An Adaptive Redundancy Protocol for Mesh Based Multicasting

Ravindra Vaishampayan
Department of Computer Science
University of California Santa Cruz
Santa Cruz, CA 95064
ravindra@soe.ucsc.edu
Phone: 203-988-1826
Fax: 203-876-0700

J.J. Garcia-Luna-Aceves
Department of Computer Engineering
University of California Santa Cruz
Santa Cruz, CA 95064
jj@soe.ucsc.edu

Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304

Katia Obraczka
Department of Computer Engineering
University of California Santa Cruz
Santa Cruz, CA 95064
katia@soe.ucsc.edu

***Abstract*— Mesh-based multicast routing protocols for multi-hop wireless ad hoc networks (MANETs) build multiple paths from senders to receivers. Higher redundancy results in higher reliability because packets can be delivered even in the presence of links breaking. However, in less dynamic environments, in which links break less frequently, the additional redundancy may not be needed in terms reliability and may significantly increase overhead. This paper investigates the tradeoffs between reliability and efficiency in mesh-based multicast protocols for MANETs. We introduce an adaptive mesh-based multicast mechanism that controls mesh redundancy based on link reliability in the neighborhood of the node. Mesh redundancy is measured by the number of paths from each receiver to the core of the group's mesh, which serves as the address of the group. We introduce a metric called Mesh Reliability Index (MRI), which allows nodes to estimate the reliability of the mesh in their neighborhood, and determine whether redundancy needs to be increased or decreased.**

Through simulations, we compare the performance of the adaptive mesh-building protocol against the non-adaptive version (which builds the mesh with maximum redundancy), and against ODMRP for a wide range of scenarios with varying mobility, group members, number of senders, traffic load, number of multicast groups and terrain size. Our results show that adjusting mesh redundancy based on link reliability can maintain high packet delivery ratios with less overhead compared to non-adaptive mesh-based multicast protocols.

***Keywords*— Ad hoc networks, routing, multicasting, multicast mesh, multicast tree.**

I. Introduction

[1] The objective of a multicast routing protocol for mobile ad hoc networks (MANETs) is to support the dissemination of information from a sender to all multicast group members while trying to use the available bandwidth efficiently in the presence of frequent topology changes. In MANETs, node mobility and the fact that wireless links are more prone to transmission errors result in higher packet drop probability when compared to wired networks. Therefore, multicast routing protocols that provide route redundancy (i.e., routing packets along multiple paths from source to receivers), typically outperformed multicast routing mechanisms that offer no redundancy. Based on this observation, numerous MANET multicast protocols that provide route redundancy by building a *mesh* connecting senders to receivers have been proposed including [1], [2], [3], [4], [5]. Section II describes this prior work. Because MANET nodes typically have limited power, processing, storage, and communication capabilities, redundancy can incur significant overhead in these resource-constrained networks, even though it provides higher packet delivery ratios.

The contributions of this paper are two-fold. First, in Section III, we introduce an adaptive mesh-based multicast mechanism that controls mesh redundancy based on link reliability. We introduce a metric called Mesh Reliability Index (MRI) for this purpose. This is motivated by experimental results showing that the impact of redundancy is directly correlated to the reliability of network links. In other words, when link reliability is high, having larger redundancy in the network does not significantly increase packet delivery ratio. However, when links are more volatile, having greater redundancy does have a considerable impact on the packet delivery ratio.

The proposed adaptive mesh-based multicast mechanism controls mesh redundancy based on link reliability. It elects a core and every node in the network knows its distance to the core based on multicast announcements exchanged periodically by every node. Neighbors of a node with a shorter distance to the core are considered its parents. Mesh redundancy is measured by the number of paths from each receiver to the mesh's core. Every node in the network estimates reliability in its neighborhood as described in Section III-G.1. Mesh redundancy is increased as reliability decreases.

The process of mesh building is receiver initiated. A node controls the amount of redundancy in the mesh depending on the reliability of its links in its neighborhood. If the reliability of the links is very low, the number of parents included in the mesh are increased. If the reliability of the links is high enough, the the number of parents included in the mesh is decreased. If the redundancy is appropriate for the measured reliability of links, then the number of parents included in the mesh is left unchanged.

The other contribution of the paper is to investigate the tradeoffs between reliability and efficiency in mesh-based MANET multicast protocols. To this end, we compare the performance of adaptive mesh-based multicast against its non-adaptive version and ODMRP [3] in Section IV, a widely-known mesh-based MANET multicast protocol. Through extensive simulations using a wide range of MANET scenarios with varying mobility, traffic load, group members, number of senders, number of multicast groups and terrain size, we show that adjusting mesh redundancy based on link reliability in the neighborhood can maintain high packet delivery ratios at higher protocol efficiency

[1]This work was supported in part by the Jack Baskin Chair of Computer Engineering at UCSC and by the National Science Foundation under grant CNS-0435522

when compared to non-adaptive mesh-based multicast protocols.

II. Related Work

As many of the possible applications for ad hoc networks e.g. battlefield scenarios, search and rescue operations, policing, firefighting etc. have the need for one to many communication, there has been considerable research in multicasting in ad hoc networks. Over the years, several multicast routing protocols have been proposed for MANETs [1], [2], [3], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [4], [16], [5]. For the purposes of our discussion, the approaches taken to date can be classified into tree-based and mesh-based approaches.

A tree-based multicast routing protocol establishes and employs either a shared multicast routing tree or multiple source-based multicast routing trees (one for each group source) to deliver data packets from sources to receivers of a multicast group. Recent examples of tree-based multicast routing approaches are the multicast ad hoc on-demand distance vector protocol (MAODV) [7], and the adaptive demand-driven multicast routing protocol (ADMR) [11]. In contrast, a mesh-based multicast routing protocol maintains a mesh consisting of a connected component of the network containing all the receivers of a group. Two well-known examples of mesh-based multicast routing protocols are the protocol for unified multicasting through announcements (PUMA) [1] and the on-demand multicast routing protocol (ODMRP) [3].

MAODV maintains a shared tree for each multicast group, consisting of only receivers and relays. Sources wishing to send to the group acquire routes to the group on demand in a way similar to the ad hoc on demand distance vector (AODV) [17] protocol. Each multicast tree has a group leader, which is the first node to join the group in the connected component. The group leader in each connected component periodically transmits a group hello packet to become aware of reconnections. Receivers join the shared tree with a special route request. The route replies coming from different multicast tree members specify the number of hops to the nearest tree member. The node wishing to join the tree joins through the node reporting the freshest route with the minimum hop count to the tree. We have shown [1] that the performance of MAODV is very good for small groups, low mobility, and light traffic loads, but its performance degrades sharply once a given value of group size, mobility, or traffic load is reached. This is due to a sharp increase in the MAODV control packets transmitted to maintain the multicast tree of a group, as well as the lack of redundancy.

ADMR [11] maintains source-based trees, i.e., a multicast tree for each source of a multicast group. A new receiver performs a network-wide flood of a multicast solicitation packet when it needs to join a multicast tree. Each group source replies to the solicitation, and the receiver sends a receiver join packet to each source answering its solicitation. An individual source-based tree is maintained by periodic keep-alive packets from the source, which allow routers to detect link breaks in the tree by the absence of data or other control packets. A new source of a multicast group also sends a network-wide flood to allow existing group receivers to send receiver joins to the source. MZR [16] like ADMR, maintains source based trees. MZR performs zonal routing; hence, the flooding of control packets is less expensive. Compared to approaches based on shared trees, the use of source-based trees creates much more state at routers participating in many groups, each with multiple sources.

ODMRP requires control packets originating at each

source of a multicast group to be flooded throughout the ad hoc network. The control packet floods help repair the link breaks that occur between floods. Receivers on receiving these announcements from the senders send out their own announcements which are propagated along the reverse path resulting in the creation of the mesh. This mesh construction scheme in ODMRP leads to the inclusion of every shortest path between every sender and every receiver within the mesh. The limitations of ODMRP are the need for network-wide packet floods and the excessive redundancy of the mesh.

