Efficient Geocasting to Multiple Regions in Large-Scale Wireless Sensor Networks

Cuong Truong, Kay Römer
Institute of Computer Engineering, University of Lübeck, Germany
{truong, roemer}@iti.uni-luebeck.de

Abstract—**Recently, large sensor networks with several thousands of nodes are being deployed over large geographic areas in the context of smart city projects. In these settings, there is often a need to send a message to all sensors contained in one of multiple geographic regions, for example, to query for a free parking spot in several streets. We present Recursive Multi-region Geocasting (RMG), a novel multi-region geocast routing protocol which addresses the problem of delivering data from a source to multiple remote geocast regions in large-scale wireless sensor networks. The key idea is to treat a remote group of geocast regions as a point destination and forward data packets along a straight line towards the group, until a *division point* at which the group is divided, and the packet is forwarded towards the sub-groups in the same fashion. RMG is lightweight as no state has to be maintained at the nodes and the computations are simple. Simulation shows that our protocol achieves lower path length overhead and network relay load while incurring less computation overhead when compared to state-of-the-art protocols.**

I. Introduction

While originally most sensor network deployments are rather small with tens or few hundreds of nodes [1], there is a recent trend towards much larger scale deployments with several thousands of nodes being deployed over large geographical areas. In particular, the Internet of Things envisions globally interconnected sensor networks, and in the "smart cities" application domain we are witnessing first actual large-scale deployments. For example, the FastPrk smart parking solution[1] relies on a mesh network of several thousand parking spot occupancy sensors that are deployed over parking areas of the city of Barcelona, to help drivers find empty parking spots to minimize search traffic and time. For a similar purpose, the SFPark project[2] has deployed sensor nodes on 7000 out of 28.800 parking spots of the city of San Francisco, USA. At an even bigger scale, the U-City project [2] is being deployed in South Korea with the ultimate vision of creating a ubiquitous society where the urban environment is soaked with ubiquitous sensor networks and RFID systems.

In such systems, there is often a need to send a message to sensor nodes located in multiple geographic regions. With smart parking systems, for example, a user would send a request for a free parking spot to all parking spot sensors located in certain streets, where each street defines a geographical region. In a smart city, a request to locate a lost object (equipped with a wireless tag whose presence can be detected by close-by sensor nodes) would be sent to sensors in multiple geographic regions where the user usually spends time (i.e., home, office, streets on the way from home to office, gym, restaurants) [3].

[1]www.worldsensing.com
[2]http://sfpark.org/about-the-project/

The underlying problem is *geocasting a message from a source to multiple geographic regions*, respectively to all sensors located in one of those regions. Although geocasting in general is a well-studied problem, most existing work focuses on geocasting to either a *single destination node* at a given location, to *few destination nodes* where the location of each destination node is given, or to a *single geographic region* respectively all nodes located in this single region. Although one could invoke those protocols repeatedly to send the same message to multiple regions, this would not be efficient, especially in sensor networks with their severely limited energy, networking bandwidth, and computational resources.

In this paper, we therefore study the problem of multi-region geocast routing to a set of remote geocast regions in geographically large-scale WSN. Our contribution is two-fold. Firstly, we design the Recursive Multi-region Geocasting (RMG) protocol to address the above problem of multi-region geocasting. RMG is tailored to large-scale networks and large numbers of destination nodes. Secondly, we evaluate RMG and compare it to state-of-the-art protocols. We find that RMG (i) minimizes the total number of transmissions needed for successfully delivering a packet to save network bandwidth and energy; (ii) minimizes the path length between the source node and all destination regions; and (iii) minimizes the computation overhead at intermediate nodes along the path.

The paper continues with a discussion of related work before introducing our approach along with its underlying models and assumptions. Then, we introduce the RMG protocol and evaluate it.

II. Related Work

We structure the discussion of related geocasting approaches according specification of destinations: a set of nodes, a single region, and finally – the focus of our work – multiple regions.

A. Geocasting to a Set of Nodes

Approaches in this class support the delivery of a message from a source to a set of destinations nodes where the geo-location of each destination node is given. Specific protocols have been designed where the destination nodes are a set of base stations [4], actuators [5], or other sensors [6], [7]. The GMR protocol in [4] is somewhat similar to our work as it divides the destination group into subgroups. For each forwarding node, a minimal subset of the node's neighbours that promises most progress towards the destinations is selected as the next relay of the packet. The selection is performed based on the cost-over-the-progress ratio. A drawback of GMR is that the computation of such a minimal subset is performed at all intermediate relay nodes along the routes from the source to all destinations. This is expensive especially when the network density is high and the routes are long (e.g., in geographically large scale WSN). Our approach, in contrast, performs a lighter computation only when a particular condition is violated, therefore saving processing resources.

However, all of the above protocols have been designed for small-scale networks and for a small set of destinations (whose locations all have to be included in the message header). In contrast, the multi-region geocast problem does not consider individual destination sensors but geocast regions containing many sensors each. Specifically, our solution is tailored for regions located remotely from the source in geographically large-scale sensor networks. Comparison results in section V between our protocol RMG and the above GMR protocol show that RMG excels in such settings.

B. Geocasting to a Single Region

Single region geocast routing in WSN deals with the problem of delivering data packets from a source to all sensor nodes located in a particular geographical region. The protocol in [8] uses flooding with restricted flooding zone to deliver a packet to the geocast region. Although flooding zones reduce bandwidth usage when compared to conventional flooding, it does not scale in geographically large-scale WSN. Moreover, the protocol will fail in the presence of network partitions within the flooding zone.

To improve scalability and reliability, geographic unicast routing is used. The packet is unicasted to a node located inside the geocast region, from which it is further disseminated to all other nodes in the region. If a void is encountered during unicasting, face routing (e.g., [9]) is invoked to detour the packet around the void, thus guaranteeing the arrival of the packet at the geocast region. Representatives of this approach are GEAR [10] and GFG [11], which differ from each other in the way they disseminate the packet inside the geocast region.

