Geographic Routing with Early Obstacles Detection and Avoidance in Dense Wireless Sensor Networks

Moraru, Luminita; Leone, Pierre; Nikoletseas, Sotiris; Rolim, Jose

How to cite

MORARU, Luminita et al. Geographic Routing with Early Obstacles Detection and Avoidance in Dense Wireless Sensor Networks. In: Proceedings of the 7th International Conference on Ad-hoc, Mobile and Wireless Networks, ADHOC-NOW 2008. Sophia-Antipolis, France. [s.l.] : Springer, 2008. p. 148–161. (Lecture Notes in Computer Science) doi: 10.1007/978-3-540-85209-4_12

This publication URL: https://archive-ouverte.unige.ch//unige:34067
Publication DOI: 10.1007/978-3-540-85209-4_12

Geographic Routing with Early Obstacles Detection and Avoidance in Dense Wireless Sensor Networks

Luminita Moraru[1] *, Pierre Leone[1], Sotiris Nikoletseas[2], and Jose Rolim[1]

[1] Computer Science Department
University of Geneva
1211 Geneva 4,Switzerland
[2] University of Patras and CTI
26500 Patras, Greece

Abstract. Existing geographic routing algorithms for sensor networks are mainly concerned with finding a path toward a destination, without explicitly addressing the impact of obstacles on the routing performance. When the size of the communication voids is increased, they might not scale well with respect to the quality of paths, measured in terms of hop count and path length.
This paper introduces a routing algorithm with early obstacle detection and avoidance. The routing decisions are based on path optimality evaluation, made at the node level, gradually over time. We implement our algorithm and evaluate different aspects: message delivery performance, topology control overhead and algorithm convergence time. The simulation findings demonstrate that our algorithm manages to improve significantly and quite fast the path quality while keeping the computational complexity and message overhead low. The algorithm is fully distributed, and uses only limited local network knowledge.

1 Introduction

Geographic routing algorithms represent one of the most suitable solution for routing within sensor networks, mainly due to their stateless nature. The path is built only with information about the one hop neighbors and of the destination, thus they require negligible memory at sensor nodes - a direct consequence is network scalability - no additional topology control traffic is needed when the network changes.

The simplest *geographic routing* strategy, *greedy*, chooses for forwarding the neighbor closest to the destination [16], [12], [3]. But it has a main drawback, called the local maximum phenomenon: when the current node has no neighbor closer to the destination then itself, the delivery of the message fails. This is often the case if there is an obstacle or a void in the network, or in low density network areas.

* Research funded by FP6-015964 AEOLUS

The solution to this problem is a recovery mode, an alternative routing method with guaranteed delivery, used when greedy fails. Several classes of algorithms have been proposed for this purpose. Further we will discuss the class of memoryless recovery mechanisms, *perimeter* routing, based on planar graph traversal techniques. The algorithms in this class work only on planar graphs, thus before entering this mode, a planar subgraph of the initial graph must be available. The basic idea behind this algorithms is as follows: a message is forwarded clockwise along a face of a planar graph. When it reaches a link that intersects the line between the source and the destination, it switches to the adjoining face. A message will leave the *perimeter* mode when it will find a node closer to the destination than the perimeter entry point.

Geographic routing algorithms scale well with respect to effectiveness of the path when the size of the communication voids is varied. But these paths are not optimal in terms of length, and in fact they might be quite long, thus inefficient. This is due mainly to the nature of the protocol used during the rescue mode: *perimeter* routing. It will choose sometimes relays that are further away from the destination than the current node. Additionally, it requires graph planarity, and the planarization process preserves the shortest links, thus increasing the hop count.

The complexity of obstacle avoidance problem is influenced as well by the shape of the obstacles. Difficulties appear mainly in avoiding concave obstacles (see Fig 1(a)). Even if we consider only the case of convex obstacles (see Fig 1(b)), an important constraint remains: nodes should exploit only local information.

In this paper we consider the behavior of *geographic routing* algorithms within network configurations with obstacles and local irregularities. Our contribution is to identify the presence of the object early on the routing path and redirect the messages on a shorter path as soon as possible. The strategy we are proposing is as follows: during message forwarding, each node evaluates the optimality of the paths that go through it. The node tags itself based on the outcome of the node optimality evaluation method - the evaluation is positive if a node has at least one neighbor tagged as optimal closer to the destination then itself. If a node is non-optimal, than we consider that any path toward the destination using it will be as well non-optimal.

