The design of a reliable multicast protocol with short delay

Jean-Lien C. Wu and Ching-chang Chen
Department of Electronic Engineering
National Taiwan University of Science and Technology
43, Keelong Road, Section 4, Taipei, Taiwan, R.O.C. 106
E-mail: jcw;roy@nlhyper.et.ntust.edu.tw

Abstract

This paper describes the Short Delay Reliable Multicast (SDRM) protocol, a transport protocol for reliable multicast with short delay. In a multicast environment, when a large number of NACKs hits the sender it will incur the feedback implosion problem. Some mechanisms defer the transmission of NACKs, they also defer the receivers to complete their loss recoveries. Without the need of designated representatives and postponing the loss recovery, the SDRM scales well, and is realistic and efficient. It not only reduces the number of NACKs that hit the sender, but also eliminates the delay in emitting the NACKs from the receivers.

Keywords: IP multicast, reliable multicast, feedback implosion.

1 Introduction

Multicasting provides an efficient way to pass copies of a single packet to a potentially large number of receivers. Instead of sending a separate copy of the data to each individual receiver, the sender just sends a single copy to all the receivers. Recently, researches have demonstrated multicasting real-time data, such as real-time audio and video, over Internet using the multicast backbone (MBone) [1][7]. The distribution of data over Internet, an unreliable network, does not guarantee reliable delivery. Most real-time applications can tolerate some data loss but not the delay, however, some applications require a reliable multicast service, which transports an error-free information to a group of recipients. It is the prime requirement for several important applications that include software updates, whiteboard conference, and distribution of billing data. They require an error-free reliable multicast protocol to disseminate data from a sender to a group of receivers. The importance of this type of service will increase in the future when the network environment becomes more popular.

While many researches emphasize multicast routing algorithms [4][5], the design of a reliable multicast transport protocol in broadband packet-switched networks has received attentions only recently [2][8][10][13][14][15]. In order to achieve a reliable multicast, some mechanisms are needed to ensure the data delivery between senders and receivers. There are two basic categories in reliable multicast, they are the sender-initiated ACK-based protocol and the receiver-initiated NACK-based protocol [17][18]. In the sender-initiated approach, the sender is responsible for providing reliable multicast. It must continuously track both the state information on all receivers and the changing set which it is multicasting. This is accomplished by having the receivers return positive acknowledgements (ACKs) for every correctly received data packet, and having the sender use timers to detect potential packet losses. As the number of receivers grows large, the number of ACKs sent by receivers increases. This may incur the feedback implosion problem on the sender and result in network link congestion.

In the receiver-initiated approach, the receivers have the most responsibility for reliable data delivered to them. Each receiver is responsible for detecting its own packet loss and informing the sender via negative acknowledgements (NACKs) that the retransmission of a packet is required. Receiver-initiated protocols solve the implosion problem by generating much fewer NACKs instead of ACKs. It has been observed the superiority of receiver-initiated multicast protocol over sender-initiated approach [17][18]. Although the responsibility of maintaining reliable delivery is shared among the receivers, the sender will encounter the NACK-implosion problem if a loss occurs at the location close to the sender.

To solve the feedback implosion problem, there are two classes of schemes: the structure-based and the timer-based schemes. In some structure-based schemes, intermediate nodes are used to process and combine feedback information [10][11]. Other structure-based schemes use designated receivers (DRs) as a representative to perform local retransmissions. [6][14][15] The timer-based schemes do not rely on network nodes [3][8][9], rather, they use delayed NACKs to avoid an implosion.

Existing schemes provide only partial solutions to the above problems. Floyd *et. al.* proposed the SRM (Scalable Reliable Multicast) [8] protocol which relies on the topology information to set its timer values. Whenever a host detects a lost packet, it schedules a request for a random time in the future. When the request timer expires, the host multicasts a request for the missing data, then doubles the request timer to wait for the repair. Once any other host (which may be the original source) receives a request and it is capable of answering, it sets its repair timer in the future. When the repair timer goes off, the host multicasts a repair for retransmission. The scheduled packets on each node will be suppressed or canceled if the node receives identical packets generated by other hosts; this is *suppression*. Thus, SRM provides a good solution to alleviate the NACK implosion problem. It distributes loss recovery to all members in the group, and is robust with respect to changes in group membership or topology. However, its timer-based implosion control mechanism increases recovery latency.