In-region packet dissemination is challenged by network partitions (e.g., due to sparse topology or obstacles). To overcome this issue, GFPG [11], VSF [12], and [13] propose to include out-region nodes in packet dissemination. GFPG uses face routing on the planar faces intersecting the region border in addition to flooding inside the region to reach all nodes. [13] proposes to route the packet to the entrance zone which is an internal border ring of the region to make sure the packet reaches every side of the region thus all partitions (if any) are reachable. The idea in VSF is to repeatedly merge all faces intersecting with the region to obtain a large virtual surrounding face that covers the region. The nodes on this face are then traversed for disseminating the packet into the geocast region.

C. Geocasting to Multiple Regions

The protocols reviewed above mainly focus on one geocast region. For multiple geocast regions there are few works including [14], [15], [16], [17]. The work in [15] relies on flooding to discover routes to the geocast regions using "route discovery" and "route reply" messages. This approach clearly does not scale with network node density and quantity. Also relying on flooding but with a hierarchical approach, [16] groups nodes into clique-clusters. A super cluster head is elected among these cluster heads. The packet is sent to super cluster head which then floods it to all clique-cluster heads. Each clique-cluster head then forwards the packet to its members if they belong to one of the regions. This approach is supposed to be energy efficient (as claimed by authors), but it comes with the extra overhead of management and maintenance of the clusters. The protocol in [17] geographically partitions the deployment area of the network into disjoint and equally sized cells, and performs geocast routing on top of these cells' managers. Again, cell management and maintenance should be considered as extra overhead for this protocol.

The closest work to ours is the GGP protocol in [14]. GGP employs the concept of Fermat point, which is the point within a triangle from which the sum of distances to the vertices of the triangle is minimized. To route a packet to a pair of geocast regions, the packet is first greedily forwarded to the precomputed Fermat point of the triangle formed by the packet's source and two centres of the regions, is then duplicated and forwarded to the two regions. If more than two geocast regions are given, e.g., for three regions A, B, C, GGP computes the Fermat point F_1 for the triangle formed by the source and the centres of A and B, then computes F_2 for the triangle formed by the source, C's center, and F_1. The packet is routed to F_2 first where it is duplicated and routed to F_1 and C. When the packet reaches F_1 it is duplicated again and routed to A and B. The same principle is applied for a larger number of regions. We can see that GGP delivers a packet to the geocast regions in a sequential fashion which may result in a long path for the packet to reach all regions. In contrast, our approach is to group closely located regions and forward the packet along a forwarding line towards all group's members in parallel, thus requiring much shorter paths. Simulation results in section V validate this claim.

III. Assumptions and Approach

We outline the basic approach of our multi-region geocast protocol RMG using an analogy with the real world, after presenting basic assumptions and models underlying RMG.

A. Network Model

We study the multi-region geocast routing problem in geographically large-scale WSN. For the ease of exposition, we first assume that each node has a fixed circular transmission range, which is identical for all nodes. The dynamicity of the

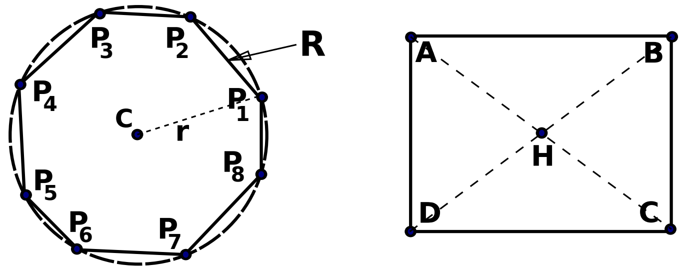

Fig. 1: Description of geocast regions

network topology is the consequence of node mobility, failures and additions. The radio link is perfectly bidirectional, i.e., two nodes that lie within each other's transmission range can exchange data without packet loss. Note that our protocol will still work with relaxation of these assumptions, as we will discuss in detail in section V.

We also assume that all nodes are aware of their locations, e.g., through GPS receivers, or by employing a distributed location discovery algorithm, such as in [18]. Each node also knows the locations of its neighbours via a simple Hello protocol. Moreover, the information about the shape of the geocast regions are known to the source before the data packet is sent out.

B. Shape of Geocast Regions

Many single-/multi-region geocasting protocols such as in [8], [11], [14], either treat the shape of the geocast regions as a convex closed polygon (square, rectangle, circle), or as in [12], as a concave closed polygon shape. The authors implicitly assume that a closed polygon is described by a set of points and some extra information, e.g., a circle is given by its center and radius, which are included in the packet header.

There is a trade-off between how detail a geocast region can be described and the amount of information included in the packet header (storage overhead). For example, the region R in Fig. 1 could either be more accurately described by 8 points $P_1, .., P_8$ which requires a storage overhead of 16 real numbers, or be less accurately described by the circle centring at C whose radius is r which requires only 3 real numbers. The choice for this trade-off depends on particular applications. Our proposed protocol is flexible and supports any type of convex geometric shape as long as the description of the geocast regions and a formula to compute their centres are given.

Throughout this paper we use rectangles as geocast regions as an example since they provide a good trade-off between detail and overhead, as well as geometric flexibility. In real life, many scenarios can benefit from this approximation, e.g., a building, a subregion in a forest, a coast region by a sea, etc, because their shape can naturally be decomposed into a set of rectangles.

We assume that the message's source can consult a service that resolves a geocast region into a set of rectangles described by their width, height, 1 corner, and the polar angle formed between the polar axis and its width, e.g., the rectangle $ABCD$ in Fig. 1 is described by its corner (x_A, y_A), its width $|AB|$, its height $|AD|$, and a polar angle of 0. Thus, 5 real number are required to describe the region. Multiple geocast regions will then be represented as a set of regions of rectangular shape, which are included in the packet header before the packet is sent out. To send a packet to a region, e.g., $ABCD$, we forward the packet towards the region's center (x_H, y_H). Thus from now on, we refer to the region's center as the destination of the data packet.

