A Zone-Based Shared-Tree Multicast Protocol for Mobile Ad Hoc Networks

Aniruddha Rangnekar[1], Ying Zhang[1], Ali A. Selcuk[2], Ali Bicak[1], Vijay Devarapalli[3] and Deepinder Sidhu[1]

Abstract**—*****This paper proposes a new multicast protocol for mobile ad hoc networks (MANETs). The proposed protocol, Shared-Tree MZR, is a shared tree variant of the Multicast Routing Protocol based on Zone Routing (MZR). The concept of zone-based multicast routing for mobile ad hoc networks was originally proposed in MZR [1]. The new protocol utilizes the advantages of the shared-tree together with the advantages of the zone-based routing. The performance of the protocol is analyzed for various network conditions. The test results show that the new protocol performs well and has significantly low overhead in scenarios with multiple sources.*

Index Terms**—Ad hoc networks, routing protocols, multicast routing, zone routing.**

I. Introduction

Mobile ad hoc networks (MANETs) [2] are self-organizing network architectures in which a collection of mobile nodes, with wireless network interfaces, may form a temporary network without the aid of any established infrastructure or centralized administration. The characteristics that distinguish these networks from wired networks include a distributed peer-to-peer mode of operation, multi-hop routing over wireless links, and relatively frequent changes in topology. In MANETs, it is more effective to use multicast communication, because in a typical ad hoc environment, mobile nodes mostly work as a group and are involved in collaborative computing. Several multicast protocols have been proposed for ad hoc networks. Excluding the basic flooding, they can be grouped under two approaches according to their packet distribution algorithms, namely tree-based and mesh-based protocols.

AMRoute [3], AMRIS [4] and MAODV [5] are tree based protocols, in which a shared tree is created involving the entire multicast group. CAMP [6] and ODMRP [7] allow multiple paths to cope with link failures, resulting in a mesh structure. In ODMRP, the mesh is created using the *forwarding group* concept and a reactive approach is followed to keep the forwarding group current. On the other hand, CAMP exemplifies a proactive mesh based protocol. A hybrid approach is used to control the overhead and provide scalability. Multicast Routing Protocol based on Zone Routing (MZR) [1] follows the hybrid approach by creating a multicast source tree based on the zone routing concept [8]. In a zone routing network, every node maintains a proactive unicast route to every other node within a certain range. Zone routing facilitates the creation of multicast trees and source-based routing enables the usage of the optimal tree for every source and better distribution of the network traffic. Despite its advantages, source-based routing can cause significant overhead when there are many sources in a group. If the nodes in the network are highly dynamic, overhead of maintaining a separate tree for every source can be prohibitively expensive.

In this paper, we propose a shared-tree variant of the MZR protocol (SH-MZR). Following the description of SH-MZR, we present the results of extensive simulation tests carried out to analyze the performance of SH-MZR.

II. Shared Tree MZR

A tree based multicast routing protocol is either a source-tree or a shared-tree protocol. In the family of source tree protocols, multiple trees rooted at the sources of the multicast session are created. This becomes a bottleneck because when a mobile node moves, a large number of source trees might need reconstruction, causing excessive overhead. These drawbacks are overcome by a shared tree protocol. Only one tree is created for a multicast group. Hence the overhead of tree maintenance is reduced as compared to the source tree protocols. Underneath the SH-MZR multicast protocol, a zone routing protocol is run to maintain the zone routes of the nodes in the network.

A. Concept of Zone Routing

The concept of zone routing [8] is a hybrid of proactive and reactive routing protocol components. The scope of the proactive procedure is limited to the node's local neighborhood, called the zone. Each node keeps track of nodes in its zone by running a proactive routing protocol. For routes to destinations outside a node's zone, a reactive route discovery process is initiated.

