REIHE INFORMATIK
TR-04-002

A Hierarchical Approach to Position-Based Multicast for Mobile Ad-hoc Networks

Matthias Transier[*], Holger Füßler[*], Jörg Widmer[†], Martin Mauve[‡], and Wolfgang Effelsberg[*]

[*] Universität Mannheim
Praktische Informatik IV
L15, 16
D-68131 Mannheim, Germany

[†] EPFL
Institute of Communication Systems
CH-1015 Lausanne, Switzerland

[‡] Universität Düsseldorf
Institut für Informatik
Universitätsstraße 1
D-40225 Düsseldorf, Germany

A Hierarchical Approach to Position-Based Multicast for Mobile Ad-hoc Networks

Matthias Transier*, Holger Füßler*, Jörg Widmer†, Martin Mauve‡, and Wolfgang Effelsberg*
* University of Mannheim, Germany
† EPFL, Switzerland
‡ University of Düsseldorf, Germany

***Abstract*— In this paper we present Scalable Position-Based Multicast (SPBM), a multicast routing protocol for ad-hoc networks. SPBM uses the geographic position of nodes to provide a highly scalable group membership scheme and to forward data packets in a way that is very robust to changes in the topology of the network. SPBM bases the forwarding decision on whether there are group members located in a given direction or not, allowing for a hierarchical aggregation of membership information: the further away a region is from an intermediate node the higher the level of aggregation should be for this region. Because of aggregation, the overhead for group membership management scales logarithmically with the number of nodes and is independent of the number of multicast senders for a given multicast group. Furthermore, we show that group management overhead is bounded by a constant if the frequency of membership updates is scaled down with the aggregation level. This scaling of the update frequency is reasonable since the higher the level of aggregation the lower the number of membership changes for the *aggregate*. The performance of SPBM is investigated by means of simulation, including a comparison with ODMRP, and through mathematical analysis. We also describe an open source kernel implementation of SPBM that has been successfully deployed on hand-held computers.**

I. Introduction

Many applications envisioned for mobile ad-hoc networks rely on group communication. Communication during disaster relief, networked games, and emergency warnings in vehicular networks are common examples for these applications. As a consequence, multicast routing in mobile ad-hoc networks has received significant attention over the recent years.

In this paper we present Scalable Position-Based Multicast (SPBM), an ad-hoc multicast routing protocol comprising a multicast forwarding strategy and a group membership scheme to determine where members of a multicast group are located. The forwarding strategy uses information about geographic positions of group members to make forwarding decisions. In contrast to existing approaches it neither requires the maintenance of a distribution structure (i.e., a tree or a mesh) nor resorts to flooding. The group membership scheme uses knowledge about geographic positions for a hierarchical aggregation of membership information.

The forwarding of packets by SPBM is a generalization of position-based unicast routing as proposed, e.g., in [1] and [2]. In these protocols, a forwarding node selects one of its neighbors as a next hop in a *greedy* fashion, such that the packet makes progress toward the geographic position of the destination. It is possible that a node has no neighbor with progress toward the destination although a valid route to the destination exists. The packet is then said to have reached a local optimum. In this case a *recovery strategy* is used to escape the local optimum and to find a path toward the destination. The most important characteristic of position-based routing is that forwarding decisions are only based on local knowledge. It is not necessary to create and maintain a global route from the sender to the destination. Therefore, position-based routing is commonly regarded as highly scalable and very robust against frequent topological changes. In order to extend position-based routing to multicast, SPBM provides an algorithm for splitting multicast packets in intermediate nodes when destinations for that packet are no longer located in the same direction. This strategy includes both greedy forwarding and the recovery strategy.

The second important element of SPBM is its group membership scheme. It relies on geographic information to achieve scalability: instead of maintaining a fixed distribution structure, an intermediate node just needs to know whether group members are located in a given direction or not. This allows for a hierarchical aggregation of membership information: the further away a region is from an intermediate node the higher can be the level of aggregation for this region. Therefore, group membership management can be provided with an overhead that scales logarithmically with the number of nodes and that is independent of the number of multicast senders in a multicast group. A second observation is then used to reduce this overhead further: the higher the level of aggregation (i.e., the more nodes are aggregated) the lower will be the frequency of membership changes for the aggregate. In SPBM we therefore propose to scale down the frequency of membership update messages exponentially with the level of aggregation. This results in a constant upper bound on the overhead as the number of nodes in the network increases.

The remainder of this paper is structured as follows: In the next section, we discuss related work. We describe the SPBM protocol and give analytic properties of the group management scheme in Section III. Section IV contains simulation results on the performance of SPBM as well as the protocol we compare it against, ODMRP. Our implementation of SPBM for Linux is presented in Section V and Section VI concludes the paper and gives an outlook on future work.

II. Related Work

Due to the very large amount of literature and protocol proposals in the area of mobile ad-hoc networks, we limit our discussion to work closely related to SPBM. Related work is divided into two main groups, topology-based ad-hoc multicast protocols (Section II-A) and position-based ad-hoc routing protocols (Section II-B).

A. Topology-based Ad-Hoc Multicast Protocols

Topology-based multicast protocols for mobile ad-hoc networks can be categorized into two main classes: tree-based and mesh-based protocols. The tree-based approaches build a data dissemination tree which contains exactly one path from a source to each destination. For its construction topological information is used. The trees can further be sub-classified into source trees and shared trees. Representatives of the first are ABAM [3], MZR [4], DDM [5], and ADMR [6]. In these protocols, each single source builds its own tree to distribute its packets. In contrast to that, a shared tree is a tree where each connected node is able to send packets to all other nodes using one and the same tree. Shared trees are built among others by LAM [7], AMRoute [8], MAODV [9], and AMRIS [10]. Tree-based approaches often use local repair mechanisms to protect the distribution structure from link failures caused by mobility.

The second main category are mesh-based approaches, building meshes of data paths to make the multicast routes more stable against topological changes. This comes at the expense of a higher overhead during data delivery. A mesh can contain multiple possible paths from a source to a destination. Members of this class are CAMP [11], ODMRP [12], MCEDAR [13], NSMP [14], SRMP [15], and DCMP [16].

In the performance evaluation in Section IV, we compare our protocol to the On-Demand Multicast Routing Protocol (ODMRP) which was has been shown to be a comparatively performant competitor [17]. ODMRP is a mesh-based protocol which can be seen as a successor to FGMP [18]. ODMRP uses soft state information to manage forwarding and multicast group membership. Control packets, which optionally can contain data payload, are periodically flooded through the whole network. The protocol has an extension which allows to exploit position information (if available) for predicting node mobility. A distinctive feature is that the protocol can be also used for unicast routing, thus making an additional unicast protocol unnecessary.

The building of a new multicast mesh is initiated by the source. A node which wants to send data to a multicast group periodically creates join request messages. These are flooded to all nodes within the ad-hoc network in order to advertise a multicast group.

