A Performance Comparison of On-Demand Multicast Routing Protocols for Ad Hoc Networks

Jorjeta G. Jetcheva and David B. Johnson

December 15, 2004
CMU-CS-04-176

School of Computer Science
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

This work was supported in part by NASA under grant NAG3-2534 at Rice University; by NSF under grants CNS-0209204, CNS-0325971, CNS-0338856, and CNS-0435425 at Rice University; by a gift from Schlumberger to Rice University; and by the Air Force Materiel Command (AFMC) under DARPA contract number F19628-96-C-0061 at Carnegie Mellon University. The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of NASA, NSF, Schlumberger, AFMC, DARPA, Rice University, Carnegie Mellon University, or the U.S. Government or any of its agencies.

Keywords: wireless, ad hoc networks, routing, performance comparison, multicast, mesh networks.

Abstract

In this paper, we present a comparative performance evaluation of three general-purpose *on-demand* multicast protocols, namely ADMR, MAODV, and ODMRP, focusing on the effects of changes such as increasing number of multicast receivers or sources, application sending pattern, and increasing number of nodes in the network. We use mobile networks composed of 100 or 200 nodes, with both a single active multicast group and multiple active multicast groups in the network, in a wide range of multicast scenarios. Although some simulation results for these protocols have been published before, the three protocols have not been compared, and prior studies have focused on smaller networks using a small set of simulation scenarios, many with only a single active multicast group. We focus here on the effects of the protocols' relative degree of on-demand behavior and their performance in different multicast scenarios.

1. Introduction

Routing in ad hoc networks requires the discovery of multi-hop paths between the wireless mobile nodes in the network that wish to communicate. In *multicast* routing, packets from one sender must in general be delivered to multiple receivers making up a *multicast group*; any node can send packets to a group at any time, and any node can join or leave a multicast group at any time. Multicast routing is a hard problem in wired networks, and is even more challenging in ad hoc networks, due to the dynamic topology changes in the network due to node motion and wireless propagation variability, and due to the limited wireless network bandwidth and node energy resources available.

Because of these factors, multicast routing protocols designed for the Internet are not well suited for the ad hoc networking environment. In recent years, a number of multicast protocols for ad hoc networks have been proposed (e.g., [1, 4, 6, 8–11, 16–20]), some of which rely on *reactive* (on-demand) mechanisms and discover routes only when they are needed for current communication, and others which rely on *proactive* mechanisms such as periodic neighbor sensing or flooding, or periodic routing table exchanges.

In this paper, we present a performance comparison of three *on-demand* multicast routing protocols for ad hoc networks: the Adaptive Demand-Driven Multicast Routing protocol (ADMR) [9], the Multicast Ad Hoc On-Demand Distance-Vector protocol (MAODV) [16], and the On-Demand Multicast Routing Protocol (ODMRP) [17]. All of these protocols contain a significant on-demand (reactive) component, but they differ in how reactive and proactive mechanisms are combined to make the complete protocol: ADMR uses source-based trees and does not utilize any periodic control packet transmissions, MAODV uses a shared group tree and uses periodic Hello messages for link break detection and periodic group leader floods for group information dissemination, and ODMRP uses a group forwarding mesh for packet forwarding and utilizes periodic flood-response cycles for multicast state creation and maintenance.

In this performance evaluation, we focus on the effects of changes such as increasing number of multicast receivers or sources, application sending pattern, and increasing number of nodes in the network. We use mobile networks composed of 100 and 200 nodes, with both a single active multicast group and multiple active multicast groups in the network. We study a wide range of multicast scenarios representative of a variety of multicast applications, such as conferencing and single-source vs. multi-source groups. Although some simulation results for these protocols have been published before, the three protocols have not been compared, and prior studies have focused on smaller networks using a small set of simulation scenarios, many with only a single active multicast group. We focus here on the effects of the protocols' relative degree of on-demand behavior and their performance in different multicast scenarios.

Section 2 of this paper summarizes related work. The operation of the three protocols we studied, ADMR, MAODV, and ODMRP, is summarized in Section 3. Section 4 discusses our evaluation methodology. We present results in Section 5 and conclude with Section 6.

2. Related Work

A number of evaluations of ADMR, MAODV and ODMRP have been presented in the literature, all using ad hoc networks composed of 50 or fewer nodes [3, 6, 9, 13–16]. The only exception is the work of Kang et al. [12] who present the Scalable Multi-source Multicast Routing Protocol (SMMRP) and compare its performance to that of ODMRP in a network of 150 nodes. Their performance evaluation does not include ADMR and MAODV, and is only based on simulations of one multicast scenario: 1 group with 10 sources and 10 receivers. In contrast, in this work, we present an extensive evaluation of ADMR, MAODV and ODMRP, in networks with 100 and 200 nodes, in a wide range of multicast scenarios.

Lee et al. [15] compare the performance of ODMRP, CAMP [6], AMRoute [1] and Amris [20]. Their simulations are based on 50-node networks with a variable number of multicast senders and receivers, all part of a single multicast group. Lee et al. [14] propose a new multicast protocol called Neighbor-Supporting Multicast Protocol (NSMP), and compare its performance to that of ODMRP in a 50-node network with a single active multicast group as well. Similarly, Royer et al. [16] present an evaluation of MAODV in a 50-node network with only one multicast group, and Kunz et al. [13] compare the performance of MAODV and ODMRP in a 50-node network also in scenarios with a single active multicast group. Garcia-Luna-Aceves et al. [6] compare the performance of CAMP to that of ODMRP in 30-node networks with 1 or 2 multicast sources, where all nodes in the network are receivers.

Bunchua et al. [3] compare the performance of ABAM [19] to that of ODMRP in a 40-node network, and in our work, we have compared the performance of ADMR [9] to that of ODMRP in a 50-node network. Both of these studies study the protocols' behavior in networks with both a single as well as multiple active multicast groups in the network but in a limited set of multicast scenarios.

3. Protocol Descriptions

In this section, we present brief overviews of the operation ADMR, MAODV and ODMRP.

