On-Demand Multicast in Mobile Wireless Networks

Ching-Chuan Chiang and Mario Gerla
Computer Science Department
University of California, Los Angeles
{ccchiang,gerla}@cs.ucla.edu

Abstract

In this paper we propose an "on demand" multicast routing protocol for a wireless, mobile, multihop network. The proposed scheme has two key features: (a) it is based on the forwarding group concept (i.e., a subset of nodes is in charge of forwarding the multicast packets via scoped flooding) rather than on the conventional multicast tree scheme; (b) it dynamically refreshes the forward group members using a procedure akin to on demand routing (hence the name). "On Demand" multicast is well suited to operate in an On Demand routing environment where routes are selectively computed as needed between communicating node pairs instead of being maintained and updated globally by a routing "infrastructure" (like in Distance Vector or Link State, for example). On Demand Multicast is particularly attractive in mobile, rapidly changing networks, where the traffic overhead caused by routing updates and tree reconfigurations may become prohibitive beyond a critical speed; and, in large networks with sparse traffic requirements, where the traffic, processing and storage overhead of the routing infrastructure solution compromises scalability. Via simulation, we compare On Demand multicast with a traditional tree multicast scheme, DVMRP, and with a version of forwarding group multicast which uses conventional distance vector routing instead On Demand routing. This allows us to assess the penalty of the tree and of the global routing infrastructure as a function of mobility and sparseness.

1. Introduction

Wireline network multicast routing protocols (e.g., DVMRP, PIM, CBT, etc.) are based on two fundamental principles: use of distribution trees for efficient delivery of multicast packets, and; use of a preexisting routing infrastructure for the maintenance of such trees. In ad hoc wireless, mobile networks, however, the validity of these principles is undermined by the broadcast nature of the channel and the continuously changing network connectivity. First, the use of trees in a rapidly reconfiguring environment requires frequent repairs of branches, and has two negative consequences: high channel and processor O/H, and; high risk of packet loss during branch reconfiguration. Secondly, in the conventional routing infrastructure the exchange of routing vectors or link state tables must be made progressively faster as mobility increases, leading to higher overhead. This is because, in addition to periodical updates (i.e., fixed update period), event driven updates are necessary to cope with frequent changes in connectivity. That is, the higher the mobility, the the higher the actual update frequency is. Another problem of global routing is the size of the routing tables which increases linearly with number of nodes and thus limits the scalability to large networks.

To overcome routing limitations, we propose a multicast protocol which embeds its own on-demand routing scheme, thus avoiding channel overhead and increasing scalability. To overcome tree topology limitations, we use the concept of "Forwarding Group", a set of nodes which are responsible for forwarding multicast data. The Forwarding Group infrastructure reduces storage overhead and requires a much looser connectivity among multicast members. It suffices that the mesh topology formed by multicast members and forwarding group nodes be connected (no islands). The reduction of channel and storage overhead and the relaxed connectivity make this protocol more scalable for large networks and more stable for mobile wireless networks. The Forwarding Group multicast protocol was first introduced in [3] using a conventional routing structure (Distance Vector). This paper extends that scheme and makes it scalable by incorporating on-demand routing in it.

Section 2 introduces the forwarding group multicast protocol using underlying routing information which is maintained by the update of global routing tables [2]. Section 3 describes the On-Demand multicast protocol in details. Section 4 reviews the network infrastructure. Section 5 addresses the simulation environment. Section 6 details the performance evaluation. Section 7 concludes the paper.

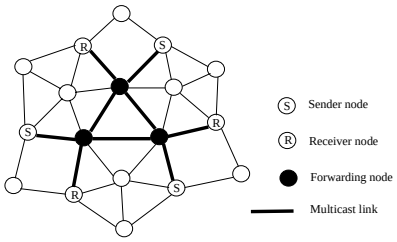

Figure 1. An example of FGMP

Table 1. Format of join_request packet

Mcast Group id	Receiver member Id	Sequence #	TTL

2. Forwarding Group Multicast Protocol (FGMP)

The FGMP scheme without on-demand routing (first introduced in [3]) is reviewed here for completeness. FGMP keeps track not of links but of groups of nodes which participate in multicast packet forwarding. To each multicast group G is associated a forwarding group, FG. Any node in FG is in charge of forwarding multicast packets of G. That is, when a forwarding node (a node in FG) receives a multicast packet, it will broadcast this packet if it is not a duplicate. All neighbors can hear it, but only neighbors that are in FG will first determine if it is a duplicate (based on a historical list of source sequence numbers) and then broadcast it in turn. Figure 1 shows an example of a multicast group containing three senders and three receivers. Three forwarding nodes take the responsibility to forward multicast packets. This scheme can be viewed as "limited scope" flooding. That is, flooding is contained within a properly selected forwarding set.

