

# Geographic Routing Made Practical

Young-Jin Kim<sup>†</sup>    Ramesh Govindan<sup>†</sup>  
<sup>†</sup>*University of Southern California*  
*Los Angeles, CA 90089*  
{youngjki, ramesh}@usc.edu

Brad Karp\*  
<sup>\*</sup>*Intel Research/CMU*  
*Pittsburgh, PA 15213*  
bkarp@cs.cmu.edu

Scott Shenker<sup>‡</sup>  
<sup>‡</sup>*UCB/ICSI*  
*Berkeley, CA 94704*  
shenker@icsi.berkeley.edu

## Abstract

Geographic routing has been widely hailed as the most promising approach to generally scalable wireless routing. However, the correctness of all currently proposed geographic routing algorithms relies on idealized assumptions about radios and their resulting connectivity graphs. We use testbed measurements to show that these idealized assumptions are grossly violated by real radios, and that these violations cause persistent failures in geographic routing, even on static topologies. Having identified this problem, we then fix it by proposing the Cross-Link Detection Protocol (CLDP), which enables provably correct geographic routing on *arbitrary* connectivity graphs. We confirm in simulation and further testbed measurements that CLDP is not only correct but practical: it incurs low overhead, exhibits low path stretch, always succeeds in real, static wireless networks, and converges quickly after topology changes.

## 1 Introduction

There is a very broad literature on geographic routing algorithms, particularly on the sub-class that uses face routing on a planar subgraph [2, 7, 13, 17, 18, 24]. These algorithms are attractive for wireless ad hoc networks because they have been shown to scale better than other alternatives: they require per-node state independent of network size, dependent only on network density. More recently, geographic routing algorithms have been proposed for use as a routing primitive for static sensor networks, as building blocks for data storage and flexible query processing in sensor networks [20, 23], and even as a fallback routing mechanism for reduced state routing in the Internet [9].

Despite research activity on geographic routing spanning half a decade, we know of no work in which researchers have *implemented* and *deployed* geographic routing protocols in realistic environments. Using our implementation of the GPSR geographic routing algorithm [13]—which we believe to be the first of its kind—

we first show that GPSR incurs permanent packet delivery failures between node pairs on two different sensor network testbeds where we had no control over node placement. To wit, GPSR leaves over 30% of node pairs permanently disconnected in one testbed experiment, and over 10% disconnected in another. The significant incidence of these delivery failures and their permanent nature suggest that known geographic routing techniques are impractical for use in real deployments.

GPSR is built upon graph planarization algorithms that are amenable to distributed implementation [2, 13]. These planarization algorithms rely purely on neighbor location information to determine whether or not links to neighbors belong in the planarized subgraph. When greedy forwarding is impossible, GPSR delivers a packet by successively traversing the faces of the planar subgraph cut by the line between the packet's source and destination. A body of subsequent work (including GOAFR+ [17] and its many variants) has extended this face routing technique to offer shorter worst-case paths than GPSR. A common assumption made by the planarization algorithms used by all these geographic routing protocols is that connectivity between nodes can be described by *unit graphs*. In such graphs, a node is always connected to all nodes within its fixed, "nominal" radio range, and never connected to nodes outside this range.

We show that our implementation of GPSR incurs permanent delivery failures precisely because real radios routinely violate the unit graph assumption. Such violations can cause three kinds of pathologies in the planarization process: a link in the planar subgraph is removed when it should not be (partitioned planar subgraph); the nodes at the two ends of a link disagree on whether or not the link belongs in the planar graph (unidirectional links); or a pair of crossed links remain in the supposedly planar subgraph (crossing links). These pathologies, in turn, can result in persistent routing failures in the network, where geographic routing fails to

find a path for at least one source-destination pair. A previously proposed “fix” to these planarization techniques, the mutual-witness procedure [11, 12, 24], fails to eliminate many instances of routing failure on our testbeds.

We remedy this problem by proposing a distributed Cross-Link Detection Protocol (CLDP) that, given an arbitrary connected graph, produces a subgraph on which face traversal cannot cause a routing failure, regardless of radio irregularities and localization errors. In CLDP, each node probes the faces on which each of its links sits to determine if there exists a crossing link. Crossing links are eliminated only when doing so would not disconnect the resulting subgraph. This algorithm is *qualitatively* different from the planarization algorithms used by earlier face routing protocols, in both its approach and its correctness. The unmodified GPSR algorithm conducts perimeter-mode forwarding using the subgraph produced by CLDP.<sup>1</sup> CLDP retains geographic routing’s desirable scaling properties. Moreover, we have proven that CLDP prevents routing failures in an arbitrary connected graph.<sup>2</sup>

Finally, we present measurements from simulations and experiments on two different wireless sensor network testbeds that validate CLDP’s correctness, and show that CLDP incurs moderate overhead, converges quickly, and picks low-loss paths. Because CLDP renders geographic routing correct on real radio networks, we believe it represents the first generally scalable and *practical* approach for any-to-any routing in large-scale wireless settings.

## 2 Preliminaries and Related Work

We now review prior work in geographic routing protocols and describe the essentials of the workings of geographic routing that provide the context for our work.

There is a very broad literature on geographic routing: from initial sketches suggesting routing using position information [4, 15]; to the first detailed proposals, including GFG [2], GPSR [13], and the GOAFR+ family of algorithms [17]; to refinements of these proposals for efficiency [7], robustness under real network conditions [18, 24], and even routing geographically when node location information is unavailable [21, 22].

We now describe the shared characteristics of the GFG, GPSR, and GOAFR+ algorithms, and hereafter refer to this family of algorithms simply as geographic routing.<sup>3</sup>

Geographic routing schemes use *greedy routing* where possible. In greedy routing, packets are stamped with the positions of their destinations; all nodes know their own positions; and a node forwards a packet to its neighbor that is geographically closest to the destination, *so long as that neighbor is closer to the destination*. *Local maxima* may exist where no neighbor is closer to the destina-

tion. In such cases, greedy forwarding fails, and another strategy must be used to continue making progress toward the destination. In particular, the packet must only find its way to a node closer to the destination than the local maximum; at that point, greedy routing may once again make progress.

In the case where a network graph has no crossing edges<sup>4</sup>—that is, the graph is *planar*—geographic routing schemes recover similarly by *face routing*. Note that a planar graph consists of *faces*, enclosed polygonal regions bounded by edges. Geographic routing uses two primitives to traverse planar graphs: the *right-hand rule*, and *face changes*. The right-hand rule tours a face endlessly in a cycle, and can thus be used to walk a face. Figure 1 shows an example of the rule, which dictates that upon receiving a packet on a link, the receiving node forwards that packet on the first link it finds after sweeping counter-clockwise about itself from the ingress link.

Consider the planar graph in Figure 2, in which the source node  $S$  and destination node  $D$  are indicated. Observe that the line segment  $\overline{SD}$  must cut a series of faces in the planar graph; these faces are numbered and bordered in bold. Geographic routing algorithms exploit this property by successively walking the faces cut by this line. That is, they use the right-hand rule to tour a face. While walking a face, upon encountering an edge that crosses the line segment  $\overline{SD}$  at a point closer to  $D$  than the point at which the current face was entered, geographic routing algorithms perform a *face change*: they begin walking the bordering face that is next along the line segment  $\overline{SD}$ .<sup>5</sup> The numbering of faces in Figure 2 shows the order in which faces are traversed from  $S$  to  $D$  on that planar graph. Should a face be toured in its entirety without discovering an edge that crosses line segment  $\overline{SD}$  at a point closer to  $D$  than the point at which the current face was entered, face routing fails. On a planar graph, such a loop on a face only occurs when the destination is disconnected.

Note that if the graph is not planar, face routing may fail. Figure 3 shows an example graph on which this pathology occurs. In this example,  $D$  is located physically in the interior of a face, but is only connected to the rest of the network graph by an edge that crosses this enclosing face. Face routing walks successive faces cut by the line from  $S$  to  $D$ , until it reaches the face enclosing  $D$ , whose first edge crosses line segment  $\overline{SD}$  at point  $p$ . The right-hand rule then tours this face in its entirety, but fails to find an edge that crosses line segment  $\overline{SD}$  at a point closer to  $D$  than  $p$ . Thus, face routing fails.

