IRACST – International Journal of Computer Networks and Wireless Communications (IJCNWC), ISSN: 2250-3501
Vol.3, No6, December 2013

Path Diversity in Stateless Multicast Protocol for Adhoc Networks

Dr. K.Karuppasamy#1, Ms.L.Daisy Catherine*2, Mr.K.S Prabhu#3

1 Professor, 2 ,3 Faculty of Information Technology
Department of Information Technology
R.V.S College of Engineering and Technology, Coimbatore
Email: kps_cse@yahoo.co.in,cathy.fastrack@gmail.com,prabhuks4u@gmail.com

Abstract— In this paper we have developed a Stateless receiver based multicast protocol that uses a list of multicast members 'addresses embedded in packet headers so that individual nodes do not have to maintain state information hence enabling receivers to decide the most efficient way to forward the multicast traffic. Multicast routing protocols depend on a priori creation where the state information is maintained by the individual nodes in the network. The maintenance of state information adds a lot of memory overhead and adds a large amount of communication between the nodes in dynamic networks with bursty traffic. The Receiver-Based Multicast uses the geographic locations of the nodes so that the need for costly state maintenance is removed and making it suitable for dynamic networks. A path diversity scheme is implemented for the networks where there two channels between two nodes and when the first channel process is a failure channel one will send information to channel two and hence data transmission will occur through channel two and hence the overall delay is reduced.

Keywords— Path Diversity, RBMulticast

I. INTRODUCTION

There are several applications that require data delivery to multiple destination nodes to manage and reduce traffic. These networks are usually dynamic where the nodes are mobile and providing robust multicast routing in these networks is challenging. In some of the wireless multicast applications the source and intermediate nodes are mobile and the multicast recipients and the destination nodes are fixed and known priori by the nodes in the network. In other multicast applications, all nodes and the destinations are also mobile. Thus the source nodes must know the locations of the multicast destination nodes in order to support any type of multicast applications. Thus a service discovery protocol can be used which is independent of the routing protocol which updates the source with the current locations of sink nodes.

RBMulticast a stateless cross-layer multicast protocol where packet routing, splitting packets into multiple routes and the access of individual nodes rely on the location of destination nodes[1]. This includes a list of multicast members' locations in the packet header which prevents the overhead of building a multicast tree at the intermediate nodes because all the information is included in the packet header which and thus does not require any state information or any priori operations. Thus it does not require any tree creation or neighbour maintenance and it is suitable for dynamic networks. The receiver-based protocol, which means the potential receivers decide the relay node of a packet transmission and thus do not require routing tables and uses the current spatiotemporal neighbourhood. In RBMulticast the multicast routing uses the concept of "virtual node" and "multicast region" for forwarding packets to the nodes closer to the destination multicast members and determine when packets should be split into separate routes. The total number of hops to reach the destination is an important performance factor as it provides an indication of bandwidth usage and of the energy efficiency of the protocol.

RBMulticast is a receiver-based protocol, which means that the relay node of a packet transmission is decided by the potential receivers of the packet in a distributed manner. This routing approach does not require routing tables and enables the use of the current spatiotemporal neighbourhood; this can be compared to proactive and reactive routing protocols where the route is decided using the latest available information, which can be stale. This is a crucial property, especially for dynamic networks. In RBMulticast, receivers contend for the channel based on their potential contribution toward forwarding the packet, which is inspired by the cross-layer protocol XLM [2], a receiver based unicast protocol designed for wireless sensor networks.

Nodes that make the most forward progress to the destination will contend earlier and hence have a higher chance to become the next-hop node. In RBMulticast, the multicast routing uses the concepts of "virtual node" and "multicast region" for forwarding packets closer to the destination multicast members and determining when packets should be split into separate routes to finally reach the multicast members. The total number of hops that packets travel to reach their destination is an important performance metric for routing protocols, as it provides an indication of bandwidth usage and of the energy efficiency of the protocol. In this paper, we derive a mathematical model for the lower and upper bounds on average hop count realized by RBMulticast given the network parameters: target area, node density, duty cycle of the nodes, number of multicast members, and the communication range.