While forwarding such a join request, the nodes keep track of the upstream node from which the first copy of the request was received by using a routing table. When a multicast group member receives a join request, it updates the entry belonging to that source in its member table. As long as a node has entries in its member table, it periodically broadcasts a join table message containing the upstream nodes which were stored in the routing table. A neighbor whose ID is listed in this message, considers itself as member of the forwarding group, adds an entry to its forwarding group table and broadcasts its own join table to the neighbors. This way, the join tables construct the shortest path routes from each member to the multicast source which altogether build a mesh.

To deliver packets, the source broadcasts them to the nodes within its transmission range. The nodes having an entry in their forwarding group table forward this packet by re-broadcasting it to their neighboring nodes.

B. Position-Based Routing

Exploiting knowledge of a node's geographic position for data packet forwarding has first been suggested some time ago [19]. Recently, position-based routing (PBR) has also been investigated for mobile ad-hoc networks and led to several publications, surveys of which can be found in [20], [21], [22].

In position-based routing the forwarding decisions are usually based on the node's own position, the position of the destination, and the position of the node's direct radio neighbors. Since no global distribution structure—such as a route—is required, position-based routing is considered to be very robust to mobility. It typically performs best when the next-hop node can be found in a greedy manner by simply minimizing the remaining distance to the destination. However, there are situations where this strategy leads to a local optimum and no greedy neighbor can be found to further forward the packet, although a route exists. In this case, a so-called recovery strategy is invoked. Among the protocols that utilize greedy forwarding and a recovery strategy are GPSR [2], face-2 [1], and GOAFR+ [23]. In addition to these purely position-based algorithms, there are protocols that are position-aided (e.g., LAR [24]) and make use of position information to improve topology-based routing.

Knowledge about the geographical position of nodes has been used for Dynamic Source Multicast (DSM) [25]. In DSM each node floods the network with information about its own position, thus each node knows the position of all other nodes in the ad-hoc network. The sender of a multicast packet then constructs a multicast tree from the position information of all receivers. This tree is encoded in the header of the packet. While DSM uses location information, the resulting distribution tree is completely determined by the sender. This eliminates the most important advantage of position-based routing. Due to periodic flooding of the network, the scalability of this approach is limited.

In [26], the authors report on "Location-Guided Tree Construction Algorithms" using the position of nodes to build an application-level distribution tree. This approach enjoys the benefits of position-based routing but it is limited to receiver groups small enough so that the address of each destination can be included in each data packet.

A generalization of position-based unicast forwarding has been described in [27]. As for the "Location-Guided Tree Construction Algorithms" the sender includes the addresses of all destinations in the header of a multicast packet. In addition the location of all destinations is included as well. It remains open how the sender is able to obtain the position information and the scaling limitations seem to be similar to those discussed above.

In contrast to the existing position-based multicast protocols, SPBM retains the advantages of position-based routing while not being restricted to small receiver sets.

For position-based routing, the sender of a packet needs to know the position of the destination. The mapping from an ID to the position where the node with this ID is located is called a position service. Several algorithms for positions services have been proposed, such as GLS [28], GRSS [29], Homezone [30], or the location service part of DREAM [31].

Of these, we briefly discuss GLS and GRSS since, as SPBM, they rely on a quad-tree hierarchy to structure the network area. In GLS, each node has multiple location servers, one for each hierarchy level. Each node sends its current position to these location servers. A resolution algorithm enables querying nodes to find the location server on the nearest common hierarchy. In GRSS, position information is aggregated for each square by means of "a node is in square *i*". Since this approach can easily result in very large control packets, the authors suggest to use Bloom Filters or their compressed variant [32] to reduce the size. The paper also describes a unicast packet forwarding strategy. While both GLS and GRSS perform hierarchical aggregation as it is also done in SPBM, they are location services designed to map one node id to a position but they are not designed for multicast group management.

III. The Protocol

We now introduce the two building blocks of our algorithm: the *group management scheme* is responsible for the dissemination of the membership information for multicast groups, so that forwarding nodes know in which direction receivers are located. The *multicast forwarding algorithm* is executed by a forwarding node to determine the neighbors that should receive a copy of a given multicast packet. This decision is based on the information provided by the group management scheme.

A. Group Management

Position-based multicast requires that the forwarding nodes know the locations of the destinations. Including all of the destinations explicitly in the data packet header does not scale well as the size of the multicast group increases. To improve scalability, our proposal introduces hierarchical group membership management.[1]

To this end, the network is subdivided into a quad-tree with a predefined maximum level of aggregation L. Figure 1 shows a quad-tree with four levels. Single squares are identified by their concatenated level-n to level-1 square numbers. In the example the identifier "442" identifies a level-0 square that is located in the level-3 square comprising the whole network, in the level-2 square "4" and in the level-1 square "44". In level-0 squares, all nodes are within radio range of each other (i.e., level-0 squares have at most a diameter of half the radio range).

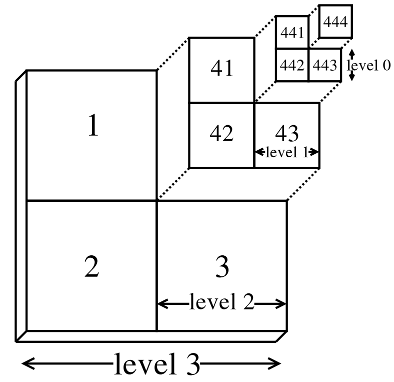


Fig. 1. Network represented by a quad-tree ($L = 3$)

1) Algorithm: The aim of the membership update mechanism is to provide each node in the ad-hoc network with an aggregated view of the position of group members. For this purpose, each node maintains a global member table containing entries for the three neighboring squares for each level from level 0 up to level $(L-1)$. In addition each node has a local member table for nodes located in the same level-0 square.

Each entry in the global member table consists of the square's identifier and the aggregated membership information of all nodes contained in that square. Each entry in the local membership table consists of a node ID and the membership information of that node. Membership information is stored and transmitted as membership vectors where each bit represents one multicast group. A bit set to 1 indicates group membership. Thus the amount of state maintained in a node scales logarithmically with the size of the network. Table I shows an example for a node located in square "442" with a membership vector length of 8. In this example the first entry of the global member table can be interpreted as follows: there is at least one multicast receiver for groups 3, 4 and 5 located in the level-2 square "1". The first entry of the local member table contains the information that node 14 is in the same level-0 square as the node maintaining the table and that 14 is member of group 7.

A node indicates its group membership status by broadcasting *announce* messages within its level-0 square (i.e., its direct neighbors). An announce message contains the ID of the node and a membership vector describing its subscribed groups. Announce messages are broadcast periodically, but need not be forwarded by any other node since all nodes within the same level-0 square are within radio range of each other.

[1] In other contexts the term group management is also used for group address assignment such as in SAP/SDP [33]. Our scheme is not intended to provide address assignment, instead existing approaches should be used.

