HAL open science

Efficient Greedy Geographical Non-Planar Routing with Reactive Deflection

Fabrice Theoleyre, Eryk Schiller, Andrzej Duda

To cite this version:

Fabrice Theoleyre, Eryk Schiller, Andrzej Duda. Efficient Greedy Geographical Non-Planar Routing with Reactive Deflection. ICC 2009 - 2009 IEEE International Conference on Communications, Jun 2009, Dresden, Germany. pp.1-5, 10.1109/ICC.2009.5198972 . hal-02650657

HAL Id: hal-02650657

https://hal.science/hal-02650657v1

Submitted on 29 May 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Efficient Greedy Geographical Non-Planar Routing with Reactive Deflection

Fabrice Theoleyre, Eryk Schiller, and Andrzej Duda
Grenoble Informatics Laboratory
CNRS and Grenoble-INP, France
Email: {firstname.lastname}@imag.fr

***Abstract*—We present a novel geographical routing scheme for spontaneous wireless mesh networks. Greedy geographical routing has many advantages, but suffers from packet losses occurring at the border of *voids*. In this paper, we propose a flexible greedy routing scheme that can be adapted to any variant of geographical routing and works for any connectivity graph, not necessarily Unit Disk Graphs. The idea is to reactively detect voids, backtrack packets, and propagate information on blocked sectors to reduce packet loss. We also propose an extrapolating algorithm to reduce the latency of void discovery and to limit route stretch. Performance evaluation via simulation shows that our modified greedy routing avoids most of packet losses.**

***Index Terms*—greedy geographical routing, detection of voids, packets deflection, wireless mesh networks**

I. Introduction

We consider wireless mesh networks composed of a large number of wireless routers providing connectivity to mobile nodes. They begin to emerge in some regions to provide cheap network connectivity to a community of end users. Usually they grow in a *spontaneous* way when users or operators add more routers to increase capacity and coverage.

We assume that mesh routers benefit from abundant resources, may only move, quit, or join occasionally, so that the topology of a typical mesh network stays fairly stable. The organization of mesh networks needs to be *autonomic*, because unlike the current Internet, they cannot rely on highly skilled personnel for configuring, connecting, and running mesh routers. Spontaneous growth of such networks may result in a dense and unplanned topology with some uncovered areas.

Unlike traditional approaches, *geographical routing* presents interesting properties for spontaneous wireless mesh networks: it does not require any information on the global topology since a node choses the next hop among its neighboring routers on the basis of the destination location. Consequently, the routing scheme is scalable, because it only involves local decisions. Geographical routing is simple, because it does not require routing tables so that there is no overhead for their creation and maintenance. Joining the network is also simple, because a new mesh router only needs an address based on its geographical position. Such addresses can be obtained from a dedicated device (e.g. GPS) or with methods for deriving consistent location addresses [1], [2]. The most familiar variant of geographical routing is *greedy forwarding* in which a node forwards a packet to the neighbor closest to the destination [3], [4]. Greedy forwarding guarantees loop-free operation, but packets may be dropped at *blocked nodes* that have only neighbors in the backward direction. Blocked nodes appear at some places near uncovered areas (*voids*) or close to obstacles.

Our main contribution is to propose a new greedy routing that correctly deals with voids. First, we define a new mechanism to reactively detect voids and surround them, which significantly reduces packet loss. The information of detected voids propagates backwards so that subsequent packets to the same direction benefit from this reactive detection. Second, we propose a mechanism in which voids deviate packets and shorten the length of a route compared to classical approaches. Our routing scheme works in any network topology independently of whether it corresponds to a planar graph or not.

We start by describing the related work on geographical routing in Section II. Section III presents the details of the proposed new greedy routing protocol. Then, we evaluate its performance via simulation in Section IV and conclude. The reader can read [5] for a more complete version of this article.

