

# Bandwidth-Efficient Geographic Multicast Routing Protocol for Wireless Sensor Networks

Juan A. Sanchez, Pedro M. Ruiz, *Member, IEEE*, Jennifer Liu, and Ivan Stojmenovic, *Member, IEEE*

**Abstract**—We present geographic multicast routing (GMR), a new multicast routing protocol for wireless sensor networks. It is a fully localized algorithm that efficiently delivers multicast data messages to multiple destinations. It does not require any type of flooding throughout the network. Each node propagating a multicast data message needs to select a subset of its neighbors as relay nodes towards destinations. GMR optimizes the cost over progress ratio where the cost is equal to the number of neighbors selected for relaying and the progress is the overall reduction of the remaining distances to destinations. Such neighbor selection achieves a good tradeoff between the bandwidth of the multicast tree and the effectiveness of the data distribution. Our cost-aware neighbor selection is based on a greedy set merging scheme achieving a  $O(Dn \min(D, n)^3)$  computation time, where  $n$  is the number of neighbors of current node and  $D$  is the number of destinations. As in traditional geographic routing algorithms, delivery to all destinations is guaranteed by applying face routing when necessary. Our simulation results show that GMR outperforms position based multicast in terms of cost of the trees and computation time over a variety of networking scenarios.

**Index Terms**—Geographic routing, multicast, wireless sensor networks (WSNs).

## I. INTRODUCTION AND MOTIVATION

A WIRELESS SENSOR NETWORK (WSN) consists of a set of networked sensor nodes that communicate among each other using wireless links. They work in a distributed way, and collaborate to perform automated tasks requiring sensing capabilities. Sensor nodes are usually small, inexpensive and with limited communication, computation and energy resources. The number of such sensors in a WSN is expected to be large, in the order of hundreds or thousands. We assume that sensors know their position via GPS or another virtual position system [1], [2]. They periodically share this information with their neighbors (nodes placed within their radio range). In most WSNs, communications are performed in a multihop way. Sensors use their neighbors to send messages to nodes located out of their radio range.

Possible applications of WSNs are endless, including habitat monitoring, wildfire detection, pollution monitoring, etc. In

many of these scenarios, there are applications in which a single sensor needs to send the same data to multiple destinations. Those applications can benefit from the use of multicast communications to reduce bandwidth consumption in the network. Examples of those applications include data replication, assignment of tasks or sending of commands (especially in sensor and actuator networks) to a specific group of sensors, queries to multiple sensors, etc. For instance, one real application in which we are using is the control of water sprinklers for water irrigation, in which, in many cases, the controller needs to send orders to a particular set of sensors, whose location is known. Some of these applications are data-intensive, therefore, it is of paramount importance for them, to count on an efficient multicast mechanism being able to alleviate the overall consumption of resources in the network. Multicasting is a technique used to deliver messages efficiently from a source to a set of destinations. Multicasting protocols try to minimize the consumption of network resources taking advantage of the fact that some parts of the paths from the source to the destinations can be shared by multiple destinations. The larger the path shared, the lower overall bandwidth consumption is obtained.

There have been a lot of multicast routing proposals for ad hoc networks [3], each of them based on different design decisions. Unfortunately, they cannot fulfill the unique requirements of WSNs effectively. They are mostly designed to deal with highly mobile nodes, with higher processing and storage capacity, and a much limited amount of nodes. In addition, WSNs are characterized by their topological changes due to node failures or duty-cycle operation. These characteristics make localized routing algorithms [4] more appropriate for WSNs. Unlike centralized ones, localized algorithms do not need to know the complete topology to take routing decisions. Furthermore, centralized algorithms introduce too much overhead to be used in WSNs.

Providing efficient multicast routing in WSNs poses special challenges compared with unicast data delivery. In fact, the problem of computing a minimal bandwidth consumption multicast tree in wireless multihop networks was recently proven [5] to be NP-complete. This becomes specially challenging when overhead needs to be kept low due to the limited battery, storage capacity, bandwidth, and processing power of sensor nodes.

In this paper, we present a new multicast geographic routing protocol in which the main contribution is a new heuristic neighbor selection scheme. The new scheme requires a low computational cost and is able to compute very efficient multicast paths. In addition, the protocol does not require any type of networkwide flooding. It is solely based on local geographic information obtained from neighboring nodes. By selecting

Manuscript received June 15, 2006; revised September 8, 2006; accepted October 2, 2006. This work was supported in part by the Spanish MEC by means of the “Ramon y Cajal” Research Program, in part by the SMART TIN2005-07705-C02-02 Project, and in part by the Natural Sciences and Engineering Research Council of Canada (NSERC). The associate editor coordinating the review of this paper and approving it for publication was Dr. Subhas Mukhopadhyay.

J. A. Sanchez and P. M. Ruiz are with the Department of Communications and Information Engineering, University of Murcia, Espinardo, 30071 Murcia, Spain (e-mail: jlaguna@dif.um.es; pedrom@dif.um.es).

J. Liu and I. Stojmenovic are with SITE, University of Ottawa, Ottawa, ON K1N 6N9, Canada (e-mail: jenniferliu@hotmail.ca; ivan@site.uottawa.ca).

Digital Object Identifier 10.1109/JSEN.2007.894149

neighbors based on our cost-aware metric, the proposed protocol is able to outperform previous schemes in terms of the cost of multicast packet delivery, and the computational cost of computing such trees.

Our work is focused on the localized construction of bandwidth optimal multicast trees. Thus, given that the focus is on the efficient neighbor selection function, we assume throughout this paper that positions of destinations are known to multicast sources. To know that information, sources in geographic routing protocols employ location systems such as ALS [6]. That is, in our case of geographic multicast routing those systems correspond to the functions of group membership and management. Furthermore, given that the number of destinations is expected to be low for the multicast scenarios considered for sensor networks, our proposed scheme is mainly concerned with an enhanced neighbor selection criterion. The proposed multicast protocol GMR selects neighbors based on the cost over progress framework that was first introduced by Kuruvila *et al.* [7] (for routing problem) integrated with a greedy neighbor selection. Our cost function considers the number of transmissions based on the results of Ruiz *et al.* [5], that showed that the optimality of a multicast tree in terms of bandwidth consumption needs to be evaluated in terms of the minimization of the number of transmissions performed. Our scheme avoids hard-to-tune parameters, has lower computational costs, and computes multicast paths with a lower overall cost. Moreover, our scheme is general enough to be easily coupled with any scalable group management scheme.

The remainder of this paper is organized as follows. Section II provides an overview of the state of the art, and Section III describes the GMR protocol. Our proposed greedy neighbor selection algorithm is explained in Section IV. We evaluate the performance of our solution using simulation in Section V. Finally, Section VI provides some conclusions and discusses open issues.

## II. RELATED WORK

There have been a lot of multicast routing proposals for ad hoc networks; each of them based on different design decisions. Multicasting protocols proposed for mobile ad hoc networks can be divided into tree-based and mesh-based protocols depending on the forwarding structures they employ. Mesh-based protocols expand a multicast tree with additional paths that can be used to forward multicast data packets when some of the links break. Examples of those protocols are ODMRP [8], CAMP [9], and PUMA [10]. These protocols have proven to be particularly suited for scenarios with high mobility rates. However, the maintenance of these structures through periodic broadcasts and the large amount of nodes which are required to forward multicast data messages make them impractical for sensor networks. Tree-based protocols such as MAODV [11], ADMR [12], and AMRIS [13] do typically use a lower number of relay nodes, but the tree needs to be reconstructed when links break due to mobility of the nodes. In addition, they also rely on periodic flooding, which is a costly operation for a sensor network. So, in both cases, their nonlocalized operation produces an excessive control overhead for a sensor network.