Only flag timer and historical source sequence number list are needed for each forwarding node. When the forwarding flag is set (as described in following subsections), each node in FG forwards non-duplicate data packets belonging to G until the timer expires. Storage overhead, a major problem in traditional multicast protocols, is minimal, thus improving the scalability. Timer is refreshed by the forwarding group updating protocol. Stale forwarding nodes are deleted from FG after timeout.

A key component of FGMP is the election and maintenance of the set FG of forwarding nodes. The size of FG should be as small as possible to reduce wireless channel overhead. Yet, the forwarding path from senders to receivers should be as short as possible to achieve high throughput.

Table 2. Format of member table at the sender members

Mcast Group id	
Refresh Timer	
receiver member id	timer
⋮	⋮

Table 3. Format of forwarding table FW

Mcast Group id	
receiver member id	next hop
⋮	⋮

2.1. FG Maintenance

In order to select the set FG, we require each source to periodically transmit control packets to all member destinations. In the process, all nodes along the shortest path from source to destination are "inducted" into FG. This procedure presumes that each source knows the member destinations. This is obtained via receiver advertising. Namely, each **receiver** periodically and globally floods its member information (join request) formatted as in table 1. TTL limits the scope of flooding. Each **sender** maintains a member table as shown in table 2. When a sender receives the join request from receiver members, it updates its member table. Expired receiver entries will be deleted from the member table. Non-sender nodes simply forward the request packet. After updating the member table, the sender creates from it the forwarding table FW shown in table 3. Next hop on the shortest path to the receiver is obtained from preexisting routing tables. The forwarding table FW is broadcast by the sender to all neighbors; only neighbors listed in the next hop list (next hop neighbors) accept this forwarding table (although all neighbors can hear it). Each neighbor in the next hop list creates its forwarding table by extracting the entries where it is the next hop neighbor and again using the preexisting routing table to find the next hops, etc. After the FW table is built, it is then broadcast again to neighbors and so on, until all receivers are reached. Note that FW is discarded after use. The member table on the other hand is permanent. The forwarding table FW propagation mechanism essentially "activates" all the nodes on the source tree rooted at the sender. These nodes become part of the FG. At each step, nodes on the next hop neighbor list enable the forwarding flag and refresh the forwarding timer. Soft state dynamic reconfiguration provides the ability to adapt to a changing topology.

Figure 2 shows an example of multicasting forwarding tables. Node 12 is the sender. Five nodes are forward-

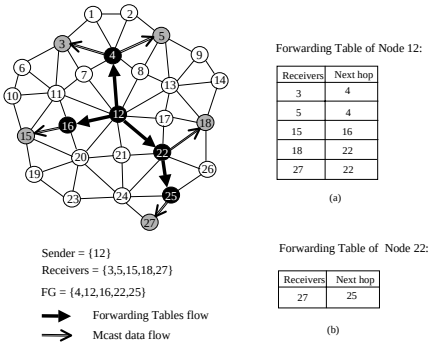

Figure 2. Example of Forwarding tables for FGMP-RA

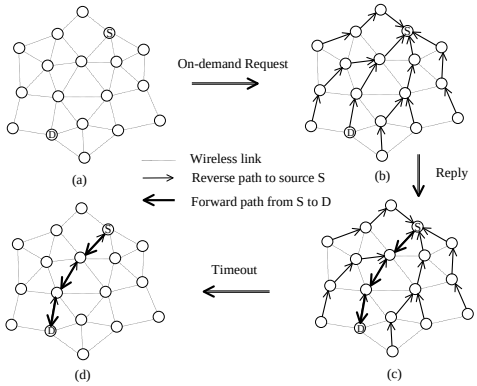

Figure 3. An example of On-Demand Routing

ing nodes, $FG = \{4, 12, 16, 22, 25\}$, because they are in the next hop list. Only sender and internal nodes, in our case 12 and node 22, need to create a forwarding table (figure 2(a),(b)) and broadcast it. Forwarding nodes 4, 16, and 25 do not need to create their forwarding tables since they are "leaves".

Another way to advertise the membership is to let the senders flood sender information. Sender advertising is more efficient than receiver advertising if the number of senders is less than the number of receivers. Most multicast applications belong to this category. Like in receiver advertising, senders periodically flood the sender information. Receivers will collect senders' status, then periodically broadcast "joining tables" to create and maintain the forwarding group FG.

