Scalable Session Messages in SRM

Puneet Sharma*
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90291
Ph: 310-822-1511 ext. 742
Fax: 310-823-6714
puneet@isi.edu

Deborah Estrin
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90291
Ph: 310-822-1511 ext. 253
Fax: 310-823-6714
estrin@isi.edu

Sally Floyd
Lawrence Berkeley Laboratory
1 Cyclotron Road
Berkeley, CA 94720
Ph: 510-486-7518
Fax: 510-848-7930
floyd@ee.lbl.gov

Lixia Zhang
4531G Boelter Hall
University of California
Los Angeles, CA 90024
Ph: 310-825-2695
Fax: 310-825-2273
lixia@cs.ucla.edu

August 1, 1997

Abstract

Multi-party applications are of great importance, and perhaps the greatest challenge is achieving both a) robustness in terms of adaptation to dynamic topology and group membership, and b) scalability in terms of bandwidth, state, and processing, as the size of a group grows. Scalable Reliable Multicast (SRM) [1] is a rich example of a robust design intended to work across a wide range of group sizes and dynamic topologies. However, the adaptation mechanisms in SRM rely on shared group state achieved via exchange of session messages. Similar synchronization is likely to be of importance in other multiparty applications and services.

Various mechanisms have been proposed to reduce the overhead of this loose group synchronization. This paper applies the concept of self-configuring hierarchy to SRM. Unlike previous proposals, our mechanism uses a stochastic algorithm for self-configuration based on randomized timers and local appropriateness measures. We present initial evaluations of the impact of this mechanism on SRM performance, and evaluate the hierarchical structure formed by the protocol participants. Many interesting questions remain to be investigate in future work such as self-evaluation of appropriateness, dynamics and stability of self-configuration process.

Keywords: Multicast, Reliable Multicast (SRM), Self-configuration
Areas of Interest: Internetworking, Distributed Network Algorithms

*Corresponding Author

1 Introduction

Various multicast transport protocols [1, 2, 3, 4, 5, 6] have been proposed to provide reliability and other transport level functions on top of IP multicast [7]. One such protocol, *Scalable Reliable Multicast* (SRM) [1], is a reliable multicast framework for application level framing and light-weight sessions. SRM takes a receiver-driven approach where each member of the session is responsible individually for detecting loss and requesting retransmission, while all session members may collaborate in the error recovery process to reduce the recovery delay and distribute the load among members. SRM members exchange *session messages* to detect data loss and also to compute the data delivery delay to other members.

Periodic global exchange of session messages among the members scales poorly with the size of the group. Though session messages can be rate limited based on session size, this can degrade the adaptivity for very large groups. We propose to improve SRM scaling properties by limiting the distribution scope of session messages. In this paper we investigate self-configuring hierarchy mechanism applied to session message distribution. We also evaluate the effect of this modified session message distribution on the loss recovery performance of SRM.

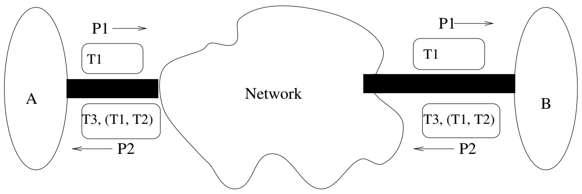
This paper is organized as follows: We begin with an overview of SRM mechanisms and the related scaling problems in the next section. Section 3 describes how scalable session messages address these problems. Section 4 presents the comparison of proposed scheme with original SRM based on simulation results. We conclude with We conclude with a summary and a few comments on future directions in Section 5.

2 Overview of SRM

Scalable Reliable Multicast (SRM) adopts a collaborative approach to error recovery. Though each member detects the loss individually, any receiver that has successfully received the data may initiate a retransmission. The requests for retransmission and the repairs are multicast to the whole group. Randomized timers are used to suppress some of the duplicate requests and repairs for same loss. To reduce the recovery delay it is desirable that the member closest to the lossy link is the first member to send a request. Similarly, the first reply should preferably be sent by the member closest to the lossy link. This desired behavior is achieved by choosing the random timer from a time interval proportional to the propagation delay between the source and requestor (and similarly between requestor and replier). Thus, each member needs to estimate and maintain the distance to every other member.

Two major components of SRM are session message exchange for group state synchronization and loss recovery algorithms. This section discusses salient features of these two components and presents problems associated with scaling SRM to very large groups.

2.1 Session Message Exchange



Session Message Exchange in SRM

Figure 1: Exchange of timestamps for distance estimate in SRM

Scalable Reliable Multicast (SRM) participants exchange session messages to determine the distance to other members. The distance estimation is based on a simplified version of the NTP [8] algorithm. Figure 1 shows one round of session message exchange to estimate the distance between two members. Session message $P1$ is sent by member A at time $T1$. Member B records the time $T2$ when $P1$ arrives at B. Some time later, at time $T3$, member B sends session message $P2$. Along with the timestamp $T3$, message $P2$ also carries the tuple $(T1, T2)$. Member A upon receiving $P2$ at time $T4$ can estimate the one-way distance to member B using the following expression.[1]

$$distance = (T4 - T3 + T2 - T1)/2.$$

Every member multicasts session messages to the whole group. The session messages have timestamps for all the session members. One round of session message exchange is required for estimation of distance between any two members. Since the propagation delay can vary with changes in network conditions such as network load and route changes, session messages are exchanged at regular intervals. As mentioned earlier, the error recovery algorithm of SRM takes these delays as parameters for scheduling *requests* and *repairs* for the lost data packets.

Besides estimating distances, session message exchange synchronizes the state of the group among various members. Each member sends session messages to report its current view of the group state. The current group state consists of the largest sequence number data packet that a member has received from each sender. A member detects missing data segments by comparing the sequence numbers reported in various session messages.

2.2 Loss Recovery

The second component of SRM performs recovery of lost data packets. When a member i detects loss of a packet from a source s it schedules a request. The time for scheduling a request is chosen randomly from a uniform distribution on interval $[C1 * dist(i, s), (C1 + C2) * dist(i, s)]$, where $C1$ and $C2$ are constants and $dist(i, s)$ is the distance of the source from member i. The intervals are backedoff by a factor of 2 after sending a request or receiving a request. A similar algorithm is used for scheduling repairs when a request is received. Any member that has received data can participate in the repair process. Algorithms for adapting the various parameters for scheduling requests and repairs have also been proposed in [1].

2.3 Scaling Issues

The SRM session messages are sent to the whole group at regular intervals. If the session messages are sent at fixed intervals, the bandwidth consumed by global distribution of session messages is proportional to the square of the size of the group. To counter this problem SRM rate limits the session messages to a small fraction (e.g. 5%) of the data bandwidth. SRM uses the same algorithm as RTP (Realtime Transport Protocol) [9] for adapting the session message frequency to the size of the group. As the size of the group grows, each member reduces the frequency of session message transmission. While this keeps the session message overhead low it can lead to very poor response times. The interval between session messages grows linearly with the size of the group. Timely exchange of session messages is particularly important when network topology and session membership dynamics occur [10]. Low frequency session messages result in slow adaptation to the network conditions.