There have also been some proposals taking advantage of position information to perform multicast routing. Protocol DDM [14] works in an opportunistic way. Destinations are included in the data packet and nodes decide next hops for each data packet based on the distance of the destination measured in number of hops from its unicast routing table. However, maintaining its unicast routing table requires the additional overhead of the unicast routing protocol itself. Other existing proposals such as LAM [15] (which is built upon TORA [16] and CBT [17]) make extensive use of broadcast, being impractical for WSNs.

Mizumoto *et al.* [18] proposed a protocol to send multicast messages to a geographic area rather than to multiple destinations. They use position information to build a multicast tree that aims at minimizing the number of links. However, because of wireless medium that allows one to many communication, the cost of the tree is better characterized by the number of nodes. Thus, the resulting trees are not optimal.

Geographic unicast routing protocols [19]–[21] have proven to be very effective in providing unicast routing in the common resource-constrained scenarios [4], [22], [23] that WSNs present. They work with local information, require a low computational cost, adapt very fast to changing network conditions, and are able to route messages with a very low control overhead. In geographic routing, each node taking part in a routing process takes decisions about which neighbor is the best one to be selected as next forwarder in order to carry the message as nearer to the position of the destination as possible. Thus, the information about position is fundamental. Although the use of hardware-based positioning systems such as GPS might be possible, there are scenarios (e.g., indoor) in which they cannot be effectively used. However, they can also work based on virtual coordinates, as shown in [1] and [2]. Similarly, our proposed scheme can also work based on these virtual coordinates.

Another interesting aspect is how the sources know the position of the destinations. To do that, different distributed localization systems such as GLS [24], ALS [6], or the Quorum-Based Location System [25] can be used. Also, some schemes based on triangulation of coordinates such as SERLOC [26] have been used in the literature. In fact, for the particular case of multicast, these systems also take care of group memberships. They basically store such information across the nodes in the network so that multicast sources can access to the information with a low control overhead. In our case, allowing the multicast source to get the position of nodes interested in some particular data (or multicast group). Our proposed protocol can use any of those systems for group management. In fact, given that ALS [6] has showed a good performance compared with the rest, it could be a very good candidate.

Given the difficulty of adapting existing ad hoc multicast protocols to a fully localized operation, new protocols have been recently proposed to meet those requirements. One of such protocols is position based multicast routing (PBM) protocol proposed by Mauve *et al.* in [27]. This protocol, though not initially thought for sensor networks, fulfills most of the desired design criteria of localness and limited network overhead. It is a generalization of routing to operate over multiple destinations. It builds a multicast tree, whose shape can vary from the shortest

path tree, to an approximation of a minimum cost multicast tree depending on a parameter denoted as  $\lambda$ .

Authors in [27] try to find a good tradeoff between the total number of nodes forwarding the message and the optimality of individual paths towards the destinations. Each node evaluates all possible subsets of neighbors ( $W$ ) using  $f(w) = \lambda N + (1 - \lambda)S$ ,  $0 \leq \lambda \leq 1$  to evaluate each  $w \in W$ , where  $N$  is the number of neighbors in the considered subset ( $|w|$ ) divided by the total number of neighbors ( $n$ ), and  $S$  is the summation of the minimal distances from nodes in  $W$  to destinations, normalized by the summation of distances from the current node to all destinations. If at some node there is no node providing advance towards one or more destinations, the authors use, only toward those destinations, a variant of face routing like the one we describe in Section III-C.

The main problem with this approach is that determining the optimal value for  $\lambda$  is not a trivial task. In fact, the authors evaluated different values of  $\lambda$  but they never came out with a determination of an optimal value. An additional issue is the fact that the algorithm is computationally expensive. Evaluating all possible neighbor subsets has an exponential computational cost as the number of neighbors increases.

For networks with a very large number of multicast receivers, PBM may not scale well due to the need to include all destinations in multicast data packets. Scalable position based multicast for mobile ad hoc networks (SPBM) [28] was designed to improve scalability. It uses the geographic position of nodes to provide a scalable group membership scheme and to forward data packets. SPBM is mainly focused on the task of managing multicast groups in a scalable way. However, they fail to provide efficient multicast forwarding, because they use one separate unicast geographic routing for each destination. In addition, the interchange of routing tables between neighbors makes the protocol not so scalable to the number of multicast groups as PBM is.

### III. GEOGRAPHIC MULTICAST ROUTING (GMR)

We present in this paper GMR, a new multicast routing protocol for WSNs. The problem we are facing can be described as follows: given a multicast message generated by a source node, find a subset of nodes in the network so that the message is delivered to all destinations (sinks) with a very low consumption of bandwidth. We must minimize the number of messages sent, which means using a limited bandwidth and using as few sensors as possible to route the message to the destinations. It is important to design algorithms with a low computational cost, and a constrained memory consumption.

To achieve those goals, our protocol will be based on the idea of geographic routing, which is able to meet those requirements. Of course, we adapt it so that it can deal with multiple destinations. As in previous geographic routing schemes, we assume that sensor nodes know their position, and they communicate their position to neighbors using periodic beacons. If real position (e.g., GPS coordinates) is not available, virtual coordinates can also be used effectively [2]. The source is also assumed to know the position of the destinations as in previous geographic routing protocols. To do that, schemes such as ALS [6] can be used.

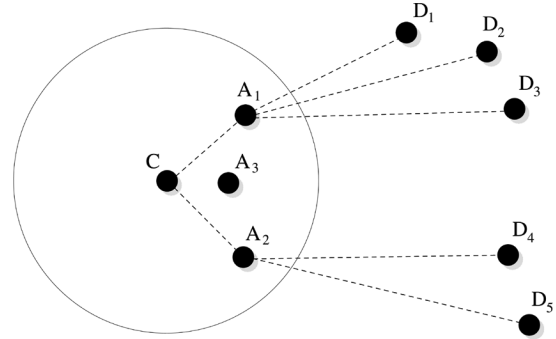


Fig. 1. Evaluating the candidate forwarding from C to A1 and A2.

Sensor nodes running GMR use the position of their neighbors to select which subset of them is the best to propagate the message towards the destinations. In our case, the selected subset is the one with the lowest total distance to destinations per unit of cost. When several neighbors are selected, each of them takes care of routing towards part of the overall set of destinations. Of course, due to the use of local information only, it may happen that the decision taken by a node is not always right. For instance, voids can make the protocol fall into a local minima, reducing thus its performance. When that happens, (i.e., for some destinations no neighbor of the current node can reduce the distance) the algorithm uses face routing to exit the local minima until a new node providing advance is found. Within the remainder of this section, we will explain the general operation of the protocol in detail. Section IV describes the way in which forwarding neighbors are selected.

#### A. Network Model and Problem Formulation

In this paper, we model a WSN as an undirected graph  $G = (V, E)$ , where  $V$  is the set of vertices and  $E$  is the set of edges. The model, known as unit disk graph (UDG), assumes that the network is two-dimensional (every node  $v \in V$  is embedded in the plane) and wireless nodes are represented by vertices of the graph. Each node  $v \in V$  has a transmission range  $r$ , which is equal for all nodes. Let  $\text{dist}(v_1, v_2)$  be the distance between two vertices  $v_1, v_2 \in V$ . An edge between two nodes  $v_1, v_2 \in V$  exists  $\iff \text{dist}(v_1, v_2) \leq r$  (i.e.,  $v_1$  and  $v_2$  are able to communicate directly).

