The Effects of Mobility on Multicast Routing in Mobile Ad Hoc Networks

Manoj Pandey and Daniel Zappala
Computer and Information Science, University of Oregon, Eugene, Oregon 97403-1202
manoj@cs.uoregon.edu, zappala@cs.uoregon.edu

***Abstract*— Performance evaluation of ad hoc routing protocols typically depends on simulation, since the deployment of ad hoc networks is still relatively rare. However, past evaluations of multicast routing protocols have utilized a single, simple mobility model, and thus do not capture the variety of mobility patterns likely to be exhibited by ad hoc applications. In this paper, we explore the impact of several different mobility models on multicast routing performance, using techniques that have been demonstrated to be effective for unicast routing protocols. We demonstrate that three key mobility metrics help to explain the performance variations that we observe for flooding, ODMRP, and ADMR. In addition, we study a high density, high traffic scenario and find that ODMRP copes with this extreme situation much better than ADMR.**

I. INTRODUCTION

Mobile ad hoc networks have numerous practical applications, such as emergency and relief operations, military exercises and combat situations, and conference or classroom meetings. Each of these applications can potentially involve different *mobility patterns*, with movement dependent on interactions among participants and the environment. For example, in a search-and-rescue operation, individuals may fan out to search a wide area, each moving independently in a confined area. In a battlefield, however, the movement of soldiers is heavily influenced by the movements of their commander. Similarly, the environment can influence movement, such as cars moving on a freeway or patrons in an exhibit hall moving among a selected group of displays.

In this paper, we study the effects of mobility patterns on multicast routing performance. We focus on simulation-based evaluation because simulation plays an important role in prototyping and evaluating routing protocols, particularly since ad hoc networks are still an emerging area and deployment is relatively rare. While communication patterns are likely to also have an impact on routing performance, in this study we want to isolate the effects of mobility, so we use long-lived, constant bit-rate traffic streams among randomly-chosen multicast groups.

Motivation for this research originates from the fact that previous evaluations of multicast routing protocols have utilized a single, simple mobility model, and thus do not capture the variety of mobility patterns likely to be exhibited by ad hoc applications. The most comprehensive performance comparison of ad hoc multicast routing protocols uses the *Uniform* model, in which nodes move in a random direction with constant velocity and then "bounce" off the boundary of the simulated field [17]. Most other studies [15], [12], [22], [13] use the *Random Waypoint* model, in which each node moves to a random destination, pauses for a specified period, and then chooses a new destination. While this model may approximate the movement of a search and rescue operation, it fails to depict loads that can be exerted in a myriad of other application-specific patterns. Moreover, a recent study has been shown that the average speed of a node using Random Waypoint decreases over time, meaning that results obtained from this model can get unreliable as the simulation advances [26].

Reflecting this need for a wider range of mobility models, some researchers have created new mobility models and evaluated unicast routing performance under these models. Johansson et al. [14] study a range of scenario-based mobility models, such as a conference, the floor of a stock exchange, and a disaster recovery scene. Each scenario defines specific movement patterns, speed, and communication patterns for each of the nodes, as well as obstacles in the field. The authors find that the relative speed of the nodes influences protocol performance and that reactive protocols (such as AODV [20] and DSR [15]) perform better than a proactive protocol (DSDV [19]). Hong et al. [7] create a model that generalizes group-based movement for various applications and likewise conclude that mobility patterns can affect connectivity and routing performance. Most recently, several mobility frameworks have been developed to provide insight into unicast routing performance. The *Mobility Vector* model [8] can be used to generate various movement patterns based on velocity and acceleration vectors. The IMPORTANT framework [1] characterizes movement based on spatial dependence, relative speed, and other factors and illustrates how these metrics impact unicast routing performance.

Our interest lies in examining the impact of application-specific mobility models on multicast routing performance. For our study we use a set of mobility models that represent a range of application-based mobility patterns, similar to those described in the IMPORTANT framework. We introduce two new metrics – reachability and node density – to help differentiate between these models. We then examine the performance of flooding [6], ODMRP [16], and ADMR [12]

under each of the mobility models. Our results show that mobility patterns do affect multicast routing performance and that the observed performance variations can be explained by three key mobility metrics – number of link changes, reachability, and node density. We also investigate the special case of high density and high traffic rate, with our results indicating that ODMRP copes with this extreme situation much better than ADMR.

II. RELATED WORK

Simulation based-evaluation of ad hoc routing protocols depends on mobility models that characterize the movement of mobile users [3]. Routing protocols typically establish paths over which packets will be sent. Mobility breaks those paths and hence disrupts the communication, exerting a load on routing protocols. The location, duration and frequency of these disruptions varies with the pattern of user movement.

In many mobility models, each node moves independently from the others. Often, the speed and direction are chosen randomly, as with *Brownian Motion* [9], Random Gauss-Markov [18], *Random Waypoint* [15], and *Random Direction* [21]. Several models restrict the direction in which nodes may move. For example, Hu and Johnson use a *Column* model, based on a design suggested by Sanchez [23], in which nodes move with randomly-selected velocities within a column formation [9]. Tian et al. explore graph-based mobility, designed to model the constraints of real-world locations, such as trains connecting cities [24]. In this model, vertices in a graph represent possible destinations and edges represent paths on which nodes can travel. Likewise, Davies uses a *City Section Mobility* model, where several parts of a city are modeled as streets with their speed limits [4]. When a node has to move from one point to another, it selects the shortest path possible with the given street constraints.

Recent work by Jardosh et al. incorporates the use of obstacles in defining mobility patterns [10]. In this work, obstacles are placed in a field and then paths to the obstacles (i.e. doorways into buildings) are computed using a Voronoi Diagram. Nodes choose a destination randomly and then move along the designated paths using a shortest-path computation.

Group-based mobility models introduce dependency among the mobile nodes. Johansson et al. propose a *Disaster Area* scenario, in which individual groups consisting of rescue agents intercommunicate with each other [14]. Hu and Johnson use a pursue model, again suggested by Sanchez, in which nodes follow a group leader by trying to intercept it [9]. Hong et al. develop a *Reference Point Group Mobility* model, which is a generalization of the pursue model [7]. In this model, each node belongs to a group with a logical center, and a node's velocity is defined as the sum of the velocity of the center its own random velocity. By adjusting the movement of the logical center, this model can be used to produce various real-life group-based scenarios, such as disaster management, a convention center, etc.

Finally, several mobility frameworks have been developed to characterize a wide variety of movements. The *Mobility Vector* model [8] uses a pair of vectors to model smooth changes in direction and speed. They show how various mobility scenarios can be generated from this basic model, including location-dependent movement, targeting, and group mobility. The purpose of the IMPORTANT framework [1], on the other hand, is to explore the impact of mobility patterns on unicast routing. Hence, this framework includes a variety of models and then characterizes those models using both mobility metrics and connectivity metrics. The authors show that spatial dependence, relative speed, and link duration are able to differentiate between mobility models and can thus be used to explain variation in routing performance under various scenarios.