3.1. ADMR

ADMR does not employ any periodic control packet exchanges, such as neighbor sensing or periodic flooding, and does not rely on lower layers within the protocol stack to perform such functions; it performs both its route discovery and route maintenance functions on demand, and automatically prunes unneeded multicast forwarding state, and expires its multicast mesh when it detects that the multicast application has become inactive.

When there are no multicast sources or receivers for a given multicast group **G**, ADMR does not generate any packet transmissions. If multicast receivers and sources for **G** exist, ADMR creates a source mesh between each multicast sender **S** and the multicast receivers for the group. Source-specific forwarding enables the protocol to support source-specific multicast joins and to route along shorter paths than protocols that use group-shared forwarding.

Packet forwarding along the ADMR source mesh does not follow any predetermined sequence of hops, but instead each non-duplicate data packet is forwarded by each mesh node, thus following the current shortest-delay paths within the mesh, to the multicast receivers. This type of forwarding increases robustness against packet loss due to collisions or broken links.

The multicast sources and receivers in ADMR cooperate to create the multicast source mesh. Each source floods its first data packet for a group, and each receiver responds to that flood with a RECEIVER JOIN packet which sets up forwarding state along the shortest path back towards the source. A flood-response cycle is initiated by each receiver when it first joins the group as well. To resolve partitions, multicast sources may occasionally flood a data packet, e.g., every several tens of seconds.

To detect broken links within the mesh, the ADMR routing layer at a multicast source monitors the traffic pattern of the multicast source application, and includes the expected inter-arrival time of future packets in each data packet. Each mesh node records this information upon forwarding a packet, and uses it to detect that it has become disconnected from the mesh. Once a broken link is detected, the node downstream from it (relative to the multicast source) will attempt to perform a local repair using a localized RECONNECT packet flood. Before launching the local repair, the disconnected node sends a REPAIR NOTIFICATION packet downstream, i.e., towards the multicast receivers, in order to inform downstream mesh nodes that it is going to perform the repair and they should cancel their disconnection detection timers; nodes farther away from the source than the node downstream of the broken link, would detect the broken link later, because the disconnection timer incorporates the hop count to the multicast source, which node extract from forwarded data packets. Receiver nodes that receive the REPAIR NOTIFICATION postpone their disconnection timers rather than canceling them. If no data arrives after some time, the receivers assume that the local repair has failed and re-join the group using the request-response cycle invoked during group joins.

When the application is temporarily not sending data, the routing layer generates keepalive packets to enable detection of broken links during this inactive period. If the application does not send packets in significant deviation of its sending pattern, the keepalives stop and the multicast mesh silently expires. These mechanisms allow ADMR to detect broken links without the use of periodic control packet exchanges and to make informed decisions about whether or not it is worth maintaining a multicast mesh based on an application's communication behavior and needs.

3.2. MAODV

MAODV builds a group tree, shared by all sources and receivers for a group. This enables it to localize group joins and connection of newly active sources to the multicast tree, as well as, repairs when the tree becomes disconnected. The use of a shared tree and the localized connection and reconnection to the tree result in longer forwarding paths for data packets. Such paths have a higher likelihood of packet loss due to collisions, and higher end-to-end delay; they are also more likely to break which also leads to packet loss and a more frequent invocation of the route repair mechanisms within the protocol. MAODV requires the use of periodic neighbor detection packets for detection of broken links, and periodic group leader control packet floods (e.g., every 5s) for disseminating a multicast group's sequence number.

MAODV creates a shared tree between the multicast sources and receivers for a multicast group. The root of each group tree is a multicast source or receiver for the group that has been designated as a group leader. When an application on a node **R** issues a join request for a multicast group **G**, the MAODV routing layer at **R** floods the network with a ROUTE REQUEST packet. If no response is received, the flood is repeated. If no response is received to several such floods, node **R** becomes the group leader for **G**. When a new source wants to send packets to a group, it performs the same steps.

The group leader periodically floods the network, e.g., every 5 seconds, with a GROUP HELLO packet to inform network nodes of the existence of group **G** and of **G**'s current sequence number. Nodes that wish to join group **G** or want to send packets to **G**, and have recently received a GROUP HELLO packet from its leader, unicast a ROUTE REQUEST packet to the group leader, rather than flooding it. Once the leader receives a ROUTE REQUEST, or any node on the shared group tree receives a flooded ROUTE REQUEST, it unicasts a ROUTE REPLY packet back to the originator of the REQUEST, which responds with a MULTICAST ACTIVATION packet. The MULTICAST ACTIVATION packet sets up multicast forwarding state between the newly joined receiver and the shared tree.

Each MAODV multicast tree node keeps a list of its upstream and downstream neighbors in the shared tree. Each data packet is forwarded to all nodes on this list except the node from which it was received. The packet is forwarded as either a unicast to each such neighbor, or as a broadcast, when it needs to be forwarded on to multiple nodes.

Broken links are detected with the help of periodic HELLO packets broadcast by each node in the network. In addition, any packet overheard from a node, can serve as an indication that the link to that node is operational. When a node does not receive any packets from another node within several HELLO interval times, the link between the two nodes is assumed to be broken, and the node uses expanding ring search flooding to reconnect to the shared tree.

MAODV also employs mechanisms for network partition detection and partition healing, which include group leader selection, to ensure that there is only one group leader in each network partition [16].

3.3. ODMRP

ODMRP builds a group-shared forwarding mesh for each group. Each source performs periodic flood-response cycles, which create multicast forwarding state regardless of existing forwarding state. The frequent state discovery enabled the protocol to discover the current shortest paths between each source and the multicast receivers and improves the robustness of the protocol because multiple forwarding paths may exist between the members of the group. This is also why ODMRP's packet delivery ability improves as the number of sources and receivers per multicast group increases and sometimes with increased mobility: the redundant forwarding state improves ODMRP's packet delivery ability because it serves as a form of forward error correction, and makes the protocol less susceptible to mesh disconnection due to broken links. However, the frequent discovery floods and high number of data transmissions significantly increase network load.

Each multicast source for a group **G** in ODMRP periodically floods the network with a JOIN QUERY packet which is forwarded by all nodes in the network. This packet is sent every REFRESH_INTERVAL, e.g., every 3 seconds. Each multicast receiver responds to this flood by sending a JOIN REPLY packet which is forwarded along the shortest path back to the multicast source that originated the QUERY. Before forwarding this packet, each node waits for JOIN_AGGREGATION_TIMEOUT, and combines all JOIN REPLYs for the group received during this time into one JOIN REPLY. Each node that forwards the REPLY packet creates (or refreshes) forwarding state for group **G**.