II. Related Work

Geographic information can largely reduce complexity of routing in spontaneous mesh networks. The most simple and widely used protocol is greedy geographic routing [3], [6], [7], [8]: when a node receives a packet, it uses the following forwarding rule:

> ”forward the packet to the node with the best improvement”.

Improvement is usually defined with respect to the distance towards the destination. Since improvement is not negative, there is no routing loop and the algorithm is scalable because the decision is local.

We assume that the nodes either have a dedicated localization device (e.g. GPS) or are able to estimate their positions distributively (e.g. [1], [9] that use respectively triangulation and angle of arrival). A pragmatic approach to this problem is to assume that a subset of mesh routers know their exact positions via GPS devices and other nodes can compute their positions with respect to its neighbors [10].

The main drawback of greedy geographical routing is packet loss at blocked nodes near voids or obstacles. A node must

This work was partially supported by the European Commission project WIP under contract 2740, the French Ministry of Research project AIRNET under contract ANR-05-RNRT-012-01 and ARESA under contract ANR-05-RNRT-01703.

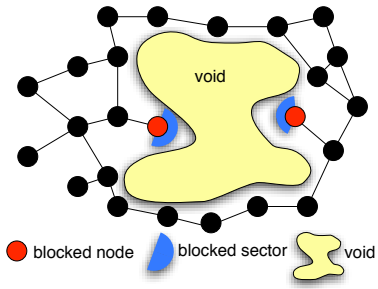


Fig. 1. Blocked nodes in greedy geographical routing.

drop a packet when the improvement associated with any of its neighbors is negative (cf. figure 1). In *face routing* the *left-hand* rule [4] tries to go around a void, but it requires the connectivity graph of nodes to be planar. Relative Neighborhood Graphs can yield planar graphs for Unit Disk Graphs (UDG) [11], but in real wireless environments, the conditions for obtaining planar graphs are not satisfied due to asymmetric links and not circular radio coverage [12]. To the best of our knowledge, there is no efficient and localized planarization algorithm proposed for a general connectivity graph. A possible solution to this problem is the following method: a border node initiates local flooding to find the next hop closer to the destination [13]. However, it results in long delays and significant overhead. Fotopoulou et al. [14] propose to adapt this method to establish and maintain a *virtual circuit*. Funke et al. [15] propose an algorithm inspired by topological geometry to discover the void limits, i.e. the *border nodes* by creating isosets. However, this method does not work for non UDG since isosets are not rings in general connectivity graphs.

We propose here a generic method to deal with voids by backtracking packets and discovering blocked areas near voids: deflection routing deviates packets outside them. Thus, the algorithm presented here is perfectly supplementary to existing methods: deflection improves greedy routing by trying to surround voids and any other technique presented above can be used when a void is reached.

III. Reactive Deflection

Geographical routing is attractive for mesh networks, but suffers from two main drawbacks: blocked nodes can drop many packets and the route length may drastically increase when a surrounding mechanism tries to deviate a packet around a void (e.g. the left-hand rule in UDG). In this paper, we assume a general connectivity graph and propose to reactively detect blocked nodes and locally advertise blocked sectors to avoid packet losses. Such a technique is efficient in any type of networks and graphs since it does not assume any particular graph property.

Detection of blocked nodes can be done in a proactive way: locally flood information to detect voids. For example, we can discover the topology of the wireless mesh to detect elementary cycles in which no other node is located inside the ring. The location of nodes helps to surround voids. However, such an approach requires a complete knowledge of the mesh topology and is computationally intensive.

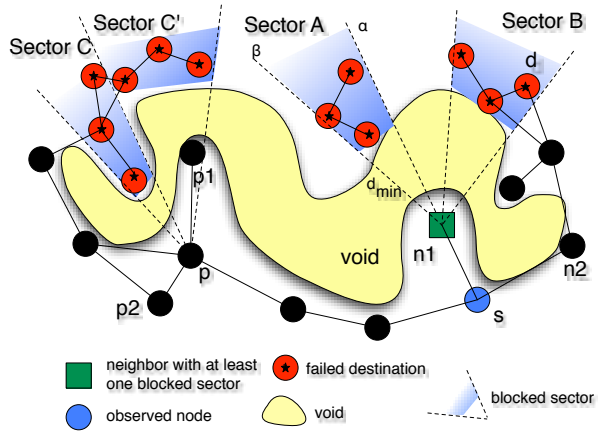


Fig. 2. Examples of blocked nodes and blocked sectors.

