Unicast Performance Analysis of the ODMRP in a Mobile Ad hoc Network Testbed

Sang Ho Bae, Sung-Ju Lee, and Mario Gerla
Wireless Adaptive Mobility Laboratory
Computer Science Department
University of California
Los Angeles, CA 90095-1596

Abstract ***– The On-Demand Multicast Routing Protocol (ODMRP) is an effective and efficient routing protocol designed for mobile wireless ad hoc networks. One of the major strengths of ODMRP is its capability to operate both as a unicast and a multicast routing protocol. This versatility of ODMRP can increase network efficiency as the network can handle both unicast and multicast traffic with one protocol. We describe the unicast functionality of ODMRP and analyze the protocol performance in a real ad hoc network testbed of seven laptop computers in an indoor environment. Both static and dynamic networks are deployed. We generate various topological scenarios in our wireless testbed by applying mobility to network hosts and study their impacts on our protocol performance. We believe that the performance study in a testbed network can help us analyze the protocol in a realistic way and point us to the future research direction.***

I. INTRODUCTION

With the advance in networking and communications technologies, portable wireless devices are found in our common activities. Most people carry and use laptop computers, cellular phones, and pagers that support nomadic computing of network users. An ad hoc network, or a packet radio network, which is one form of wireless networks, is recently receiving a lot of attention from the wireless communication research community. Ad hoc networks are built without the infrastructure support of wired base stations. Hence, they are attractive in situations where a network must be easily deployable. Since no base station exists, each node must communicate with one another via packet radios. Because of the limited radio propagation range, the destination node may not be within the transmission range of the source node. In order to communicate with nodes outside the proximity, multihop routes need to be built and intermediate nodes must forward packets from the source to the destination. Each node in the network can be mobile, and hence multihop routes can be disconnected frequently. Routing and multicasting protocols are therefore extremely important in ad hoc communication networks. Mobility, in combination with limited bandwidth and power, makes the routing protocol design challenging.

The On-Demand Multicast Routing Protocol (ODMRP) [12], [13] is a multicast protocol for mobile ad hoc networks. ODMRP builds routes *on demand* and uses a *mesh* to create alternate and redundant multicast routes in the face of mobility and topology changes. A *soft-state* approach is taken in ODMRP to maintain and refresh routes. One of the key strengths of ODMRP is its unicast routing capability. Not only can ODMRP coexist with any unicast routing protocol, it can also operate efficiently as a unicast routing protocol. Thus, a network equipped with ODMRP does not require a separate unicast protocol. Ad-hoc On-Demand Distance Vector (AODV) [15] is another protocol that can function both as unicast and multicast, but its real network implementation has not been completed. Other ad hoc multicast routing protocols such as Adhoc Multicast Routing (AMRoute) [2], Core-Assisted Mesh Protocol (CAMP) [6], Reservation-Based Multicast (RBM) [3], and Lightweight Adaptive Multicast (LAM) [8] must be run on top of a unicast routing protocol. In contrast, ODMRP offers the advantage of sharing the same optional software for both unicast and multicast operation.

This paper presents the mechanism of operating ODMRP as a unicast routing protocol. We then describe the ODMRP implementation details and report performance analysis in a real wireless ad hoc network testbed composed of seven laptop computers. Unicast performance of the protocol is evaluated in both static network and mobile network. We generate various topological scenarios by introducing mobility to different hosts of the network. In one experiment, the destination node is mobile and in another experiment, an intermediate node roams within the building of our testbed. We believe the performance study in a testbed network can help us analyze the protocol in a realistic way and point us to the future research direction.

There are several related works that built ad hoc wireless testbeds. Monarch project team of Carnegie Mellon University recently developed a multihop wireless ad hoc network testbed on existing BSD Unix network stack [14]. A unicast routing protocol Dynamic Source Routing (DSR) [10] was implemented and tested in an outdoor environment. University of California at Santa Cruz developed wireless Internet Protocol (IP) routers, Wireless Internet Gateways (WINGS) [5] and Secure Protocols for Adaptive, Robust, Reliable, and Opportunistic WINGs (SPARROW) [4] for ad hoc networks. Using the C++ Protocol Toolkit (CPT), protocol softwares were transitioned from a simulation environment to an embedded system. University of Maryland also developed an ad hoc network testbed on Linux 2.1 kernel [9]. Other works that built ad hoc network testbeds include the SURAN project [1] and Task

This work was funded in part by Intel and the Defense Advanced Research Projects Agency (DARPA) under contract DAAB07-97-C-D321, as a part of the Global Mobile Information Systems (GloMo) program.