RBMulticast is lightweight and robust, making it ideally suited for multicast applications in ad hoc networks such as WSNs and mobile ad hoc networks (MANETs). RBMulticast differs from previous location-based approaches in that it is completely stateless and hence no costly state maintenance is required. The state maintenance of conventional multicast protocols requires extra traffic to keep the state information up to date, as well as requiring processing of the state information communicated and storage of this state information. On the other hand, in RBMulticast, only the node's own location and the location of the multicast members are needed for multicast packet routing.

II. RELATED WORK

Existing multicast protocols for WSNs and MANETs generally use a tree to connect the multicast members [4], [5],[6], [7], [8], [9]. For example, the Takahashi-Matsuyama heuristic can be used to incrementally build a Steiner tree for multicast routing [10], [11]. Additionally, multicast algorithms rely on routing tables maintained at intermediate nodes for building and maintaining the multicast tree [12], [13]. In location-based approaches to multicast routing [14], [15], [16], nodes obtain location information by default as an application requirement (e.g., a home fire detection sensor would know where it is located) or as provided by a system module (e.g., GPS or a location-finding service). If location information is known, multicast routing is possible based solely on location information without building any external tree structure. For example, PBM [17] weights the number of next-hop neighbour nodes and total geographic distance from the current node to all destination nodes and compares this to a predefined threshold to decide whether or not the packet should be split. PBM is a generalization of Greedy-Face-Greedy (GFG) [18] routing to operate over multiple destinations. GMR [19] selects neighbours based on a cost over progress framework integrated with greedy neighbour selection. Geocast [20] delivers multicast packets by restricted flooding. Nodes forward multicast packets only if they are in the Forwarding Zone calculated at runtime from global knowledge of location information.

RBMulticast differs from previous location-based approaches in that it is completely stateless and hence no costly state maintenance is required. The state maintenance of conventional multicast protocols requires extra traffic to keep the state information up to date, as well as requiring processing of the state information communicated and storage of this state information. On the other hand, in RBMulticast, only the node's own location and the location of the multicast members are needed for multicast packet routing. Receiver-based communication is an opportunistic way of thinking about protocol design in that decisions are not required to be made at the sender side but instead are made at the receiver side. For example, a source node in ExOR [21] broadcasts packets that include a potential forwarders' list inside the header, and these potential forwarders contend to forward the packet through the use of different backoff times, which depend on the network distance to the destination. A source node in XLM [2] broadcasts packets with the destination's geographic location in the header, and every receiver contends to forward the packet through the use of different backoff times, which depend on the geographic distance to the destination. SOAR [22] uses the same idea, but in addition supports multiple paths for strategically selecting relay nodes. In other words, in receiver-based routing, decision making is deferred to the possible receivers, who make decisions in a distributed manner.

III. PROTOCOL DESCRIPTION

RBMulticast is a receiver-based cross-layer protocol that performs multicast routing based on receiver-based geographic unicast protocols such as XLM [2]. The receiver based unicast only needs the sender node's location and the final destination node's location, which are provided in the MAC packet, to decide the next hop along the route. We assume that the "void" (hole) problem in geographic routing is solved implicitly, for example, using the right handed rule as in GPSR [23]. We assume that the multicast members are stationary, such as multiple stationary sinks in WSNs or stationary roadside access points in vehicular ad hoc networks. The intermediate nodes can be either static or mobile. Although mobile intermediate nodes result in route breaks in conventional multicast protocols, since no multicast tree or mesh is used in RBMulticast, mobile intermediate nodes are supported at no additional cost in RBMulticast.