C. Recursive Forwarding Approach

We illustrate the operation of our protocol with the following analogy. Imagine you live in Berlin and are visiting your friends living in Paris. In Berlin, you do not see any detail of Paris because the city is too far away. To you, Paris looks just like a point, thus it does not make much sense for you to spend time and effort to calculate the paths to each an every friend in Paris at departure in Berlin, because the travel distance is dominated by the distance between the two cities. So you decide to travel to Paris first, before you plan the visit of your friends. The shortest way to get to Paris is obviously the straight line connecting the two cities. When arriving at the entrance of Paris (the city gate that you enter from the highway), you know in which districts your friends live. Since the districts are still far apart, again you decide to travel to the districts before planning the visit of friends. The shortest path to a district is again the connecting line between its center and the entrance of Paris. This strategy is recursively repeated until you can directly see the house of a friend (e.g., from an end of the street where your friend's house resides). Now that you can see the house, you approach it and knock on the door.

Consider a group of geocast regions (destinations) that are located closely to each other, and a source is transmitting data to the group. The distance from the source to the group is much larger than the average distance between members of the group. Applying the spirit of the above analogy, we forward data packets along the straight line connecting the source and a *division point*, which we define, similarly to the entrance of Paris in the analogy, as the point where new routing decisions have to be made. At this point, the destination group is divided, and the packet is forwarded to the sub-groups in the same manner. We call that straight line the *forwarding line*.

The advantage of this approach is two-fold. Firstly, it is lightweight because we only have to compute the division points and perform destination group division at some intermediate nodes during the delivery of a packet. Secondly, it saves bandwidth because instead of sending a packet separately towards each an every destination (i.e., n transmissions), we only send the packet once along the forwarding line towards all the destinations. The approach, however, raises three questions: (i) how do we compute the forwarding line; (ii) how to calculate a division point; and (iii) how do we divide a group of destinations into sub-groups. We will address these questions in the next section.

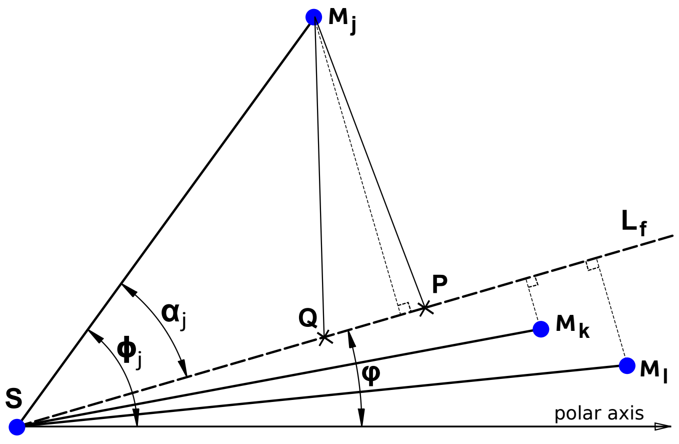

Fig. 2: The recursive forwarding approach

IV. Recursive Multi-region Geocasting

We describe in detail the main two elements of RMG: the computation of the forwarding line and the division of a set of destination regions into subsets. Finally, we outline how these two elements are integrated into a complete protocol.

A. Forwarding Line & Division Point

To understand the computation of the forwarding line and the division point, we take a look at an illustration of our approach in Fig. 2, where the source S is sending data packets to a remote group of geocast regions $G = \{M_j, j = 1..m\}$, where M_j is a center point of a rectangular geocast region. The dashed line L_f connecting the source S and the division point P is the forwarding line along which data packets are sent. To compute P, we need to compute φ and $|SP|$, where $|\bullet|$ stands for Euclidean distance.

Consider a destination M_j. If we assume relay cost is proportional to $|SM_j|$ then the minimum relay cost we could achieve by sending a packet along L_f is when L_f coincides with the line $S\vec{M}_j$ i.e., $|\phi_j - \varphi| = 0$. To minimize relay cost to all M_j by using only L_f, we need to find a φ such that $\sum_{j=1}^{m} |\phi_j - \varphi|$ is minimized. Hence

$$\varphi = \frac{1}{m}\sum_{j=1}^{m}\phi_j \tag{1}$$

Since we want to use only L_f to send a data packet to all M_j to minimize the total relay cost, we want to place the division point P on L_f to be as close as possible to all M_j. This means to find P such that $\sum_{j=1}^{m} |PM_j|$ is minimized. However, if the data packet is routed via P on L_f to all M_j, the individual relay cost to each M_j is higher than sending the packet along SM_j. We define the individual relay overhead of the transmission of the data packet from S to M_j via P as

$$\gamma_P^j = 1 - \frac{|SM_j|}{|SP| + |PM_j|} \tag{2}$$

Consider a point $Q \in L_f$. Due to the triangles inequality, if $|SQ| < |SP|$ then $\gamma_Q^j < \gamma_P^j$, which means reducing individual relay cost would increase total relay cost and vice versa. A good trade-off would be to place P at a position on L_f such that P is closest to all M_j and S, i.e., to minimize the sum of the distances from P to all M_j and S. Such position can be found by minimizing:

$$|SP|^2 + \sum_{j=1}^{m}|PM_j|^2 \tag{3}$$

There are two reasons why we use square instead of absolute value in this situation. First, square is continuously differentiable therefore is helpful when we want to find a minimum. Second, square emphasizes larger differences thus an asymmetric minimum would be avoided.