Model	**Applications**
Random Waypoint	Wandering in a room
Uniform	Movement on a set of freeways
Manhattan	Movement in an urban area
Exhibition	Visitors to a museum
Battlefield	Soldiers following a commander

TABLE I
MOBILITY MODELS AND THEIR APPLICATION

III. MOBILITY MODELS

In this work, we use a variety of mobility models designed to capture a wide range of mobility patterns for ad hoc applications. We choose models from different classes of motion, including random, path-based, and group-based movements. As a result, we are able to differentiate these models based on a set of mobility and connectivity metrics.

The models we use are listed in Table I and described below:

- *Uniform*: Each node starts at a random position and moves in a random direction with a constant velocity. The speed of each node is chosen randomly between a minimum and maximum value Whenever a node reaches a boundary of the simulated field, it bounces off and continues moving in a new direction. The Uniform model is based on work by Lee et al. [17] and is included in our study because each node's movement is independent but with high temporal dependency. In our implementation we use a minimum speed of 0.1 meters/second.
- *Random Waypoint*: Each node chooses a random destination within the simulated field and a speed between some minimum and maximum bounds. The node then moves to the destination, pauses for a fixed period of

time, and then chooses a new destination. We use the Random Waypoint model because of its popularity in other evaluations and because it represents a class of applications exhibiting random, independent movements. In our implementation, we choose a non-zero minimum speed in order to avoid the problem of having each node's average speed decrease over time [26]. For a given maximum speed, we set the minimum speed to 5 meters/second less. We use a pause time of 10 seconds.

- *Manhattan*: Each node moves along a set of pre-defined streets, which are arranged in a grid pattern. All nodes use the same speed, and each node may choose any direction when reaching an intersection. Nodes are initially placed uniformly along the streets. Our Manhattan model is based on previous grid-based models [1], [4] and represents path-based motion with low spatial dependence. In our implementation, the grids are placed 150 meters apart, and the speed of each node is set to a fixed value.
- *Exhibition*: Each node chooses a destination from among a fixed set of exhibition centers and then moves toward that center with a fixed speed. Once a node is within a certain distance of the center it pauses for a given time and then chooses a new center. This model is similar to the event scenario described by Johansson et al. [14] and represents independent movement but with high node density. Unless otherwise specified, our implementation uses 10 centers placed uniformly. When a node travels to a center, it stops when it is within 20 meters of the center and then pauses for 30 seconds. The speed of a node is random between a minimum and maximum value, as with Random Waypoint.
- *Battlefield*: Each node follows a group leader by choosing a destination close to where the leader is currently located and then moving to that destination. The group leader uses the Random Waypoint Model with a pause time of 10 seconds. As with the Exhibition model, each node maintains a minimum distance from the group leader. This model is similar to the RPGM model [7] and represents group-based mobility with high spatial dependence. In addition, this model can lead to significant partitioning, which is a useful boundary condition for routing protocols. In our implementation, we use 16 group leaders, which helps to increase the amount of partitioning that occurs. Each node adjusts its intended destination after every meter of movement, based on where its group leader is now located. The node will stop moving once it is within 20 meters of the leader. The speed of all nodes is random between a minimum and maximum value, as with Random Waypoint.

For each of these models, we use a range of speeds, including the case where all nodes are static. This tests the typical case where a group of people spontaneously meet in a workplace or public meeting space but remain seated during their encounter. For all models we avoid sharp turns and sudden changes in velocity by using acceleration and deceleration vectors [2].

Symbol	Interpretation
R	Radio range
$v_i(\vec{t})$	Velocity of node i at time t
$D_{i,j}(t)$	Distance between nodes i and j at time t
RD	$\frac{v_i(\vec{t}).v_j(\vec{t})}{\lvert v_i(\vec{t}) * v_j(\vec{t})\rvert}$
SR	$\frac{min(v_i(\vec{t}),v_j(\vec{t}))}{max(v_i(\vec{t}),v_j(\vec{t}))}$
$p_{i,j}$	$= \begin{cases} 1, D_{i,j}(t) < R \\ 0, D_{i,j}(t) > R \end{cases}$

TABLE II
SYMBOLS USED IN MOBILITY AND CONNECTIVITY METRICS

IV. MOBILITY AND CONNECTIVITY METRICS

Characterizing mobility models in terms of mobility and connectivity metrics can help to explain the impact of these models on routing performance. We implemented each of the metrics defined in the IMPORTANT framework [1] to determine whether they help to differentiate between the models we are using. Of these metrics, we found that spatial dependence and the average number of link changes are able to differentiate between our mobility models and help to explain multicast routing performance. These are defined as:

- *Spatial dependence:* Spatial dependence [1] characterizes the degree to which two nodes are moving in a similar direction with similar speed. For two nodes, i and j at time t, spatial dependence is defined as :

  $$D_{spatial}(i,j,t) = RD(v_i(\vec{t}), v_j(\vec{t})) * SR(v_i(\vec{t}), v_j(\vec{t})),$$

  where RD is the relative direction and SR is the speed ratio as defined in Table II.
  Since spatial dependence matters only when nodes are nearby, the following condition also applies:

  $$D_{i,j}(t) > C_1 * R \implies D_{spatial}(i,j,t) = 0,$$

  where C_1 is a constant that we set to 2.
- *Number of Link Changes:* The number of link changes seen during the course of a simulation [1]. A link change is defined as an event when two nodes come within radio range when previously they had not been able to communicate directly.

In addition, because we are interested in studying the effects of partitioning and high node density on multicast routing protocols, we have created two new connectivity metrics reflecting these characteristics:

- *Neighbor Density:* The number of nodes which are within radio range of a given node. For node i, this

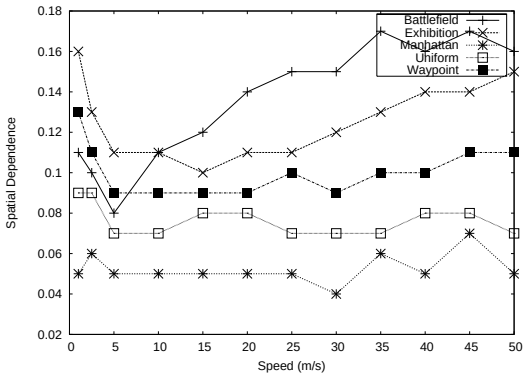

Fig. 1. Spatial Dependence

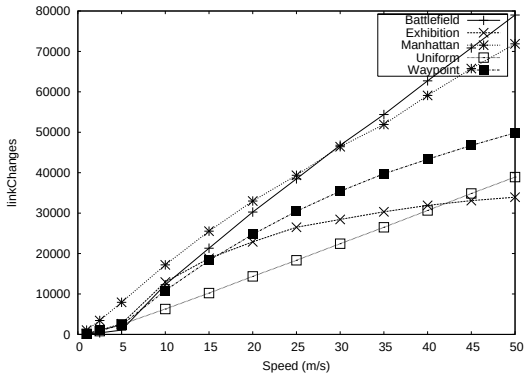

Fig. 2. Number of Link Changes

is defined as

$$N_i = \sum_{j=1}^{N} p_{i,j},$$

where p determines whether two nodes are within radio range, as shown in Table II.