DCMP [4] is an extension to ODMRP that designates certain senders as cores and reduces the number of senders performing flooding. NSMP [5] is another extension to ODMRP aiming to restrict the flood of control packets to a subset of the entire network. However, DCMP and NSMP fail to eliminate entirely ODMRP's drawback of multiple control packet floods per group.

PUMA on the other hand elects a core and all receivers connect to the core along *all* shortest paths creating a mesh which connects all receivers together. Control packet overhead is reduced significantly in PUMA compared to ODMRP because only the core sends announcements as opposed to ODMRP where every sender floods announcements. PUMA as described in [1] also leads to the construction of a more efficient mesh as the construction of the mesh is receiver initiated as opposed to sender initiated in ODMRP.

As described in [1] mesh based protocols ODMRP and PUMA in most cases achieve considerably better packet delivery ratio than tree based protocol MAODV, due to redundancy. Redundancy, however has a cost associated with it, both in terms of energy as well as bandwidth due to the forwarding of a greater number of packets. However there are numerous situations when link breakages are low as a result of which even protocols which do not provide redundancy have excellent packet delivery ratios. In such situations the extra cost incurred by redundant mesh based protocols like ODMRP and PUMA is unnecessary and often wasteful.

III. Adaptive Mesh-Based Multicast

A. Overview

We build upon our work on PUMA [1], which supports the IP multicast service model where sources need not: (1) know the constituency of multicast groups to which they send data and (2) join a multicast group in order to send data packets to the group.

Like CAMP [2] and MAODV [7], we use a receiver-initiated approach in which receivers join a multicast group using the address of a special node (core in CAMP or group leader in MAODV), without the need for network-wide flooding of control or data packets.

We implement a distributed algorithm to elect one of the receivers of a group as the core of the group and to inform each router in the network of at least one next-hop to a group's elected core.

Every receiver connects to the elected core along several shortest paths between the receiver and the core. This number is varied depending on the redundancy desired in the mesh. All nodes on the selected shortest paths between any receiver and the core collectively form the mesh. A sender sends a data packet to the group along *any one* shortest path between the sender and the core. When the data packet reaches a mesh member, it is flooded within the mesh, and nodes maintain a packet ID cache to drop duplicate data packets.

Signaling uses a single control message, or *multicast announcements*. Each multicast announcement specifies a

sequence number, the address of the group (group ID), the address of the core (core ID), the distance to the core, a mesh membership code, and a parent. Successive multicast announcements have higher sequence numbers than previous multicast announcements sent by the core. With the information contained in such announcements, nodes elect cores, determine the routes for sources outside a multicast group to unicast multicast data packets towards the group, join and leave a group's mesh, and maintain the mesh.

B. Core Election

When a receiver joins the group, it checks to see if it has ever received a multicast announcement for that group. If so, then it does not participate in core election and the current core of the group remains unchanged. On the other hand, if it has never received a multicast announcement for that particular group, then it considers itself the core of the group and starts transmitting *multicast announcements* periodically to its neighbors having itself as the core of the group and a 0 distance to itself. Nodes only propagate multicast announcements with the highest core ID they receive from their neighbors. Eventually, each connected component has only one core. If one receiver joins the group before other receivers, then it becomes the core of the group. If several receivers join the group concurrently, then the one with the highest ID becomes the core of the group within a finite time.

Core election is also held if the network is partitioned. The partition that does not have the currently elected core performs the election. A node detects a partition if it does not receive a fresh core announcement for a period of time corresponding to $3 \times multicast\_announcement\_interval$. Once a receiver detects a partition, it behaves in exactly the same way it would upon joining the group and participates in the core election.

C. Propagation of Multicast Announcements

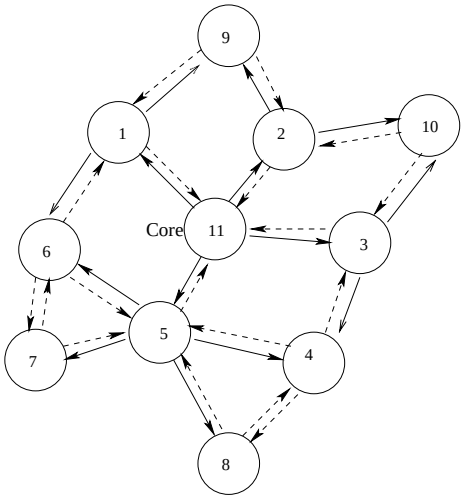
A node that believes itself to be the core of a group transmits multicast announcements periodically for that group. As the multicast announcement travels through the network, it establishes a *connectivity list* at every node in the network. When multiple groups exist, nodes keep a connectivity list for every group. Using connectivity lists, nodes are able to establish a mesh, and route data packets from senders to receivers.

A node stores the data from all the multicast announcements it receives from its neighbors in the connectivity list. Fresher multicast announcements from a neighbor (i.e., one with a higher sequence number) overwrite entries with lower sequence numbers for the same group. Hence, for each multicast group, a node only keeps in its connectivity list the latest information from a particular neighbor for a given core.

Each entry in the connectivity list, in addition to storing the multicast announcement, also stores the time when it was received, and the neighbor from which it was received. The node then generates its own multicast announcement based on the entries in its connectivity list.

During core election, on receiving a multicast announcement with higher core ID, all entries in the connectivity list with a lower core ID are erased. For the same core ID, only multicast announcements with the highest sequence number are considered valid. Hence all valid entries in the connectivity list at any point of time have the same core ID and sequence number. Among these valid entries, the entries with a shortest distance to core qualify as *best entries*, and the neighbors corresponding to these entries are called *parents*.

The connectivity list stores information about one or



Connectivity List at node 6
Core Id = 11 Group Id = 224.0.0.1, Seq No 79

Neighbor	Multicast Announcement	Time
	Distance To Core	(ms)
5	1	12152
1	1	12180
7	2	12260

Fig. 1. Dissemination of multicast announcements

more routes that exist to the core. Figure 1 illustrates the propagation of multicast announcements and the building of connectivity lists. The solid arrows indicate the neighbors from which a node receives its best multicast announcements. For example, node 6 has three entries in its connectivity list for neighbors 5, 1, and 7. Both nodes 5 and 1 qualify as parents because they both report a shortest distance to core of 1. As described in Section III-A a multicast announcement has the following fields : Group ID, Core ID, Sequence Number, Distance to Core, Mesh Membership Code, and Parent ID. A multicast announcement generated by a node has the same Core ID, Group ID and Sequence Number of its parent(s). The reported distance to core is one more than the distance to core of its parents. A node chooses one of its parents to set its parent field, which we refer to as the *chosen parent*. The manner in which this selection is made determines the amount of redundancy in the mesh and is discussed in Section III-G. A node sets its membership code field based on whether it is a non-member, only a receiver, only a mesh-member or both and is described in Section III-D.

When a node wants to send a multicast data packet to the group it encapsulates it within a unicast packet and forwards it to one of its parent. If that link is broken then it chooses another one, till it runs out of parents or the transmission is successful. Hence each node in the network has one or more routes to the core. The multicast announcement sent by the core has distance to core set to zero and parent field set to *invalid_address*, because the core has no parents.

In our current implementation, multicast announcements are generated by the core every three seconds. After receiving a multicast announcement with a fresh sequence number, nodes wait for a short period (e.g. 100 ms) to collect multicast announcements from multiple neighbors before generating their own multicast announcement.

When multiple groups exist, nodes aggregate all the fresh multicast announcements they receive, and broadcast them periodically every *multicast_announcement_interval*. However, multicast announcements representing groups being heard for the first time, resulting in a new core, or resulting in changes in the mesh membership code are forwarded immediately, without aggregation. This is to avoid delays in critical operations, like core elections and mesh establishment.