As the zone radius is significantly smaller than the network radius, the cost of learning the zones' topologies is a very small fraction of the cost required by a global proactive mechanism. Zone routing is also much cheaper (in terms of control traffic and congestion) and faster than a global reactive route discovery mechanism, as the number of nodes queried in the process is very small. Each mobile node participating in an ad hoc network constructs a zone around itself with a pre-configured zone radius. Every node periodically broadcasts an ADVERTISEMENT packet, identifying itself. The advertisement packets are sent once every ADVERTISE_INTERVAL. The propagation of the advertisement packets is restricted to a zone by setting the time-to-live (TTL) value of these packets to

(1) Maryland Center for Telecommunications Research, University of Maryland, Baltimore County. 1000 Hilltop Circle, Baltimore MD 21250. Email: {arangn1,yizhang,bicak}@cs.umbc.edu, sidhu@umbc.edu

(2) Dept. of Computer Engineering, Bilkent University, 06533, Ankara, Turkey. Email: selcuk@cs.bilkent.edu.tr

(3) Nokia Research Center, 313 Fairchild Drive, Mountain View, CA 94043. Email: vijayd@iprg.nokia.com

the zone radius. When a node B receives the advertisement packet from A, a route entry for A is created and stored in B's zone routing table with the appropriate value of hop count. A soft state approach is followed to remove stale routes from the zone routing table.

The zone routing table is kept up-to-date through this proactive protocol built on periodic advertisements. If the distance metric in a route entry is equal to the zone radius, the corresponding destination becomes a border node. All the other nodes are the interior zone nodes.

B. Multicast Tree Creation and Maintenance

In SH-MZR, a multicast routing tree is created when a source node wants to send data to the members of a group. There are two stages for the tree creation:

Search for the existing tree: A node, that wants to join the multicast group, will first look for the existing tree. The node sends a REQUEST message to the nodes within its zone. If the tree exists, a node on the tree will reply back by sending a REPLY message. If the sender does not get the REPLY message within an interval of WAIT_REPLY, it unicasts a REQUEST_PROPAGATE message to the border nodes to extend the search inside their zones. The border nodes would now send the REQUEST to all nodes in their zones. If any node in the border node's zone is in the tree, it would reply to the border node with a REPLY message. The border node will then send a REPLY to the original source Otherwise, the same procedure would be followed to extend the search through the network.

Only a node which is part of the tree can send a REPLY message. The REPLY is unicast using the reverse route maintained by each intermediate border node. The intermediate border nodes also maintain the reverse route of the REPLY packet. The original source node might get more then one REPLY message. It will select the first one and send an ACTIVATE message to the node from which it received the REPLY message. The ACTIVATE message will activate the tree link between the request node and the node which sent the REPLY message.

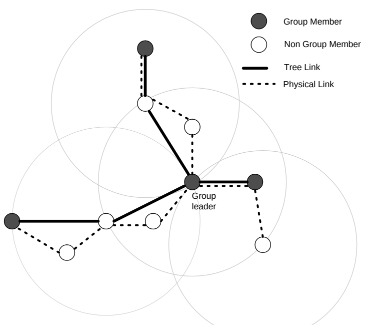

Fig. 1. Tree Extension in SH-MZR

Initiate tree creation: After sending REQUEST_PROPOGATE message, the sender will wait for WAIT_PROPAGATE_REPLY seconds. If it does not get a reply within this time interval the search is repeated REQUEST_RETRY times. If it still does not receive any replies, it assumes that there is no tree in the network and claims itself to be the group leader. The group leader is responsible for maintaining the multicast tree.

The group leader broadcasts a TREE_CREATE message within its zone. When a zone node, interested in the multicast group session, receives the TREE_CREATE packet, it creates a multicast route entry and adds the source as its upstream node. It then replies to the source with a TREE_CREATE_ACK packet. After receiving the TREE_CREATE_ACK, the source node adds the sender to its downstream list in the multicast route entry. This mechanism allows the source to create a multicast tree, rooted at the source and extending throughout its zone.