- *Reachability:* The number of nodes that are reachable via forwarding through the ad hoc network. We measure this using a recursive coloring algorithm. We begin by placing a given node into a queue. As long as the queue is not empty we remove a node from the queue, color all nodes within radio range, and place any newly-colored nodes into the queue. When the queue is empty, the number of colored nodes indicates the size of a partition; the size of this partition is the number of nodes reachable by all nodes within the partition. Reachability is averaged over all nodes in the system. The ideal value for reachability is $N - 1$, where N is the number of nodes in the system.

To determine whether these metrics help to differentiate among our mobility models, we ran several simulations and measured the above metrics. We extended GloMoSim-2.03 [27] to include the Uniform, Manhattan, Exhibition, and Battlefield mobility models. We use 50 nodes, randomly placed over a square field whose length and width is 1000 meters. We use IEEE 802.11 as the MAC protocol and a free-space model for radio signal propagation. All simulations were run for 600 seconds and we average the results of 25 simulations for each data point.

Figures 1- 4 show values for each of the metrics, averaged across all nodes, as a function of speed. For spatial dependence and node density we collect 50 samples during the course of a simulation and average them. We collect link change and reachability information each time a node moves, and reachability is averaged over the course of the simulation.

As was seen with the IMPORTANT framework, spatial dependence is highest for group-based models. However, the differences between our models are much smaller than seen in their work, since we use a much larger number of centers

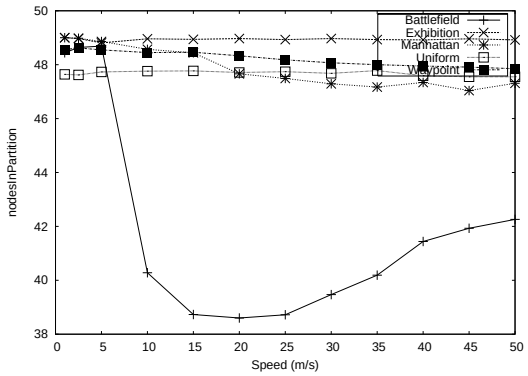

Fig. 3. Reachability

or leaders for the group-based models. One surprise is that the number of link changes is able to clearly differentiate between the models, whereas this was not seen with IMPORTANT. This is significant because the number of link changes has a significant impact on a multicast routing protocol – each link change may potentially break the multicast tree and cause a receiver or intermediate node to attempt a repair.

Another significant difference among the protocols is that all of them maintain high reachability except for Battlefield. This means that the Battlefield model is a good test for determining how well a protocol reacts to a partition in the network.

As we expected, node density is much higher for the group-based models than it is for the others. This effect is primarily seen during higher speeds because this enables the group members to more easily congregate around their center or leader. This effect is not seen at lower speeds because nodes are initially placed randomly; low speeds will make it difficult for a node to catch up to its leader or reach a center.

Finally, we note that we also implemented the relative speed and link duration metrics from the IMPORTANT framework but were unable to use them to differentiate between our mobility models.

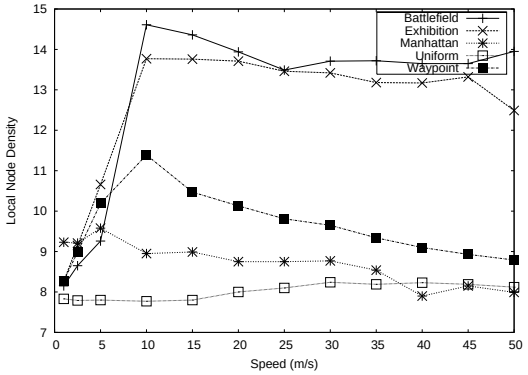

Fig. 4. Node Density

V. SIMULATION ENVIRONMENT AND METHODOLOGY

A. Protocols Evaluated

To determine the impact of mobility models on multicast routing protocol performance, we simulated flooding, ODMRP [16], and ADMR [12]. We chose these latter two protocols because they operate *on-demand* rather than proactively maintaining routes. Several performance studies indicate that these protocols perform well [17], [12].

For flooding, we use a simple protocol in which each node receiving a packet for a group first checks whether it is a duplicate and, if not, forwards the packet by retransmitting it. To check for duplicates, each node stores the sequence number of the last packet it receives for each multicast group. More details on the use of flooding for multicast can be found in [6].

1) ODMRP: ODMRP is a mesh-based demand-driven multicast protocol, similar to DVMRP [25] for wired networks. A source periodically builds a multicast tree for a group by flooding a control packet throughout the network. Nodes that are members of the group respond to the flood and help the source to establish the tree. Nodes that are on the tree use *soft state*, meaning their status as forwarders for a given group times out if not refreshed. Because the source rebuilds the tree periodically, the set of forwarders at any one time actually forms a mesh, providing robustness for the mobile receivers.

In particular, an active ODMRP source periodically floods a JOIN QUERY message throughout the entire network. Each node receiving this message stores the previous hop from which it received the message. When a group member receives the JOIN QUERY, it responds by sending a JOIN REPLY to the source, following the previous hop stored at each node. Nodes that forward a JOIN REPLY create soft forwarding state for the group, which must be renewed by subsequent JOIN REPLY messages. If the node is already an established forwarding member for that group, then it suppresses any further JOIN REPLY forwarding in order to reduce channel overhead.

The basic trade-off in ODMRP is between throughput and overhead. A source can increase throughput by sending more frequent JOIN QUERY messages. Each message rebuilds the multicast tree, repairing any breaks that have occurred since the last query, thus increasing the chance for subsequent packets to be delivered correctly. However, because each query is flooded, increases the query rate also increase the overhead of the protocol. ODMRP can also control redundancy via the soft-state timer for node forwarding state. A longer timer will increase the size of the mesh and hence provide more redundant paths for packets to be delivered. Of course, increasing the soft-state timer also increases overhead since many of the links in the mesh will result in duplicate packets being delivered.

2) ADMR: ADMR creates source-specific multicast trees, using an on-demand mechanism that only creates a tree if there is at least one source and one receiver active for the group. Unlike ODMRP, receivers must explicitly join a multicast group. Sources periodically send a network-wide flood, but only at a very low rate in order to recover from network partitions. In addition, forwarding nodes in the multicast tree may monitor the packet forwarding rate to determine when the tree has broken or the source has become silent. If a link has broken, a node can initiate a repair on its own, and if the source has stopped sending then any forwarding state is silently removed. Receivers likewise monitor the packet reception rate and can re-join the multicast tree if intermediate nodes have been unable to reconnect the tree.

To join a multicast group, an ADMR receiver floods a MULTICAST SOLICITATION message throughout the network. When a source receives this message, it responds by sending a unicast KEEP-ALIVE message to that receiver, confirming that the receiver can join that source. The receiver responds to the KEEP-ALIVE by sending a RECEIVER JOIN along this same unicast path.

In addition to the receiver's join mechanism, a source periodically sends a network-wide flood of a RECEIVER DISCOVERY message. Receivers that get this message respond to it with a RECEIVER JOIN if they are not already connected to the multicast tree.