Now according to the law of cosines we have:

$$|PM_j|^2 = |SM_j|^2 + |SP|^2 - 2|SM_j||SP|\cos\alpha_j \tag{4}$$

Supplying (4) into (3) and expanding, we obtain:

$$(m+1)|SP|^2 - 2|SP|\sum_{j=1}^{m}|SM_j|\cos\alpha_j + \sum_{j=1}^{m}|SM_j|^2 \tag{5}$$

Taking the first derivative of (5) with respect to $|SP|$ and setting it to zero, we have:

$$2(m+1)|SP| - 2\sum_{j=1}^{m}|SM_j|\cos\alpha_j = 0$$

Since $2(m+1) > 0$, (3) is minimized when

$$|SP| = \frac{1}{m+1}\sum_{j=1}^{m}|SM_j|\cos\alpha_j \tag{6}$$

Equation (6) says that instead of separately forwarding the packet from S to each M_j, we can achieve a good trade-off between individual and total relay cost by forwarding the packet from S, along L_f until P, then separately forwarding the packet to each M_j.

The point P is our division point and acts as the “entrance” of the city in our Berlin-Paris example, while the group of geocast regions acts as the city.

B. Group Division

In this subsection we discuss when and how we divide a group of geocast regions into subgroups. According to the philosophy of our approach, we use the group's forwarding line to forward a packet as long as the group still looks “small”, until the group looks “big” such that it cannot be considered a point destination any more. The group therefore needs to be divided.

Considering the Berlin-Paris analogy, the size of Paris as perceived by a traveller depends on the ratio between the diameter of the city and the distance from the traveller to the city. Similarly, to quantify how small a group G looks from the perspective of a relay node, we denote $\delta_G \in [0, 1]$ as the *smallness* of G such that

$$\delta_G = \frac{2}{\pi}\max_{j\in G}\alpha_j \tag{7}$$

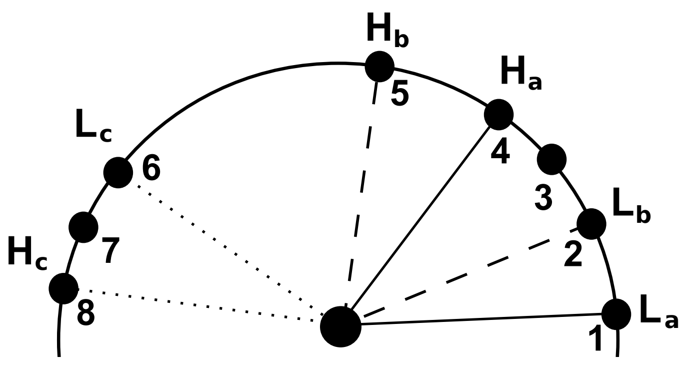

Fig. 3: The GMGD algorithm

Where α_j is the angle enclosed by the forwarding line and the line towards geocast region M_j. Now given an application-defined threshold $\delta_{th} \in (0, 1]$, G is said to look small if

$$\delta_G < \delta_{th} \tag{8}$$

Based on inequality (8), we propose the greedy multi-geocast group division (GMGD) algorithm. The idea is to divide G into a minimum number of subgroups, each of which has the maximum number of destinations that meet condition (8).

We explain the algorithm by illustration (see Fig. 3). We first sort the members of G in increasing order of their polar angles and assign an integer number corresponding to their order. Then we pick the member whose polar angle is smallest (number 1) and iterate over G in increasing order of polar angles, until a member (number 4) such that condition (8) is violated i.e., $\delta_{G_{\overline{1,4}}} \geq \delta_{th}$ (where $G_{\overline{1,4}}$ is the subgroup consists of the just visited 4 members). We insert this pair of indices, i.e., $(L, H) = (1, 4)$ into an index list $iList$ that is sorted by decreasing number of members between the pair (i.e., nodes with polar angles greater than or equal to L's and smaller than H's). Note that each index pair (L, H) will define a subgroup. We repeat this process for each member until all 8 members of the group are picked.

After the $iList$ is built, we iterate over the entries of $iList$ and add a pair (L_i, H_i) $(i = a, b, c, ...)$ to a final list $fList$ if the pair does not overlap with any pair that is already in $fList$. For example in Fig. 3, two pairs (L_a, H_a) and (L_b, H_b) are overlapped but (L_a, H_a) and (L_c, H_c) are not. Note that since $iList$ is sorted by decreasing subgroup size, biggest subgroups are always added to $fList$ first. The iteration is done when all entries of $iList$ have been visited.

To build the list of subgroups, we iterate all entries (L_j, H_j) $(j = a, b, c, ...)$ of $fList$. Each entry corresponds to a subgroup whose members are the members of G that are between L_j and H_j.

The detail of GMGD is given in Algorithm 1 (written in pseudo Java programming language). The function *violateFrom*(L) returns the smallest index H such that the subgroups formed between two indices L and H would violate condition (8). The function $overlap(L, H, fList)$ checks if the pair (L, H) overlaps any pair of indices in $fList$. The notation $|G|$ stands for the number of members of the group G.

Algorithm 1 The GMGD algorithm

```
1:  Sort members of $G$ (using Quicksort algorithm)
2:  $for\ (L = 0; L < |G|; L++)\ \{$
3:    $H = violateFrom(L);$
4:    $iList.insertInDecreaseOrder(L, H - 1);$
5:  $\}$
6:  $while\ (!iList.isEmpty())\ \{$
7:    $(L, H) = iList.get(0);$
8:    $if(!overlap(L, H, fList))$
9:      $fList.add(L, H);$
10:   $iList.remove(0);$
11: $\}$
```

To investigate the optimality of GMGD, we compare its performance against an exhaustive algorithm that finds the minimum number of subgroups. The result in Fig. 4 is the average of 100 experiments where the geocast regions are randomly distributed over a square area whose width is 30 times the transmission range. The source is placed at the center of the area. The figure shows that GMGD on average is within 20% of the optimal solution.