Once the source is done with its zone, it tries to extend the multicast tree to the entire network. The source unicasts a TREE_PROPOGATE message to the border nodes for its zone. When a border node receives a TREE_PROPOGATE packet, it creates a multicast route entry for the session, and then sends a TREE_CREATE packet to all its zone nodes. If a node in the border node's zone is interested in the session, it replies to the border node with a TREE_CREATE_ACK. The border node in turn unicasts a TREE_CREATE_ACK packet to the source. This basically extends the multicast tree into the border node's zone with a unicast link between the source and the border node and multiple tree branches within the border node's zone. Once the border node is done with its zone nodes, it sends a TREE_PROPOGATE message to all its border nodes. These border nodes in turn try to extend the multicast tree within their zones. This continues until every node in the MANET gets a TREE-CREATE packet corresponding to the multicast tree being created. The tree thus created can be seen in fig. 1.

Data Transmission: The source starts transmitting data packets to the group members once the multicast delivery tree is created. When a node on the multicast tree receives a data packet, it replicates the data packet and sends a copy to its upstream and each node in the downstream list other than the link on which it received the packet.

Tree Refreshment: Periodically, the group leader sends the TREE-REFRESH message. On receiving this message, the tree nodes will refresh the tree information and update the group leader information.

Link Failure: Node mobility can cause frequent link breaks in the multicast delivery tree. The downstream node is responsible for detecting link breaks and reconfiguring the tree. The node A (that lost connection to its upstream node) initiates a search for the multicast tree by using the zone routing mechanism. It broadcasts a REQUEST to the nodes in its zone. This message also contains the distance from this downstream node to the group leader in terms of hops. If any node in A's zone is on the multicast tree and its distance to the group leader is less than the distance advertised , it sends a REPLY packet to A. Before sending the REPLY packet, it adds the node from which it received the request to the list of downstream nodes in the multicast route entry. If the node A receives multiple REPLY packets, it will send an ACTI-

VATE message responding to the first REPLY. If node A does not get a reply from its zone nodes, it tries to propagate its search through the entire network by sending a REQUEST_PROPOGATE packet to all its border nodes. These border nodes in turn search their zone. If they get a response from any of their zone nodes, they send a REPLY to A. If not, they propagate the search to their border nodes. Using this mechanism, node A's request may be propagated to the entire network. If A does not get a reply at all, it assumes that the network has been partitioned.

Tree Partition: A node tries to repair the broken link REQUEST_RETRIES times. If the link can not be repaired, it assumes that there is a network partition. Then the node claims itself to be the group leader of the part of tree that is on this side of the network partition. It then sends periodic TREE-REFRESH packets to inform other nodes that there is a new group leader.

Tree Merging: If a node gets two TREE_REFRESH messages from different group leaders, it indicates that the partitioned tree could be reconnected again. This node can initiate the tree merge, if it lies on the tree whose group leader's node ID is smaller of the two. It request permission from its group leader to repair the tree by sending a REQUEST packet, with the "permission" flag set, to its group leader (GL1). If the group leader receives multiple packets, it grants permission to only one such request. It sends a REPLY message, with permission flag set, in response to the first REQUEST message. After receiving this REPLY, this node sends a REQUEST message with the "merge" flag set to the group leader of the other tree (GL2). GL2 responds by sending a REPLY packet with the merge flag set. This reply is sent to the node initiating the merge, and creates a tree link between the two partitioned trees. Then the node sends a message to GL1 saying that the tree merge is complete. As this message traverse up to the tree, the direction of the tree links are reversed. GL2 becomes the group leader of the merged tree, and sends a TREE_REFRESH message to update the group leader information throughout the tree. Fig. 2 shows the tree merge procedure.