Each node acting as a receiver or forwarder maintains a counter of recently received packets, and if a certain number of consecutive packets (2 for our simulations) are not received by a receiver, then it concludes that it has become disconnected for the group and it starts a repair process. A node that is a pure receiver (and not a forwarder for that source/group) simply re-joins the group by sending a MULTICAST SOLICITATION message. A node that is only a forwarder sends a REPAIR NOTIFICATION message down its subtree to determine whether it is the closest node to where the packet loss is occurring. Any downstream nodes cancel their own disconnect timers when getting this notification. Once a node has determined that it is the most upstream node

that has been disconnected, it transmits a hop-limited flood of a RECONNECT message. Any forwarder receiving this message forwards the RECONNECT up the multicast tree to the source. The source in return responds to the RECONNECT by sending a RECONNECT REPLY as a unicast message that follows the path of the RECONNECT back to the repairing node.

A receiver keeps track of how many times it has had to initiate a repair due to a disconnection timeout. If this number reaches a certain threshold then the receiver believes that it has encountered a situation of high mobility. In the next RECEIVER JOIN message sent to the source, the receiver sets a high mobility flag as a signal to the source indicating that the network is encountering high mobility. When the source receives a particular number of join messages with the high mobility flag on, then it switches to flooding for a limited amount of time. During flooding, all the data packets are sent as network-wide flood and all repair messages are suppressed.

B. Evaluation Metrics

In our simulations we have analyzed the following metrics to study the effects of mobility on each of the multicast routing protocols:

- **Throughput**: The ratio of the number of packets received to the number of packets sent.
- **Delay**: The difference between the time when the packet is sent by the source and when it is received by a receiver.
- **Transmission Overhead**: The ratio of the number of data messages transmitted (originated or forwarded) and the number of data messages received. This metric is a measure of the efficiency of a routing protocol – a lower value for transmission overhead indicates that fewer forwarders were needed. A value of 1 indicates perfect efficiency, meaning all receivers were within radio range of the source, but this may not be achievable for a given configuration.
- **Control Overhead**: The ratio of the number of control messages originated or forwarded over the combined total of data and control messages originated or forwarded. This metric indicates the percentage of all messages that are control messages.

Note that previous studies have combined transmission overhead and control overhead into a single metric called normalized overhead, but this can obscure the differences between the two.

For ODMRP, control packets consist of JOIN QUERY, JOIN REPLY, and ACKNOWLEDGMENTS. For ADMR, control packets consist of RECEIVER DISCOVERY, MULTICAST SOLICITATION, KEEP-ALIVE, RECEIVER JOIN, REPAIR NOTIFICATION, RECONNECT, RECONNECT REPLY, and also ACKNOWLEDGMENTS sent by the end receivers in order to maintain the tree. Since ODMRP JOIN QUERIES and ADMR RECEIVER DISCOVERY messages also have data piggybacked with them, we count these packets as both data and control messages.

C. Methodology

For our simulations we use GloMoSim-2.03 [27], which we have extended to include the Uniform, Manhattan, Exhibition, and Battlefield mobility models. We used the ODMRP implementation provided in the GloMoSim distribution, but fixed several major bugs to correct implementation flaws that we discovered [1]. Since GloMoSim did not include ADMR, we wrote our own implementation based on the original ADMR publication [12] and a specification published as an Internet draft [11]. We did not implement source pruning (where the source stops sending data if there are no receivers) so that we could study the effects of partitioning on packet loss. We also wrote a simple flooding protocol for GloMoSim.

One important implementation detail for multicast routing protocols is the use of randomization (or jitter) to avoid collisions due to protocol synchronization. Each node that forwards a multicast message adds a random delay between 0 and 10 ms before forwarding the packet. Likewise, at the application layer, we avoid starting sources at the same time, since they use CBR and would thus remain synchronized for the duration of the simulation. Finally, for ADMR we exclude any startup delay caused by buffering packets before sufficient receivers have joined the group.

To verify our protocol implementations, we ran simulations identical to those reported in [12] that compare ADMR and ODMRP. Our results are very close, with slightly higher delay due to the jitter we have added at each node. The results reported in this paper differ more substantially from [12] because we are using a different field size.

For each simulation we use 50 nodes, randomly placed over a square field whose length and width is 1000 meters. For the Manhattan model, nodes may only be placed on one of the streets. To generate multicast traffic, we use three multicast groups, each consisting of 7 receivers. The multicast groups are not overlapping, which means that with the 3 senders and 21 receivers combined about half of the nodes are participating in a multicast session. Each multicast source uses a Constant Bit Rate (CBR) flow, transmitting a 64 byte packet every 250 milliseconds. Nodes communicate using IEEE 802.11 for the MAC protocol, with free-space radio signal propagation.

We run each simulation for 600 seconds and we average the results of 25 simulations for each data point.

VI. RESULTS

Mobility can affect routing performance in three different ways. First, mobility can break the existing mesh or tree established by the multicast routing protocol. This leads to

[1] See `http://nrg.cs.uoregon.edu/adhoc-net/index.html` for patches.

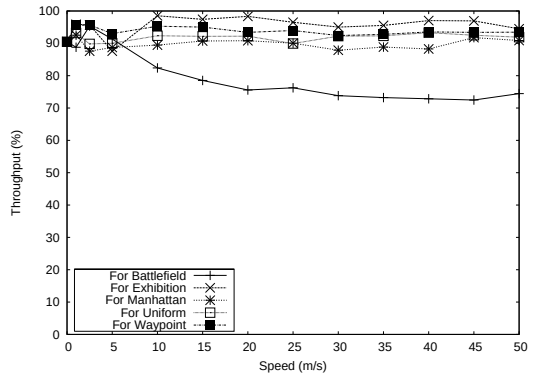

Fig. 5. Throughput for Flooding

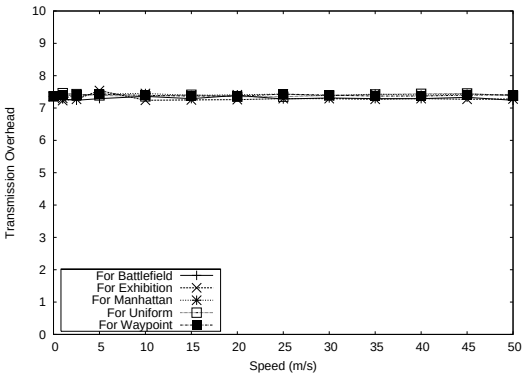

Fig. 6. Transmission Overhead for Flooding

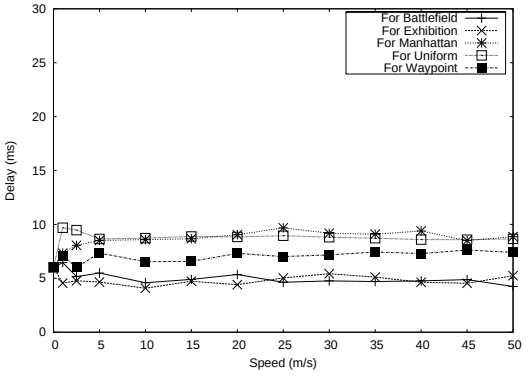

Fig. 7. Delay for Flooding

packet loss and may cause the multicast routing protocol to initiate a repair event in order to restore connectivity among the group members. Second, mobility may actually partition the system, in which case even a simple flooding protocol will be unable to deliver packets to the entire group. During this period, the main goal of a multicast routing protocol is to avoid high overhead due to attempted repairs. Finally, mobility can increase the density of nodes in some areas of the system. This is a mixed blessing: although there are more nodes to facilitate transfer of data, they may also cause collisions at the physical layer.