In opposition to this approach, we have chosen a reactive method: a node becomes *blocked* with respect to a given destination when it cannot forward a packet to any neighbor closer to the destination. Hence, the part of the network not concerned by forwarding this packet does not generate any control traffic so that this approach is more scalable.

Let us first adopt the following notation:

- $d(A, B)$: the Euclidean distance between the geographical coordinates of nodes A and B
- $\angle(AB, AC)$: the oriented angle between two coordinates of nodes (A, B) and (A, C) (by convention, we denote by $\angle(AB)$ the normalized angle $\angle(\begin{pmatrix}1\\0\end{pmatrix}, AB)$)
- $\mathbb{S}(S, \alpha, \beta, d_{min})$: the sector of node S composed of all nodes N such that $\alpha \leq \angle(SN) \leq \beta$ and such that $d(S, N) \geq d_{min}$

In our approach, a node chooses a neighbor closer to the destination and not blocked for this direction. If a node fails to forward a packet to a given destination, it will consider itself as blocked for this direction. It will advertise backwards a list of blocked directions so that its neighbors will not choose it as a next hop for these directions. If several non blocked neighbors exist, the forwarder chooses the neighbor closest to the destination, i.e. with the best *improvement*.

For advertising blocked directions, we propose to use the notion of *blocked sectors*: a node N advertises that it is blocked for any destination that falls in sector $\mathbb{S}(N, angle_{min}, angle_{max}, dist_{min})$. Let us consider the topology illustrated in Figure 2. Node n_1 wants to forward a packet to destination d and it discovers that it is blocked for this destination since no neighbor exists in this direction. Thus, it backwards the packet to s with its two blocked sectors. Based on this information, s marks n_1 as blocked and forwards the packet to another neighbor closer to d (node n_2 in this case).

To limit the overhead, a node tries to merge all its blocked sectors before advertising them. It can only merge overlapping sectors having the same minimal distances (within some tolerance Δ_d). Otherwise, the merged blocked sector may include nodes that are reachable—consider for instance the

topology of figure 2: if node p merges sectors C and C', node p_1 may appear in the blocked sector. Thus, it would become unreachable from p_2. Clearly, we must avoid such a merging. Only sectors with the same d_{min} will be merged : tolerance Δ_d allows some merging of sectors with approximately equal minimal distances.

More formally, node N executes Algorithm 1. Procedure $ReactiveDeflection()$ finds the next hop for forwarding a packet to destination D: the next hop must be closer to the destination and must be unblocked for D. If it does not return any node, it means that node N becomes *blocked* for destination D (variable BLOCKED(N,D) becomes true). Thus, node N updates its blocked sectors and sends the packet backwards to the previous hop with its list of blocked sectors piggybacked onto the packet. This scheme is loop-free: when a node sends a packet backwards, the receiver will update its blocked sectors and it cannot choose the same next hop for subsequent packets, because Algorithm 1 does not forward packets to blocked nodes.

Algorithm 1 ReactiveDeflection(N→D)

```
next ← ∅
for all n ∈ Neighbors do
    if d(n,D) < d(N,D) and !BLOCKED(n,D) and
    d(n,D) < d(next,D) then
        next ← n
    end if
end for
if next = ∅ then
    Blocked(N,D) ← true
    next ← previous_hop
end if
return next
```

In networks with non UDG topologies, when a node does not have any neighbor closer to the destination, it is blocked. Such a border node has to discover a *virtual* next hop in a larger vicinity[3], [14]. A border node can for example flood at a fixed distance to discover a node closer to the destination. However, the packets still keep on being forwarded until they reach these blocked nodes, even if the packets must after that turn back to surround a void.