Each node with forwarding state for **G** forwards each data packet sent by a multicast source for **G** that it receives. A data packet thus follows the shortest paths to the multicast receivers within the forwarding mesh, though is also forwarded towards other sources for the group who may not be group members. Forwarding state is expired after a multiple of the periodic flooding interval to ensure that in the event that some number of forwarding nodes' multicast state is not refreshed due to packet loss, the forwarding state created from a previous flood would still be valid. This mechanism improves the robustness of the protocol, but may cause multiple overlapping trees to be active in the network simultaneously, each created during a subsequent JOIN QUERY flood [9].

Table 1: Scenario Characteristics

Metric	100 nodes	200 nodes
Average Node Degree	23.24	23.40
Average Shortest-Path Length	2.34	3.32
Maximum Shortest-Path Length	7.5	10.9
# Link Changes Per Sec.	43.1	96
# Shortest-Path Changes Per Sec.	496	3130

4. Methodology

4.1. Simulation Environment

The simulation setup used in this evaluation is consistent with that commonly used in ad hoc network routing protocol performance studies.

We implemented the three protocols in the ns-2 [5] discrete event packet-level simulator with Monarch wireless extensions [2], which include implementations of models of signal strength, radio propagation, wireless medium contention, capture effect, and node mobility. The radio model is based on the Lucent Technologies WaveLAN 802.11 product [7], which provides a 2 Mbps transmission rate and a nominal transmission range of 250m.

The values of the main parameters of each protocol are listed in tables 2, 3, and 4, and have been set as suggested by their designers in published work, and through personal communication.

4.2. Mobility Scenarios

The experiments include networks with 100 nodes placed on a site with dimensions 1200×800 meters, and networks with 200 nodes on a site with dimensions 1720×1120 meters. The node density and ratio of dimensions of each site are nearly identical between the two sites to enable comparison.

Nodes in each scenario move according to the random waypoint model [2], in which each node independently picks a random destination and speed from an interval $(0, Max\_Speed)$ and moves towards the chosen destination at the selected speed. When the node reaches the destination, it stops for *pause time* seconds and then repeats the process. The pause time in all experiments is 0, i.e., nodes move continuously; the maximum speed is 20 m/s.

Each simulation is run for 900 seconds. Ten randomly generated scenarios are run for each parameter combination, and each point in the graphs is the average of the results of these ten scenarios.

The characteristics of the movement scenarios are shown in Table 1, where each value is an average over the 10 scenarios per parameter combination. The *average node degree* is the number of nodes that are within direct transmission range of a node, and is averaged over all nodes and over the duration of the simulation. The *average shortest-path length* is the average path length computed over all shortest paths between all pairs of nodes, over the duration of the simulation. The *maximum shortest-path length* is the length of the longest shortest-path between a pair of nodes encountered during the simulation. The *number of link changes per second* is the number of times a link is formed or breaks divided by the length of the simulation. The *number of shortest-path changes per second* is the average number of times that a shortest route between two nodes breaks or a shorter path becomes available.

Table 2: ADMR Parameter Settings

Parameter Name	Value
MAX_RECEIVER_JOIN_FWDS_PER_FLOOD	3
DEFAULT_EXPECTED_PKT_INTERARRIVAL_TIME	0.2
DEFAULT_KEEPALIVE_COUNT	16
DEFAULT_MULTIPLICATION_FACTOR	1
BEGIN_LOCAL_REPAIR_DELAY	0.2
LOCAL_REPAIR_TTL	2
NUM_MISSING_PKTS_TRIGGER_DISCONNECTION	3
NUM_MISSING_PKTS_TRIGGER_EXPIRATION	10
TEMPORARY_STATE_EXPIRATION	3 sec.

Table 3: MAODV Parameter Settings

Parameter Name	Value
HELLO_INTERVAL	1 sec.
ALLOWED_HELLO_LOSS	3
RREQ_RETRIES	2
GROUP_HELLO_INTERVAL	5 sec.
ACTIVE_ROUTE_TIMEOUT	6 sec.
RREP_WAIT_TIME	2 sec.
TTL_INCREMENT	2
TTL_THRESHOLD	7

Table 4: ODMRP Parameter Settings

Parameter Name	Value
REFRESH_INTERVAL	3 sec.
JOIN_AGGREGATION_TIMEOUT	0.025 sec.
RREQ_RETRIES	2
FORWARDING_STATE_TIMEOUT	$3\times$REFRESH_TIMEOUT
JOIN_REPLY_PASSIVE_ACK_TIMEOUT	2 sec.
MAX_NUM_JOIN_REPLY_RETRIES	7

In the 200-node scenario, the average node degree is similar to that in the 100-node scenario because the network density between the scenarios is nearly the same. The number of link changes per second is 2.2 times higher in the 200-node case as there are twice as many nodes, while the number of shortest path changes is 6.3 times higher, since paths are longer and thus the creation or breaking of one link may lead to the creation or breaking of multiple paths.

4.3. Communication Scenarios

The multicast sources begin sending data and the multicast receivers join a multicast group at uniformly randomly chosen times between 0 and 180 seconds from the beginning of the simulation and remain in the multicast session until the end of the simulation. In all scenarios, Constant Bit Rate (CBR) traffic generators send 64-byte packets. This packet size was chosen in order to reduce the likelihood of congestion. The packet rates are chosen to continuously probe the routing ability of the protocols rather than to represent any particular application.

In the rest of this paper, we use the notation G×S×R, where *G* is number of multicast groups, *S* is number of multicast sources per group, and *R* is number of multicast receivers per group. In addition, in the notation G×N, *G* is the number of multicast groups, and *N* is the number of nodes in each group, where each node is both a source and a receiver for the group.