Theorem 1: The worst case complexity of GMGD is $O(m^2)$, where m is the number of geocast regions that the considered node is responsible for.

Proof: The worst case complexity of the Quicksort algorithm is $O(m^2)$. The worst case complexity of the for loop would be $O(m^2)$ when there is no violation of the condition (8). In that case, line 3 would need $m - 1$ comparisons and line 4 would need 1 comparison for m iterations in total.

The complexity of the $while$ loop is dominated by line 8. In the worst case, the index list would consist of m disjoint pairs of indices of length |H-L|=1, thus over m iterations of the $while$ loop, line 8 would need $1 + 2 + .. + m = \frac{m^2+m}{2}$ comparisons to check if there is an overlap.

In total, the worst case number of comparisons is $\frac{5m^2+m}{2}$, which gives us an algorithm with $O(m^2)$ comparison steps. ■

We can see in the proof that the complexity of GMGD is dominated by the number of destinations (number of members of the geocast group). GMGD is therefore less complex than the merging algorithm in [4] which is $O(mk\min(m, k)^3)$ (k is the number of neighbours of the current node), especially when both the number of destinations and network density increase. Moreover, we perform GMGD only when the condition (8) is violated which further reduces the overhead of our algorithm compared to the one in [4] as the latter is run at every relay node. Given that the worst case is highly improbable, the average complexity of GMGD can be expected to be much smaller. This prediction is verified by simulation in the next section.

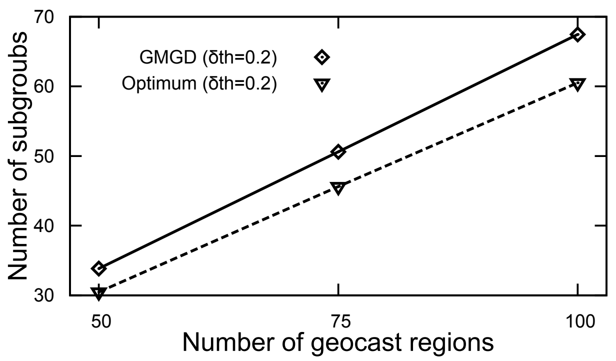

Fig. 4: GMGD: Optimality investigation

Algorithm 2 The RMG protocol

```
1:  if Node does NOT cover Division Point then
2:      Forward the packet towards division point.
3:  else
4:      if Condition (8) is violated then
5:          Perform $GMGD$ algorithm.
6:      end if
7:      repeat
8:          Pick a sub-group. Calculate $\varphi$ and $P$.
9:          Select next relay node $r$ for this sub-group.
10:     until All sub-groups has been visited.
11:     Add all $\varphi$, $P$, and $r$ to packet's header.
12:     Broadcast the packet to 1-hop neighbours
13: end if
```

C. The Recursive Multi-region Geocasting Protocol

Based on the discussion in previous subsections, we present the Recursive Multi-region Geocasting (RMG) protocol (see Algorithm 2). The protocol is completely localized and performed on a per-packet basis by individual nodes in the network. At the source of the packet or the node whose transmission range covers and is closest to the division point among its 1-hop neighbours, the group of geocast regions that the node is responsible for is divided into subgroups using GMGD if the condition (8) is violated. The forwarding line and division point of each subgroup are computed using equations (1) and (6). For each subgroup, the next relay of the packet towards the subgroup is selected using greedy geographic forwarding with recovery mode [9]. The packet is then broadcast to 1-hop neighbours.

On receiving the packet, a relay strips out from the packet's header all but the information about the subgroup that it is responsible for, and forwards the packet along the subgroup's forwarding line towards the division point. Note that, nodes in WSN do not necessarily fall on a forwarding line thus the forwarding nodes select those neighbours that are closest to the forwarding line as relays instead. Also, routing voids encountered during forwarding are circumvented by face routing [9]. This process repeats until all destinations are reached, i.e., a node that is inside a geocast regions is reached. Restricted flooding is then performed to disseminate the packet to all nodes inside a geocast region.

V. Evaluation

We study the performance of our RMG protocol, where the message is first routed to the center points of all destination regions as described before and constrained flooding is then used to disseminate the message within each region. We use simulation to compare RMG with two related protocols: GGP [14] and GMR [4]. GGP is selected because it also addresses the multi-region geocasting problem, and its approach to compute a branching point to fork the routing tree during forwarding is similar to our approach. Although GMR does not support multiple regions but only multiple destination nodes, it can be applied to multiple regions by first geocasting to the center points of all destination regions and then flooding the messages within each region similar to RMG.

We consider the following comparison metrics:

- *Relay load*: The ratio between the number of transmissions needed to successfully deliver the data packet, and the total number of network nodes. The lower this ratio, the lesser network resources (energy and bandwidth) are consumed for the delivery of the packet.
- *Average path length overhead*: Path length overhead is the ratio between the length of the shortest path (number of hops) from the source to the center point of a region, and the actual path length (number of hops) that the packet took to reach to that region. Suppose the packet has traversed k hops from the source to the center point of a geocast region M_j then the path length overhead for M_j is $\varepsilon_j = 1 - \frac{d_{shortest}(S,M_j)}{k}$. The average path length overhead is given by $\overline{\varepsilon} = \frac{1}{m}\sum_{j=1}^{m}\varepsilon_j$.
- *Computation time*: The total execution time it takes to successfully deliver a packet from the source to all nodes in all geocast regions. As a node receives a packet, the routing engine implemented in the node is invoked to compute the next relay(s) for the packet, which takes a certain period of time. In order to eliminate the impact of background activities, we run this routing engine for 1000 times and take the average of those time periods as the execution time of the routing engine. This approach for measuring execution time was suggested in [19]. For every node in the path that the packet has traversed from the source to all nodes in all geocast regions, we record the execution time and add them up to get the total execution time.