Note that when we reduce packet loss with the previously described algorithm, we also reduce in the long term the route length. Indeed, the nodes around the void discover that they have blocked sectors. When they propagate the information about blocked sectors, nodes with all blocked neighbors also become blocked for this destination. Finally, each node discovers a blocked area and forwards packets outside this area. However, we need several useless packet transmissions and backtracking before the network converges, and blocked sectors are correctly constructed. We propose a mechanism to accelerate the convergence of this propagation process by extrapolating the location of a blocked area.

We propose to detect the border of a void based on only local neighborhood knowledge. We will show that even if a node has only local knowledge, i.e. about nodes at a limited distance, voids can be efficiently surrounded. When a node must transmit a packet backwards, it locally floods an `hello` packet containing the list of its neighbors and blocked sectors in a k hop scope.

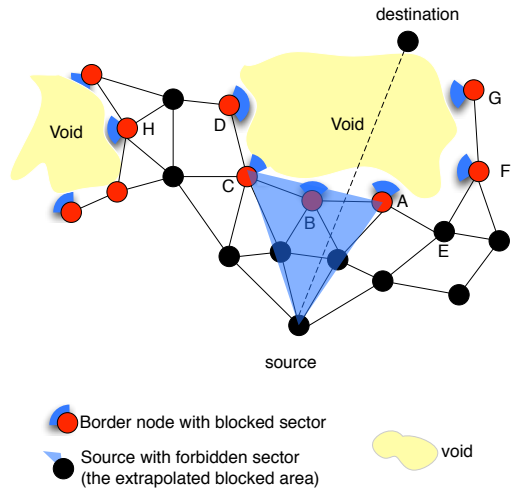


Fig. 3. Method for locally detecting the border of a void.

To detect the border of a void, node N first searches for the blocked k-neighbor closest to the direction of the destination D, i.e. minimizing angle $\angle((N,D),(N,BN))$ for all blocked nodes BN. Then, N constructs the Maximum Connected Set of blocked nodes that contains BN: it adds BN to this set, and recursively adds all its blocked neighbors. Finally, N computes the *forbidden sector* that spans the maximum connected set—it extrapolates the blocked area.

Figure 3 illustrates void detection with the knowledge of the 3-neighborhood topology. First, the source node detects if it knows a node with a blocked direction and takes the closest one to the direction of the destination. In the example the blocked node is B. Then, the source constructs the connected set of blocked nodes that includes node B: it obtains set $\{A,B,C,D\}$. Obviously, node F is not present in the set since it is not connected to A via other blocked nodes. In the same way, border node H is not in set $\{A,B,C,D\}$, because it is 2 hops away: it is border to another void. Finally, we obtain the forbidden sector for the destination. We can note that node E is not blocked since it can choose node F when A is blocked: E will never be blocked for the direction.

Algorithm 2 presents the modified protocol. Function ISINFORBIDDENSECTOR computes the forbidden sector and returns TRUE, if the node is located inside this sector. Function CLOSERTOSECTORLIMITS(P,Q) returns TRUE, if P is closer than Q to the forbidden sector limits.

In other words, if some next hops exist and do not lie in the computed forbidden sector, we choose the best one. Otherwise, if all possible next hops are in this forbidden sector, we choose the node closest to the limits of the forbidden sector. With this modified routing scheme, we forward packets outside the forbidden sector, because a void appears as something repellent to packets by creating forbidden sectors in a distributed manner while keeping routing loop-free.