TABLE I

GLOBAL AND LOCAL MEMBER TABLE OF A NODE LOCATED IN SQUARE "442"

Square	Groups
1	00011100
2	01000100
3	10100010
41	01010000
42	00010101
43	00100100
441	00000100
443	00010000
444	00100100

Node	Groups
14	00000001
23	01000100
51	00000100

A node stores the membership information of all nodes in its level-0 square. Update messages are then used to provide all nodes that are located in a level-1 square with the aggregated membership information of the four level-0 squares contained in the level-1 square. This is done by periodically selecting one node in each level-0 square. For now we assume that such a selection mechanism is in place, we shall show later how it can be realized by means of random timers. The selected node floods the level-1 square with an update message including the ID of the selected node, a membership vector of the aggregated group memberships, the identifier of the destination square that is to be flooded, and a sequence number for duplicate message detection. The aggregation is done by a bitwise or-operation on the membership vectors of the nodes located in the level-0 square. In order to perform flooding, each node in the level-1 square forwards this message once. In total, there will thus be four update messages flooded in each level-1 square per period, one for each level-0 square. In the example, one node in each square "441", square "442", square "443", and square "444" is selected. Those nodes aggregate their level-0 membership information and flood them in an update packet in the level-1 square "44".

The same mechanism is used to aggregate the membership information from an arbitrary level-λ square and flood it in the area of a level-$(\lambda+1)$ square. The aggregation of a level-1 or higher square is done by performing a bitwise or-operation on the membership vectors of those squares and single nodes that are known by the selected node and that are contained in the level-λ square. In the example one node in each square "41", square "42", square "43", and square "44" would be selected to aggregate their level-1 membership information and flood an update message in square "4". If the node with the membership tables depicted in Table I would be selected for square "44", it would perform the aggregation by a bitwise or-operation on the membership vectors for the individual nodes 14, 23, 51 and on the aggregated information from the level-0 squares "441", "443", and "444".

Since the size of a square increases exponentially with each level, the likeliness that the aggregated group membership information changes in a given time-span decreases rapidly. We therefore propose to decrease the frequency of flooding membership information exponentially with the level of aggregation. Let f_0 be the frequency of announce messages. Then the frequency f_λ of update messages from a single square on level λ is defined as follows:

$$f_\lambda = q^\lambda \cdot f_0 \qquad \text{for} \qquad \lambda = 1, \ldots, L \quad \text{and} \quad 0 < q \leq 1$$

It remains to be shown how one node is selected to send an update message. The selection mechanism is performed by random timers. Every node maintains an update timer for each level. When the timer expires the node is selected, transmits the update message for the appropriate level and resets the timer. When a node receives an update message for a square that it belongs to, its timer is reset without sending the packet thus suppressing the transmission of the update message. The main component of each timer is determined by the update frequency of that level. In order to avoid that all nodes in a given square flood the same update information simultaneously, each timer has also a random exponential element. The total runtime of a timer for a given level is chosen as follows:

$$t(x) = \left(\frac{1}{f_0} - r \cdot \log x + \frac{1}{2}\right) \cdot \left(\frac{1}{q}\right)^\lambda$$

with x being a random variable that is uniformly distributed between 0 and 1 and r the maximum difference between the highest and lowest possible timer values on level 0.[2] This behavior is adapted from [34]. Through the exponential distribution, the probability of having a short timeout value is much smaller than the probability of a high timeout value. Thus, the vast majority of timers will not expire before an update message from another node is received. Note that the largest part of the timer is deterministic. The random component used for the selection process has therefore no significant impact on the frequency with which the flooding of squares is performed.

2) Scalability Analysis: The group management algorithm is proactive and thus its overhead is independent of actual data traffic and the number of senders in a given multicast group. In the following, we quantify this overhead to examine the algorithm's performance and scaling characteristics.

Let the radio range r be constant. To ensure connectivity within level-0 squares (under the assumption of a unit disk graph), the size A_0 of level-0 squares is:

$$A_0 \leq \frac{r^2}{2}$$

and the area covered by the network can be determined as:

$$A(L) = A_0 \cdot 4^L.$$

We need to determine how often a level-0 square is flooded with update messages from all levels in a fixed amount of time. In a first step let us consider the case that $q = 1$ and therefore the update frequency is the same for all levels. Then, on level 0, four update messages are generated by four squares which form a level-1 square. These messages are received by each node within the level-1 square. The same holds for each

[2]For different levels, this difference is scaled along with the timer values and thus depends on the area in which the nodes should be suppressed.

level from 1 up to $L-1$. Thus the overhead c is linearly dependent on the number of levels L. If we quadruple the area of the network, thereby increasing the number of levels by one, each single lowest-level square has to be flooded with four more messages. This means that a multiplication of the size of the network area A only stresses a single node with a constant additional load.

Considering the spatial frequency reuse occurring in a network of growing (area) size, we study the overhead per area. In terms of complexity, the total cost c_A per area in the network conforms to

$$\frac{c(A(L))}{A(L)} = O(\log A(L)).$$

More general, if we allow $0 < q \leq 1$ and if n is the number of nodes in the network, then the total cost through update messages in the network c is, depending on the number of levels L,

$$c(L) = n \cdot f_0 \left(1 + 4 \sum_{\lambda=1}^{L} q^{\lambda}\right). \tag{1}$$

For a proof of (1), see appendix.

If $0 < q < 1$, the sum in equation (1) represents a geometric series which has an upper limit for all values of L. Thus, for $q < 1$, the total cost per area within the network is bounded by a small constant number of update messages per time when growing the area of the network:

$$\frac{c(A(L))}{A(L)} = O(1) \qquad \text{if} \quad q < 1.$$

This is also shown in the appendix.

B. Multicast Forwarding

To deliver multicast packets from a source to the subscribed group members, the nodes use the information stored in their member tables. By dividing the network into a quad-tree, geographic regions are build which can be used to aggregate multicast traffic to group members located geographically close to each other.

The forwarding decision is based on information about neighboring nodes. Each node maintains a table of nodes in its transmission range. This is accomplished by having each node periodically broadcast beacon messages containing the ID and position of the node. Beacon messages are not forwarded by the receiving nodes.

Algorithm 1 shows the forwarding algorithm. As an input the algorithm requires the current node n, the packet p and the list of neighbors N of n. The packet includes a list-of-destinations field which is initially set to one entry that comprises the whole network and a group address field indicating the group the packet is sent to. Once the algorithm is invoked, it first checks whether the current node n is a member of the multicast group the packet is sent to. If this is the case, then the packet is delivered.

In the next step the algorithm looks at each entry in the list-of-destinations field of the packet: if the global or the local membership tables contain a de-aggregation of the entry,

```
Require: node n, packet p, list of neighbors N
  if n ∈ receivers(group(p)) then
    deliver(p)
  end if
  D ← ∅
  for all d ∈ destinations(p) do
    if mysquare ⊆ d then
      D ← D ∪ subdivide(d)
    else
      D ← D ∪ d
    end if
  end for
  F[N] ← ∅
  for all d ∈ D do
    ν ← ∅
    if recover(d) then
      ν ← rightHand(prevHop, d)
    else
      ν ← forwardGreedy(N, d)
    end if
    if ν = ∅ then
      ν ← rightHand(n, d)
      if ν = ∅ then
        drop(d)
      end if
    end if
    F[ν] ← F[ν] ∪ d
  end for
  for all ν ∈ N do
    if F[ν] ≠ ∅ then
      send(p, ν, F[ν])
    end if
  end for
```

Algorithm 1. The forwarding algorithm

then the entry is subdivided into those squares of the next lower level that include members for the group the packet is transmitted to. At level-0 a de-aggregation is performed by replacing the square with the ID's of the nodes that are group members.

