Distributed Roadmap Aided Routing in Sensor Networks

Zizhan Zheng, Kai-Wei Fan, Prasun Sinha, Yusu Wang
Department of Computer Science and Engineering, The Ohio State University
Email: {zhengz, fank, prasun, yusu}@cse.ohio-state.edu

Abstract

Communication between arbitrary pairs of nodes has become critical to support in emerging sensor networking applications. Traditional routing techniques for multi-hop wireless networks either require high control overhead in computing and maintaining routes, or may lead to unbounded route-stretch. In order to bound the route-stretch, we propose a distributed shortest-path roadmap based routing paradigm that embodies two ideas: routing hole approximation that summaries the critical information about hole boundaries and controlled advertisement that advertises the boundary information of each hole within limited neighborhoods. We show that our approach makes a desired tradeoff between the worst case route-stretch and the message overhead through both analysis and simulations.

1 Introduction

With emerging sensor applications where packets may originate from anywhere in the network and may be destined to any node, such as pursuer-evader tracking [3] and battlefield monitoring [12], pairwise communication between arbitrary sensors becomes an essential requirement in sensor networks. New functionalities such as in-network storage [11] also require communication between arbitrary nodes. Stateful routing protocols designed for multi-hop wireless networks can incur high communication and storage overhead (up to $O(n)$ per node in the worst case where n is the number of nodes in the network), and therefore are not suitable for resource constrained sensor networks. Although stateless routing protocols based on geographic information have been proposed [2, 6, 7], and perform well in networks with dense deployments, their performance can be severely impacted in presence of holes in the network.

Geographic routing protocols typically operate in two phases. A packet is greedily forwarded towards the destination until a "local minimum" is reached, where the forwarding node has no neighbor closer to the destination due to the incidence of a routing hole. A recovery phase is then followed to bypass the hole until the greedy phase can be continued. Due to the reactive routing decisions upon encountering a hole, the discovered path may be substantially longer than the shortest path in terms of the number of hops, especially when the hole is large. Figure 1 shows a scenario from our simulations. The path found by GPSR [6], a classic geographic routing protocol, has 66 hops. However if the hole is known by the nodes in its neighborhood and bypassed in advance, the path length can be lowered to 26 hops. Large holes may exist in a network due to the presence of large obstacles such as a building or a lake, or an irregular deployment of sensing nodes.

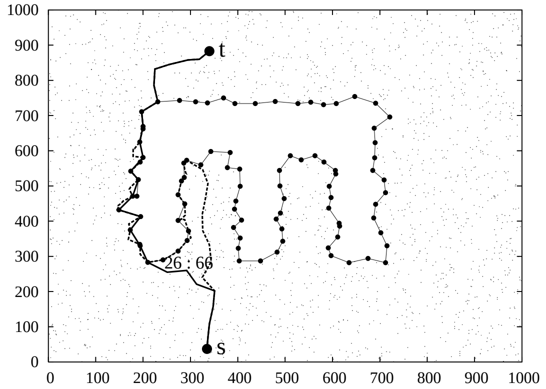


Figure 1. Geographic routing can result in sub-optimal routes and high route-stretch. The dashed line is the routing path of GPSR, and the straight line is the routing path if the hole can be bypassed before its boundary is touched. The boundary nodes of the hole are highlighted.

One of the critical metrics for routing in sensor networks is the route-stretch, which is the ratio of the number of hops on the computed route to the number of hops on the shortest route. The route-stretch is closely related to the end-to-end latency as well as the energy consumption. In Figure 1, as the "concave" regions of the hole can be arbitrarily deep, the stretch is unbounded for GPSR. Some recent works have dealt with holes explicitly so that packets can bypass holes in advance without getting trapped [1,5,14]. However, none of these works ensure a bound on route-stretch and some of them [5, 14] only support "many to one" communications, where all the data packets are forwarded towards a single sink. The key observation is that *the presence of a hole has to be made known to at least the nodes in the vicinity of the hole to bound the stretch*.