Subsequent message forwarding decisions will analyse first the suitability of optimal nodes when choosing the relays. If no optimal node is suitable (e.g. no neighbor closer to the destination than the current node is optimal), then a non optimal node is used.

When obstacles are present, the consequences of our method are the tagging of the nodes in the vicinity of the object as non optimal, and the early redirection of the message toward the edge of the object, resulting in a significant decrease of the path length. The cost is a small overhead, depending on obstacle size and shape, (independent of the network size) and paid only once.

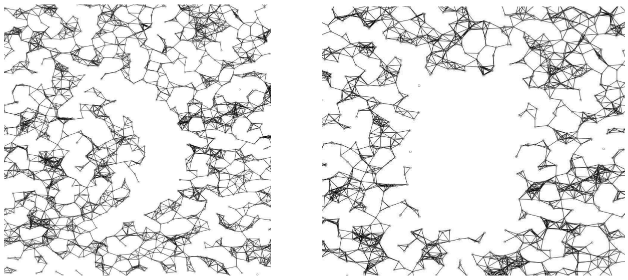


Fig. 1. Communication Voids

2 State of the Art and Comparison

We address the problem of early detection and avoidance of obstacles in geographic routing algorithms. Although several geographic routing with obstacles avoidance techniques were proposed so far, most of them are concerned mainly in guaranteeing the delivery: finding some path when greedy forwarding is not possible. Moreover, there are situations where the constraints like the stateless nature (i.e. the low memory needed) of geographic routing, are in contrast to the quantity of data they need to make a decision. Further we will introduce the techniques with guaranteed data delivery, outlining their characteristics and drawbacks. The solutions are divided in the following categories, as described in [4]: *planar graph based, geometric obstacle detection, cost based, flood based, and hybrid.*

Planar graph based obstacle avoidance techniques, [1], [10], [8],[5], are used since they were proved to guarantee delivery if a path exists. In the initial stage, these strategies use greedy. When a node has no neighbor closer to the destination, greedy is replaced by one of existing planar graph traversal algorithms [12], [22], [14], [13], [18]. Since the representation of the network is not always a planar graph, this class of strategies uses a distributed planarization algorithm, like those proposed in [21],[7],[11]. The performances of these strategies depend on two factors: the graph traversal and the distributed planarization algorithms. Nevertheless, most of the algorithms are concerned with improving the planar graph traversal algorithms while ignoring the optimality of the path. Still, the gain in path length (compared with the optimal path) becomes significant when obstacles are present and it is proportional with their size.

An optimality evaluation method is described in [17]. It can be built on top of any method based on planar graph traversal. Each node keeps track of the ratio between greedy decisions and the total number of routing decisions. If the ratio is higher than a specific threshold, then the node is considered as being optimal. The main drawback of this method, is that the optimality of the path does not depend on the network topology only, this way failing to correctly evaluate some of the nodes.

Geometric obstacle detection is proposed in [6]. It uses the geometric properties of a node to determine if a message can be stuck at that node. An algorithm is developed to find holes in the network, defined as areas of the network bounded by the stuck nodes. The disadvantage of this technique is the high complexity of the detection of the holes. Additionally, it does not guarantee delivery when the destination is inside the hole.

Cost based approach [19] consists in assigning a cost to each node, proportional to the distance to the destination. When greedy forwarding fails, a node will forward a packet to a neighbor with a lower cost than itself. Although the complexity and the overhead of the algorithm is rather medium, it does not choose optimal paths. *Flooding based* techniques [20],[9] are using broadcast to forward the message, once a packet is stuck. Although the complexity is low, the overhead is high. They guarantee delivery, but path optimality is not a concern. Multipath techniques, like [15], [2], explore several paths toward the destination, to trade-off efficiency with fault tolerance. Similar with the case of flooding techniques, the overhead may be high. *Hybrid* techniques use at least a combination of two obstacle avoidance methods. The motivation is the improved efficiency of the path and the guaranteed delivery of the message. They are used when only one of the two techniques is not enough to achieve these requirements. The disadvantage is the increased overall complexity.

The methods described above are mainly concerned with guaranteeing delivery. In contrast, we aim at providing high quality paths, by keeping track of previous evaluations in a distributed manner. Additionally, our technique preserves the properties of the network, like scalability and low complexity since it works only with local information about the direct neighbors of the node currently propagating data.