Wireless networks’ connectivity graphs typically contain many crossing edges. A method for obtaining a planar subgraph of a wireless network graph is thus needed; greedy routing operates on the full network graph, but to work correctly, face routing must operate on a planar

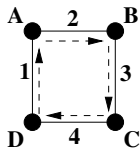


Figure 1: Right-hand rule.  $A$  sweeps counterclockwise from link 1 to find link 2, forwards to  $B$ , &c.

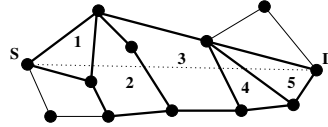


Figure 2: The faces progressively closer from  $S$  to  $D$  along line segment  $\overline{SD}$ , numbered in the order visited. Faces cut by  $\overline{SD}$  are bordered in bold.

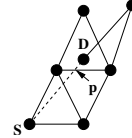


Figure 3: Example of face routing failure on non-planar graphs. There is no point closer to  $D$  than  $p$  on the face enclosing  $D$ .

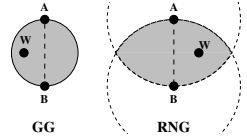


Figure 4: Definitions of the GG and RNG. A witness must fall within the shaded circle (GG) or lune (RNG) for edge  $(A, B)$  to be eliminated in the planar graph.

subgraph of the full network graph. What is required is a *planarization* technique that is simply implementable with an asynchronous distributed algorithm.

Geographic routing algorithms planarize graphs using two planar graph constructs that meet that requirement: the Relative Neighborhood Graph (RNG) [26] and the Gabriel Graph (GG) [5]. The RNG and GG give rules for how to connect vertices placed in a plane with edges based purely on the positions of each vertex's single-hop neighbors. Both the RNG and GG provably yield a connected, planar graph so long as the connectivity between nodes obeys the *unit graph assumption*: for any two vertices  $A$  and  $B$ , those two vertices *must* be connected by an edge if they are less or equal to some threshold distance  $d$  apart, but *must not* be connected by an edge if they are greater than  $d$  apart. We shall refer to  $d$  as the *nominal radio range* in a wireless network; the notion is that all nodes have perfectly circular radio ranges of radius  $d$ , centered at their own positions.

The unit graph assumption is quite intuitive for wireless networks. The simplest ideal radio model is one where all transmitters radiate fixed transmission power perfectly omnidirectionally; receivers can discern all transmissions properly when they are received with above some threshold signal-to-noise ratio; and radio transmissions propagate in free space, such that their energy dissipates as the square of distance. Under that idealized model, there indeed exists a nominal radio range.

We briefly state the definitions of the GG and RNG, as we shall refer to them repeatedly in Section 3. The planarization process runs on a *full graph*, which includes *all* links in the radio network, and produces a *planar subgraph* of the full graph. We assume that each node in the network knows its single-hop neighbors' positions; such neighbor information is trivially obtained if each node periodically transmits broadcast packets containing its own position. Consider an edge in the full graph between

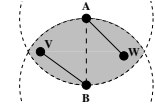


Figure 5: The RNG partitions a non-unit graph; edge  $(A, B)$  is eliminated.

two nodes  $A$  and  $B$ . Both  $A$  and  $B$  must decide whether to keep the edge between them in the planar graph, or eliminate it in the planar graph. Without loss of generality, consider node  $A$ . Both for the GG and RNG, node  $A$  searches its single-hop neighbor list for any *witness* node  $W$  that lies within a particular geometric region. If one or more witnesses are found, the edge  $(A, B)$  is eliminated in the planar graph. If no witnesses are found, the edge  $(A, B)$  is kept in the planar graph. For the GG, the region where a witness must exist to eliminate the edge is the circle whose diameter is line segment  $\overline{AB}$ . For the RNG, this region is the *lune* defined by the intersection of the two circles centered at  $A$  and  $B$ , each with radius  $|\overline{AB}|$ . We show these two regions in Figure 4.

Under the unit graph assumption, it is known that for a clustering of points in the plane, the set of edges in the Euclidean minimum spanning tree over those points is a subset of the set of edges in the RNG [26]. The edges in the RNG are in turn a subset of those in the GG; the intuition for this relationship lies in the relative sizes of the lune and circle regions. Finally, the set of edges in the GG is a subset of that in the Delaunay triangulation over the set of points [25]. These relationships dictate that the GG and RNG are both connected (so eliminating crossing edges cannot disconnect the network!) and planar, as desired. Note that if the network graph *violates* the unit graph assumption, the RNG and GG can produce a *partitioned* planarized graph [11], one that contains unidirectional links, and even one that is not planar. An example of a partitioning for the RNG appears in Figure 5. Here, there is no link between  $A$  and  $V$ , and none between  $B$  and  $W$ , though these links are shorter than the nominal radio range. Nodes  $A$  and  $B$  see witnesses  $W$  and  $V$ , respectively, though neither witness provides transitive connectivity. Both  $A$  and  $B$  conclude they should remove edge  $(A, B)$  in the planarized graph, and a partition results. Similar cases are possible in the GG.

We observe that whether radio graphs conform to the unit-graph assumption is a question of great importance, as partitioning the planarized graph used in face routing will cause routing failures. In the next section, we explore in detail the many reasons real radios violate the unit graph assumption, and give detailed examples of the pathologies these violations create in the GG and RNG.

Recently, Kuhn *et al.* have investigated relaxing the unit-graph assumption to improve the robustness of the GG planarization [18]. In the *Quasi-Unit Disk Graph* they propose, the nominal radio range is normalized to 1. Links *may not* exist between nodes greater than distance 1 apart, and links *must* exist between nodes less than a parameter  $d$  apart. For nodes between  $d$  and 1 distance apart, links may or may not exist; it's in this region where Quasi-Unit Disk Graphs are a more general class than unit graphs. Kuhn *et al.* provide an algorithm for replacing "missing" links between  $d$  and 1 in length with *virtual* links, that are essentially tunnels through multiple existing links. They show that the GG planarization succeeds on this augmented graph without partitioning it. Their analysis shows that this technique is only scalable when  $d \geq 1/\sqrt{2}$ ; for lesser values of  $d$  (for which the unit-graph assumption is progressively relaxed further) virtual links may be comprised of increasingly long paths of physical hops.

### 3 Pathologies in Real Deployments

In the previous section, we demonstrated two situations where GPSR's perimeter-mode routing may fail: when crossing links remain after planarization is applied, and when planarization partitions the network graph. It is natural to ask how prevalent these pathologies are in real deployments of GPSR: are they so rare as to be of purely theoretical interest, or do they significantly negatively affect reachability between pairs of nodes? We confirm in this section that the latter is the case, using measurements taken on real wireless networks.

#### GPSR Implementation and Testbeds

We implemented GPSR for Mica-2 sensor motes. Our full-fledged nesC [8] implementation includes the GG and the RNG planarization algorithms (chosen via a configuration parameter), as well as greedy- and perimeter-mode packet forwarding. It also includes a hop-by-hop retransmission mechanism, as the default Mica-2 MAC layer does not implement link-layer retransmission. Finally, our implementation rejects wireless links whose quality—measured by probing link loss rate—is below a configurable threshold. This mechanism incorporates hysteresis to avoid oscillatory behavior on links whose quality is near the threshold. Our complete implementation is over 4500 lines of nesC code.

We measured this implementation's behavior on two testbeds. Each consists of Mica-2 motes that span a floor

of an office building: one with 75 motes in Berkeley's Soda Hall, where offices are separated by floor-to-ceiling walls, and one with 51 motes at Intel Research Berkeley, where cubicles are separated by low dividers. We report only the Soda Hall results in the interest of brevity.<sup>6</sup>

Motes instrument most offices and some of the hallways in Soda Hall. Because the testbed is shared, we were able to use only a 50-node subset of it. As we could not control the placement of these devices, the GPSR failures discussed below are not contrived by careful node placement. We did, however, have one tool for controlling network topology: radio transmit power. At the default power setting on the testbed, all nodes were within two hops of each other. To generate an interesting multi-hop topology, we reduced the radio transmit power from 15 to 2. In the resulting topology, the average path length was around 5 hops, and the average node degree was 5.2. Note that controlling transmit power is roughly equivalent to appropriately scaling the geographic dimensions of the testbed. Finally, we statically configured nodes with their locations.