D. Mesh Establishment and Maintenance

Mesh redundancy is controlled depending on the reliability of network links. The amount of redundancy is selected by having nodes adjust the number of parents to be included in the mesh. Mesh establishment is initiated by receivers. A receiver decides on the number of parents it wants to include in the mesh depending on link reliability measured

by the receiver. (see Section III-G). It then sets the parent field of its multicast announcement so that the number of parents with node ID greater than or equal to its parent field equals the number of parents it wants to include in the mesh. Receivers start by setting mesh membership code to 1 in the multicast announcements they send.

All non-receiver nodes start by setting their mesh membership code to 0. Nodes consider themselves mesh members and increment their mesh membership code by 2 if they have at least one *mesh child* in their connectivity list. A neighbor in the connectivity list is a mesh child if: (1) its mesh membership code is greater than 0; (2) the distance to the core of the neighbor is larger than the node's own distance to the core; (3) its parent field is less than or equal to the node's ID; and (4) the multicast announcement corresponding to this entry was received within a time period equal to two multicast announcement intervals. Condition (4) is used to ensure that a neighbor is still in the neighborhood.

Hence the mesh membership code in each multicast announcement can be set to one of the four values, namely:

- 0 which implies that the node is neither a receiver nor a mesh member
- 1 which implies that the node is a receiver but not a mesh member
- 2 which implies that the node is a mesh member but not a receiver
- 3 which implies that the node is both a receiver as well as a mesh member.

Non-receivers set the parent field in their multicast announcements similarly to receivers, based on the number of parents they want to include in the mesh.

If a node has a mesh child and is hence a mesh member, then it means that it lies on a shortest path from a receiver to the core. As illustrated in Figures 2 and 3, node 50 is elected as the core and all nodes in the network know their distance to the core by adding one to their parents' distance to the core as advertised in the connectivity list (see Section III-C). Figures 2 and 3 depict the mesh constructed when mesh members/receivers include one and all parents in the mesh, respectively. The X and Y axes in these figures refer to the X and Y coordinates in meters, of the various nodes. The numbers in parentheses alongside nodes 42, 48, 44, 27, 29, 19 and 50 indicate their distance to the core. Both figures depict the mesh state of the protocols exactly 20 seconds after the beginning of Experiment 1 as described in Section IV-A.

Node 42 with a distance to core of 3 has three parents in its connectivity list viz. nodes 48, 27, 44. When it wants to include only one of its parents in the connectivity list as shown in Figure 2, it sets the parent field in its multicast announcement to 48. The parents viz. nodes 48, 27 and 44 receive the multicast announcement from node 42 with membership code = 1 and distance to core (3) greater than their own distance to core (2). However node 42 qualifies to be a mesh child of only node 48 because the parent field of its multicast announcement is equal to 48. Hence node 48 becomes a mesh member and nodes 27 and 44 do not. Similarly node 48 which is now a mesh member has three parents in its connectivity list viz. nodes 8, 19 and 29. It sets the parent field in its multicast announcement to 29 which makes node 29 become a mesh member but not nodes 19 and 8. Node 8 eventually becomes a mesh member because of node 46.

Although each receiver/mesh member in Figure 2 includes only one parent as part of the mesh, occasionally more than one parent may eventually be included in the mesh. (e.g., nodes 8 and 29 which are parents of node 48 as

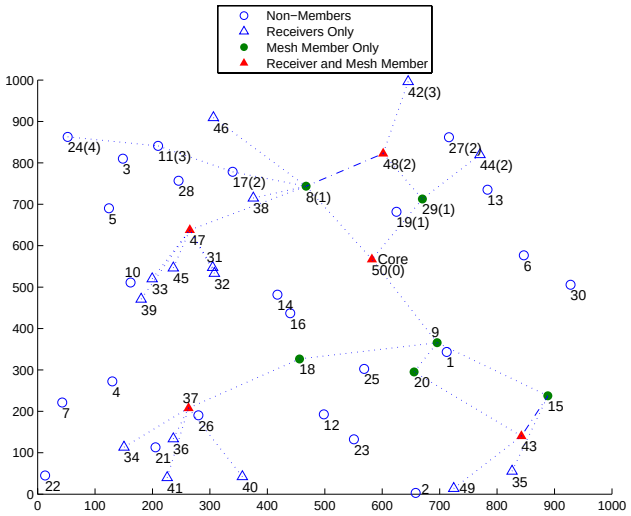


Fig. 2. Single parent included in mesh

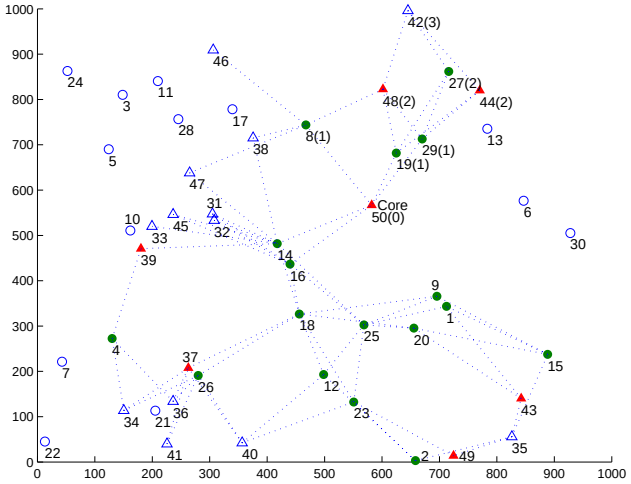


Fig. 3. All parents included in mesh

shown in Figure 2 are both part of the mesh although node 48 explicitly includes only node 29). Node 8 in included in the mesh by node 46. Similarly node 43 has two parents nodes 15 and 20 in the mesh although it explicitly includes only node 20. We call this kind of redundancy (which exists at node 48 and 43) as *accidental redundancy* and is denoted dashed-dotted lines. Accidental redundancy can be expected to increase as the number of receivers in the network increases.

When node 42 wants to include all its parents in the mesh (as shown in Figure 3), it sets the parent field in its multicast announcement to 27. If it wanted to include only two of its parents in the mesh it would set its parent field to 44. Similarly, when node 48 sends its multicast announcement it sets its parent field to 8 due to which nodes 8, 19 and 29 are included in the mesh. The mesh in Figure 2 includes 11 nodes whereas the mesh in Figure 3 includes 24 nodes. As a result, their data packet overhead is differs by more than 100%.

E. Forwarding Multicast Data Packets

A data packet travels from sender to the receiver in two stages. In the first stage it travels hop by hop from the sender towards the core. As soon as it reaches a member of the mesh, it is flooded within the mesh by mesh members (nodes with a membership code of at least 2).

When a sender sends a multicast data packet, it encapsulates it within a unicast data packet. The destination address of the unicast data packet is set to one of the parents of the node. A sender generally chooses its parent with the highest ID as the destination of the encapsulated data packet. The node receiving the encapsulated data packet similarly forwards the data packet towards its own parent. In this way the encapsulated data packet moves hop by hop towards the core. Once the data packet reaches a mesh member, the multicast data packet within the encapsulated unicast packet is removed and is flooded within the mesh.

This is illustrated in Figure 2. Node 24 (in the top left corner) which has a distance to core of 4, sends multicast data packets encapsulated within unicast packets with destination address set to 11. Similarly node 11 forwards the packets it receives from node 24 setting the destination address to 17. Similarly node 17 sets the destination address of the unicast packet to 8 in the packets it forwards. Once the packet is received by node 8 which is a mesh member, it removes the multicast packet within the unicast packet

and floods it within the mesh, with mesh members using a packet ID cache to discard duplicates. Please note that the mesh comprises of only nodes with a mesh membership code of at least two. i.e. nodes which have at least one mesh child. Nodes which are simply receivers do not participate in the flooding of data packets.