We propose a distributed shortest-path roadmap based

1-4244-2575-4/08/$20.00 ©2008 IEEE

routing technique where each routing hole is treated as a polygonal obstacle. Instead of handling holes passively, we proactively advertise information on holes, but within a controlled region where route-stretch is most affected by the holes. Our approach is composed of two components:

- **Hole approximation and advertisement**: Each routing hole is approximated by its *core*, a simple polygon enclosed by the hole, controlled by a single parameter α. Each routing core is then flooded to the k-hop neighborhood of the hole, where k is proportional to the size of the hole.
- **Hole bypassing routing**: Each node sets up a shortest-path roadmap [8] locally by treating the cores it knows as obstacles, and makes routing decisions based on that map. The real path mimics the planned one and is realized by greedy forwarding and hole traversing.

The contributions of this paper are:

- We propose a protocol for routing between any pair of nodes in a sensor network, which has low storage and message overhead, and guarantees a bounded route-stretch.
- We prove that the route-stretch of our protocol is bounded by $1/\cos(\alpha/2)$ if all the cores are advertised to the entire network. We also derive a bound for the route stretch when the cores are advertised locally.

In the next section, we present the main idea of the hole bypassing routing protocol. The details on the approximation and advertisement of routing holes are discussed in Section 3, and the bounds on route-stretch are analyzed in Section 4. A mechanization that further optimizes our protocol can be found in [13], and is omitted due to space constraints. Our approach is compared with GPSR [6] and GLDR [10] in Section 5 through simulations. The related work are summarized in Section 6.

2 Hole Bypassing Routing (HBR)

We assume in this paper that each node knows the positions of itself and its 1-hop neighbors, and each routing hole H is a closed region bounded by a simple polygon ∂H, where there is a node at each vertex, and two adjacent boundary nodes are within the transmission range of each other. Notice that, routing holes are not necessarily disjoint from each other and may share boundary nodes or edges. In the continuous domain where the network density is so high that we can assume that there is a node at every point in the target field, it is well known from motion planning [8] that if every node s knows the complete boundaries of all routing holes, s can build a complete shortest-path roadmap locally by viewing routing holes as polygonal obstacles, and then for a given destination t, s can find an optimal path to t. Furthermore, an optimal path is composed of line segments connecting convex boundary vertices defined as follows.

Definition 2.1. Convex vertex, concave vertex: *A convex (resp. concave) vertex of a polygon P is a vertex of P for which the interior angle is less (resp. greater) than π.*

The main problem of applying this approach to a sensor network directly is its high message overhead due to the flooding of the complete hole boundaries to the entire network and the high storage overhead at each node to maintain a complete shortest-path map. The main idea of our approach is to approximate each routing hole H with a *core*, a simple polygon H_c enclosed by H that satisfies special requirements defined precisely in Section 3.2. One of these requirements relevant now is that every convex vertex of H_c must be a boundary node of H.

Cores are then advertised so that each node can build a shortest-path roadmap locally by viewing the cores it knows as obstacles. Since each core is contained in a routing hole, cores do not intersect with each other except possibly at boundaries. Furthermore, a path computed is composed of segments ending at the convex vertices of cores, which are boundary nodes by definition. Therefore, by substituting every such segment with a subpath implemented by either greedy forwarding or hole traversing, a realistic path can be constructed. We will show that such a hole approximation and advertising approach can significantly reduce the control overhead while still ensuring a desired route-stretch.

In this section, we present the main idea of the routing protocol, and assume that all the routing holes have been discovered, approximated, and advertised. These mechanisms will be presented in Section 3.

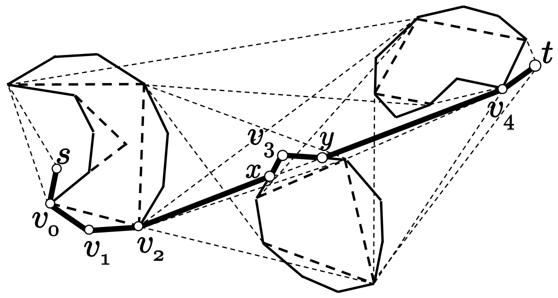


Figure 2. A shortest-path roadmap and a path from s to t computed by HBR in the continuous domain. The three polygons are routing holes and the dashed polygons are their cores. The dotted segments are the bitangent edges of the map. The path computed using cores is (s, v_0, v_2, v_4, t). The real path is highlighted. x and y are the intersections of line v_2v_4 with the boundary of the hole in the middle.

