Ad Hoc Networks xxx (2010) xxx–xxx

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

GeoCross: A geographic routing protocol in the presence of loops in urban scenarios

Kevin C. Lee *, Pei-Chun Cheng, Mario Gerla

University of California, Los Angeles, CA 90095, United States

A R T I C L E I N F O

Article history:
Available online xxxx

Keywords:
Geographic routing
Cross-link detection
VANET routing

A B S T R A C T

In this paper,we propose GeoCross, a simple, yet novel, event-driven geographic routing protocol that removes cross-links dynamically to avoid routing loops in urban Vehicular Ad Hoc Networks (VANETs). GeoCross exploits the natural planar feature of urban maps without resorting to cumbersome planarization. Its feature of dynamic loop detection makes GeoCross suitable for highly mobile VANET. We have shown that in pathologic cases, GeoCross's packet delivery ratio (PDR) is consistently higher than Greedy Perimeter Stateless Routing's (GPSR's) and Greedy Perimeter Coordinator Routing's (GPCR's). We have also shown that caching (GeoCross + Cache) provides the same high PDR but uses fewer hops.

Published by Elsevier B.V.

1. Introduction

The ever growing spread of vehicles and roadside traffic monitors, the advancement of navigation systems [2], and the low cost of wireless network devices provide incentives for car manufacturers to equip vehicles with advanced services ranging from real-time traffic reports to crash avoidance alerts and emergency networking. The major objective of emerging vehicular communication standards [1] has been to improve driver safety. One of the requirements is to provide efficient communications among remote vehicles in emergencies using exclusively the vehicular grid. In fact, when equipped with Wireless Access in Vehicular Environments (WAVE) [3] communication devices, cars and roadside units form a highly dynamic network called Vehicular Ad Hoc Network (VANET). Similar to Mobile Ad Hoc Networks (MANETs), vehicle radios have limited radio range and must communicate with one another by multi-hop routing protocols. Designing an efficient multi-hop routing protocol between vehicles is thus an important and challenging problem.

On-demand approaches such as AODV [31] or DSR [15] suffer from flooding overhead. Consequently, these approaches often do not scale well when traffic density increases. Proactive approaches such as OLSR [7] do not cope well with high node mobility. Hierarchical routing approaches such as HSR [30] or LANMAR [29] may not be efficient since clusterheads need to be updated frequently and communication between two nodes needs to go through their clusterhead even though the two nodes are within radio range of each other. *Position-based* routing has proven to be well suited for highly dynamic environment such as VANETs due to its simplicity and efficiency.[1] Data are routed to nodes based on their geographic location. Due to the low cost and prevalence of global positioning system (GPS) and recent research on Geo-Location Services [8,25,33], geographic routing protocol such as GPSR [19], its variants [24,27], and CBF [9,10] become a practical and suitable routing solution [14,26,28] for VANET. In GPSR, nodes forward packets greedily by choosing the neighbor with shortest distance to the destination. However, there are some cases where packets will reach a local maximum.

* Corresponding author. Fax: +1 3108257578.
E-mail addresses: kclee@cs.ucla.edu (K.C. Lee), pcheng@cs.ucla.edu (P.-C. Cheng), gerla@cs.ucla.edu (M. Gerla).

[1] Refer to [34] for a survey of position-based routing in vehicular ad hoc networks.

1570-8705/$ - see front matter Published by Elsevier B.V.
doi:10.1016/j.adhoc.2009.12.005

In such a case, the node switches from greedy mode to perimeter mode.

In perimeter mode, GPSR requires a planar graph, a graph with no cross-edges,[2] in order to make sure that packets do not get trapped in routing loops. Distributed algorithms [5,19] have been proposed to produce Relative Neighborhood Graph (RNG) and Gabriel Graph (GG) that are known to be planar. These RNG and GG planarization algorithms require the *unit graph* assumption that states the a link can only exist between two nodes if and only if these two nodes are within some threshold distance. However, in a typical sensor network, due to obstacles and irregular communication range, nodes can still communicate with each other outside of the threshold distance and violate the unit graph assumption. Kim et al. [20,21] have observed anomalies of these planarization algorithms in GPSR. More specifically, these anomalies are network partitions, asymmetric links, and cross-links. Mutual witness protocol [13–15] have tried to remove anomalies of network partitions and asymmetric links. CLDP [20] and LCR [12] have tried to remove cross-links by probing for cross-links and removing them proactively. Nodes would then store some of their links as unroutable to their neighbors to avoid cross-links in the future.