#### B. Cost Over Progress Metric

We will now explain how the cost over progress metric can be used to select next hops towards destinations when nodes know their position.

Consider the case in Fig. 1 as illustration of the general principle. As we can see, a source wishes to send a packet to a number of destinations (sinks) with known positions. It is assumed that the number of such destinations is small, which is a reasonable assumption for the considered scenarios. Unlike PBM, we describe here a solution which does not need any parameter. Assume that a node  $C$ , after receiving a multicast message, is responsible for destinations  $D_1, D_2 \dots D_5$ , and that it evaluates neighbors  $A_1, A_2$  as possible candidates for forwarding. The whole task could be sent to one neighbor only (if there exist one that is closer to all destinations than  $C$ ), or could

be split to several neighbors, each with a subset of destinations to handle. Hop count is expected to be proportional to distances.

The current total distance for multicasting is  $T_1 = |\overline{CD_1}| + |\overline{CD_2}| + |\overline{CD_3}| + |\overline{CD_4}| + |\overline{CD_5}|$ . If  $C$  considers  $A_1$  and  $A_2$  as forwarding nodes, covering  $D_1, D_2, D_3$ , and  $D_4, D_5$ , respectively, the new total distance is  $T_2 = |\overline{A_1D_1}| + |\overline{A_1D_2}| + |\overline{A_1D_3}| + |\overline{A_2D_4}| + |\overline{A_2D_5}|$ , and the progress made is  $T_1 - T_2$ . Our aim is also to minimize the consumption of bandwidth, which is proportional to the total number of forwarding nodes selected. Thus, the cost is the number of selected neighbors, which in the above example is 2. Thus, the forwarding set  $\{A_1, A_2\}$  is evaluated as  $(2)/(T_1 - T_2)$ . Among all candidate forwarding sets, the one with minimal value of this expression is selected. If there is no neighbor closer than  $C$  towards one or more of the destinations, then we have to enter into face mode. Section III-C describes how to proceed in this case.

### C. Multicast GFG

Multicasting is normally expected to proceed in greedy mode. That is, a node selects neighbors closer to the destinations than itself. However, a node  $C$  currently routing a message might not have any neighbor providing advance towards some of the destinations included in the message header. This situation is known as a local optimum for the greedy mode. Those destinations are included in a list called *multicast face list*. In unicast geographic routing, a recovery scheme called face routing [20] can be applied. It describes how to find an alternative route to escape from the local optimum until the message reaches a node where greedy mode can be continued. The route toward that particular destination is said to be in perimeter mode during the change from greedy to perimeter mode, and during the search for a node closer to the destination than the node that experienced local minima.

When a node  $C$  has to route in perimeter mode for some of the destinations included in the message, it decides the next hop for every destination in the face list according to GFG routing [20]. The protocol first decides which of the edges incident to  $C$  belong to a planar subgraph. The most common planar subgraph used is the Gabriel Graph (GG), which contains an edge  $CA$  if the disk with diameter  $CA$  contains no other nodes inside it. Note that node  $C$  may decide which of its edges belong to GG based on the position of itself and its neighbors, without sending any message for the purpose of constructing GG. Face routing is applied independently toward each destination in the face list. However, it may happen that the next selected neighbor is the same for several destinations. In this case, one single message is sent in perimeter mode, including the list of all such destinations.

When a message with some destinations being routed in perimeter mode arrives at a node  $C$ , it checks whether its position is closer to any of the destinations than the node where the multicast perimeter mode started. If the test is positive, such destination is removed from the multicast face list and added to the list of destinations to be routed in greedy mode. Multicast perimeter mode ends when the multicast face list is empty.

It is also possible that the current node uses the same neighbor to forward traffic to different destinations being in

different modes. That is, greedy routing may be in progress for some destinations, while perimeter mode can be in progress for others. Current node  $C$ , in fact, transmits a single message (counted as one in the overall cost), listing all destinations, the mode being applied to each of them, and the neighbors that need to handle each of these destinations. As a result, a particular neighbor may be assigned to handle some destinations in greedy mode and some destinations in the perimeter mode.

### D. Packet Format

When multicast data is being forwarded, only those neighbors selected by the current node have to process the message. We add a GMR header to data messages to allow these nodes to realize that they are selected. It is also used to mark which destinations require perimeter mode.

That header is made of the position of the sender and a list of records, with one record associated to each relay. A message transmitted by a node is received by all its neighbors. This means that one single message is sent, that is received by all the selected relays. This property is commonly known as the *wireless multicast advantage* and it can help us to further reduce the overhead of our protocol. Every record contains information about the identifier of the selected relay and a list of the destinations it is responsible for either in greedy or in perimeter mode.

## IV. GREEDY NEIGHBOR SELECTION

In this section, we give a detailed explanation of the neighbor selection function. That is, what is the concrete algorithm used by the current node to decide which subset of its neighbors will forward multicast data messages towards which subset of destinations. In this section, we describe that part of the protocol, showing the benefit in terms of computational cost compared to previous works.

Given  $k$  destinations, a possible algorithm can consider all  $S_k$  partitions of the set of destinations. For each subset in a given set partition, the node checks whether it is possible to find a neighbor that is closer to all destinations in that subset than the current node  $C$ . If this is not possible for a subset, then this partition is ignored. If this is possible for each subset in the given set partition, then we measure the cost/progress ratio. Finally, after all the evaluations, we choose the best among all measured ones. This solution is applicable for a small number of destinations, e.g., up to five. For a larger number, it becomes exponential in  $k$ , and therefore a faster greedy solution like the one presented below is needed.

We start with the set of destinations  $\{D_1, D_2, \dots, D_k\}$  for which there is a neighbor of the current node providing advance. We first group together, into the same subset, those destinations for which the neighbor providing the most advance is the same. For instance, in Fig. 1, the initial set partition to consider would be  $\{\{D_1, D_2, D_3\}, \{D_4, D_5\}\}$ , where  $A_1$  serves  $D_1, D_2, D_3$  and  $A_2$  serves  $D_4$  and  $D_5$ . In general, the set partition could be  $\{M_1, M_2, \dots, M_m\}$  with each destination being in exactly one of these subsets. Each  $M_i$  has its own cost-progress ratio, and the whole set partition also has its own cost-progress ratio as we explained before. The cost-progress ratio for the subset  $\{D_1, D_2, D_3\}$  in Fig. 1 can be computed as follows: The current

total distance for multicasting is  $T_1 = \lceil \overline{CD_1} \rceil + \lceil \overline{CD_2} \rceil + \lceil \overline{CD_3} \rceil$ . The new total distance is  $T_2 = \lceil \overline{A_1D_1} \rceil + \lceil \overline{A_1D_2} \rceil + \lceil \overline{A_1D_3} \rceil$ , and the progress made is  $T_1 - T_2$ . The cost is the number of selected neighbors, which in the above example is 1. Thus, the forwarding cost over progress ratio is evaluated as  $(1/T_1 - T_2)$ .

Let  $P_i$  be the progress in  $M_i$  (each  $P_i$  is  $T_1 - T_2$  from above explanation). The cost in each  $M_i$  is 1 since all destinations in a given subset are served by the same (one) neighbor. The overall cost-progress ratio for the set partition is then  $(m/\sum_{i=1}^m P_i)$ . We are looking for the set partition with optimal ratio.