2.1 Building Roadmaps

Once a node s learns about a set of cores, it builds a shortest-path roadmap locally, which is defined as follows [8]. See Figure 2 for reference.

Definition 2.2. Shortest-path roadmap: *A shortest-path roadmap at node s consists of the set of the convex vertices V_s of the cores that s knows and the set of edges E_s. For any two vertices $a, b \in V_s$, $(a, b) \in E_s$ if a and b are visible to each other, and either (a, b) is an edge of a core or the line* *going through a and b is a bitangent line, that is, a tangent line at a and at b with respect to the cores they belong to.*

The shortest-path roadmap at node s can be built in $O(|V|^2 \log |V|)$ time, where V is the set of core vertices that s knows [8].

2.2 Routing Protocol

Consider an arbitrary source-destination pair (s, t). Algorithm 1 shows how a packet p is forwarded from s to t. See Figure 2 for reference.

Algorithm 1 Hole bypassing routing

```
1. Initialize: $v \leftarrow s$.
2. $v$ makes a routing plan locally to select the next convex vertex $v'$ on the shortest path from $v$ to $t$.
3. $p$ is forwarded towards $v'$ as follows.
    (a) If $v$ and $v'$ are consecutive vertices of a core $H_c$ for some hole $H$, $\partial H$ is traversed to reach $v'$.
    (b) Otherwise, $p$ is forwarded greedily towards $v'$. If a hole $H$ is touched before reaching $v'$, $\partial H$ is traversed to reach the node that is closest to the intersection of $\partial H$ and $\overline{vv'}$ where $v'$ is visible, then greedy forwarding continues.
4. Let $v \leftarrow v'$ and repeat steps 2 and 3 until $t$ is reached.
```

At step 2, the routing plan at node v is made as follows. First, the shortest-path roadmap is extended by connecting v and t to all the visible roadmap vertices. Second, Dijkstra's algorithm is applied to find the first convex vertex on a shortest path from v to t.

At step 3(b), there are two possible directions when traversing ∂H is required. Suppose p touches ∂H at node x. x will make a routing plan towards v' using only the core of H to determine which direction to go. If xv' is disjoint from the core, the packet will follow the direction so that the real path is also disjoint from the core. For instance, in Figure 2, a packet forwarded by x towards y traverses the hole boundary in clockwise order, which is planned by x.

To assist the routing protocol, each data packet header carries the locations and node IDs of the source, the destination, and the next route planning node(a convex vertex).

In the above protocol, a new routing plan is made at each convex vertex on a path from s to t. This can be optimized as follows. First, when a planned path includes a sequence of contiguous convex vertices on the same hole, routing plans could be made only at the first and last convex vertices, and let the packet header carry the last one so that the intermediate convex vertices can simply forward the packet towards the last vertex. Second, a packet header could carry part of the routing plan (a sequence of convex vertices) made at the source or an intermediate node. Third, a node could cache the routing plans made for certain destinations. In this case, a protocol that handles outdated plans due to the changes of network topology is needed, which is left for future work.

3 Hole Approximation and Advertisement

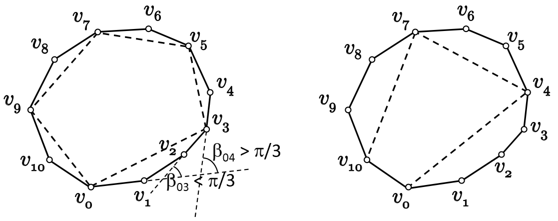
3.1 Hole Discovery

We apply the approach proposed in [4] to discover routing holes , which involves a local rule that finds nodes where packets may get stuck in greedy forwarding and an algorithm that discovers routing holes with stuck nodes on their boundaries. The nodes on the same hole boundary then cooperate to elect the node with the smallest node ID as the hole coordinator. That ID is also used as the hole ID. Each boundary node then sends a message containing its position to the coordinator. The coordinator then approximates the hole boundary as discussed below.