In this section, we explore the effects of different mobility patterns on three different multicast protocols – flooding, ODMRP, and ADMR. We conclude by examining the effects of high node density.

A. Flooding

As expected, flooding attains very high throughput at the expense of high transmission overhead (Figure 5 and Figure 6). This is due to each node forwarding every non-duplicate packet it receives. Note that the transmission overhead of 7 is a worst-case scenario because there are 49 nodes forwarding every packet and only 7 receivers.

There are two interesting cases with respect to throughput. First, throughput is sometimes lower when the nodes are static because the initial placement may result in partitioning. These partitions are never healed due to lack of any mobility. Second, the Battlefield model results in a relatively low throughput for flooding. Because all packets are flooded, this must be due to partitioning. This verifies the impact of the reachability parameter discussed in Section IV and sets an upper bound on the performance any other multicast routing protocol can achieve.

Flooding also establishes a lower bound on delay for each of the mobility models (Figure 7). Delay is lowest for Battlefield and Exhibition because group members have a higher likelihood of being near the source (i.e. if the source is the group leader, following the same leader, or visiting the same center). This is predicted by these two models having higher node density (Figure 4). Note that this also correlates with these two models having higher spatial dependence – these two metrics are very similar. Delay is highest for Uniform and Manhattan because nodes are likely to be both well-connected and spread out over the entire field.

B. ODMRP

For ODMRP, throughput (Figure 8) depends on the model and the ordering from worst-to-best is roughly predicted by the number of link changes shown in Figure 2. Throughput for Battlefield is even worse than for Exhibition, despite a similar number of link changes, because of its much lower reachability. Throughput for the Uniform model is the only exception to this ordering, and this can be explained by its lower node density.

The correlation between link changes and throughput makes sense because ODMRP's design represents a basic trade-off between reaction time to link changes and overhead. Recall that an ODMRP source sends a periodic JOIN QUERY to establish the mesh. In between queries, the mesh degrades whenever mobility causes a link to break, and these breaks are not repaired until the next JOIN QUERY. Furthermore, as the speed of the nodes increases, link breaks occur more frequently, so the throughput decreases with speed.

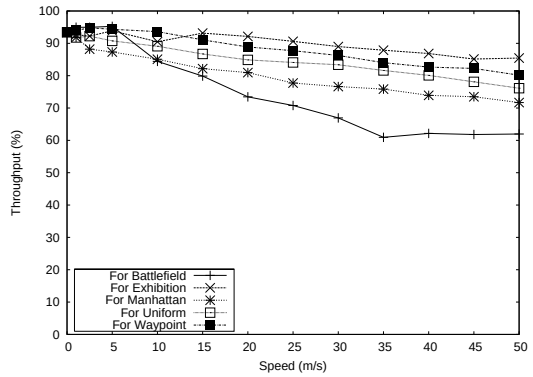

Fig. 8. Throughput for ODMRP

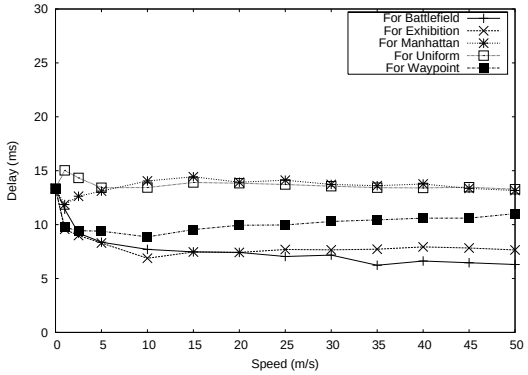

Fig. 10. Delay for ODMRP

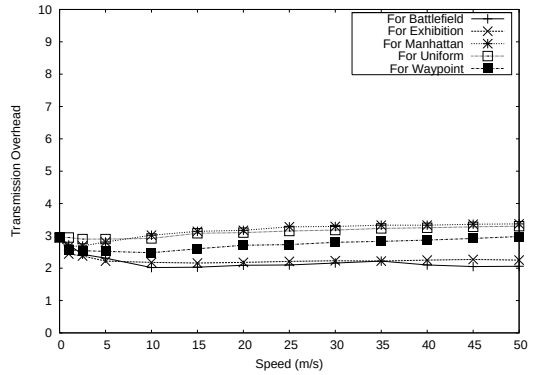

Fig. 9. Transmission Overhead for ODMRP

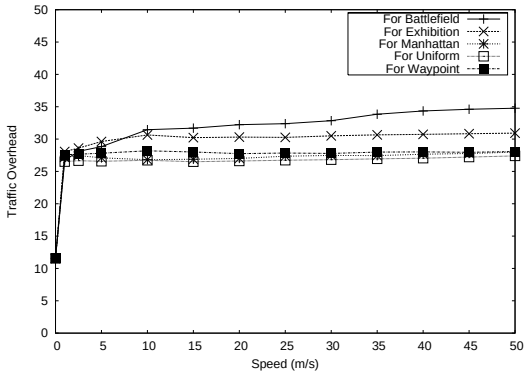

Fig. 11. Control Overhead for ODMRP

Nevertheless, what makes ODMRP significantly better than flooding is it's ability to achieve good throughput with much lower transmission overhead (Figure 9). For ODMRP, approximately 2.5 packets are forwarded for every packet received. ODMRP could have very high throughput by increasing the join query rate, but then this becomes flooding at very high rates, with a corresponding increase in transmission overhead. For both transmission overhead and delay, the ordering among models is the same as for flooding and again correlates well with node density, for similar reasons. Group-based mobility results in group members having a higher likelihood of being near the source, which can be expected to reduce delay and transmission overhead.

The increased efficiency of ODMRP results in added control overhead, which was absent in case of flooding (Figure 11). Note that the high values shown for overhead ($25 - 35\%$) are due to the combination of low traffic rate (4 packets per second) and periodic flooding (once per 3 seconds). With higher traffic rates, the percentage of overhead becomes much lower.

The ordering of models in this graph is again similar to that of node density, although the correlation between node density and control overhead is a little harder to explain. With ODMRP, a receiver sending a JOIN REPLY must receive an acknowledgment or else the reply is retransmitted. If the reply is sent to a forwarding node, then when the forwarding node transmits the reply toward the sender this transmission acts as a passive acknowledgment. However, when the sender receives a JOIN REPLY, it must respond with an explicit acknowledgment since it will not be forwarding the reply further. These explicit acknowledgments are more likely to occur when there are more group members near the sender, as is the case with models having higher node density.

C. ADMR

The most sophisticated of all the three protocols studied, ADMR is able to maintain high throughput for nearly all of the mobility models, even as speed increases (Figure 12). This is due to two mechanisms in ADMR. First, forwarding nodes are able to initiate local repair of the multicast tree when they determine that packet loss is occurring. Second, receivers experiencing high packet loss can ask ADMR to switch to flooding.