F. Dynamics of Packet loss

Packets are lost in mobile ad hoc networks because of two distinct reasons: (1) Because of the inherent unreliability of the wireless medium and interference between simultaneous transmissions. (2) Because of the fact that the node transmitting the data packet and the node receiving it are not in each others range, due to mobility.

From now on we will refer the first kind of packet losses as *packet drops due to collisions* and the second kind of packet losses as *packet drops due to broken routes*. Although all multicasting protocols to date do mention both these reasons for packet loss, we are not aware of any paper which tries to quantify the relative importance of the two in terms of influencing packet delivery ratio.

In order to get an estimate of packet loss due to collisions we performed the following simple experiment: In a static network we randomly placed 50 nodes in a terrain size of 1000 m X 1000 m. Complete details of the simulation environment are given in Figure 7. 5 nodes were randomly selected as senders which generated a traffic of 2 packets per second. All nodes simply forwarded data packets from these five senders. This ensured that all data packets from all senders were flooded throughout the network. Nodes maintained a packet ID cache and dropped duplicates. From among the nodes which were neither senders nor receivers, two nodes A and B, were selected so that they were in each others radio range. Node A transmitted data packets at a relatively infrequent rate viz. 1 packet in every 5 seconds. The experiment was continued till Node A had transmitted 10000 data packets. The data packets transmitted by node A were not retransmitted by node B or any of A's neighbors. We repeated the results for 10 different seed values. On an average, node B received 9953.4 data packets sent by node A, giving a link reliability of 99.534 for a traffic load of 10 data packets per second.

Assuming this rate of packet drops due to collisions alone, even when the average hop count from sender to receiver was 5, and there was absolutely no redundancy, packet drops due to collisions could account for a packet drop of only 2.5%. In practice however, tree based protocols give a much lower packet delivery ratio even though there always exists some redundancy due to accidental redundancy , as described in Section III-D. Even mesh based protocols do not always have such a high packet delivery ratio in high mobility as shown in Table I and described in Section IV. This leads us to believe that packet drops due to link breaks is the more important source of packet drops in situations of low to moderate traffic load.

Technically speaking the only factor which determines the frequency of link breaks is the mobility of nodes in the network. However there are certain design decisions which can be made in the routing protocol to minimize the impact of link breaks. We will now discuss how the adaptive mesh based multicast protocol attempts to minimize packet loss due to link breaks. We will also discuss future modifications to the adaptive mesh based multicast protocol so as to minimize packet drops due to link breaks even further.

1) Handling link breaks in the path from sender to the mesh: As described in Section III-E a data packet from a sender to the receivers travels in two stages. In the first stage it travels hop by hop from the sender towards the core, where it is encapsulated in a unicast data packet. As soon

as it reaches a member of the mesh, the multicast packet within the unicast packet is flooded within the mesh.

In order avoid link breaks on the path from sender to the mesh, adaptive mesh based multicast protocol uses implicit acknowledgments. This is illustrated in Figure 4. The dashed line indicates wireless connectivity, and the number besides a node indicates its reported distance to core. Assume, at time t = 10, 10.02 and 10.05 seconds, node S receives multicast announcements from nodes D, E, F respectively each reporting a distance to core of 2. Assume that S is a source of multicast data packets transmitting one packet every 100 ms. Suppose node S initially chooses node F for forwarding data packets towards the core. Multicast data packets transmitted by node S are encapsulated within a unicast data packet with destination F. Node F similarly sets the destination address to one of its own parents when it forwards the data packet it receives from node S. Suppose at time t=11 seconds node F moves out of the radio range of node S to F", as shown in Figure 4. If node S simply keeps transmitting data packets with destination address set to F, till the next batch of multicast announcements which will arrive at around t = 13 seconds, indicates that the link to F is broken, potentially 20 data packets could be dropped. (In our present implementation multicast_announcement_interval is set to three seconds). However because all communication is broadcast whenever node F forwards a data packet that it receives from node S, this data packet is also received by node S within ACK_TIMEOUT. This is termed as an *implicit acknowledgment*. If just one implicit acknowledgment is not received it could be because of either the data packet or the implicit acknowledgment was lost in a collision. However if two consecutive implicit acknowledgments are not received it is highly probable that the recipient node in no longer in range. Thus node S is able to infer that its link with node F has been broken, and sends subsequent packets to node E. If the link to node E is also broken packets could also be sent to node D. The main advantage of this scheme is that it allows link breaks to be fixed quickly without any control overhead.

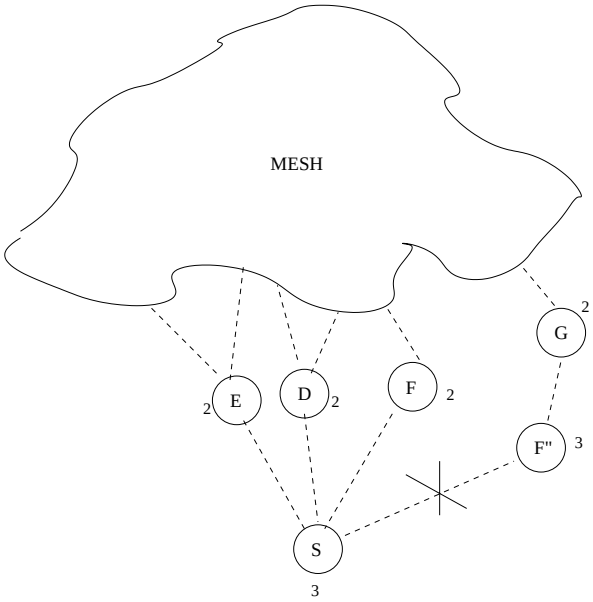


Fig. 4. Fixing link breaks on packet path from sender to the mesh

One alternative to this scheme is the explicit acknowledgment of each data packet which would add control overhead equal to the number of encapsulated data packets transmitted, and is clearly not desirable. The other alternative is to increase the frequency of multicast announcements, so that the average time that nodes would have to wait before a broken link would be restored would be reduced. This would lead to lower number of data packets dropped but lead to a higher control overhead. However if the link was broken immediately after a node received a multicast announcement a significant number of data packets could still be dropped depending on the rate of traffic generation at the sender.

2) Handling link breaks within the mesh: Link breaks within the mesh are less frequent simply because the

mesh offers greater redundancy, which depends on the number of parents included in the mesh by individual nodes. However link breaks can still occur, due to which packets are lost. Figure 5 shows such a network and a mesh constructed using adaptive mesh based multicast protocol. Shaded nodes represent mesh members and unshaded nodes represent non-members. The dashed arrows point from children to their parents. Dashed lines represent wireless connectivity between neighbors which do not share a parent child relationship. Based on the entries in its connectivity list, node 11 has three parents, nodes 12, 13 and 14. At time t = 15 seconds node 11 sends out a multicast announcement setting the parent field to 13, effectively including parents 13 and 14 in the mesh, but not node 12. Also assume that node 11 forwards a packet originating from node 5 every 100 ms. Assume at time t = 16 seconds node 14 moves out of range of node 11. Node 11 detects the break when it fails to receive two consecutive *implicit acknowledgments* as described in Section III-F.1. Node 11 however does not take any action as it can still forward data packets towards the core through node 13. However assume that at time t = 17 node 13 also moves out of range of node 11. Node 11 can wait for the next round of multicast announcements at around t = 18, but that would mean that 10 data packets originating from node 5 would be dropped. As node 11 has another parent node 12, which was not previously included in the mesh, node 11 sends out a multicast announcement with parent field set to 12, thus including node 12 as part of the mesh. Node 12 would also send out a multicast announcement indicating a change in its mesh member status, as a result of which node 17 would also be included in the mesh. Subsequent packets from the sender node 5 would then be routed to the core via nodes 11, 12 and 17. Please note that even though node 11 had not explicitly