As mentioned earlier, the requests and repairs are scheduled based on the distances to the other members. Thus each member must maintain a table of distances to every other member. The space requirement for this table increases proportionally with the size of the group. The scaling problem is exacerbated by the fact that the size of session messages also increases linearly with the group size.

In summary, global distribution of session messages and storing information about all other members can result in scaling problems. In this paper we investigate the scalability improvement by hierarchically distributing the

[1] This assumes symmetric paths but it does not assume any synchronized clocks

session messages with limited scope. We briefly discuss some of the existing work addressing these scaling issues before presenting our approach.

2.4 Related Work

The scaling problems described in the previous section have been observed for SRM and some other end-to-end multiparty protocols [11]. Some of the approaches that have been proposed are:

1. **Delay Report and Echo** :

   Montgomery [12] suggests a report-echo mechanism for computing the delays between the various members. The members send a delay-report (DR) requesting other members to echo back the information about the delay. The members sends report-echo (DRE) in response to the delay-reports. This approach also uses timestamp exchange mechanism to compute the delays between various members. The DRs are exchanged only when the delay measurement is needed. Members send DRs at join time, after sending a request and after sending a repair. The time between the delay-reports is adapted to the change in size of the group. Each member has to store distance to every other member. Thus this scheme scales better than the original SRM scheme but not for very large groups.

2. **Distance Vector Style Distribution** :

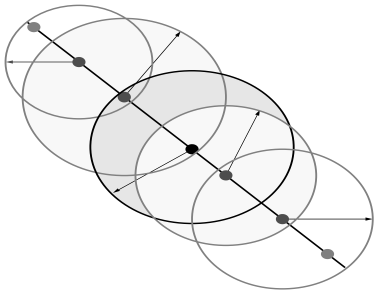


Figure 2: Distribution of SRM session messages using Distance Vector like approach

   An approach similar to the distance vector routing information exchange has been proposed for SRM session messages [13]. The members only exchange session messages with their neighbors and aggregate group states received from their neighbors. Such message exchange requires an overlapped chain of scopes as shown in Figure 2. In order to guarantee an overlapped chain of scopes, a member's session messages have to reach at least one neighbor in each direction. Each member determines its closest downstream and upstream neighbors and sets the scope to the maximum of the two distances. Each member includes its distance to every other member in the session message. A member updates its distances to other members based on the distance information in the session messages that it receives. In this scheme every node still maintains state about all the members. More importantly, similar to the distance vector routing protocols, the distance computation takes a longer time to converge and are dependent on the worst case delay between any two members of the session. We will build on this concept of scoped messages, however, using hierarchy to improve scaling.

3. **Proxy Servers** :

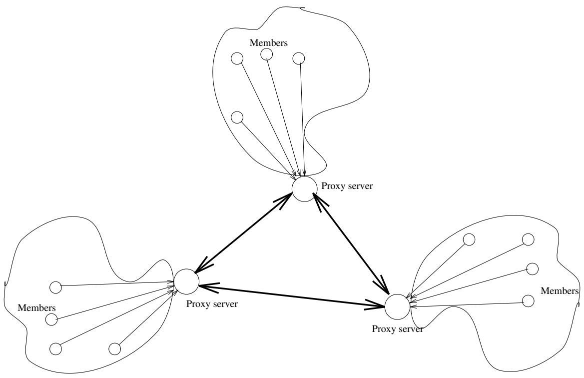


Figure 3: Proxy servers for distribution of session messages

Proxy servers have been used for web caching and other network optimizations. For example, the Session Advertisement Protocol (SAP) [14] proposes uses of proxy servers for advertising on behalf of local members. A similar approach, where the proxy servers distribute the session messages on behalf of members local to it, can be used in SRM. As shown in Figure 3, the members that are close to each other form a cluster and send their session messages to the proxy server that is nearest to the cluster. The proxy server aggregates the messages that it receives from local members and distributes them globally. Proxy servers limit the distribution of session messages to a local region. However, because it relies on static configuration this approach is not efficient for SRM like protocols that have dynamically changing membership and assume underlying topology dynamics [15]. Also it might require non-participating members to be involved in distribution of session messages. Our approach addresses this issue by attempting a similar steady state behavior with a dynamic, and adaptive algorithm.

3 Scalable Session Messages

As observed in Section 2 session message mechanisms do not scale well for very large groups. To improve the scalability of session message distribution, we developed a self-configuration protocol to dynamically organize the participants in a hierarchy and distribute session messages with limited scope. Use of hierarchy reduces [16] the number of members at any level and hence improves the scaling properties of SRM. In particular, distribution of session messages with limited scope tackles the bandwidth usage problem because not all members send session messages with global scope. The scope of a member's session messages is determined by its level in the hierarchy. This scoping of session messages is similar to the scoping of control messages used in Landmark Hierarchy routing protocol [17] and interdomain PIM [18]. In this study we assume a two level hierarchy for SRM session message dissemination, although we believe our approach is extensible to any number of levels in hierarchy, for this study we assume a two level hierarchy for session message dissemination. A top level member is called a *representative* and it distributes global session messages on behalf of members local to it. A member at a lower level is called a *local member* and has a *representative* whose messages it uses to approximate the distances to other members beyond its local scope. While other approaches have suggested the use of hierarchy, our work focuses on self configuration of the hierarchy. Our self-configuration algorithm limits the number of session members sending global session messages to be within an acceptable range. When this number exceeds a threshold, the members automatically configure themselves into a hierarchy of local and global members. Associated with our approach are two kinds of mechanisms, one for organizing the session members into a distribution hierarchy and other for exchanging session messages in accordance with this hierarchy. In the following subsections we describe these mechanisms in detail.

3.1 Session Message Distribution Protocol

In this section we assume a hierarchy of members and describe how the session messages are distributed such that the group state is disseminated and delays between various members can be computed. All the session messages carry the latest sequence number information of all the senders. The distance computation in SRM requires a two way exchange of timestamps among every pair of members. As mentioned earlier, not all members send global session messages. Local members send their session messages with a scope sufficient to reach their representative. These local session messages carry timestamps echoing the session messages heard from other members in the neighborhood. As a result each local member can compute the distance to its representative and other local members in its vicinity.[2]

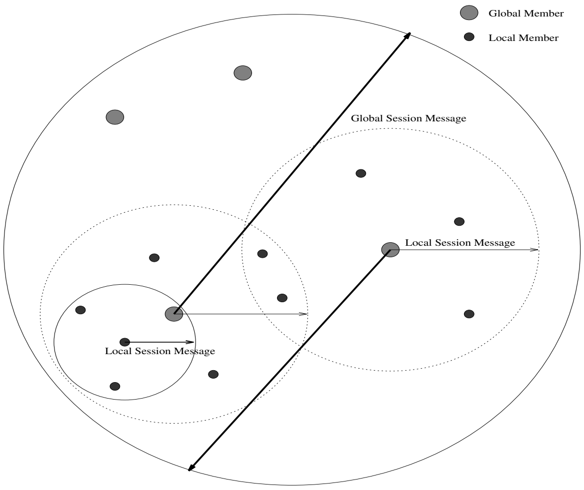


Figure 4: Hierarchical distribution of session messages

The global members, also called *representatives*, send session messages to the whole group. These session messages carry timestamps echoing the session messages heard from other global session messages. Thus, each global member can compute the distance to every other global member. They do not, however, echo the timestamps for local members that are not in their neighborhood. So local members do not have complete information for computing the distance to the global members other than their representative. The local members use the distance between their representative and global members in other areas as to approximate their distance to members in other areas. Hence each global member needs to distribute its distance from other global members. Each global member also sends separate session messages with distance information to its local members. This distribution mechanism is illustrated in Figure 4.