For example, consider the situation where the node in square "442" (see Figure 1) sends a multicast packet to the group number 1. It initializes the packet with the whole network as the single destination area and sets the multicast address to 1. Then the packet is handed to the forwarding algorithm. After checking whether the current node is a receiver of multicast group 1 the destinations are de-aggregated: based on the membership tables given in Table I for multicast group 1 the complete network can be de-aggregated in the level-2 square "2" (since bit 1 of the membership vector is set), the level-1 square "41", and the individual node 23 in the same level-0 square as the forwarding node.

After de-aggregation of the destinations it is checked which neighbor is best suited to forward the packet to each destina-

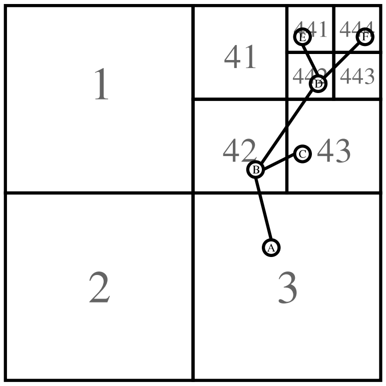


Fig. 2. Forwarding on the quad-tree

tion. This is done in a fashion similar to position-based unicast routing (see [20]): in order to determine the most suitable next hop for a packet and a given destination, the source compares the geographic progress for each of the neighbors in respect to the destination and picks the neighbor with the highest progress. In case that the destination is a square, the position of the nearest point in that square is used as the destination position.

After finding the next hop for each destination, the current node *n* makes a copy of the data packet for each of these next hops. In the list-of-destinations field, it enters a list of the destinations which shall be reached through this specific next hop and sends the packet to the next hop by using unicast transmission. The use of unicast increases the reliability of data delivery at the expense of bandwidth utilization as each copy of the packet will be acknowledged on the MAC layer but has to be sent separately.[3]

Figure 2 shows an example of the forwarding procedure.[4] Node *A* wants to send a packet to the group in which nodes *C*, *E* and *F* are members. Thus *A*'s member table contains the information that there is at least one receiver in square "4". It sends the packet in this direction and node *B* is the first node located in the level-2 square "4". Consequently, it has the information that there are nodes subscribed to the group in the level-1 squares "43" and "44". It therefore updates the information in the packet header accordingly. Node *C* is the first forwarding node in square "43". Besides delivering the packet, it checks its member table and recognizes that it does not need to forward the packet to any additional receivers in square "43". In square "44", node *D* replaces square "44" in the packet header by the level-0 squares "441" and "444". After receiving the packet, nodes *E* and *F* replace their square by potential additional destination nodes in this square. If there were any, the packets would now directly be sent to the receivers since the radio ranges of *E* and *F* cover the complete squares "441" and "444", respectively.

[3]This is a design decision, depending on the application and the environment of the ad-hoc network one may choose to transmit the packet using broadcast.

[4]The figure only depicts nodes which are involved in the process of refining the destination square information.

If, for one or more destinations, a forwarding node does not find a next hop that yields geographic progress, a recovery strategy has to be employed. Similar to position-based unicast routing [2], [1], SPBM uses a distributed planarization of the network graph combined with the right-hand rule to route around void regions. When there is a destination with no suitable next hop, the algorithm first planarizes the surrounding network graph. Then, the node determines the angles counter-clockwise between the line from the node to the destination and the line from the node to the particular neighbor for each remaining neighbor and chooses the neighbor which leads to the smallest angle. This destination is marked as a *recovery destination* and the current position is stored in the packet to inform the following hops about the position where the recovery mechanism started. The chosen next hop is then handled as for normal destinations.

A node which receives a packet containing a recovery destination first checks whether itself is located closer to the destination than the position which is stored in the packet as the recovery starting point. The destination is always known by every node in the network since the recovery mode is only needed for destination *squares*, whose positions are known by definition. In this case, the recovery mark is removed and the destination is dealt with as usual. If this is not the case and the node is located farther away from the destination than the recovery starting point, the node has to continue the recovery process. After performing planarization, it chooses the neighbor with the smallest angle counter-clockwise.

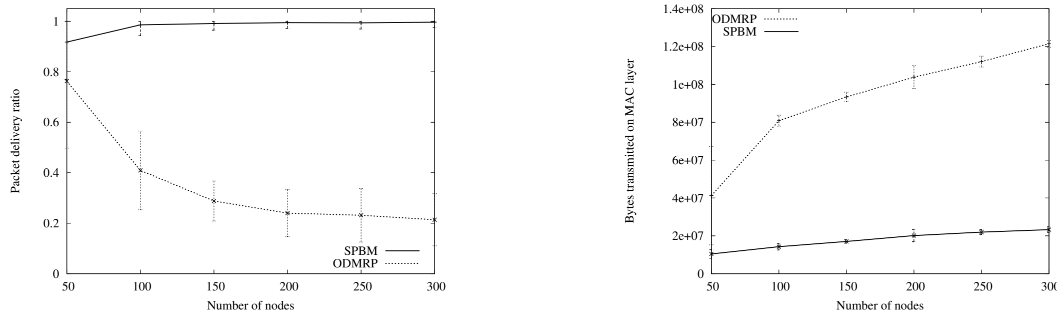
The recovery strategy works independent from the grid structure. As long as a destination is marked as a recovery destination, it is not necessary to change or replace it because only the nodes at the destination have enough information to refine the destination square.

IV. Simulations

A. Simulation Setup

The simulations were performed using the network simulator *ns*-2 [35]. As a reference, the ODMRP implementation from [36] was chosen. The MAC layer in all simulations was IEEE 802.11 with a maximum bandwidth of 2 MBit/s and the transmission power resulted in a radio range of 250 meters. Since the transmitted packets were relatively small, the use of RTS/CTS was disabled. The modeled scenario was a square of 1000 meters by 1000 meters, where 100 to 300 randomly placed nodes moved according to the random waypoint model [37] with a pause time of 0 and a minimum speed of 1 meter per second. The data payload had a size of 64 bytes per packet and each source transmitted one packet per second. All runs were simulated five times with different random seed values and movement scenarios and results were averaged over the runs. A run represented the simulated time of 300 seconds where nodes joined at the beginning of the simulation and the first data packet was sent after 60 seconds in order to give the group management enough time to initialize.