1. *Varying the number of multicast receivers:* This set of experiments was designed to explore the effect on protocol performance of the number of multicast receivers within a multicast group. We used three single-source and three multi-source, multi-group scenarios in which the number of receivers is progressively increased: 1×1×10, 1×1×50, 1×1×99 and 2×3×10, 2×3×20, 2×3×40. In all scenarios, each source sends four 64-byte packets per second.
2. *Varying the number of multicast sources:* This set of experiments was designed to explore the effect on protocol performance of the number of sources in a multicast group. We used three scenarios with one group and three scenarios with multiple groups, in which the number of sources is progressively increased: 1×1×20, 1×5×20, 1×10×20 and 2×1×20, 2×5×20, 2×10×20. The generated traffic is kept the same across all single group scenarios and across all multi-group scenarios. In particular, the load is ten 64-byte packets per second per source, in the 1-source scenarios, two 64-byte packets per second per source in the 5-source scenarios, and one 64-byte packet per second per source in the 10-source scenarios.
3. *Single-source groups vs. multi-source groups*: This set of experiments was designed to explore the effect of the distribution of sources among groups. We used 3 sets of two scenarios, with an increasing number of sources between sets: 1×3×15, 3×1×5, 1×5×50, 5×1×10, and 1×9×27, 9×1×3. The generated traffic is kept the same across each pair of scenarios and is four 64-byte packets per second per source.
4. *Conferencing*: This set of experiments was designed to explore the performance of the multicast protocols in serving conferencing applications in which all nodes are simultaneously multicast sources and receivers. In particular, we used the following set of scenarios: 1×5, 1×10, 1×20. The generated traffic is kept the same across all scenarios. In particular, the load is ten 64-byte packets per second per source, in the 5-source scenario, two 64-byte packets per second per source in the 10-source scenario, and one 64-byte packet per second per source in the 20-source scenario.
5. *Increasing network size*: This set of experiments was designed to explore the differences in the protocols' performance when the network size is increased. We increase the number of nodes from 100 to 200, and use the multicast scenarios from item 2 above (varying the number of multicast sources).

4.4. Performance Metrics

Protocol performance will be evaluated using the following metrics, which are computed over the whole duration of the simulation:

- *Packet Delivery Ratio:* The fraction of packets sent by the multicast application that are received by the multicast receivers. For example, if there are 2 sources and 5 receivers, and each source sends 10 packets, and each receiver receives 12 multicast packets total, the packet delivery ratio would be $(5 \times 12)/(5 \times 20)$, which is 0.6.
- *Control Packet Overhead*: The total number of control packets originated and forwarded by the protocol.

- *Normalized Packet Overhead:* The number of control and data transmissions performed by the protocol per successfully delivered data packet. This metric evaluates the overall effort that the protocol expends for the delivery of a each data packet. For example, multicast efficiency of 5 means that the protocol makes 5 packet transmissions on average for each data packet that is delivered to a multicast receiver.
- *End-To-End Delay:* The time between the transmission of a data packet from a multicast source and the time of its reception by a multicast receiver, averaged over all packets.

We do not discuss control byte overhead in this paper as the results are very similar to those of control packet overhead. ADMR adds several bytes to each data packet but because of its efficient forwarding, generates less data byte overhead than MAODV and ODMRP. In addition, packet overhead is a better determinant of performance because it dominates the time spent by packets at the MAC layer.

5. Results

5.1. Varying the Number of Multicast Receivers

Increasing the number of receivers leads to the creation of more multicast forwarding state in the network, and consequently to a higher number of data packet transmissions; this increase leads to a higher packet delivery ratio since nodes have more opportunities to receive a transmission of each packet. MAODV and especially ODMRP create a large amount of redundant forwarding state even for small groups, and so increasing the number of receivers in these protocols has a small effect on their packet delivery ratio.

All three protocols have high packet delivery ratios in these scenarios (Figure 1). In the single-source scenarios, the protocols perform comparably and deliver almost all of their packets; in the multi-source scenarios, collision losses caused by the higher network load lead to slightly lower packet delivery ratios. The decrease is most pronounced in MAODV, since it uses longer data forwarding paths, which are more prone to collision losses.

ADMR generates up to 14 times less control packet overhead than MAODV and up to 5 times less overhead than ODMRP (Figure 2). The high control overhead in MAODV is due to the periodic group leader floods and the periodic neighbor "Hello" packets. The periodic nature of the overhead is also the reason why it does not change significantly between scenarios. ODMRP's high overhead is as a result of its periodic source flood and response cycles, with the response part of the cycle growing with the number of receivers. While ADMR's and ODMRP's control packet overhead increases when there are more receivers in the network, MAODV's overhead decreases. In ADMR and ODMRP, the presence of more receivers means a larger number of receiver joins, and in the case of ADMR, more mesh repairs. The decrease in overhead with MAODV is due to the fact that its ability to localize join and repair floods improves when there are more nodes in its group shared tree.

ADMR is the most efficient of the three protocols overall and has the lowest normalized packet overhead, while MAODV generates the highest such overhead (Figure 3). The normalized packet overhead decreases with an increase in the number of receivers, because it is dominated by the data packet transmissions generated by the protocols, which are better amortized when the number of receivers increases.

End-to-end delay is highest for MAODV due to the longer paths that data packets have to follow within the shared tree and due to the higher network load caused by the high number of control

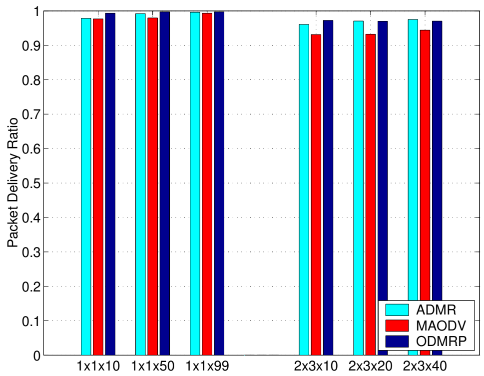

Figure 1: Varying the Number of Receivers: Packet Delivery Ratio (100 Nodes)

