Scalable Multicast Routing for Ad Hoc Networks

Manoj Pandey
Cisco Systems
mpandey@cisco.com

Daniel Zappala
Computer Science Department
Brigham Young University
zappala@cs.byu.edu

***Abstract*—Routing in a mobile ad hoc network is challenging because nodes can move at any time, invalidating a previously-discovered route. Multicast routing is even more challenging, because a source needs to maintain a route to potentially many group members simultaneously. Providing scalable solutions to this problem typically requires building a hierarchy or an overlay network to reduce the cost of route discovery and maintenance. In this paper, we show that a much simpler alternative is possible, by using source specific semantics and relying on the unicast routing protocol to find all routes. This separation of concerns enables the multicast routing protocol to focus on minimizing join latency, repair latency, and control overhead. We design a routing protocol based on these principles and demonstrate its effectiveness through simulations.**

I. Introduction

Group communication is an important service for mobile ad hoc networks, because many proposed applications involve coordination among groups of people, such as for emergency or military operations. To provide efficient group communication, a number of multicast routing protocols have been proposed for ad hoc networks, usually building either a tree or a mesh. The focus of the first generation of protocols was to have a fast reaction to network changes, so that packet loss could be minimized [1], [2], [3], [4], [5]. More recent work in this area addresses the scalability of multicast routing, in terms of network size, group size, and the number of groups [6].

Two approaches have been used to provide scalable multicast routing in ad hoc networks. One approach is to create a hierarchy by partitioning the network into regions, with a group leader elected in each region [7]. Multicast packets are then sent among group leaders, and each leader forwards packets to members within its region. Another approach is to build a peer-to-peer overlay among the nodes, and then construct a tree based on the overlay [8], [9], [10], [11], [12]. In this case, multicast is provided entirely at the application layer, and packets are sent among overlay nodes using TCP connections.

While these two approaches have been shown to scale well, they both have the drawback that additional structure, either a hierarchy or an overlay, must be maintained in order to provide multicast. This incurs additional overhead, which increases the more nodes move. Both approaches may also increase delay, since packets are not necessarily delivered along the shortest or fastest paths between the source and the group members. Since overlay multicast is built out of unicast connections, it also introduces additional *stress*, or excess packets, as compared to multicast forwarding in the network layer.

What these approaches overlook is that scalable multicast routing has already been designed on much larger scale for the Internet, using the Source Specific Multicast (SSM) [13] protocol. In SSM, each multicast tree has a single source, the group owner, which is the root of the tree. Using a single source eliminates the need to perform an Internet-wide search for any user that wants to be a source for the group. Group members join a multicast tree by sending a *join* message that follows the unicast route to the source for that tree. If the join reaches a router already on the tree, then it *merges* at that point. If any group member wants to send data, it can either relay its data through the source, or it can notify the group members and build a new tree rooted at that group member. This design scales well because it separates group advertisement and source discovery (which operate at the application layer) from tree maintenance and data forwarding (which operate at the network layer).

In this paper we apply this principle of separation of concerns to the design of an ad hoc multicast routing protocol, called ASSM. As with SSM, the ASSM protocol assumes that trees are identified by a group owner. This means that, in order for a group member to join a multicast tree, it only needs to know the unicast route from itself to the group owner. Since scalable unicast routing protocols have already been developed for ad hoc networks, ASSM can leverage this work to provide scalable multicast.

Based on this architecture, we investigate two fundamental questions: (1) *Can a purely receiver-oriented multicast routing protocol provide low repair latency while also minimizing overhead?* (2) *Does this design scale as well in ad hoc networks as it does in the Internet?* Repair latency is a critical measure because it indicates how quickly the multicast routing tree can be rebuilt when nodes move. Existing protocols have been designed to minimize repair latency, but often at the expense of high overhead, since all nodes participate in the repair process. We have designed ASSM so that joining and repair decisions are made locally by a group member, without any cooperation from other nodes on the tree. While at first this seems like a disadvantage, we show that this design actually makes it possible to have both low latency and low overhead.

While ad hoc network deployments are typically not large, scalability is still an important goal because network bandwidth is often very limited. As the number of groups increases, more flows compete for this limited bandwidth, and it is important that the network provide efficient delivery, with low loss rates and low control overhead. Since each application may

1-4244-2575-4/08/$20.00

need several multicast groups, for example separate groups for video, voice, text, and control messages, only a small number of applications need to be active for scalability to become an issue. We show that ASSM achieves good scalability because it uses a tree, rather than a mesh, and because control overhead is very low.