Some simulation parameters were varied to investigate their influence on the results. During each series of simulation runs,



(a) Packet delivery ratio (b) Bytes transmitted on MAC layer

Fig. 3. Performance w.r.t. node density (2 senders, 10 receivers, 10 m/s, 1 Pkt/s)

only one parameter was changed leaving the others constant. The number of nodes was increased from 50 to 300 with an increase of 50 nodes per step. The number of senders ranged from 1 to 10, all senders and receivers did belong to one multicast group, but senders and receivers were disjoint. Mobility was varied from 0 to 20 meters per second.

The protocol specific parameters of SPBM were set as follows: the beacon interval was 2 seconds and a neighbor expired after 1.5 beacon intervals or 3 seconds, respectively. The basic update frequency f_0 was $\frac{1}{6\,s}$ and the basic span r for the exponential part of the timer was set to 2 seconds. Because the width of the smallest square is 125 meters and the maximum speed in the simulations is 20m/s, a node needs on average at least 6 seconds to cross such a square. Therefore, the chosen update period is reasonable. The timeout for entries in the member table amounted to 2.5 times the corresponding update interval. The number of levels was set to 4 as in the example depicted in Figure 1 (i.e., $L = 3$). ODMRP's protocol specific parameters were: a join refresh interval of 3 seconds, an acknowledgment timeout for join table messages of 25 milliseconds, and a maximum number of join table transmissions of 3.

To improve comparability, all these protocol specific parameters were kept constant throughout all simulations.

B. Performance Metrics

The metrics used to evaluate the protocol performance are packet delivery ratio and overhead. The *packet delivery ratio* (PDR) is defined as the sum of all data packets received over the sum of all data packets that should have been delivered (sum of sent packets multiplied by the number of receivers).

The *overhead* is the total number of bytes transmitted at the MAC layer, including acknowledgments in case of unicast transmissions. To measure the overhead on the MAC layer it is necessary to capture MAC layer retries induced by mobility or packet collisions. These effects would be invisible if the overhead was counted on the network layer.

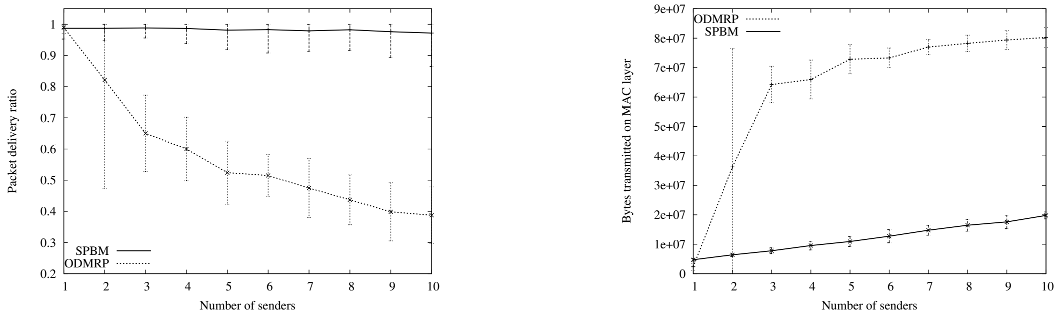
For simulation scenarios with dynamically growing receiver groups, the *average join latency* is given as the time difference between the join request of a node and the first packet reception of the corresponding multicast group averaged over all receiving nodes.

C. Results

1) Node Density: Figure 3 shows the performance of SPBM and ODMRP with respect to an increasing node density on a simulated area of $1000m \times 1000m$, with 2 senders, 10 receivers, random way-point mobility with a maximum speed of 10 m/s, and a packet sending rate of 1 pkt/s. As can be seen in the graph, SPBM performs well in terms of PDR with a linear increase in overhead. Even for the 50 nodes case, where position-based routing suffers slightly from the lack of greedy forwarders, SPBM achieves a higher PDR than ODMRP, although ODMRP generates almost four times the overhead. As can be seen from the graph, the flooding used in ODMRP significantly increases network load when node density increases. This additional overhead causes the PDR to diminish further due to packet collisions. We chose to use 2 senders for these simulation runs, as ODMRP behaved less predictable and with a significantly worse performance when increasing the number of senders.

2) Number of Senders: The next figure (Figure 4) shows the respective PDR and overhead when the number of senders increases. The other parameters were kept constant in this setup. ODMRP faces similar problems as the ones described above. With only three senders it reaches a saturation of the network (on average 2.8 MBit/s), resulting in a high number of collisions. Thus, the PDR reduction is—even at the low sending rate of 1 Pkt/s—mainly due to the unacceptably high overhead. As in Figure 3, a high increase in load is accompanied by a high decrease in the ratio of delivered packets.

SPBM, in contrast, sustains a satisfactory packet delivery ratio. The increase in overhead is mainly due to the increased number of data forwarding operations for the data packets of the additional senders. The proactive group management overhead of SPBM remains constant, while the number of neighborhood beacons decreases. This is caused by the use of implicit beaconing where beacon information is prepended to data packets whenever possible.



(a) Packet delivery ratio (b) Bytes transmitted on MAC layer

Fig. 4. Performance w.r.t. number of senders (10 receivers, 1 Pkt/s, 10 m/s, 100 nodes)

A similar result was achieved when varying the number of receivers while keeping the number of senders constant. In this case, ODMRP quickly saturates the network resulting in a constantly high network load, while SPBM still operates with a satisfactory packet delivery ratio with a load increase mainly caused by the higher number of forwarding operations.

3) Node Mobility: An important aspect of MANET routing protocols is their behavior in the presence of node mobility. In 100 node scenarios with 2 senders and 10 receivers (depicted in Figure 5), both SPBM *and* ODMRP suffer from increasing node mobility.

For SPBM, this is on one hand due to nodes crossing square “boundaries” and on the other hand due to forwarding failure induced by discrepancies in the neighbor table used for next-hop selection. If a node is selected as a forwarder but moved out of radio range, the current forwarder has to wait for a link layer notification before it is able to select a different node[5]. For the link layer to decide that a next-hop is not reachable, four unsuccessful retries are necessary, resulting in higher network load. Since the nodes are moving, the number of forwarding group members, which rebroadcast data packets, grows. Thus, new forwarding nodes are selected each time and these forward all data traffic until their forwarding group timer expires. As in all other simulations, ODMRP’s increase in network load is accompanied by a decrease in the ratio of delivered packets.

Of course, the problems SPBM faces can be alleviated by a different setup in beacon and group management message intervals. Figure 5(b) also shows the proactive part of the protocol overhead. As can be seen in the graph, even for our low-data-rate traffic this part is strongly dominated by the data forwarding. In a network with constant high mobility one would probably accept a higher proactive load to lower traffic induced overhead.