The information about the representative for a particular sender is carried in the global session messages. Each sender can also attach the representative information with the data packets. Similarly the *request* message also carries the information about the representative of the requestor.

Various schemes can be used for approximating distance between any two members that are not exchanging timestamps. Table 1 shows one such scheme for computing the distance from member a to another member b. In this scheme a local member computes a longer distance to a far away member than that computed by its

[2]In some cases, more than one representative might echo the local-session messages. In this case a local member can estimate its distance to either representative.

(a ,b)	b is Local Member	b is Global Member
a is Local Member	$dist(rep_a, rep_b) + dist(a, rep_a)$	$dist(rep_a, b) + dist(a, rep_a)$
a is Global Member	$dist(a, rep_b)$	$dist(a, b)$

where:
members a and b do not exchange timestamps, except when both are global
rep_x is representative of member x
$dist(x, y)$ is the distance between members x and y computed using timestamp exchange.

Table 1: Distance computation matrix

representative. Since the intervals for request and repair timers are proportional to the calculated distances, a representative is more likely to send a repair or request if this scheme is used. Other distance approximation schemes can be used for achieving other desired behaviors.

Our approach requires mechanisms for restricting the distribution scope of session messages. The two possibilities are use of separate multicast groups for each local area, or hop-counts [19].

- Separate multicast groups : In this case, each representative includes the address of the multicast group for the local sub-group in its global session message. A newly-arriving group member obtains the address of the local group that it needs to join from these global messages.
- Hop counts : The representative sends the local session messages with a limited scope of H hops. A new member to the group will automatically receive local session messages from representatives that are nearby.

This initial study uses a TTL based scope control mechanism for our scheme as it is less complex. Future work will investigate the tradeoffs associated with this choice in more depth.

3.2 Self Configuring Hierarchy

Manually configured hierarchies are undesirable in the presence of dynamic membership and topology. The hierarchies should adapt to the changes in the network conditions. In this section we first discuss some of the self-configuring approaches that have been used earlier in computer networks. A number of self-configuration approaches have been used in Packet radio networks to discover appropriate hierarchical structures for network connectivity. As opposed to SRM and other multiparty protocols, Packet radio self-configuration algorithms have been designed for tightly coupled entities. Hence those approaches can not be directly applied to adapting structures in multiparty protocols. Recently, coincident with the development of the work presented here, there have been some proposals for self-organization of members in multiparty protocols [20, 5, 6, 21]. One such protocol, Self-Organizing Multicast, MTP/SO [20] proposes self-organization of the members of a group into local regions for addressing the NACK implosion problem. Each member advertises it reception quality and the member with the best reception quality is elected as a *Repeater* for the local region. The Local Group Concept [21] another example of use of advertisement based approach for dynamically adapting a hierarchy of group controllers. Another multicast transport protocol TMTP [6] also organizes the group members into a hierarchy of subdomains. TMTP members use *expanding ring search*(ERS) to solicit response from potential connection points into the tree. The closest respondent member is selected as the next level parent. A similar ERS based scheme is used by Resident Multicast [22], a continuous media dissemination protocol, for structuring a loss recovery tree. Each member constructs a list of potential parents and selects one of them based on the reported loss rate. In summary, these schemes are based either on advertisement of potential representatives followed by election of representatives from a set of candidates or expanding ring search for soliciting replies from potential parents. We propose a stochastic algorithm for self-configuration of the hierarchy based on randomized timers and appropriateness measures. The algorithm constructs and maintains the hierarchy for distribution of session messages as described in previous subsection.

We designed self-configuring mechanism with the following constraints in mind. First, the mechanisms for choosing representatives for session messages have to take into account the dynamic nature of session membership. There might never be a completely stable membership. Also the whole multicast group does not come into existence all at once with a fixed set of members, but is built up one by one as members join the multicast group. All of this argues for self-selection rather than election mechanisms. Second, it is preferable to have a very high degree tree. A low-degree tree that ends with clusters of size one, incurs more overhead for hierarchy maintenance. For example, for large groups we might have 1000 session members all sending global session messages, and each of those 1000 top-level session members could represent a cluster of up to 1000 children.

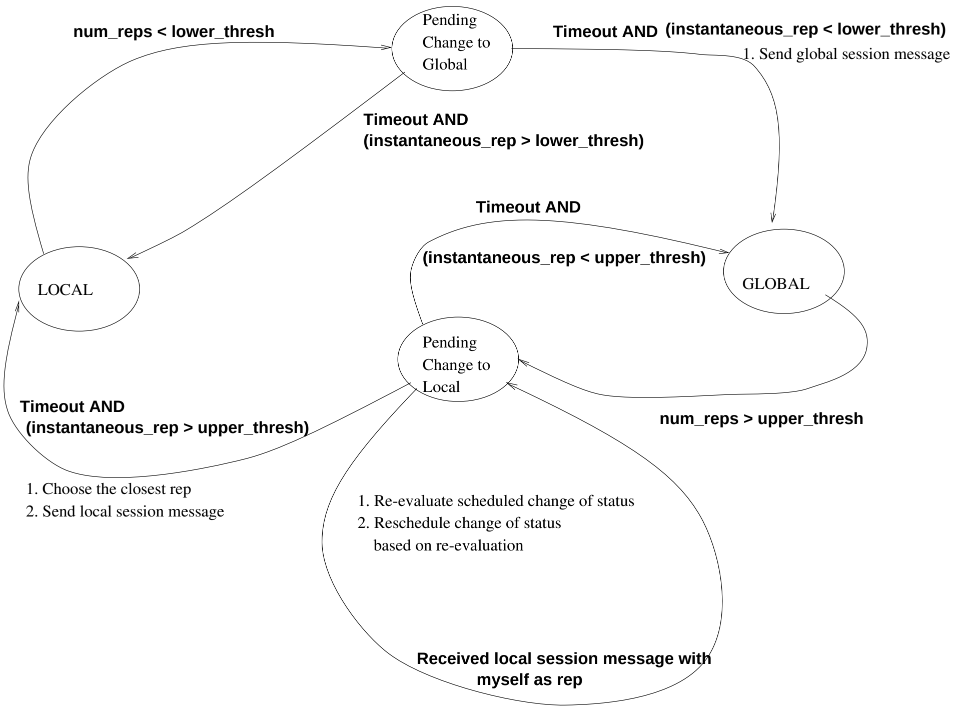


Figure 5: State diagram for hierarchy configuration

For this study, we assume that a two level hierarchy would be sufficient for SRM session message distribution. We use a distributed self-selection mechanism for the hierarchy configuration. Each member makes status change actions independently based on the session messages that it receives[3]. Members wait for a random time before taking these actions. The goal of the hierarchy configuration scheme is to maintain the number of global members, during non-transient phases, between a lower and upper threshold ($threshold_{lower}, threshold_{upper}$). The state diagram for the hierarchy configuration protocol is shown in Figure 5.