Hierarchical schemes used in [6][10], or the RMTP (Reliable Multicast Transport Protocol) [15], provide only approximate solutions to scoped recovery. Moreover, they are less fault-tolerant and robust to topology changes, because they rely on designated receivers or loggers to perform retransmissions. ARM (Active Reliable Multicast)

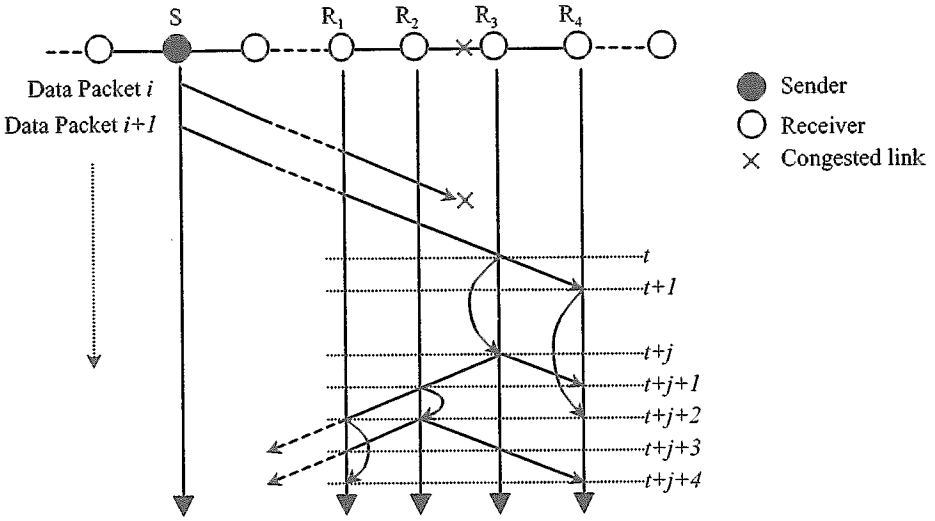


Figure 1 The suppression mechanism used in SRM

[11] depends on Activate Network Technology [16], although it does not require all intermediate nodes be active nodes, which are important roles that perform customized computation on multicast data packet type and caching packets. Further, active nodes provide a fixed amount of "best-effort" soft-state storage that improves the multicast performance.

In this paper, we present a reliable multicast protocol, the short delay reliable multicast (SDRM), that achieves reliability, and maintains low end-to-end delay. SDRM is based on the group delivery model, which is the centerpiece of the IP multicast protocol. In IP multicast, the senders simply send data to the group's multicast address without the need of any knowledge of the group membership. To receive any data sent to the group, receivers simply announce that they are interested—no knowledge of the group membership or active senders is required. The processing of forwarding and duplicating data packets are done at the intermediate nodes, routers, via IGMP (Internet Group Management Protocol) [4]. Our work differs from previous work on reliable multicast in some significant ways. First, SDRM allows receivers to send out NACK immediately upon detecting a loss. This eliminates unnecessary delay. Second, SDRM neither places too much responsibility on intermediate nodes, nor requires intermediate nodes to cache the packets for future retransmission.

The rest of this paper is organized as follows. Section 2 describes the suppression mechanism used in SRM. Section 3 discusses the network model and the assumptions made in the design of SDRM. Section 4 describes our protocol in detail. Simulations and results are provided in Section 5. The last section gives our conclusions.

2 The Suppression Mechanism used in SRM

SRM uses the suppression mechanism to reduce the number of request/repair packets generated due to a lost packet. In this section, the basic concept on top of a chain network topology is used to describe the suppression mechanism. Figure 1 shows a chain topology that has many multicast members, receivers and a sender, attached to it. All members are connected in a chain. Each link has distance in time unit of 1. The random delays used in SRM are D_1 and D_2 for scheduled request and repair packets, respectively, where D_1 is the distance from the sender S to receiver R_x that detects the loss, and D_2 is the distance between R_x and the receiver that is capable of sending a repair. In Figure 1 the distance D_1 between S and R_3 is assumed to be *j*.