The greedy set partition selection algorithm for multicast presented in algorithm 1 works as follows. First, the initial partition set  $M = \{M_1, M_2, \dots, M_m\}$  is initialized as we explained before. The current cost-progress ratio is computed accordingly as  $(m/\sum_{i=1}^m P_i)$ . The selection process then proceeds in rounds. In each round, all pairs  $\{M_i, M_j\}$  are checked for possible improvement over previously best cost over progress ratio. Two partitions can be combined only if there are neighbors of the current node, providing advance towards all the destinations in both partitions  $M_i$  and  $M_j$ . However, their merging, if possible, may not result in a better cost over progress ratio. The pair that provides the best improvement is selected and merging is performed, creating the new set partition  $M$ . The process advances to the next round and starts over again with the new set  $M$ . If no pair provided any improvement then the process stops and the best set partition  $M$  is found.

---

### Greedy set partition selection algorithm

---

- 1:  $M = \{M_1, M_2, \dots, M_m\}$  so that  $M_i = \{D_j\}$   
same neighbor provides most advance}
  - 2:  $C \text{ Ratio} = (m/\sum_{i=1}^m P_i)$
  - 3: **repeat**
  - 4:  $\text{BestReduction} = 0$
  - 5: **for all** pairs  $\{M_i, M_j\}$  **do do**
  - 6: Find cost over progress *reduction* by merging of  
 $\{M_i, M_j\} \in M$
  - 7: **if**  $\text{reduction} > \text{BestReduction}$  **then**
  - 8:  $\text{BestReduction} = \text{reduction}$
  - 9:  $\text{BestMerge} = \{M_i, M_j\}$
  - 10: **end if**
  - 11: **end for**
  - 12: **if**  $\text{BestReduction} > 0$  **then**
  - 13:  $M = \{M_1, M_2, \dots, \{M_i, M_j\}, \dots, M_m\}$
  - 14:  $C \text{ Ratio} = (|M|/\sum_{i=1}^{|M|} P_i)$
  - 15: **end if**
  - 16: **until**  $\text{BestReduction} = 0$
- 

We now explain in more detail how two partitions merge, and how is reduction calculated. In order to merge two subsets  $M_i$  and  $M_j$  as one single set  $M_{i,j}$ , the algorithm considers the set of

neighbors that are closer to all destinations in  $M_i \cup M_j$  than the current node  $C$ . If that set is empty, then merging is not possible, and  $\text{reduction} = 0$ . Otherwise, among all such neighbors, we select the one which provides the best cost over progress ratio for this new subset  $M_{i,j}$  (that is, the one that maximizes the corresponding progress  $\text{BestReduction} = T_1 - T_2$ ).

This algorithm, instead of testing all possible subsets, only needs to test  $O(D^3)$  of them (in the worse case),  $D$  being the total number of destinations. As discussed in the next section, when the number of neighbors of the current node is lower than the number of destinations, there is no need to test more than  $n^3$  subsets,  $n$  being the number of neighbors.

We merge two subsets of destinations because it reduces the cost of retransmitting from 2 to 1. A node providing advance toward all destinations in the merged set may not improve overall progress significantly, but reduces retransmission cost in half, and the overall cost over progress ratio improves. We will now elaborate on this in more detail. Let us assume that  $M = \{M_1, M_2, \dots, M_m\}$  is the initial set partition in which all destinations in every  $M_i$  are served by the same closest node ( $B_i$ ) among all neighbors of the current node  $A$ . Let  $N(A)$  be the set of  $A$ 's neighbors. Given two subsets  $M_i, M_j \in M$ , we will analyze the conditions under which some neighbor with a lower progress can still provide a better tradeoff.

Let  $m$  be the cost of the initial partition, and  $\sum_{k=1}^m P_k$  be the progress made with such election. If we merge  $M_i$  and  $M_j$  into a single set served by a single node  $B_{i,j} \in N(A)$ ,  $B_{i,j} \neq B_i, B_{i,j} \neq B_j$ , the cost of the new subset  $M_i \cup M_j$  is 1. The overall cost after merging is then  $m - 1$ . In addition, the new progress made after merging would be  $\sum_{k \neq i, k \neq j} P_k + P_{i,j}$ , where  $P_{i,j}$  is the progress made by  $B_{i,j}$  towards destinations in  $M_{i,j}$ .

For the new cost-progress to be better than before merging, the following inequality must be satisfied:

$$\frac{m}{\sum_{k=1}^m P_k} > \frac{m-1}{\sum_{k \neq i, k \neq j} P_k + P_{i,j}}$$

Thus, the new overall progress of candidate neighbors to merge subsets must satisfy

$$\sum_{k \neq i, k \neq j} P_k + P_{i,j} > \frac{m-1}{m} \sum_{k=1}^m P_k$$

This means that all those nodes  $x \in N(A)$  whose progress towards destinations satisfy above inequality can provide a better cost over progress ratio than the initial selection, even though they do not provide the best progress for any of the destination sets. Every reduction of cost by merging two subsets decreases the progress made up to  $(\sum_{k=1}^m P_k/m)$ .

In Fig. 2, we can see the current node  $S$  and two neighbors  $n_1$  and  $n_2$  within its radio range. We also see destinations  $D_1$  and  $D_2$ , where  $n_1$  and  $n_2$  are the neighbors providing the best progress to  $D_1$  and  $D_2$ , respectively. The shaded zone is the one in which there may exist other neighbors that can improve the overall cost over progress ratio. They do not provide the best advance for any of the individual destinations, but they can provide a better tradeoff. The legend of the graph represents the

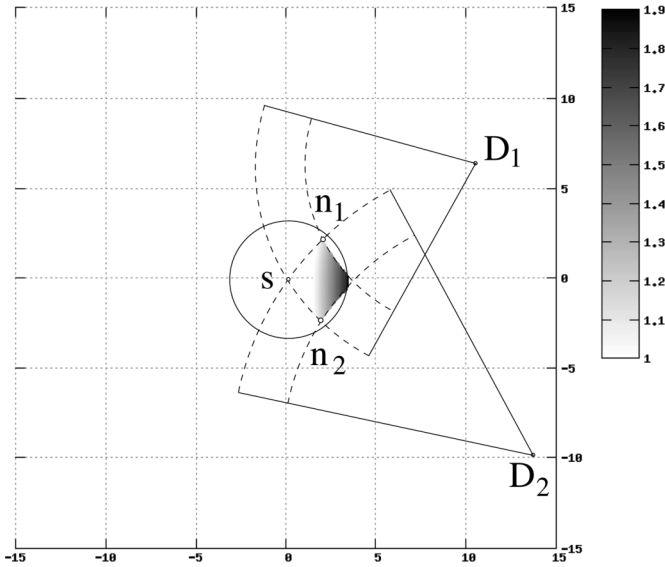


Fig. 2. Identification of nodes according to their goodness.

amount of improvement in cost/progress ratio over the configuration before merging.

#### A. Algorithm Complexity

In this section, we evaluate the worst case complexity of the neighbor selection algorithm and we show that it is asymptotically lower than that of PBM for most of the cases.

*Theorem 1:* The complexity of GMR is  $O(Dn \min(D, n)^3)$ , where  $D$  is the number of destinations and  $n$  the number of neighbors of the node currently multicasting the message.

*Proof:* The first step of the algorithm builds initial partition set  $M = \{M_1 \dots M_m\}$  by dividing destinations in subsets. Every subset  $M_i$  consists of those destinations whose closest neighbor of the current node is the same. That is, if a neighbor gives the best progress to two different destinations, that two destinations will be in the same initial subset. This stage will need  $D * n$  comparisons.