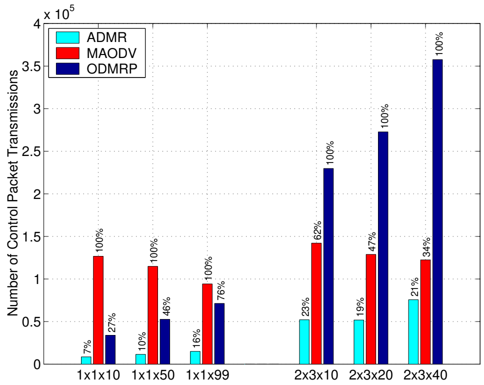

Figure 2: Varying the Number of Receivers: Control Packet Overhead (100 Nodes)

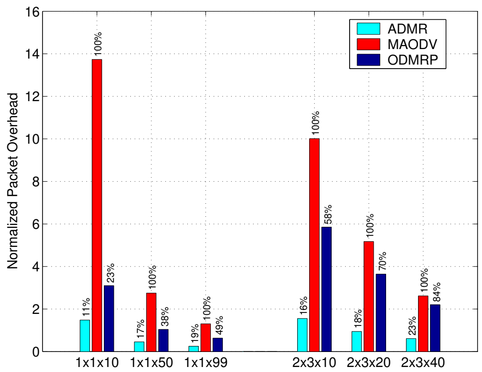

Figure 3: Varying the Number of Receivers: Normalized Packet Overhead (100 Nodes)

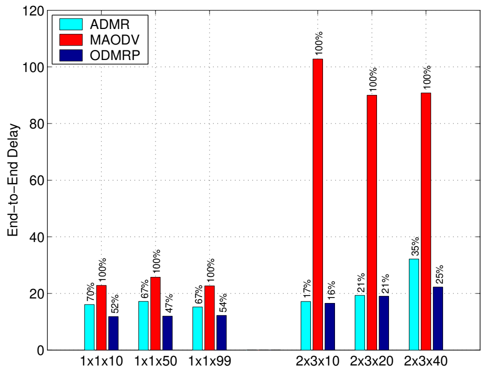

Figure 4: Varying the Number of Receivers: End-to-End Delay (100 Nodes)

and data packet transmissions (Figure 4). Network load translates into a busier wireless medium, which causes nodes to have to wait longer before forwarding each packet. ODMRP's end-to-end delay is lowest because, due to its frequent state discovery floods, it uses the shortest forwarding paths among the three protocols.

5.2. Varying the Number of Multicast Sources

As the number of sources grows, MAODV and ADMR maintain their packet delivery ratio, while ODMRP's decreases (Figure 5). This decrease is more noticeable in the multi-source scenarios and is a result of packet loss due to increased congestion in the network caused by the high control and data packet overhead generated by the protocol (Figure 7).

ODMRP's control packet overhead scales approximately linearly with each added multicast source because each ODMRP source initiates periodic control packet request and response cycles and so the per-source control overhead is fixed given a fixed set of receivers. ADMR scales its overhead sublinearly with the number of sources since even though each source creates and operates its own tree, when the receiver members join a multicast group before a given source

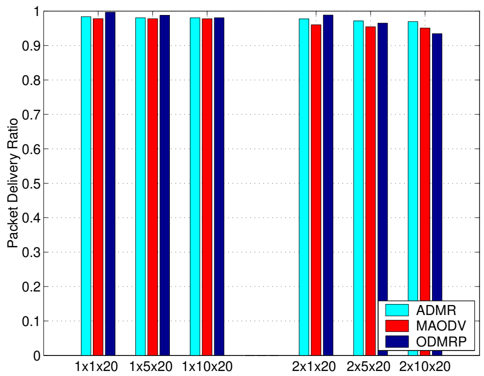

Figure 5: Varying the Number of Sources: Packet Delivery Ratio (100 Nodes)

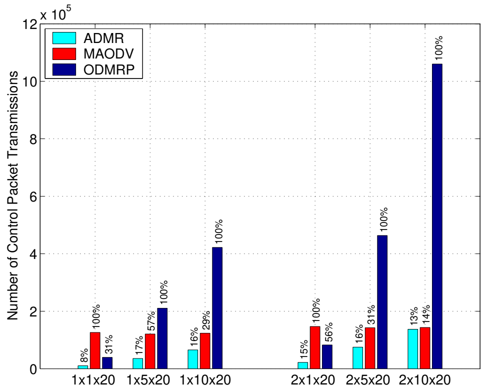

Figure 6: Varying the Number of Sources: Control Packet Overhead (100 Nodes)

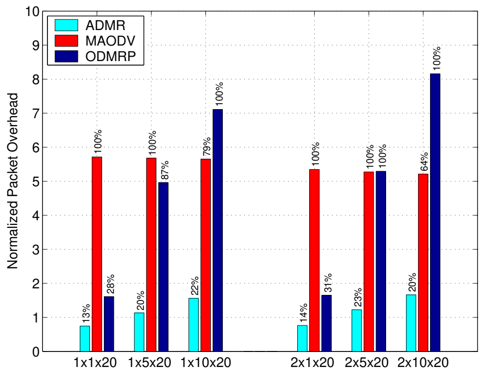

Figure 7: Varying the Number of Sources: Normalized Packet Overhead (100 Nodes)

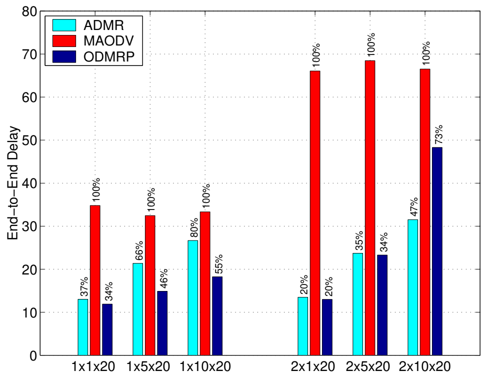

Figure 8: Varying the Number of Sources: End-to-End Delay (100 Nodes)

becomes active, once the source does become active, many RECEIVER JOIN packets sent by the receivers in response to a new source's data flood are filtered. MAODV displays near-constant overhead as additional sources for the same group, only generate a small amount of overhead to join the shared tree through the closest tree node to them.

ADMR generates the lowest normalized packet overhead in all scenarios (Figure 7). Both ADMR and ODMRP's normalized packet overhead increases when there are more multicast sources due to the higher number of state that needs to be created and maintained for each new active source. MAODV's normalized packet overhead remains nearly the same across all scenarios since the shape of its shared tree does not change significantly when new sources are added to it.