Let us consider the situation that data packet *i* is lost at the link between R_2 and R_3. R_3 and R_4 will detect the loss when subsequent data packet *i+1* arrives at time *t* and *t+1* on R_3 and R_4, respectively. Since the distance between S and R_3 is *j*, thus R_3 schedules its request at time *t+j*. The distance D_1 from S to R_4 is *j+1*, therefore, R_4 schedules its request at time *(t+1)+(j+1)*. Once the scheduled time of R_3 expires, R_3 will multicast its request to the network, and all the members on the network will receive this request. R_2, R_1, and S are capable of sending repairs, and the respective distances D_2 from R_3 are to be 1, 2, and *j*. Therefore, they schedule their repairs at time *(t+j)+1+1*, *(t+j)+2+2*, and *(t+j)+j+j*, respectively. While R_1, R_2, and S schedule their repairs in the future, R_4 receives the request sent by R_3 at time *t+j+1* and then cancels its request scheduled originally at time *t+j+2*. This is the request suppression phase. When the repair timer goes off on R_2 at *t+j+2*, it sends out the repair immediately. Both S and R_1 cancel their scheduled repairs, and R_3 and R_4 receive the repair at time *t+j+3* and *t+j+4*, respectively. This is the repair suppression phase. The loss recovery is done here and thereby the loss recovery latencies are *j+3* for both R_3 and R_4.

From the above illustration, there is only one repair and one request sent to the network. However, it is an ideal situation; all nodes are connected in a chain, and their D_1 and D_2 are deterministic times. In a star or tree topology, the random delay time is chosen from a distribution function $[C_1D_x, (C_1+C_2)D_x]$; where D_x are D_1 and D_2, C_1 is an amplification, and C_2 is the random space. Thus, it is possible to have more than two repairs/requests sent to the network simultaneously.

3 Network Model and Assumptions

We assume that the network provides IP-multicast style multicast routing, e.g. MOSPF (Multicast Open Shortest Path First) [12], in which a tree rooted at the sender is formed to deliver multicast packets. In particular, we

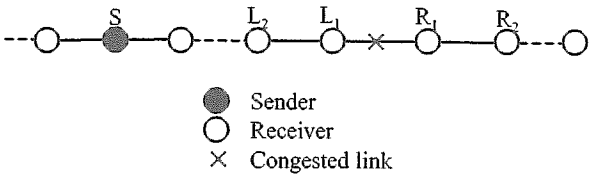


Figure 2 A chain topology

assume that the underlying network is unreliable, and that the packets can be lost, duplicated, or delayed. The network model we considered is similar to traditional packet networks. SDRM uses the receiver-initiated NACK approach; receivers are responsible for the reliable transport of data packets. Although we target for reliable data transfer applications, the protocol can also be applied to real-time applications like audio and video. The following are assumptions made in the protocol design.

Network topology — The underlying network is similar to Internet. On the intermediate nodes, the tributaries of a multicast tree are maintained as long as there are members in the downstream links. Each tributary link connects either another intermediate node or a receiver. Packets take one unit of time to travel on each link, i.e. all links have distance of 1.

IP multicast — A multicast tree is set up using Internet multicast protocol, with the sender located at the root of the multicast tree. All members are both senders and receivers, i.e. sender and receiver are not distinct in capability. Membership is handled by IGMP. Hosts can independently join or leave the group at any time. Whenever a member generates new data, the data is multicast to the group.

Data packets are unique — We assume that all data packets have a unique identifier, e.g. sequence number. Members in the group know the identifier well. Packets are sent from the sender in sequence. The out-of-order packets will be held while asking the repair for the right packet so that the packets passed to the upper layer are in sequence.

Members trust each other — All receivers are able to send repair packets for other members if it is capable to do so, and all members keep a window size of cached data.

4 The Short Delay Reliable Multicast Protocol

SDRM is a receiver-initiated NACK-based scheme in which receivers are responsible for the detection and request for lost packets. Because all data, request, and repair packets are multicast to the group, the main goal of the SDRM algorithm is to achieve the shortest loss recovery while keeping the number of request and/or repair packets low. It allows receivers to send their request/repair out without any delay.