If the number of destinations is lower than the number of neighbors,  $D > n$ , then the number of subsets in  $M$  is, in the worst case, of size  $D$  (i.e., every subset contains only a single destination). If the number of destinations is greater than the number of neighbors,  $D < n$ , then the number  $m = |M|$  of subsets in the initial partition  $M$  is  $\leq n$  because it is impossible to have more subsets than neighbors. In this case,  $|M_i| > 1$  for one or more  $M_i \in M$ . Thus,  $0 \leq m \leq \min(D, n)$ .

The process of computing the best subsets  $M_i, M_j \in M$  to merge requires testing each pair of subsets. The number of tested pairs is  $(m * (m - 1)/2)$ . The largest number of iterations occurs when at every step it is always possible to find two subsets to be merged that improve the ratio. The total number of iterations in that case is  $m - 1$ . Then,  $m$  may range from 1 to  $\min(D, n)$ . Given that iteration with  $m$  subsets may test  $O(m * (m - 1)/2)$  pairs, the worst case number of merging operations is  $(m^3 - 6m^2 - m)/6$ . This gives us an algorithm with  $O(m^3)$  merging steps. Merging two subsets may involve verifying all  $n$  neighbors for their feasibility, and testing  $O(D)$

destinations in two subsets being merged. Thus, in the worse case, the algorithm has a complexity of  $O(Dn \min(D, n)^3)$ . ■

In the case of PBM, in all cases (i.e., worst, average, or best one), all possible subsets of neighbors (and not the subsets of destinations) are computed to select the best one. These neighbors are called forwarding nodes, and, for each destination, one of them takes responsibility for forwarding. Given  $n$  neighbors, the number of possible subsets is  $\sum_{k=0}^n \binom{n}{k} = 2^n$ . This leads to an exponential time complexity, which is much higher than the time complexity of selecting forwarding neighbors for GMR.

In addition, we must consider that in the average case of our algorithm, the cost is expected to be much lower. The computation time is evaluated experimentally in the next section.

## V. PERFORMANCE EVALUATION

Our simulations compare our proposed cost over progress scheme, with six variants of the position-based multicast routing PBM [27]. The reason for choosing PBM, is that it is, to the best of our knowledge, the best localized geographic multicast routing algorithm to date. These variants correspond to different values of the  $\lambda$  parameter. Being more specific, we use values of  $\lambda = 1, 0.8, 0.6, 0.4, 0.2, 0$ . The reason is that PBM behavior varies significantly depending on the value of  $\lambda$ , and there is no mechanism to find the best  $\lambda$  for a particular scenario. Authors of PBM showed that  $\lambda = 0.4$  in their simulations is the one that seemed to provide the best performance. However, as we will see in this section, the best value of  $\lambda$  depends on specific network parameters. In addition, we also compare the quality of the trees produced by GMR with shortest path trees (SPTs), which are the ones used by traditional ad hoc multicast routing protocols such as ODMRP [8] and MAODV [11]. We consider a perfect MAC layer without collisions, and a UDG model. That is, a message sent out by a node is received by every other node in its radio range.

#### A. Simulation Setup

The simulation scenario consists of 1000 nodes randomly placed in an area of  $1000 \times 1000$  m. The radio range is varied in order to achieve different network densities in terms of a mean number of neighbors. Note that this is equivalent to using a fixed radio range and increasing the simulation area. Densities considered were a mean number of 5, 6, 7, 8, 9, and 10 neighbors per node, whereas the number of multicast destinations was 5, 10, 25, and 50. Receivers were also randomly selected from the set of sensor nodes. For each configuration, our results are the mean over a total number of 50 simulation runs, which proved to be sufficient to provide a small 95% confidence interval.

#### B. Performance Metrics

To assess the performance of the proposed schemes, we considered the following performance metrics.

- Number of transmissions. This metric measures the efficiency of the multicast paths selected. The lower the number of transmissions, the lesser the network resources consumed to deliver the data message to all destinations. This is the most important performance parameter.

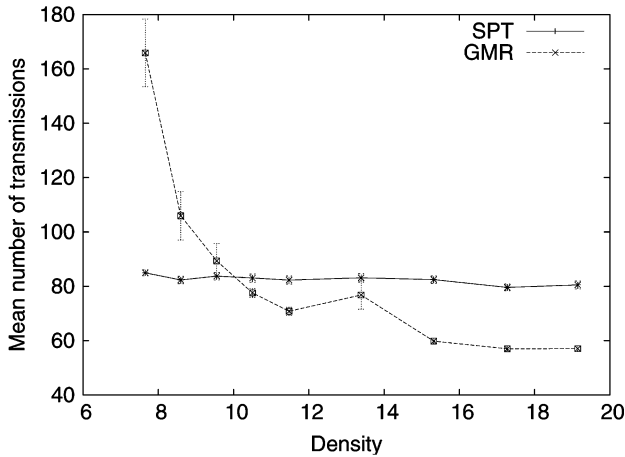


Fig. 3. Mean number of transmission for ten destinations at varying density.

- **Stretch factor.** The stretch factor is the maximum difference in hops between the SPT to a receiver and the actual path. This metric provides an indication of how much does the protocol diverge from shortest paths.
- **Computation time.** This metric evaluates how costly in terms of computational power the protocol is. It is also useful to compare the mean case performance of the different protocols in comparison with the theoretical worse case results we obtained in the previous section.

### C. Number of Transmissions

We analyze in this section the effectiveness of neighbor selection schemes. We shall see that our proposed scheme achieves a good tradeoff between the lengths of individual paths, and the overall number of transmissions. That is, our scheme only splits paths to multiple destinations, when there is no cost-effective next hop towards the whole set of destinations. By using that cost-based path splitting strategy, we are able to identify when it is really interesting to create different paths.

To study the impact of the density in GMR and showing its performance against traditional ad hoc routing protocols such as MAODV [11] and ODMRP [8], we simulate both GMR and SPTs at increasing densities. Fig. 3 shows that, as expected, GMR has a lower performance in very sparse scenarios because of the extensive use of face routing, due to a high number of void areas. Nevertheless, since usually WSNs are densely deployed, this should not be a concern in real deployments. As the figure shows, the higher the density, the better the performance obtained by GMR. As expected, when the mean density is high enough to let the protocol work in greedy mode most of the time, the cost of multicast trees found out by GMR is much lower than those of traditional ad hoc routing protocols, since they create shortest paths towards each destination.

Fig. 4(a) shows that GMR outperforms PBM regardless of the values of the  $\lambda$  parameter and in a variety of network densities. The higher the density, the bigger the improvement obtained over PBM. As expected, the value of  $\lambda$  has an influence on PBM's performance. For values of  $\lambda$  close to 0 PBM tends to create shortest paths. However, those paths are not optimal in terms of the number of transmissions. When  $\lambda$  approaches

1, PBM tries to minimize the number of transmissions regardless of the length of the paths. The problem with  $\lambda = 1$  is that by not considering progress to destinations, paths may become very large in terms of number of hops and overall number of transmissions. As we can see, even for those values of  $\lambda$ , GMR manages to outperform PBM because GMR not only minimizes the number of transmissions but at the same time the mean path length. We can see in Fig. 4(a) that GMR achieves an improvement between 2%–35% for a mean density of five neighbors per node. The higher the density the biggest the improvement obtained by GMR. For instance, for a mean density of ten neighbors per node, the improvement ranges between 30%–95%. In general, GMR is 2%–25% better than the best case of PBM (using  $\lambda = 0.2$ ) and 38%–95% better than the worse case of PBM (using  $\lambda = 1$ ). The key to that improvement is the goodness of the neighbor selection function used by GMR. For higher densities, most of the routing is performed in greedy mode (almost no face routing is required). Thus, the better neighbor selection function achieves its higher advantage.