Algorithm 2 ModifiedReactiveDeflection(D,ForbiddenSector)

```
next ← ∅
for all n ∈ Neighbors do
    if d(n, D) < d(S, D) and !BLOCKED(n,D) then
        if !ISINFORBIDDENSECTOR(n) and { ISINFORBIDDENSECTOR(next) or d(n, D) < d(next, D) } then
            next ← n
        else if ISINFORBIDDENSECTOR(next) and ISINFORBIDDENSECTOR(n) and CLOSERTOSECTORLIMITS(n,next) then
            next ← n
        end if
    end if
end for
if next = ∅ then
    Blocked(N,D) ← true
    next ← previous_hop
end if
return next
```

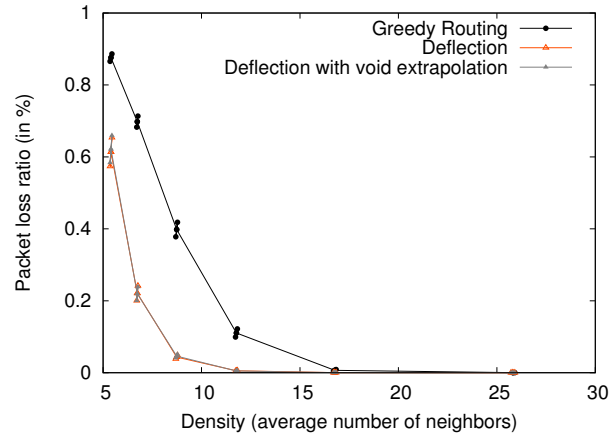
IV. Performance evaluation

We generated random meshes of 1000 nodes with two models: UDG [16] and what we call a *proxi-graph* (a graph based on proximity). In a proxi-graph, each node chooses a *radio range* following a Gaussian distribution centered at 1 with standard deviation Std depending on the radio range (here, $Std = 25\% \cdot (radio - range)$). The proxi-graph owns a rectangular central void of size two fifth of the simulation disk radius. Besides, we discard disconnected topologies and use a disk simulation area to reduce border effects. Data traffic consists of 1,000 flows of 10 packets each from a random source to a random destination. A node advertises its neighborhood and blocked sectors 3 hops far since it achieves the best performances, as highlighted below. We evaluated the properties of routing itself: we assume ideal radio and MAC layers, i.e. packets do not experience any loss due to channel or MAC behavior to only test routing properties. We also simulated this protocol with ns2 to take into account more realistic radio conditions, but do not report here these results because of lack of space. The reader can refer to [5] for a more exhaustive view of all the results.

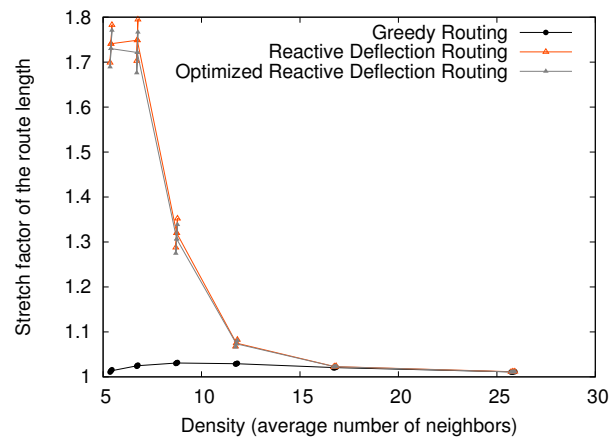
We compare our routing algorithm with greedy geographic routing to quantify the reduction of packet loss: a node chooses the neighbor closest to the destination (other variants that minimize the angle difference, or the hop length do not exhibit better results). We measured packet losses (the proportion of packets sent by a source that never reach the destination), route length (the average route length in hops for the delivered packets), and stretch factor (the average ratio of the route length for a packet and the length of the shortest route for the associated source/destination pair). We plot the average values and the associated 95% confidence intervals.