The design of SDRM is based on a tree topology that combines aspects of both chains and stars. For a chain, the nodes receive packets in the order according to the distance to the sender. Figure 2 shows a chain topology where all nodes in the chain are members of the multicast session. In a normal situation, node S sends packets to members, the receivers receive packets in the sequence of L_2, L_1, R_1, and then R_2. Once a data packet is lost or corrupted at the congested link, receivers, L_1 and L_2, on the sender side of the congested link will not be affected, and all the receivers at the far end from the sender will miss the same data packet. When subsequent data packet is sent along the path,

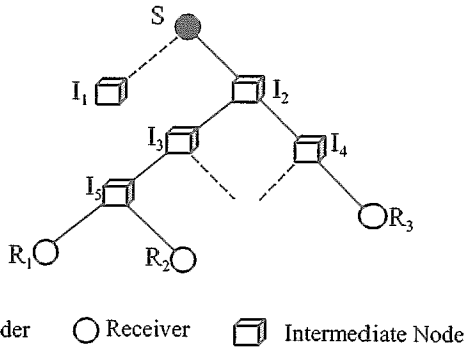


Figure 3 A tree topology

R_1 receives it before R_2 does, and R_1 will detect the loss before R_2 does.

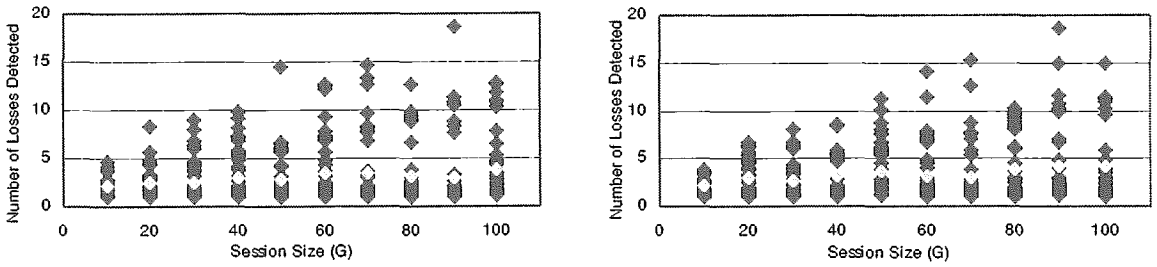
In Figure 2, while node R_2 detects a lost packet, it is only necessary to ask for a repair packet from its upstream adjacent node R_1. Because firstly, if R_1 also lost the same data packet, it requests a repair from L_1 before R_2 does absolutely, secondly, if R_1 has the packet that node R_2 needs, R_1 may send it as a repair packet to R_2. Because multiple hosts may detect the same losses and may handle the same repair request, the main issue of SDRM is to limit the multicast scope. Hence, besides forwarding and duplicating data packets, intermediate nodes in the SDRM algorithm have to filter out the duplicated request/repair packets. This is the key concept in reducing the unnecessary packets that arrive at (hit) the sender—thereby avoiding the feedback implosion problem and improving the throughput of reliable data delivery.

We now turn to the tree topology that is more realistic for Internet that the SDRM focuses on. In addition to the sender and the receivers, there are intermediate nodes, e.g. routers, in its multicast tree, which run the IGMP protocol. In Figure 3, the solid lines connect all members and intermediate nodes to construct a multicast tree. Dash lines denote that there are no group membership existing in such downstream links, and they are not the tributaries of the multicast tree. We will use (X, Y) as a notation for a link between nodes X and Y in the following discussion.

In order to limit the multicast scope, there are some mechanisms used in the design of SDRM:

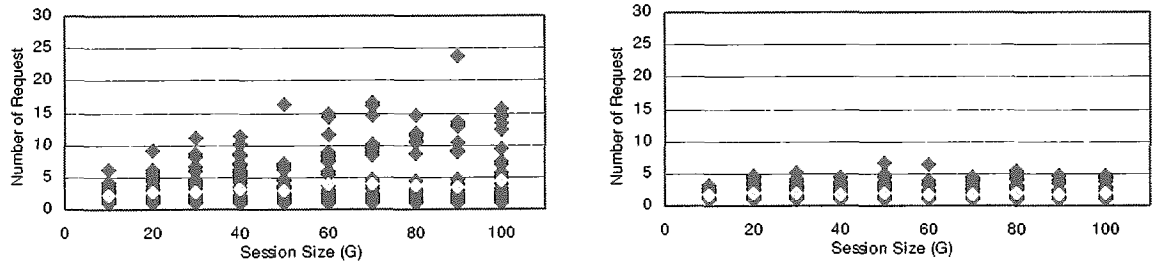
- The field TTM (time-to-multicast) is embedded in the request packet.
- Intermediate node must keep track of the number of multicast tributaries on it.
- Intermediate node has the capability of suppressing subsequent identical packets.
- The adjustment of TTM is needed to grant necessary repair.