Force XXI [18].

The rest of the paper is organized as follows. Section II explains how ODMRP functions as a unicast routing protocol followed by the protocol implementation description in Section III. Protocol performance evaluation in our seven node wireless mobile ad hoc network testbed is presented in Section IV, and concluding remarks are made in Section V.

II. UNICAST ROUTING FUNCTIONALITY OF ODMRP

In this section, we describe how ODMRP operates as a unicast routing protocol. Readers are referred to [13] for the multicast mechanism of ODMRP.

ODMRP builds and maintains routes on demand by the source. A query phase and a reply phase comprise the protocol. When a source has to communicate with a node but no route information to that destination is known, it floods a control packet called Join Query with a piggybacked data payload. When a node receives a non-duplicate Join Query, it stores the last hop node information in its route table (i.e., backward learning) and rebroadcasts the packet. When the Join Query packet reaches the destination, the destination replies back to the source via the selected route with the Join Reply packet.[1] Intermediate nodes of the route forward the Join Reply to the next hop towards the source of the route. The next hop node information is obtained from the routing table where the entry was recorded when Join Query was received. The Join Reply packet is propagated until it reaches the source of the route. This process constructs the route from the source to the destination. Figure 1 depicts the route $<S\text{-}i\text{-}j\text{-}k\text{-}D>$ establishment procedure. Note that we assume asymetric links and hence when node D needs to build a route to node S, it establishes a separate route that may or may not be the same as the route from S to D.

One drawback of on-demand routing protocols is the route acquisition latency. Since routes are only built when needed, the source must wait until a route is discovered before transmitting the first data packet of the session. To eliminate this delay, Join Query packets carry user data traffic in our protocol. Since the destination will receive the packet unless the network is partitioned, no route acquisition delay is needed. The size of flooded packet however, becomes larger. There is a tradeoff between delay and efficiency. When data payload size is too large, data piggybacking on Join Query should be avoided.

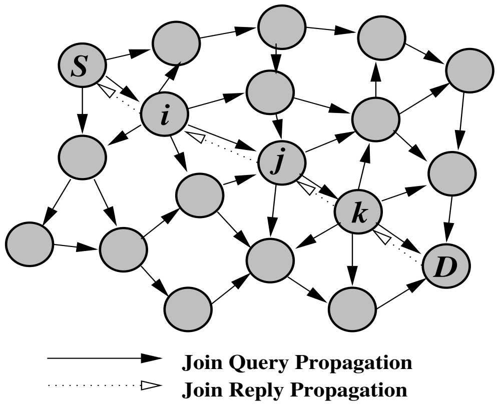


Fig. 1. On-demand procedure for route setup.

To use the most recent route information, our protocol enforces two policies that are different from other well-known on-demand routing protocols such as AODV and DSR. First, intermediate nodes are not allowed to reply from cache. Intermediate nodes cannot send a Join Reply in response to a Join Query even when they have route information to the destination node in their route table.[2] One reason is to deliver the data payload of the Join Query to the destination. If intermediate nodes send replies to the source and drop the Join Query packet, the destination cannot receive the data portion of the packet. The other reason is to utilize the most up-to-date topology information and build the shortest-distance route. Routes obtained from intermediate nodes yield longer hop distances since they do not account for node locations and network topology during and after movements.

Second, as long as the source still need to communicate with the destination, Join Query is periodically broadcasted to the entire network to update the route. Therefore, fresh routes are continuously built and utilized. The selection of periodic route refresh interval should be adaptive to network environment (e.g., traffic type, traffic load, mobility pattern, mobility speed, channel capacity, etc.). When small route refresh interval values are used, fresh route information can be obtained frequently at the expense of producing more packets and causing network congestion. On the other hand, when large route refresh values are selected, even though less control traffic will be generated, routes may not use recent topology information. Thus in highly mobile networks, using large route refresh interval values can yield poor protocol performance.