4) Join Latency: Apart from the delivery performance, there is a trade-off between overhead and join latency. Figure 6 shows the average join latencies for one sender and different numbers of receivers joining at uniformly distributed points of time. In contrast to the previous simulation runs, the members join *after* the source has started to send data. As before, the receivers remain in the group until the end of the simulated time. While ODMRP reacts rather quickly by extending the forwarding group, SPBM has to distribute the membership information. Since there is no reactive triggering of this distribution mechanism, this simply means to wait for expiring timers. However, it is sufficient to propagate the new membership to a level that is already receiving packets of the multicast group. Thus, the average join latency decreases with an increasing number of receivers. Consequently, SPBM is more suitable for long-lived multicast session, where a longer join time is acceptable.

V. Linux Implementation

In order to perform experiments with a real system, we implemented SPBM as a Linux kernel module. To receive incoming packets, the module registers a new layer-3 protocol defining a protocol number for SPBM. Every incoming packet containing the proper protocol number in its protocol field is directly delivered to the SPBM module.

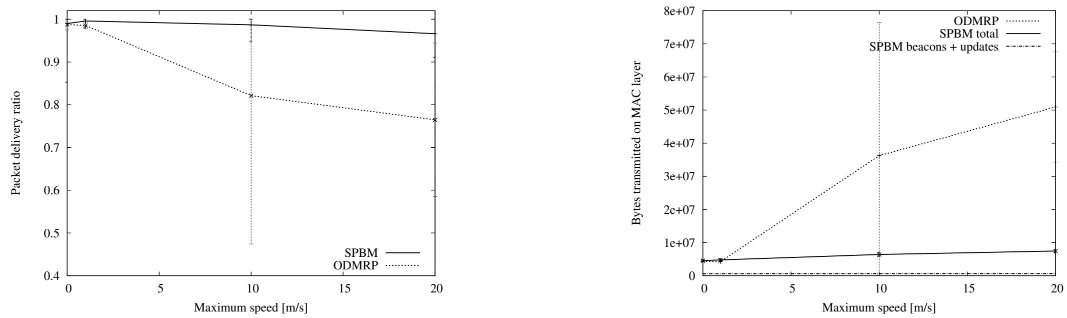
Outgoing packets generated at the local host are captured via the netfilter interface at the `NF_IP_LOCAL_OUT` hook (see [39]) in order to analyze their destination. If a packet is addressed to a multicast group, it is directed to the SPBM module.

There are three subtypes of SPBM packets: beacons, update messages and data packets. If a node receiving an SPBM data packet is a member of the destination group, the module injects the packet back into the protocol stack as if it was received directly from the network interface.

The module uses the proc interface of the kernel to communicate with programs in the user space. Within the directory `/proc/spbm`, there are different virtual files through which the user or program can control the behavior of the module. Table II lists these files and their function.

Within the module, a virtual coordinate system is used. It extends to 16 Bits in *x*-direction and 16 Bits in *y*-direction. The current position has to be fed to the module via the `/proc/spbm/pos` file, as a string of two space-separated 16-Bit hexadecimal values.

[5]This effect has been extensively described in [38].



(a) Packet delivery ratio (b) Bytes transmitted on MAC layer

Fig. 5. Performance w.r.t. maximum movement speed (2 senders, 1 Pkt/s, 10 receivers, 100 nodes)

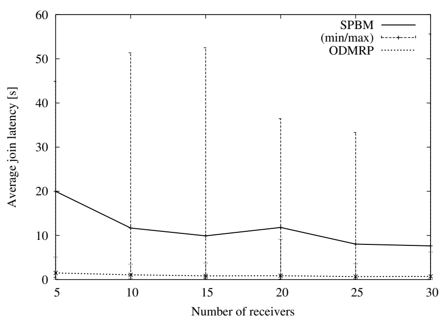


Fig. 6. Join latency (1 sender, 1 Pkt/s, 100 nodes)

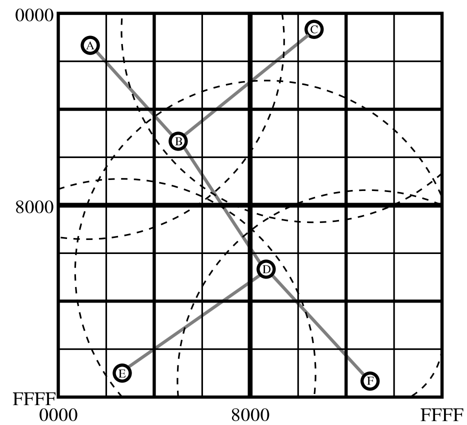


Fig. 7. Setup for the real world test

The mapping from real to virtual coordinates is done by a user space positioning daemon. This gives a high grade on flexibility regarding the used positioning system. The daemons on each node have to be configured to provide a consistent coordinate system. E.g., the GPS coordinates have to be mapped to identical virtual coordinates on every node.

The SPBM implementation has been installed and tested both on laptop computers and iPaq hand-held computers. It is available for download from our website at *anonymized for review*. For a meaningful study of multicast, tens or ideally hundreds of nodes are required to avoid that multicast degenerates to network wide flooding. While real-world experiments are crucial [40], [41] and we intend to conduct such experiments with large numbers of nodes in the near future, our current setup is more intended to analyze feasibility than to do actual performance measurements.

First preliminary tests with a simple setup of six nodes have already been completed. Their goal was to validate the implementation and get a first understanding of the potential performance of the forwarding algorithm. For the experiment the nodes were "virtually" located as depicted in Figure 7. In order to enable reproducible experiments the physical location of the nodes was directly next to each other with the topology being enforced by filtering packets from nodes with a virtual position beyond the transmission range as depicted by circles in Figure 7. This set-up leads to an increase in the congestion level of the network since all nodes are in interference range of each other.

TABLE II
ENTRIES IN THE /PROC/SPBM DIRECTORY

Entry	Read from file	Write to file
pos	get position	set position
join	get subscribed groups	join a group
leave	–	leave a group
neighbortable	get current neighbors	–
membertable	get current member table	–

During each experiment we transmitted packets from node A to a multicast group that was joined by all other nodes. Group membership management, beaconing, and data forwarding was performed according to the SPBM algorithms as defined above. The sending rate of node A was limited only by the rate accepted by the MAC of node A, the size of the data payload was set to 1000 bytes, IEEE802.11 was set to 11 MBit/s, thus about 2.2 MBit/s gross for each link in Figure 7. The experiment was conducted 10 times. As a result all nodes B through F, which were iPaq 3660 devices, on average received data with the rate of 408 kBit/s, while no packet loss occurred. The latter was to be expected since there was no node mobility and all transmissions of data packets were performed using unicast and MAC-level retransmissions. It is assumed that the bottleneck in these experiments is the CPU power of the iPaq hand-held devices. This assumption has to be further investigated by means of extensive performance analyzes which we plan for the near future.

VI. Conclusions and Future Work

In this paper we described a novel ad-hoc multicast routing protocol. It differs significantly from previous work in that it introduces a hierarchical organization of nodes for membership management as well as packet forwarding, similar in spirit to hierarchical location services proposed in [28], [29]. We show through simulation, that in terms of overhead as well as packet delivery ratio our protocol performs orders of magnitude better than ODMRP, one of the most performant multicast protocol described in the literature.