TTM simulates the TTL (time-to-live) field used in an IP packet. Normally, TTL is set to be zero for a local broadcast. TTL will be decreased by one by each hop if it is greater than zero. We set TTM to be 2 for a request packet, but set it to be a larger value (may be the maximum depth of the multicast tree) for data and repair packets. TTM is decreased by one only on an intermediate node with three tributaries of a multicast tree. For example, I_2 and I_5 will decrease TTM, but I_3 and I_4 will not. Initially, TTM is set to be 2. The receiver sends a request packet out, and waits



(a) SDRM (b) SRM

Figure 4 The number of losses detected by receivers.



(a) SDRM (b) SRM

Figure 5 The number of requests generated by the receivers that detected a lost packet.

for a RTT time. If none of repair packet is received, the request scope is expanded by increasing the TTM value. Then, the request packet will be sent again until get a repair packet back. In the worst case, the sender itself may have to respond to a request. The TTM value is maintained in receiver's local variable for future use during a multicast session.

Now, let us consider a situation that congestion occurs on (I_5, R_1). R_1 sends a request packet with TTM of 2. I_5 will decrease TTM to 1, and then forwards this packet along the links (I_5, I_3) and (I_5, R_2). R_2 then may send a repair packet back, and I_3 will forward it to I_2 without any modification. When this request packet reaches I_2, its TTM is 1. I_2 will decrease TTM to 0, and then forwards this packet to S and I_4 in turn. Finally, both S and R_3 will send their repair packets back. It seems a lot of repair packets are flooding over the multicast tree, because the multicast tree is small in Figure 3. In Section 5, we will demonstrate that the number of request packets could be limited effectively as the multicast group grows up.

The second situation for consideration, congestion occurs at link (I_5, I_3). Both R_1 and R_2 will send their own request packet independently. I_5 processes only one request packet sent by either R_1 or R_2. No matter which packet I_5 processed, the subsequent request packet will be dropped. Hence, only one request is forwarded to I_3 exactly. The rest of the forwarding paths are the same as the previous example. After S and R_3 receive the request packet, they will send their repair packets back along the multicast tree. However, only the repair packet of S is forwarded, the repair packet of R_3 will be ignored. Because the repair packet from S will arrive at I_2 before the repair packet sent by R_3. Therefore, only one request packet and one repair packet travel along each link.

5 Simulation and Results

In this section, we simulate the SDRM protocol to evaluate its performance, to measure the tradeoffs it has between the recovery latency and the number of packets affect the sender, and compare it with those of SRM.

To compare SDRM with SRM directly, the same assumptions about the network are made as Floyd *et. al.* did in their analysis of the SRM [8]. The network is a balanced bounded-degree tree of N=1000 nodes, with interior nodes of degree four. In these simulations the session size G is significantly less than N, that is, not all the hosts are session members. The SRM simulations examine loss recovery behavior on a per-loss basis; they do not consider scenarios in which requests and repairs are lost in addition to fresh data packet.

In the simulation, we randomly choose G of the N nodes as multicast group members (G varies from 10 to 100) for each session. Fifty simulations were run for each value of G. For each simulation, a new random tree was constructed. A source S was randomly chosen from among the G members. With respect to each simulation, ten experiments were run, and a congested multicast path was randomly selected for each experiment. Some network behavior and the measurements are described in the following.

A. Simulation model

- Single data source

In each experiment, there is only a selected sender sending fresh data packets. Nevertheless, multiple receivers could send request/repair packets simultaneously. The sender is a member of the multicast group.

- No loss in requests and repairs

To simplify the circumstance and make the measurement more precise, only fresh data packets will be dropped during a transmission phase. Request and repair packets are transferred smoothly.

- Instantaneous link congestion

The congested link is used to cause a data packet dropped.