MAODV packets experience the highest end-to-end delay, as a result of the longer paths along which they are forwarded, and as a result of the high network load caused by control and data packet transmissions performed by the protocol. In ADMR and ODMRP end-to-end delay increases with the increase in network load, and in the case of scenario $2\times10\times20$, ODMRP even begins to route along longer paths than ADMR as the concentration of traffic along the shortest paths is high and leads to congestion and collision losses.

5.3. Single-Source vs. Multi-Source Groups

Group shared mesh and tree protocols are usually intended to optimize the performance in multi-source groups, while single-source protocols are often assumed to work best for single-source groups. However, in our experiments, these statements frequently do not hold (Figure 9). The packet delivery ratio for multi-source groups is higher than the packet delivery ratio for single-source groups for ADMR in all simulated scenarios, and the opposite is true for all but the 3-source ODMRP scenario. In the case of ADMR, the higher packet delivery ratio for multi-source groups is due to a higher data forwarding redundancy resulting from the fact that each multicast mesh contains more receivers, and thus more paths, than in the single-source scenarios. ODMRP's packet delivery ratio decreases in multi-source groups with more than 3 sources because it generates too much forwarding redundancy and control overhead when each group contains multiple sources and this additional network load hurts its performance. MAODV's packet delivery ratio is higher in the multi-source experiments than in the single-source experiments since only one tree is built per group, and so with fewer groups, there is less network overhead. In addition, MAODV also benefits from higher data forwarding redundancy when there are more receivers within a single group, as a packet forwarded by one node may be overheard by more nodes that need to forward or receive it.

ODMRP incurs higher control overhead in multi-source groups (Figure 10) than in single-source groups because more JOIN REPLY packets are sent when there are more receivers, and since the mesh is also larger in this case, each REPLY is forwarded more times. MAODV generates more control overhead when more shared trees need to be setup because its overhead increases with the number of groups in the network. As a result, its overhead is higher in the single-source scenarios.

ADMR generates almost the same control packet overhead between the single and multi-source scenarios in all cases except in the heavy (9-source) scenario, where it generates a little more control overhead in the multi-source case. The generated overhead is similar between multi-source and single-source scenarios since ADMR builds a source mesh for each multicast source. The different behavior in the 9-source scenario occurs because the high number of receivers within a single source mesh results in traffic concentration which leads to collision losses which may trigger repairs even though no links are actually broken.

Overall, normalized packet overhead is lower in the multi-source scenarios for all protocols. ADMR generates the lowest normalized packet overhead in all scenarios, and MAODV generates the highest such overhead.

End-to-end delay is lowest for ODMRP due to the shorter paths that it uses followed closely by ADMR, with MAODV having the highest end-to-end delay. In the $1\times9\times27$ scenario, ODMRP and MAODV are affected by the high control and data packet load they impose on the network, and incur a significantly higher end-to-end delay.

5.4. Conferencing

Both ADMR and MAODV perform well in the conferencing scenarios and deliver around 95% of their packets (Figure 13). ODMRP delivers less than 90% of its data in these scenarios because it is not able to create enough redundant forwarding state to compensate for links that break due to node mobility. As group size increases, it is able to create more forwarding state and its packet delivery ratio improves.

Both ADMR and ODMRP's control packet overhead increases with the increase in group size, while MAODV's control overhead remains the same (Figure 14). In MAODV the presence of

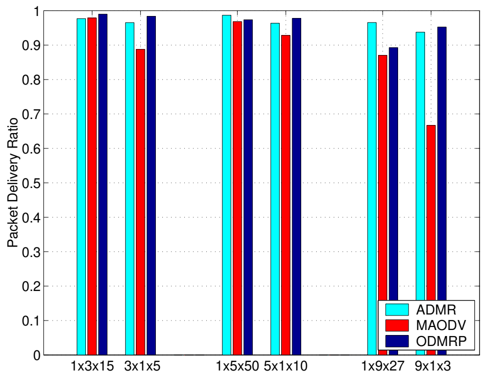

Figure 9: Single Source vs. Multi-Source Groups: Packet Delivery Ratio (100 Nodes)

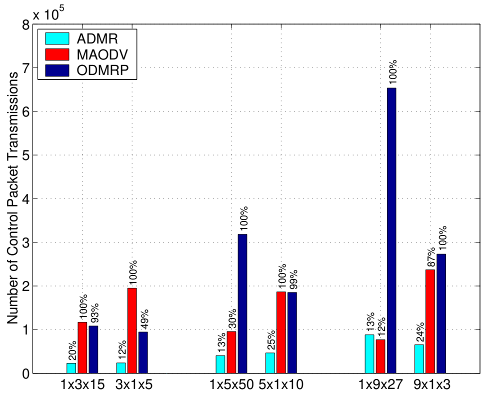

Figure 10: Single Source vs. Multi-Source Groups: Control Packet Overhead (100 Nodes)

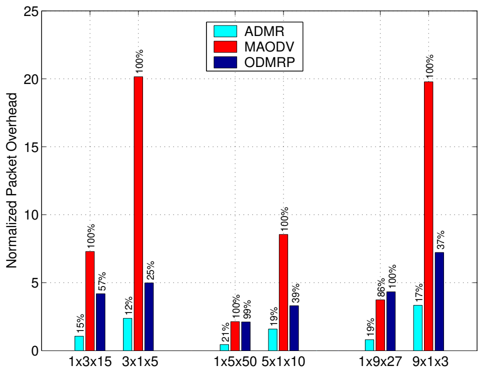

Figure 11: Single Source vs. Multi-Source Groups: Normalized Packet Overhead (100 Nodes)

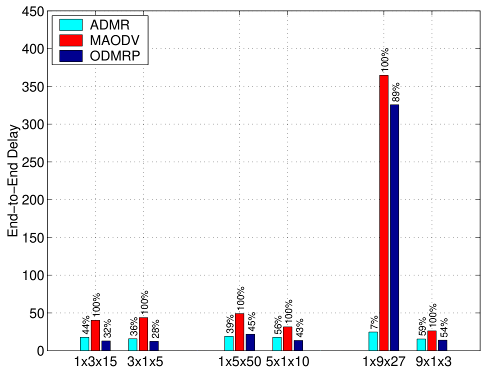

Figure 12: Single Source vs. Multi-Source Groups: End-to-End Delay (100 Nodes)

more shared tree nodes in the network enables better localization of its group joins and repairs. This localization is so successful, that MAODV generates the least control overhead in the 1×10 and 1×20 among the three protocols.