Only without movement and for very low numbers of senders and receivers, the performance of both protocols is comparable. In particular, ODMRP packet delivery ratios frequently drop to values around 50% as soon as there is more than one multicast sender, while SPBM maintains a delivery rate around 95%.

However, this increase in performance comes at the expense of a higher join latency with our group management. This increase is caused by the hierarchical organization of nodes and the timers chosen for dissemination of group management updates on the different levels of the hierarchy.

Through our simulations as well as the real-world implementation for Linux we have shown that our approach is feasible. The parameters that were chosen for the implementation are very conservative and can be tuned for improved performance depending on the environment. We plan to investigate this in more detail in the future. Furthermore, our kernel implementation of SPBM has only been tested in a very small experimental environment. We plan to investigate its scalability under more realistic settings with a much larger number of participating nodes.

As described in Section IV-C, the forwarding strategy slightly suffers from higher mobility due to increasing inaccuracy of the neighbor tables. While this effect could be reduced by increasing the neighborhood beacon frequency, there is a new position-based unicast forwarding proposal called CBF [38] eluding this problem by obsoleting the neighbor tables. An application of this forwarding scheme for multicast will also be a subject of future work.

To summarize, we believe that a hierarchical approach to multicast is a very promising solution if the protocol is intended to scale to a reasonable number of nodes. While for some scenarios where nodes frequently join and leave at short intervals the increase in join latency can be problematic, this is easily compensated by the very desirable properties of our protocol in terms of scalability and protocol overhead.

Appendix

Theorem 1 (Cost Function): Consider an ad-hoc network of square geometry and n the number of nodes. Let L be the maximum hierarchy level, $0 < q \leq 1$ be the timer frequency coefficient, and f_0 the smallest-square frequency. Then the average number of (proactive) radio transmissions per time of the SPBM group management protocol is given as

$$\begin{aligned} c &= n f_0 \left(1 + 4 \sum_{\lambda=1}^{L} q^{\lambda}\right) \qquad (2) \\ &= \begin{cases} n f_0 \, (1 + 4L) & q = 1 \\ n f_0 \left(1 + 4 \left(\frac{1-q^L}{1-q}\right)\right) & 0 < q < 1. \end{cases} \end{aligned}$$

Proof: Be c_λ the average number of transmissions per second on level λ $(\lambda = 0, \ldots, L)$ for the whole network. On level 0, each node sends f_0 packets per second. Thus

$$c_0 = n f_0.$$

At every higher level λ $(\lambda = 1, \ldots, L)$ $4^{L-\lambda}$ squares exist, each with $\frac{n}{4^{L-\lambda}}$ nodes on average. With a frequency of f_λ, one of the nodes of each square at level λ sends update packets. Each of these packets are relayed by all nodes in the 4 adjacent squares of level λ which belong to the same square of level $\lambda + 1$. This induces $\left(4 \frac{n}{4^{L-\lambda}}\right)$ packet transmissions for each square of level λ:

$$c_\lambda = 4^{L-\lambda} \cdot 4 \cdot \frac{n}{4^{L-\lambda}} \cdot f_\lambda = 4 \cdot n \cdot f_\lambda \qquad (\lambda = 1, \ldots, L)$$

Aggregating the cost on all levels, we have

$$\begin{aligned} c &= c_0 + \sum_{\lambda=1}^{L} c_\lambda \\ &= n f_0 + \sum_{\lambda=1}^{L} 4 n f_\lambda \\ &= n \left(f_0 + 4 \sum_{\lambda=1}^{L} f_\lambda\right). \end{aligned}$$

Incorporating the definition of the frequencies $f_\lambda = q^\lambda f_0$ gives

$$c = n f_0 \left(1 + 4 \sum_{\lambda=1}^{L} q^{\lambda}\right)$$

directly leading to the theorem. ■

Corollary 1 (Cost Complexity): With the definitions of Theorem 1, the SPBM group management protocol overhead per area and time has a complexity of $O(1)$ for $q < 1$ and $O(\log A)$ for $q = 1$ with respect to the total size of the network area A.

Proof: Let us assume that we have a network consisting only of one square of size A_0. We further assume to have a limited node density d denoting the number of nodes per A_0 area. The number of nodes in the complete network is then given as $n = dA$, where A is a multiple of A_0. Whenever the network area increases, we quadruple the network area by increasing the hierarchy level by one, such that the new area is covered by the new square, i.e., the number of hierarchies is calculated as

$$L(A) = \lceil \log_4 A \rceil < 1 + A_0 \log_4 A. \tag{3}$$

With the increase of the area, the possibility of spatial frequency reuse grows linearly. Thus, we consider the cost per area c_A. Following Equation (2), the average overhead cost per time and area is

$$\begin{aligned} c_A &= \frac{c}{A} \\ &= d\, f_0 \left(1 + 4 \sum_{\lambda=1}^{L} q^{\lambda}\right). \end{aligned}$$

With Equation (3), an upper bound $\overline{c_A}$ for the cost per area can be specified:

$$\overline{c_A} = d\, f_0 \left(1 + 4 \sum_{\lambda=1}^{1+\log_4 A} q^{\lambda}\right)$$

Considering the case $q = 1$, this upper bound results in

$$\overline{c_A} = d\, f_0 \,(5 + 4 \log_4 A)$$

which conforms to $O(\log A)$.

With $0 < q < 1$, the geometric row converges and is bounded:

$$\begin{aligned} c_A &= d\, f_0 \left(1 + 4 \sum_{\lambda=1}^{L} q^{\lambda}\right) \\ &< d\, f_0 \left(1 + 4 \frac{q}{1-q}\right), \end{aligned}$$

which is independent of the chosen area size or the maximum level, respectively. Thus, the complexity is $O(1)$. ∎

References

[1] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc Wireless Networks," in *Proc. of ACM DIAL-M '99*, Seattle, WS, August 1999, pp. 48–55.

[2] B. N. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," in *Proc. of ACM MobiCom '00*, Boston, Massachusetts, August 2000, pp. 243–254.

[3] C.-K. Toh, G. Guichal, and S. Bunchua, "ABAM: On-Demand Associativity-Based Multicast Routing for Ad Hoc Mobile Networks," in *Proc. of IEEE VTC Fall '00*, Boston, MA, September 2000, pp. 987–993.

[4] V. Devarapalli and D. Sidhu, "MZR: A Multicast Protocol for Mobile Ad Hoc Networks," in *Proc. of IEEE ICC '01*, vol. 3, Helsinki, Finland, June 2001, pp. 886–891.

[5] L. Ji and M. S. Corson, "Differential Destination Multicast—A MANET Multicast Routing Protocol for Small Groups," in *Proc. of IEEE Infocom '01*, Anchorage, Alaska, April 2001, pp. 1192–1202.

[6] J. G. Jetcheva and D. B. Johnson, "Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks," in *Proc. of ACM MobiHoc '01*, Long Beach, California, October 2001, pp. 33–44.