#### Pathologies

Figure 6 depicts the full network topology on the 50-node Soda Hall testbed, as is used by GPSR's greedy-mode forwarding. Our GPSR implementation does not forward on links with packet loss rates in excess of 30%; those links are not shown in the figure. Many links cross one another, particularly in the dense region of the network toward the left. It is the job of GPSR's planarization to eliminate these crossing links, to produce a planar graph for use by GPSR's perimeter-mode forwarding.

We measure the fraction of all pairs of nodes on this network that can reach one another with GPSR routing. In these measurements, we iterate over all nodes in the network, allowing one node at a time to send traffic to each other node in the network. We send 10 packets, and retransmit at the link level. If one or more packets reach the destination, we count that directed pair of nodes as connected, and in this way, measure routing algorithm success rather than short-term packet loss characteristics. We find that only 68.2% of directed node pairs can communicate successfully in the testbed—a significant fraction of node pairs experience *permanent partition!*

To help elucidate the reasons for these routing failures, we present in Figure 7 the network subgraph that results after our GPSR implementation distributedly applies the GG planarization to the full topology. There are three classes of pathology present in this network subgraph:

**Network partitions:** While the full network is connected, there are two connected components in Figure 7; the majority of the network comprises one connected component, and the nodes at the lower left of the figure the other. Such cases arise in situations such as those

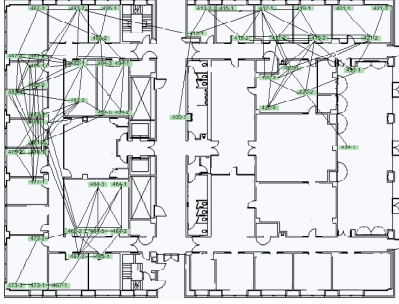


Figure 6: 50-node testbed. Links with packet loss rates over 30% are not shown.

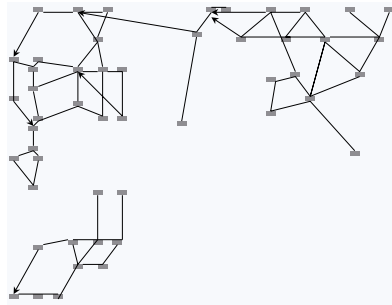


Figure 7: GPSR's GG subgraph on the 50-node testbed.

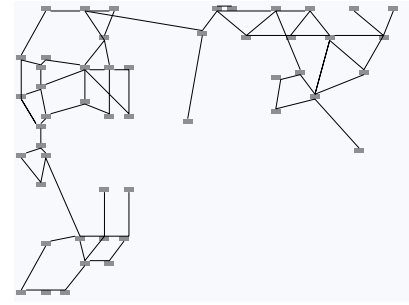


Figure 8: GPSR's GG/MW subgraph on the 50-node testbed.

previously described in Figure 5.

**Asymmetric links:** Links denoted with an arrow exist in the planar subgraph *only in the direction indicated*. Such links may give rise to unidirectional partitions in the planar subgraph, where an asymmetric link represents the only connectivity between two connected components. The GG and RNG planarizations produce asymmetric links in cases similar to that depicted in Figure 5; consider the case where  $W$  is not present in the graph. On that topology,  $A \rightarrow B$  will remain, but  $B \rightarrow A$  will not.

**Crossing links:** There are a few instances of crossing links that remain in Figure 7. For example, consider the long horizontal link that spans the hallway, and crosses a far shorter link. The GG and RNG planarizations may produce such pathologies when there are highly irregular radio ranges, as is the case here: the node at the right end of the long link cannot see any witnesses, and thus will not remove the long link; nor do the nodes at either end of the short, vertical link see any witnesses.

Radio range irregularities, which may be exacerbated by elimination of high-loss links, thus cause significant routing failures for GPSR in a real deployment. We expect other variants of GPSR to behave similarly, since they all use planarization methods based on unit-disk graphs. For context, we note that several measurement studies [1, 6, 27] have documented non-ideal radio behavior; however, ours is the first to quantify their impact on existing geographic routing protocols.

We have also implemented and experimented with a previously proposed fix to the GG's and RNG's tendency to partition graphs when radio ranges are irregular. The fix in question is the *mutual witness* (MW) extension to GPSR [11, 12, 24]. When node  $A$  considers whether to keep link  $(A, B)$  from the full graph in the RNG or GG planar graph, mutual witness dictates that  $A$  only eliminate link  $(A, B)$  if there exists at least one witness in the RNG or GG region that is visible *both* to  $A$  and  $B$ . This fact may be directly verified with local communication: if all nodes broadcast their neighbor lists (only a single hop), then all nodes may verify whether a particular

neighbor shares a particular other neighbor. The intuition for this mutual witness is that it preserves connectivity: links are only eliminated in the planar graph if a transitive path through a witness is explicitly verified, rather than relying on the location of the witness to assure such a transitive path's existence. Unfortunately, MW suffers from another ill; on some non-unit graphs, it will *leave crossing links* in the graph produced by the RNG and GG. Indeed, in our experiments with MW, we observed this behavior: GPSR augmented with MW enables connectivity between only 87.8% node pairs in one experiment, leaving more than 10% of node pairs persistently disconnected. Figure 8 shows the subgraph the MW extension generates in this experiment; note the crossing edges that remain that give rise to routing failures.

In sum, these results suggest that current geographic routing protocols are impractical. Although we have demonstrated this only using relatively unsophisticated Mica-2 radios, we believe our conclusions hold for other kinds of wireless devices as well, since the failure of the unit-disk assumption as a result of obstacles or multipathing is fairly fundamental. We spend the rest of the paper discussing a qualitatively different and practicable approach to geographic routing. As an aside, we note that while many of the pathologies we describe above are caused by radio range irregularities, localization errors can also cause the same pathologies [14, 24]. We leave measurement of the effects of localization errors in testbed deployments to future work.

## 4 Cross-Link Detection Protocol

We have established that existing planarization techniques frequently cause face routing to fail on real wireless networks, where the unit-graph assumption is violated. We now proceed to describe the Cross-Link Detection Protocol (CLDP), a planarization technique that cannot cause face routing to fail on any connected graph. As such, CLDP is also robust to arbitrary localization errors [14]; we omit a detailed discussion herein for lack of space.

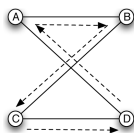


Figure 9: CLDP Probing using right-hand-rule, Case 1.

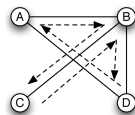


Figure 10: ..., Case 2.

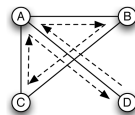


Figure 11: ..., Case 3.

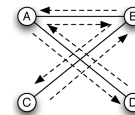


Figure 12: ..., Case 4.

#### 4.1 CLDP Overview

To describe the essential ideas behind CLDP, we first consider a static graph consisting of several nodes and links. We make no assumptions about the connectivity of this graph (*i.e.*, to which other nodes a given node may be connected). However, we assume that nodes in the graph are assigned positions in some 2-dimensional coordinate system, that the graph is connected, and that all the links are bi-directional. Initially, we also make several other idealized assumptions (like link-serialized execution of the protocol) to simplify exposition. We will return a bit later to consider the applicability of CLDP to more realistic wireless networks: in particular, we will consider the impact of node and link dynamics, and present a truly distributed, parallel realization of CLDP. We do not explicitly consider node mobility in our evaluation of CLDP, and leave that to future work.<sup>7</sup>

The high-level idea behind CLDP is simple: each node, in an entirely distributed fashion, *probes* each of its links to see if it is *crossed* (in a geographic sense) by one or more other links. A probe initially contains the locations of the endpoints of the link being probed, and traverses the graph using the right-hand rule. For example, in Figure 9, consider a probe originated by node *D* for the link  $(D, A)$ . It contains the geographic coordinates of *D* and *A*, and traverses the graph using the right-hand rule, as shown by the dashed arrows. When the probe is about to traverse the link  $(B, C)$ , node *B* “notices” that this traversal would cross  $(D, A)$ ; *B* records this fact in the probe so that when the probe returns to *D*, *D* notices a cross-link and “removes” either the  $(A, D)$  link or the  $(B, C)$  link (after a message exchange with *B*). By symmetry, the cross-links would have been detected by a probe of  $(A, D)$  originated by *A* or a probe of  $(B, C)$  originated either by *B* or *C*.