3 Non optimality evaluation methods

The algorithm presented in this paper is part of a class of algorithms based on non optimal nodes detection. It will be presented in parallel with the previous work in the same area. In each case we propose a different method for the detection of non-optimal nodes. We define a node as *non optimal* if any message using the node as a relay will eventually use rescue mode to reach the destination. A *non optimal* path between a source and the destination is a path containing at least one non optimal node.

3.1 Behavior based tagging (BBT)

In [17], the optimality of a node is evaluated as follows: if a node uses greedy forwarding, then a positive counter is incremented, if perimeter mode is used, then a negative counter is incremented. A node is considered on an optimal path if the ratio between the greedy decisions and the total number of decisions is higher than a specified threshold.

The routing algorithm will consider the result of the evaluation of the nodes while selecting the relays for a message. When a message is routed in the greedy mode, the node will first search for neighbors closer to the destination and marked as optimal. If no neighbor is found, it will switch to perimeter. When a message is routed in the perimeter mode, the current relay will switch back to greedy if it finds a node closer to the destination than the perimeter entry point, otherwise it will continue in the perimeter mode.

The behavior of this method is shown in the examples in Fig 2(a), 3(a), 4(a), and it will be discussed in the next subsection. The drawback of this approach is the wrong evaluation of some nodes as non optimal, due to the influence of the position of the perimeter entry point on the routing mode used at each node (this behaviour will be explained in more details in the next subsection). Therefore, a more precise evaluation method is needed.

3.2 Neighborhood based tagging (NBT)

The evaluation method is as follows: a node will mark itself as non-optimal toward a certain direction if it does not have optimal neighbors (or does not have neighbors at all) toward that direction. The impact of this method on the network is the apparition of a marked convex region along some of the faces of the object. Further, we will give a formal definition of non optimal nodes.

Let $G = (N, E)$ be a graph representation of the network, where N represents the set of nodes and E the set of links. We select $n_k \in N$ a random node in the network and d the sink receiving all the messages. Let $S_k = \{n_i | (n_k, n_i) \in E\}$ be the set of one hop neighbors and $S'_k = \{n_i | n_i \in S_k \wedge dist(n_i, d) < dist(n_k, d)\}$. If $M \subset N$ is the set of non optimal nodes in the network, then $n_k \in M$ if $S'_k \cap M = S'_k$.

Algorithm 1 Optimality Evaluation Method

```
this.setProperty(optimality,'NON-OPTIMAL')
for all n_i in S do
   if this.closer(D, n_i) and n_i.getProperty(optimality) ==' OPTIMAL' then
      this.setProperty(optimality,'OPTIMAL')
      break
   end if
end for
```

The pseudocode of the algorithm is presented herein. Algorithm 1 describes the optimality evaluation method. *this* refers to the node making the evaluation. Algorithm 2 describes the routing strategy that includes non optimality of the nodes for path evaluation.

We define the marked area as the area in the vicinity of the object containing nodes tagged as non optimal. The unmarked area is represented by the rest of the network. The influence of optimality tag on routing decisions is as follows:

Algorithm 2 Optimality based Routing Strategy

```
if routing_mode is "perimeter" then
  next ← get_next_hop("perimeter", neighbors)
else
  selected_neighs ← filter_by_property(neighbors, optimality, 'OPTIMAL')
  next ← get_next_hop("greedy", selected_neighs)
  if ! ∃ next then
    next ← get_next_hop("greedy", neighbors \ selected_neighs)
    if ! ∃ next then
      next ← get_next_hop("perimeter", neighbors)
    end if
  end if
end if
evaluate_optimality
```

- Unmarked area: the behaviour of the routing protocol remains unchanged. Once a node in the marked area, it will use greedy to get to the destination. Once there are no closer neighbors, the node uses perimeter.
- Border: The routing protocol tries to avoid the entry into the marked area. Therefore, for a message in the greedy mode, a node will search first a neighbor, closer to the destination than itself, between the optimal nodes. If it fails, it will start a new search considering the set of non optimal nodes, closer to the destination than itself.
- Marked area: similar with the unmarked area.