In VANETs, some [24,27] have recognized that urban roads naturally form a planar graph where intersections are nodes and edges are formed by two intersections. The natural formation of a planar graph from the underlying roads eliminates the need for node planarization and speed up recovery from the perimeter mode. Since the unit of a perimeter is from one intersection to the next in perimeter mode, each node need not make ‘‘baby” steps towards the destination. Rather, nodes can forward as far as they can from one intersection to the next. Decision about which road to forward along will then be made by a node at a junction.

However, problems, in particular, cross-links, still arise when nodes do not exist at a junction. The presence of cross-links would never exist in a planar urban grid (with unobstructed road segments and no tunnels and bridges) if at least one node existed at each junction. Unless we install traffic lights with routers at each intersection, we must learn how to deal with cross-links resulting from vehicles within range of each other across an empty junction. Furthermore, the problem can exacerbate where intersecting roads are on several levels (e.g., bridges and highway interchanges) as most of the time cars do not stop at the intersection.

To this end, we propose GeoCross protocol, a simple, yet novel, event-driven protocol that routes packets in the presence of cross-links in an urban VANET. Unlike CLDP and LCR, no preliminary probing is needed: GeoCross removes cross-links as it routes the packets. We coin this feature as *dynamic loop detection*. Because GeoCross does not probe, its resource overhead in space and time is significantly lower. In space, a mobile network is not overwhelmed by the sheer traffic of probing messages. In time, a node need not wait for probe messages to notify the node that the path is clear of cross-links before it can forward packets. GeoCross' dynamic loop detection makes it ideally suited for urban VANETs. The rest of the paper is organized as follows: Section 2 provides a short discussion of the current efforts in dealing with cross-links in sensor networks. In Section 3, we describe the source of cross-link problem in VANET and pose desiderata of our solution. Section 4 motivates our solution by first showing a model of cross-link frequency and verifying it with realistic scenarios. In Section 5, we formally introduce GeoCross to solve cross-links by showing its design, examples, and optimizations. Section 6 presents the evaluation of GeoCross using a realistic city map with building blocking model. Section 7 concludes the paper and presents our future work.

2. Cross-links in geographic routing

In this section, we briefly describe the Greedy Perimeter Stateless Routing (GPSR), its planarization algorithms to remove cross-links, the pitfalls of these planarization algorithms, and further remedies to planarization algorithms.

2.1. Greedy perimeter stateless routing

The Greedy Perimeter Stateless Routing (GPSR) algorithm belongs to the category of position-based routing, where a node forwards a packet to an immediate neighbor which is geographically closer to the destination node. This mode of forwarding is termed *greedy mode*. Due to the non-uniform distributions of nodes and to the existence of radio power limitations, it is possible that a packet reaches a local maximum where its distance to the destination is closer than its neighbors' distance to the destination. To make further advance from the local maximum, a recovery mode is used to forward a packet to a node that is closer to the destination than the node where the packet encountered the local maximum. The packet resumes forwarding in greedy mode when it reaches a node whose distance to the destination is closer than the node at the local maximum to the destination.

Many recovery algorithms have been developed including GPSR [19], Compass [22], Face-1 and Face-2 [6], or GOAFR+ [23]. GPSR recovers from a local maximum using *perimeter mode* based on the right-hand rule shown in Fig. 1. The rule states that when a node x first enters into the recovery mode, its next forwarding hop y is the node that is sequentially counterclockwise to the virtual edge formed by x and destination D. Afterwards, the next hop z is sequentially counterclockwise to the edge formed by y and its previous node x shown in Fig. 1. While walking the face, however, if the edge $\overline{yz}$ formed by the current node and the next hop crosses the virtual edge $\overline{xD}$ and results in a point that is closer than the previous intersecting point x, perimeter mode will perform a *face change* in that the next hop w is chosen sequentially counterclockwise to the edge $\overline{yz}$ where the closer intersecting point was found.