Member table and forwarding table size pose a scaling limitation when the multicast group grows to hundreds or even thousands of nodes. A possible solution (which we are currently exploring) is to dynamically (and randomly) elect a small set of "core" nodes which lie on the path between senders and receivers. These core nodes advertise (at a fairly low frequency) their presence, i.e., their ID, to all nodes. Senders and receivers alike send short join messages to each of the core nodes, activating the FG flag in all the nodes encountered along the path. The scheme scales well in both storage and channel O/H. It does not however guarantee shortest paths between all senders and receivers. It also introduces the additional complexity of core node elections. We are evaluating some of these tradeoff in our current research.

3. On-Demand FGMP

In the previous FGMP version, each intermediate node needs routing information to forward the table FW. This routing information can be created using an on-demand approach rather than global routing tables in order to reduce the channel overhead, improve the delivery efficiency and more generally enhance scaling and mobility support. The routing table entry format of on-demand routing is presented in table 4. Entries with expired timer are removed from the routing table. There are two phases to establish the next hop information for on-demand FGMP. One is "Request phase" which is issued by multicast members to create the routing information (next hop) for forwarding/joining table. The other is "Recovery phase" which is used to request the next hop information if it is not available while the forwarding node is sending the forwarding/joining table. Since these two phases are found also in most on-demand routing algorithms, we briefly review on-demand routing in the next section.

3.1. On-Demand Routing Review

Several routing protocols have been recently proposed for ad hoc networks [5, 15, 10]. While almost any ad-hoc routing scheme could be used for our purposes, for reasons of efficiency, we have used the on-demand routing proposed by [14] and have incorporated it in FGMP. The basic principle of on-demand routing as described in [14], is to keep only routing information for active source/destination pairs. When a source S wants to send data to a destination D and does not find D in its routing table, it broadcasts a REQUEST for route to D. REQUEST messages are flooded until reaching nodes which have the routing information to D. A REPLY message is sent back to S by each such node. Routes are computed using the well known "backward learning" principle, which has been applied, among others, in LAN interconnection via spanning tree bridges. Namely, REQUEST messages create reverse path routing entries to S. REPLY messages create forward path routing entries from S to D. After one or more routes to the destination are established, the source can send data by se-

lecting an appropriate route. The major difference between distance vector and on-demand route is that in the latter routing tables are dynamically created and updated according to the traffic demand instead of topology changes, and thus reduce the storage overhead for routing tables in sparse traffic situations. This storage economy is very important for mobile nodes and makes it more scalable for large networks. Figure 3 shows an example of on-demand routing: 3(b) displays the reverse paths from all nodes to source S after flooding the REQUESTs; 3(c) presents the forward path from S to destination D (in this example, only D knows the route to itself, and therefore can return the REPLY, and; 3(d) shows the final situation when stale paths are removed.

Table 4. Format of On-Demand routing table entry

Destination	Next hop	Hop Count	Sequence #	Timer

Table 5. Format of on-demand join_request packet

Mcast Group Id	Receiver member Id	Sequence #	TTL	Sending node id	Hops

3.2. On-Demand Request phase

In on-demand FGMP, the advertising messages (join requests) issued by receiver members are used to create the path information from senders to receivers. To this end, some additional attributes are needed in the join request as shown in table 5. "Sending Id" is the node id which is currently sending the request; "Hops" is the hop count traversed by the request message. When a node n receives a join request message, which contains receiver member id R_i, from neighbor m (the sending node), node n examines its routing table. If there is no routing entry for R_i, a new entry containing fields $< R_i$(destination),m(next hop),hops,sequence number,timer$>$ is added into the routing table. Otherwise, the entry for R_i is refreshed if the request is more recent. A more recent request means either larger sequence number or smaller hop count and same sequence number. Join requests are flooded to the network scoped by TTL. By the time the sender members receive the join requests, path information from senders to receiver members has been created/updated and thus the forwarding table can be readily delivered to the set of receiver members, thus updating the forwarding group. As a difference from unicast on-demand routing, note that senders do not need to send back any reply message to receiver members since the paths to receivers are already established after the first phase and the paths from receivers to senders are not required.

Similar process is applicable to the sender advertising version FGMP-SA, in which case sender members flood sender information to create/update paths from receiver members to sender members, and; joining table is delivered to sender members along the paths. Note that the FGMP-SA route discovery is essentially the same as the unicast on-demand route discovery if the multicast group ID field in Tables is replaced by the destination ID.