In subsequent sections, we describe the design of ASSM, then use a simulation study to demonstrate how ASSM achieves low repair latency and good scalability.

II. Background

When designing a multicast routing protocol for a mobile ad hoc network, there are two main concerns that arise due to mobility: source discovery and tree maintenance. Because nodes may be mobile, a group member does not know the location of those nodes who want to send data to the group. Once the sources are found, the group members join a tree or mesh that connects them to the sources. The routing protocol must then repair the tree or mesh whenever any node on the tree moves.

ODMRP [1] solves these problems using a source-oriented protocol. To provide source discovery, each source for each group floods a JOIN QUERY message throughout the entire network. Each node receiving this message stores the previous hop from which it received the message. When a group member receives a JOIN QUERY, it responds by sending a JOIN REPLY to the source, following the previous hop stored at each node. The collection of state created by this and subsequent *Join Reply* messages forms a mesh that connects all sources to all group members. To repair the tree when nodes move, ODMRP simply re-sends the JOIN QUERY periodically and times out mesh state periodically. Over time, the mesh adapts to any changes in node location.

The basic trade-off in ODMRP is between throughput and control overhead. A source can increase throughput by sending more frequent JOIN QUERY messages. Each message rebuilds the multicast mesh, repairing any breaks that have occurred since the last query, thus increasing the chance for subsequent packets to be delivered correctly. However, because each query is flooded, increasing the query rate also increases the control overhead of the protocol. ODMRP also can trade off redundancy and forwarding overhead. By increasing the soft-state timer for node forwarding state, ODMRP can increase the robustness of the mesh and hence provide more redundant paths for packets to be delivered. Of course, the richer the mesh, the greater the overhead when forwarding multicast packets.

ADMR [2] solves these two problems using a receiver-oriented protocol, combined with aggressive maintenance of the multicast tree. To find multicast sources, a group member floods a MULTICAST SOLICITATION message throughout the network. When a source receives this message, it responds by sending a unicast KEEP-ALIVE message to that receiver; the receiver joins the tree by sending a RECEIVER JOIN along the reverse path. When many group members repeat this action, a separate tree is formed for each multicast source. To ensure the tree is repaired whenever a node moves, each node monitors incoming packets and begins a repair process if it misses some number of consecutive packets. The repair is done by having the node downstream from the failure transmit a hop-limited flood of a RECONNECT message, to reattach to the existing tree.

MAODV [3] is similar to ADMR in that it also uses a receiver-oriented protocol. Rather than building a separate tree for each source, however, MAODV builds a single tree shared among all sources. A group member finds the tree by broadcasting a RREQ message; any node currently on the tree responds with a RREP. The group member can then choose the best route and connect itself to the tree. To maintain the tree, each node exchanges HELLO messages with its parent. If a node stops hearing from its parent, it reconnects itself by broadcasting a RREQ.

One of the key advantages of MAODV and ADMR is that they are receiver-oriented, so a new group member can quickly join the multicast tree, rather than waiting for the source to add it. In addition, they both use a tree, rather than a mesh, so they have lower forwarding overhead than ODMRP. In terms of repairing the tree, ADMR can react to broken links more quickly because each node monitors packet reception, rather than waiting for a timer to fire. However, both ADMR and MAODV allow any node to repair the tree. This can lead to very high overhead during periods of high mobility, since many nodes on a single tree will initiate repairs simultaneously, leading to a large number of broadcast messages. Another problem with ADMR is that it switches to flooding when packet loss is high; if the packet loss is due to congestion, the flooding of packets actually makes congestion worse [14].

III. ASSM

ASSM is a lightweight, receiver-oriented multicast routing protocol designed to provide scalable multicast routing for ad hoc networks. With ASSM, multicast functionality is divided into the following components:

- *group advertisement and source discovery*: As with SSM [13], groups are identified by a combination of the source and group address (S, G), rather than by the group address (G) alone. Essentially, every source is the group owner for its own collection of multicast addresses, which allows addresses to be assigned permanently to groups. The group and owner identifiers can then be advertised via any application, on a web page, or even configured directly into an application. This means that when a group member wants to join a multicast tree, ASSM already knows the identity of the root of the tree. This avoids using a network-wide broadcast to discover multicast sources, as is necessary in many other multicast routing protocols designed for ad hoc networks [1], [2], [3].
- *unicast routing*: A unicast routing protocol finds and maintains routes to all nodes in the ad hoc network. Whenever ASSM needs to send a message, it sends it