IV. RBMULTICAST OVERVIEW

Nodes in RBMulticast create what we call "multicast regions" centered around themselves. There are several ways to create these regions. However, we use a quadrants approach due to its simplicity and good performance, where each multicast region corresponds to one quadrant of the network, for a grid centred at the node. When a user initiates a request to send (RTS) a packet to a multicast group, data are passed down to the RBMulticast module in the protocol stack. Once the RBMulticast module gets this packet, it retrieves the group list from its group table, assigns the group nodes to the multicast regions based on their locations, and using these locations, calculates a "virtual node" location for each multicast region. RBMulticast replicates the packet for each multicast region that contains one or more multicast members and appends a header consisting of a list of destination nodes (multicast members) in that region, Time to Live (TTL) value, and a checksum value. The destination of a replicated packet is the "virtual node" of the corresponding multicast region, which can be determined in several ways as the geometric mean of the locations of all the multicast members in that multicast region.

In the end, all packets for all multicast regions are inserted in the MAC queue, and are then broadcasted to the neighbourhood. The node closest to the virtual node (within the available relay nodes as determined by receiver-based contention at the MAC layer) will take responsibility for forwarding the packet. The procedure for transmitting packets is summarized in pseudocode in Algorithm 1.

```
Algorithm 1.  RBMulticast Send
Require: Packet output from upper layer
Ensure: Packets inserted to MAC queue
Get group list N from group table
for node n in group list N do
    for multicast region r in 4 quadrants regions R do
        if n 2 r then
            Add n into r:list
        end if
    end for
end for
for r 2 R do
    if r:list is non-empty then
        Duplicate a new packet p
        Add RBMulticast header (TTL, checksum,
        r.list) to p
        Insert p to MAC queue
    end if
end for
```

When a node receives a multicast packet, RBMulticast first examines the checksum in the packet header, and drops the packet if any corruption exists in the packet. It also drops the packet if it is not in the forwarding zone. The forwarding zone is the area within the radio range of the sender that has a smaller distance to the destination than the sender-destination distance.

After a node receives a multicast packet, it then retrieves the destination node list from the RBMulticast packet header. If this node is inside the destination list, it removes itself from the list and passes a copy of the packet to the upper layers in the protocol stack. RBMulticast then checks the TTL value and drops the packet if the TTL is lower than 0. Finally, if there still remain nodes in the destination list, multicast regions and virtual nodes are recalculated, and new packets are generated if required. The packets (one per multicast region that contains multicast members) are then inserted in the MAC queue for transmission. The procedure executed after receiving packets is summarized in pseudocode in Algorithm 2.

```
Algorithm 2. RBMulticast Receiver
Require: Packet input from lower layer
Ensure: Forwarded packets inserted to MAC queue
Calculate checksum. Drop packet if error detected
Drop packet if not in Forwarding zone
Get destination list D from packet header
for node d in destination list D do
    if I am d then
        Duplicate the packet and input to upper layer
        Remove d from list D
    end if
end for
if TTL in header ¼ 0 then
    Drop the packet
    return
end if
for d 2 D do
    for multicast region r in 4 quadrants regions R do
        if d 2 r then
            Add d into r:list
        end if
    end for
end for
for r 2 R do
    if r:list is non-empty then
        Duplicate a new packet p
        Add RBMulticast header (TTL _ 1,
        checksum; r:list) to p
        Insert p to MAC queue
    end if
end for
```

Fig. 1 gives an example of how RBMulticast is employed. There two multicast regions, the southwest and northwest quadrants, contain only one multicast member each, and thus packet is sent directly to these multicast destinations.

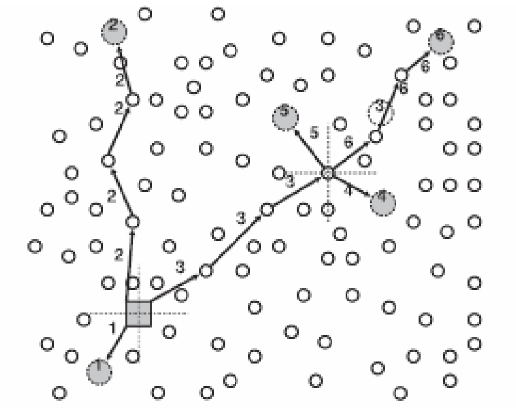


Fig. 1. Example showing how RBMulticast delivers multicast packets. The source node is the square node. Multicast members are shaded circles, and virtual nodes are dotted circles..