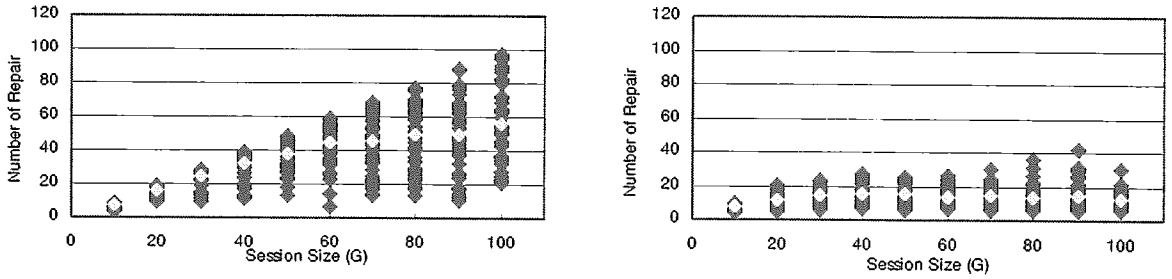


Figure 6 The number of repair packets sent by the receivers that had capability to perform a retransmission.

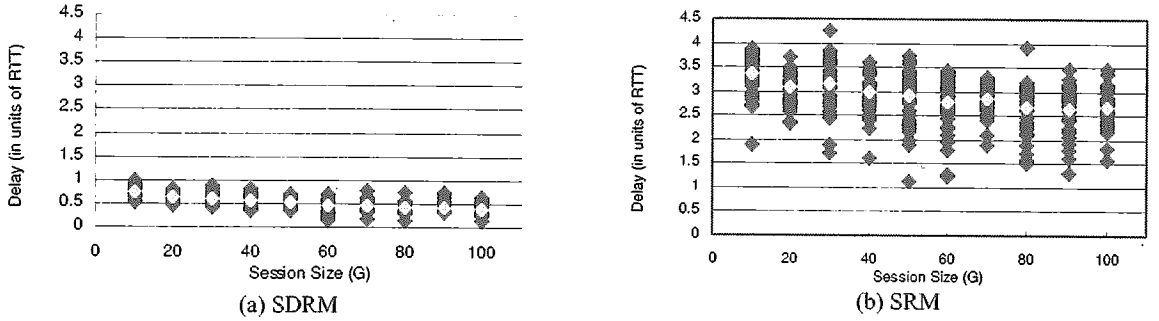


Figure 7 The loss recovery latency

After dropping a data packet, the congested link is resumed immediately. The following requests and/or repairs may be forwarded through the link with a loss probability of zero.

- Single link congestion

Only one link is chosen randomly from the multicast tree, i.e. multiple losses will not occur in our experiments.

B. Measurements

For the simulations on a bounded-degree tree, Figures 4 to 9 show the results we measured with different aspects. In each figure the x-axis shows the session size G. Each simulation is represented by a black diamond, and the white diamonds shows the median from fifty simulations. All the measured values for each simulation are the average values for ten experiments.

Figure 4 shows the average number of members that detect the lost packet in each simulation. If a packet is dropped on a link, all members on the downstream side will detect the loss. This figure implies that the behaviors of detecting the lost packet for both SRM and SDRM are similar to each other. The number of members that detect the loss varies, e.g. from 1 to 14 for session size of 100 in SDRM, it depends on the location where the congestion occurred.

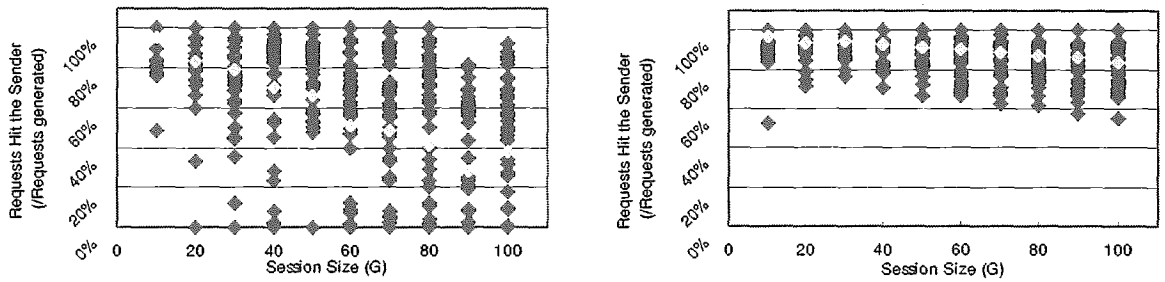
Whenever a lost packet is detected by a receiver, it sends a request packet to the network asking for a retransmission immediately. In Figure 5(a), we find that the number of generated requests is almost the same as the number of losses detected in Figure 4(a), because the receivers generate a request whenever they detect a lost packet in SDRM. However, in Figure 5(b), the number of request packets in SRM is less effected by the session size, because suppression mechanism is applied. Nevertheless, the average number of request packets for each session size in SDRM is low. The network topology used in our simulation is a sparse tree, where any congested link only affects a few hosts.