Overall, ADMR is the most efficient in the conferencing scenarios, generating up to 7 times less normalized packet overhead than MAODV and up to 5 times less normalized packet overhead than ODMRP (Figure 15).

End-to-end delay (Figure 16) in MAODV is high relative to that of ADMR and ODMRP in the 5 and 10 group scenarios, but decreases as the group size increases and in the 20 source scenario is lower than that of ADMR. The reduction in end-to-end delay is due to the fact that the packet load generated by the protocol is nearly constant and is amortized among an increasing number of nodes. In ADMR and ODMRP, end-to-end delay increases due to the higher number of overhead packets and an increase in path length due to traffic concentration along the shortest paths.

5.5. Increasing Network Size

The 200-node scenarios are more challenging than the 100-node scenarios since data forwarding paths are longer, and the number of link and route changes is higher (Section 4.2). As a result, the

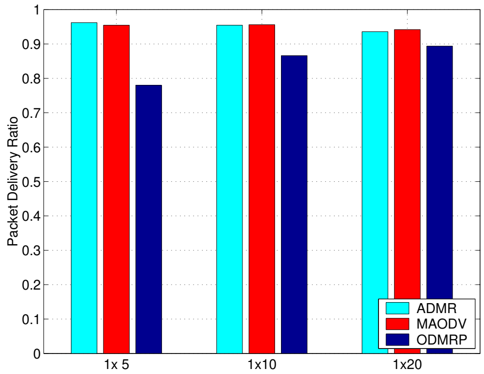

Figure 13: Conferencing Applications: Packet Delivery Ratio (100 Nodes)

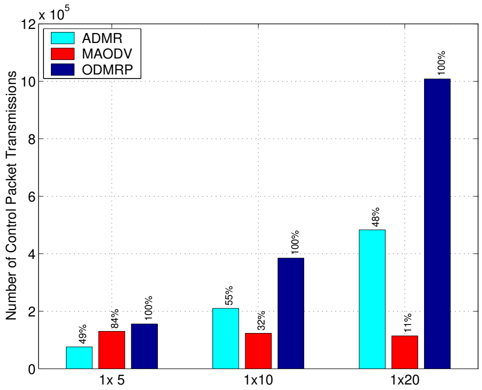

Figure 14: Conferencing Applications: Control Packet Overhead (100 Nodes)

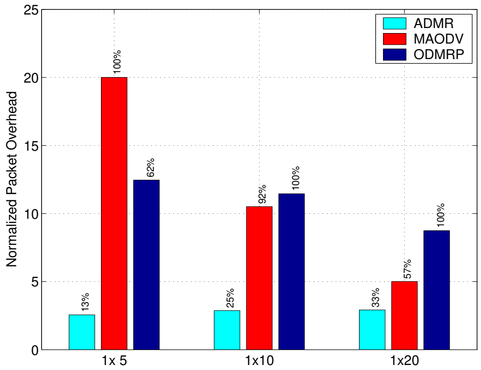

Figure 15: Conferencing Applications: Normalized Packet Overhead (100 Nodes)

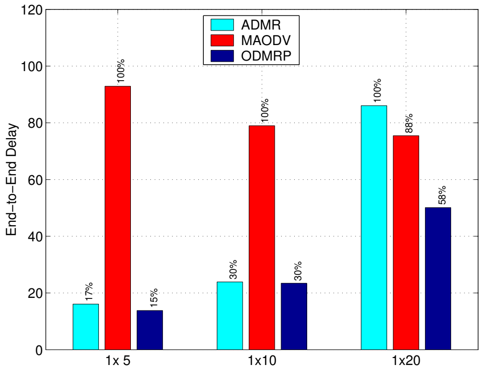

Figure 16: Conferencing Applications: End-to-End Delay (100 Nodes)

likelihood of packet loss is higher, and the number of falsely triggered repairs increases as well. In addition, in the 200-node scenarios, group members are more sparsely distributed in the network than they would be in a 100-node scenario, which leads to the creation of more forwarding state, though there is less redundant forwarding and as a result packet loss has a stronger impact on the packet delivery ratio.

Table 5 shows the ratio of the values of all performance metrics for the 200-node and 100-node scenarios in the experiment in which the number of sources is increased (Section 5.2).

ODMRP's packet delivery ratio is the same as in the 100-node scenarios, while ADMR and MAODV's decrease slightly in the multi-source scenarios. This decrease is due to the fact that neither protocol performs as much redundant forwarding as ODMRP and since they also do not perform buffering, more packets are lost during mesh repairs, which are more frequent in the 200-node scenarios due to the longer data forwarding paths. MAODV is more affected than ADMR because it uses longer forwarding paths.

Control packet overhead increases by about a factor of 2 for all protocols, with MAODV and ADMR experience a slightly higher increase than ODMRP. The increase is due to the higher number of nodes forwarding each flood, and in the case of ADMR and MAODV, due to the higher number of link and route changes, which require more repairs to be performed.

Table 5: Increasing Network Size: Ratio between Metrics for 200-Node Scenarios and 100-Node Scenarios

Metric	Protocol	1×1×20	1×5×20	1×10×20	2×1×20	2×5×20	2×10×20
Packet Delivery Ratio	ADMR	0.99	0.99	0.99	0.98	0.98	0.97
	MAODV	1.00	1.00	1.00	0.95	0.95	0.96
	ODMRP	1.00	1.00	1.00	1.00	1.00	0.99
Control Packet Overhead	ADMR	2.34	2.32	2.37	2.40	2.45	3.23
	MAODV	2.27	2.11	2.29	2.11	2.09	2.07
	ODMRP	1.82	1.87	1.90	1.83	1.95	1.85
Norm. Packet Overhead	ADMR	1.97	2.10	2.18	1.96	2.16	2.48
	MAODV	2.07	2.05	2.07	2.03	2.05	2.01
	ODMRP	1.79	1.91	1.95	1.76	1.92	1.97
End-To-End Delay	ADMR	1.50	1.32	1.17	1.46	1.33	2.49
	MAODV	5.07	4.88	3.55	3.63	4.74	4.17
	ODMRP	1.45	1.51	1.54	1.43	1.63	3.11

Overall, in the 200-node scenarios, ADMR remains the most efficient and generates the least normalized packet overhead, while MAODV generates the most; all protocols generate about twice as much such overhead than in the 100-node scenarios.