3.2 Hole Approximation

In this section, we discuss how to approximate a routing hole by its core. Our approach can be viewed as a polygon simplification approach since each routing hole is bounded by a polygon. Although many work on polygon simplification have been done in computational geometry, they can not be directly applied to our scenario because the approximation has to satisfy the following two properties to simplify the routing protocol and bound route-stretch: (1) A core is contained in the corresponding routing hole, and its convex vertices must be the boundary nodes of the hole. (2) A core should be a good approximation of the hole so that the route-stretch of HBR can be bounded.

We will first consider how to approximate a routing hole bounded by a convex polygon, and then extend the approach to a general polygon. In both cases, we assume a routing hole has at least 4 boundary nodes since a polygon with less than 4 vertices can not be simplified any further.



$\alpha = \pi/3$ $\alpha = \pi/2$

Figure 3. The approximation of convex holes.

3.2.1 Holes with Convex Boundary

Consider a routing hole H bounded by a convex polygon ∂H with vertices $v_0, v_1, v_2, ..., v_{n-1} (n \geq 4)$ sorted in counterclockwise order. The idea is to divide ∂H into chains and replace each chain with a line segment. The approximation has only one parameter α, which determines how vertices are grouped. Let $\beta_{kk'} \in [0, \pi), k' \geq k + 1$ [1] denote the

---

[1] In this section, all the arithmetic operations on subscripts are modulo n operations, and $v_n := v_0$.

angle from line $v_k v_{k+1}$ to line $v_{k'-1} v_{k'}$ (see Figure 3). Algorithm 2 is named as α-approximation, which begins at a vertex v_0 and traverses the hole boundary in counterclockwise order. See Figure 3 for reference.

Algorithm 2 α-approximation of a convex hole H

```
$H_c \leftarrow \{v_0\}, k \leftarrow 0$
repeat
    find the largest $k'$ s.t. $k < k' \leq n$ and $\beta_{kk'} \leq \alpha$
    $H_c \leftarrow H_c \cup \{v_{k'}\}, k \leftarrow k'$
until $k = n$
```

The time complexity of the algorithm is $O(n)$. The following proposition states that the size of a core in terms of the number of vertices on it is independent of the size of the hole, and is only determined by α, which directly follows from the fact that the sum of the exterior angles of any convex polygon is 2π.

Proposition 3.1. *The size of the core of a convex hole obtained by α-approximation is bounded by $\lfloor 2\pi/\alpha \rfloor$.*

3.2.2 Holes with non-Convex Boundary

Consider a routing hole H bounded by a non-convex polygon ∂H with vertices $v_0, v_1, v_2, ..., v_{n-1} (n \geq 4)$ sorted in counterclockwise order. ∂H can be divided into multiple disjoint interleaving convex and concave chains defined as:

Definition 3.1. Convex chain, concave chain: *A sequence of vertices $C_{ij} = (v_i, v_{i+1}, ..., v_j)$ (without repetition) of a polygon P is a convex (resp. concave) chain if every vertex in the chain is a convex (resp. concave) vertex, and the chain can not be extended further to maintain the property.*

The approximation of a non-convex ∂H works as follows.

1. Apply α-approximation to every convex chain of ∂H starting at one end of the chain, with the additional requirement that the line segments used to replace the chain must lie in the core computed so far. Name the resulting polygon P.
2. Simplify every concave chain of P (discussed below) to get H_c.

There are two things to be noted. First, a line segment $\overline{v_i v_j}$ lies in a polygon P iff (1) $\overline{v_i v_j}$ is disjoint from any edges of P except possibly at v_i and v_j and (2) for any point x in the segment other than v_i and v_j, x lies in P. Both conditions can be checked in $O(n)$ time. Therefore, the running time of the above procedure is $O(n^2)$. Second, step 1 and step 2 may be applied alternately more than once to reduce the size of H_c.

To simplify concave chains, we recall that the concave vertices of a core are not part of the shortest-path map, and therefore a concave chain could be simplified without considering the error criterion – the worst case route-stretch in our scenario. Consider a concave chain C_{ij}. Let $w_{kk'}, k' \geq k+1$ denote the intersection of line $v_k v_{k+1}$ and line $v_{k'-1} v_{k'}$ (see Figure 4). Algorithm 3 embodies the similar idea of α-approximation and outputs C, the simplification of C_{ij}. The running time of the algorithm is $O(|C_{ij}|n)$ and the total time complexity of the approximation of a routing hole with non-convex boundary of size n is therefore $O(n^2)$.