Even though the periodic route refresh reconstructs the routes, when a node of the route detects a route break during data propagation, it sends a Route Error message back to the source to invoke a fast route recovery process. A link disconnection is detected either by MAC layer feedbacks using reliable MAC protocols such as IEEE 802.11 [7], or by passive acknowledgments [11]. The source, upon receiving the Route

---

[1] Packet types Join Query and Join Reply have the term "Join" because ODMRP is originally a multicast protocol. These packets are exchanged to collect multicast group membership information as well as to build routes in multicast sessions, hence the term "Join." We keep the packet names the same in unicast mode even though group membership information is not obtained.

[2] Intermediate nodes can *relay* Join Replies from the destination to the source, of course.

ERROR packet, sends a JOIN QUERY for route reconstruction. In addition, it adjusts the next route refresh time to the current time plus the route refresh interval. Note that ROUTE ERROR messages do not exist in the ODMRP multicast operation since redundancy is created by multiple routes. In the unicast operation however, single path is maintained for each <source, destination> pair and no alternate path is available. Therefore, immediate route reconstruction is necessary.

III. IMPLEMENTATION

A. Implementation Platform

A.1 Operating System and Software

Our protocol is developed on Linux kernel version 2.2.12, the version provided by the Red Hat Linux version 6.1. All tools and software packages used in our development originate from software bundle incorporated within the Red Hat Linux version 6.1 operating system package. The Linux operating system is chosen for its availability, familiarity, and kernel level support for IPv4 forwarding.

In Section IV, the bandwidth utilization of applications in wireless multihop ad hoc network is studied by routing the unicast traffic through mobile routers and the end nodes running ODMRP.

A.2 Hardware

Ad hoc network nodes consist of Intel Pentium II based Hewlett Packard Omnibook 7150 laptops and WinBook XL laptops equipped with Lucent IEEE 802.11 WaveLan radio devices [19]. The WaveLan devices operate on 2.4 GHz bandwidth and communicate at the maximum capacity of 2 Mb/s with the semi-open space range of 150 meters. The WaveLan devices are operated in ad hoc mode.

B. Software Architecture

ODMRP uses the kernel level IPv4 forwarding support built into the Linux operating system. The Linux kernel supports packet forwarding by performing the following procedures. The user enables the IP forwarding option in the network protocol stack. The network interfaces accept and send all packets to the kernel. The kernel accepts all packets, checks the destination address with the kernel level routing table and decides whether to forward them. The messages are forwarded by altering the forwarding destination interface of the messages and buffering them to the corresponding interfaces. A message with the destination not specified in the routing table is forwarded to the default gateway. If there is no viable forwarding location, the packet is dropped and Internet Control Message Protocol (ICMP) [16] destination error message is sent. The message is sent to user level only if the routing host is the destination. This process avoids the costly kernel-to-user crossing and improves efficiency. The destination interface is changed in accordance with the listings in the kernel level routing table. The kernel level routing table is updated and maintained by a user level routing daemon which keeps its own user level routing table. The user level table is copied to the kernel level table as the updates are made. We used this basic routing table interface to build and maintain ODMRP routing tables both on kernel and user level. In the following sections, our schemes to manage the control packets and the routing table are described and a forwarding scheme is discussed.

B.1 Packet and Table Management

There are three types of control packets in ODMRP (JOIN QUERY, JOIN REPLY, and ROUTE ERROR). When a JOIN QUERY packet arrives at the router, the content of the packet is cached into temporary route table (`tr_table`) and the timer for the entry is started. If the router does not receive a corresponding JOIN REPLY in time, the timer expires and the cached entry is removed. If a JOIN REPLY which has a corresponding entry in the `tr_table` arrives before the timeout, the user level route table (`route_table`) is searched to find the <source, destination> pair that matches the `tr_table` entry. If such a pair is found, the soft state timer for the entry is reset and the router waits for the next event. If the pair can not be found in the `route_table`, a new entry is created and inserted into the table. The `route_table` is periodically checked for timer expiration and expired entries are removed. The trigger for the update of the kernel level route table (`kr_table`) is activated whenever an entry is inserted or deleted.

C. ODMRP Agent for Nodes with Fixed routes

Operating systems such as Microsoft Windows 95/98 and NT do not allow dynamic reconfiguration of the network routes. ODMRP Agent(ODA) was created to allow forwarding to the hosts with a static route. ODA operates on a Linux host serving as a gate way to ad hoc network for static-route node. Currently, ODA serves only the designated host and the host which employs ODA service must remain within its radio range. ODA performs routing tasks such as sending and receiving the JOIN QUERY and the JOIN REPLY, and updating the route table in behalf of the static-route node. Unicast traffic of static-route nodes can be forwarded through ODMRP ad hoc network with ODA. We experimented with existing Windows and Linux applications over the multihop testbed using ODA. The experiences and the results from the experiments are discussed in Section IV.