For a different number of destinations, results follow a similar trend. The higher the number of destinations, the higher the number of transmissions required. However, in general, the improvement achieved by GMR compared with PBM is within the same percentages.

Fig. 4(b) shows the rate at which destinations are reached for a mean density of 7. From it we can assess the efficiency of the transmissions for each tested protocol. The lower the number of transmissions needed to reach a certain number of destinations, the better the efficiency of the protocol. As we showed before, GMR clearly needs fewer messages to reach 100% of destinations than PBM using different  $\lambda$  values. This figure also shows that GMR makes a very efficient use of its transmissions because for a given number of transmissions, the percentage of destinations reached is higher. The reason for that is that GMR's neighbor selection algorithm manages to efficiently delay the splitting of paths to approximate the minimum number of transmissions trees, while still providing a good advance toward destinations. Fig. 5 illustrates the different behaviors of GMR as density increases. As expected, the higher the density of the network, the better the performance obtained. The reason is that with higher densities, the probability of entering into face mode drops rapidly. Hence, the neighbor selection algorithm becomes even more effective.

Summing it up, GMR behaves always better than the best case for PBM. Unlike PBM, GMR does not need to adjust any parameter. GMR is a self-adapting algorithm that is able to find out only with local information the right point in which to split multicast messages in order to maintain a good tradeoff between path length and total number of messages transmitted.

### D. Stretch Factor

Usually, achieving a low number of data packet transmissions requires deviating from SPTs to allow many destinations to share the same paths as much as possible. For some applications with hard delay requirements, a big deviation from the SPT may not be desirable. To account for the deviation from a SPT, we compute the stretch factor for each of the protocols being evaluated.

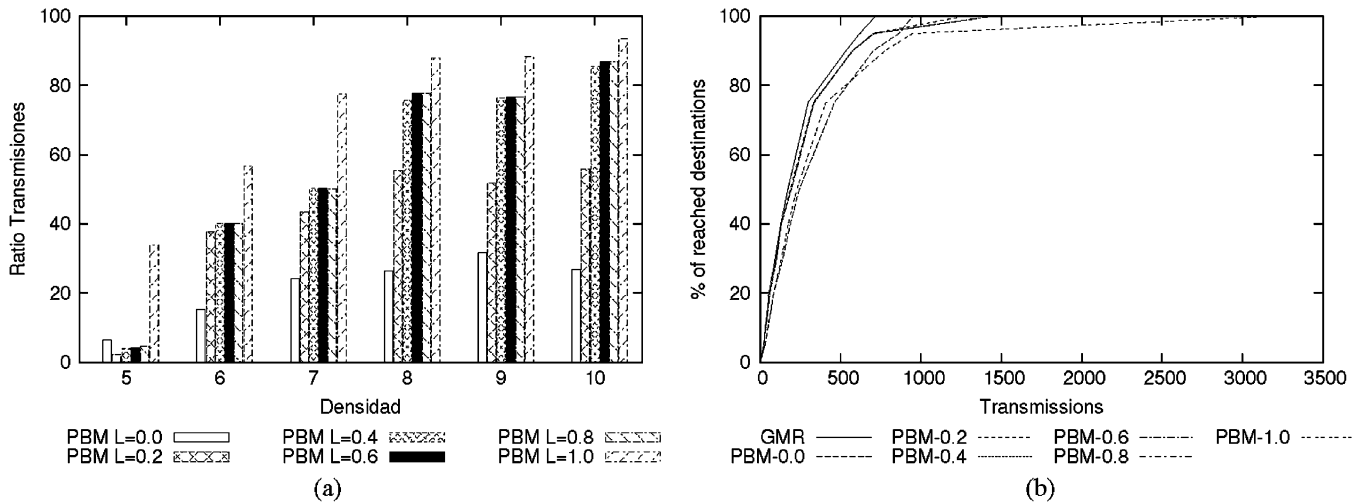


Fig. 4. Improvement in number and efficiency of transmissions for 25 destinations. (a) Percentage of improvement over PBM. (b) Efficiency of Tx density= 7.

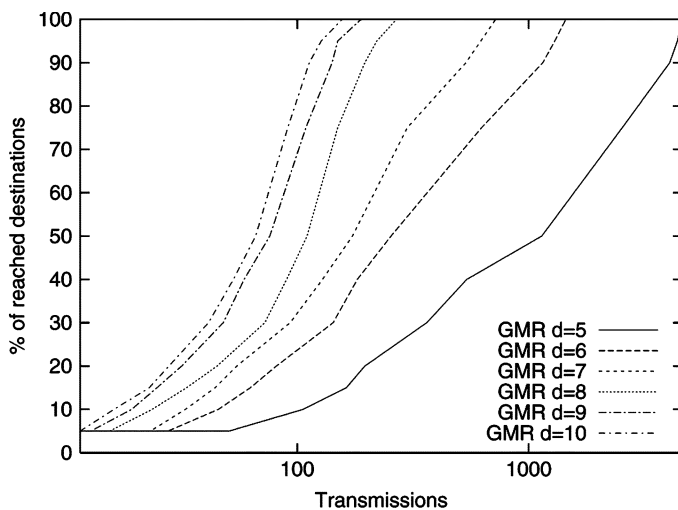


Fig. 5. Efficiency of Tx for GMR for 25 destinations and different densities.

For these experiments, we use a mean density of ten neighbors per node to make the comparison as fair as possible. The reason is that for lower densities these protocols fall into face routing very often. So, the overall results for those cases will be mainly determined by the number of times the protocol entered into face routing rather than the neighbor selection itself.

Figs. 6(a) and (b) compare, in absolute and relative values, respectively, the performance between the protocols for different values of  $\lambda$  and varying number of destinations. As expected, the lower the value of  $\lambda$ , the better the stretch factor PBM achieves. This is due to the fact that when  $\lambda$  approaches 0, PBM tries to compute a SPT. In general, GMR is getting better stretch factor values than PBM for most of the tested configurations. Only in a few configurations in which  $\lambda = 0$ , GMR does not outperform PBM. The reason in those few cases is that the goal of GMR is not to find the shortest path but a good tradeoff between the length of the paths and the overall consumption of network resources. As we can see in the figures, the tradeoff obtained by GMR is very good because it is superior to PBM for

every  $\lambda \geq 0.2$ , while still needing a lower number of transmissions. In Fig. 6(b), we can see that the improvement in stretch factor achieved by GMR over PBM is bigger than 40% for every  $\lambda > 0.2$ . For lower  $\lambda$  values, GMR still performs around 15% better than PBM except in some particular cases.

#### E. Neighbor Selection Computation Time

As we showed before, the computational cost of PBM is exponential in  $n$ , while the one of GMR is polynomial in  $n$  and  $D$ . Given that the average number of neighbors in our experiments is small (the highest density is 10), this may not translate into better performance for GMR. However, we show in our experimental results that the average case performance of GMR is very consistent and scalable across network parameters.

To measure the computational time, we execute one single step of the routing algorithm. This step makes the first neighbor selection (the initial one at the source node). For each set of parameters, we executed the single step 50 times and measured the mean time of those 50 executions. Fig. 7 shows the results for the case that would be the best case for PBM.