3.3. Next hop Re-Patch

On-demand multicast uses member advertising messages to create/update routing path information. Recall, however, that the original purpose of advertising messages was to update membership status rather than to maintain the routing status. In conventional FGMP, every node just forwards the advertising messages (flooding) without leaving any trace in the intermediate nodes since routing information is maintained by RTU. In the on-demand case, in addition to forwarding the advertising messages, each node updates its routing tables. On-demand routing does not increase the frequency of advertising. It only marginally increases the channel overhead (message size is increased by only two fields; yet much smaller than routing table size). On demand routing produces all the routing information needed for FGMP, i.e., the next hop information to create the forwarding/joining table which in turn maintains the forwarding group (FG). However, the path information created/updated by the advertising messages becomes quickly obsolete in high mobility. That is, when a forwarding node receives a forwarding/joining table and wants to create a new forwarding/joining table to forward according to its routing table, it may discover that it does not have the next hop information for some destinations. To overcome this problem, when the next hop to member m_k is not existent, a path request, which is an on-demand unicast request, is issued to open a new path to member m_k. This path request need not be flooded to the entire network like a generic on-demand unicast request. Since path information to m_k has already been created in the network via m_k's advertising messages, two or three hops probes (TTL = 2 or 3) will be enough to reach nodes which have routing information for m_k, keeping latency low. After replies come back, the next hop information is restored and the forwarding/joining table is delivered to the new path.

4. Multihop Network Infrastructure

The infrastructure used in our experiments is a clustered multihop infrastructure [4, 9]. In our distributed clustering algorithm, nodes are elected as clusterheads based on preferential criteria (e.g., lowest ID number, etc.). Neighbors are discovered with periodic Hello messages. All nodes within transmission range of a clusterhead belong to the same cluster, and can communicate directly with a clusterhead and (possibly) with each other. Nodes belonging to more than one cluster are called gateways. Gateways support communications between adjacent clusters [8, 9].

Within each cluster, the MAC protocol provides for efficient transfer of packets between neighbors. For our experiments we have selected polling. Namely, the clusterhead polls the nodes to allocate the channel. Polling was chosen here for several reasons. First, polling is consistent with the IEEE 802.11 standard (Point Coordination Function) [6]. Secondly, polling gives priority to the clusterhead, which is desirable since only gateways and clusterheads can be FG members, and thus routes are forced to go through clusterheads.

For the sake of simplicity we assume that nodes (and in particular gateways) can receive on multiple codes simultaneously (e.g., using multiple receivers). This property does not enhance communications within a cluster, since all wireless nodes are tuned to the same code anyway. It does, however, permit conflict free communications with the gateways, and in particular conflict free multicast from clusterhead to gateways. Without the multiple code reception, the gateway must tune on different codes (of the adjacent clusters) and can receive correctly only if it is tuned to the transmitting clusterhead code. An example is offered in [11].

Nodes have a finite buffer. Packets are dropped when buffers overflow, or when there is no route to the intended destination. The latter occurs when the topology is disconnected or the route is not available. Packet drop, channel interference, noise, fading and mobility lead to packet loss, thus making the multicast protocol just described unreliable. End to end reliable delivery can be restored with other means such as Scalable Reliable Multicast [7, 13]. SRM works at the transport application level and can be built directly on top of our multicast scheme. Another reliable wireless multicast scheme is reported in [13]. That scheme works at the network layer and exploits our cluster infrastructure (but not our multicast protocol).

5. Simulation Environment

A multihop, mobile wireless network simulator was developed using the parallel simulation language Maisie/PARSEC [1, 12]. The simulator is very detailed in that it models all the control message exchanges at the cluster, MAC (e.g., polling) and network layer (Distance Vector routing tables and join/quit m-cast messages). Thus, the simulator enables us to monitor the traffic O/H of the protocols. The network consists of 100 mobile hosts roaming randomly at a predefined average speed in a 1000x1000 meter square. At each time tick, a random direction and step size are chosen. A reflecting boundary model is assumed. Radio transmission range is 120 meters. Free space propagation channel is assumed unless otherwise specified. Data rate is 2 Mb/s. Packet length is 10 kbit for data, 2 kbit for routing tables, and 500 bits for MAC control packets and multicast control tables. Thus, transmission time is 5 ms for data packet, 1 ms for routing table, and 0.25 ms for control packet. Buffer size at each node is 10 packets.

Mobility (km/hr)	Soft state parameters (time interval in ms)				
	Distance Vector & On-Demand				On-Demand
	Member Advertising				
	refresh	timeout	FW Tab refresh	FG timeout	Route Table Entry timeout
0.02	400	960	200	560	960
0.70	400	960	200	560	960
1.41	400	960	200	560	960
2.81	400	960	200	560	960
5.62	400	960	200	560	960
11.25	400	960	200	560	960
22.50	400	960	160	480	960
30.00	400	960	120	400	960
45.00	400	960	80	320	960
60.00	400	960	60	280	960
90.00	400	960	40	240	960

Table 6. Soft state parameters

Routing tables and control messages have higher priority over data. Channel overhead (e.g., code acquisition time, preamble, etc.) is factored into packet length. Routing tables are updated every second. This low update rate is consistent with typical wired network operation and is adequate for a static network. As node mobility increases, however, the topology starts changing rather rapidly. In order to maintain accurate routing information, changes in local link status and new routing tables from neighbors trigger new updates. Other soft state parameters are listed in table 6.