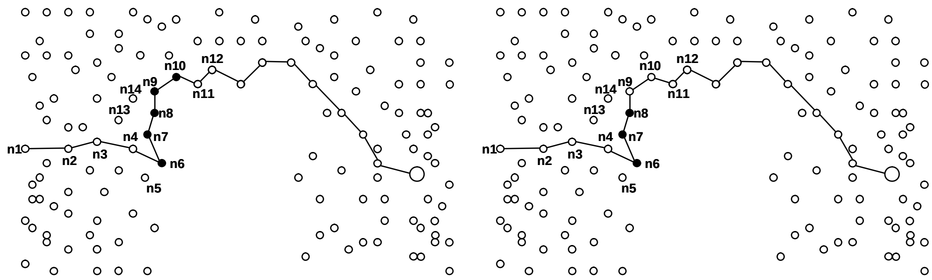
Our algorithmic design is aiming at the following improvements:

- *Smaller marked area* - there are nodes which have greedy neighbors toward the destination, but they are using perimeter routing since they are not closer to the destination than the perimeter entry point. The tagging method based on neighborhood will mark them as optimal, while the method based on behavior would have marked them as non optimal.
- *Shorter paths* - since greedy tries to route around the marked area, reducing this area will result in reducing the length of the path.
- *More accurate evaluation* of the optimality, since the dependence of the perimeter entry point and the position of the source is eliminated.

3.3 Example

An example of the behavior of the algorithm is presented in Fig 2, 3, 4. They show both the evaluation (tagging) and routing path chosen by the network during three transmitted messages. The evaluation is made progressively, during the routing tasks: each time a node has to make a routing decision, it checks the status of its neighbors.

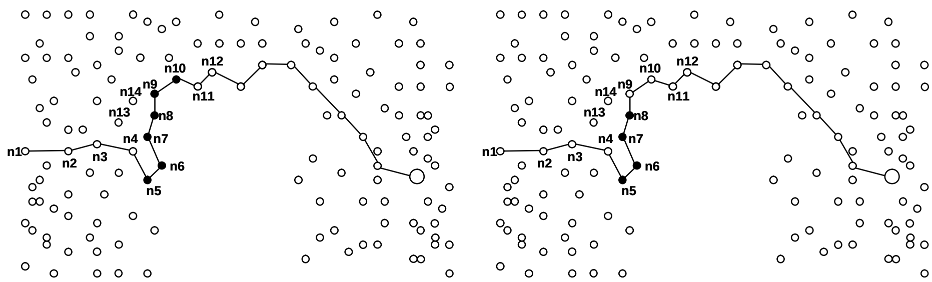
Figure 2(a) shows the transmission of the first message. The message is originated at node $n1$. Each node from $n1$ to $n4$ has a greedy neighbour toward the



(a) Behavior based tagging (b) Neighborhood based tagging

Fig. 2. The path of the first message

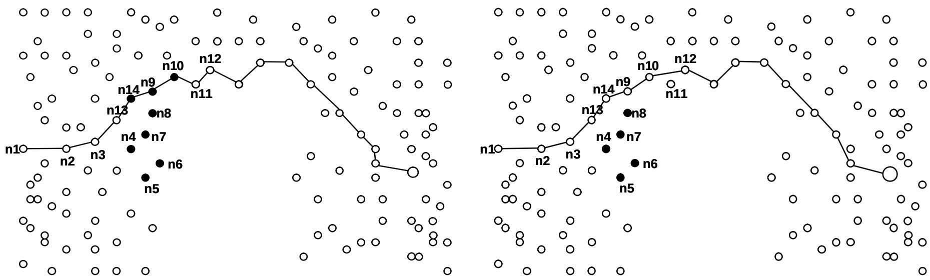
destination. Node $n6$ has no greedy node toward the destination, therefore the algorithm switches to rescue mode, with $n6$ as the perimeter entry point. Since none of the nodes $n7 - n10$ is closer to the destination than $n6$, all these nodes will use perimeter mode. All the nodes $n6 - n10$ will increase their negative counter and will be evaluated as non-optimal. $n11$ is closer to the destination than $n6$, therefore the routing mode will be switched to greedy. Greedy mode will be kept until the destination since all the remaining nodes on the path have neighbours closer to the destination than themselves.



(a) Behaviour based tagging (b) Neighborhood based tagging

Fig. 3. The path of the second message

Figure 2(b) shows the path of the same node when neighborhood based tagging is used. Nodes $n1 - n4$ have a neighbor closer to the destination than themselves. Therefore they are marked as optimal. Nodes $n6 - n8$ have no neighbor closer to the destination than themselves, therefore they are marked as non optimal. Starting from $n9$, the nodes are optimal again. Similar with Fig. 2(a), $n6$ is the perimeter entry point, and $n10$ is the perimeter exit point. At this step, neighborhood based tagging has no influence on the routing method.