We can see that the mean execution time for GMR is stabilized between 2–3 s regardless of the density of the network. However, PBM's mean execution time grows exponentially with density. This is because the proposed neighbor selection algorithm is able to compute a good set of neighbors efficiently, thanks to the heuristic for subset merging. In fact, the algorithm shows a good performance in the mean case.

## VI. CONCLUSION AND FUTURE WORK

We introduced GMR, a geographic multicast routing protocol for WSNs. GMR uses a cost-based neighbor selection at each routing step, allowing it to find a good tradeoff between the optimality of the multicast tree, and the efficiency of data delivery. GMR is a fully localized algorithm and does not require the use of extensive broadcast to route messages to a set of destinations. We assume the position of destinations is known by the source. In fact, GMR has been designed for scenarios with a limited number of receivers.

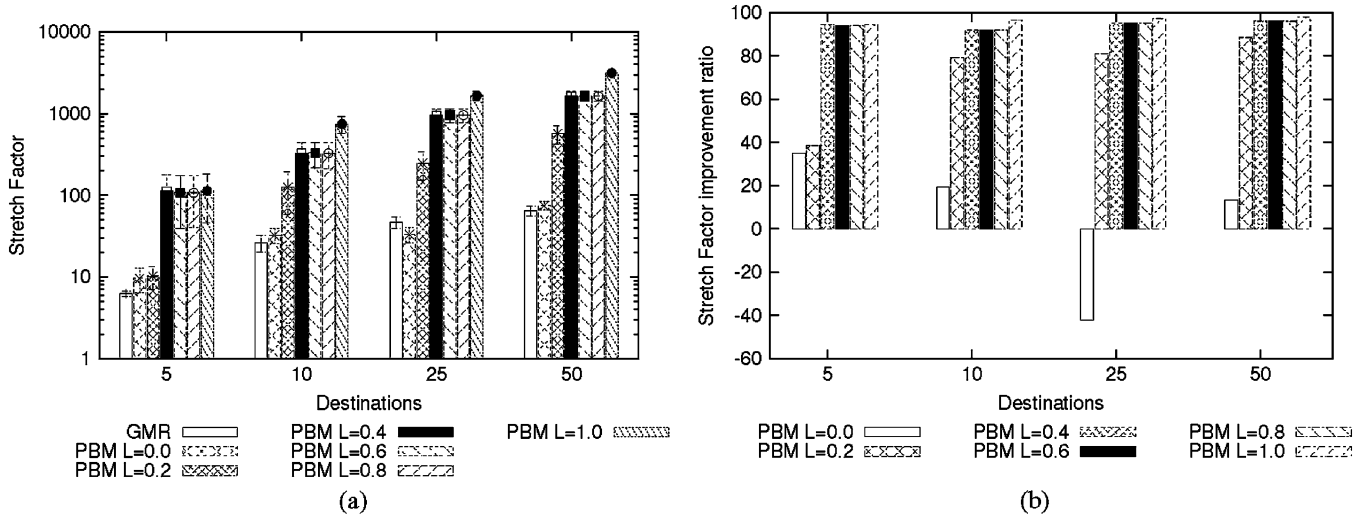


Fig. 6. Stretch factor comparison for a mean density of ten neighbors per node. (a) Stretch factor density = 10. (b) Percentual stretch factor improvement over PBM.

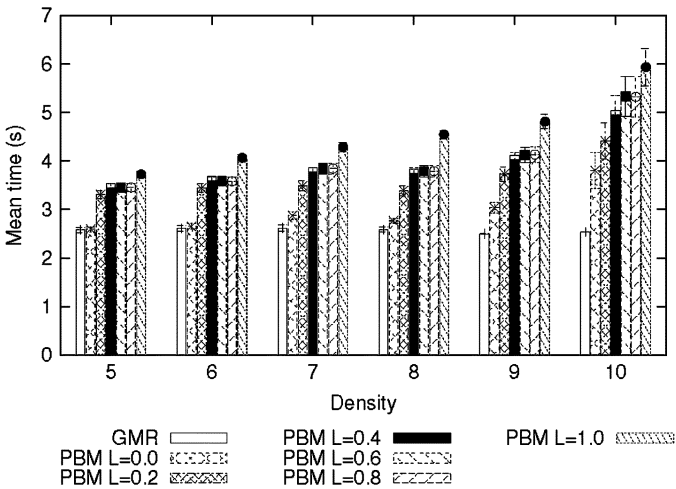


Fig. 7. Computation time for varying destinations.

Computing the optimal multicast tree in terms of bandwidth consumption is similar to finding the multicast tree with the minimum number of forwarding nodes, and was shown to be NP-complete (see [5]). Thus, the use of the greedy forwarding scheme is fully justified. In addition, using this greedy neighbor selection is also a good idea for sensor networks because the routing decision at each single node is performed based on the current network topology. Thus, GMR is able to adapt to topological changes due to sensors operating in a duty cycle. However, individual sensors are assumed to be static.

One of the key aspects of GMR is the cost-aware neighbor selection function. The reason is that the shape of the tree is one of the key parameters governing the overall optimality of the tree. Given that the complexity of testing all possible neighbor subsets grows exponentially, we proposed an heuristic algorithm which has shown to be efficient compared with PBM. In addition, our performance results show that GMR outperforms PBM in terms of the efficiency of the resulting trees, the deviation from SPTs (stretch factor), and on the computational time.

For future work, we plan to analyze different alternatives to optimize multicast face routing. In addition, we also plan to work on GMR variants (based on slightly different cost functions) to deal with energy efficiency and realistic physical layers. Finally, GMR works perfectly when the number of receivers is small or medium. Dealing with larger number of receivers needs to consider new approaches to the problem.

ACKNOWLEDGMENT

The authors thank the anonymous reviewers for their comments and insights.

REFERENCES

- [1] R. Fonseca, S. Ratnasamy, D. Culler, S. Shenker, and I. Stoica, "Beacon vector routing: Scalable point-to-point in wireless sensor networks," *Intel Research, IRB-TR-04*, vol. 12, 2004.
- [2] A. Carus, A. Urpi, S. Chessa, and S. De, "GPS-free coordinate assignment and routing in wireless sensor networks," in *Proc. 24th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM '05)*, Mar. 2005, vol. 1, pp. 150–160.
- [3] P. M. Ruiz and I. Stojmenovic, T. Gonzalez, Ed., "Cost-efficient multicast routing in ad hoc and sensor networks," in *Handbook on Approximation Algorithms and Metaheuristics*. London, U.K.: Chapman & Hall, 2006.
- [4] S. Giordano, I. Stojmenovic, and L. Blazevic, "Position based routing algorithms for ad hoc networks: A taxonomy," *Ad Hoc Wireless Networking*, pp. 103–136, 2004.
- [5] P. Ruiz and A. F. Gomez-Skarmeta, "Approximating optimal multicast trees in wireless multihop networks," in *Proc. 10th IEEE Symp. Comput. Commun. (ISCC'05)*, La Manga del Mar menor, Spain, Jun. 2005, pp. 686–691.
- [6] R. Zhang, H. Zhao, and M. A. Labrador, "The anchor location service (ALS) protocol for large-scale wireless sensor networks," in *Proc. 1st Int. Conf. Integr. Internet Ad Hoc and Sensor Networks (InterSense 06)*, 2006, pp. 18–.
- [7] J. Kuruvila, A. Nayak, and I. Stojmenovic, "Hop count optimal position based packet routing algorithms for Ad hoc wireless networks with a realistic physical layer," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 6, pp. 1267–1275, 2005.
- [8] S. J. Lee, W. Su, and M. Gerla, "On-demand multicast routing protocol in multihop wireless mobile networks," *Mobile Networks and Appl.*, vol. 7, no. 6, pp. 441–453, 2002.
- [9] J. Garcia-Luna-Aceves and E. L. Madruga, "The core-assisted mesh protocol," *IEEE J. Sel. Areas Commun., Special Issue on Ad-Hoc Networks*, vol. 17, no. 8, pp. 1380–1394, Aug. 1999.