Figure 6 shows the number of repair packets that was sent to the multicast tree. In SDRM, a receiver multicasts a repair packet immediately if it is capable of retransmitting a data packet, which is required by other receivers. The average number of repair packets generated in SDRM is high, and it grows with the session size. In some cases, it is almost the session size. This is because we do not use suppression mechanisms; all receivers send their requests upon detecting the loss. Since SRM uses suppression mechanisms to reduce most duplicated packets, and thereby some scheduled repair packets were cancelled while they were waiting to send for a random delay time. All members are volunteers to rush on to be the first to help each other without any delay in SDRM.

Figure 7 compares the loss recovery delay of SDRM with that of SRM. The loss recovery latency is significantly lower for SDRM than for SRM. The loss recovery latency is the time that a receiver first detects a loss until it receives the first repair for that loss. In SDRM, the worst case recovery delay is the round trip time (RTT) between the sender and the receiver experiencing the loss. To avoid the implosion problem, SRM uses suppression mechanisms to reduce the unwanted packets. These mechanisms require the receivers to wait for a random time before sending the packets to the network. The longer the waiting period is, the more the identically scheduled packets will be cancelled. This increases the loss recovery latency seriously.

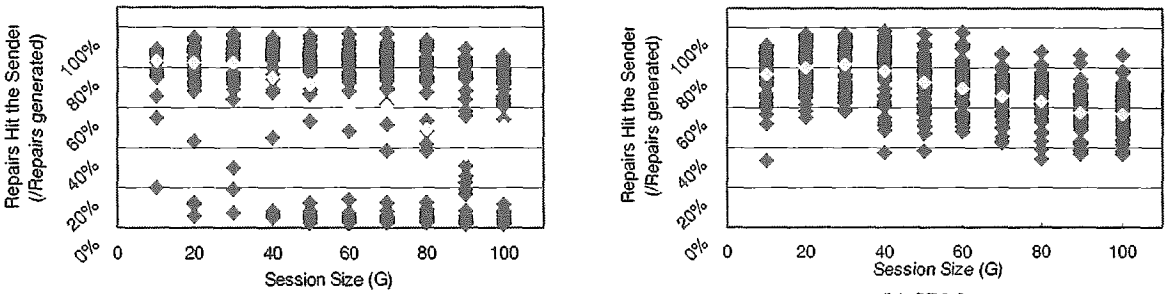
In the suppression mechanism of SRM, scheduled packets are withdrawn before timer goes off. Once the packets are sent to the network, there is no chance to stop its forwarding. In SDRM, the intermediate nodes will drop all the subsequent identical packets. It implies that only a small portion of request packets and repair packets will arrive at the sender even though a large number of request packets and repair packets were sent onto the network.

Figures 8 and 9 show the ratio of request packets and repair packets that arrived at the sender finally. The number of



(a) SDRM (b) SRM

Figure 8 The number of request packets that arrived at the sender.



(a) SDRM (b) SRM

Figure 9 The number of repair packets that arrived at the sender.

request packets and repair packets to their generated packets is calculated as the ratios, respectively. Although a lot of request packets and repair packets are generated, only a portion of them arrived at the sender, whereas most of the generated packets hit the sender in SRM.

6 Conclusions

Despite of the number of request/repair packets generated, the number of packets arrived at the sender is the major cause of the feedback implosion problem. SRM uses suppression mechanisms to reduce the request and repair packets as much as possible. These mechanisms increase the loss recovery latency to approximately three times of RTT. While ARM uses active routers to improve the local recovery and avoid the feedback implosion problem. The recovery latency is reduced to approximately 0.2 RTT, but all intermediate routes have to cache multicast data. Without an active router, recovery latency in ARM is 1 RTT. In general, we believe that SDRM can be applied to any multicast application requiring prompt feedback. SDRM does not require intermediate nodes (routers) to keep the data packet for future repair, but ARM does. If there are many multicast groups existing simultaneously, the intermediate nodes for ARM have to offer a lot of resources, and keep track of the multicast session for each group. It is not easy to achieve.