A. *Multicast Regions*

Once a node receives a multicast packet (from the application layer or from a previous hop node), it divides the network into multicast regions, and it will split off a copy of the packet to each region that contains one or more multicast members. We show two possible divisions of the network into multicast regions in Figs. 2a and 2b. There is no method that is clearly best. Influencing factors include the sink node locations and how the relay nodes are distributed. For the quadrants approach, the multicast region decision only needs two comparisons (X and Y axes) for each multicast member and is extremely fast. We believe that it is preferable for systems with low computational capacity such as wireless sensor nodes.

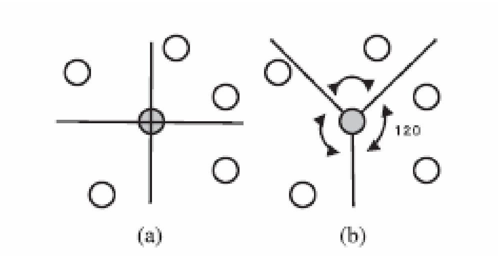


Fig. 2. Two possible ways to divide the space into multicast regions: (a) dividing the space into four quadrants and (b) dividing the space into three 120-degree regions.

B. Packet Splitting

In Algorithms 1 and 2, we describe the RBMulticast method that splits packets at relay nodes for which the multicast destinations reside in different regions. This method is used in the protocol description due to its simplicity. In a variation of this method, namely, RBM-V, the packets are instead split off at the neighbour nodes of the virtual node, which delays splitting the packets compared to the former method. Hence, in RBM-V, certain packets need to travel backward after splitting, which may increase the total hop count. However, as will be shown in Section 4.3, this variation of RBMulticast requires similar or lower average number of hops to reach all members.

C. Virtual Node

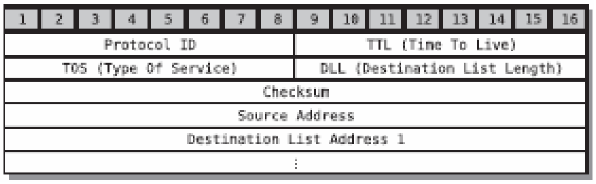
In RBMulticast, because we assume no knowledge of neighbour nodes and no routing tables, we assign a “virtual node” located at the geographic mean of the multicast members for each multicast region. This virtual node is used as an imaginary destination for the multicast packet in that region. The virtual nodes are not necessarily reachable or even physically exist as illustrated in Fig. 1. The idea behind this is that even if a virtual node does not exist, we can still find a route using the assumed receiver-based MAC protocol to get the packet closer to the location of the virtual node. On the other hand, when using the nearest multicast node as the destination, all node addresses physically exist and virtual nodes are not necessary.

D. RBMulticast Header

The goal of a stateless approach is to keep intermediate nodes from having to store any data for routing and medium access. This is possible only if all information required to multicast a packet is carried along with the packet. The question is how much information the multicast packet needs to carry for successful delivery to all multicast members. Fig. 3 shows the structure of an RBMulticast header. The first byte Protocol ID is for protocol identity in the protocol stack [24]. TTL provides a maximum time, in hop number, that a packet should last in the network. Type Of Service (TOS) indicates four kinds of packets in RBMulticast, which are “data,” “join,” “leave,” and “update” packets. The update packets are used in group management and periodic group list updates. Destination List Length (DLL) indicates how many nodes are in the node list, and thus will determine the length of the header.