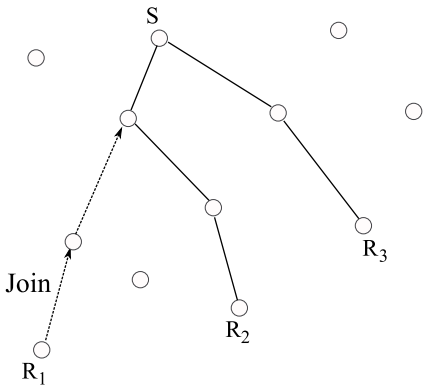


Fig. 1. ASSM: Joining a Multicast Tree

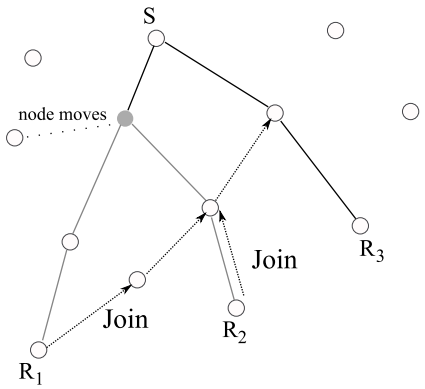


Fig. 2. ASSM: Repairing a Multicast Tree

using a unicast route. This design concentrates route discovery in a single component, which eliminates duplicate broadcasts that are often performed by multicast routing protocols. This also allows multicast to take advantage of any optimizations applied to the unicast routing protocol; making unicast routing more scalable directly benefits multicast routing as well.

- *multicast routing*: ASSM builds and maintains a separate tree for each multicast group, which reduces forwarding overhead as compared to a mesh. ASSM uses a receiver-oriented design, so that join latency is small. ASSM also localizes the repair decision, so that only the group members rebuild the tree, rather than asking all nodes on the tree to repair it when it breaks. This minimizes repair overhead, particularly during periods of high mobility.

Due to this separation of concerns, ASSM is able to provide scalable multicast routing for large networks, large groups, and large numbers of groups. In addition, this design greatly simplifies the multicast routing protocol, which only needs to install and maintain multicast forwarding state using existing unicast routes. We describe how ASSM builds and maintains trees, as well as how additional sources can send to the same multicast group.

A. Building a Tree

When a node joins a group, it already knows the identity of the owner of the group, S, because this is provided through the group advertisement mechanism. The owner is also the root of the multicast tree. Figure 1 illustrates how a new group member joins a multicast tree. The new member sends a JOIN message that follows the unicast route to the source S. Whenever a node receives a JOIN message, it checks whether forwarding state exists for the group already. If it does exist, the node adds a new child to the forwarding state (the new child is the previous node that sent the JOIN message) and then the JOIN message stops. If the state does not exist, the node creates it, adds the child, and sets the parent to the next node on the route. The node then forwards the JOIN to the parent until it reaches the source. In the example shown, R2 and R3 have previously joined the tree, so when R1 sends a JOIN it stops when it reaches the tree.

The state that ASSM creates is "soft state", meaning it times out if it is not refreshed periodically. All group members resend a JOIN message periodically, and these JOIN messages are forwarded upstream if the state has not already been refreshed recently. If a group member wants to leave the tree, it can stop refreshing the JOIN messages and the state on its branch will eventually time out. To leave the group more quickly, a node can send a PRUNE message upstream to immediately delete the state on its branch.

B. Repairing a Tree

The advantage of using soft state and refresh messages is that it also rebuilds the multicast tree automatically whenever a node moves. As a JOIN message propagates upstream, it gets forwarded upstream based on the current unicast route. If this route has changed, ASSM will automatically be built on the new path. If the JOIN message cannot be transmitted from one hop to the next hop, this indicates that either the next node has moved or the frame was lost due to contention. We use the MDA protocol [15] to determine the cause of the lost frame, and if it is due to mobility then the unicast routing protocol sends a failure message to the originator of the *Join* message, which is the group member. The group member then sends a new *Join* message, which triggers a new route request with the unicast routing protocol. Once the new route is computed, ASSM can send the JOIN upstream to rebuild the tree.