Let us assume that, at any time t, $ng_i(X, t)$ denotes the number of unique global session messages that member m_i has received in the time interval $[(t-X), t]$. Similarly $nl_i(X, t)$ represents the number of unique local members from which member m_i has heard. Each member maintains a sliding measurement window of period $W(= X1 * session\_interval)$, where $X1$ is greater than 1. The size of the measurement window is set to $X1$ times the session interval, to account for the loss of session messages. At any time t, if $ng_i(W, t)$ exceeds the upper threshold, member m_i schedules an event for becoming a local member. The scheduling time is chosen randomly

[3]No separate *meta-control* messages are exchanged for managing the hierarchy.

from a uniform distribution $[S1 * session\_interval, (S1+S2) * session\_interval]$[4]. If a global member m_i, with a pending change-to-local status event, receives a new local session message indicating that m_i has been chosen as a representative by some other member, m_i re-evaluates the change-to-local event. When the timeout occurs, the member m_i changes its status to local only if the current number of representatives is still more than the threshold. Similarly when the number of representatives goes below the lower threshold, local members schedule an event for changing from local to global.

A member estimates the current number of representatives by counting the number of global members that have sent session messages during a small window of time $B(= Z1 * session\_interval)$, where $Z1$ is less than $X1$. When a global member changes to local, it chooses the closest global member as its representative. Whenever a member changes its state from local to global or vice versa, it sends a session message so that the other members can detect the change[5].

The values of the parameters $S1$, $S2$ were assumed to be fixed for this study. In general, the parameters should be computed at each member based on the kind of status change operation and self-evaluated appropriateness for the operation. If the appropriateness is high, the parameters should be adjusted to pick a smaller timeout value. For instance, a representative with relatively few local members will have higher appropriateness to change its state to local than representatives with a large number of local members. Some of the self-evaluation mechanisms require additional information about the hierarchy in the session messages, such as cluster sizes,. For the results presented in this paper, we used a very simple bias to set the parameter $S1$. The value of $S1$ is set to 0 for the member that do not have any local members and it is set to 1 for representatives with local members. Mechanisms for self-evaluation of appropriateness are further discussed in Section 5.

Because things can be highly dynamic, we use randomized waiting periods, so that all members do not act at once. This reduces oscillations in the hierarchy.

We now discuss the effect of each of the parameters on self-configuration process.

3.2.1 Exploring the Parameter Space

The behavior of the hierarchy configuration protocol is dependent on the values of various parameters. The parameters $S1$ and $S2$ determine the timer interval to pick the event-timers. The size of the instantaneous measurement window B is proportional to the parameter $Z1$. The default values of the parameters $(S1, S2, X1, Z1)$ were set to $(0, 5, 3, 1.5)$ for the simulations presented in this paper. Each member can have different *appropriateness* for making a status change. Each member can select different values for various parameters based on its appropriateness for a particular status change operation.

- $S1$: The parameter $S1$ introduces a minimum deterministic wait before a member makes any status change step. $S1$ provides a stabilizing effect when the measurement windows of the members are randomly asynchronized. On the other hand if the the measurement windows of members are synchronized no status change operation can take place before the delay introduced by $S1$. Hence, larger values of $S1$ mean longer delays in status change operations.

- $S2$: The larger values of $S2$ increase the range of the time interval for random timers. This spreads out the status change operations amongst various members and allows more time to members for detecting change in status of other members. Thus larger values of $S2$ reduce the oscillations in hierarchy, but at the same time can also increase the configuration time of hierarchy.

- $Z1$: The value of $Z1$ determines the size of window for capturing the most recent changes in the hierarchy. The value of $Z1$ should always be greater than 1. If $Z1$ is too large, the measurement window fails to

[4] $S1$, $S2$ and $Z1$ are hierarchy configuration parameters

[5] A similar hierarchy configuration scheme is being considered for interdomain multicast routing [18].

capture the changes that have taken place recently.

- Acceptable Number of Representatives : The hierarchy achieves a stable state faster if the acceptable region for number of representatives ($threshold_{upper} - threshold_{lower}$) is large.

As mentioned earlier, all the members send global or local session messages when they change their status. Far away members can detect change to local status only after a period, W. On the other hand the members can detect changes in global status upon receiving global session messages. Thus, when the number of global members exceeds the upper threshold, all the members can detect this very quickly. In such a case if S2 is too small, then a large number of members can become local before they can detect that other members have also changed to local. As a result, the hierarchy configuration process might stabilize around the lower threshold if the scheduling interval is small.

In the next section we describe the simulations we conducted to evaluate the proposed approach. We study the effect of these parameters on the hierarchy configuration. The interval for scheduling status change events can be varied based on the members *goodness* measure. Members that will potentially be better representatives choose their schedule time from a longer period and vice-versa. We discuss self-evaluation of appropriateness later in Section 5.

4 Simulations of the Scalable Session Messages

In this section we describe the simulation studies we conducted to study the scalable session messages. We modified the SRM implementation in *network simulator (ns)* [23] to use hierarchy configuration and proposed session message distribution. We conducted three sets of studies using the simulator. The first set of experiments compares the loss recovery performance of original SRM with use of scalable session messages in SRM. We then studied the improvement in the scaling properties of SRM when using the modified session message exchange protocol. We finally explored the parameter space for hierarchy configuration.

We conducted simulations on a wide range of scenarios with different topologies, session sizes and member placement. In this paper we present the simulation results using the topology shown in the Figure 6. This topology is a 100 node transit-stub topology generated using Georgia Tech Internet Topology generator [24]. Transit-stub topologies are two level hierarchical graphs generated by interconnecting transit and stub domains. Several measurements were taken for various session sizes. Other topologies also showed similar results.

While we recognize that the scaling problems described in the paper are not prohibitive for the session sizes simulated, and therefore, do not in themselves warrant use of mechanisms proposed here. Nevertheless, as an initial step, we chose these session sizes to facilitate our initial detailed study of the proposed mechanisms and their behavior. Future work will extend our evaluation to larger set of simulations to verify the generality of our initial findings.

In the next subsection we present the effect of scalable session messages on SRM loss-recovery performance.

4.1 Effect of Scalable Session Messages on SRM Performance

There are three metrics that can be used for comparing the performance of various flavors of SRM, i) number of requests per loss, ii) number of repairs/loss, and iii) recovery delay. We have compared the number of requests/loss and recovery delay for the two schemes.