Is either one of these mechanisms – local repair or adaptive flooding – more important in obtaining high throughput? We disabled the adaptive flooding mechanism and found that throughput declines for some models, particularly at higher speeds (Figure 13). In other simulations, we increased the threshold for initiating local repair (from 2 consecutive

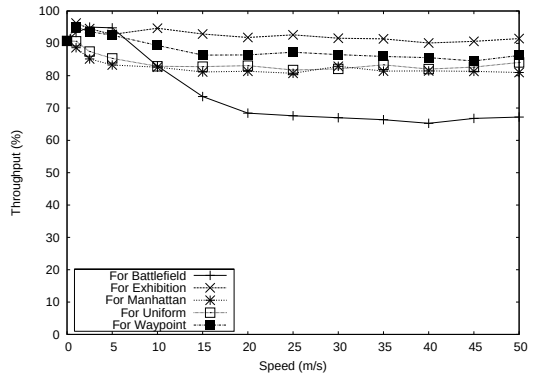

Fig. 12. Throughput for ADMR

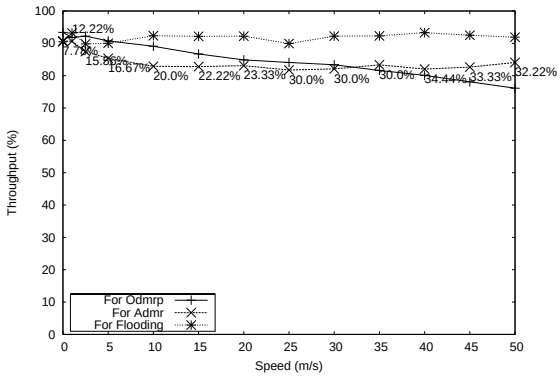

Fig. 14. Adaptive fbooding for ADMR at high speeds with the Uniform model

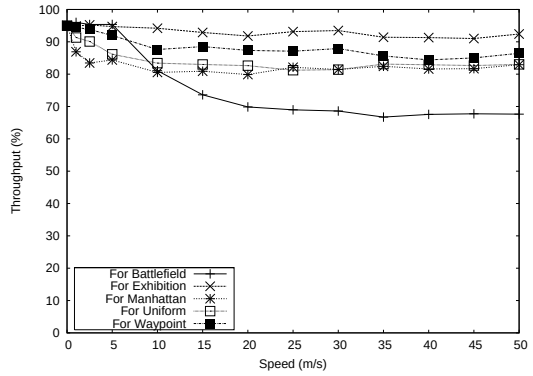

Fig. 13. Throughput for ADMR with adaptive fbooding disabled

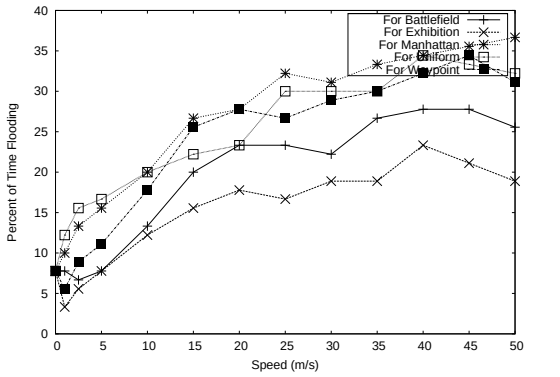

Fig. 15. Percent of time ADMR uses fbooding

losses to 10 losses) and found that this also decreased throughput for some models. This indicates that both local repair and adaptive flooding are important parts of ADMR, though further study is needed to determine whether these mechanisms can be improved further.

Motivation for improving ADMR's adaptive mechanisms can be seen by comparing the throughput of each of the multicast protocols for the Uniform mobility model (Figure 14). For some models, both flooding and ODMRP are able to achieve higher throughput than ADMR at low speeds. This could indicate that local repair can be used more efficiently to recover from loss at low speed.

Another interesting finding is that the percentage of time spent flooding does not continue increasing as speed increases (Figure 15). For most models, ADMR maintains an even level of flooding about $10 - 20\%$ of the time once the speed increases above $10m/s$. This behavior is a consequence of ADMR automatically switching back to normal operation after a fixed period of time. Conceivably, ADMR could increase the amount of time spent flooding each time the adaptive flooding mechanism is triggered. This would allow ADMR to effectively become a flooding protocol for high speed environments. Despite not doing this, ADMR still manages to achieve throughput of $80 - 90\%$ at high speeds for most of the mobility models, which is impressive.

The consequence of performing both adaptive flooding is that this increases transmission overhead for ADMR when speeds increase (Figure 16). At higher speeds, ADMR's transmission overhead approaches that of flooding. As with ODMRP and flooding, the relative performance of the mobility models correlates to node density for both transmission overhead and delay (Figure 17). ADMR does have slightly higher delay than ODMRP; this can be explained by the increased number of control messages in ADMR, which may lead to collisions and retransmissions at the MAC layer.

Control overhead for ADMR may decrease as the speed increases, depending on the mobility model (Figure 18). This is because ADMR switches to flooding, which decreases the amount of control traffic due to local repair and member adaptation to loss. This trend is not as evident with the group-based mobility models because flooding in areas of high node density can lead to more collisions and hence more control traffic (when nodes try to recover from the resulting loss).

D. High Node Density

Studying packet-level traces of ODMRP and ADMR has led us to investigate the behavior of these protocols under

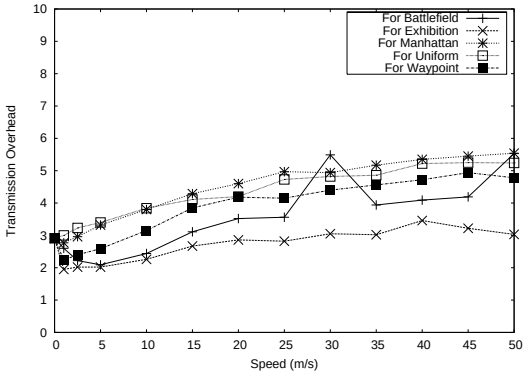

Fig. 16. Transmission Overhead for ADMR

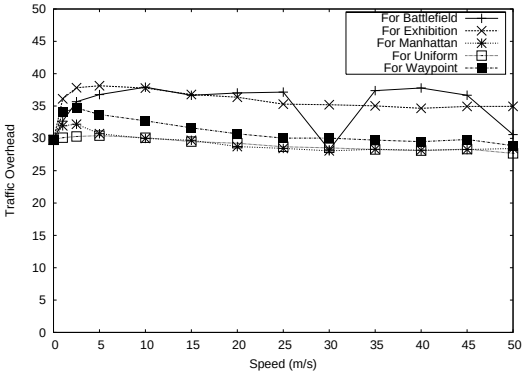

Fig. 18. Control Overhead for ADMR

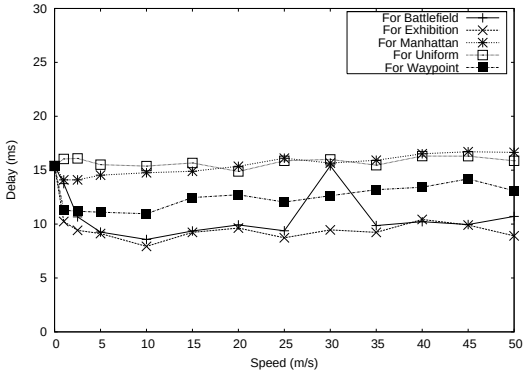

Fig. 17. Delay for ADMR

conditions of high node density. Figure 19 and 20 show the ODMRP and ADMR traffic, respectively, for three nearby nodes for a period of 8 seconds. The horizontal lines in the graph indicate the different views seen by each of these nodes. The y-axis on each graph is the sequence number of a packet, modulo 30, as observed by each of the three nodes. Each mark on the graph represents a different type of packet being received, as indicated by the legend. These traces are taken from our previous simulations of the Battlefield model using a maximum speed of 50 meters/second. Note that the time period for each trace is identical.