The process of repairing a tree due to mobility is shown in Figure 2. In this case, a node upstream has moved, and previous JOIN messages have caused the downstream receivers to be notified of this failure. When new routes are computed to the source of the tree, ASSM sends a JOIN message that follows the new route. To prevent the JOIN from merging on the old route in this case, ASSM sets a bit indicating that it is repairing the route, so that it can progress further upstream. A node sends a repairing JOIN upstream if it hasn't already done so recently.

C. Multicast Forwarding

Each node on the multicast tree has a parent node (null for the root of the tree) and a set of children nodes (null for a leaf). To forward multicast packets, the node first checks whether

the packet arrived from the parent, and if not it discards the packet. If the packet is not discarded, the node must deliver it to each of the children.

By default, multicast packets are forwarded down the tree using broadcast at the MAC layer. However, broadcast traffic often suffers when it must compete with traffic sent reliably by the MAC layer (e.g. using an RTS/CTS exchange). In this case, the sender of a packet may set a field in each packet asking that forwarding be done with reliable broadcast [16] or some other method.

D. Additional Sources

In this paper, we study ASSM with a single source per group. However, there are a number of methods ASSM could use to allow multiple sources to send data to the same multicast group:

- *relaying:* A source can relay data through the group owner by sending it to owner via unicast. When the group owner receives the data, it multicasts it over the tree. The primary disadvantage of this method is that it adds additional delay. This method can also be inefficient, since some routers may handle the same data twice – once when sending it to the root as a unicast packet, and again when forwarding it as a multicast packet.
- *additional groups:* A new source can send a unicast message to the group owner, informing it of its identity. The group owner then relays a multicast message to the group, informing them of the presence of the new source. Finally, the group members join a separate multicast tree rooted at the new source. This method avoids the delay incurred by relaying all data through the group owner, but adds additional unicast and multicast routing overhead.
- *shared tree:* A new source can send data using the same tree as the group owner, provided multicast forwarding is done using a shared tree. For a shared tree, the parent and child nodes are treated equally when forwarding packets – a packet may arrive from any of these nodes and is then forwarded to the rest of them. Using Figure 1 as an example, R1 could send a packet to the group using this same tree, by having its parent accept the packet and send it further up the tree. This provides both low delay, low forwarding overhead, and low routing overhead.

To implement multicast using a shared tree, every node must cache data packets to prevent forwarding loops. Imagine node A sends a packet to child node B, which forwards it to child C. In a wireless network, when node B forwards the packet, A will also receive it. In a directional tree, A will discard the packet, since it did not arrive from its parent in the tree. However, in a shared tree, this may be a new packet that B is forwarding on behalf of a downstream source. A needs to be able to distinguish between new packets and packets it has already sent. A small cache of packets a node has already forwarded solves this problem.

IV. Evaluation

We simulate ASSM in mobile ad hoc networks based on IEEE 802.11 standard using the GloMoSim simulator [17]. In our simulations, the data rate is set to 11 Mbps and ASSM uses a 1 second refresh timer for the JOIN message. We measure the following metrics:

- *repair latency:* the time between when a node moves and when it begins to receive multicast data again,
- *control overhead:* the number of control messages sent to repair the tree; in some cases we report this as a percentage of all of the messages, including data,
- *packet delivery ratio:* the ratio of the number of packets received to the number of packets sent, and
- *transmission overhead:* the ratio of the number of data messages transmitted (originated or forwarded) to the number of data messages received at leaf nodes; this measures the efficiency of a routing protocol.

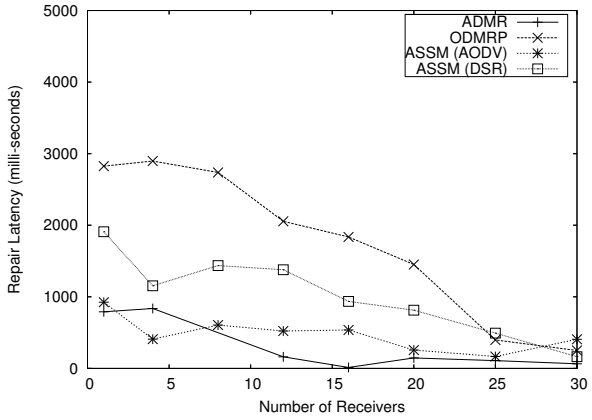
We repeat each experiment 5 times and average the results.