Care must be taken in dealing with degenerate crossings caused by exactly colinear links. A correct way to deal with these is to randomly, but slightly, perturb the reported location of each node to make the likelihood of such links vanishingly small. To simplify our discussion, we ignore such degeneracies in the rest of this paper.

We have described CLDP in a decentralized fashion, but to understand CLDP’s properties, it helps to envision the results of applying CLDP on all links of a static (*i.e.*, unchanging), arbitrary (*i.e.*, no specific connectivity assumptions), connected graph. Initially, assume that all the links in this graph are marked *routable*. Then, sup-

pose that each link is probed repeatedly and in some order with the constraint that only one probe is active at any given time (this is an idealization we relax later). As we have described above, a probe may cause a link to be removed. When we say CLDP “removes” a link, we mean that the link is marked *non-routable*. The set of routable links forms a *routable subgraph*. Furthermore, *all CLDP probes traverse the current snapshot of the routable subgraph*. Cross-links are not always marked non-routable; we show later how CLDP preserves cross-links the deletion of which would render the routable subgraph disconnected. This property implies that if the graph is connected to start with, CLDP does not partition it. The probing stops when subsequent probing of links would not cause any link to be marked non-routable.

We say a graph is *safe* if face routing between all pairs of nodes in the graph is guaranteed not to fail. As we discuss in Section 4.5 (and our simulations and experiments Section 5 bear this out as well), CLDP always produces a safe routable subgraph from any arbitrary input connected graph. This result is surprising for the following reason. It is easy to see that CLDP attempts to planarize the routable subgraph by removing cross-links, and face routing is known not to fail on a planarized graph. However, there is no *a priori* reason to believe (and no prior literature that suggests) that using the right-hand rule repeatedly to detect and remove cross-links will always result in a planarization (modulo the cross-links that need to be preserved to avoid disconnections) on an arbitrary graph.

As a practical matter, other forwarding strategies also work perfectly on the CLDP-derived routable subgraphs, such as GPSR’s combination of greedy- and perimeter-mode traversals [13], and GOAFR’s improvement that uses ellipses to bound face traversals when possible [17]. Note further that greedy forwarding uses the full graph (including links marked “non-routable” by CLDP); only face routing uses the CLDP-derived routable subgraph during recovery from local maxima.

In describing CLDP, we have made two simplifying assumptions: strictly sequential probing of links, and no node or link dynamics. In the following sub-sections we relax these two assumptions. Before doing so, however, we consider two other problems: how CLDP deals with cross-links whose removal would partition the routable subgraph, and how CLDP detects multiple cross-links.

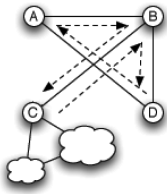


Figure 13: Effect of “clouds” on probes.

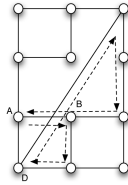


Figure 14: Routable sub-graph depends on probe ordering.

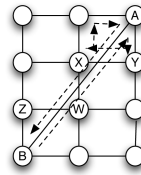


Figure 15: Multiple Cross-Links.

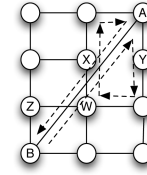


Figure 16: Repeated CLDP probes.

## 4.2 Partitions in the Routable Subgraph

In Figure 10, the removal of the  $(B,C)$  link would disconnect  $C$  from the rest of the network. Similarly, in Figure 11, the removal of the  $(A,D)$  link would disconnect  $D$ , and in Figure 12 the removal of either crossing link would partition the network.

To understand how CLDP deals with this situation, examine the paths taken by the CLDP probes originated by  $D$  in each of the figures (by symmetry, one can make similar observations about probes initiated by  $C$ ). Notice that in every case, when disconnecting a crossing link would partition the graph, the CLDP probe traverses that link *once in each direction*. In Figure 11, for example, the CLDP probe returns to  $D$  over the link on which it was sent (*i.e.*, the  $(A,D)$  link). Intuitively, it is clear why this should be so: there is no closed face over which the probe can return. In Figure 10, the CLDP probe originated by  $D$  traverses link  $(B,C)$  once in each direction. From this,  $B$  (or  $C$ ) can infer that removing link  $(B,C)$  would cause a partition.

While we have given the simplest possible examples, our observations generalize easily to arbitrary topologies attached to the “non-removable” link. For example, if in Figure 10, node  $C$  were connected to many “clouds” (Figure 13), the CLDP probe would return on the  $(B,C)$  link. Thus, when a CLDP probe traverses either the link being probed (or its cross-link) in both directions, CLDP infers that removal of that link could disconnect the routable subgraph, and does not remove the link. By this rule, CLDP would mark both the  $(A,D)$  and the  $(B,C)$  links in Figure 12 routable. We point out an important property of the routable subgraphs derived by applying CLDP—they may contain crossing links.

Thus, the correct rule for marking links non-routable can be stated as follows. Suppose any node  $N$  probes an attached link  $L$  and finds a cross-link  $L'$ :

- Case 1:** If both  $L$  and  $L'$  can be removed (*i.e.*, the CLDP probe traversed neither link twice), remove  $L$ .
- Case 2:** If  $L$  can be removed, but  $L'$  cannot, remove  $L$ .
- Case 3:** If  $L$  cannot be removed, but  $L'$  can, signal the appropriate nodes to remove  $L'$ .
- Case 4:** If neither link can be removed, do nothing.

Consider the application of this rule to the network in Figure 14, which illuminates an important property of

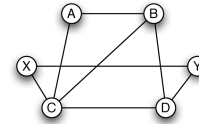


Figure 17: Probing a link may not detect a cross-link.

CLDP: that *different* routable sub-graphs may be generated by applying CLDP to the same graph, depending upon the order in which links are probed. For example, if  $(A,B)$  were probed first, then  $(C,D)$  would be removed, and vice versa.

## 4.3 Multiple Cross-Links

Thus far in our discussions, we have assumed that a link is crossed by at most one other link. But consider the situation depicted in Figure 15 where a long link  $(A,B)$  is crossed by three other links. In arbitrary graphs, of course, this situation will not be uncommon.

CLDP generalizes rather easily to this case. Repeatedly probing a link until no removable cross-links are found will keep the resulting routable sub-graph safe. Consider Figure 15 and assume that  $B$  probes link  $(A,B)$ . The first such probe will traverse the faces shown, detecting the cross-link  $(X,Y)$ , which will be removed. A second probe sent by  $B$  (Figure 16) will detect the  $(X,W)$  cross-link, resulting in the removal of that link (and so on). Our examples of multiple cross-links are a bit misleading, as they suggest that repeatedly probing a link will detect *all* cross-links. This is not, in general, true: probing *one* of a pair of cross-links is not guaranteed to find the crossing (intuitively, that link may be obscured by other, perhaps non-removable) links. The other link may also have to be probed (from both ends) before the cross-link is detected. Consider, for example, the topology in Figure 17. In this topology, CLDP probes from either end of the  $(B,C)$  link are confined to the adjoining triangles, and are unable to detect the  $(X,Y)$  link. The  $(B,C)$  cross-link is only detected after repeatedly probing the  $(X,Y)$  link.

## 4.4 Concurrent Probing

Thus far, we have assumed that CLDP probes are *serialized*. However, this kind of global serialization is unachievable without significant messaging cost in large networks. A design that permits nodes to probe links concurrently is clearly more desirable.

Unfortunately, concurrent probing can render the routing subgraph disconnected. Consider Figure 9 and assume that while  $D$  probes link  $(A,D)$ ,  $C$  concurrently probes link  $(B,C)$ . When each probe returns,  $C$  and  $D$  each detect a cross-link, and mark their directly attached links non-routable (assume that either link can be removed), leaving the routable subgraph disconnected. Such a race condition can be prevented using a simple *tie-break* rule that deterministically decides which cross-link should be deleted. However, the tie-break rule does not guarantee correctness in the general case.

A simple approach would be to *lock* a link while it is being probed. CLDP drops probes that encounter a locked link in either direction, and retries them later. This approach effectively ensures that the faces adjoining the locked link are not altered while the link is locked (modulo changes caused by node failures or additions, which we discuss later).