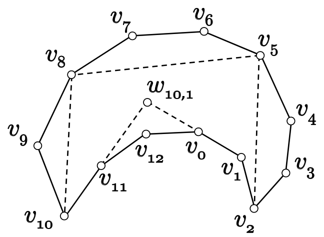


Figure 4. The approximation of a routing hole with one convex chain and one concave chain. $\alpha = \pi/2$ for the convex chain. The hole is simplified to the polygon with vertices $v_1, v_2, v_5, v_8, v_{10}, w_{10,1}$.

Algorithm 3 Simplification of a concave chain $(v_i, ..., v_j)$

```
$C \leftarrow \{v_{i-1}\}, k \leftarrow i - 1$
repeat
    find the largest $k'$ s.t. $k < k' \leq j + 1$ and both segments $\overline{v_k w_{kk'}}$ and $\overline{w_{kk'} v_{k'}}$ lie in the polygon computed so far
    $C \leftarrow C \cup \{w_{kk'}, v_{k'}\}, k \leftarrow k'$
until $k = j + 1$
```

3.3 Controlled Hole Advertisement

After H_c is computed, the coordinator of H computes C_H, the *minimum bounding circle* of H, which can be done in linear time [9]. Suppose C_H is centered at x. The coordinator then sends a message containing all the vertices of H_c sorted in counterclockwise order. The message is flooded to all the nodes within a big circle C'_H centered at x with radius $R_H = \lambda p_H$, where $\lambda \geq 1$ is a constant and p_H is the perimeter of H, which equals to the size of H in the discrete domain. The impact of different choices of λ on the worst case route-stretch will be discussed in Section 4 . To reach obstructed regions within C'_H , the flooding messages also hug the boundary of holes intersecting with C'_H.

4 Worst Case Stretch in Continuous Domain

In this section, we analyze the worst case route-stretch of HBR in the continuous domain, assuming routing holes are approximated using Algorithms 2 and 3. The proofs of the theorems stated in this section can be found in [13]. We first assume all the cores are advertised to the *entire* network, and show that the worst case route-stretch is only determined by α. Consider an arbitrary source-destination pair (s, t). Let f denote the path from s to t found by HBR with α-approximation and f_{opt} denote the shortest path. We have the following theorem.

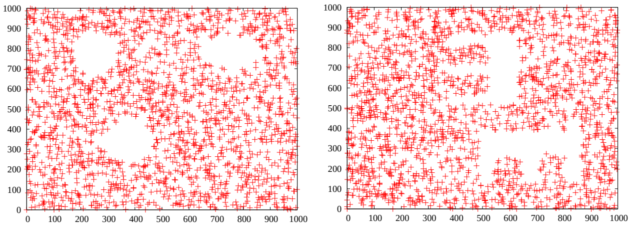
Theorem 4.1. $|f| \leq |f_{opt}|/cos(\alpha/2)$.

We next consider the impact of controlled advertisement on route-stretch. For an arbitrary source-destination pair (s, t), we still use f to denote the path from s to t found by HBR, with both α-approximation and controlled advertisement applied, and let h denote the number of routing holes whose core intersects f' but is not used in route planning. We have the following theorem.

Theorem 4.2. $|f| \leq |f_{opt}|(1/cos(\alpha/2) + h/(\lambda - 1))$.

5 Evaluation

Simulation settings: We evaluated the HBR protocol using *ns2*. 2000 nodes are randomly deployed in a $1000m \times 1000m$ area. The transmission range of each node is 40m and the average node degree is around 15. Besides the small holes formed randomly in a network, two types of big holes are artificially introduced: elliptical holes and rectangular holes with concave regions, as illustrated in Fig.5. The semimajor and semiminor axes of an elliptical hole are uniformly distributed on $[a/2, a]$, and the length and width of a rectangular hole are uniformly distributed on $[3b/4, b]$ and $[9b/16, 3b/4]$, respectively, where a and b are parameters. For each type of holes, we first fixed the number of holes to be 2 and varied the size (a or b) of holes. Then we fixed the size of all holes to be 300, and varied the number of holes. The positions of holes are randomly selected. Given the number, size, and shape of holes, 5 network scenarios are generated randomly, and the routes between all pairs of nodes are computed in each scenario. The results are averaged over all these scenarios. In all simulations, α is set to $\pi/2$, λ is set so that the cores of the artificially introduced big holes are advertised to the entire network.