End-to-end delay increases by 1.5 to 3 times for ODMRP and ADMR, while MAODV's end-to-end delay increases by 3.5-5 times. The higher increase in MAODV is due to the higher increase in path length and also the traffic concentration due to its use of shared group trees.

6. Conclusion

In this paper, we presented an extensive performance comparison of ADMR, MAODV, and ODMRP in a variety of simulation scenarios. We studied mobile networks composed of 100 or 200 nodes, with both a single active multicast group, and multiple active multicast groups in the network, along with a variety of multicast scenarios such as conferencing and single-source vs. multi-source groups. Our experiments show that all protocols perform well in terms of packet delivery ratio, and the differences in behavior are in their efficiency. ADMR is the most efficient across all scenarios and typically generates 3 to 5 times less normalized packet overhead than MAODV and ODMRP, because it scales its overhead to meet existing communication demands. MAODV incurs a high level of packet overhead due to its use of periodic Hello packets and because it uses a shared group tree, which enables it to localize multicast group joins but results in longer data forwarding paths. Due to its use of longer forwarding paths, MAODV performs more data packet transmissions, suffers from more packet losses and collisions, and higher end-to-end delay, compared to ADMR and ODMRP. ODMRP uses the shortest paths among the three protocols, and overall incurs the smallest end-to-end delay, because it performs frequent periodic state discovery floods. These floods also result in the creation of a large amount of forwarding state within the network, which improves the robustness of the protocol against mesh disconnection or packet loss, but at the cost of significantly increasing network load.

References

[1] Bommaiah, McAuley, and Talpade. AMRoute: Adhoc Multicast Routing Protocol. Internet-Draft, draft-talpade-manet-amroute-00.txt, February 1999. Work in progress.

[2] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta G. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 85–97, October 1998.

[3] S. Bunchua and C.-K. Toh. Performance Evaluation of Flooding-Based and Associativity-Based Ad Hoc Mobile Multicast Routing Protocols. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, September 2000.

[4] C.-C. Chiang, Mario Gerla, and Lixia Zhang. Forwarding Group Multicast Protocol (FGMP) for Multihop, Mobile Wireless Networks. *ACM Baltzer Journal of Cluster Computing: Special Issue on Mobile Computing*, 1(2):187–196, 1998.

[5] Kevin Fall and Kannan Varadhan, editors. *ns* Notes and Documentation. The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, November 1997. Available from http://www-mash.cs.berkeley.edu/ns/.

[6] J.J. Garcia-Luna-Aceves and E.L. Madruga. A Multicast Routing Protocol for Ad-Hoc Networks. In *Proceedings of the IEEE Conference on Computer Communications, INFOCOM 99*, pages 784–792, March 1999.

[7] IEEE Computer Society LAN MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std 802.11-1997. The Institute of Electrical and Electronics Engineers, New York, New York, 1997.

[8] Jorjeta G. Jetcheva, Yih-Chun Hu, David Maltz, and David B. Johnson. A Simple Protocol for Multicast and Broadcast in Mobile Ad Hoc Networks. Internet-Draft, draft-ietf-manet-simple-mbcast-00.txt, November 2000. Work in progress.

[9] Jorjeta G. Jetcheva and David B. Johnson. Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks. In *Proceedings of the 2001 ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2001)*, pages 33–44, October 2001.

[10] L. Ji and M. S. Corson. A Lightweight Adaptive Multicast Algorithm. In *Proceedings of IEEE GLOBECOM '98*, pages 1036–1042, December 1998.

[11] L. Ji and M. S. Corson. Differential Destination Multicast (DDM) Specification. Internet-Draft, draft-ietf-manet-ddm-00.txt, July 2000. Work in progress.

[12] H-J. Kang and M-J. Lee. A Multi-source Multicast Routing Protocol for Ad-Hoc Networks. In *Lecture Notes in Computer Science Journal (LNCS)*, pages 309–320, January 2002.

[13] T. Kunz and E. Cheng. Multicasting in ad-hoc networks: Comparing MAODV and ODMRP. In *Proceedings of the Workshop on Ad Hoc Communications*, September 2001.

[14] S. Lee and C. Kim. Neighbor Supporting Ad Hoc Multicast Routing Protocol. In *Proceedings of the First Annual Workshop on Mobile Ad Hoc Networking and Computing (MobiHoc 2000)*, August 2000.

[15] 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 2000*, pages 565–574, March 2000.

[16] Elizabeth M. Royer and Charles E. Perkins. Multicast Operation of the Ad-hoc On-Demand Distance Vector Routing Protocol. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking, Mobicom '99*, pages 207–218, August 1999.

[17] S.-J.Lee, Mario Gerla, and C.-C. Chiang. On-Demand Multicast Routing Protocol. In

Proceedings of the IEEE Wireless Communications and Networking Conference, WCNC '99, pages 1298–1304, September 1999.

[18] P. Sinha, R. Sivakumar, and V. Bharghavan. MCEDAR: Multicast core extraction distributed ad-hoc routing. In *In Proceedings of the Wireless Communications and Networking Conference, WCNC '99.*, pages 1313–1317, September 1999.

[19] C.-K. Toh, Guillermo Guichala, and Santithorn Bunchua. ABAM: On-Demand Associativity-Based Multicast Routing for Ad Hoc Mobile Networks. In *Proceedings of IEEE Vehicular Technology Conference, VTC 2000*, pages 987–993, September 2000.

[20] C.W. Wu, Y.C. Tay, and C-K. Toh. Ad hoc Multicast Routing protocol utilizing Increasing id-numberS (AMRIS). Internet-Draft, draft-ietf-manet-amris-spec-00.txt, November 1998. Work in progress.