The RBMulticast header size is not fixed since the destination list length is variable. Source Address is the address of the source node, which equals the RBMulticast group ID of this packet, and Destination List Address stores the locations of the DLL destination nodes. The RBMulticast group ID is not actually needed in this protocol since all the multicast members are included in the packet header. Because we assume a receiver-based MAC layer, the next hop is determined by a joint decision among potential receivers. Hence, the RBMulticast header does not need to carry any state for routing the packet. However, we still need to decide when the packet must be split off to different destinations. This is usually implied by tree branches in treebased multicast approaches. Because of the location information assumption, we can use multicast regions to decide when packets must be split off without any tree structure.

A packet will be split off to each multicast region if multicast members exist in these regions. Therefore, a destination list is the only requirement for multicast packet delivery: this destination list must be carried inside the packet header. As with any multicast protocol that uses a destination list, the packet header length will increase linearly with the number of destination nodes. The maximum number of multicast members allowed in a group is restricted by the packet size.



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Protocol ID								TTL (Time To Live)							
TOS (Type Of Service)								DLL (Destination List Length)							
Checksum															
Source Address															
Destination List Address 1															
⋮															

Fig. 3. Packet header of the RBMulticast protocol.

E. Group Management

RBMulticast supports multicast group management where nodes can join or leave any multicast group. Some nodes manage the multicast groups and act as the group heads. Nodes join and leave a group by sending “join” and “leave” packets to the group head. Join and leave packets are multicast packets with destination lists that contain only the group head address. RBMulticast supports Many-to-Many multicast mode, and thus every node in a multicast group can multicast packets to all other nodes in the same group. The extra burden is that the node must maintain group node lists for groups it has joined. In the case of nodes joining or leaving, the group head must send “update” packets including a list of its updated multicast group members to all group nodes. Nodes send “join” packets periodically to the group head, and nodes that die without sending “leave” packets are removed from the list after a time-out period. Thus the group leaders information have to be maintained.

V. PATH DIVERSITY

Path diversity allows CUs to switch dynamically among different paths for communicating with each other in presence of space-domain-dependent PU activity. Figure 1a shows how the PU activity can affect a routing process whenever it varies in space domain. Here, CU*B* and CU*C* are under the transmission range of two different PUs. By exploiting the path diversity, CU*A* can reach CU*D* through the optimal path CUA ⟶ CUB ⟶ CUD (when PU2 is not active); or the sub-optimal path CUA ⟶ CU*C* ⟶ CUD (when PU2 is active but PU3 is not), without the need of a new route discovery process.

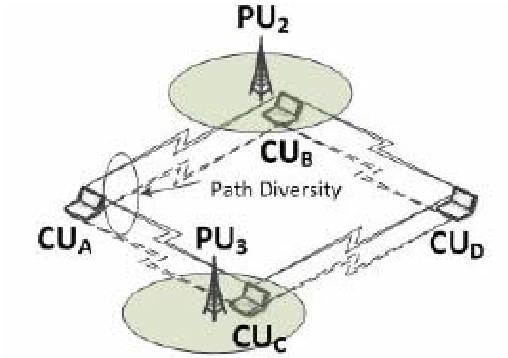


Fig. 4. Path Diversity

There are two channels between two nodes and if the first channel process is failure channel 1 will send information to channel 2 and the second channel will get allocated. The usage of which channel will be displayed in the dialog box. So the end-to-end delay is reduced and the packets are delivered efficiently.

VI. SIMULATION RESULTS

The usage of path diversity in the network reduces the end to end delay. The two channels between the nodes help in transmitting the data to the receiver in a more efficient way and thus the overall delay is reduces to a large extent.

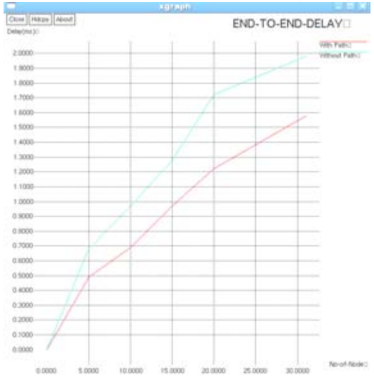


Fig 4.End to end delay

The control overhead is decreased to a large extent because the state information is not maintained by the individual nodes thus a large amount of communication overhead is decreased.