Note that if the graph is not planar, that is, there are cross-edges in the graph, routing loops may occur.

[2] In this paper, we will refer cross-edges as cross-links.

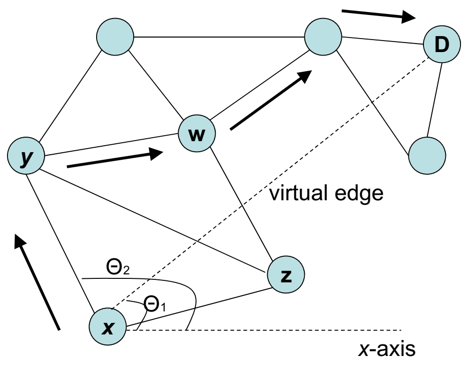


Fig. 1. Demonstration of perimeter forwarding mode by the right-hand rule. The arrows indicate packet forwarding as a result of face change. Perimeter forwarding by right-hand rule and face change.

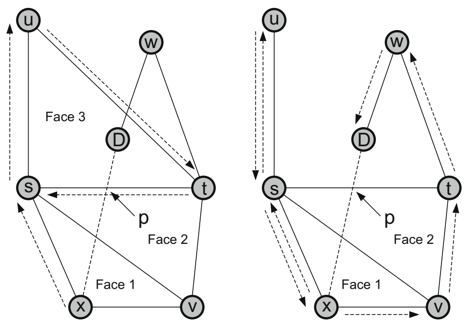


Fig. 2. On the left, packet will loop around face 3; on the right, packet will eventually route to D through $u,\ s,\ x,\ v,\ t$, and w.

Consider Fig. 2, x tries to reach D in perimeter mode. The packet will eventually loop around face 3 with no intersecting point closer than p. Had the cross-edge $\overline{ut}$ been removed, the packet would travel the exterior face $u,\ s,\ x,\ v,\ t$, and w to reach D.

2.2. GG and RNG planarization algorithms

Given that perimeter mode must operate on planar graphs to avoid routing loops, two distributed algorithms were introduced to produce Relative Neighborhood Graph (RNG) [32] and Gabriel Graph (GG) [11] that are known to be planar. Both RNG and GG yield a connected planar graph so long as the connectivity between two nodes obeys the *unit graph assumption*: for any two vertices, they must be connected by an edge if the distance between them is less than or equal to some threshold distance d and must *not* be connected by an edge if the distance between them is greater than d. The algorithm in producing either a GG or RNG graph is run distributedly by every node upon receiving its neighbors' locations by periodic beacons in the network. Fig. 3 demonstrates how node A or node B removes the link $\overline{AB}$ in the presence of a *witness* node w. Note that the link $\overline{AB}$ is removed as long as there is a node in the shaded region. For GG, the shaded region is the circle with diameter $\overline{AB}$. For RNG, the shaded region is the intersection of two circles centered at A and B, each with radius $\overline{AB}$.

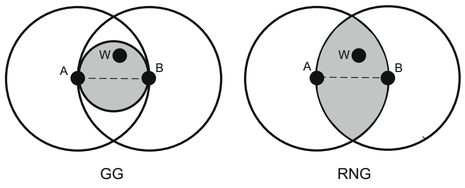


Fig. 3. Shaded region defined in GG and RNG.

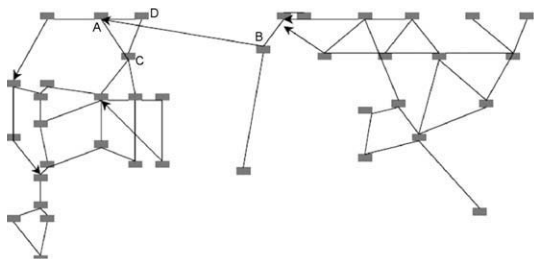


Fig. 4. Cross-link pathology in real office suite.