- [10] R. Vaishampayan and J. Garcia-Luna-Aceves, "Protocol for unified multicasting through announcements(PUMA)," in *Proc. 1st IEEE Int. Conf. Mobile Ad-Hoc and Sensor Syst. (MASS '04)*, 2004.
- [11] E. M. Royer and C. E. Perkins, "Multicast operation of the Ad-hoc on-demand distance vector routing protocol," in *Proc. 5th ACM/IEEE Int. Conf. Mobile Comput. Netw. (MobiCom'99)*, Sep. 1999, pp. 207–218.
- [12] J. G. Jetcheva and D. B. Johnson, "Adaptive demand-driven multicast routing in multi-hop wireless Ad hoc networks," in *Proc. 2th ACM Int. Symp. Mobile Ad Hoc Netw. Comput. (MobiHoc'01)*, 2001, pp. 33–44.
- [13] C. W. Wu and Y. C. Tay, "AMRIS: A multicast protocol for ad hoc wireless networks," in *Proc. Military Commun. Conf. (MILCOM 99)*, Nov. 1999, vol. 1, pp. 25–29.
- [14] L. Ji and M. S. Corson, "Differential destination multicast-A MANET multicast routing protocol for small groups," in *Proc. 20th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM'01)*, 2001, pp. 1192–1202.
- [15] L. Ji and M. S. Corson, "A lightweight adaptive multicast algorithm," in *Proc. 41st Annu. IEEE Global Telecomm. Conf. (GLOBECOM'98)*, Dec. 1998, pp. 1036–1042.
- [16] V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *Proc. 16th Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM '97)*, Apr. 1997, pp. 1405–1413.
- [17] T. Ballardie, P. Francis, and J. Crowcroft, "Core based trees (CBT)," in *Proc. Conf. Commun. Architectures, Protocols and Applications (SIGCOMM '93)*, 1993, vol. 23, pp. 85–95, pt. 4.
- [18] A. Mizumoto, H. Yamaguchi, and K. Taniguchi, "Cost-conscious geographic multicast on MANET," in *Proc. 1st IEEE Commun. Soc. Conf. Sensors, Mesh and Ad Hoc Commun. Netw. (SECON '04)*, Oct. 2004, pp. 44–53.
- [19] G. G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," Univ. Southern California, Information Sciences Institute, Los Angeles, CA, Tech. Rep. ISI/RR-87-180, Mar. 1987.
- [20] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," *Wireless Netw.*, vol. 7, no. 6, pp. 609–616, 2001.
- [21] H. Takagi and L. Kleinrock, "Optimal transmission ranges for randomly distributed packet radio terminals," *IEEE Trans. Commun.*, vol. 32, no. 3, pp. 246–247, Mar. 1984.
- [22] H. Fler, M. Mauve, H. Hartenstein, C. Lochert, D. Vollmer, D. Herrmann, and W. Franz, "Position-based routing in Ad-hoc wireless networks," in *Inter-Vehicle-Communications Based on Ad Hoc Networking Principles—The FleetNet Project*. Karlsruhe, Germany: Universitätsverlag Karlsruhe, Nov. 2005, pp. 117–143.
- [23] I. Stojmenovic, "Position-based routing in Ad hoc networks," *IEEE Commun. Mag.*, vol. 40, no. 7, pp. 128–134, 2002.
- [24] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris, "A scalable location service for geographic Ad hoc routing," in *Proc. 6th Annu. ACM/IEEE Int. Conf. Mobile Computing and Networking (MobiCom '00)*, 2000, pp. 120–130.
- [25] D. Liu, I. Stojmenovic, and X. Jia, "A scalable quorum based location service in ad hoc and sensor networks," in *Proc. 3rd IEEE Int. Conf. Mobile Ad-Hoc and Sensor Syst. (MASS '06)*, Oct. 2006.
- [26] L. Lazos and R. Poovendran, "SeRLoc: Robust localization for wireless sensor networks," *ACM Trans. Sensors Netw.*, vol. 1, no. 1, pp. 73–100, 2005.
- [27] M. Mauve, H. Füßler, J. Widmer, and T. Lang, "Position-based multicast routing for mobile ad-hoc networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 53–55, 2003.
- [28] M. Transier, H. Füßler, J. Widmer, M. Mauve, and W. Effelsberg, "Scalable position-based multicast for mobile ad-hoc networks," *First Int. Workshop on Broadband Wireless Multimedia: Algorithms, Architectures and Applications (BroadWim '04)*, Oct. 2004.



Engineering (DIIC), UMU.

**Juan A. Sanchez** received the B.Sc. degree in computer science from the University of Murcia (UMU), Murcia, Spain, in 1999. He is currently working towards the Ph.D. degree at UMU.

For over four years, he was involved in several national and international R&D projects in the field of wireless and mobile networks, multimedia applications, and next-generation networks as a member of the R&D Department of Agora Systems S.A. Currently, he is an Adjunct Professor at the Department of Communications and Information



**Pedro M. Ruiz** (M'02) received the B.Sc., M.Sc., and Ph.D. degrees in computer science from the University of Murcia (UMU), Murcia, Spain, in 1999, 2001, and 2002, respectively.

He is a Ramon y Cajal Researcher in the Department of Information and Communication Engineering (DIIC), UMU. He has also held post-doctoral research positions at the International Computer Science Institute (ICSI), University of California, Berkeley, Kings College London, and the University of California at Santa Cruz. During these

years, he has acted as principal investigator in several projects mainly funded by the European Union, Spanish government, and private companies. He is on the Technical Committee of several conferences, and has served as a reviewer for major IEEE journals and conferences. He has published over 60 refereed papers in international journals and conferences. His main research interests include sensor networks, mobile and ad hoc wireless networks and distributed systems.

Dr. Ruiz is a member of the IEEE Communications Society. He is on the Editorial Board for the *International Journal on Parallel, Emergent, and Distributed Systems*.



**Jennifer Liu** received the B.Sc. degree in electrical engineering from Tsinghua University, Beijing, China, in 1996. She is currently working towards the M.S. degree at the University of Ottawa, Ottawa, ON, Canada.

She is a Senior Engineer at WindRiver and had worked at Nortel and Motorola as a wireless system engineer. She is interested in network routing, wireless network and simulation.



**Ivan Stojmenovic** received the Ph.D. degree in mathematics from the University of Zagreb, Zagreb, Croatia, in 1985.

He held positions in Serbia, Japan, U.S., Canada, France, and Mexico. He published over 200 different papers, and edited four books on wireless, ad hoc and sensor networks, and applied algorithms with Wiley and the IEEE.

Dr. Stojmenovic is currently an Editor of several journals including the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS. He is founder and editor-in-chief of three journals. He is in the top 0.56% most cited authors in Computer Science (Citeseer 2006). One of his articles was recognized as the "Fast Breaking Paper" for October 2003 (as the only one for all of computer science), by *Thomson ISI Essential Science Indicators*.