CLDP uses this basic strategy, but takes care to avoid race conditions in cases where the cross-link (and not the probed link) must be removed. Furthermore, it reduces convergence time using a few simple optimizations, since the basic strategy can cause many dropped probes. Finally, it also reduces probing overhead by avoiding probes on links which have already been determined to be routable, unless one of the adjoining faces has changed. We now describe these modifications.

First, CLDP uses *lazy* locking. That is, when CLDP needs to probe a link, it *first* sends a probe without locking the link. If this probe returns indicating either that there are no cross-links or that this link and its cross-link cannot be removed (Case 4, Figure 12), CLDP marks the link to be routable. Thus, in this case (which one expects to be common for small faces on dense networks), CLDP converges quickly without locking links. Routable links are marked *dormant* and not subsequently probed unless woken up; we later describe how this happens.

There are two other possible outcomes of a probe message; either the probed link needs to be removed from the CLDP-derived graph (e.g., Case 2, Figure 10), or its cross link needs to be removed (e.g., Case 3, Figure 11). In the former case, CLDP enters a *commit* phase, where it locks the probed link, and re-probes the link but using a specially marked “commit” message. All probes traversing a locked link in either direction are dropped. However, when a commit message traverses a locked link, a deterministic tie-break is applied which ensures that if two links on the same face are being “commit”-ed simultaneously, only one of the commit messages succeeds in traversing the face. When the “commit” probe succeeds, CLDP unlocks the probed link, and marks it as *non-routable*. The act of marking a link non-routable changes the faces adjacent to the link. As Figures 15 and 16 show, removal of a link can reveal cross-links

(e.g., the  $(X,W)$  link does not see the  $(A,B)$  cross-link until the  $(X,Y)$  link is removed from the graph). Accordingly, the changed faces must be re-probed. To accomplish this, when CLDP removes a link (i.e., marks it non-routable), it awakens the two adjacent *dormant* (see above) links (i.e., those obtained by applying the right-hand rule and the left-hand rule from this link).

The last case to consider is when a probe indicates that the cross-link (e.g., link  $(B,C)$ , Figure 11) must be removed. Recall (Figure 17) that, in general, a probe of the cross-link might not reveal the crossing. For this reason, when a probe indicates the cross-link needs to be removed, CLDP walks the corresponding face again using a “commit” probe, and locks the cross-link after the probe reaches it. When that probe succeeds, the node notifies both ends of the cross-link to mark the link non-routable.

Finally, we describe CLDP’s behavior when a link is added to or deleted from the underlying connectivity graph. When a link is added to the underlying graph, CLDP awakens the adjacent dormant links. This causes links on the corresponding faces to be probed again, eliminating cross-links when necessary. Link deletion presents a more subtle problem. Consider Figure 15, and suppose that links  $(Z,W)$  and  $(X,Y)$  have been marked non-routable. Now, suppose that link  $(A,B)$  fails. The simplest way to restore the links  $(Z,W)$  and  $(X,Y)$  to the routable sub-graph would be to periodically re-probe these links. This is what CLDP does. It is possible to design optimizations that can reduce the overhead of periodic probing. For example, node  $A$  could remember which cross-links were removed when  $(A,B)$  was probed, and notify the ends of those cross links when  $(A,B)$  fails. We have left the design of these optimizations for future work.

CLDP implements its probing actions using a simple state machine and a protocol consisting of several message types. In the interest of brevity, we refer the interested reader to [14] for a detailed specification of the CLDP protocol.

#### 4.5 Statement of Correctness

Space constraints limit us only to state the theorems that prove CLDP’s correctness. In this formal analysis, we assume that the full network graphs are static and have no degeneracies: no vertices are coincident, and no pairs of edges at a single node have the same incident bearing; there is a provably correct way to handle the latter degeneracy, elided because of space constraints. Thus, the notion of a “crossing” is well-defined. For each graph define a (perhaps empty) set of crossings  $C$ ; each element of  $C$  is a pair of edges that intersect in the plane.

Our results are based on the fact that all face walks eventually return to their starting points. We use the



following terminology to describe how a face walk returns to its starting point. An edge is *singly-walked* if a face walk starting on that edge does not return via that same edge (in the opposite direction). An edge is *doubly-walked* if it returns via the same edge in the opposite direction. The general rule in CLDP is that when a crossing is detected, no doubly-walked edge can be removed, but if one of the crossing edges is singly-walked, then an edge is removed. Our first result is a general observation about crossings in connected graphs.

**Theorem 4.1** *If a connected graph  $G$  has at least one crossing, then there is at least one face with a crossing.*

This result shows that if we had used a version of CLDP that eliminated *all* crossings then we would end up with a set of connected planar components. To help state our next result, we term a graph *CLDP-stable* if CLDP would not eliminate any edge in the graph, were the edges probed in serial fashion. We then have:

**Theorem 4.2** *Geographic routing never fails on a connected CLDP-stable graph.*

This says that if we use CLDP's rules about when to eliminate crossings, then we end up with a connected graph on which one can reliably use geographic routing.

## 5 Simulation Results

The above theorems assert CLDP's correctness on static graphs. However, to show that CLDP is practical on real wireless networks, we examine the performance of CLDP through simulation in this section, and through experimentation in the next.

**Methodology and Metrics** We implemented CLDP (and other geographic routing protocols, described below) in TinyOS [10], the event-driven operating system used on the Mica-2 motes. TinyOS code can be directly executed on TOSSIM [19], a process-level simulator that can be used to directly debug and evaluate sensor network applications and protocols. Our implementation of CLDP in TinyOS is 750 lines of nesC code. In this section, we report simulation results obtained from running CLDP and other protocols using TOSSIM's support for packet-level simulation.

In this section, we compare (whenever appropriate) CLDP's performance against three alternatives, *GPSR* denotes the full implementation of GPSR using the Gabriel Graph for planarization, greedy forwarding, and perimeter traversal for routing around voids. We use *GPSR* to provide context for CLDP's performance. *GPSR/NOPLAN* denotes a protocol that forwards packets using GPSR on the full connectivity graph (*i.e.*, *without* planarization). *GPSR/NOPLAN* delineates the baseline performance of face walking on the networks we study. *GPSR/GG/MW* includes, in addition to GPSR

and planarization, an implementation of the "mutual witness" procedure for avoiding unidirectional links and disconnections in the planarized graph when the unit-graph assumptions are violated (Section 3). *GPSR/GG/MW* quantifies the inadequacy of that proposed fix for planarization failures, thereby highlighting the need for CLDP. *GPSR/CLDP* denotes our proposed protocol using CLDP, greedy forwarding, and perimeter traversal.

In each of our simulations, we use a 200-node topology in which nodes are randomly positioned on a fixed-size two-dimensional surface. We conducted simulations on two types of networks: wireless networks with an idealized radio model with circular radio ranges (we introduce reality in the form of obstacles), and Bernoulli random graphs which have a fixed connection probability for any pair of nodes, regardless of Euclidean distance between the nodes. For our wireless network simulations, we evaluate the performance of various geographic routing protocols as a function of node density. Our measure of density is the average number of neighbors of a node. We scale the area of the surface in order to vary node density; for our highest density we use an area of 1300 x 1300 units, while for our lowest, we use an area of 2000 x 2000 units. The radio range is 180 units.

In our simulations with obstacles, the number of obstacles is indicated by a parameter  $f$ , such that  $fN$  is the total number of obstacles ( $N$  is the number of nodes). Each obstacle is of fixed length (45 units) in each of our simulations. The mid-point of the obstacle is randomly positioned on the two-dimensional surface, and the orientation of the obstacle is equally likely to be either vertical or horizontal. This obstacle model helps us stress CLDP and other protocols to varying extents in order to measure their performance.

Our Bernoulli random graphs are generated in the obvious way: we flip a weighted coin for each pair of nodes, assigning a link between them with the desired connection probability.