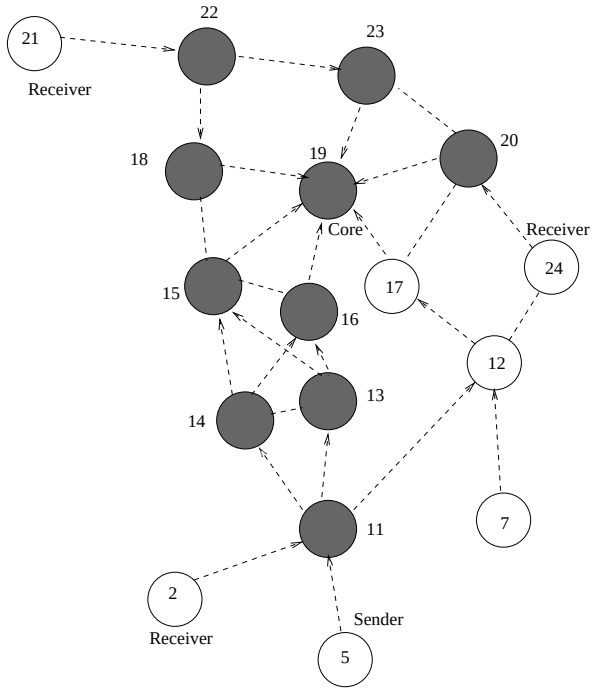


Fig. 5. Fixing link breaks within the mesh

included node 12 in the mesh, node 12 could already be part of the mesh because some other receiver included it in the mesh.e.g. node 7. (Accidental redundancy as described in Section III-D). In that case node 11 would continue transmitting data packets from node 5 even though links to nodes 13 and 14 were broken. It would not transmit any multicast announcements as it would know that node 12 was already a member of the mesh and would transmit data packets received from node 11 anyway. Please note that these methods for handling link breaks will work only if data traffic exists through these nodes. However if data traffic does not exist then there is no point fixing the links anyway. Even in the absence of data packets, broken links are eventually fixed when the next multicast announcement originates from the core.

3) Reducing average link hops: Assuming the average probability of packet drops across different links roughly similar over a period of time, one can infer that the probability of packet delivery from sender to receiver on

a path without redundancy, decreases exponentially with an increase in the path length. Hence a protocol that has a lower average hop count from senders to receivers can be expected to provide lower number of packet drops. In a future extension to adaptive mesh based multicast protocol we propose to modify the core election so that instead of choosing the first receiver that joins the group, or the receiver with highest ID as the core, the receiver which is closest to other receivers is chosen as the core. Additionally instead of sending data packets towards the core as described in Section III-E, senders would send data packets towards the mesh. This would reduce the average number of hops from a sender to a receiver by avoiding cases of data packets being transmitted to a far off core, and then coming all the way back. In addition of reducing packet drops, this would also result in a lower delay.

4) Selection of parents based on signal strength: Currently the nodes(s) with highest ID from among the parents are selected as the destination for their encapsulated unicast packet by non-members, and nodes to be included in the mesh by receivers and mesh members. A simple extension to adaptive mesh based multicast protocol can make the selection based on nodes which report the highest signal strength. A higher signal strength will in most cases mean that the parent is closer to the node, compared to other parents of lower signal strength. Choosing such a parent for forwarding a data packet will reduce the likelihood of link breaks due to mobility.

G. Controlling Redundancy within the Mesh

Even though adaptive mesh based multicast protocol tries its best to prevent link breaks, and fixing link breaks quickly when then do occur, some links break and packet drops will always occur due to mobility and collisions. Redundancy is beneficial in these scenarios as even if a particular link breaks, data packets can be delivered via an alternate path. The main challenge is to include redundancy only in the situations that require it, so as to maximize packet delivery ratio and minimize overhead.

Section III-D describes how nodes set the parent field of their multicast announcement so that the number of parents with node ID equal to or greater than the parent field equals the number of parents that they want to include in the mesh. This section describes how nodes decide how many parents they want to include in the mesh. Our research has shown that even in the most extreme cases of mobility having more than 3 parents does increase the overall probability of packet delivery ratio. The main reason for that is that having three separate parents is more than enough to ensure that a packet is delivered to the next hop towards the core. The packet drop that do occur are generally due to regions in the mesh where so many parents are not available. Even two parents are generally enough, and only in the cases of extreme mobility does having three parents actually yield any benefit. We believe that the reason that in situations of high mobility three parents provide better packet delivery ratio than two parents is not because having three parents significantly increases the chances of getting the packet one hop closer to the core compared to two, but because having three parents increases the number of nodes in the mesh. As a result nodes which have only a single parent may be able to reach the core through an additional path even though it may not be the shortest. The reduction in packet drops involving such nodes leads to an increased packet delivery ratio. This is illustrated in Figure 5. Node 24 which is a receiver has only one parent thus receives data packets from only that parent viz. node 20. Similarly node 11 includes two parents, viz nodes 13 and 14 in the mesh. If node 11 also includes 12 in the mesh, then it will not significantly

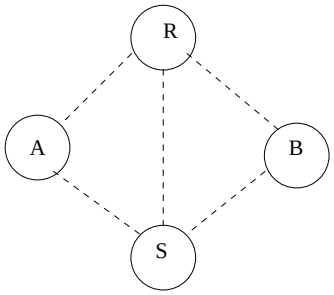


Fig. 6. Effect on redundancy on reliability

improve the connectivity of node 11 to the mesh, as it is already connected to the mesh though two parents. However node 12 being included in the mesh would mean that node 24 would receive data packets through two nodes instead of one, substantially increasing its packets received, even though node 12 does not provide a shortest path to the core.

How increasing the number of parents in the mesh increases the redundancy of the mesh is a difficult problem to analyze. Having each node select three parents could mean that a node three hops away from the core could have $3^3 = 27$ disjoint paths to the core, but in practice many of the nodes would have common parents hence the total number of paths would be much less. In order to get an idea about how redundancy affects the packet delivery ratio consider the network shown in Figure 6. The dashed lines shows wireless connectivity between nodes. Assume node S is a sender and node R is a receiver. Also assume that the link reliability is 90%. If nodes A and B do not participate in the forwarding of data packets, then the probability of a packet sent by S being received by R is simply 90%. If node A also participates in the forwarding the probability of packet delivery at R increases to 98.1%. If node B also participates in the forwarding then the probability goes up to 99.64%. (The probability of receiving at least one copy of a packet through multiple disjoint paths = 1 - probability of not receiving it through any of the paths.)

1) Measuring reliability: Mesh based protocols provide multiple paths between senders and receivers. Hence a useful metric to estimate the reliability of the mesh should consider the overall reliability offered by multiple paths. Moreover as reliability can vary in different parts of the mesh, it should be possible for nodes to estimate reliability locally, and take appropriate action based on the reliability measured.

Mesh Reliability Index (MRI) is thus defined as the probability that a packet transmitted by a particular node reaches one hop closer to the core. Every time a node transmits a data packet received from one of its children or a non-member, and which has not already been transmitted by its parents, it starts a timer. If within ACK_TIMEOUT the node receives a copy of the data packet from any one of its parents it counts the event as a successful mesh reliability acknowledgment. If the node does not receive a copy of the packet from even a single parent, it counts the event as a failed mesh reliability acknowledgment. Nodes keep track of the number of successful mesh reliability acknowledgments in the last hundred mesh reliability acknowledgments initiated, and this number divided by 100 is the Mesh Reliability Index of the node. It is important to exclude packets received from parents in the calculation of MRI, because mesh members transmit a packet only once. Thus a data packet transmitted once by the parents will not be transmitted again.