Two multicast membership configurations are evaluated. "One-to-many" multicast consists of one sender and 9 receiver members. "Many-to-many" multicast has 10 members each of which is both sender and receiver.

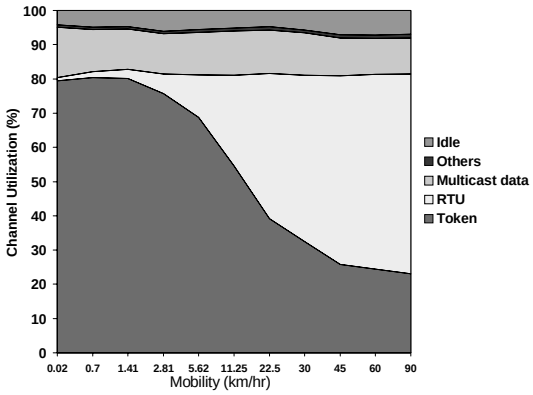

Figure 4. Channel Utilization of FGMP-SA (Distance Vector) [One-to-Many]

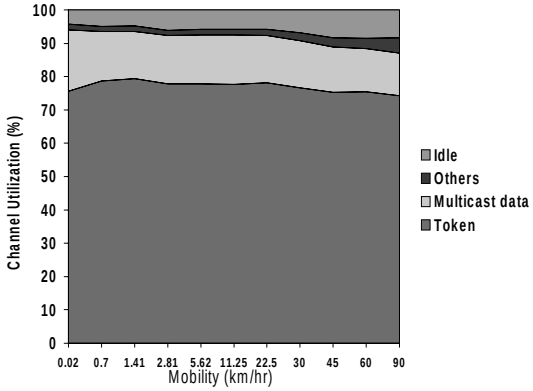

Figure 5. Channel Utilization of FGMP-SA (On-Demand) [One-to-Many]

One-to-many multicast may correspond to a broadcast service such as news reports, while many-to-many may correspond to workgroup collaboration. Packet interarrival times are exponentially distributed with mean $1/\lambda$, where $1/\lambda_{one-to-many} = 25ms$ and $1/\lambda_{many-to-many} = 250ms$. In addition to multicast, there is light background uniform unicast load (datagram) originating from each node at the rate of $1/\lambda = 5sec$. The total received load (for either one-to-many or many to many) is 3.8 Mbps – a very high load considering channel capacity (2Mbps) and multihop penalty (reduction =.25). Space diversity across clusters saves the day! On average, 10 to 15 clusters are formed.

Total simulation time for each experiment is 4x10^6 simulation ticks. One simulation tick corresponds to 50 μs. Thus, each run represents 200 seconds of simulated time, or 76,000 received packets, given our traffic assumptions.

6. Performance Evaluation

In this section we present the simulation results in which the performance of on-demand FGMP multicast is extensively evaluated and compared with FGMP (Distance Vector). A limited comparison with DVMRP and shared tree is also provided. Channel utilization, control messages and storage overhead are first evaluated to explore efficiency of on-demand routing. Multicast performance is then measured to prove the efficiency of on-demand multicast. One-to-many multicast using FGMP-SA is evaluated from section 6.1 to section 6.5. Section 6.6 and section 6.7 explores other configurations and loads.

6.1. Channel Utilization

Total network capacity is defined as $C = S * B$ (bits), where S is the average number of clusters and B is the wireless bandwidth (bits/sec). The channel overhead of RTU is then given by:

$$CU_{RTU} = \frac{\text{(Total number of RTU) * (Routing Table Size(bits))}}{C * T},$$

where T is the total simulation time. Figure 4 shows the channel utilization by message type for FGMP-SA using Distance Vector routing; Figure 5 shows the results of FGMP-SA using on-demand routing. Token utilization is the fraction of bandwidth available for data (i.e., token). As expected, on-demand routing eliminates channel overhead due to routing table updates and increases the token utilization for multicast traffic. From these results, it is quite obvious that beyond, say, 10 km/hr, for this particular network configuration the overhead introduced by the routing updates makes on-demand routing much more attractive then Distance Vector routing.

6.2. Control Message Overhead

On-demand multicast creates/maintains routing information by using on-demand requests/replies and hello messages. The on-demand request includes member advertising messages, next hop re-patch requests, and unicast requests. Figure 7 shows the overhead (in total bits) of all control messages for on-demand FGMP-SA. Compared with figure 6 which is using Distance Vector, the overhead of on-demand control messages is much less than that of routing table updates (RTU). Note that the RTU message overhead in figure 6 was divided by 40 so as to fit it in the graph.