For each simulation we first generate a network topology. We then ensure that the topology is connected. At the beginning of the simulation, TOSSIM enforces a boot-up time during which nodes are started randomly. In our simulations, 200 nodes are started randomly in the first 30 seconds. Following the boot phase, each simulation consists of two phases. In the first phase, we let the appropriate routability determination protocol (CLDP, or GPSR's planarization and/or mutual witness procedure) execute at each node long enough for the network to converge. In the second phase, we send packets pairwise bidirectionally between nodes in a staggered manner to minimize wireless collisions. This latter phase tests for routing failures. For each data point in the graphs below, we run 50 random topologies. We have verified that this is sufficient to produce negligible 95% confidence inter-

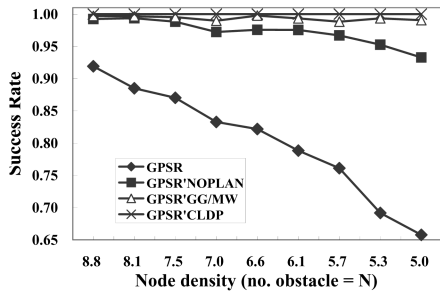


Figure 18: Success rate for  $1.0N$  obstacles.

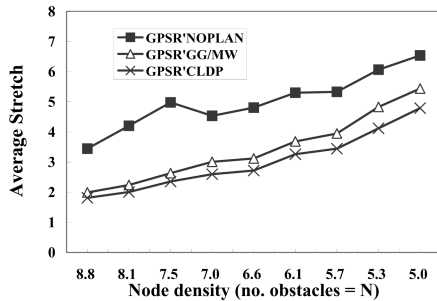


Figure 20: Average stretch for  $N$  obstacles.

vals for the mean values of our metrics.

We do not simulate packet losses due to interference or buffer overrun in either phase. Our simulations do drop packets, however, when face routing fails. Packet losses would increase the convergence time of CLDP, or would alter the level of concurrent probing in CLDP. Our simulation methodology already introduces significant concurrency by ensuring that all nodes start at nearly the same time. (Note that our testbed measurements include interference and buffer overrun effects, of course.)

We use two primary measures of performance. The *success rate* measures the fraction of sender/receiver pairs for which packet transmissions from the sender are successfully received. The *average stretch* measures the average of path stretch for all sender/receiver pairs. The stretch of a path is the ratio of the number of hops using the routing scheme in question to the number of hops in the shortest path. We also evaluate the overhead and convergence time of CLDP; we define these metrics below.

Given space constraints, we only present a sampling of simulation results extensively described in [14]. In particular, we omit results validating CLDP's correctness on networks with localization errors as well as a detailed discussion of CLDP's performance on random graphs.

**Wireless Networks with Obstacles** Figure 18 shows the success rate as a function of node density for our various protocols, in the presence of  $N$  obstacles. Note that this is an extremely harsh environment, with as many obstacles as nodes. As expected, CLDP allows perfect delivery success across all node densities we evaluated. Interestingly, GPSR's planarization procedure fails rather dramatically in the presence of even a moderate num-

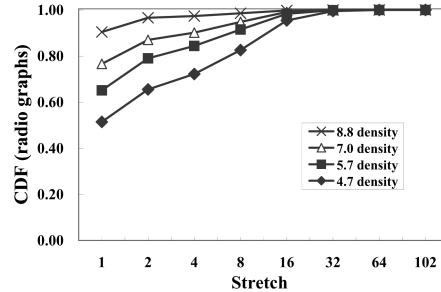


Figure 19: CDF of stretch ( $1.0N$  obstacles).

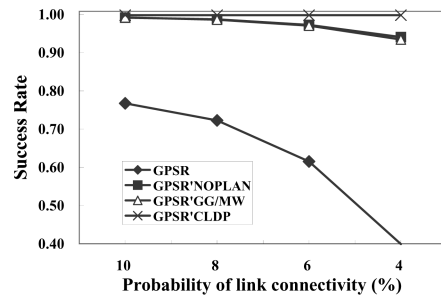


Figure 21: Random graph success rate.

ber of obstacles. In these circumstances, it appears to be more advantageous simply to use GPSR on the connectivity graph without planarization. The mutual-witness procedure fixes many of GPSR's shortcomings and is close to perfect in some cases. At most densities it can establish paths between 99% or more node pairs, but it is never perfect. In a real deployment, however, MW fails far more dramatically, as discussed in Section 3.

Figure 20 plots the average stretch as a function of node density for our various protocols, in the presence of  $N$  obstacles. CLDP exhibits an average stretch between 2 and slightly above 4, with a higher stretch at lower densities. CLDP outperforms GPSR/GG/MW in this respect; CLDP removes only cross links, but GPSR/GG/MW removes all links that are witnessed by planarization and hence incurs higher stretch. However, CLDP may exhibit long paths. This is evident from the CDF of stretch for CLDP (Figure 19, with  $N$  obstacles). Notice the long tail of the distribution, in which some paths have a stretch of over 100! Across the range of densities we explore, though, 60-95% of the paths have a stretch less than 2.

**Random Graphs** To stress CLDP, we also simulated it on Bernoulli random graphs with various connectivity probabilities. As Figure 21 shows, CLDP exhibits no routing failures, even on random graphs. By contrast, all other variants exhibit significant routing failures on sparse random graphs (low connection probabilities). In particular, MWP exhibits more systematic routing failures than on wireless networks. Clearly, none of these other protocols is practical for routing on random graphs.

**Overhead** We measured how many CLDP messages are needed to add a link to a wireless network with  $N$  ob-

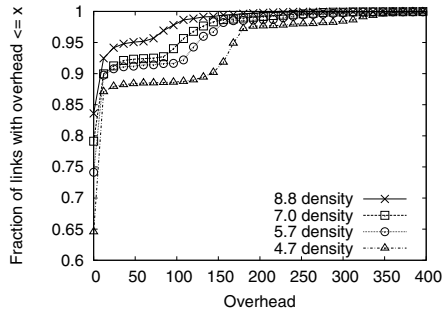


Figure 22: Overhead for wireless network with  $N$  obstacles.

stacles. This gives us some idea of the overhead incurred by CLDP. In our experiments for measuring overhead, after a network has reached steady state, two nodes not directly connected to each other are randomly selected and an additional link between them is activated.

The *overhead* is the total number of CLDP control messages (probe and commit) traversing a link in either direction until the network has converged. Figure 22 plots the distribution of link overheads averaged over 20 link additions on each of 200 wireless topologies. It shows that about 85%-90% of links see fewer than 4 messages, but a very small fraction of links see upwards of 100 messages. This latter phenomenon can be explained as follows. Assume that a new link is added which crosses existing edges. When CLDP removes these crossing edges, it needs to wake up all links on the faces adjacent to the removed link in order to detect successively hidden cross-edges. These links generate probe messages to see if they are crossed by others. Hence, the number of messages observed on a link depends on the size of the face. Clearly, in our wireless topologies (particular in the ones with lower density), there exist long faces.

**Network Convergence Time** We measured how quickly CLDP converges both on wireless networks with  $N$  obstacles and on Bernoulli random graphs. In experiments of convergence time, 200 nodes are initially started roughly simultaneously. In our CLDP implementation, nodes periodically probe their attached links before the links become dormant. Thus, the convergence time of CLDP is a function of this periodic timer. The *convergence time* of a link is defined as the number of CLDP probing intervals before a link becomes dormant and remains thus (Section 4.4). Notice that our experiments measure link convergence at *startup*; one would expect that in steady-state, the time for convergence after a single link failure and recovery can be expected to be considerably lower. Figure 23 shows the convergence time distribution for wireless networks with  $N$  obstacles. In Figure 23, about 95% of links converge within 4 probe

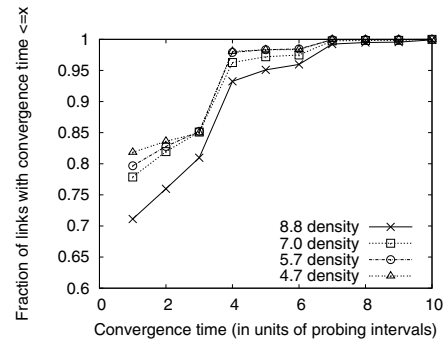


Figure 23: Convergence time distribution for wireless network with  $N$  obstacles.

intervals and all links converge within 9. In practice (Section 6), convergence times are slightly longer.