MRI allows nodes to estimate redundancy and mobility using a single parameter. For a given number of parents (redundancy) higher mobility reduces the MRI. However for a given mobility, increasing the number of parents increases the MRI as the probability of receiving a copy of the packet from at least one parent increases.

2) Calculating number of parents: In order to correlate MRI to the number of parents we ran the experiments described in Section IV-A for 4 versions of PUMA, V_1, V_2, V_3, and V_All. As the name implies in these versions nodes included 1, 2, 3 and all parents in the mesh respectively. (Including a certain number of parents was of course subject to the restriction that that those number of parents did in fact exist. e.g. if a node had only 2 parents then it could not include 3 parents in the mesh). We found a strong correlation between MRI and the overall packet delivery ratio reported in various simulations. For all our simulations if the average MRI was greater that 92% the overall packet delivery ratio was never less than 90%. Moreover we found that having an MRI greater than 95% did not have any significant impact on packet delivery ratio.

Hence in adaptive mesh based multicast protocol nodes attempt to maintain the MRI between 92% and 95%. The number of parents to be included in the mesh is initialized to 1. Every time a node sends a multicast announcement, the node checks its MRI. If it is between .92 and .95 then it leaves the number of parents unchanged. If it is more than .95 it decreases the number of its parents by 1, and if it is less than .92 it increases the number of its parents by 1. However the number of parents is not increased beyond 3 and not decreased beyond 1. Please note that the actual mesh reliability is greater than the MRI. e.g. if the MRI was 94% it meant that the data packet as well as its acknowledgment had been received correctly in 94% of cases. If mesh reliability was p, then it meant that $p^2 = 94\%$ or $p = 96.95\%$.

IV. Performance Comparison

We compared the performance of the adaptive mesh-based multicast protocol against V_1 (where mesh members included one of their parents in the mesh), V_All (where mesh members include all of their parents in the mesh which is the same as PUMA), and ODMRP [3], a well known mesh-based protocol.

Simulator	Qualnet 3.5
Total Nodes	50
Simulation Time	700 seconds
Simulation Area	1000m X 1000m
Node Placement	Random
Pause Time	0
Mobility Model	Random Waypoint
Radio Range	250m
Channel Capacity	2 Mbps
MAC protocol	IEEE 802.11−1997
Data packet size	512 bytes

Fig. 7. Simulation environment

For our experiments, we used the Qualnet [18] network simulator. Figure 7 lists the general parameters that characterize the simulation environment. The field size used is 1000 m X 1000 m for all experiments except Experiments 6 and 11. Nodes moved at a constant speed (maximum speed was set equal to the minimum speed in the random waypoint model) as specified in the individual experiments.

At the MAC layer, all data packets as well as control packets (multicast announcements) were sent unreliably using only CSMA/CA. Each simulation was run for four different seed values. Multicast announcements in all protocol versions were sent every three seconds.

As our main objective in this paper was to measure the trade-off between packet delivery ratio and redundancy in mesh-based protocols, we used **packet delivery ratio (pdr)** and **average mesh size** as performance metrics. However, we also provide results for **total control overhead** as well as **total overhead** which includes control packets as well as data packets transmitted.

A. Simulation Scenarios

We carried out the following experiments:

- Experiment 1 : Mobility varied across {0, 10, 20, 50, 100, 200} m/s. Senders = 5, Receivers = 20, Traffic Load = 10 pkts/sec, Multicast groups = 1.
- Experiment 2 : Senders varied across {1, 2, 5, 10, 20}. Mobility = 5 m/s, Receivers = 20, Traffic Load = 10 pkts/sec, Multicast groups = 1.
- Experiment 3 : Receivers varied across {5, 10, 20, 30, 40}. Mobility = 5 m/s, Senders = 5, Traffic Load = 10 pkts/sec, Multicast groups = 1.
- Experiment 4 : Traffic Load varied across {1, 2, 5, 10, 25, 50} pkts/sec. Mobility = 0, Senders = 5, Receivers = 20, Multicast groups = 1.
- Experiment 5 : Multicast Groups varied across {1, 2, 5, 10}. Mobility = 5 m/s, Senders = 5 per group, Receivers = 20 per group, Traffic Load = 20 pkts/sec.
- Experiment 6 : Terrain size varied across {800 m X 800 m, 1058 m X 1058 m, 1265 m X 1265 m, 1442 m X 1442 m, 1600 m X 1600 m}. Mobility = 5 m/s, Senders = 5, Receivers = 20, Traffic load = 10 pkts/sec, Multicast Groups = 1. The terrain sizes have been chosen so that each consecutive value of terrain size increases the area by $4800\,m^2$
- Experiment 7 : Mobility varied across {0, 10, 20, 50, 100, 200} m/s. Senders = 5, Receivers = 5, Traffic Load = 10 pkts/sec, Multicast groups = 1. (Same as Experiment 1 except that number of receivers was 5 instead of 20)
- Experiment 8 : Mobility varied across {0, 10, 20, 50, 100, 200} m/s. Senders = 5, Receivers = 10, Traffic Load = 10 pkts/sec, Multicast groups = 1. (Same as Experiment 1 except that number of receivers was 10 instead of 20)
- Experiment 9 : Mobility varied across {0, 10, 20, 50, 100, 200} m/s. Senders = 5, Receivers = 20, Traffic Load = 25 pkts/sec, Multicast groups = 1. (Same as Experiment 1 except that traffic load was 25 pkts/sec instead of 10)
- Experiment 10 : Mobility varied across {0, 10, 20, 50, 100, 200} m/s. Senders = 5, Receivers = 20, Traffic Load = 50 pkts/sec, Multicast groups = 1. (Same as Experiment 1 except that traffic load was 50 pkts/sec instead of 10)
- Experiment 11 : Mobility varied across {0, 10, 20, 50, 100, 200} m/s. Senders = 5, Receivers = 20, Traffic Load = 10 pkts/sec, Multicast groups = 1, Terrain Size = 1442 m X 1442 m (Same as Experiment 1 except that terrain size was 1442 m X 1442 m instead of 1000 m X 1000 m)

Experiments 1-6 were carried out to determine the effect of mobility, number of senders, number of receivers, traffic load, number of multicast groups and terrain-size, respectively. For the traffic load test (Experiment 4) we set mobility to 0, because we wanted to focus on packet drops caused by congestion. Both the senders and receivers were chosen randomly from among the 50 nodes. Traffic load was equally distributed among all senders. For the multiple-group test (Experiment 5), random allocation of nodes to groups could result in a single node being a member of multiple groups. Experiments 1-4 are almost the same as those used by the developers of ODMRP [3] in [6] to compare their protocol against CAMP [2], AMRIS [10] and AMROUTE [9]. The main difference is that in the mobility test (Experiment 1) we vary the mobility to a much higher value (200 m/s), as the possibility of link breaks is higher at higher mobility, due to which redundancy could have an impact. Though we realize that such a high value of mobility is unrealistic, we wanted to test the adaptability of our protocol in extreme conditions. Moreover we do test

our protocol at realistic mobilities as well.

We especially wanted to test the performance of our protocol in situations of frequent link breaks/packet loss to make sure that adaptive mesh based multicast protocol would be able to adapt mesh redundancy accordingly. Intuitively we felt that frequent link breaks / packet losses would occur in the following situations:

a) High Mobility : Higher number of link breaks can be expected in situations of high mobility.

b) Small number of Receivers : Smaller number of receivers leads to less accidental redundancy as described in Section III-D.

c) High Packet Load : Significant number of packets could be lost due to congestion and collisions.

d) Large Terrain Size : For the same number of nodes, a larger terrain size would mean a sparser network, making it more susceptible to link breaks.

Experiments 7-11 measure the impact of scenarios combining the effects of mobility along with small number of receivers, high packet load and large terrain size. We were especially interested in measuring the impact of redundancy in such scenarios where a large number of links could be expected to break.