6.3. Storage Overhead

Another benefit of on-demand routing is the reduction of routing table size. The average number of RTEs per node is

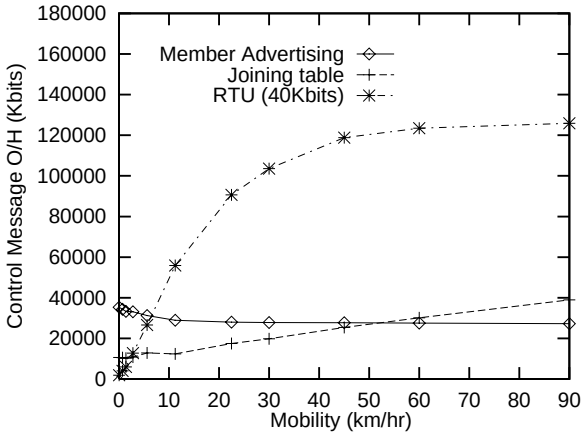

Figure 6. Control Messages of FGMP-SA (Distance Vector) [One-to-Many]

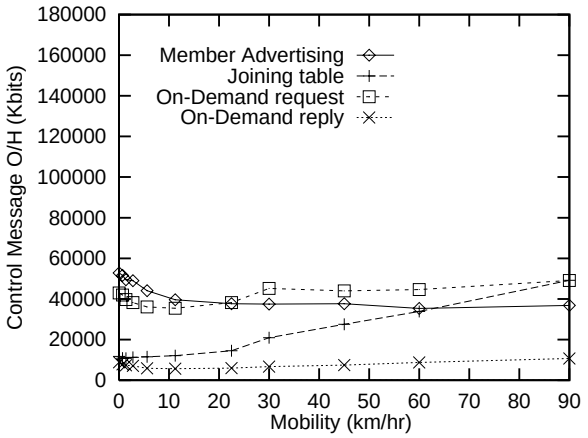

Figure 7. Control Messages of FGMP-SA (On-Demand) [One-to-Many]

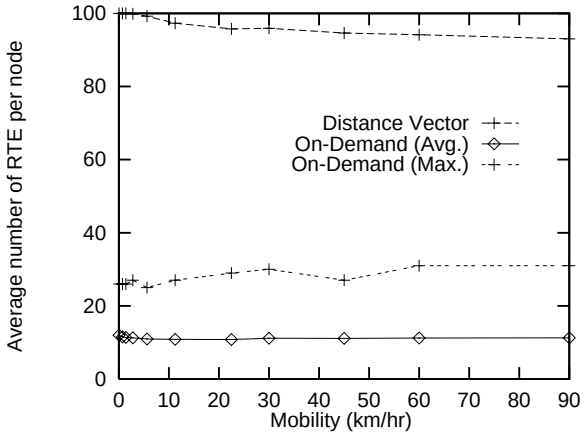

Figure 8. Average number of RTE [One-to-Many]

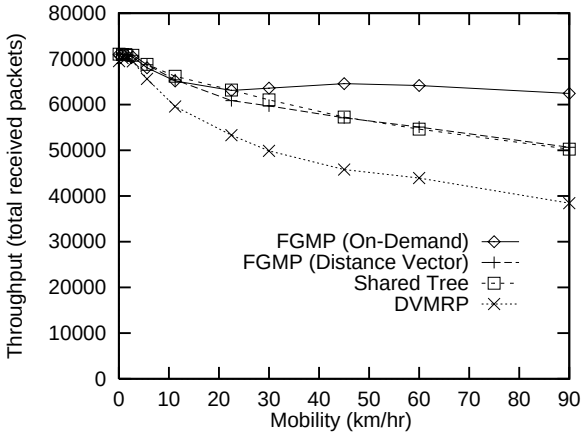

Figure 9. Throughput [One-to-Many]

shown in figure 8. We note that on-demand only needs approximately 12 RTEs while Distance Vector maintains between 95 and 100 RTEs (depending of speed) for a 100 node infrastructure. In figure 8 we also show the maximal number of RTE used by on-demand routing during the experiment, which is also much less than 100 used by Distance Vector.

6.4. Throughput

To evaluate multicast performance , we measure the "throughput" at the receivers. The "throughput" is defined as total multicast packets received at all receiver members excluding duplicated packets. As previously explained, not all packets will be delivered because of buffer overflow and dropping. More precisely, throughput performance is affected by two factors: (a) the temporary loss of a multicast route because of mobility, and; (b) the line O/H caused by control messages. Line O/H indirectly causes congestion and buffer overflow. Thus, net throughput is a good cumulative measure of multicast performance (i.e., the ability to withstand mobility and to reduce O/H).