A. Performance for Unit Disk Graphs

We first measure the route length obtained for deflection routing with different values of the k-neighborhood with density of 8 neighbors per node in Unit Disk Graphs. For densities of (1,2,3,4,5) neighbors, we obtain route lengths of respectively (33.6, 33, 32.7, 33.2, 33.6) hops. Thus, we chose $k = 3$ since it provides an accurate route length and limits the overhead. Besides, for large k values, deflection routing tends to overestimate the size of voids and increase the route length.



(a) Packet loss ratio

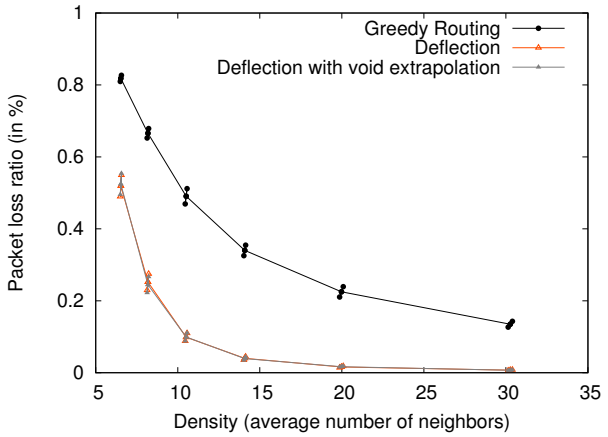


(b) Stretch factor

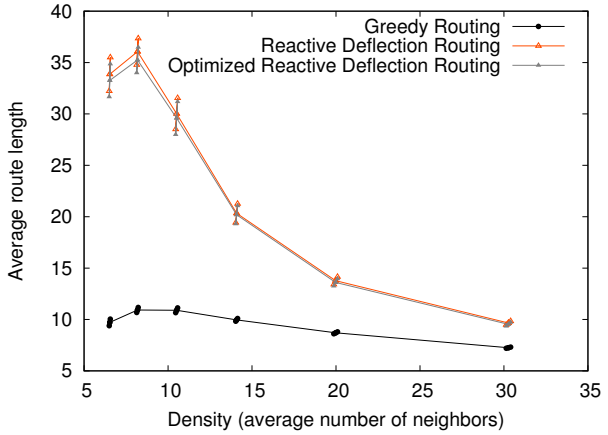
Fig. 4. Performances under different routing schemes for UDG.

Then, the following experiment shows (fig. 4(a)) that packet loss for greedy routing decreases with the increase of density: probability of having a large area without any node decreases so that voids are less probable to appear. However, more than 70% of packets are lost for low density. On the contrary, the proposed routing scheme lowers packet loss: almost no packet is lost (less than 4%) when density exceeds a small threshold (8 neighbors per node). Thus, nodes reroute less packets by means of reactive discovery so that the overhead is lower and delay improved. We can also notice that route length optimization has no impact on delivery ratio.

We have also measured the route length stretch factor (fig. 4(b)). Greedy routing drops many packets when voids exist. Moreover, the packet drop probability is larger when the destination is farther. Since the route length is only measured for delivered packets, a poor delivery ratio creates mechanically a lower average route length. We can observe



(a) Packet loss ratio



(b) Route length

Fig. 5. Performances under different routing schemes for a proxi-graph with one central void.

that our optimized algorithm succeeds in slightly reducing the route length. More importantly, route length optimization results in deviating packets from voids and decreasing the load on the void's borders. The stretch factor is larger for low density since more voids are present and packets must be deflected and backtracked more often. The reader can note that greedy routing forwards one packet until it reaches a blocked node: this increases the load on these block nodes, even if they implement a void bypassing. Finally, the route length is reduced only for very low densities with the optimized version of deflection routing since blocked sectors interpolating is working only when voids are sufficiently large. Moreover, we could over-estimate the forbidden sector by adding a guarding angle around the blocked nodes: we would reduce the route length, but increase packet drops since we would over-estimate the presence of voids.