D. ODMRP Route Refresh Timer

For the ODMRP soft state timer value, we selected 1 second for the route refresh interval.

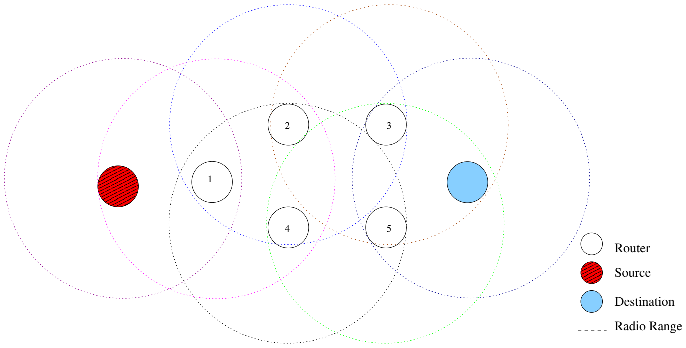


Fig. 2. Multihop testbed topology in a static network.

TABLE I

UNICAST BANDWIDTH DISTRIBUTION IN A STATIC WIRELESS NETWORK.

	Value	% of bandwidth	% of packet loss
Avg. session length	65.64 sec	N/A	N/A
Control packet O/H	0 kb/s	0 %	0 %
Throughput	311.70 kb/s	100 %	28.75 %

TABLE II

UNICAST BANDWIDTH DISTRIBUTION IN A STATIC ODMRP NETWORK.

	Value	% of bandwidth	% of packet loss
Avg. session length	66.76 sec	N/A	N/A
Control packet O/H	1.44 kb/s	0.47 %	28 %
Throughput	304.03 kb/s	99.5 %	29.32 %

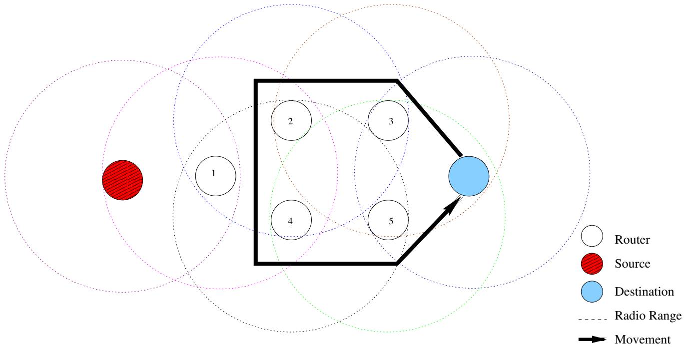


Fig. 3. Multihop testbed topology with end node mobility.

IV. PERFORMANCE EVALUATION

We created a testbed consisting of seven nodes. The bandwidth utilizations of static-routing and ODMRP are studied in this section.

A. *Efficiency Evaluation of Static Multihop Channel*

A static multihop wireless network was constructed in a topology shown in Figure 2. The figure depicts the conceptual view of the building where the experiments were conducted. The router nodes were placed in each corner of the building and had line-of-sight accesses to two other routers. The walls of the building prevented the radio contact and the routers had accesses to one another only when the transceivers were in line-of-sight. A UDP packet transfer program described in the previous section was used to send 2307 packets of size 1556 bytes from the source node to the destination node. The results are presented in Table I. The packet loss rate and ODMRP control overhead were measured to record the channel efficiency in static networks with no mobility. Even in a static network, the multihop channel suffers from large packet losses. Packets are lost because of the channel contention caused by the intermediate nodes competing to transmit, buffer overflow, channel noise, and packet collision.

B. *ODMRP Performance Evaluation*

B.1 ODMRP Performance in a Static Network

The initial experiment was conducted to investigate the performance of ODMRP in a non-mobile environment. We used the topology shown in Figure 2 to make performance comparison with the network running static routing. The same UDP packet transfer procedure was used. The efficiency of the channel was evaluated in Table II. The result differs from that of the static network shown in Table I because of the control packet overhead and the packet loss caused by the route updates. The routes change frequently even when there is no mobility among the nodes. The JOIN QUERY packets arrive at the destination node through several alternate paths and the first arrived packet invokes a route update in the reverse path. The condition of the radio channel changes because of the ambient noise. Even when the network has no mobility, the optimal route may differ for each route update period due to the change in the channel condition. Once the packet transfer starts, UDP packets dominate the channel usage and often disrupts the route discovery sequence. The control packet has to contend for the channel with data packets and in the worst case, as little as one fourth of JOIN QUERY packets is forwarded all the way to the destination. This small JOIN QUERY delivery rate makes routes to change less frequently once the data transmission begins. Since there is no mobility in this experiment, low route change rate gives better performances as less packet losses are caused by route updates.