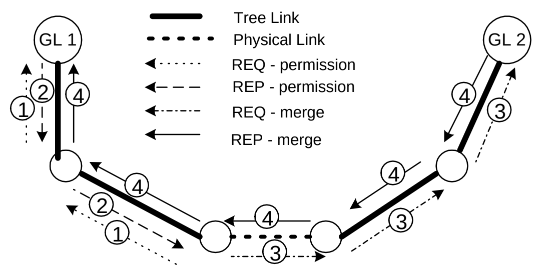

Fig. 2. Tree Merge in SH-MZR

Tree Pruning: Only a leaf node can prune itself by sending a REQUEST message with the "prune" flag set. When a node receives this REQUEST message, it removes the sender from its list of downstream nodes. If the sender is the group leader, then this node assumes itself to be the new group leader, and then sends a TREE_REFRESH message to update the group leader information.

III. Simulation and Results

We carried out extensive simulation tests to analyze the performance of the shared-tree MZR protocol. In this section we describe the simulation model and summarize the results of the simulations.

A. The Simulation Model

The simulation tests were performed on the NIST Network simulator [9] developed at the National Institute of Standards and Technology. To evaluate the performance of the multicast routing protocol, we setup a packet-level simulation, which allowed us to observe and measure the performance of the protocol under a variety of conditions.

Each mobile node is defined by its position and moves around on a flat two-dimensional grid. Nodes in the simulation move according to the "random waypoint" model [10]. The movement scenarios are characterized by a *pause time* and *distance* between successive positions of a node. Pause time is the interval that a node remains stationary. The distance between the old and new position is distributed uniformly between a minimum and maximum value. The wireless link is characterized by the link distance and the link bandwidth. The transmission range in our model is set to 100 meters. The wireless link capacity is assumed to be 2 Mbps. The link transmission delay, being dependent on link capacity and packet size, works out to be 2 ms for a packet of size 500 bytes. A separate module generates group information and supplies it to the mobile nodes. A wireless application created at the source generates multicast data for the group members at a constant data rate of 64 Kbps.

The variable simulation parameters include the number of nodes in the system, maximum distance between successive positions of a node, pause time, and the number of sources in a group.

B. Simulation Metrics

We analyze the protocol over two performance measures: packet delivery ratio and protocol overhead.

B.1 Delivery Ratio

Packet delivery ratio is the ratio of the number of data packets actually delivered to the multicast group members to the total number of data packets that were supposed to be received. A measure of this ratio tell us how many packets were lost and not received. A number of factors like node mobility, pause time and erroneous transmissions could be responsible for the packet loss.

B.2 Protocol Overhead

The protocol overhead is calculated to include both the control packets and the data packets. Counting the control overhead without the data transmission overhead is not sufficient, because it is always possible to reduce the control overhead by increasing the data transmission.