(a) Behavior based tagging (b) Neighborhood based tagging

Fig. 4. The path of the n-th message

Figure 3 shows the path of the second message between the same source and destination. In both cases, $n4$ will choose the neighbor tagged as optimal and closer to the destination - $n5$. In Fig. 3(a), $n5$ will have no optimal neighbor closer to the destination, therefore it will start perimeter mode and increase the negative counter, becoming non optimal. In Fig. 3(b), $n5$ has no optimal neighbour closer to the destination and will tag itself as non optimal.

In Fig. 4 we will see the path of a message after a few other retransmissions. NBT finds an optimal path around the obstacle, while BBT will have some nodes marked as non-optimal on a path that could use only greedy forwarding towards the destination.

4 Algorithm analysis

Each node makes routing decisions based on the optimality of the neighbors. Therefore each node has to inform its neighbors about its current state. There are several options for transferring this information. First is by piggybacking it on the network control messages - periodic beacon messages, advertising their current status and position. This solution is suitable for the case of frequent state changes (i.e. behavior based routing).

The second option is to send an status update to the neighbors each time a node changes its state. This is suitable for a small number of node state changes, such is the case for neighborhood based routing. We will further show that for a static network, the state of node can switch at most once. Therefore, this option is more suitable for our case. We propose first a separation of nodes into layers, as follows:

- Layer 0: Nodes that have no greedy neighbors toward the destination: $L_0 = \{n_i | S'_i = \emptyset\}$
- Layer 1: Nodes that have greedy neighbors toward the destination only nodes of Layer 0: $L_1 = \{n_i | \forall n_k \in S'_i, n_k \in L_0\}$.
- Layer n: Nodes that have greedy neighbors toward the destination only nodes of Layers 0..n-1: $L_n = \{n_i | S'_i = \{n_k | n_k \in L_0 \cup L_1 \ldots \cup L_{n-1}\}\}$.

Proposition 1. *The Neighborhood Based Tagging Algorithm is stable: the tag of a node is switched only once.*

Proof. The status of a node $n_i \in L_0$ depends only on the network topology. If it is static, then the status of n_i once tagged as non-optimal, remains unchanged. The status of a node $n_i \in L_1$ depends only on its neighbors $n_k \in S'_i$, but $\forall n_k \in S'_i, n_k \in L_0$, therefore, once evaluated, their state will not change either. Similarly, the state of a node $n_i \in L_n$ depend only on $n_k \in L_0 \cup L_1 \ldots \cup L_{n-1}$, which are stable, therefore the nodes $n_i \in L_n$ are stable as well.

Another issue is the size of the tagged area. The total number of non optimal nodes depends only on the density and the topology of the network (the relative position of destination toward the object, and the size of the object). We define the smallest density for which the the number of tagged nodes is both limited and proportional with the size of the marked area as the *critical density*. Experimentally, we found a critical density around 10.

For densities higher than critical density, the messages coming from sources for which exists a greedy path toward the destination, will generate the detection of a limited number of non optimal nodes before finding this greedy path that they will use afterwords, as shown in Fig. . Further we will prove that the algorithm preserves the greedy paths.

Theorem 1. *If there is a path $P = n_0, n_1, ..., n_i$ between a source s and a destination d, such that*

$$dist(n_i, d) > dist(n_{i-1}, d)...dist(n_2, d) > dist(n_1, d) > dist(n_0, d)$$

then no node $n_k \in P$ is tagged as non-optimal.

Proof. We proof the theorem by induction. The node n_0 is directly connected to the destination d, therefore it is *optimal.* The node n_1 has a neighbor closer to the destination, the node n_0, therefore it is *optimal.* We assume that the node n_{i-1} is optimal. Then n_i has an *optimal* neighbor toward the destination, therefore it is *optimal.*

Corollary 1. *If we can enclose the obstacle in a region such that for all the nodes outside this region it exists a greedy path toward the destination, then the marked region cannot exceed this region around the obstacle.*

In order to extend the suitability of the algorithm for any network configuration - nodes density smaller than the critical density, we redefine our algorithm by considering a new parameter during the optimality evaluation: the layer to which a node belongs, as defined at the begining of this section. We will shouw that the size of a layer is finite and if we limit the number of layers of marked nodes, then the algorithm is convergent to a stable state.