**Network Dynamics** Finally, we conducted simulations to evaluate CLDP's resilience to network dynamics. These experiments were done on 200 wireless networks with  $N$  obstacles as well as 200 Bernoulli random graphs. In all experiments, we took each given topology, randomly selected some links, and marked them non-routable in order to force those links to be re-probed by CLDP. Then we let CLDP execute at each node. Initially, these non-routable links are not used for CLDP probing. Over time, however, these links are woken up and are CLDP-probed. After all links had reached a dormant state, we determined whether packets could be routed between all pairs. In every case, CLDP converged to a network with 100% pairwise connectivity. Note that if a link flaps, CLDP will continuously attempt to probe the link. It might be possible to dampen this activity, but we have not investigated such mechanisms.

**Summary** In every simulation experiment, CLDP establishes routing paths between all node pairs. It exhibits reasonable stretch, overhead, and convergence times. Moreover, it works well under network dynamics. We next measure how CLDP performs on actual wireless testbeds.

## 6 Experimental Results

In this section, we describe CLDP's performance in deployment on wireless sensor network testbeds.

### Testbeds and Experiments

We deployed CLDP on two different sensor node testbeds; as geographic routing's behavior is sensitive to the detailed placement of nodes and obstacles, we sought to demonstrate CLDP's behavior for multiple node and obstacle placements, to the extent possible using testbed resources at our disposal. The first testbed we shall label  $R$ , and consists of 75 Mica-2 "dots" with 433 MHz radios, deployed roughly one per room on one floor of Berkeley's Soda Hall. As described in Section 3, this was a shared testbed infrastructure, so we had no control

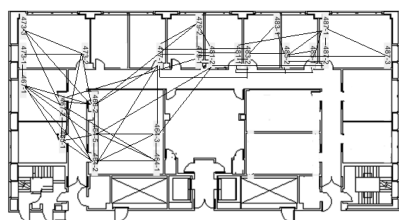


Figure 24: Node layout for  $R_s$ .

over node layout and were able to use only a subset of the nodes for our experiments. We report performance measurements obtained on two different subsets of this testbed:  $R_s$  (Figure 24) which contains 23 nodes, and  $R_m$  (Figure 6) which contains 50 nodes.

The second testbed, which we shall call  $C$ , consists of 51 Mica-2 “dots” deployed across a floor of Intel Research Berkeley, of which we were able to use 36. In addition to environmental differences (cubicles in  $C$  vs. rooms in  $R$ ), the testbeds differ in that  $C$ ’s nodes are suspected to have poorer quality radios. Furthermore,  $C$ ’s radios operate at 916 Mhz, and incur interference from other nearby devices in that unlicensed band. Again, on  $C$  we had no control over node layout.

As described in Section 3, in these testbeds we adjusted node transmit power to obtain a multi-hop topology. For  $R_m$  and  $C$ , notice that the topologies stress geographic routing protocols significantly—they contain two or more “clusters” of sensor nodes linked by one or two links, a configuration that triggers perimeter-mode routing frequently. Of course, such topologies aren’t very practical since their capacity would be constrained by the bottleneck links. However, they can give some idea of worst-case CLDP performance, as we discuss below.

We thus conducted three sets of experiments:  $R_m$ ,  $R_s$ , and  $C$ . In each experiment, nodes were configured with their locations. We started all nodes roughly simultaneously and let CLDP probing converge. We logged every packet (all devices in both testbeds had console access through a serial port), and we also recorded pair-wise link quality. In addition, for  $R_s$ , we conducted an experiment in which we sent 50 packets between each pair of nodes in order to measure packet delivery performance. Our packet forwarding implementation tries up to three link-layer retransmissions per hop.

## Results

In this section, we report on the performance of CLDP according to a variety of metrics. At the outset, we point out that in all three experiments, CLDP was immune to the pathologies described in Section 3 and established pairwise connectivity between 100% of node pairs.

**Path Performance** One aspect of a routing protocol’s path performance is stretch. For most node pairs (Figure 26), CLDP’s stretch is reasonable (2 or 3). However, CLDP does exhibit fairly significant stretch (up to

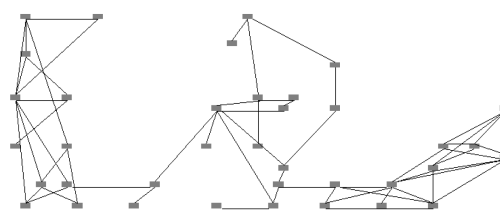


Figure 25: ... for  $C$ .

20 in some cases) for a small fraction of node pairs. High stretch arises from long paths between pairs of nodes. Often, such long paths arise during traversal of the outer perimeter of the network.

One might argue that comparing CLDP paths with shortest paths is unrealistic, since shortest-path routing is known to offer low throughput [3] over a wireless network whose links span a wide range of packet delivery rates. For this reason, we measure the quality of CLDP’s path selection. Figure 27 computes the distribution of pairwise packet delivery rates (the fraction of delivered packets) for both CLDP (measured on  $R_s$ ) and ETX (computed from link quality estimates on  $R_s$ ).<sup>8</sup> CLDP’s packet delivery performance is comparable to, but slightly worse than this “idealized” ETX. A comparison with a real implementation of ETX might lessen the discrepancy between the two considerably. Finally, we note that ETX (when implemented on a proactive protocol like DSR, on a network with a dynamic topology) is likely to incur higher overhead than CLDP.

**Convergence Time** Figure 28 shows that most links converge within 15–20 probe intervals; with a 15 second probe timer, this corresponds to about 4.0 minutes. However, some links exhibit very long convergence times (up to 70 intervals). Our experiment measures *startup* convergence, since all the nodes are started roughly simultaneously. For CLDP, this is the worst case: when all links are simultaneously probed, link locking will delay convergence significantly. This also explains why  $R_m$  and  $C$  show a qualitatively different behavior; the bottleneck links between clusters induce significant probe contention.

A more realistic measure of link convergence time is the time it takes for a single link to converge when the rest of the network is in steady state. Even for a moderate size network, we couldn’t automate this experiment easily, so we estimate this interval. We obtained our *estimated steady-state convergence time* by counting only those CLDP probes that do not encounter locked links. By this measure, CLDP converges very fast (Figure 29); more than 99% of the links converge within 6 intervals in all three experiments.

In addition, we also conducted an experiment where we started with a converged network, and manually disabled and then re-enabled an arbitrary link chosen from ten arbitrarily selected nodes. We then measured the time

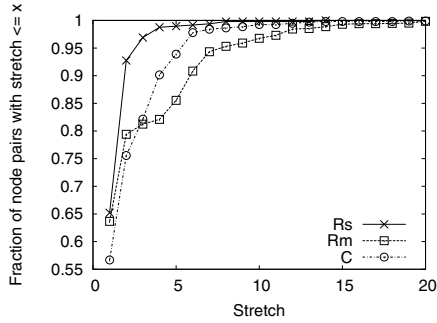


Figure 26: CDF of stretch.

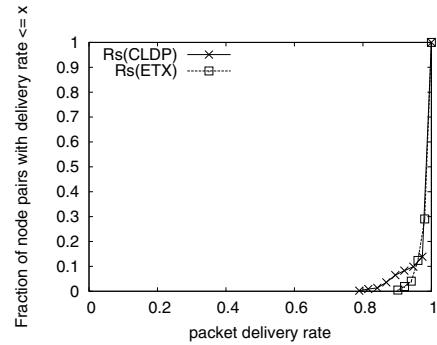


Figure 27: CDF of pairwise packet delivery rate.

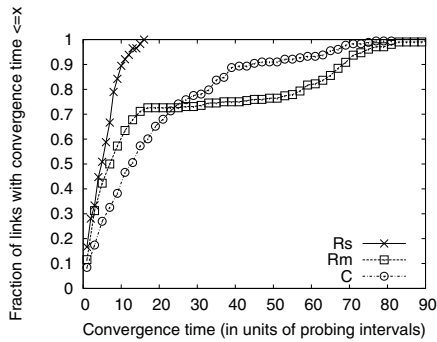


Figure 28: CDF of convergence time.