A. Repair Latency and Overhead

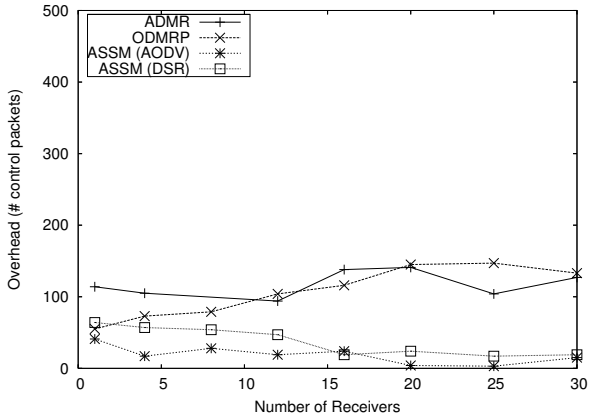
We first examine repair latency and overhead for ASSM. For these experiments we place the group owner and 50 other nodes at random locations within a $1000m^2$ field. We then build a multicast tree, with 1 to 30 members. The source transmits four 64-byte packets per second. In one experiment we randomly select a group member, move it to a random location, and measure how fast it can rejoin the multicast tree. In a second experiment we move just the source to a new location and measure how fast the tree is repaired.

Figure 3 shows the repair latency and overhead for the case when the group member moves. As expected, since ASSM localizes the repair decision at the group member, repair overhead is very low compared to ADMR and ODMRP. Overhead drops as the number of group members increases because this makes it easier for the unicast routing protocol to find nearby nodes that already have routes to the source. Repair latency generally decreases as the number of receivers increases because it is easier to find a nearby branch on the tree. Repair latency for ASSM is low when it uses AODV as a routing protocol, however it is higher when ASSM uses DSR for unicast routing. This occurs because DSR caches routes, and when a node moves the cached routes may all become invalid. It sometimes takes several attempts before all of the cached routes are attempted and discarded before the group member can rejoin the tree.

Figure 4 shows the repair latency and overhead for the case when the source moves. Here, the repair latency is fairly low for all protocols, though ODMRP does particularly well because it uses a mesh. Repair overhead for ASSM is low when it uses DSR for unicast routing, but is much higher with AODV. The increase with AODV arises because it uses an expanding ring search, which incurs greater overhead than a simple network-wide flood if the source has moved a large distance. In general, repair overhead rises for all protocols as the number of group members increases, because this means more members have to find routes when the source moves.

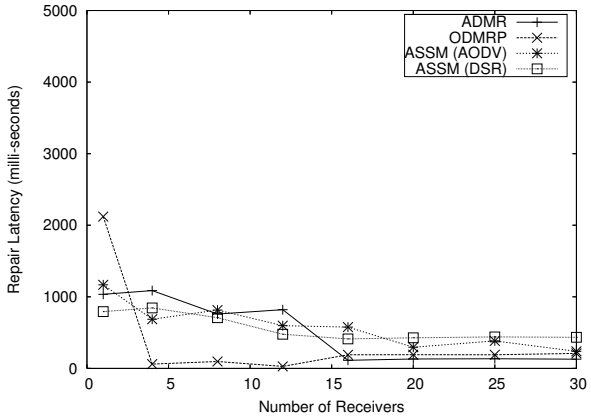


(a) Repair Latency

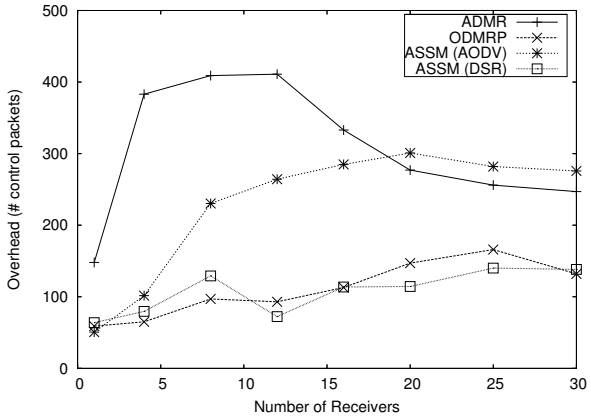


(b) Control Overhead

Fig. 3. Group Member Moves



(a) Repair Latency



(b) Control Overhead

Fig. 4. Source Moves

To test how well these benefits translate when all nodes are mobile, we run additional experiments with 50 mobile nodes in a $1000m^2$ field. In each experiment, we create three multicast groups with 7 members each. Nodes move using the random waypoint model, with a pause time of 30 seconds and a speed of 1 to 50 meters per second. The group leader sends 4 packets a second, using 64-byte packets.