Various simulations were run by populating the topology in the figure 6 with session sizes - 20, 30, 40, 50, 60 and 70. For each simulation a set of group members, one source and one bad link on the distribution tree were chosen randomly. For each graph the x-axis shows the session size; ten simulations were run for each value of session size. Each simulation is represented by $\diamond$. The request and repair algorithms in both the schemes used

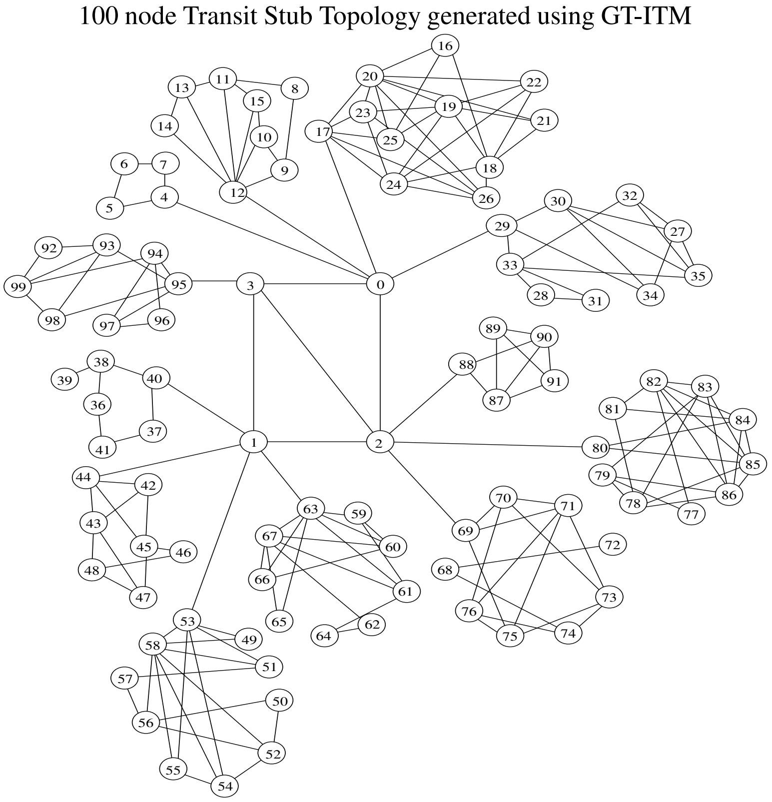
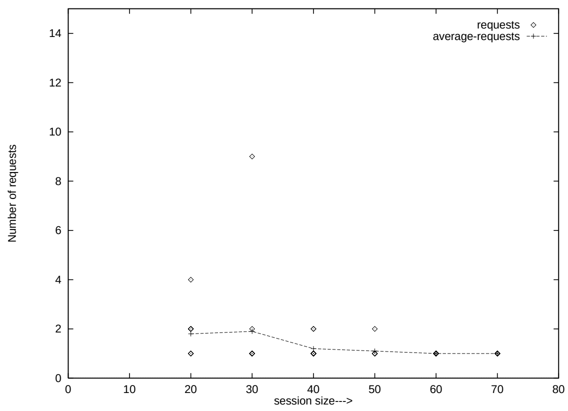
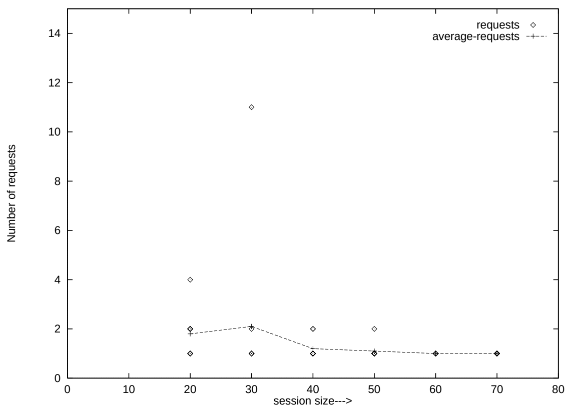


Figure 6: A simulated topology generated using gatech topology generator



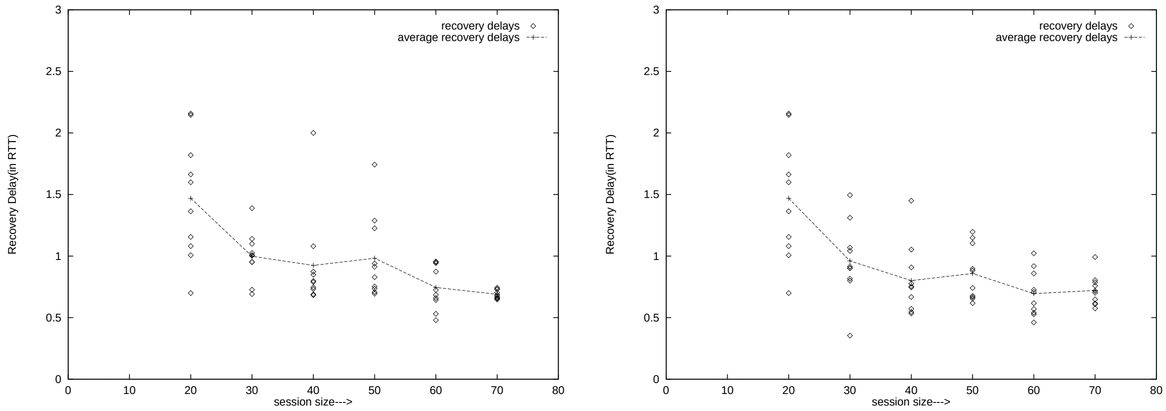
(a) Without Scalable Session Messages



(b) With Scalable Session Messages

Figure 7: Comparison of number of requests for single loss

fixed timer parameters for error recovery. The values of the loss recovery parameters $(C1, C2, D1, D2)$ was set to $(2.0, 2.0, 1.0, 1.0)$ for the simulations presented here. We are conducting more performance comparisons using adaptive timer parameters as described in [1]. Figure 7 shows the number of requests per loss for the two schemes. We observed that scalable session messages did not significantly increase the number of requests per-loss. The number of repairs is not considerably affected too as shown in [25].



(a) Without Scalable Session Messages

(b) With Scalable Session Messages

Figure 8: Recovery delay for single loss

Figure 8 shows the loss recovery delay of the last member in the session to receive the repair. The recovery delay has been normalized in terms of the distance of the data from the requestor. The recovery delays in case of scalable session messages were normalized using the delays computed using flat distribution of session messages and not the approximated distances. SRM with scalable session messages has similar recovery delays as the original SRM. Thus the loss recovery performance of SRM is not degraded by adopting scalable distribution of session messages.

SRM session messages are not very critical for detecting loss of data packets if the data rate is high. Under scenarios where the data rate is low or last packet of a data stream is lost, time taken for loss detection depends on session messages. The loss detection time depends on the latency incurred due to: a) session message interval and b) in case of hierarchy, the relaying of the sequence number state of the source to its representative and further to some of the members that have experienced loss. If the session message intervals are same for the hierarchical and the flat distribution schemes this latency is more for the hierarchical distribution of session messages. But employing hierarchical distribution allows session messages to be sent at a higher rate compensating for the increased latency. So our approach does not negatively impact the loss detection time.

Another set of simulations to study the bandwidth and state savings with scalable session messages are presented in the next subsection.

4.2 Bandwidth and State Savings

We also examined the potential savings of the proposed mechanisms. Our approach improves the scaling behavior of SRM by both reducing the amount of state stored in members and by allowing higher frequency of session messages for better adaptability.