Proposition 2. *The size of a layer is finite.*

Proof. By induction on i.

Basis i=1 The size of Layer 0 is proportional with the object, therefore finite. A node in Layer 1 must have at least one greedy neighbor in Layer 0, it has to be in the transmission range of a node in Layer 0. Therefore the size of the Layer 1 is proportional with the size of Layer 0 and finite.

Inductive step Suppose the size of Layers 0,1,2.. n-1 is finite. The nodes in Layer n have only greedy neighbors in one of the lower ranked layers. Therefore the size of the Layer n is finite.

The algorithm is convergent if the number of layers is finite. We can limit the number of layers by introducing a new parameter, a layer threshold. If a non optimal node is detected in a layer above this limit, then it will not switch its state. This will limit the evaluation to the nodes in the vicinity of the obstacle.

5 Simulation results

In this section we numerically validate the expected behavior and performance of our algorithms. The simulations we present compare our geographic routing algorithm and the well known greedy face greedy (GFG) algorithm which is considered a reference algorithm in the state of the art. Additionally we compare with a similar tagging based class of heuristic algorithms, described in [17].

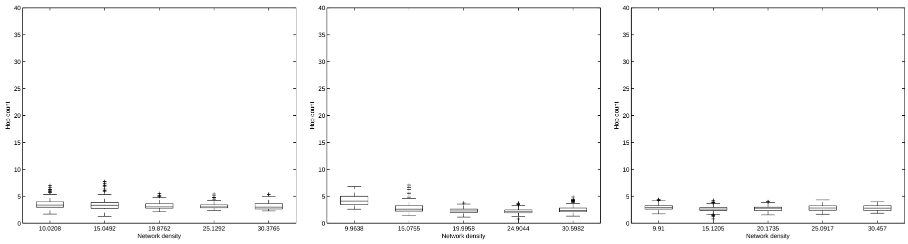
To make the comparison, the criteria we are interested in are (a) whether the tagging algorithm is convergent: whether the number of tagged nodes becomes constant after some time, (b) the total number of tagged nodes and (c) the performance in terms of path length and hop counts. The numerical experiments show that our algorithm competes well with the GFG and behavior based routing evaluation in terms of the total number of nodes on the routing paths, while reducing the number tagged nodes, thus the topology control traffic.

5.1 Details on the experiments and the representation of results

The experiments are made with a network of nodes randomly distributed on a 200x200 units area. The size of the object (rectangulary shaped) is 30x50 units and the position of the upper left corner is 70x110. The transmission range of the nodes is constant, equal to 7 units, The total number of nodes varies between 2800 and 7900 such as to obtain different densities between 10 and 30.

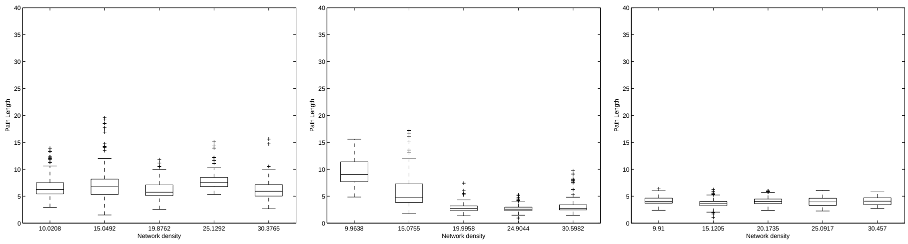
For each step of the simulation, a new message is sent from a random source to a single destination (110,85), such that all the trajectories will intersect the object. The initial network setup is similar with Fig. 2. Within a step, a node that acts as a relay reads all the messages sent by its neighbors in the previous step and schedules them for retransmission within this step.

Each experiment is repeated 100 times with a different network topology, and the outcomes are presented in a box plot graphic. Box plots are composed of a box with the lower line being the lower quartile, the middle one the median and the upper one being the upper quartile of the sample. The dashed lines extending above and below the box show the span of the other samples. The plus sign represents outliers.