The simulation setup consists of sensor nodes randomly placed in a square deployment area. The data source is placed at the center of the area. To achieve varying geographical scale of the network, we vary the width of the deployment area in terms of the number of hops, i.e., $gscale = \frac{AreaWidth}{TransmissionRange}$. We vary the mean number of neighbours per node ($meanNB$) to achieve different network densities. Configuration detail will be presented in the subsequent sub-

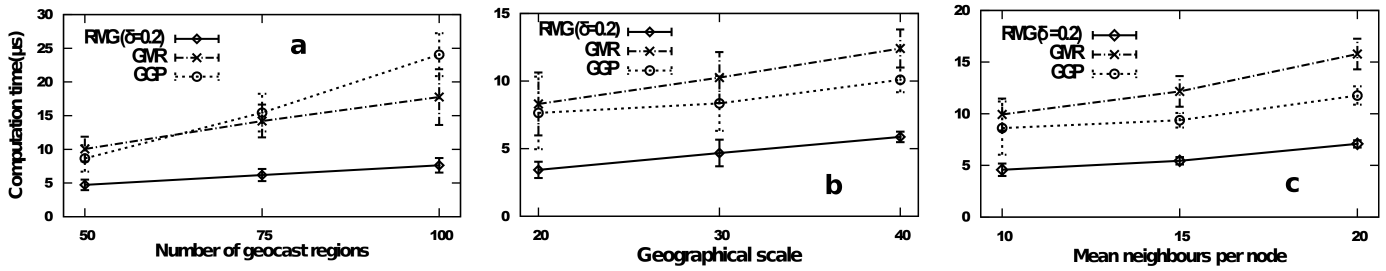

Fig. 5: Computation time evaluation

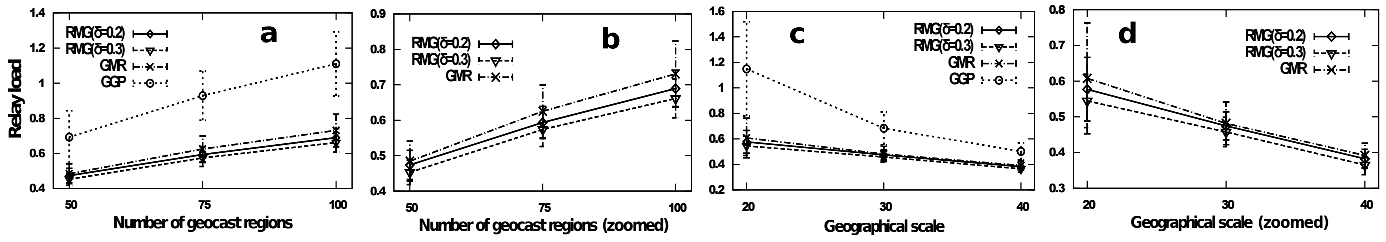

Fig. 6: Relay load evaluation (with circular transmission range)

sections. Our results for each configuration are the average over 500 simulation runs.

To summarize our results, our proposed RMG protocol achieves the best performance across all comparison metrics i.e., computation overhead, relay load, and path length overhead, compared to GMR and GGP. Moreover, RMG adapts well to various application requirements in terms of relay load and path length overhead by tuning the value of δ_{th} accordingly, which is not supported by the other protocols.

A. Choice For δ_{th}

Although δ_{th} is given by the application, we are interested in its best experimental value in an average case. We investigate relay load and detour rate for values of δ_{th} ranging from 0.1 to 0.9. The investigation is performed with a $gscale$ of 40, number of geocast regions ($nregion$) is 50, and a $meanNB$ of 20 neighbours per node. After 100 simulation runs for each value of δ_{th}, and taking averages, we found that the best experimental value is $\delta_{th} = 0.2$. We will use this value to run our RMG protocol in comparison with GMR and GGP.

B. Computation Time

In this subsection, we evaluate the computation time (in μs) of the protocols for three cases, namely varying $nregion$, varying $gscale$, and varying $meanNB$ (see Fig. 5 (a, b, c)). The general result shows that RMG requires the least computation time among all three protocols. The reason why the computation time of RMG is less than that of GGP in spite of the fact that an $O(m^2)$ algorithm (GMGD) is executed is two-fold: (i) First, GMGD benefits from the distribution of the geocast regions: In the best case where all regions reside no farther than δ_{th} from each other, GMGD does not even have to be performed. Moreover, GMGD is only run when a group of regions needs to be divided, which is expected to be rare in the average case; (ii) Second, the computation time of both RMG and GGP depends on the greedy neighbour selection algorithm, because the packet is greedily forwarded by RMG to a division point and by GGP to a Fermat point. Thus the packet's total travelling distance determines the computation time of the two protocols. With GGP, the packet has to go through all Fermat points which increases travelling distance for geocast regions at the beginning of the Fermat chain. With RMG, the packet always progresses directly towards a subgroup of geocast regions, which results in a much shorter total travelling distance when compared to GGP. The evaluation results in subsection V.D confirm this argument.

The computation time of GMR is expected to be high because it depends on GMR's exhaustive neighbour selection algorithm (an $O(mk\min(m,k)^3)$ algorithm) which is executed every time the data packet is forwarded. This expectation is confirmed in the two cases of varying $gscale$ (Fig. 5b) and varying $meanNB$ (Fig. 5c) as we can see the computation time of GMR grows faster than RMG's and GGP's. There is an exception in the case of varying $nregion$ where the computation time of GMR grows slower than GGP's (Fig. 5a). An explanation for this case is that as $nregion$ increases the total travelling distance of the packet with GGP increases to be much larger than with GMR, thus GGP's greedy neighbour selection algorithm is invoked many more times than GMR's exhaustive neighbour selection algorithm.