Figure 9 compares the throughput of FGMP(Distance Vector), On-Demand FGMP, shared tree and DVMRP. At low speed, performance is near maximum and is comparable for all. At high speed, on-demand multicast reaches by far the highest throughput because of the reduction of routing table updates. Shared tree does as well as FGMP-BF. DVMRP throughput drops rapidly as speed increases. Figure 11 evaluates throughput in a many-to-many cast scenario. The results are comparable to the one-to-many case.

6.5. Delay

Average delay is measured at each receiver. Each multicast packet carries the time stamped by the sender. Total delay is accumulated and averaged over the entire simulation

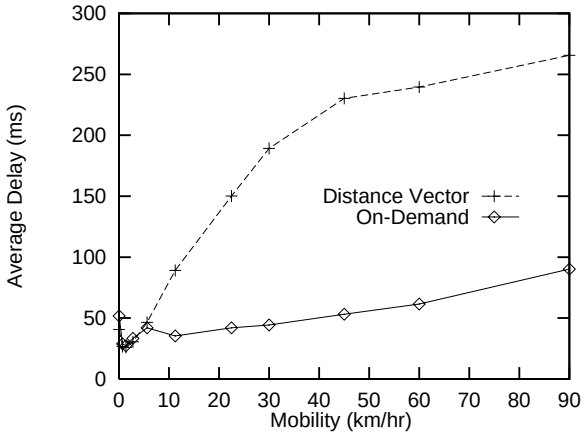

Figure 10. Average Delay [One-to-Many]

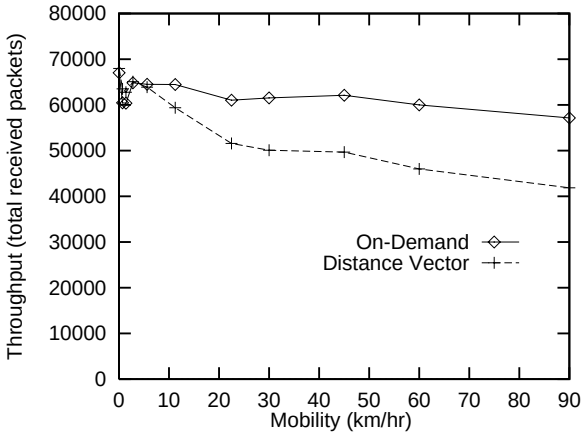

Figure 11. Throughput [Many-to-Many]

run. The delay includes transmission and queueing delay. From figure 10, on-demand multicast has much smaller delay at high mobility due to the channel efficiency (no queueing behind RTUs) and token efficiency (without competition of neighbors). This result is somewhat surprising since usually, on-demand routing for unicast has longer delay due to the latency of request/reply. However, in FGMP, most on-demand requests are performed by member advertising message and no replies are required, thus avoiding the latency.

6.6. Light Load Experiment

A set of light load experiments (with offered load reduced by a factor of 20) was also produced and the results are shown in figure 12. There is no packet loss due to buffer overflow. There is no throughput gain for on-demand multicast either since packet dropping due to the topology changes is the same for both schemes. The average delay, however, is still much better for on-demand than Distance Vector.

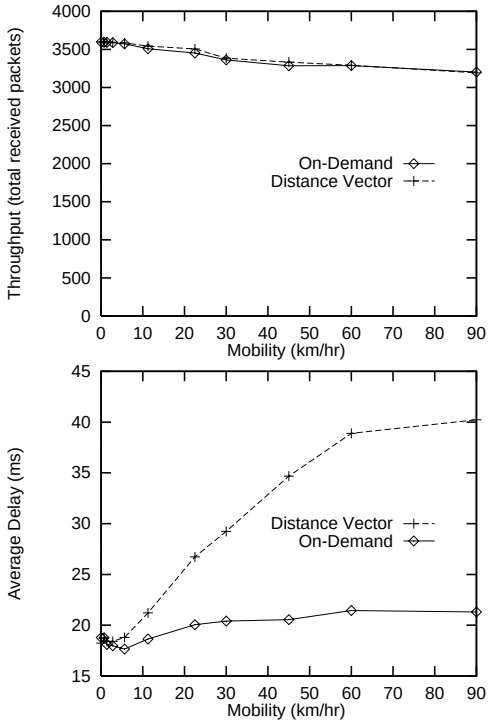

Figure 12. Performance Evaluation [One-to-Many: light load]

6.7. Scalability

The final sets of experiments explores the scalability. We increase member size to 50 (1 sender and 49 receivers). That is, one half of nodes are multicast members. The results are shown in figure 13. In spite of the fact that on-demand is at a disadvantage with dense membership, it still achieves much better performance than Distance Vector under all measures. The overhead of RTU in Distance Vector greatly affects the throughput and average delay even for dense membership, thus reducing the scalability.