B. Analysis

1) Comparison with V_1 and V_All: As we have mentioned before, the motivation for this research was to create a multicasting protocol which would adapt the redundancy in the mesh depending on the need, so as to have a high packet delivery ratio for a low overhead. Our comparisons with V_1 and V_All which represent protocols with minimum and maximum redundancy respectively, illustrate how well the adaptive protocol performed in this regard. Based on Table I the average packet delivery ratio per simulation for V_1, “Adaptive”, and V_All is 0.78, 0.83 and 0.86, respectively. As we can observe, the average packet delivery ratio of the adaptive protocol is quite close to V_All. V_1’s average packet delivery ratio is not very low either, and it may be a good choice at low mobility. However as can be seen from Table I the packet delivery ratio of V_1 falls significantly below that of “Adaptive” and V_All at high mobility. In some instances of high packet load V_All suffers larger number of collisions due to high packet overhead, as a result of which its packet delivery ratio is actually lower than that of V_1 and “Adaptive”, as can be seen from Table I Experiments 4, 9 and 10. All protocols have a low packet delivery ratio for sparse networks as can be seen from Table I Experiments 6 and 11. This is because sparse networks are more susceptible to partitions and link breaks.

Table III indicates that the average total packets (both data well as control) transmitted per node by V_1, “Adaptive” and V_All are 2683.66, 3878.77 and 6030.30, respectively. This indicates that “Adaptive” has a significantly lower overhead than V_All. This can also be observed in Table IV which shows that the average mesh size for V_All is significantly more than V_1 and “Adaptive” for all experiments. Additionally, we can see that in Table IV Experiments 2, 3, 4, 5 and 6 where mobility is not very high, the mesh size of “Adaptive” is very similar to that of V_1 indicating that “Adaptive” does not incur additional overhead in situations of low mobility. Mesh size is directly related to data packet overhead as it indicates the number of nodes involved in flooding of data packets. As can be seen by comparing Tables II and III, the control overhead is a very small fraction of total overhead for V_1, “Adaptive” and V_All, which means that most of the overhead incurred at nodes is data packet overhead. Hence “Adaptive” is important in the sense that it reduces data packet overhead,

which is the dominant part of the total overhead.

2) Adaptive Mesh-Based Multicast vs ODMRP: Table III indicates that the average total packets (both data well as control) transmitted per node by "Adaptive" and ODMRP are 3878.77 and 9263.47 respectively. The total overhead for ODMRP is even higher than V_All. The exceedingly high overhead for ODMRP is not only because of the greater redundancy in the mesh, but also because of a much higher control overhead as shown in Table II. The data packet overhead of ODMRP is similar to that of V_All as can be seen from their similar mesh sizes as given in Table IV. Comparing Tables II and III shows that unlike V_1, "Adaptive" and V_All, the control overhead for ODMRP is a significant fraction of the total overhead. The reason for the exceedingly high control overhead for ODMRP is that, unlike "Adaptive", V_1, and V_ALL, where only one node (the core) floods multicast announcements, in ODMRP every sender periodically floods JOIN request packets. In addition, members of the mesh transmit JOIN tables in ODMRP. "Adaptive", V_1, and V_ALL only send out multicast announcements.

Based on Table I, the average packet delivery ratio per simulation for "Adaptive" and ODMRP is 0.83 and 0.82, respectively, which means that the average packet delivery ratio for "Adaptive" is *actually* higher than that of ODMRP. ODMRP does in fact perform better than "Adaptive" at high mobility as can be seen from Table I. However in experiments involving larger traffic load, ODMRP suffers greater number of packet drops due to collisions because of its greater control overhead. Experiment 2 (greater control overhead due to multiple senders), Experiment 5 (greater overhead due to multiple groups) and Experiments 4, 9, and 10 (greater traffic load) result in large scale packet drops for ODMRP as can be seen in Table I. As a result the average

Experiment-1 Receivers=20 Senders=5 Load=10pkt/sec Mcast-Grps=1	Mobility (m/s)	0	10	20	50	100	200
	V_1	0.98	0.89	0.85	0.78	0.76	0.77
	Adaptive	0.99	0.94	0.91	0.88	0.89	0.88
	ODMRP	0.99	0.97	0.97	0.96	0.95	0.94
	V_All	0.99	0.97	0.96	0.95	0.95	0.95
Experiment-2 Mobility=5m/s Receivers=20 Load=10pkt/sec Mcast-Grps=1	Senders	1	2	5	10	20	
	V_1	0.94	0.93	0.93	0.92	0.92	
	Adaptive	0.96	0.96	0.95	0.95	0.95	
	ODMRP	0.75	0.97	0.98	0.96	0.70	
	V_All	0.98	0.98	0.98	0.98	0.98	
Experiment-3 Mobility=5m/s Senders=5 Load=10pkt/sec Mcast-Grps=1	Receivers	5	10	20	30	40	
	V_1	0.91	0.91	0.93	0.95	0.95	
	Adaptive	0.93	0.94	0.95	0.96	0.96	
	ODMRP	0.99	0.98	0.98	0.97	0.96	
	V_All	0.97	0.97	0.98	0.98	0.98	
Experiment-4 Mobility=5m/s Receivers=20 Senders=5 Mcast-Grps=1	Traffic Load	1	2	5	10	25	50
	V_1	0.98	0.98	0.98	0.98	0.84	0.69
	Adaptive	0.97	0.98	0.99	0.99	0.88	0.71
	ODMRP	0.97	0.98	0.99	0.99	0.82	0.57
	V_All	0.98	0.99	0.97	0.99	0.90	0.68
Experiment-5 Mobility=5m/s Receivers=20 Senders=5 Load=20pkt/sec	Multicast Groups	1	2		5	10	
	V_1	0.92	0.87		0.76	0.71	
	Adaptive	0.94	0.90		0.81	0.79	
	ODMRP	0.88	0.73		0.46	0.35	
	V_All	0.92	0.90		0.85	0.83	
Experiment-6 Mobility=5m/s Receivers=20 Senders=5 Load=10pkt/sec Mcast-Grps=1	Terrain Size	800 X 800	1058 X 1058	1265 X 1265	1442 X 1442	1600 X 1600	
	V_1	0.99	0.98	0.92	0.67	0.32	
	Adaptive	0.99	0.98	0.91	0.66	0.32	
	ODMRP	0.99	0.98	0.90	0.65	0.32	
	V_All	0.99	0.99	0.91	0.67	0.32	
Experiment-7 Receivers=5 Senders=5 Load=10pkt/sec Mcast-Grps=1	Mobility (m/s)	0	10	20	50	100	200
	V_1	0.98	0.83	0.76	0.68	0.64	0.62
	Adaptive	0.98	0.91	0.87	0.81	0.78	0.76
	ODMRP	1.00	0.98	0.97	0.92	0.87	0.81
	V_All	0.99	0.95	0.92	0.90	0.90	0.89
Experiment-8 Receivers=10 Senders=5 Load=10pkt/sec Mcast-Grps=1	Mobility (m/s)	0	10	20	50	100	200
	V_1	0.98	0.85	0.80	0.72	0.68	0.68
	Adaptive	0.98	0.92	0.89	0.85	0.84	0.82
	ODMRP	0.99	0.98	0.97	0.95	0.92	0.89
	V_All	0.99	0.95	0.94	0.93	0.93	0.93
Experiment-9 Receivers=20 Senders=5 Load=25pkt/sec Mcast-Grps=1	Mobility (m/s)	0	10	20	50	100	200
	V_1	0.84	0.81	0.77	0.72	0.71	0.71
	Adaptive	0.88	0.87	0.85	0.83	0.82	0.81
	ODMRP	0.82	0.78	0.80	0.81	0.83	0.83
	V_All	0.90	0.85	0.84	0.84	0.84	0.86
Experiment-10 Receivers=20 Senders=5 Load=50pkt/sec Mcast-Grps=1	Mobility (m/s)	0	10	20	50	100	200
	V_1	0.69	0.70	0.67	0.62	0.60	0.61
	Adaptive	0.71	0.65	0.63	0.61	0.62	0.65
	ODMRP	0.57	0.54	0.54	0.56	0.58	0.61
	V_All	0.68	0.60	0.60	0.60	0.62	0.66
Experiment-11 Receivers=20 Senders=5 Load=10pkt/sec Terrain Size = 1442mX1442m	Mobility (m/s)	0	10	20	50	100	200
	V_1	0.67	0.64	0.56	0.48	0.46	0.44
	Adaptive	0.66	0.71	0.64	0.58	0.56	0.52
	ODMRP	0.65	0.77	0.73	0.67	0.64	0.61
	V_All	0.67	0.74	0.68	0.65	0.65	0.63