Reduction in amount of state stored in members With scalable session messages, each member does not have to store information(state) about all other members of the groups. Since there is a limit on the number of global representatives and on the number of local members for a single global representative, the amount

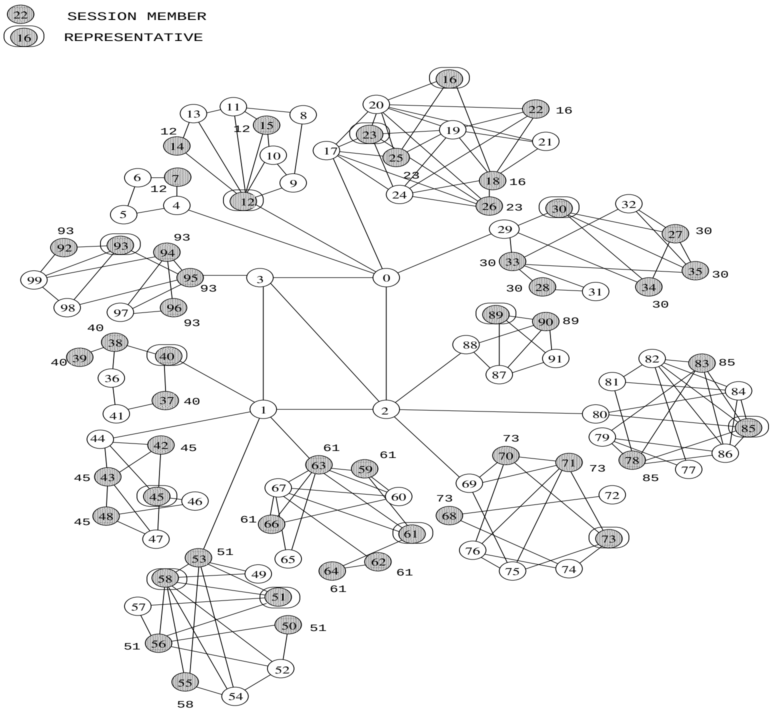


Figure 9: An example of self-configured hierarchy (session-size = 60)

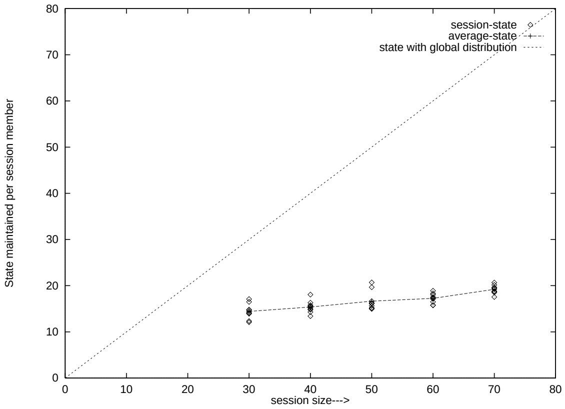


Figure 10: State requirement comparison

of state required in each member is bounded. Figure 10 compares the amount of state maintained at each member for the simulations presented in previous subsection. In Figure 10 the dotted line plots the amount of state maintained at each member for the original SRM scheme. One state unit corresponds to information stored about one member. We also plot the average amount of state maintained by each member in our scheme. Each $\diamond$ represents one simulation of scalable session messages. The amount of state stored in each member grows linearly with the size of the group in the original scheme. In our approach the state does not show similar growth.

Reduced bandwidth and increased frequency of session messages Instead of limiting the session message traffic and comparing the frequency of session messages in the two schemes, we compared the session message bandwidth assuming same targeted frequency of session messages. We observed that, as expected, the bandwidth used by scalable session messages is less than the global distribution of session messages. These bandwidth savings can be translated into increased frequency of session messages when the session message traffic is rate limited.

4.3 Evaluation of the Self Configuration Scheme

In this section we evaluate the self-configuration schemes for scalable session messages. A wide range of topologies were simulated to study the hierarchies formed using our approach. Figure 9 shows one example of a self-configured hierarchy for a SRM session in the transit-stub topology shown in the Figure 6. It is difficult to compare hierarchies based on a single quantitative measure. Hence we describe several measures of the *goodness* of a particular hierarchy.

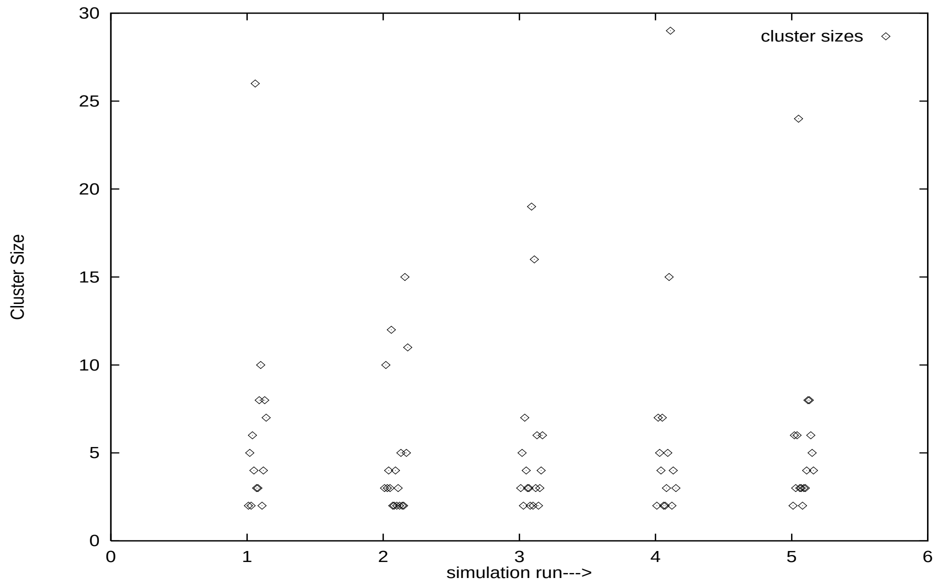
We compared hierarchies based on two metrics, namely, a) the distribution of cluster size, and b) the distribution of session message traffic on various links. The size of a cluster determines the amount of space required to maintain the distance information in the members of the cluster. The space requirements grow with the size of the cluster. Hence the size of the largest cluster should have a maximum bound.

Another important factor to consider is the range of the clusters and placement of representatives within a cluster. The placement of the representative in a cluster affects the error introduced in delay estimates between members in different clusters. This error in delay estimates between various members is bounded by the distance between the local member and its corresponding representative. Hence the distance between a local member and its representative should not be large.

Optimally, the clusters in a hierarchy should be so placed that there is not much overlap among various clusters. The regions where clusters overlap experience local session message traffic of the overlapping clusters. The cluster overlaps in a hierarchy can be captured by the distribution of session message traffic on various links. We also looked at the distribution of the session message traffic on various links in the network for comparing hierarchies. It must be noted here that besides the constructed hierarchy, traffic distribution is very sensitive to the underlying network topology and placement of session members. The total session message traffic is another potential metric for such comparisons. Some of the analyses performed on the hierarchies constructed using our self-configuration algorithm are presented here.

Our simulations used transit-stub topologies generated by the Georgia Tech Internet Topology generator [24]. In this section we present some simulation results with a 440 node, 629 links, transit-stub topology with 2 transit domains. The session size was 90 and the members were chosen randomly from the stub domains. No member was placed in the transit domains. Each member sends a session message every 3 secs[6]. The values of the parameters $(S1, S2, Z1)$ were set to $(0, 5, 1.5)$.

