Bypassing Routing Holes in WSNs with a Predictive Geographic Greedy Forwarding

Minh Thiep Ha, Priyadharshini Sakthivel, and Hyunseung Choo

School of Information and Communication Engineering
Sungkyunkwan University, Korea
{thiepha,priya,choo}@skku.edu

Abstract. Applications in wireless sensor networks (WSNs) experience the routing hole problem. That is, the current node cannot forward to the destination, although it is the closest node, but not a neighbor of the destination. Accordingly, the packets from the source cannot be delivered to the destination. Jiang *et al.* recently proposed a SLGF approach to address this problem. However, SLGF still has long routing paths, since it uses the right-hand rule. In this paper, we describe Predictive Geographic greedy Forwarding, PGF. PGF uses information on the hole to build the virtual convex polygon, predict the routing path and choose the shorter path. PGF reduces the length and the number of hops of routing paths. Computer simulation shows our PGF scheme can reduce the average number of hops of routing paths by about 32% compared to Geographic greedy Forwarding, GF, and about 15% compared to SLGF.

Keywords: Routing hole problem; sensor networks.

1 Introduction

A typical wireless sensor network (WSN) is formed by a large number of distributed sensors together with an information collector [1]. These sensors are deployed to collect information and monitor situations in scenarios, such as forests, farmlands, harbors, and coal mines. They offer special benefits and versatility for wide applications in landslide prediction, object localization, and surveillance. However, due to the preservation of limited resources of sensor nodes, energy efficiency is a crucial objective of protocols designed in WSNs. Since routing is the most energy consumption operation of sensor nodes (hereinafter, referred to as nodes), it is very important to have a short routing path, *i.e.*, the number of nodes attending to forwarding operation is small. Greedy forwarding has been proven the most promising way for routing in WSNs. In greedy forwarding, each sensor forwards its packets to a 1-hop neighbor that is closer to the destination than it is. This method is repeated until the packet reaches the destination. However, geographic forwarding suffers from the so called *local minimum phenomenon*. In this, a node cannot forward its packet to its 1-hop neighbor, because all its 1-hop neighbors are further from the destination.

A BOUNDHOLE process [3] is used to construct the boundary of the hole, where the packets may be stuck, to mitigate the local minimum phenomenon.

C.S. Hong et al. (Eds.): APNOMS 2009, LNCS 5787, pp. 171–180, 2009.

When a packet is stuck at a node, the routing process will start a perimeter routing phase, where the packet is routed by the downstream node, until it reaches a node that is closer to the destination than that stuck node. Then, the routing returns to the greedy forwarding. Recently, Jiang *et al.* [5] proposed a new information model in which nodes are marked as either safe or unsafe nodes. This safety information is used to bypass the hole. However, these approaches may experience a long detour path in the perimeter routing. In this paper, we propose PGF, a Predictive Geographic greedy Forwarding approach to reduce the number of forwarding nodes. The information of the boundary hole is used to build a virtual convex polygon, form the virtual routing paths, select the shorter one, and create the real routing path that includes neighbor nodes in the network. Therefore, PGF can achieve a short routing path, *i.e.*, the number of forwarding nodes is small.

We implement our proposed PGF scheme using a simulator built in C++ to evaluate performance. We also implement and compare our proposed scheme to Geographic greedy Forwarding [3], GF, which uses the BOUNDHOLE algorithm, and Safety Information based Limited Geographic greedy Forwarding [5], SLGF. Since these schemes address the routing hole problem using different approaches, we think they are good benchmarks for our proposal. Simulation shows PGF can reduce the average number of hops of routing paths about 32% compared to GF, and about 15% compared to SLGF. The remainder of the paper is organized as follows. In section 2, we briefly describe related work. Our proposed scheme PGF is presented in section 3. Section 4 provides the performance evaluation with simulation results. Finally, we conclude our work in the last section.