C. Relay Load

We compare the relay load that each protocol exerts on the network under varying $nregion$ and $gscale$. Specifically, $nregion$ is 50, 75, and 100 (see Fig. 6a) and $gscale$ is 20, 30, and 40 (see Fig. 6c). Comparison results are in line with our expectation that GGP incurs high relay load on the network because data packets have to go through all the Fermat points before they reach all geocast regions, which creates unnecessary detours for regions at the begin of the Fermat chain.

The performance of RMG is slightly better than that of GMR, which surprises us because we expect that the exhaustive neighbour selection algorithm of GMR that ensures that the next set of selected relays is minimal with regard to the cost-over-progress metric, would result in minimal relay load to be exerted on the network. An explanation for this is that our greedy group division algorithm (GMGD) tries to group as many as possible geocast regions into one transmission which reduces network relay load. A closer look at the performance comparison between RMG and GMR can be found in Fig. 6b and Fig. 6d.

D. Average Path Length Overhead

In this experiment we consider the same setup as in the above relay load experiment. The comparison results in Fig. 7a and 7c agree with our expectation that there will be a decent superiority of path length overhead of RMG over GMR's, and a huge jump from GGP's. This is because RMG restricts path length overhead according to δ_{th} while GMR's main goal is to maximize cost-over-progress ratio which does not necessarily decrease path length overhead. The GGP protocol only forwards data packets to Fermat points in spite of the actual distribution of geocast regions, thus creating extra path length overhead.

E. Flexibility

In this subsection we consider the effect of value of the parameter δ_{th} on the performance of RMG. If we take a closer look at the relay load and path length overhead evaluation results given in Fig. 6b and 6d as well as in Fig. 7b and 7d, we can see that increasing δ_{th} reduces relay load but at the same time increases path length overhead and vice versa. More specifically, with $\delta_{th} = 0.3$, RMG achieves further reduced relay load compared to that of GMR but at the same time still achieves lower path length overhead than that of GMR in case of varying $gscale$ (see Fig. 7d). This observation shows that RMG is flexible in tuning the value of δ_{th} to adapt to varying application need in terms of network relay load or path length overhead. For a particular application with specific relay load or path length overhead requirements, an appropriate value for δ_{th} that best fits the application requirements could be selected.

F. Non-Circular Transmission Range

We elaborate here our statement in subsection III-A that our proposed RMG protocol will still work with non-circular transmission range. The work in [20] proposes a realistic radio model called Radio Irregularity Model (RIM) that is based on empirical data from real sensor devices. The model introduces realistic Degree of Irregularity (DOI) [21] values for simulation purpose thus it provides a good approximation of radio irregularity for simulations. We use this model as a relaxation for our circular transmission range assumption. Each node determines its transmission range using the RIM model (with $DOI = 0.004$) at the beginning of a simulation, and uses this range throughout the simulation. This means each node has a completely different irregular geometric shape of the transmission range. An example of such a shape is given in Fig. 10.

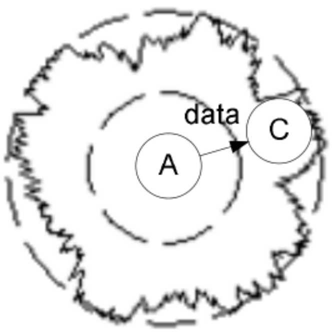

Fig. 10: Radio irregularity (from [20])

The evaluation results given in Fig. 8 and Fig. 9 are the average of 500 simulation runs. As we observe, RMG and GMR are not much affected by the RIM model as they both achieve a similar performance of relay load and path length overhead when compared to the UDG model (see previous section). However, a closer look at the relay load comparison between RMG and GMR is given in Fig. 8b and 8d showing that RMG still outperforms GMR under the RIM model.

In contrast, GGP exhibits a performance degradation in both relay load and path length overhead under the RIM model. We know that in GGP the data packet must traverse the main route connecting all Fermat points in order to reach all geocast regions. The irregularity of the transmission range of the nodes on that route makes the route become more zigzag and longer than under the UDG model, thus incurring extra relay load and path length overhead.

VI. Conclusion

In this paper we have presented a novel multi-region geocast routing protocol called Recursive Multi-region Geocasting (RMG), which addresses the problem of delivering data from a source to multiple remote geocast regions in large-scale wireless sensor networks. We compared the performance of RMG with two state-of-the-art protocols, namely GMR and GGP, using simulation. The comparison has shown that RMG outperforms the other protocols in all comparison metrics including computation time, network relay load, and transmission latency overhead. Furthermore, RMG is flexible with respect to application needs due to its capability of tuning the parameter δ_{th}. A performance evaluation of RMG under a more realistic wireless model [20] reveals that our proposed protocol also works well with irregular transmission regions.

VII. Acknowledgement

This work has been partially supported by the European Union under contract number ICT-2009-258885 (the SPITFIRE project).

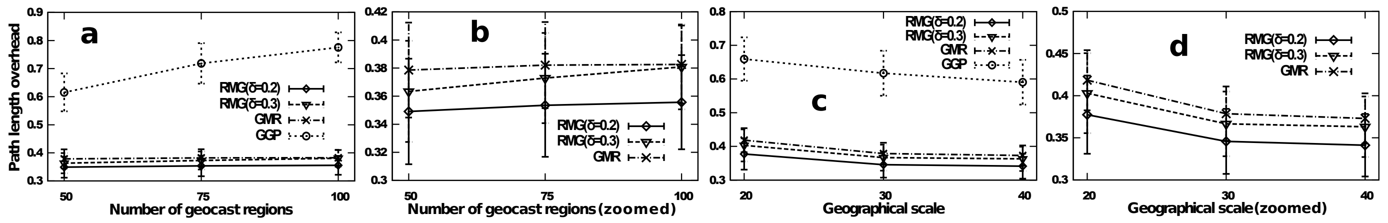

Fig. 7: Path length overhead evaluation (with circular transmission range)