B.2 ODMRP Performance with End-Node Mobility

In our second experiment, we measured the performance of the ODMRP ad hoc network when the end node was mobile. The basic topology remained the same as the previous sections, but mobility was introduced to the destination node. The destination node was transported following the path indicated in Figure 3 in an approximate speed of 1 meter/second. The UDP packet transfer was performed from the sender to the receiver in the same manner as in the previous experiments.

TABLE III

UNICAST BANDWIDTH DISTRIBUTION IN AN ODMRP NETWORK WITH END-NODE MOBILITY.

	Value	% of bandwidth	% of packet loss
Avg. session length	63.13 sec	N/A	N/A
Control packet O/H	1.53 kb/s	0.49 %	23 %
Throughput	307.72 kb/s	99.5 %	29.32 %

TABLE IV

UNICAST BANDWIDTH DISTRIBUTION IN AN ODMRP NETWORK WITH INTERMEDIATE NODE MOBILITY

	Value	% of bandwidth	% of packet loss
Avg. session length	39.14 sec	N/A	N/A
Control packet O/H	0.367 kb/s	0.12 %	77.06 %
Throughput	238.73 kb/s	99.85 %	67.46 %

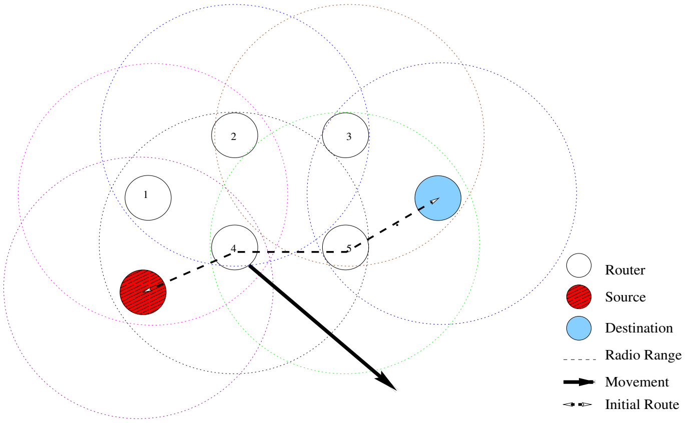


Fig. 4. Multihop testbed topology with intermediate node mobility.

As the destination node moves, the routes are changed. Even though route update is disrupted by the normal flow of the data, it is not crucial in protocol performance. Data packets initially follow the route of $<source\text{-}router1\text{-}router2\text{-}router3\text{-}destination>$. As the destination moves closer to router 3, it enters into the radio range of router 3. The route change may take up to four seconds and by the time the actual route update occurs, the destination node may be within the transmission range of router 2. This delay in route update does not interfere with the flow of data as long as the destination is within the radio range of router 3. In the next successful route discovery sequence, the route is updated to $<source\text{-}router1\text{-}router2\text{-}destination>$. Because the route update has a minimal effect on data transmissions, the channel efficiency of ODMRP with the end node mobility is equivalent to the efficiency shown in no mobility case (shown in Tables III and II, respectively).

