Kasera, J. Kurose, and D. Towsley. Improving reliable multicast using active parity encoding services. *Proceedings of IEEE Infocom*, March 1999.

[8] D. Towsley, J. Kurose, and S. Pingali. A comparison of sender-initiated and receiver-initiated reliable multicast protocols. *IEEE Journal of Selected Areas in Communications*, April 1997.

[9] R. Yavatkar, J. Griffioen, and M. Sudan. A reliable dissemination protocol for interactive collaborative applications. *Proceedings of ACM Multimedia*, November 1995.