TABLE I
PACKET DELIVERY RATIO

Protocol	Mean Control Pkts Transmitted	Std Dev
V_1	264.12	19.63
Adaptive	293.49	16.57
ODMRP	3891.53	3552.53
V_All	270.26	19.20

TABLE II
CONTROL PACKETS TRANSMITTED PER NODE

Protocol	Mean Total Pkts Transmitted	Std Dev
V_1	2683.66	2071.67
Adaptive	3878.77	3331.21
ODMRP	9263.47	5158.61
V_All	6030.30	4311.41

TABLE III
TOTAL PACKETS TRANSMITTED PER NODE

Experiment No	Protocol	Mean Mesh Size	Std Dev
Experiment-1	V_1	13.39	1.74
	Adaptive	19.70	3.07
	ODMRP	32.28	1.34
	V_All	31.41	2.82
Experiment-2	V_1	11.06	0.17
	Adaptive	13.93	0.66
	ODMRP	25.80	11.92
	V_All	28.79	0.30
Experiment-3	V_1	10.55	3.67
	Adaptive	13.29	4.95
	ODMRP	29.93	6.06
	V_All	26.68	7.51
Experiment-4	V_1	12.56	1.53
	Adaptive	13.51	0.14
	ODMRP	34.75	10.52
	V_All	26.19	3.64
Experiment-5	V_1	13.84	2.10
	Adaptive	16.77	4.67
	ODMRP	20.55	7.32
	V_All	27.67	0.99
Experiment-6	V_1	12.92	4.82
	Adaptive	13.38	5.10
	ODMRP	25.63	11.32
	V_All	21.52	8.55
Experiment-7	V_1	6.52	1.08
	Adaptive	10.30	1.67
	ODMRP	20.26	2.41
	V_All	19.92	3.32
Experiment-8	V_1	9.61	1.36
	Adaptive	14.91	2.26
	ODMRP	26.96	2.31
	V_All	26.23	3.23
Experiment-9	V_1	12.15	1.60
	Adaptive	17.58	2.17
	ODMRP	27.19	1.29
	V_All	28.64	2.84
Experiment-10	V_1	10.55	1.47
	Adaptive	15.89	4.11
	ODMRP	17.37	1.28
	V_All	22.38	3.37
Experiment-11	V_1	11.81	1.48
	Adaptive	14.75	1.01
	ODMRP	22.38	1.99
	V_All	21.29	1.69

TABLE IV
AVERAGE MESH SIZE

packet delivery ratio of ODMRP is slightly less than that of "Adaptive".

V. CONCLUSIONS

This paper investigates the tradeoffs between reliability and efficiency in mesh-based MANET multicast protocols. We introduce an adaptive mesh-based multicast mechanism which adapts mesh redundancy based on link reliability. Mesh redundancy is measured by the number of paths from each receiver to the mesh's core. In addition to adapting redundancy based on link reliability, the adaptive mesh based multicast protocol is able to adapt traffic routes to avoid broken links without incurring any additional control overhead. However, an important assumption that we make is that links are bidirectional.

Through simulations, we compare the performance of the adaptive mesh-building protocol against non-adaptive versions of the protocol which build the mesh with maximum and minimum redundancy, as well as against ODMRP. We conducted experiments for a wide range of scenarios with varying mobility, group members, number of senders, traffic load, number of multicast groups and terrain size. Our results indicate that adaptive mesh-based multicast exhibits high average packet delivery ratio (viz., 3% less than the maximum-redundancy version of the protocol), with half of the overhead when compared to the maximum-redundancy version of the protocol. When compared to ODMRP, adaptive mesh-based multicast exhibits comparable average packet delivery ratio with less than half of the overhead.

REFERENCES

[1] R. Vaishampayan and J.J. Garcia-Luna-Aceves, "Protocol for unified multicasting through announcements (puma)," in *Proceedings of the 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, October 2004.

[2] J. J. Garcia-Luna-Aceves and E.L. Madruga, "The core assisted mesh protocol," *IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks*, vol. 17, no. 8, pp. 1380–1394, August 1999.

[3] S.J. Lee, M. Gerla, and Chian, "On-demand multicast routing protocol," in *Proceedings of WCNC*, September 1999.

[4] S.K. Das, B.S. Manoj, and C.S. Ram Murthy, "A dynamic core based multicast routing protocol for ad hoc wireless networks," in *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, June 2002.

[5] S. Lee and C. Kim, "Neighbor supporting ad hoc multicast routing protocol," in *Proceedings of the ACM International Symposium*

on Mobile Ad Hoc Networking and Computing (MobiHoc), August 2000.

[6] S.J. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia, "A performance comparison study of ad hoc wireless multicast protocols," in *Proceedings of IEEE INFOCOM, Tel Aviv, Israel*, March 2000.

[7] E. Royer and C. Perkins, "Multicast operation of the ad hoc on-demand distance vector routing protocol," in *Proceedings of Mobicom*, August 1999.

[8] Park and Corson, "Highly adaptive distributed routing algorithm for mobile wireless network," in *Proceedings of IEEE INFOCOM*, March 1997.

[9] J. Xie snd R.R. Talpade, A. McAuley, and M. Liu, "Amroute: Ad hoc multicast routing protocol," *Mobile Networks and Applications (MONET)*, December 2002.

[10] C.W. Wu and Y.C. Tay, "Amris: A multicast protocol for ad hoc wireless networks," *Proceesings of IEEE MILCOM*, October 1999.

[11] J.G. Jetcheva and David B. Johnson, "Adaptive demand-driven multicast routing in multi-hop wireless ad hoc networks," in *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, October 2001.

[12] L. Ji and M. S. Corson, "A lightweight adaptive multicast algorithm," in *Proceedings of IEEE GLOBECOM 1998*, December 1998, pp. 1036–1042.

[13] L. Ji and M.S. Corson, "Differential destination multicast - a manet multicast routing protocol for small groups," in *Proceedings of IEEE INFOCOM*, April 2001.

[14] P. Sinha, R. Sivakumar, and V. Bharghavan, "Mcedar: Multicast core extraction distributed ad-hoc routing," in *Proceedings of the Wireless Communications and Networking Conference, WCNC*, September 1999, pp. 1313–1317.

[15] C.K. Toh, G. Guichala, and S. Bunchua, "Abam: On-demand associativity-based multicast routing for ad hoc mobile networks," in *Proceedings of IEEE Vehicular Technology Conference, VTC 2000*, September 2000, pp. 987–993.

[16] V. Devarapalli and D. Sidhu, "Mzr: A multicast protocol for mobile ad hoc networks," in *ICC 2001 Proceedings*, 2001.

[17] C. Perkins and E. Royer, "Ad hoc on demand distance vector (aodv) routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, February 1999.

[18] Scalable Network Technologies, "Qualnet 3.5," http://www.scalable-networks.com/.