For ODMRP, the only possible events in the trace are packets being received or packets being forwarded for some other group. Gaps in the sequence space indicate packet loss events, for which ODMRP takes no special action. With ADMR, however, we see that packet loss events may trigger substantial activity, with many types of control messages sent. In exploring a variety of traces, we found that periodic bursts of load on the network were not uncommon.

ADMR's bursts of control traffic are due to both local repair events at a forwarding node and repair events at a receiver. For the local repair event, intermediate nodes flood REPAIR NOTIFICATION messages within a limited area to determine which node is at the top of the *loss tree*. Once this is determined, the repairing node attempts to find some other node that is still connected to the multicast tree by flooding a RECONNECT message in a limited area. For the receiver repair event, a receiver floods a MULTICAST SOLICITATION message throughout the entire network in an effort to locate the source. These message floods, combined with other unicast control messages, can cause significant amounts of traffic. Moreover, repair events are not coordinated across multicast groups, so congestion-induced packet loss can potentially cause a repair event storm.

These observations motivate us to study a high density scenario – how do ADMR's repair mechanisms react to congestion-induced loss as opposed to mobility-induced loss? To explore this idea, we simulated a scenario in which a group of people gather for a spontaneous meeting in a public area. For this scenario we use the Exhibition model with only one center and no mobility. The nodes are placed near the center so that they are all within radio range of each other. In addition, we use a single group of 30 nodes so that most nodes in the gathering place are participants in the meeting. The sender still uses a CBR flow, but with 100 byte packets. We then increase the traffic rate for the group from 4 packets per second to 200 packets per second.

Under this high density scenario, both flooding and ADMR experience a dramatic loss of throughput once the packet rate approaches 30 packets per second (Figure 21). There is a corresponding increase in the number of MAC-layer collisions at this time (Figure 22). This behavior is surprising because 30 packets per second, with 100 byte packets, is only a 24 kbps flow! This result is not surprising for flooding, because each node must forward every packet, even though they all receive it directly from the source. In effect, this multiplies the amount of traffic 50 times.

ADMR's poor performance in this high density situation is due to a combination of factors. First, ADMR reacts to widespread packet loss by switching to flooding. This is exactly the wrong behavior when the packet loss is due to congestion; flooding greatly multiples the amount of traffic in the system. Second, initiating a local repair can

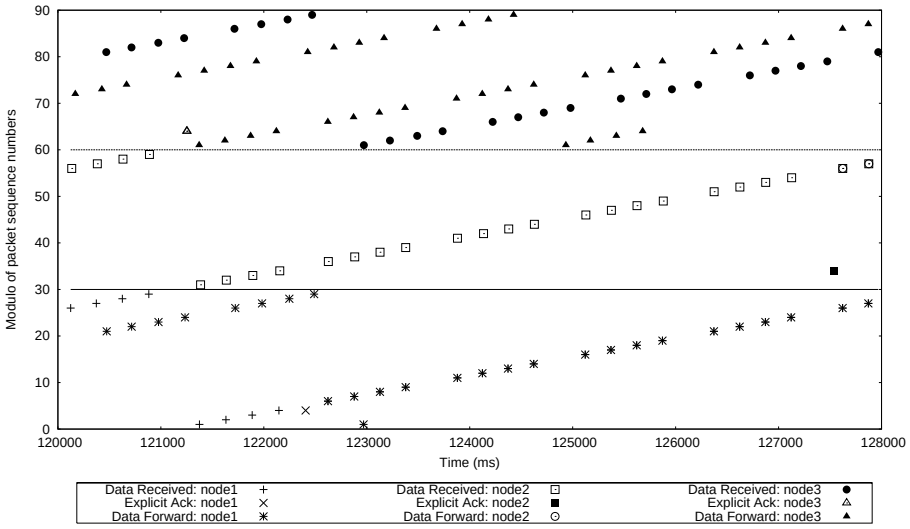

Fig. 19. Packet Trace for ODMRP

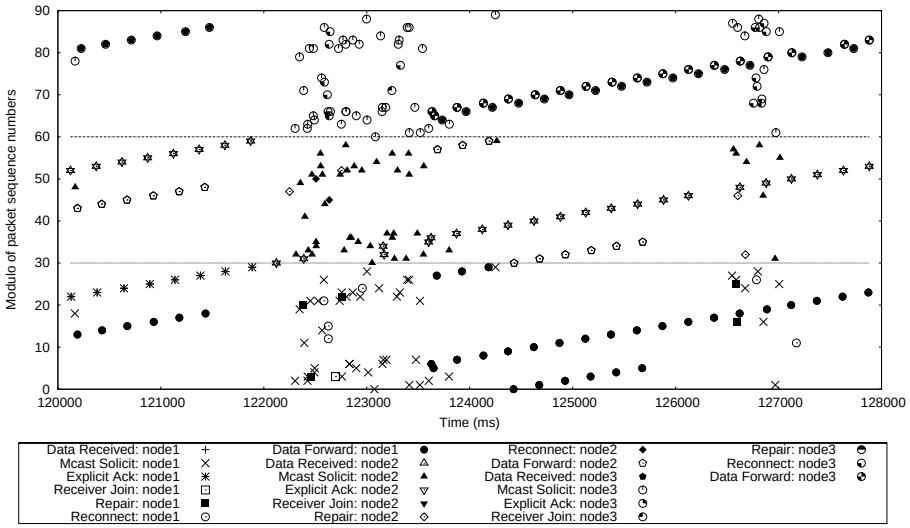

Fig. 20. Packet Trace for ADMR

also be harmful as a reaction to congestion-induced losses. Each repair attempt involves flooding a number of control messages, as shown in the packet traces above, causing additional congestion. Finally, we were surprised to find that using ADMR in high-density scenarios causes *ack implosion* at the source [5].

In ADMR, each forwarding node may automatically prune itself from the multicast tree if it determines that it is no longer a necessary part of the tree. To make this decision, each node listens for passive acknowledgments in the form of some downstream node forwarding its packets. As long as these passive acknowledgments continue, the node maintains its multicast forwarding state. However, nodes that are at the leaves of the multicast tree will not receive any passive acknowledgments. As a result, all group members must send explicit acknowledgments to their parent in the tree, indicating that the parent must continue acting as a forwarder [11].

In our high density scenario, there are usually no forwarders in the multicast tree, since the source can reach all receivers with its own transmission. However, this means that the source will not receive any passive acknowledgments and must collect active acknowledgments from at least once receiver. Unfortunately, ADMR does not include any mechanism for damping these acknowledgments. Each time the source sends a packet, all receivers send it an acknowledgment, leading to *ack implosion*. In effect, an ack implosion acts like a flood of the original message, except that the explicit acknowledgment does not carry a payload.