B.3 ODMRP Performance with Intermediate Node Mobility

In this experiment, we measured the performance of the ODMRP ad hoc network when an intermediate node was mobile. The router node 4 was abruptly transported out of range of ad hoc network in path $<source\text{-}router4\text{-}router5\text{-}destination>$ depicted in Figure 4. The UDP packet transfer was performed from the source to the destination in the same manner as in the previous sections. As the router moves out of propagation range of its neighbors, one of two following scenarios occurs. If the mobile router was a part of the active route, the data transfer to the destination abruptly halts and packets are lost. Since the source is not immediately aware of the disruption in the data path, it continues to transmit data. Since router 4 is now isolated from the network, there is less contention in the MAC layer. The source is able to send more packets quickly since no forwarding node exists to contend for the channel. There are much larger volume of packets flowing out of sender then there are from router 1 and router 1 cannot grab the channel. In this experiment, ODA is running on router 1 with the sender as it is a client node. Router 1 is the node which initiates the route refresh process so no new route can be discovered until it succeeds in transmission. The route update delay caused by the channel capture effect forces the low channel efficiency noted in Table IV. The re-established channel can slow down the transfer if the destination does not receive the JOIN QUERY packet from the path $<source\text{-}router1\text{-}router2\text{-}router3\text{-}destination>$, and receives the packet on its way back from router 5 in the path $<source\text{-}router1\text{-}router2\text{-}router3\text{-}router5\text{-}destination>$, establishing a non-optimal route (in hop distance) as the new route. The second scenario is the case where the router 4 that moved out of range was not part of the current forwarding path. In this scenario, the transmission continues without delay. The non-optimal route described above can also be built in this scenario. In Table IV, the result for the first scenario is collected and analyzed. The second scenario yields result almost identical to Table II and hence is not shown.

C. *Experiences in using Applications over Ad Hoc Network*

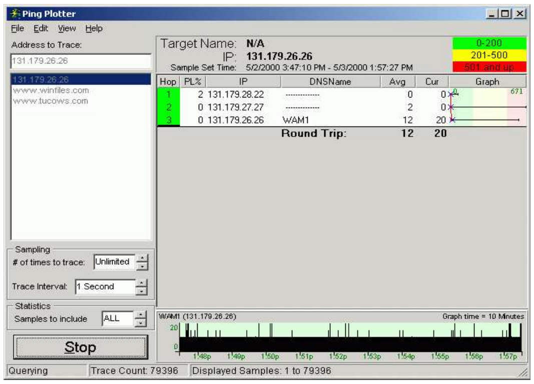
The testbed setup was operated with the existing applications to verify the reliability and robustness of ad hoc routing scheme in day to day operations. Virtual Network Computing (VNC) client-server [17] by AT&T was used to access and remotely control the end nodes. Telnet and FTP sessions were held to test the end-to-end TCP continuity. Live video streams were generated with Microsoft Netmeeting (Figures 5 and 6) to test the feasibility of multimedia application in multihop wireless networks. As expected, the performance of these applications were adequate, but less then spectacular. The applications often had the packet loss problem because of some random environmental interferences (e.g., pedestrians, elevator, cordless phone, etc.) even when there existed a strong radio channel. In a wired environment, packet loss indicates congestion along the route. The applications either compensate for the congestion by sending less packets or changing the data compression

131.179.32.18

Kyle Bae
Wireless Lab

In a call

Fig. 5. Microsoft Netmeeting and operating over the ad hoc testbed.



Address to Trace: 131.179.26.26

131.179.26.26
www.winfiles.com
www.tucows.com

Target Name: N/A
IP: 131.179.26.26
Sample Set Time: 5/2/2000 3:47:10 PM - 5/3/2000 1:57:27 PM

Hop	PL%	IP	DNSName	Avg	Cur	Graph
1	2	131.179.28.22	-------------	0	0	
2	0	131.179.27.27	-------------	2	0	
3	0	131.179.26.26	WAM1	12	20	
			Round Trip:	12	20	

Sampling
# of times to trace: Unlimited
Trace Interval: 1 Second

Statistics
Samples to include: ALL

WAM1 (131.179.26.26) Graph time = 10 Minutes

Querying | Trace Count: 79396 | Displayed Samples: 1 to 79396

Fig. 6. PingPlotter operating over the ad hoc testbed.

scheme, wait for a timeout and rerouting. When applied to the wireless environment, the heuristics built into the applications did not improve the performance. The concept of transparent layering dictates that the application layer should not be aware of layers underneath it. However, to optimize the performance, the application has to be keenly aware of its environment and take an active part in applying appropriate heuristics.

V. CONCLUSION

We presented the unicast operation of ODMRP and our implementation experience in a mobile wireless ad hoc network testbed. ODMRP is capable of applying on-demand route construction for both unicast and multicast sessions. Periodic route refresh is performed to utilize up-to-date route and topology information.

We studied the performance of ODMRP in a real ad hoc network testbed with seven networks hosts. We learned that protocols suffer from packet losses even in static networks because of channel contention, noise, and interference. We introduced various node mobility to the network and presented the throughput results. Our experiments demonstrated ODMRP's ability to dynamically adapt to a mobile routing environment. An end-to-end unicast connection was carried on with a minimal network overhead. We also discussed the need for application's awareness toward its environment to optimize network performances.