Figure 5 shows that the packet delivery ratio for ASSM (running with AODV) drops to about 80% as speed increases. This is comparable to ODMRP, and slightly lower than ADMR. ASSM's performance is not as good as it could be, because it uses periodic Join messages to repair the tree, which causes packet loss at higher speeds, whereas ADMR monitors packet loss and reacts instantly when loss occurs. Monitoring packet loss and triggering refresh messages should enable ASSM to improve its performance, while maintaining its advantage of lower control overhead. We will address this issue in future work.

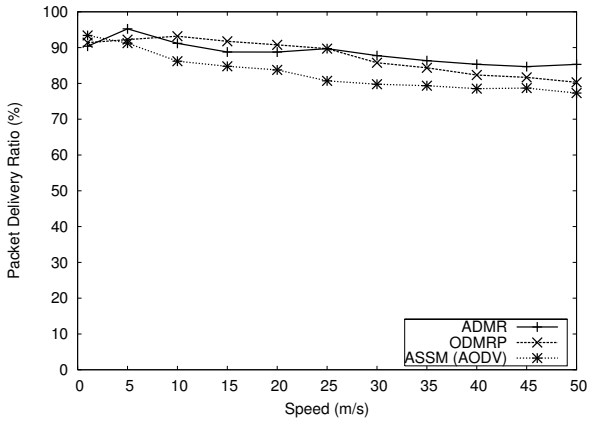
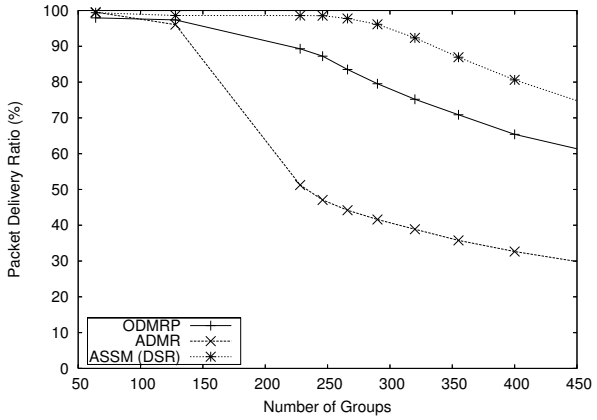


Fig. 5. Packet Delivery Ratio When All Nodes Are Mobile

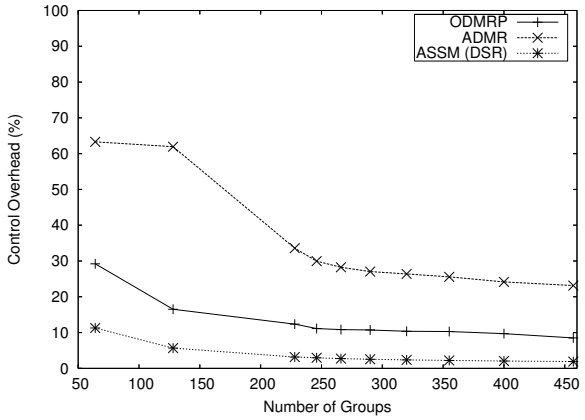
B. Scalability

An important consideration for multicast routing protocols is how well they scale as the number of groups increases. We place 100 nodes in a $1000m^2$ field, then create up to 450 multicast groups. Each group has three members, all located near the group leader. The group leader sends 10 packets a second, using 4 KB packets.

Figure 6 shows the packet delivery ratio and control overhead for ASSM as the number of groups increases. ASSM maintains a high delivery ratio until the large volume of traffic leads to congestion. ODMRP's delivery ratio drops earlier because its use of a mesh is less efficient than a tree. ADMR's



(a) Packet Delivery Ratio



(b) Control Overhead

Fig. 6. Scalability with Groups

delivery ratio drops very early because it switches to flooding when packet loss occurs, and this causes even more packet loss.

ASSM has much lower control overhead than the other protocols because of its reliance on unicast routing. The other protocols require a separate broadcast to find the source of each multicast tree, but ASSM needs only one broadcast per source, regardless of the number of groups. ODMRP has the highest overhead because it continues to periodically flood messages even when none of the nodes move, whereas ASSM and ADMR only react when nodes move.