The only time we observed good performance for ADMR

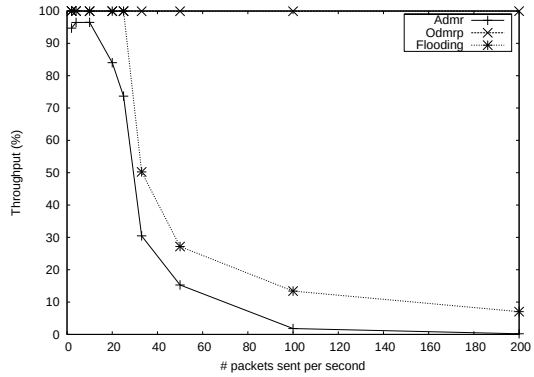

Fig. 21. Throughput with increasing traffic rate

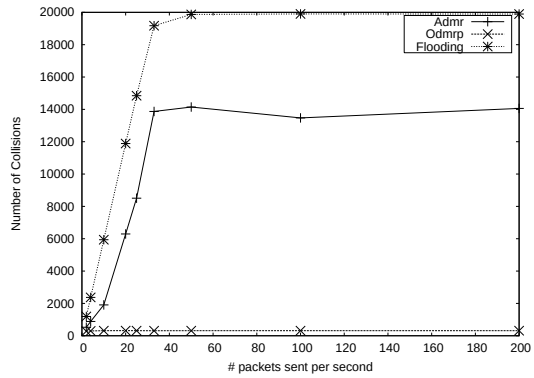

Fig. 22. Collisions with increasing traffic rate

in a high density situation was when all three of these mechanisms – local repair, adaptive flooding, and active acknowledgments – were disabled. This is clearly not a practical solution as this essentially cripples ADMR. We must conclude that it is problematic to use loss events as an indicator of mobility.

VII. CONCLUSIONS

We find that mobility patterns can significantly affect the performance of a multicast routing protocol. Our results show that the number of link changes imposed by a particular application is a good predictor of throughput, with greater amounts of link changes indicating worse performance. Even when the number of link changes is small, low node density and low spatial dependence can also degrade throughput. We show that high values of node density, which is typically exhibited by group-based mobility patterns, can lead to lower delay and lower transmission overhead. However, group-based mobility patterns can also cause partitioning, leading to significant packet loss.

Our study reveals some interesting results for ADMR. With low levels of traffic, ADMR's adaptive flooding and local repair mechanisms combine to provide good throughput even at high speeds. The cost of these mechanisms is increased transmission overhead, so that ADMR represents a compromise between flooding and ODMRP. However, we find that during high node density and high traffic situations ADMR interprets congestion-induced packet loss as a sign of mobility. This causes ADMR to invoke adaptive flooding and local repair, which exacerbate the situation and lead to congestion collapse. Another significant factor in the congestion collapse is ADMR's tendency to induce ack implosions during high density.

These results lead us to question whether it is possible to design a multicast routing protocol that can achieve good throughput during periods of high mobility, yet still function well in a high density scenario. We plan to investigate this question by applying our methodology to other ad hoc multicast routing protocols.

REFERENCES

[1] F. Bai, N. Sadagopan, and A. Helmy. IMPORTANT: A framework to systematically analyze the Impact of Mobility on Performance of RouTing protocols for Adhoc Networks. In *IEEE INFOCOM*, 2003.

[2] C. Bettstetter. Smooth is Better than Sharp: A Random Mobility Model for Simulation of Wireless Networks. In *ACM MSWiM*, July 2001.

[3] T. Camp, J. Boleng, and V. Davies. A Survey of Mobility Models for Ad Hoc Network Research. *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2002.

[4] V.A. Davies. Evaluating Mobility Models Within an Ad Hoc Network. Master's thesis, Colorado School of Mines, Mathematical and Computer Sciences, 2000.

[5] A. Erramilli and R.P Singh. A Reliable and Efficient Multicast Protocol for Broadband Broadcast Networks. In *ACM SIGCOMM*, August 1987.

[6] C. Ho, K. Obraczka, G. Tsudik, and K. Viswanath. Flooding for Reliable Multicast in Multi-Hop Ad Hoc Networks. In *International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 1999.

[7] X. Hong, M. Gerla, G. Pei, and C. Chiang. A Group Mobility Model for Ad Hoc Wireless Networks. In *ACM MSWiM*, August 1999.

[8] X. Hong, T. Kwon, M. Gerla, D. Gu, and G. Pei. A Mobility Framework for Ad hoc Wireless Networks. In *ACM Second International Conference on Mobile Data Management*, January 2001.

[9] Y. Hu and D.B. Johnson. Caching Strategies in On-Demand Routing Protocols for Wireless Ad Hoc Networks. In *ACM MobiCom*, August 2000.

[10] A. Jardosh, E. M. Belding-Royer, K. C. Almeroth, and S. Suri. Towards Realistic Mobility Models for Mobile Ad hoc Networks. In *ACM MobiCom*, September 2003.

[11] J. Jetcheva and D. B. Johnson. Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks. Internet Draft: work in progress, July 2001.

[12] J. Jetcheva and D.B. Johnson. Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks. In *ACM MobiHoc*, October 2001.

[13] Lusheng Ji and M. Scott Corson. Differential Destination Multicast - A MANET Multicast Routing Protocol for Small Groups. In *IEEE INFOCOM*, 2001.

[14] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark. Scenario Based Peformance Analysis of Routing Protcols for Mobile Ad Hoc Networks. In *IEEE MobiCom*, August 1999.

[15] D.B. Johnson and D.A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[16] S. Lee, W. Su, and M. Gerla. On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks. In *ACM/Baltzer Mobile Networks and Applications, special issue on Multipoint Communication in Wireless Mobile Networks*, 2000.

[17] S. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia. A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols. In *IEEE INFOCOM*, 2000.

[18] B. Liang and Z.J. Haas. Predictive Distance-Based Mobility Management for PCS Networks. In *IEEE INFOCOM*, March 1999.

[19] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *ACM SIGCOMM*, 1994.

[20] C.E. Perkins and E.M. Royer. Ad Hoc On Demand Distance Vector (AODV) Algorithm. In *IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[21] E. Royer, P. Melliar-Smith, and L. Moser. An Analysis of the Optimum Node Density for Ad hoc Mobile Networks. In *IEEE International Conference on Communications*, June 2001.

[22] E.M. Royer and C.E. Perkins. Multicast Operation of the Ad-Hoc On-Demand Distance Vector Routing Protocol. In *Mobile Computing and Networking*, 1999.

[23] M. Sanchez. Re: Mobility pattern in a MANET June 25. IETF MANET Mailing list, 1999.

[24] J. Tian, J. Haehner, C. Becker, I. Stepanov, and K. Rothermel. Graph-Based Mobility Model for Mobile Ad Hoc Network Simulation. In *ACM Simulation Symposium*, 2002.

[25] D. Waitzman, C. Partridge, and S. Deering. Distance Vector Multicast Routing Protocol. RFC 1075, November 1988.

[26] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful, 2003.

[27] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks. In *Workshop on Parallel and Distributed Simulation*, May 1998.