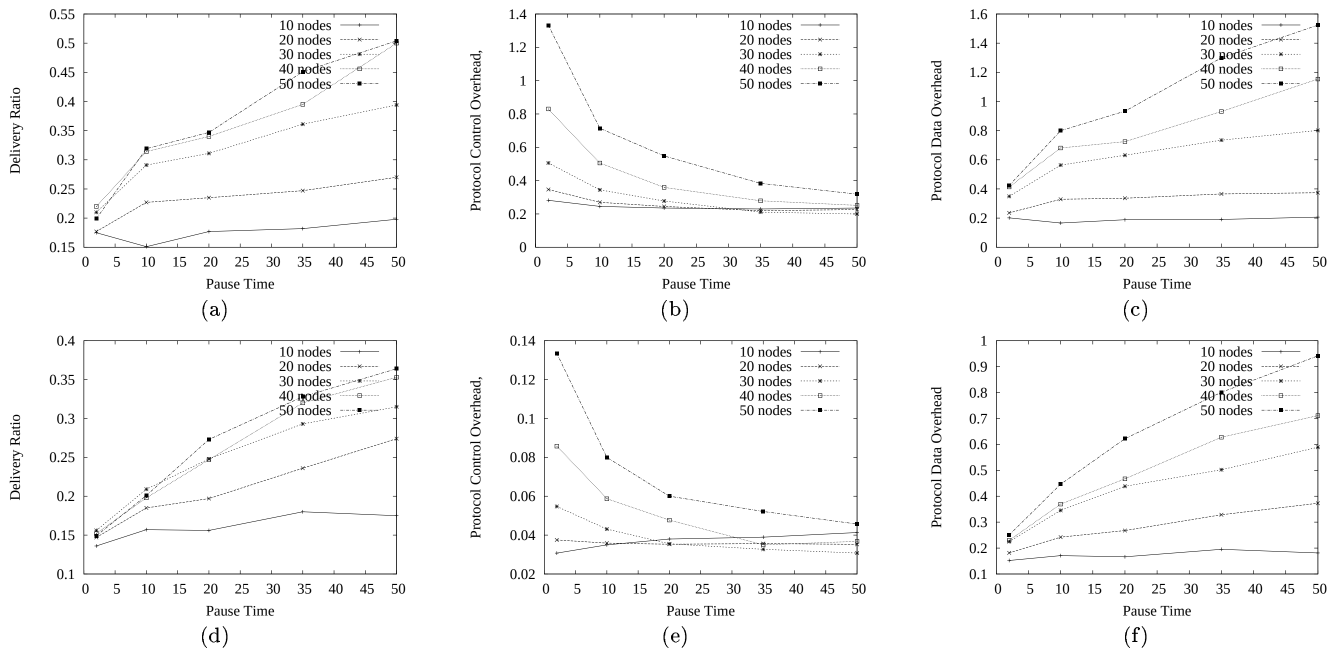

Fig. 3. Performance of Shared Tree MZR. Max distance = 50 mts. (a) Delivery ratio, single source. (b) Control Overhead, single source. (c) Data Overhead, single source. (d) Delivery ratio, six sources. (e) Control Overhead, six sources. (f) Data Overhead, six sources

Control overhead is calculated as the ratio of control packets generated to the total data packets generated by the source(s). Mobile nodes create link breakages in the multicast tree and therefore more branch reconstructions. Since tree reconstruction involves control traffic, node mobility is a major factor that influences control overhead.

Data overhead is important as it determines the efficiency of the multicast delivery structure. It gives a measure of how many non-group-member nodes are present in the data delivery structure. It is calculated as a ratio of data packets received by the nodes in the delivery structure to the product of number of group members and the number of data packets generated by the source(s). Data overhead is indirectly influenced by node mobility.

C. Discussion of the Test Results

The results for SH-MZR consist of two sets of simulations. Fig. 3 represents the first set of simulations where the maximum distance between consecutive positions of a node is fixed at 50 meters.

Fig. 3(a)-(c) show the graphs for various metrics as the pause time is varied from 2 seconds to 50 seconds. For these experiments, a single multicast session with one source was considered. The packet delivery ratio is less for highly mobile nodes. It increases as the pause time increases, ie. as mobility reduces. Another observation from fig. 3(a) is that increase in the number of nodes increases the packet delivery ratio. This is because the dynamic nature of the network topology also depends on where the nodes are located. If the number of nodes is small, then the network is sparsely populated and hence network connectivity is low. There may even be network partitions. As the number of nodes increases, the connectivity increases and hence the packet delivery ratio also increases. Fig. 3(b) shows the variance in routing control overhead as the pause time is varied. Increase in pause time implies an increasingly static network. As mobility reduces, link breakages are rare and therefore the need to repair broken tree links is less. Thus the routing control overhead decreases. Fig. 3(c) shows that the graph for protocol data overhead is quite similar to the graph for delivery ratio. This is due to the fact that reduction in link breakages increases the number of group members present in the tree and reachable from the source. This has a twin effect of increasing delivery ratio as well as data overhead. Fig. 3(d)-(f) show the graphs for the same experiments but the number of sources in the multicast session is increased to six. It can be seen from fig. 3(d) and 3(f) that the protocol delivery ratio and the protocol data overhead do not change significantly as the number of sources is increased. But there is a drastic decrease in the control overhead. This is due to the fact that the data is delivered by a shared tree. The control overhead is divided over the multiple sources. Thus as the number of sources increases, the divided overhead decreases.