V. Conclusion

Our design of ASSM demonstrates that it is possible to design a scalable multicast routing protocol for ad hoc networks without using hierarchy or an overlay. Using SSM semantics eliminates much of the overhead previously required for joining a multicast group. Using existing unicast routing protocols enables protocol designers to focus on making one scalable routing protocol that can be used for both unicast and multicast services.

Our work also shows that localizing repair decisions at the group members enables the multicast routing protocol to provide both low repair latency and low repair overhead. We are currently working on improving ASSM's repair latency even further by triggering repairs instead of using a periodic timer. A group member should be able to measure the average inter-packet delay, and then trigger a repair if this delay grows too large, rather than waiting for the next timer expiration.

In the future we plan to examine alternative forwarding methods, and methods for handling multiple sources in the same group. We also plan to study scalability in terms of the network size, group size, and the traffic rate. Finally, we plan to compare ASSM to multicast routing protocols that use a hierarchy or overlay to demonstrate that multicast scales better when it builds on top of scalable unicast routing.

References

[1] S. Lee, W. Su, and M. Gerla, "On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks," in *ACM/Baltzer Mobile Networks and Applications, special issue on Multipoint Communication in Wireless Mobile Networks*, 2000.

[2] J. Jetcheva and D. Johnson, "Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks," in *ACM MobiHoc*, October 2001.

[3] E. Royer and C. Perkins, "Multicast Operation of the Ad-Hoc On-Demand Distance Vector Routing Protocol," in *Mobile Computing and Networking*, 1999.

[4] C. Wu, Y. Tay, and C. Toh, "Ad hoc Multicast Routing Protocol Utilizing Increasing id-numberS (AMRIS) Functional Specification," in *Internet-Draft, draft-ietf-manet-amris-spec-00.txt (work in progress)*, Nov 1998.

[5] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research," *Wireless Communications & Mobile Computing (WCMC)*, 2002.

[6] C. Gui and P. Mohapatra, "Scalable multicasting in mobile ad hoc networks," *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 2119–2129 vol.3, 7-11 March 2004.

[7] P. Sinha, R. Sivakumar, and V. Bharghavan, "MCEDAR: multicast core-extraction distributed ad hoc routing," *IEEE Wireless Communications and Networking Conference, 1999*, pp. 1313–1317 vol.3, 1999.

[8] C. Gui and P. Mohapatra, "Overlay multicast for manets using dynamic virtual mesh," *Wirel. Netw.*, vol. 13, no. 1, pp. 77–91, 2007.

[9] A. Passarella and F. Delmastro, "Usability of legacy P2P multicast in multihop ad hoc networks: an experimental study," *EURASIP J. Wirel. Commun. Netw.*, vol. 2007, no. 1, pp. 38–38, 2007.

[10] M. Ge, S. Krishnamurhty, and M. Faloutsos, "Application versus network layer multicasting in ad hoc networks: the alma routing protocol," *Ad Hoc Networks Journal*, vol. 4, no. 2, pp. 283–300, 2006.

[11] L. Xiao, A. P. Patil, Y. Liu, L. M. Ni, and A.-H. Esfahanian, "Prioritized Overlay Multicast in Mobile Ad Hoc Environments," *IEEE Computer 37(2): 67-74*, 2004.

[12] K. Chen and K. Nahrstedt, "Effective location-guided tree construction algorithms for small group multicast in MANET," in *IEEE INFOCOM*, June 2002.

[13] D. R. Cheriton and H. Holbrook, "EXPRESS Multicast: Making Multicast Economically Viable, ACM SIGCOMM, 1999 ," ACM SIGCOMM, 1999.

[14] M. Pandey and D. Zappala, "A Scenario Based Evaluation of Ad Hoc Multicast Routing Protocols," in *IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks*, June 2005.

[15] M. Pandey, R. Pack, L. Wang, Q. Duan, and D. Zappala, "To Repair or Not to Repair: Helping Routing Protocols to Distinguish Mobility From Congestion," *IEEE INFOCOM MiniSymposia, Anchorage*, 2007.

[16] K. Tang, K. Obraczka, S.-J. Lee, and M. Gerla, "A Reliable, Congestion-Controlled Multicast Transport Protocol in Multimedia Multi-hop Networks ," in *WPMC*, 2002.

[17] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks," in *Workshop on Parallel and Distributed Simulation*, May 1998.