2 Related Work

Greedy-Face-Greedy (GFG) [2], Greedy Perimeter Stateless Routing (GPSR) [6], and Greedy-Other-Adaptive-Face Routing (GOAFR) [7] are currently the most popular methods to mitigate the local minimum issue. When the routing process becomes stuck at an intermediate node, it will start a perimeter routing phase. In this, the packet is routed by the "right-hand rule" counter-clockwise along a face of the planar graph that represents the same connectivity as the original network, until it reaches a node that is closer to the destination than the stuck node. Then, the routing returns to the greedy forwarding phase. In recent work [4], such a routing scheme is proved to guarantee delivery in any arbitrary planar graph. However, without sufficient shape information about the holes, such a routing may use a long detour path in the perimeter routing, compared to the shortest path to the destination. The work in [3] focused on the use of the hole area that contains the stuck nodes. The hole is detected at a stuck node, where the packet can get the local minimum in greedy forwarding routing [2,6]. A process called BOUNDHOLE is initiated to form a closed circle (also called the boundary of the hole or the boundary hole). The region enclosed by the boundary will be identified as the hole area (Fig. 1). For each node along the boundary, its successor node along the boundary in clockwise order (or counter

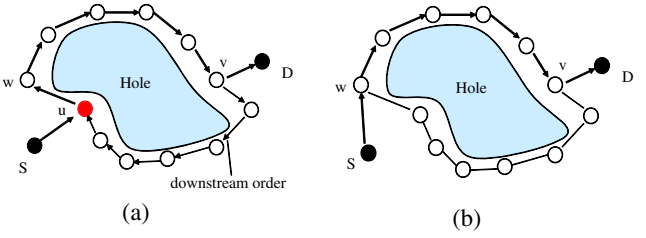

Fig. 1. (a) GF Routing. (b) SLGF Routing

clockwise order) is marked as the downstream node, *e.g.*, node w is downstream of node u, (or upstream node). Geographic greedy Forwarding, GF, is used to forward packets to the destination. In GF, when a package is stuck at a node in the boundary, *e.g.*, node u in Fig. 1a, it is forwarded to the downstream node of the current node, *e.g.*, node w in Fig. 1a, until it reaches a node that is closer to the destination than the stuck node, *e.g.*, node v in Fig. 1a. Then, the routing returns to the greedy forwarding phase.

Jiang *et al.* [5] recently proposed the Safety Information based Limited Geographic greedy Forwarding, SLGF. SLGF uses a new information model, in which nodes are labeled as safe or unsafe nodes in four quadrants. A node is labeled as an unsafe node in a quadrant if there is no safe neighbor in that quadrant. The unsafe information is spread until no new unsafe node is formed. SLGF tries to avoid forwarding packets to unsafe nodes, since using unsafe nodes will cause the local minimum. When a node has a packet, it tries to find a safe neighbor in the requesting zone, which is rectangle formed by its coordinate, and the coordinate of the destination to forward. If it cannot find a safe node in that area, it finds the first safe node in a counter clockwise direction. In the case where the destination is an unsafe node and is in the request zone of the current node, unsafe nodes are used to forward packets to the destination. However, both GF and SLGF approaches may experience a long detour path in the perimeter routing, since they find the forwarder in one direction only.

3 Predictive Geographic Greedy Forwarding Approach

We assume that each node knows its position, position of 1-hop neighbors, and position of the destination using either GPS or other location services. In this section, we describe the Predictive Geographic greedy Forwarding approach. It uses information about the hole to predict the short path to route between the source and destination nodes.

3.1 Overview

After checking if nodes are stuck nodes, the BOUNDHOLE [3] is run to build the boundary of the hole. The information of the boundary, *i.e.*, the position and

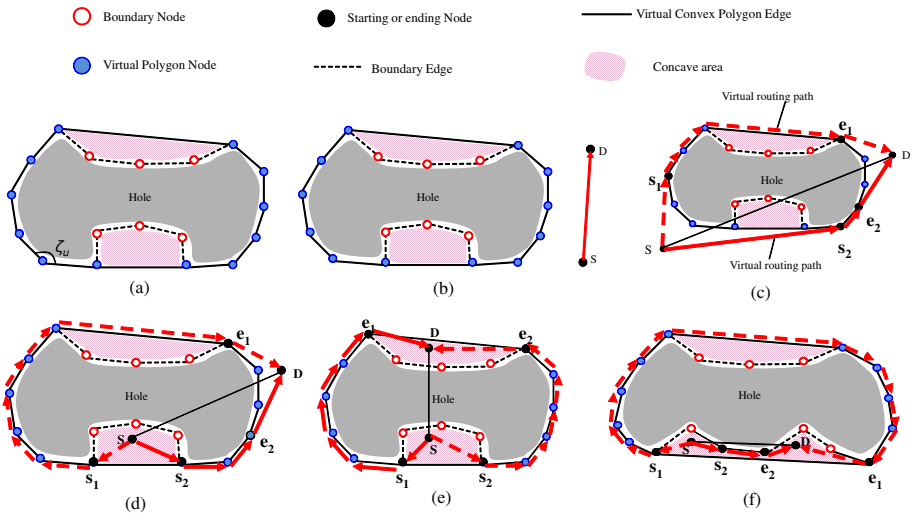

Fig. 2. (a) Boundary hole and virtual convex polygon; (b) Case 1; (c) Case 2; (d) Case 3; (e) Case 4; (f) Case 5