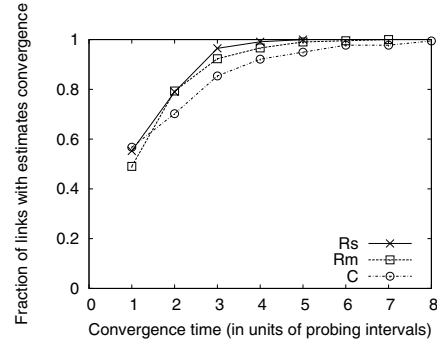


Figure 29: CDF of estimated steady-state convergence.

for CLDP to converge after each transition. After disabling a link, CLDP converged on average within 1.86 probing intervals; after enabling a link, CLDP converged on average within 0.59 probing intervals.<sup>9</sup>

**Overhead** Finally, we quantify the overhead of CLDP from our measurements. The primary metric we study is the distribution of probing overhead on individual links. However, rather than merely count the number of CLDP probe messages on each link for the entire duration of the experiment, we compute the average number of messages on each link<sup>10</sup> *per probing interval*. Normalizing the overhead this way helps us compare different experiments whose convergence times are different. Figure 30 shows that the overhead of CLDP is quite low; even on the busiest link, CLDP incurs less than one packet per second (if we assume a probe interval of 15 seconds), and on most links the overhead is significantly less.

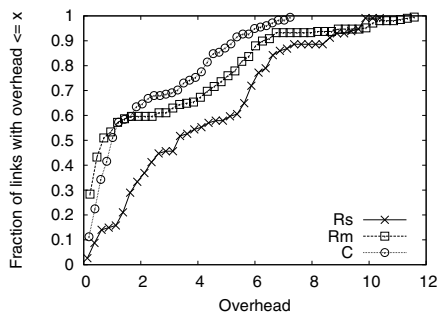


Figure 30: CDF of overhead.

## 7 Conclusion

We have motivated, described, and evaluated CLDP, which, to our knowledge, is the first distributed planarization protocol that renders geographic routing correct on arbitrary graphs. Simulations and measurements on real testbeds indicate that CLDP is quite practical: it offers high delivery rates, low overhead, and fast convergence. In future, we plan to investigate CLDP's overhead and robustness on more dynamic topologies, as well as the effect of localization errors on CLDP's path stretch in deployment.

## Acknowledgments

We thank the anonymous reviewers for their comments, and Ellen Zegura for her thoughtful shepherding of this paper. We are further indebted to Om Gnawali for the comparison with ETX, and Jerry Zhao, Xin Li, and Wei Hong for helping us to use the UC Berkeley and Intel Research Berkeley sensor network testbeds.

## Notes

<sup>1</sup>Other face routing techniques [17] can be used as well; CLDP preserves their correctness, but may affect their performance.

<sup>2</sup>For lack of space, we only present the resulting theorems, not their proofs, which may be found in [14].

<sup>3</sup>We note that there exist other routing algorithms that make use of position information, such as LAR [16], but we restrict the scope of our work to the family of face-routing algorithms in which a node forwards to a single neighbor on the basis of geographic information.

<sup>4</sup>We refer to links and edges interchangeably throughout the paper.

<sup>5</sup>Other face-change rules are possible, including changing faces at the edge whose crossing of  $\overline{SD}$  is the *closest* such crossing to  $D$  on the

current face. We use the first crossing, not best crossing, throughout this paper; this choice is known to be average-case efficient, and has been refined [17] to be worst-case optimal.

<sup>6</sup>While pathologies in geographic routing are sensitive to the particular placement of nodes and the obstacles between them, we observed similar results on the two testbeds, and thus expect similar behavior in other real deployments.

<sup>7</sup>In principle, CLDP wouldn't need additional mechanisms to function under mobility, and would work well when link disconnections due to mobility occur on much longer timescales than the time required to complete CLDP probes.

<sup>8</sup>While CLDP uses only "good" links, our simulation of ETX is not similarly constrained.

<sup>9</sup>If a link is probed exactly once before it becomes dormant, that counts as a convergence time of zero.

<sup>10</sup>Although we count the number of messages on a link, recall that in our implementation, each message on a "link" constitutes a radio broadcast. Interpreted thus, our measure of overhead indicates the number of data packets that CLDP probing displaces in our deployment. A more general measure, and one that we have not investigated since it depends on deployment density and other environmental factors, is the fraction of transmission capacity that CLDP probing overhead occupies.

## References

- [1] S. Biswas and R. Morris. Opportunistic routing in multi-hop wireless networks. In *Proc. ACM HotNets Workshop*, Boston, MA, USA, Nov. 2003.
- [2] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. In *Proc. ACM DIALM Workshop*, pages 48–55, Seattle, WA, USA, Aug. 1999. ACM.
- [3] D. De Couto, D. Aguayo, B. Chambers, and R. Morris. Performance of multihop wireless networks: Shortest path is not enough. In *Proc. ACM HotNets Workshop*, New Jersey, USA, Oct. 2002.
- [4] G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI/RR-87-180, USC/Information Sciences Institute, Mar. 1987.
- [5] K. Gabriel and R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.
- [6] D. Ganesan, D. Estrin, A. Woo, D. Culler, B. Krishnamachari, and S. Wicker. Complex behavior at scale: An experimental study of low-power wireless sensor networks. Technical Report UCLA/CSD-TR-02-0013, University of California, Los Angeles, Computer Science Department, 2002.
- [7] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Geometric spanner for routing in mobile networks. In *Proc. ACM MobiHoc*, pages 45–55, Oct. 2001.
- [8] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler. The nesC language: A holistic approach to networked embedded systems. In *Proc. ACM SIGPLAN PLDI*, San Diego, CA, June 2003.
- [9] R. Gummadi, N. Kothari, Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. Reduced state routing in the Internet. In *Proc. ACM HotNets Workshop*, San Diego, USA, Nov. 2004.
- [10] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *Proc. 9th ACM ASPLOS*, pages 93–104, Cambridge, MA, USA, Nov. 2000. ACM.
- [11] B. Karp. *Geographic Routing for Wireless Networks*. PhD thesis, Harvard University, 2000.
- [12] B. Karp. Challenges in geographic routing: Sparse networks, obstacles, and traffic provisioning. Presentation at the DIMACS Workshop on Pervasive Networking, May 2001.
- [13] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proc. ACM/IEEE MobiCom*, pages 243–254, Boston, Mass., USA, Aug. 2000. ACM.
- [14] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. Practical and robust geographic routing in wireless networks. Technical Report 04-832, Department of Computer Science, University of Southern California, 2004.
- [15] L. Kleinrock and H. Takagi. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Trans. Comm.*, 32(3):246–257, 1984.
- [16] Y.-B. Ko and N. Vaidya. Location-aided routing in mobile ad hoc networks. In *Proc. ACM/IEEE MobiCom*, Aug. 1998.
- [17] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: Of theory and practice. In *Proc. ACM PODC*, Boston, MA, USA, July 2003.
- [18] F. Kuhn, R. Wattenhofer, and A. Zollinger. Ad-hoc networks beyond unit disk graphs. In *Proc. ACM DIALM POMC Workshop*, Sept. 2003.
- [19] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: accurate and scalable simulation of entire tinyOS applications. In *Proc. ACM Sensys*, pages 126–137. ACM Press, 2003.
- [20] X. Li, Y. J. Kim, R. Govindan, and W. Hong. Multi-dimensional range queries in sensor networks. In *Proc. ACM Sensys*, Los Angeles, CA, USA, Nov. 2003.
- [21] J. Newsome and D. Song. GEM: Graph embedding for routing and data-centric storage in sensor networks with geographic information. In *Proc. ACM Sensys*, Nov. 2003.
- [22] A. Rao, S. Ratnasamy, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proc. ACM/IEEE MobiCom*, pages 96–108, Oct. 2003.
- [23] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: A geographic hash table for data-centric storage. In *Proc. ACM WSNA Workshop*, pages 78–87, Atlanta, Georgia, USA, Sept. 2002. ACM.
- [24] K. Seada, A. Helmy, and R. Govindan. Localization errors on geographic face routing in sensor networks. In *Proc. IEEE IPSN Workshop*, Berkeley, CA, USA, Apr. 2004.
- [25] R. Sokal and D. Matula. Properties of Gabriel graphs relevant to geographic variation research and the clustering of points in the plane. *Geographical Analysis*, 12:205–222, 1980.
- [26] G. Toussaint. The relative neighborhood graph of a finite planar set. *Pattern Recognition*, 12(4):261–268, 1980.
- [27] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proc. ACM Sensys*, Los Angeles, CA, November 2003.