7. Conclusion

We have proposed a novel approach to wireless, ad hoc multicasting which is based on forwarding groups and on-demand routing. Preliminary simulation results show that the proposed scheme is much more robust to mobility than the forwarding group version based on a global routing structure (Distance Vector). It also outperforms conventional tree based multicast schemes such as DVMRP and shared tree. The reasons for this superior performance

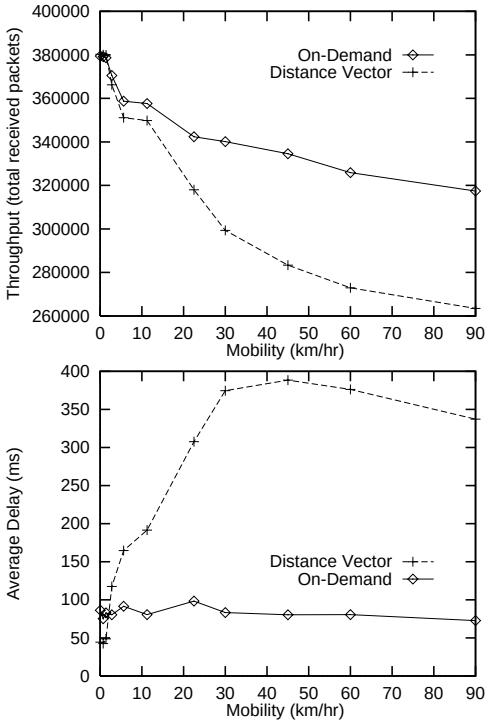

Figure 13. Performance Evaluation [One-to-Many:Dense Mode(1 sender, 49 receivers)]

must be sought in lower control overhead and in more agile recovery from path breakage. Storage scalability is also greatly enhanced by on-demand routing, especially in large networks with sparse membership. Several extensions to the basic scheme are now under study, including solutions which scale to large membership size and methods for the integration of the multicast on-demand route search with existing on-demand unicast routing algorithms.

References

[1] R. Bagrodia and W. Liao. Maisie: A language for the design of efficient discrete-event simulations. *IEEE Transactions on Software Engineering*, 20(4):225–238, 1994.

[2] D. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, 1992.

[3] C.-C. Chiang, M. Gerla, and L. Zhang. Forwarding group multicast protocol (FGMP) for multihop, mobile wireless networks. *Special Issue of Cluster Computing: the Journal of Networks, Software Tools and Applications*, 1(2), 1998.

[4] C.-C. Chiang, H.-K. Wu, W. Liu, and M. Gerla. Routing in clustered multihop, mobile wireless networks with fading channel. In *The IEEE Singapore International Conference on Networks*, pages 197–211, 1997.

[5] M. S. Corson and A. Ephremides. A distributed routing algorithm for mobile wireless networks. *ACM/Baltzer Journal of Wireless Networks*, 1(1):61–81, February 1995.

[6] B. P. Crow, I. Widjaja, J. G. Kim, and P. Sakai. Investigation of the ieee 802.11 medium access control (MAC). In *IEEE INFOCOM*, 1997.

[7] S. Floyd, V. Jacobson, S. McCanne, C.-G. Liu, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. In *ACM SIGCOMM*, pages 342–356, 1995.

[8] M. Gerla and C.-C. Chiang. Shared tree multicast with RP relocation in mobile wireless networks. Technical report, UCLA-CSD, Jan. 1998.

[9] M. Gerla and J. T.-C. Tsai. Multicluster, mobile, multimedia radio network. *ACM/Baltzer Journal of Wireless Networks*, 1(3):255–265, 1995.

[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, Kluwer Academic Publishers*, pages 153–181, 1996.

[11] C. R. Lin and M. Gerla. Maca/pr: An asynchronous multimedia multihop wireless network. In *IEEE INFOCOM*, 1997.

[12] W. W. Liu, C.-C. Chiang, H.-K. Wu, V. Jha, M. Gerla, and R. Bagrodia. Parallel simulation environment for mobile wireless networks. In *1996 Winter Simulation Conference Proceedings (WSC'96)*, pages 650–612, 1996.

[13] E. Pagani and G. P. Rossi. Reliable broadcast in mobile multihop packets networks. In *ACM MOBICOM*, pages 34–42, 1997.

[14] C. E. Perkins. *Ad-hoc On Demand Distance (AODV) Vector Routing*. IETF, Internet Draft : draft-ietf-manet-aodv-00.txt, 1997.

[15] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *ACM SIGCOMM*, pages 234–244, 1994.