order of nodes in the boundary, is transferred to the source. When a source wants to transfer a packet to a destination, it first builds a virtual convex polygon of the hole. The source node determines its position and position of the destination compared to the virtual convex polygon to form two virtual routing paths in which adjacent nodes may be not forward to each other. By calculating the distance of two virtual paths, the source node chooses the shorter one to form the real routing path. The real routing path is formed using greedy forwarding between adjacent nodes that are not neighbors.

3.2 Forming the Virtual Convex Polygon

As mentioned in section II, after checking if nodes are stuck nodes, a BOUNDHOLE is run to build the boundary of the hole. When this process finishes, nodes know if they belong to a boundary. The node that gets the information of the hole, *i.e.*, the position of nodes in the boundary in the downstream order, will transfer this information to nodes in the boundary. In our proposal, the information about the hole is transferred to the source by flooding. The source node keeps this information to predict the path to the destination. Each time a source node wants to transfer to a destination, the source node builds a virtual convex polygon and determines its position, the position of the destination compared to the virtual convex polygon. The source node finds the nodes belonging to the virtual vertex polygon to build the virtual convex polygon. Node u calculates its *rotation angle* to check if it belongs to the virtual vertex polygon:

Rotation angle: Rotation angle of a boundary node u, ζ_u, is the angle spanned by a pair of adjacent neighbors of node u and is in the direction of the hole.

We can see that the rotation angle is formed by rotating the edge between u and its previous node in the boundary to the edge between u and its following node in the boundary counter clockwise. Node u belongs to the virtual vertex polygon if its rotation angle is less than 180 degrees, *i.e.*, $\zeta_u < 180^0$. This process is run until all nodes in the virtual convex polygon have a rotation angle less than 180 degrees. The virtual convex polygon is formed by connecting nodes satisfying the above condition. Nodes that are neighbors in the virtual convex polygon may not be neighbors in the hole boundary. Fig. 2a shows an example of the virtual convex polygon of a hole boundary. We also define the concave area. A concave area is the area that lies between the hole and the virtual convex polygon (Fig. 2). If the hole and the virtual convex polygon are the same, there is no concave area.

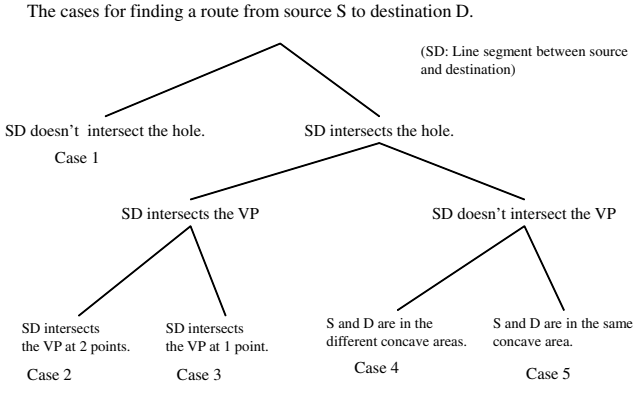

Fig. 3. Five cases for finding a route to the destination in PGF

3.3 Finding the Path to the Destination

After determining the shape of the virtual convex polygon, the source node finds the path to the destination based on its relative position and the position of the destination compared to the position of nodes in the virtual convex polygon. With different relative positions of the source S and destination D, we have a different way to find the path. The five cases are shown in Fig. 3. Each case is differentiated by the number of intersection points of the line segment SD with the hole boundary and the virtual convex polygon. These five cases are complementary. A different way to find the routing path is used for each different case. We will detail each case in the following section. The first case is the simplest case. In this case, the line segment SD does not intersect with the virtual convex polygon (Fig. 2b), we use greedy forwarding to transfer packets to the destination, because the hole does not affect the route between the source and destination. For the remaining cases, we can always find two paths to pass over the hole, *i.e.*, the path in the left side and the path in the right side of SD (Figs. 2c, 2d, 2e, or 2f), because the line segment SD intersects with the hole.