Kim et al. [12,20] have observed that real radios violate the unit graph assumption: nodes can be connected if their distance is greater than the threshold distance d and nodes can be disconnected even though their distance is less than or equal to d. The result of violation of the unit-graph assumption gives rise to three main pathologies: network partitions, asymmetric links, and cross-links. Fig. 4 illustrates the pathology of cross-links in a real office-suite setting. Due to either obstacle or irregular radio ranges, node B can communicate with node A but not vice versa. Thus the link between A and B is an asymmetric link. And since node C and D do not have any witness between them, the link $\overline{CD}$ is not removed and it crosses with link $\overline{BA}$. Network partition is also likely if both A and B can hear C and D and as a result removes the link $\overline{AB}$ between them. Thus, a cross-link removal algorithm needs to consider the complication of network partition as it removes the cross-link so that there is no disconnectivity in the resulting graph. This paper will focus on the pathology of cross-links since pathologies of network partitions and asymmetric links do not occur in the way a planar graph is constructed in VANETs. More specifically, a planar graph in VANETs is constructed using roads rather than nodes. The constructed planar graph is not subject to any assumption made by the planarization algorithms. However, in removing cross-links in VANETs, attention will be given to remove the offending cross-link in such a way that network partition does not occur. We will explain this more in detail in Section 3.

2.3. CLDP and LCR remedying planarization pitfall

So far we have shown that in a static sensor network, GPSR planarization algorithms fail, leading to routing loops. To address this problem, Kim et al. [20] proposed Cross-Link Detection Protocol (CLDP) that produces a connected subgraph[3] on which one can reliably use geographic routing. In CLDP, each node recursively[4] and repeatedly[5] probes its adjacent links to see if it is crossed by other links. As the probe follows the right-hand rule going through the network, it records the links that cross the currently probed link. When the probe returns to the original node, the node carefully decides which of its cross-links is to be removed to avoid partitioning the network. The current probing node then notifies affected nodes that such a link is unroutable. Consequently, future packet will not consider the link for forwarding.

In Fig. 5a, a CLDP probe on link $\overline{BC}$ originates from C. The probe eventually returns back to C and records link $\overline{AD}$ that crosses $\overline{BC}$. Because the packet traverses $\overline{BC}$ once in each direction, CLDP-enabled node C infers that removing link $\overline{BC}$ would disconnect the network. Instead, it removes link $\overline{AD}$. Node A and D are then notified about the unroutable link between them.

While CLDP is promising, its message complexity, scalability, and convergence time are not. The message complexity stems from the fact that probing is done in a concurrent fashion for speed. Concurrency introduces race condition and creates complex locking mechanism. As the number of cross-links grows, the global number of probes also increases. The *repeated* probing of one link to remove *all* cross-links and *recursive* probing of *all* the other links hurt CLDP's scalability. Furthermore, according to [20], for a 200-node wireless network, most links in concurrent CLDP converge within 15–20 probes, where each probe's frequency is 15 s. This corresponds to a convergence time of 4 min. This long convergence time might be acceptable in ad hoc networks with no mobility such as sensor networks. However, it is not suitable for VANET when high mobility constantly changes the underlying topology.

Realizing CLDP's message complexity, Kim et al. developed Lazy Cross-Link Removal (LCR) protocol that reduces the message complexity by only removing *loop-inducing cross-links* (LICLs). Instead of making the protocol responsible for removing all cross-links possible, the task is shifted to mutual witness algorithm [16–18] that generates an *approximate* planar graph of the underlying topology. The rest of the cross-links will then be taken care of by LCR as on an as-needed basis. In Fig. 5b, for example, packets sent from S to D will eventually loop back to S in perimeter mode and report back $\overline{KL}$ and $\overline{JM}$ as loop-inducing cross-links. By removing either $\overline{KL}$ or $\overline{JM}$, LCR can forward packets successfully from S to D.