Combining Figures 5 with 7, we found that although the number of requests in SDRM is higher than that in SRM, only 50% of the requests arrive at the sender. When the session size approaches 100, the average numbers of requests are down to approximate 5, thus only about 2.5 request packets hit the sender. In SRM, there are about two requests that hit the sender. The average recovery latency is only about 0.4 RTT in SDRM and from Figures 7, 8, and 9, the performance for a dense tree is better than a sparse tree.

In this paper, we have shown that SDRM can deal properly with the feedback implosion problem; meanwhile, the recovery delay is significantly shorter than that of previous algorithms and not too much modification are required on the network equipment.

References

[1] K.C. Almeroth and M.H. Ammar, "Multicast group behavior in the Internet's multicast backbone (Mbone)," IEEE Communications Magazine, pp.124-129, June 1997

[2] S . Armstrong, A. Freier, and K. Marzullo, "RFC-1301: Multicast transport protocol," Feb. 1992

[3] J. Crowcroft and K. Paliwoda, "A multicast transport protocol," in *Proc. ACM SIGCOMM'88*, Stanford, CA, pp.247-256, Aug. 1988

[4] S. E. Deering, "RFC-1112: Host extensions for IP multicasting," Aug. 1989

[5] S. E. Deering, and D. R. Cheriton, "Multicast routing in datagram internetworks and extended LANs'," *ACM Trans. Comp. Syst.*, Vol. 8, No. 2, pp. 85-110, May 1990

[6] D. DeLucai and K. Obraczka, "Multicast feedback suppression using representatives," in *Proceeding of IEEE Infocom '97*, Kobe, Japan, pp.464-471, Apr. 1997

[7] H. Eriksson, "MBONE: The multicast backbone," Comm. ACM, Vol. 37, No. 8, pp.54-60, Aug. 1994

[8] S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne, and L.Zhang, "A reliable multicast framework for light-weight sessions and application level framing," *IEEE/ACM Trans. on Networking*, Vol.5, No.6, pp.784-803, Dec. 1997

[9] M. Grossglauser, "Optimal deterministic timeouts for reliable scalable multicast," *IEEE JSAC*, Vol. 15, No. 3, pp.422-433, Apr. 1997

[10] H. W. Holbrook, S. K. Singhal, and D. R. Cheriton, "Log-based receiver-reliable multicast for distributed interactive simulation," in *Proceeding ACM SIGCOMM'95*, Cambridge, MA, pp.328-341, Oct.

1995

[11] L. H. Lehman, S. J. Garland, and D. L. Tennenhouse, "Active reliable multicast," in *Proceeding of IEEE Infocom'98*, San Francisco, CA, pp.581-589, Mar.-Apr. 1998

[12] J. Moy, "Multicast routing extensions," *Comm. of the ACM*, Vol. 37, pp.61-66, Aug. 1994

[13] C. Papadopoulos and G. Parulkar, "Implosion control in multipoint transport proocols," in *Proceeding IEEE Comp. Comm. Workshop*, Sept. 1995

[14] S. Paul, K.K. Sabnani, and D.M. Kristol, "Multicast transport protocols for high speed networks," in *Proceeding Int. Conf. Network Protocols*, Boston, MA, pp.4-14, Oct. 1994

[15] S. Paul, K.K. Sabnani, and J.C. Lin, and S. Bhattacharyya, "Reliable Multicast transport protocol (RMTP)," *IEEE JSAC*, Vol. 15, No. 3, pp.407-421, Apr. 1997

[16] D. Tennenhouse and D. Wetherall, "Towards an active network architecture," *Computer Communication Review*, Vol. 26, pp.5-18, Apr. 1996

[17] D. Towsley, J. Kurose and S. Pingali, "A comparison of sender-initiated and receiver-initiated reliable multicast protocols," *IEEE JSAC*, Vol. 15, No. 3, pp.398-406, Apr. 1997

[18] M. Yamamoto, J.F. Kurose, D.F. Towsley, and H. Ikeda, "A delay analysis of sender-initiated and receiver-initiated multicast protocols," in *Proceeding of IEEE Infocom'97*, Kobe, Japan, pp.481-489, Apr. 1997