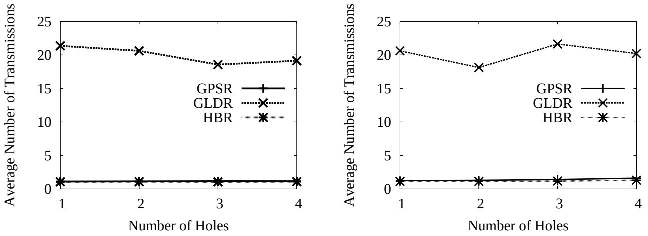
(a) Elliptic holes (b) Rectangular holes

Figure 5. Two typical simulation scenarios

HBR is compared with GPSR [6] and GLDR [10]. GLDR is a virtual coordinates based routing protocol including a landmark selection algorithm and a greedy routing protocol based on landmarks, which ensures delivery in the continuous domain. The IDs of 8 historical extreme nodes are maintained in the packet header to detect loops in the discrete domain. When local minimum happens or a loop is detected, L_1-norm and then L_∞-norm are tried. If the destination is still not reachable, scoped flooding is performed. In our simulation, 25 landmarks are used on average. The routing success rate without flooding is about 95%. The average hop distance from the node where flooding is issued to the destination is about 3 hops.

Route-stretch: Figure 6 shows the average route-stretches for the three protocols evaluated. We can see that in all the cases, HBR performs much better than GPSR. For elliptic holes, HBR performs better than both GPSR and GLDR. For rectangular holes, GLDR has a relatively stable stretch and performs better than HBR. This is because the performance of GLDR mainly depends on the distribution of landmarks and network density instead of the shapes of holes. However, GLDR achieves the low stretch by paying a relatively high overhead as discussed below. Furthermore, although Figure 6(b) shows that HBR can lead to an increasing stretch with increasing hole size, the stretch is bounded by $1/cos(\alpha/2) = \sqrt{2}$ since $\alpha = \pi/2$.

Message overhead: Figure 7 shows the normalized number of packet transmissions, which is defined as the number of transmissions for routing a packet using a specific routing protocol divided by that in shortest path routing, averaged over all source-destination pairs. For both GPSR and HBR, the values are the same as their stretches. For GLDR, the value is much higher because scoped flooding is used when destination is not reachable by greedy forwarding. Although this does not happen frequently, the impact is huge, especially for a network with high density.



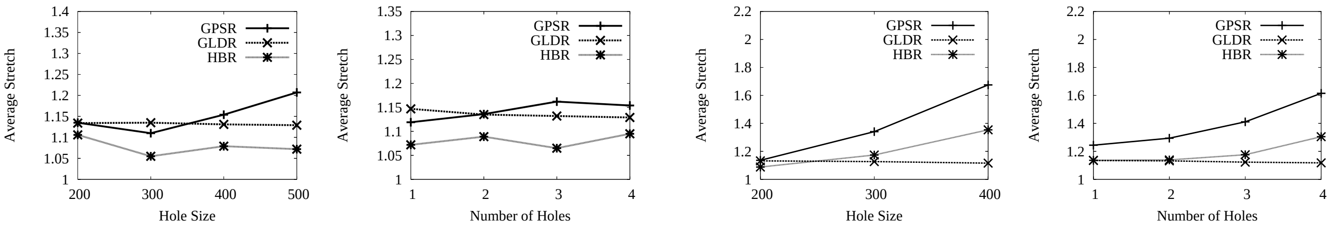
(a) Elliptic holes (b) Rectangular holes

Figure 7. Normalized number of packet transmissions in routing procedure

Besides the high transmission overhead, GLDR also suffers from high overhead for each data packet because a big packet header is used to save the distances to the 10 addressing landmarks of the destination, and IDs of the last 8 extreme nodes visited. In contrast, the packet header size of HBR can be much smaller as discussed in Section 2.