B. Performance for a proxi-graph with one rectangular void

We have evaluated packet loss rate in a proxi-graph with one central void (fig. 5(a)). We can see that packet loss increases compared to UDG, particularly for high density: dead-ends are more probable. Moreover, since the graph is not UDG, a node may choose a next hop in a greedy way although it does not have any neighbor in the direction of the destination. Thus, to increase density it is not sufficient to surround voids. We can remark that with our algorithms we significantly reduce packet loss ratio.

Finally, we have measured the route length (fig. 5(b)). We can remark the same trend as for UDG and for a proxi-graph with a void. Obviously, the route length is longer, because packets have to surround the rectangular void.

V. Conclusion

We have proposed a scheme for greedy geographical routing with reactive defect detection. The idea is to reactively detect blocked nodes and propagate the defect information by computing a set of blocked sectors. To reduce the route length and accelerate void detection in dense mesh networks, we have also proposed a method to extrapolate void location. Simulation results show packet loss as well as the route length decrease compared to greedy routing.

References

[1] S. Capkun, M. Hamdi, and J.-P. Hubaux, "Gps-free positioning in mobile ad hoc networks," *Cluster Computing*, vol. 5, no. 2, pp. 157–167, April 2002.

[2] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, and I. Stoica, "Geographic routing without location information," in *MOBICOM*. San Diego, USA: ACM, September 2003.

[3] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia., "Routing with guaranteed delivery in ad hoc wireless networks." in *DIAL'M*. Seattle, USA: ACM, 1999, pp. 48–55.

[4] B. Karp and H. T. Kung, "Greedy perimeter stateless routing for wireless networks," in *MOBICOM*. Boston, USA: ACM, August 2000.

[5] F. Theoleyre, E. Schiller, and A. Duda, "Efficient greedy geographical non-planar routing with reactive deflection," Grenoble Informatics Laboratory, http://hal.archives-ouvertes.fr/hal-00363811, Res. Rep., 2009.

[6] N. Carlsson and D. L. Eager, "Non-euclidian geographic routing in wireless networks," *Ad Hoc Networks*, vol. 5, no. 7, pp. 1173–1193, September 2007.

[7] P. He, J. Li, and L. Zhou, "A novel geographic routing algorithm for ad hoc networks based on localized delaunay triangulation," in *AINA*. Viena, Austria: IEEE, April 2006.

[8] P. Casari, M. Nati, C. Petrioli, and M. Zorzi, "Efficient non planar routing around dead-ends in sparse topologies using random forwarding," in *ICC*. Glasgow, UK: IEEE, June 2007.

[9] D. Niculescu and B. Nath, "Ad hoc positioning system (APS) using AOA," in *INFOCOM*. San Francisco, USA: IEEE, 2003.

[10] N. Bulusu, J. Heidemann, and D. Estrin, "Adaptive beacon placement," in *ICDCS*. Phoenix, USA: IEEE, April 2001.

[11] J. Cartigny, F. Ingelrest, D. Simplot-Ryl, and I. Stojmenovic, "Localized LMST and RNG based minimum-energy broadcast protocols in ad hoc networks," *Ad Hoc Networks*, vol. 3, no. 1, pp. 1–16, January 2005.

[12] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker, "Lazy cross-link removal for geographic routing," in *Sensys*. ACM, 2006.

[13] I. Stojmenovic and X. Lin, "Loop-free hybrid single path flooding routing algorithms with guaranteed delivery for wireless networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 12, no. 10, pp. 1–10, October 2001.

[14] S. Fotopoulou-Prigipa and B. McDonald, "GCRP: Geographic virtual circuit routing protocol for ad hoc networks," in *MASS*. Fort Lauderdale, USA: IEEE, October 2004.

[15] S. Funke, "Topological hole detection in wireless sensor networks and its applications," in *DIALM-POMC*. Cologne, Germany: ACM, September 2005, pp. 44–53.

[16] B. N. Clark, C. J. Colburn, and D. S. Johnson, "Unit disks graphs," *Discrete Mathematics*, vol. 86, pp. 165–177, December 1990.