(a) GFG (b) Behavior based tagging (c) Neighbors based tagging

Fig. 5. Number of hops



(a) GFG (b) Behavior based tagging (c) Neighbors based tagging

Fig. 6. Path length

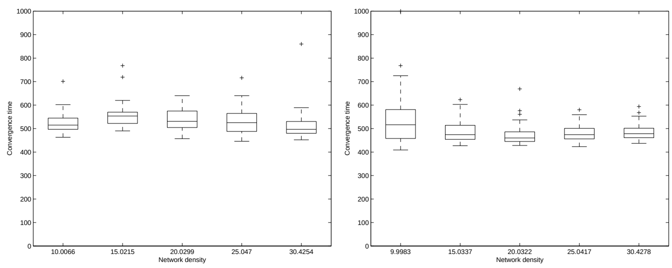
5.2 Performance evaluation

The performances in terms of path length for the three algorithms are presented in Fig 6. We are evaluating the path stretch - defined as the ratio between the total path length of a message and the minimum euclidian distance between the source and the destination, while taking into account the presence of the obstacle.

For the smallest two densities considered, *BBT* has a major drawback: it performes worse than *GFG*. The reason is the influence of voids on the routing mode. Nodes are using perimeter routing due to the presence of the voids, therefore the size of the marked area will be increased by the lack of nodes, having as a consequence an increase of path lengths. For these densities, our protocol reduces with 50% the path stretch obtained by *BBT*. Therefore, we extend the suitablity of the early obstacle avoidance to a broader range of densities. It reduces for all densities the path stretch obtained by *GFG* with 30%. We extend the suitablity of the early obstacle avoidance to a broader range of densities. Still, *BBT* has slightly better performances for the highest densities: it has a decrease of 10% of the path stretch of *NBT* (but with 4 times more nodes marked).

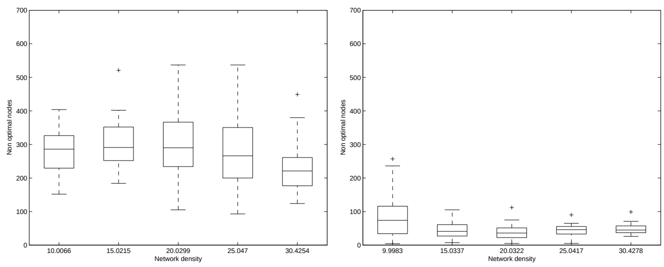
Figure 5 shows the hops stretch of a message sent from a source to a destination. It is measured as the ratio between the number of hops of a message between the source and the destination, and the ideal number of hops (measured as the ratio between the euclidian path length described above and the transmission radius). The simulations show that for the lowest density NBT improves with 30% the performances of BBT and with 20% the performance of GFG.

We note that the overhead in our algorithm is independent of the network size. Thus our method scales well. Furthermore, additional messages are sent only once, i.e. the overhead is independent of the number of events generated in the network, while all messages routed around the obstacle benefit of smaller paths. Overall, the overhead impossed by tagging nodes is much less compared to the saving in routing messages. As an example, for routing 10 messages, we save 10 times the path gain (in this case 20 hops per message) i.e a total of 200 transmissions, while we spend only 50 messages for tagging. The convergence



(a) Behavior based tagging (b) Neighbors based tagging

Fig. 7. Convergence time



(a) Behavior based tagging (b) Neighbors based tagging

Fig. 8. Number of tagged nodes

time for the two strategies is compared in Fig. 7. The variations are small, although the evaluation methods are different.Let the convergence time be the time when the number of tagged nodes remained unchanged for the last 300 steps. Therefore we consider that the algorithm is fast convergent.

A significant difference can be noticed with respect to the number of tagged nodes (Fig. 8): NBT will mark only 1/4 of the nodes marked by BBT. Another important observation is that the number of tagged nodes does not increase for higher densities. The reason is that the geometrical surface covered by tagged nodes decreases as well with the increase in density. The probability that a node has greedy neighbors toward the destination is direct proportional to the density. Since only tagged nodes transmit overhead messages, and since this is done only once, reducing the number of tagged nodes leads to a smaller overhead.

6 Conclusions

This paper presented an algorithm for early detection and avoidance of obstacles, by progressive evaluation of the nodes making routing decisions.We proved several properties of the algorithm: stability, convergence and we showed that it preserves previous properties of the geographic routing algorithms.

The simulations show the performances of the proposed algorithm, better then those of the state of the art algorithms. At the same time, the algorithm is lightweight, it needs only 1 bit of information piggybacked on the topology maintenance messages, or sent reactively, and only one extra bit of storage for each neighbor.