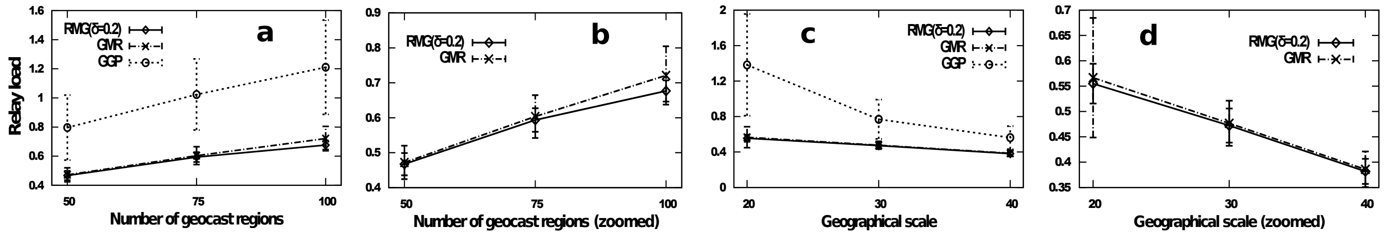

Fig. 8: Relay load evaluation (under RIM model)

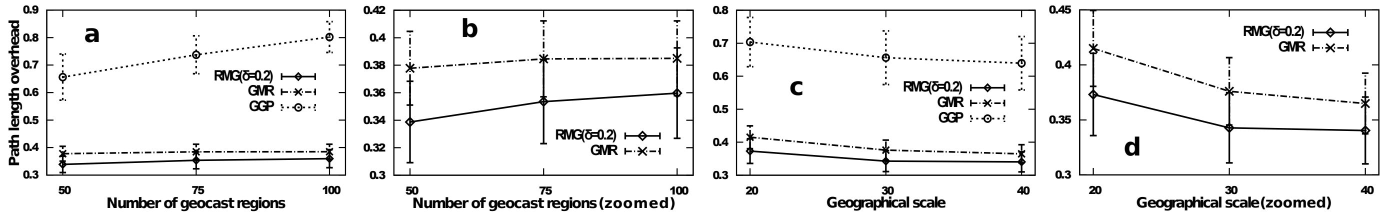

Fig. 9: Path length overhead evaluation (under RIM model)

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, 2002.

[2] D.-H. Shin, "Ubiquitous city: Urban technologies, urban infrastructure and urban informatics," *Journal of Information Science*, 2009.

[3] C. Frank, P. Bolliger, F. Mattern, and W. Kellerer, "The sensor internet at work: Locating everyday items using mobile phones," *Pervasive and Mobile Computing*, vol. 4, no. 3, pp. 421–447, 2008.

[4] J. Sanchez, P. Ruiz, X. Liu, and I. Stojmenovic, "Gmr: Geographic multicast routing for wireless sensor networks," *SECON*, 2006.

[5] J. A. Sanchez, P. M. Ruiz, and I. Stojmenovic, "Energy-efficient geographic multicast routing for sensor and actuator networks," *Computer Communication*, 2007.

[6] M. Transier, H.Fler, J. Widmer, M. Mauve, and W. Effelsberg, "Scalable position-based miulticast for mobile ad-hoc networks," *BroadWim '04*.

[7] S. Wu and K. S. Candan, "Gmp: Distributed geographic multicast routing in wireless sensor networks," *ICDCS*, 2006.

[8] Y.-B. Ko and N. H. Vaidya, "Geocasting in mobile ad hoc networks: Location-based multicast algorithms," *WMCSA '99*.

[9] P. Bose, P. Morin, and I. Stojmenovic, "Routing with guaranteed delivery in ad hoc wireless networks," *Wirel. Netw.*, 2001.

[10] Y. Yu, R. Govindan, and D. Estrin, "Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks," *UCLA Computer Science Department, Technical Report UCLA/CSD-TR-01-0023*, 2001.

[11] K. Seada and A. Helmy, "Efficient geocasting with perfect delivery in wireless networks," *WCNC*, 2004.

[12] J. Lian, K. Naik, Y. Liu, and L. Chen, "Virtual surrounding face geocasting with guaranteed message delivery for ad hoc and sensor networks," *IEEE/ACM Trans. Netw.*, 2009.

[13] I. Stojmenivic, "Geocasting with guaranteed delivery in sensor network," *IEEE Wireless Communications*, 2004.

[14] Y.-M. Song, S.-H. Lee, and Y.-B. Ko, "Ferma: An efficient geocasting protocol for wireless sensor networks with multiple target regions," *EUC Workshops*, 2005.

[15] N. Hadid and J. F. Myoupo, "Multi-geocast algorithms for wireless sparse or dense ad hoc sensor networks," *ICNS*, 2008.

[16] A. B. Bomgni and J. F. Myoupo, "An energy-efficient clique-based geocast algorithm for dense sensor networks," *Comm. and Netw. '10*.

[17] C.-Y. Chang, C.-T. Chang, and S.-C. Tu, "Obstacle-free geocasting protocols for single/multi-destination short message services in ad-hoc networks," *Wirel. Netw.*, 2003.

[18] S. Meguerdichian, S. Slijepcevic, V. Karayan, and M. Potkonjak, "Localized algorithms in wireless ad-hoc networks: Location discovery and sensor exposure," *MobiHoc '01*.

[19] B. Boyer, "Robust java benchmarking: Part 1 and part 2," *IBM's developerWorks, Technical Library*, 2008.

[20] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Impact of radio irregularity on wireless snesor networks," *MobiSys'04*.

[21] T. He, C. Huang, B. M. Blum, J. A. Atankovic, and T. F. Abdelzaher, "Range-free localization schemes in large scale sensor networks," *In Proc. MOBICOM*, 2003.