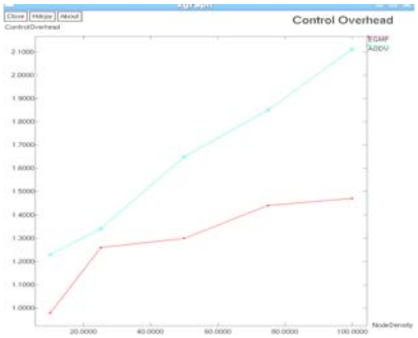


Fig 5.Control Overhead

The packet delivery ratio is increased due to the use of two channels and the information is sent with the packet headers and the states are not maintained by the nodes in the network.

Fig 6.Packet Delivery Ratio

Thus the packet delivery ratio and the node density are taken on the axis and the graph is drawn

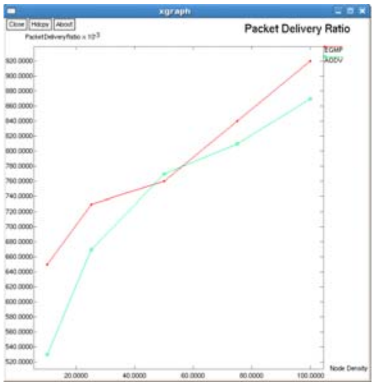


Fig 7.Packet delivery ratio versus node density

The multicast regions are divided into quadrants and thus the channel display is given in the dialog box, the either of the channel is chosen when one channel is a failure.

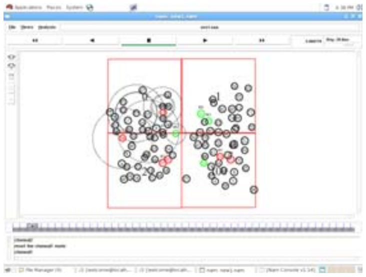


Fig 7. Dividing into multicast regions and display of channel

VII. CONCLUSION

The stateless multicast protocol does require any priori tree creation and the intermediate nodes do not maintain any state information in which lot of memory is needed and thus there is a large number of control overhead and the increase in communication overhead between the nodes,this can be overcome by this protocol and the results are simulated. The path diversity between the nodes decrease the end to end delay in the network and thus increases the packet delivery ratio.

The channel usage is done automatically when one channel fails and the transmission occurs through the other channel and thus the failed channel is reset and can be used for other transmission. Thus further enhancements can be done and the simulation results can be obtained.

REFERENCES

[1] C.-H. Feng and W.B. Heinzelman, “RBMulticast: Receiver Based Multicast for Wireless Sensor Networks,” IEEE Wireless Comm.and Networking Conf. (WCNC ’09) Apr. 2009.

[2] I. Akyildiz, M. Vuran, and O. Akan, “A Cross-Layer Protocol for Wireless Sensor Networks,” Proc. Conf. Information Science and Systems (CISS ’06), Mar. 2006.

[3] S.-J. Lee, M. Gerla, and C.-C. Chiang, “On-Demand Multicast Routing Protocol,” Proc. IEEE Wireless Comm. and Networking Conf. (WCNC ’99), vol. 3, pp. 1298-1302, 1999.

[4] J. Garcia-Luna-Aceves and E. Madruga, “A Multicast Routing Protocol for Ad-Hoc Networks,” Proc. IEEE INFOCOM, vol. 2, pp. 784-792, Mar. 1999.

[5] R. Vaishampayan and J. Garcia-Luna-Aceves, “Efficient and Robust Multicast Routing in Mobile Ad HOC Networks,” Proc. IEEE Int’l Conf. Mobile Ad-Hoc and Sensor Systems, pp. 304-313, Oct. 2004.