References

[1] D.A. Beyer, "Accomplishments of the DARPA Survivable Adaptive Networks SURAN Program," *Proceedings of IEEE MILCOM'90*, Monterey, CA, Oct. 1990, pp. 855-862.

[2] E. Bommaiah, M. Liu, A. McAuley, and R. Talpade, "AMRoute: Adhoc Multicast Routing Protocol," *Internet-Draft*, `draft-talpade-manet-amroute-00.txt`, Aug. 1998, Work in progress.

[3] M.S. Corson and S.G. Batsell, "A Reservation-Based Multicast (RBM) Routing Protocol for Mobile Networks: Initial Route Construction Phase," *ACM/Baltzer Wireless Networks*, vol. 1, no. 4, Dec. 1995, pp. 427-450.

[4] J.J. Garcia-Luna-Aceves and D. Beyer, "SPARROW: Secure Protocols for Adaptive, Robust, Reliable, and Opportunistic WINGs," University of California at Santa Cruz and Rooftop Communications Corporation, `http://www.cse.ucsc.edu/research/ccrg/projects/sparrow.html`.

[5] J.J. Garcia-Luna-Aceves, C.L. Fullmer, E. Madruga, D. Beyer, and T. Frivold, "Wireless Internet Gateways (WINGS)," *Proceedings of IEEE MILCOM'97*, Monterey, CA, Nov. 1997, pp. 1271-1276.

[6] J.J. Garcia-Luna-Aceves and E.L. Madruga, "A Multicast Routing Protocol for Ad-Hoc Networks," *Proceedings of IEEE INFOCOM'99*, New York, NY, Mar. 1999, pp. 784-792.

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

[8] L. Ji and M.S. Corson, "A Lightweight Adaptive Multicast Algorithm," *Proceedings of IEEE GLOBECOM'98*, Sydney, Australia, Nov. 1998, pp. 1036-1042.

[9] L. Ji, M. Ishibashi, and M.S. Corson, "An Approach to Mobile Ad hoc Network Protocol Kernel Design," *Proceedings of IEEE WCNC'99*, New Orleans, LA, Sep. 1999, pp. 1303-1307.

[10] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," In *Mobile Computing*, edited by Tomasz Imielinski and Hank Korth, Chapter 5, Kluwer Academic Publishers, 1996, pp. 153-181.

[11] J. Jubin and J.D. Tornow, "The DARPA Packet Radio Network Protocols," *Proceedings of the IEEE*, vol. 75, no. 1, Jan. 1987, pp. 21-32.

[12] S.-J. Lee, M. Gerla, and C.-C. Chiang, "On-Demand Multicast Routing Protocol," *Proceedings of IEEE WCNC'99*, New Orleans, LA, Sep. 1999, pp. 1298-1302.

[13] S.-J. Lee, W. Su, and M. Gerla, "On-Demand Multicast Routing Protocol (ODMRP) for Ad Hoc Networks," *Internet-Draft*, draft-ietf-manet-odmrp-02.txt, Jan. 2000, Work in progress.

[14] D.A. Maltz, J.Broch, and D.B. Johnson, "Experiences Designing and Building a Multi-Hop Wireless Ad Hoc Network Testbed," *CMU School of Computer Science Technical Report*, CMU-CS-99-116, Mar. 1999.

[15] C.E. Perkins and E.M. Royer, "Ad-Hoc On Demand Distance Vector Routing," *Proceedings of IEEE WMCSA'99*, New Orleans, LA, Feb. 1999, pp. 90-100.

[16] J. Postel, "Internet Control Message Protocol," `Request For Comments 792`, Sep. 1981.

[17] T. Richardson, Q. Stafford-Fraser, K.R. Wood, and A. Hopper, "Virtual Network Computing," *IEEE Internet Computing*, Vol.2 No.1, Jan/Feb 1998, pp. 33-38.

[18] P. Sass, "Communications Networks for the Force XXI Digitized Battlefield," *ACM/Baltzer Mobile Networks and Applications*, vol. 4, no. 3, Aug. 1999, pp. 139-155.

[19] W. van der Moolen, "IEEE 802.11 WaveLAN PC Card User's Guide," Lucent Technologies, Jun. 1998.