The routing path from the source to the destination will be the shortest path, if two paths are the shortest paths among paths in two sides and the chosen path is the shorter one between these two paths. We find the start and end nodes defined below to find these two paths. We have the routing path between the source and destination over the hole $S \Rightarrow u_1 \Rightarrow u_2 \Rightarrow ... \Rightarrow u_i \Rightarrow D$; u_1, u_2, ..., u_i are nodes in the boundary, then we name u_1 the start node and u_i the end node of this routing path. Each path has one start node and one end node. Therefore, we have two start nodes and two end nodes. Now, we will discuss how to find these nodes for each case. In the second case, the line segment *SD* intersects with the hole. *SD* also intersects with the virtual convex polygon at two points. That is, both S and D are outside the virtual convex polygon and the hole (Fig. 2c). The line segment *SD* divides the virtual convex polygon into two sides, *i.e.*, the left side and the right side. We only find the start and end nodes among nodes belonging to the boundary of the hole. A node u is a start node if it satisfies the following conditions:

1. Condition 1: u is a node of the virtual convex polygon of the hole.
2. Condition 2: The ray Su intersects with the virtual convex polygon at only u.

Since u is a node of the virtual convex polygon, *i.e.*, node in the boundary of the hole, and the ray Su intersects with the virtual convex polygon at only u, packets transferred from S to D through u will not be stuck at u. These conditions also make the routing paths more straightforward. The end nodes are found using the same conditions with the only change being that the ray Du intersects with the virtual convex polygon at only u. Nodes s_1, s_2 and e_1, e_2 are start and end nodes, respectively. After that, two virtual routing paths are formed as: $S \Rightarrow$ start node $\Rightarrow$ adjacent nodes in the virtual convex polygon $\Rightarrow$ end node $\Rightarrow D$.

From S, we transfer packets to the start node. Next, the adjacent node, *i.e.*, the neighbor node, of the start node is chosen as forwarder. If the start node is to the left side of the line segment *SD*, the chosen neighbor of the start node is the neighbor in the downstream direction. Conversely, if the start node is in the right side of the line segment *SD*, the chosen neighbor of the start node is the neighbor in the upstream direction. This selection is repeated until it meets the end node. Finally, we transfer from the end node to the destination (Fig. 2c). We calculate the total distance between nodes from S to D in two paths to choose a shorter path. The path that has the lesser distance will be selected as the routing path to transfer packets to the destination, *e.g.*, the solid red line in Fig. 2c. Since the distance between nodes in the above routing path, *e.g.*, from the source node to the start node s_2 in Fig. 2c, may be longer than the node radio range, we use greedy forwarding transfer to packets.

For the third case, the line segment *SD* intersects with the hole. *SD* also intersects with the virtual convex polygon, but at only one point. Since *SD* intersects with the virtual convex polygon at only one point, S or D must be inside the virtual convex polygon, *i.e.*, inside the concave area; the other is outside the virtual convex polygon, *e.g.*, node S is inside and node D is outside the virtual convex

polygon in Fig. 2c. We also use the line segment *SD* to determine two sides and two routing paths. However, instead of dividing the virtual convex polygon, SD divides the boundary hole into two sides and categorizes nodes in the boundary hole into two sets, *i.e.*, the set in the left side and the set in the right side. The way to find the start nodes (or end nodes) for a node that is outside the virtual convex polygon is the same as for case 2, *e.g.*, node D in Fig. 2c finds two end nodes e_1 and e_2. For that node inside the concave area, we select from two sets of nodes, *i.e.*, the set of nodes in the left side and the set of nodes in the right side, two nodes that satisfy condition 2 as end nodes (or start nodes), *e.g.*, node S in Fig. 2c finds two start nodes s_1 and s_2. Next, two virtual routing paths are formed as: $S \Rightarrow$ start node $\Rightarrow$ adjacent nodes in the boundary hole $\Rightarrow$ adjacent nodes in the virtual convex polygon $\Rightarrow$ end node $\Rightarrow D$.

After getting to the start node from S, the adjacent node in the boundary hole, *i.e.*, the neighbor in the boundary hole, of the start node is chosen as the next forwarder. If the start node is in the left side (or right side) of the *SD*, the downstream neighbor (or the upstream neighbor) will be chosen as the next forwarder. This selection is repeated until finding a node in the virtual convex polygon, also in the boundary hole. From this point, only neighbors in the virtual convex polygon can be the next forwarder, *e.g.*, two red paths in Fig. 2c. After reaching the destination, we calculate the total distance of two paths and choose the shorter one for transferring packets, *e.g.*, the solid red line in Fig. 2c. As before, greedy forwarding is used to transfer packets between nodes that are not neighbors in the virtual routing path.