[7] L. Ji and M. S. Corson, "A Lightweight Adaptive Multicast Algorithm," in *Proc. of IEEE Globecom '98*, Sydney, Australia, November 1998, pp. 1036–1042.

[8] M. Liu, R. R. Talpade, A. McAuley, and E. Bommaiah, "AMRoute: Adhoc Multicast Routing Protocol," CSHCN, University of Maryland, Tech. Rep., 1999.

[9] E. M. Royer and C. E. Perkins, "Multicast Operation of the Ad-hoc On-Demand Distance Vector Routing Protocol," in *Proc. of ACM MobiCom '99*, Seattle, Washington, August 1999, pp. 207–218.

[10] C. W. Wu and Y. C. Tay, "AMRIS: A Multicast Protocol for Ad Hoc Wireless Networks," in *Proc. of IEEE MILCOM '99*, Atlantic City, NJ, November 1999, pp. 25–29.

[11] J. J. Garcia-Luna-Aceves and E. L. Madruga, "A Multicast Routing Protocol for Ad-Hoc Networks," in *Proc. of IEEE Infocom '99*, New York City, New York, March 1999, pp. 784–792.

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

[13] P. Sinha, R. Sivakumar, and V. Bharghavan, "MCEDAR: Multicast Core-Extraction Distributed Ad hoc Routing," in *Proc. of IEEE WCNC '99*, New Orleans, LA, September 1999, pp. 1313–1317.

[14] S. Lee and C. Kim, "Neighbor Supporting Ad hoc Multicast Routing Protocol," in *MobiHoc '00*, Boston, Massachusetts, August 2000, pp. 37–44.

[15] H. Moustafa and H. Labiod, "SRMP: A Mesh-based Protocol for Multicast Communication in ad hoc networks," in *World Wireless Congress 2002*, San Francisco, CA, May 2002, pp. 43–48.

[16] S. K. Das, B. S. Manoj, and C. S. R. Murthy, "A Dynamic Core Based Multicast Routing Protocol for Ad hoc Wireless Networks," in *Proc. of ACM MobiHoc '02*, Lausanne, Switzerland, June 2002, pp. 24–35.

[17] S.-J. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia, "A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols," in *Proc. of IEEE Infocom '00*, Tel Aviv, Israel, March 2000, pp. 565–574.

[18] C.-C. Chiang, M. Gerla, and L. Zhang, "Forwarding Group Multicast Protocol (FGMP) for Multihop, Mobile Wireless Networks," *ACM-Baltzer Journal of Cluster Computing: Special Issue on Mobile Computing*, vol. 1, no. 2, pp. 187–196, December 1998.

[19] G. G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," ISI, Tech. Rep. ISI/RR-87-180, March 1987.

[20] M. Mauve, J. Widmer, and H. Hartenstein, "A Survey on Position-Based Routing in Mobile Ad-Hoc Networks," *IEEE Network*, vol. 15, no. 6, pp. 30–39, November/December 2001.

[21] S. Giordano, I. Stojmenovic, and L. Blažević, "Position-Based Routing Algorithms for AdHoc Networks: A Taxonomy," in *Ad Hoc Wireless Networking*, D.-Z. Du, Ed. Boston: Kluwer Academic Publishers, November 2003.

[22] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric Ad-Hoc Routing: Of Theory and Practice," in *Proc. of ACM PODC '03*, Boston, MA, July 2003, pp. 63–72.

[23] F. Kuhn, R. Wattenhofer, and A. Zollinger, "Worst-Case optimal and average-case efficient geometric ad-hoc routing," in *Proc. of ACM MobiHoc '03*, Annapolis, Maryland, June 2003, pp. 267–278.

[24] Y.-B. Ko and N. H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," in *Proc. of ACM MobiCom '98*, Dallas, Texas, October 1998, pp. 66–75.

[25] S. Basagni, I. Chlamtac, and V. R. Syrotiuk, "Location aware, dependable multicast for mobile ad hoc networks," *Computer Networks*, vol. 36, no. 5–6, pp. 659–670, August 2001.

[26] K. Chen and K. Nahrstedt, "Effective Location-Guided Tree Construction Algorithms for Small Group Multicast in MANET," in *Proc. of IEEE Infocom '02*, New York City, New York, June 2002, pp. 1192–1201.

[27] M. Mauve, H. Füßler, J. Widmer, and T. Lang, "Position-Based Multicast Routing for Mobile Ad-Hoc Networks," Department of Computer Science, University of Mannheim, Tech. Rep. TR-03-004, 2003.

[28] J. Li, J. Jannotti, D. S. J. DeCouto, D. R. Karger, and R. Morris, "A Scalable Location Service for Geographic Ad Hoc Routing," in *Proc. of ACM MobiCom '00*, Boston, Massachusetts, August 2000, pp. 120–130.

[29] P.-H. Hsiao, "Geographical Region Summary Service for Geographical Routing," *ACM Sigmobile Mobile Computing and Communications Review (MC2R)*, vol. 5, no. 4, pp. 25–39, October 2001.

[30] L. Blažević, S. Giordano, and J.-Y. LeBoudec, "Self-Organizing Wide-Area Routing," in *Proc. of SCI 2000/ ISAS 2000*, Orlando, July 2000.

[31] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, "A Distance Routing Effect Algorithm for Mobility (DREAM)," in *Proc. of ACM MobiCom '98*, Dallas, Texas, October 1998, pp. 76–84.

[32] M. Mitzenmacher, "Compressed Bloom Filters," in *Proc. of ACM PODC*, Newport, Rhode Island, USA, August 2001, pp. 144–150.

[33] M. Handley and V. Jacobson, "RFC 2327: SDP: Session description protocol," Apr. 1998.

[34] J. Nonnenmacher and E. W. Biersack, "Scalable Feedback for Large Groups," *IEEE/ACM Transactions on Networking (TON)*, vol. 7, no. 3, pp. 375–386, 1999.

[35] "The ns-2 network simulator," http://www.isi.edu/nsnam/ns/.
[36] "Wireless multicast extensions for ns-2.1b8."
[37] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in *Mobile Computing*, T. Imielinski and H. Korth, Eds. Kluwer Academic Publishers, 1996, vol. 353, pp. 153–181.
[38] H. Füßler, J. Widmer, M. Käsemann, M. Mauve, and H. Hartenstein, "Contention-Based Forwarding for Mobile Ad-Hoc Networks," *Elsevier's Ad Hoc Networks*, vol. 1, no. 4, pp. 351–369, 2003.
[39] J. Crowcroft and I. Phillips, Eds., *TCP/IP and Linux Protocol Implementation*. John Wiley & Sons, 2002.
[40] D. S. J. DeCouto, D. Aguayo, B. A. Chambers, and R. Morris, "Performance of multihop wireless networks: shortest path is not enough," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 82–88, 2003.
[41] C. Tschudin, H. Lundgren, and E. Nordström, "Embedding MANETs in the Real World," in *Proc. of PWC '03*, Venice, Italy, September 2003, pp. 578–589.