The complexity is low - for a fixed destination the overhead introduced depends only on the obstacle size and shape, while it is independent of the network size. Furthermore, this overhead is paid only once, independently of the load of the network, while all messages benefit of reduced path length. Additionally, the algorithm is flexible, it can be used on top of a large class of routing and planarisation algorithms. At the same time it is independent on the physical layer model used.

Future work will consider different assumptions for network topology: multiple base stations and mobile base station.

References

1. Prosenjit Bose, Pat Morin, Ivan Stojmenović, and Jorge Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wirel. Netw.*, 7(6):609–616, 2001.
2. I. Chatzigiannakis, T. Dimitriou, S. Nikoletseas, and P. Spirakis. A probabilistic algorithm for efficient and robust data propagation in smart dust networks. *Ad-Hoc Networks Journal*, 4(5), 2006.
3. Ioannis Chatzigiannakis, Sotiris Nikoletseas, and Paul G. Spirakis. Efficient and robust protocols for local detection and propagation in smart dust networks. *Special Issue on Algorithmic Solutions for Wireless,Mobile, Ad Hoc and Sensor Networks, ACM/Baltzer Mobile Networks and Applications(MONET) Journal*, 10(1-2):133–149, 2005.
4. D. Chen and P.K. Varshney. A survey of void handling techniques for geographic routing in wireless networks. *Communications Surveys and Tutorials, IEEE*, pages 50–67, 2007.

5. Susanta Datta, Ivan Stojmenovic, and Jie Wu. Internal node and shortcut based routing with guaranteed delivery in wireless networks. *Cluster Computing*, 5(2):169–178, 2002.
6. Qing Fang, Jie Gao, and Leonidas J. Guibas. Locating and bypassing holes in sensor networks. *Mob. Netw. Appl.*, 11(2):187–200, 2006.
7. K.R. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis, 1969.
8. M. Heissenbüttel, T. Braun, T. Bernoulli, and M. Wälchli. BLR: Beacon-less routing algorithm for mobile ad-hoc networks, 2003.
9. R. Jain, A. Puri, and R. Sengupta. Geographical routing using partial information for wireless ad hoc networks, 1999.
10. Brad Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Mobile Computing and Networking*, pages 243–254, 2000.
11. Young-Jin Kim, Ramesh Govindan, Brad Karp, and Scott Shenker. Geographic routing made practical. In *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*, pages 217–230, Berkeley, CA, USA, 2005. USENIX Association.
12. Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia. Compass routing on geometric networks. In *Proc. 11 th Canadian Conference on Computational Geometry*, pages 51–54, Vancouver, August 1999.
13. F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: Of theory and practice, 2003.
14. Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger. Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing. In *Proc. 4^{th} ACM Int. Symposium on Mobile Ad-Hoc Networking and Computing (MobiHoc)*, 2003.
15. Xu Lin, Mouhsine Lakshdisi, and Ivan Stojmenovic. Location based localized alternate, disjoint, multi-path and component routing schemes for wireless networks. In *MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 287–290, New York, NY, USA, 2001. ACM.
16. Rudolf Mathar and Jürgen Mattfeldt. Optimal transmission ranges for mobile communication in linear multihop packet radio networks. *Wirel. Netw.*, 2(4):329–342, 1996.
17. Luminita Moraru, Pierre Leone, Sotiris Nikoletseas, and José D. P. Rolim. Near optimal geographic routing with obstacle avoidance in wireless sensor networks by fast-converging trust-based algorithms. In *Q2SWinet '07: Proceedings of the 3rd ACM Workshop on QoS and security for wireless and mobile networks*, pages 31–38, New York, NY, USA, 2007. ACM.
18. S. Nikoletseas and O. Powell. Simple and efficient geographic routing around obstacles for wireless sensor networks. In *Proceedings of the 6th International Workshop on Efficient and Experimental Algorithms (WEA), Lecture Notes in Computer Science (LNCS)*, pages 161–174. Springer-Verlag, 2007.
19. G. Fan S. Chen and J. Cui. Avoid void in geographic routing for data aggregation in sensor networks. *International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC), Special Issue on Wireless Sensor Networks.*
20. Ivan Stojmenovic and Xu Lin. Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks. *IEEE Trans. Parallel Distrib. Syst.*, 12(10):1023–1032, 2001.
21. G. Toussaint. Some unsolved problems on proximity graphs, 1991.
22. Jorge Urrutia. Routing with guaranteed delivery in geometric and wireless networks. pages 393–406, 2002.