[6] E.M. Royer and C.E. Perkins, “Multicast Operation of the Ad-Hoc On-Demand Distance Vector Routing Protocol,” Proc. ACM MobiCom, pp. 207-218, 1999.

[7] J.G. Jetcheva and D.B. Johnson, “Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks,” MobiHoc ’01: Proc. Second ACM Int’l Symp. Mobile Ad Hoc Networking and Computing, pp. 33-44, 2001.

[8] C. Wu and Y. Tay, “AMRIS: A Multicast Protocol for Ad Hoc Wireless Networks,” Proc. IEEE Military Comm. Conf. (MILCOM ’99), vol. 1, pp. 25-29, 1999.

[9] K. Chen and K. Nahrstedt, “Effective Location-Guided Tree Construction Algorithms for Small Group Multicast in Manet,” Proc. IEEE INFOCOM, vol. 3, pp. 1180-1189, 2002.

[10] W. Wang, X.-Y. Li, and Y. Wang, “Truthful Multicast Routing in Selfish Wireless Networks,” Proc. ACM MobiCom, pp. 245-259, 2004.

[11] A. Okura, T. Ihara, and A. Miura, “BAM: Branch Aggregation Multicast for Wireless Sensor Networks,” Proc. IEEE Int’l Conf. Mobile Adhoc and Sensor Systems Conf., p. 363, Nov. 2005.

[12] C.-C. Chiang, M. Gerla, and L. Zhang, “Shared Tree Wireless Network Multicast,” Proc. Sixth Int’l Conf. Computer Comm. and Networks, pp. 28-33, Sept. 1997.

[13] D. Koutsonikolas, S. Das, H. Charlie, and I. Stojmenovic, “Hierarchical Geographic Multicast Routing for Wireless Sensor Networks,” Proc. Int’l Conf. Sensor Technologies and Applications (SensorComm ’07), pp. 347-354, Oct. 2007.

[14] S. Das, H. Pucha, and Y. Hu, “Distributed Hashing for Scalable Multicast in Wireless Ad hoc Networks,” IEEE Trans. Parallel and Distributed Systems, vol. 19, no. 3, pp. 347-362, Mar. 2008.

[15] S. Basagni, I. Chlamtac, and V.R. Syrotiuk, “Location Aware, Dependable Multicast for Mobile Ad HOC Networks,” Computer Networks, vol. 36, nos. 5/6, pp. 659-670, 2001.

[16] M.M. Holger Fuler, J. Widmer, and T. Lang, “Position-Based Multicast Routing for Mobile Ad-Hoc Networks,” ACM SIGMOBILE Mobile Computing and Comm. Rev., vol. 7, no. 3, pp. 53-55, 2003.

[17] J. Sanchez, P. Ruiz, and I. Stojmnenovic, “GMR: Geographic Multicast Routing for Wireless Sensor Networks,” Proc. Third Ann. IEEE Comm. Soc. Conf. Sensor and Ad Hoc Comm. and Networks (SECON ’06), vol. 1, pp. 20-29, Sept. 2006.

[18] S. Biswas and R. Morris, “Opportunistic Routing in Multi-Hop Wireless Networks,” SIGCOMM Computer Comm. Rev., vol. 34, no. 1, pp. 69-74, 2004.

[19] E. Rozner, J. Seshadri, Y.A. Mehta, and L. Qiu, “SOAR: Simple Opportunistic Adaptive Routing Protocol for Wireless Mesh Networks,” IEEE Trans. Mobile Computing, vol. 8, no. 12, pp. 1622-1635, Dec. 2009.

[20] B. Karp and H.T. Kung, “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks,” Proc. ACM MobiCom, pp. 243-254, 2000.

[21] C.-H. Feng, I. Demirkol, and W.B. Heinzelman, “UPS: Unified Protocol Stack for Wireless Sensor Networks,” Proc. Sixth Ann. Int’l Mobile and Ubiquitous Systems: Networking and Services, 2009.