Fig. 4 represents the second set of simulations. Here the pause time is kept constant and the maximum distance between two consecutive positions of a node is varied. Fig. 4(a)-(c) show the graphs for multicast session with a single source. Packet delivery ratio is very high for a static network. But as the nodes move further away, packet delivery ratio drops drastically. Fig. 4(a) also shows that for a particular distance and pause time, delivery ratio increases as the number of nodes increase. This happens due to increase in network connectivity. Fig. 4(b) shows that control overhead decreases with reduced mobility. The protocol control overhead also increases with an increase in

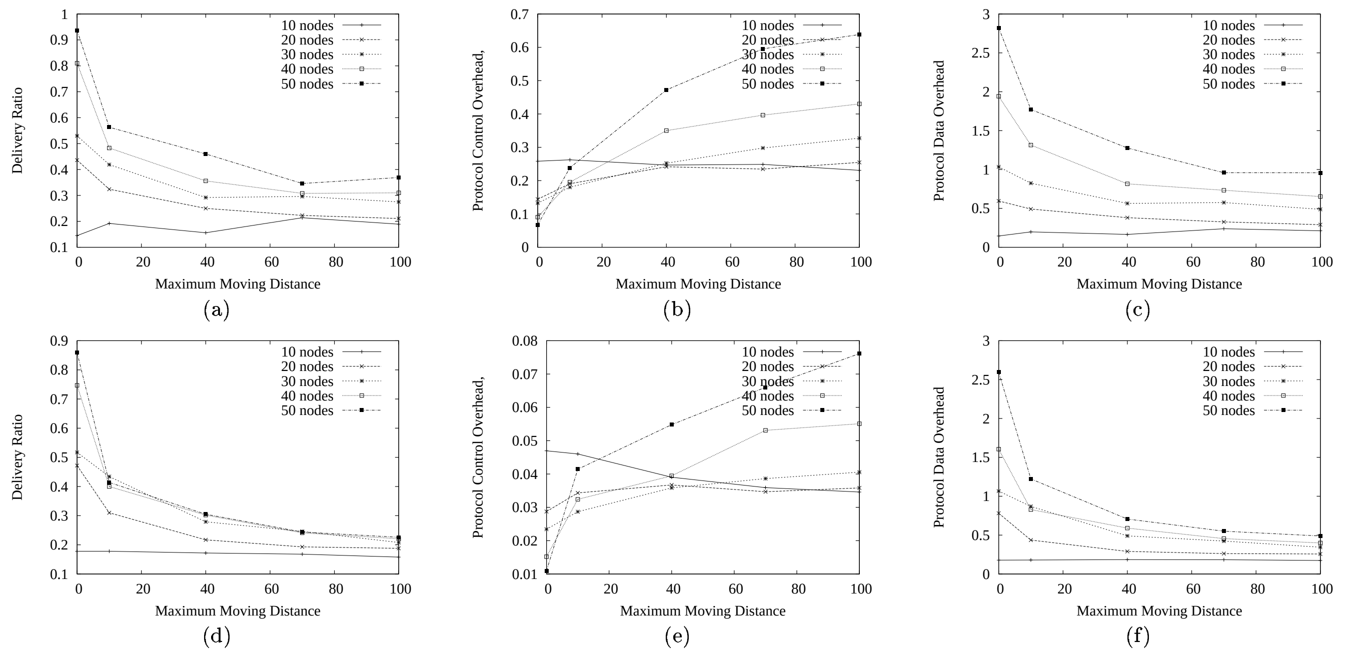

Fig. 4. Performance of Shared Tree MZR. Pause time = 20 sec. (a) Delivery ratio, single source. (b) Control Overhead, single source. (c) Data Overhead, single source. (d) Delivery ratio, six sources. (e) Control Overhead, six sources. (f) Data Overhead, six sources