[6]The value of session message interval used for simulations is small as compared to typical values from rate-controlled session messages and hence the volume of session message traffic appears larger.

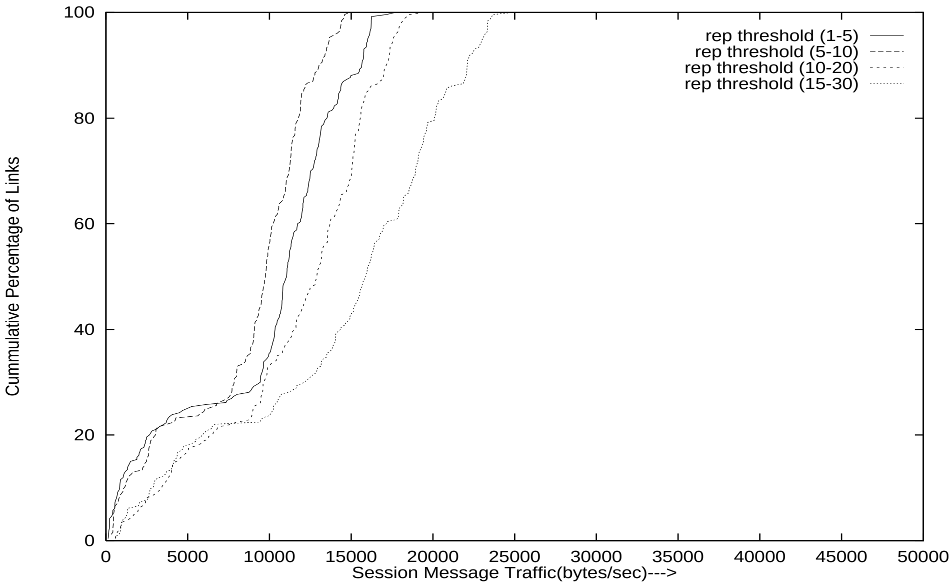


Topology : 440 node, 2 Transit Domains; Session Size: 90; Representative Thresholds : 10 to 20

Figure 11: Cluster size distribution

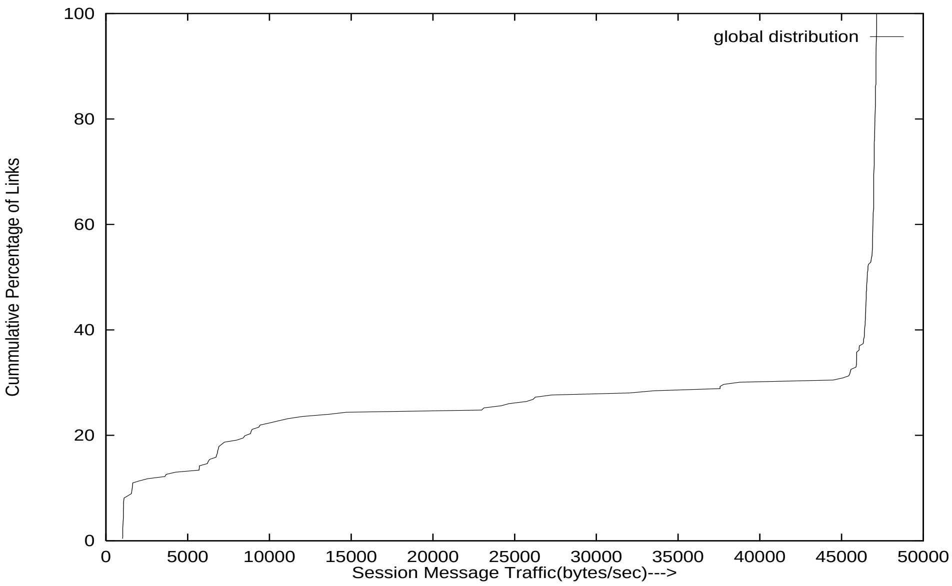
Figure 11 shows the distributions of cluster sizes for various simulation runs for the scenario described above. The x-axis shows the simulation run and y-axis shows the cluster size. Each $\diamond$ represents a cluster size. The acceptable threshold for the number of global representatives was set from 10 to 20. We observed visually that cluster formation mapped nicely to the placement of the session members and locality of representatives was preserved. Some of the cluster sizes were very large when a large number of members were placed in a small vicinity. In the current scheme the representatives do not report any hierarchy related information (such as the size of the cluster) in session messages. As mentioned briefly in Section 5 the algorithm can be extended to split large clusters for such cases. Similar cluster size distribution was observed in other topologies with different session sizes.

We also analyzed the distribution of session message traffic on various links. Figure 12 plots the traffic distribution on the links for the scenario described above. The x-axis shows the session message bandwidth in bytes per sec averaged over a 50 sec simulation. The y-axis shows the cumulative percentage of links with session message bandwidth less than x. Only those links that were on some distribution tree were taken into account for this analysis. Figure 12 shows the traffic distribution for hierarchies constructed with different thresholds for the number of representatives. The different thresholds that were used are: (1 - 5), (5 - 10), (10 - 20) and (15 - 30). Figure 13 shows the traffic distribution for global distribution of session messages at a fixed rate. Since we used a fixed frequency rather than a fixed aggregate bandwidth for session messages, the curves in figures 12 and 13 were used only to compare the shapes of the curves and not for the absolute values. With the same targeted frequency of session messages for both the schemes, the global distribution curve is used for factoring out the effect of topology and member placement on traffic distribution. The graph shows that the distribution curve for session message traffic is flat in the middle implying a bimodal distribution. Based on the session message traffic the links can be loosely put into two groups corresponding to the two modes of the distribution curve. The first group includes links with very little session message traffic. Most of these links are near the branching points of the packet distribution tree. The links in the transit domains and in the cluster overlap regions experience heavier session message traffic. A hierarchy with a low degree of overlap will not have a large number of links in the high session message traffic group. We observe by comparing Figure 12 and Figure 13 that the number of links with low traffic increases



Topology : 440 node, 2 Transit Domains; Session Size: 90; Representative Thresholds : 10 to 20

Figure 12: Cumulative distribution for session message traffic



Topology : 440 node, 2 Transit Domains; Session Size: 90; Global Distribution of Session Messages at fixed rate

Figure 13: Cumulative distribution for session message traffic

when scalable session messages are used. As the $threshold_{upper}$ decreases the session message traffic decreases. But the session message traffic becomes greater when the representative threshold becomes too low, such as (1 - 5). This happens because the size and range of clusters grow as the representative threshold decreases. Thus the number of representatives should not be too low; it can not only deteriorate SRM performance, but can also have heavy session message traffic.

Future work will investigate schemes for allocating session message bandwidth to global and local session messages and study their effect on the traffic distribution curves.

5 Conclusions and Future Work

This paper describes how to improve scaling properties of SRM by disseminating session messages using a self-configuring hierarchy. We discussed a new algorithm for self-configuration based on random timers and appropriateness measures as well as its effect on the loss recovery performance. This is a very first study of self-configuring hierarchies based on random timers and self-evaluated approapriateness. The initial results presented in this paper are promising and raise many interesting questions, some for SRM and some generally, for other multiparty protocols. In this section we describe some of the future studies planned to address these questions.