6 Related Work

Various extensions of geographic routing protocols have been proposed to guarantee delivery in the presence of holes. Our approach is different from previous works in two key aspects. First, we view routing holes as obstacles and propagate them *proactively* in a *controlled* way. In contrast, most geographic routing protocols treat holes in a reactive way and only try to bypass them when greedy forwarding



(a) Elliptic holes (b) Rectangular holes

Figure 6. Average route-stretches

fails [2, 6, 7], leading to high stretch. Some recent works have dealt with holes explicitly so that packets can bypass holes in advance. However, none of them ensure a bound on route-stretch and some of them [5, 14] only support "many to one" communications. Second, all the above works focus on dealing with local minima or non-convex regions, and apply greedy forwarding whenever possible. However, we point out that with the presence of holes, even if there is no local minima in the network, greedy forwarding can still be suboptimal [13]. Our approach can by used to improve the greedy forwarding phase of existing protocols.

7 Conclusion

Emerging sensor networking scenarios require support for routing between arbitrary pairs of nodes in the network. We propose a distributed shortest path roadmap based routing paradigm that is leveraged to achieve bounded route-stretch with low-overhead for storing and forwarding control information at each node. We demonstrate its application in the context of managing information on routing holes in the most critical regions around the holes to guarantee bounded stretch.

8 Acknowledgement

This material is based upon work supported by the National Science Foundation under Grants CNS-0546630 (CAREER Award), CNS-0721434, CNS-0721817 and CNS-0403342. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] N. Arad and Y. Shavitt. Minimizing Recovery State in Geographic Ad-hoc Routing. In *Proc. of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 13–24, 2006.

[2] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with Guaranteed Delivery in Ad Hoc Wireless Networks. In *In Proc. of the 3rd ACM International Workshop on discrete Algorithms and Methods for Mobile Computing and Communications*, pages 48–55, August 1999.

[3] M. Demirbas, A. Arora, and M. Gouda. A Pursuer-Evader Game for Sensor Networks. In *Sixth Symposium on Self-Stabilizing Systems(SSS)*, pages 1–16, 2003.

[4] Q. Fang, J. Gao, and L. J. Guibas. Locating and Bypassing Routing Holes in Sensor Networks. In *Proc. of INFOCOM*, volume 4, pages 2458–2468, Mar. 2004.

[5] W. Jia, T. Wang, G. Wang, and M. Guo. Hole Avoiding in Advance Routing in Wireless Sensor Networks. In *Proc. of WCNC*, pages 3519–3523, 2007.

[6] B. Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *Proc. of ACM MOBICOM*, pages 243–254, August 2000.

[7] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric Ad-Hoc Routing: Of Theory and Practice. In *Twenty-Second ACM Symposium on Principles of Distributed Computing*, pages 63–72, July 2003.

[8] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[9] N. Megiddo. Linear-Time Algorithms For Linear Programming in R^3 and Related Problems. *SIAM Journal on Computing*, 12(4):759C776, 1983.

[10] A. Nguyen, N. Milosavljevic, Q. Fang, J. Gao, and L. J. Guibas. Landmark Selection and Greedy Landmark-Descent Routing for Sensor Networks. In *Proc. of INFOCOM*, May 2007.

[11] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. Data-Centric Storage in Sensornets with GHT, a Geographic Hash Table. volume 8, pages 427–442, August 2003.

[12] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang. A Two-Tier Data Dissemination Model for Large-scale Wireless Sensor Networks. In *ACM/IEEE Mobicom*, pages 148–159, September 2002.

[13] Z. Zheng, K.-W. Fan, P. Sinha, and Y. Wang. Distributed Roadmap Aided Routing in Sensor Networks. Technical Report OSU-CISRC-7/08-TR41, ftp://ftp.cse.ohio-state.edu/pub/tech-report/2008/TR41.pdf.

[14] L. Zou, M. Lu, and Z. Xiong. A Distributed Algorithm for the Dead End Problem of Location Based Routing in Sensor Networks. In *IEEE Transactions on Vehicular Technology*, volume 54, pages 1509–1522, July 2005.