For the remaining cases, we use the same method as for case 3 to find start nodes, end node, s to form two virtual routing paths, and to select the shorter one for transferring packets.

3.4 Multiple Holes

For cases in which there are multiple holes between the source and the destination, the source node finds the routing path and transfers packets to the end node using the method as in the previous section. When the end node receives packets, it applies the same method as the source node to transfer packets to the destination. In Fig. 4, the packet from node S bypasses the first hole at node

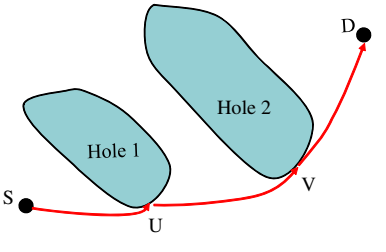

Fig. 4. Multiple holes

U, the end node in the routing path from S to D. U applies the same method as S and bypasses the second hole at node V. Finally, the packet arrives at the destination.

4 Performance Evaluation

In this section, we study the average-case performance of the proposed information model and routing algorithms, using a simulator built in C++. The performance metrics used in the evaluation are the number of hops and the length of the routing path.

In the simulations, nodes with a transmission radius of 20 meters are deployed to cover an area of $200m \times 200m$, under different deployment models. We evaluate approaches in two models. First, the nodes will be deployed uniformly. This is the ideal model (denoted by *IA*), in which the hole is only caused by sparse deployment. Usually, the size of a hole is very small. Second, we randomly set some forbidden areas inside the network area, where no nodes can be deployed. The forbidden areas, which may be irregular, are constructed to study the impact of larger holes on the proposed algorithms. Such a model is denoted by *FA*. We assume that the destination and the source are randomly selected, including both safe sources and unsafe sources. Before we test the routing performance for routing time, boundary information [3] is constructed for GF routing; safety information [5] is constructed for SLGF routing; and the virtual convex polygon is constructed for our PGF routing. Then, we test the networks when the number of nodes in the area is varied from 400 to 800 in increments of 50. For each case, 100 networks are randomly generated, and the average routing performance over all of these randomly sampled networks is reported. Fig. 5a shows the upper bound of the number of hops of the routing path. Fig. 5b shows the average number of hops of the routing path. As mentioned in section II, PGF can predict the routing path by forming the virtual routing paths, so the number of forwardings in PGF is small also.

Section II, when holes exist in the networks, GF routing may experience a long detour, because the packets are transferred to a node in the boundary hole and detour to downstream nodes before arriving at the destination. In the case of SLGF, because it uses the right-hand rule to select the next forwarder, although the right-side has the shorter path, the left-side path is still chosen. This causes a long detour. For PGF, the source node predicts the stuck nodes and chooses the shorter of the path on the left side and the path on the right side. As a result, PGF has a shorter path than GF and SLGF. As shown in Fig. 5a, PGF routing reduces the number of routing hops by about 15 percent compared to SLGF, and about 32 percent compared to GF. The above property still holds when more large holes occur under the FA model. In WSNs, the packet is forwarded hop-by-hop along the path. Reducing the number of hops can reduce end-to-end delay and furthermore support quick response to routing requests. Fig. 5c shows the corresponding average length of the entire routing path. Since PGF chooses the shorter path by comparing the total distance (length) of two paths, the length of the routing path in PGF is smaller than in GF and SLGF.