Extending the basic scheme for self-configuration The hierarchies self-configured using pure random timers can sometimes have undesirable properties. Some members are more appropriate to make certain status change operations. The timers should be so biased that it is more probable that members with high appropriateness make status change operation. A decentralized approach should be employed by each member should self-evaluate its *appropriateness* based on locally available information. A member with high appropriateness for a particular action schedules status change operation with a short timer. We have identified, but not yet analyzed several heuristics for evaluating appropriateness for making different status changes. The heuristics for some of the status change operations are given below:

Changing from Global to Local A member that is closer to another global member and does not have many local members has high appropriateness for changing from global to local.

Changing from Local to Global A local member has high appropriateness if it is far from its representative or if its representative has a large number of local members.

Changing Representatives A member might want to change representative if there is another representative closer to it. Such members should have high appropriateness.

Some of these heuristics might require some additional knowledge like sizes of various clusters. A global representative can report the number of its local members in the global session messages.

Impact of Hierarchy Transients The transients in the hierarchy due to membership changes might have an effect on the SRM performance. We describe the impact of the transients related to some of the status change operations below:

Failure of a Representative Our approach is robust to failure of representative as the local members receive session messages from other global members during this transient phase. In some cases, it can increase the loss detection time as the sequence number state of the local member might not be relayed beyond local scope.

New Representative A new representative does not have any direct impact on the loss recovery performance, though it can initiate reconfiguration of hierarchy.

Future work will study the dynamics and stability of the self-configuration scheme and its impact on the loss recovery performance.

Loss Recovery Performance The underlying topology of the network affects the loss recovery performance of SRM. Floyd et. al. [1] suggest adjusting the timer parameters based on the past behavior of loss recovery process. In this paper, we have compared the loss recovery performance using fixed timer parameters. Future work will study scalable session messages' loss recovery performance with adaptive timer parameters.

Reliable multicast applications that are more latency tolerant or have redundancy do not require timely retransmissions of lost packets. Network Text Editor (NTE) [26] relies on redundancy in new packets and is less dependent on retransmissions. NTE uses a sender driven request retransmission approach and does not need a delay matrix to other members. RPM [27] is used for reliable exchange of routing policy information among various sites. RPM tradeoff the number of retransmissions for recovery delay. RPM employs a SRM like algorithm for request and repair but worst delay between any two members is used for setting the timers. We plan to study how errors introduced in delay estimates by scalable session messages interact with these protocols.

Rate Limiting Session Messages SRM session messages are rate limited to a small fraction of session bandwidth. The rate limiting mechanism has to be extended for hierarchical distribution of session messages. We need to design mechanisms for sharing the session message bandwidth among differently scoped session messages.

Generalizing our Approach to other Protocols Global group synchronization is required by other multiparty end-to-end protocols such as RTP [9]. Such protocols that distribute information globally can also use the mechanisms suggested in this paper for improving the scaling behavior. Though we have described the scheme assuming two levels, it is can be extended to a multiple level hierarchy also.

References

[1] Sally Floyd, Van Jacobson, Steven McCanne, Ching-Gung Liu, and Lixia Zhang. A reliable multicast framework for light-weight sessions and application level framing, extended report. *LBNL Technical Report*, pages 1–37, September 1995. also submitted for publication in *IEEE/ACM Transactions on Networking*.

[2] Markus Hofmann. Adding scalability to transport level multicast. *Proceedings of Third COST 237 Workshop- Multimedia Telecommunications and Applications*, November 1996.

[3] B. Whetten, T. Montgomery, and S. Kaplan. A high performance totally ordered multicast protocol. *Lecture Notes in Computer Science*, 938:33–??, 1995.

[4] S. Armstrong, A. Freier, and K. Marzullo. RFC 1301: Multicast Transport Protocol, February 1992.

[5] John C. Lin and Sanjoy Paul. RMTP: A Reliable Multicast Transport Protocol. *Proceedings of IEEE INFOCOM '96*, pages 1414–1424, April 1996.

[6] R. Yavatkar, J. Griffioen, and M. Sudan. A Reliable Dissemination Protocol for Interactive Collaborative Applications. *Proceedings of ACM Multimedia '95*, 1995.

[7] William C. Fenner. Internet group management protocol, version 2. *Internet Draft*, May 1996.

[8] D. Mills. RFC 1305: Network Time Protocol (v3), April 1992. Obsoletes RFC1119.

[9] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RFC 1889: RTP: A Transport Protocol for real-time applications, January 1996.

[10] Kannan Varadhan. Models of Route Dynamics for End-to-end Protocol Evaluation. *PhD. Dissertation Proposal, Unpublished manuscript, Personal Communication*, September 1996.

[11] Bernard Aboba. Alternatives of Enhancing RTP Scalability. *Internet Draft*, November 1996.

[12] Todd Montgomery. Scalable Session Message Mechanisms. *Private Commincations*, 1996.

[13] Ching-Gung Liu. A Scalable Reliable Multicast Protocol. *PhD. Dissertation Proposal,Unpublished manuscript, Personal Communication*, pages 1–54, September 1996.

[14] Mark Handley. SAP: Session announcement protocol (version 1) [draft 0.2]. *Internet Draft*, June 1996.

[15] Vern Paxon. End-to-end routing behavior in the internet. In *SIGCOMM Symposium on Communications Architectures and Protocols*, Stanford, California, August 1996.

[16] L. Kleinrock and F. Kamoun. Hierarchical routing for large networks. *Comp. networks North Holland*, 1, 3:155–174, 1977.

[17] P. F. Tsuchiya. The landmark hierarchy: A new hierarchy for routing in very large networks. In *Proc. ACM SIGCOMM '88*, pages 35–42, Stanford, CA, August 1988.

[18] Cengiz Alaettinoglu, Deborah Estrin, Satish Kumar, and Dave Thaler. "Self-configuring hierarchy for Interdomain PIM". http://www.isi.edu/ kkumar/hpim.

[19] Ching-Gung Liu, Deborah Estrin, Scott Shenker, and Lixia Zhang. Local Error Recovery in SRM: Comparison of Two Approaches. Technical Report, USC TR97-648, University of Southern California, February 1997.

[20] Carsten Bormann. MTP/SO: Self Organizing Multicast. *Internet Draft*, November 1996.

[21] Markus Hofmann. Enabling group communication in global networks. *Proceedings of Global Networking*, June 1997.

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

[23] Steve McCanne and Sally Floyd. "LBNL Network Simulator". http://ee.lbl.gov/ns.

[24] E. Zegura, K. Calvert, and M. Donahoo. A Quantitative Comparison of Graph-based Models for Internet Topology. *To appear in Transactions on Networking*, 1997.

[25] P. Sharma, D. Estrin, S. Floyd, and L. Zhang. Appendix: Scalable session messages in srm. ftp://catarina.usc.edu:/pub/puneetsh/paper/ssm_appendix.ps, University of Southern California, June 1996.

[26] Mark Handley and Jon Crowcroft. Network Text Editor (NTE) A Scalable shared text editor for the MBone. In *SIGCOMM Symposium on Communications Architectures and Protocols*. ACM, September 1997.

[27] Ramesh Govindan et. al. A Fully-Replicated Internet Routing Registry System: Design and Performance. *Work Under Progress, Personal Communication*, July 1997.