the number of nodes in the network. Fig. 4(b) shows an anomaly for the case of ten nodes. The control overhead in the nearly static network (low maximum distance) is abnormally high. This is due to the sparse nature of the network. If there are partitions in the network, multiple trees will be created. These tree might need to be merged later. Maintenance of these multiple trees causes the above mentioned abnormality. Fig. 4(d)-(f) show the results when the same experiments were conducted for a multicast session with six sources. There is not a significant change in the graphs of delivery ratio and data overhead. But the control overhead is reduced considerably. This is due to the fact that all six sources use the same tree and the tree maintenance cost, ie. the control overhead, is distributed over the different sources.

Remark. It should be noted that the simulations given here are specific to the model described in Section IIIA, where there is a relatively high rate (64 Kbps) and continuous data transmission. Also the node mobility in these simulations is high. It is impossible to give the test results for all possible multicast models here due to the space limitations. The results described here should be taken as an analysis for multicast sessions with relatively high data rate and continuous transmission.

IV. Conclusions

In this paper, we proposed a new protocol for multicasting in mobile ad hoc networks which deploys a shared-tree maintained on zone-based unicast routes. Use of a shared-tree reduces the control overhead for the tree creation and maintenance in groups with multiple sources. The zone-based routing increases the robustness of the multicast tree in face of moving nodes (i.e. reduces the chance of links being broken in the tree) and enables more rapid recovery when a link is broken.

We performed extensive simulations to test SH-MZR. The test results showed that SH-MZR protocols performs quite well except in cases where network is extremely sparse. Future extensions to this work will include a comparison of SH-MZR with other multicast protocols.

References

[1] V. Deverapalli and D. Sidhu, "MZR : A multicast protocol for mobile ad hoc networks," *IEEE International Conference on Communications*, vol. Proceeedings of ICC 2001, June 2001.

[2] S. Corson and L. Macker, "Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations," *RFC 2501*, January 1999.

[3] M. Liu, R. R. Talpade, A. McAuley, and E. Bommaiah, "AM-Route: Adhoc multicast routing protocol," *Technical Report*, vol. TR 99-8, The Institute for Systems Research, Univesity of Maryland, 1999.

[4] C.W. Wu and Y.C. Tay, "AMRIS: A multicast protocol for ad hoc wireless networks," in *Proceedings of IEEE MILCOM'99*, November 1999.

[5] Elizabeth M. Royer and Charles Perkins, "Multicast operation of the ad-hoc on-demand distance vector routing protocol," in *Proceedings of MobiCom'99*, August 1999.

[6] J.J Garcia-Luna-Aceves and E.L. Madruga, "The core-assisted mesh protocol," *IEEE Journal on Selected Areas in Communication*, vol. 17, no. 8, August 1999.

[7] S.-J. Lee, M. Gerla, and C.-C. Chiang, "On-demand multicast routing protocol," in *Proceedings of IEEE WCNC'99*, September 1999.

[8] Z.J. Haas and M.R. Pearlman, "The zone routing protocol (ZRP) for ad hoc networks," *Intenet-Draft*, vol. draft-ietf-manet-zone-zrp-02.txt, June 1999.

[9] N. Golmie, F. Mouveaux, L. Hester, Y. Saintillan, A. Koenig, and D. Su, "The NIST ATM/HFC network simulator," *Operation and Programming Guide*, December 1998.

[10] J. Broch D. B. Johnson and D. A. Maltz, "The dynamic source routing protocol for mobile ad hoc networks," *Intenet-Draft*, vol. draft-vijay-manet-dsr-03.txt, October 1999.