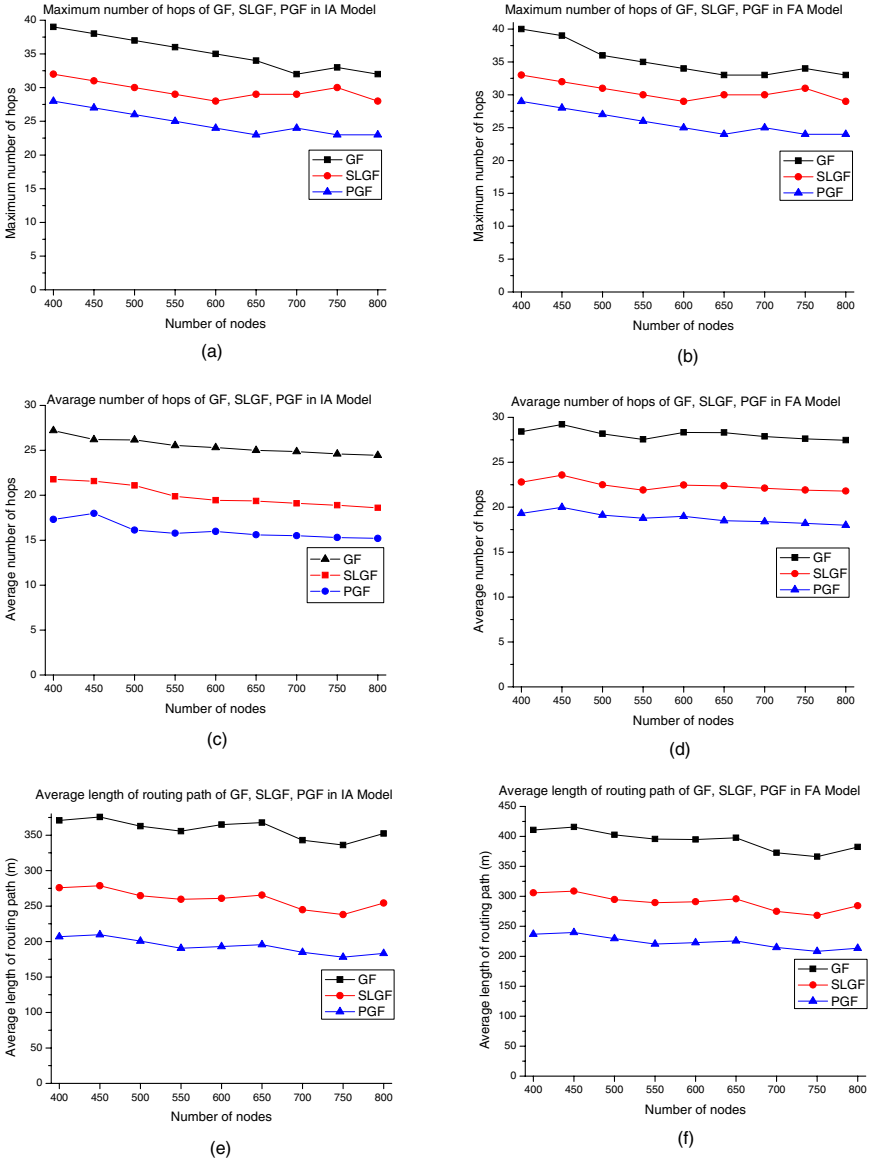

Fig. 5. (a) Maximum number of hops in IA model; (b) Maximum number of hops in FA model; (c) Average number of hops in IA model; (d) Average number of hops in FA model; (e) Average length of paths in IA model; (f) Average length of paths in FA model

5 Conclusion

In this paper, we have shown the way to bypass the routing hole using Predictive Geographic greedy Forwarding, PGF. A virtual convex polygon is built based on the information of the boundary hole to predict the routing path. Furthermore, simulations show that PGF efficiently reduces the number of hops and the length of routing paths. In future work, we will extend our work to increase the adaptability of the scheme so that it can be more efficient in networks where topology changes frequently.

Acknowledgments. This research was supported by MKE, Korea under ITRC IITA-2009-(C1090-0902-0046). Dr. Choo is the corresponding author.

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine 40, 102–114 (2002)
2. Bose, P., Morin, P., Stojmenovic, I.: Routing with guaranteed delivery in ad hoc wireless networks. In: Proc. The 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, pp. 48–55 (1999)
3. Fang, Q., Gao, J., Guibas, L.: Locating and bypassing routing holes in sensor networks. In: Proc. The 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2004), pp. 2458–2468 (2004)
4. Frey, H., Stojmenovic, I.: On delivery guarantees of face and comined greedy-face routing in ad hoc and sensor networks. In: Proc. The 12th Annual ACM/IEEE International Conference on Mobile Computing and Networking (ACM/IEEE MOBICOM 2006), pp. 390–401 (2006)
5. Jiang, Z., Ma, J., Lou, W., Wu, J.: An Information Model for Geographic Greedy Forwarding in Wireless Ad-Hoc Sensor Networks. In: Proc. The 27th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2008), pp. 825–833 (2008)
6. Karp, B., Kung, H.: GPSR: Greedy perimeter stateless routing for wireless sensor networks. In: Proc. The 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (ACM/IEEE MOBICOM 2000), pp. 243–254 (2000)
7. Kuhn, F., Wattenhofer, R., Zollinger, A.: Worst-case optimal and average-case efficient geometric ad-hoc routing. In: Proc. The 4th ACM International Sysmposium on Mobile Ad Hoc Networking and Computing (ACM MobiHoc 2003) (2003)