Although LCR does not suffer the high message complexity as CLDP, it does not completely eliminate it, either. Since it uses CLDP to remove cross-links, there is still the need of probing messages. Scalability still suffers as LCR tries to remove *all* loop-inducing cross-links going out of source node. Furthermore, "when LCR detects that a looped

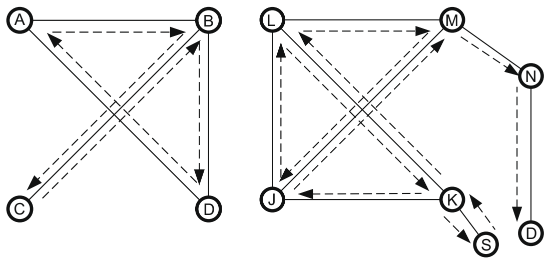


Fig. 5. CLDP and LCR examples.

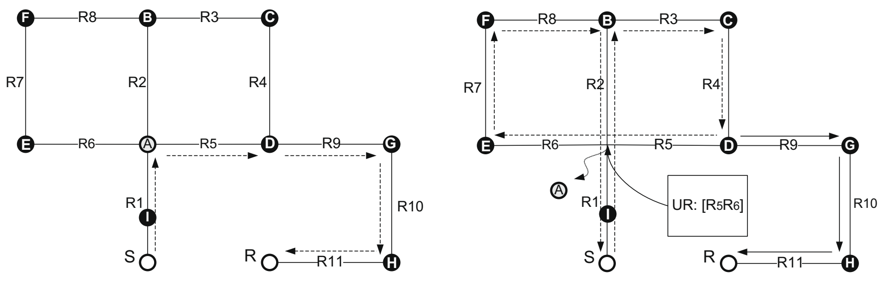
walk does not contain a cross-link, LCR initiates a recursive search on the adjacent faces for cross-link [12]". Although the approach improves the packet delivery ratio, the cost of probing messages for such improvement grows with increasing level of recursion and the number of nodes in the network. The "recursive search procedure can, in the worst-case, incur overhead comparable to CLDP [12]". The overhead worsens in VANET when nodes are highly mobile. The cost of getting the expensive none-cross-link path between the source and destination is incurred every time nodes move. It is conceivable that LCR's control packets can overwhelm the network that application-layer communication becomes nearly impossible.

3. Problem statement

Unlike sensor networks, VANETs are characterized by highly mobile nodes with relatively sufficient power and computing resources. Because nodes are highly mobile in VANETs, node planarization can become a cumbersome and continuous process. In their work of GPCR [27], Lochert et al. have observed that urban street map naturally forms a planar graph such that node planarization can be completely eliminated. In this new representation of the planar graph using the underlying roads, nodes would forward as far as they can along roads in perimeter mode and stop at junctions where decision about which next road segment to turn into can be determined. Since this new representation of planar graph does not rely on node planarization, many pathologies that arise as a result of the violation of unit graph assumption disappear, leaving only cross-links. The problem of cross-links becomes especially acute in urban VANETs filled with traffic lights and intersection roads.

A typical instance of the cross-link problem can be seen in Fig. 6a where S routes packets in perimeter mode to R. If there are vehicles at *all* the junctions, the packet can be delivered to destination R following the path shown in Fig. 6a. However, suppose that no node exists at junction A; that is, junction A is empty. Fig. 6b shows that this graph is no longer planar. We can either remove the cross-link which might disconnect the graph or we can forward over the cross-link which violates the principle of face routing and induces a routing loop. We address this problem by proposing GeoCross to detect and remove cross-links in VANETs to improve packet delivery ratio with the following desiderata:

[3] Note that the subgraph need not be planar. CLDP removes all cross-links that are removable so as not to create network partitions.
[4] All links have to be probed.
[5] For each link, CLDP may probe repeatedly to remove all cross-links as in the case of multiple cross-links in [20].



(a) Simplified street map (no cross links).

(b) Empty junction A creates cross links, $[R_1R_2]$ and $[R_5R_6]$, and induces a loop back to S. GeoCross routes over cross link and delivers packets successfully to R.

Fig. 6. GeoCross routing example.

Table 1
Comparisons of the three routing protocols.

Protocol	No control msgs	No concurrency/locks	Remove only LICLs[a]	No GG/MW	Dynamic loop detection	Complete
CLDP				✓		✓
LCR			✓			✓
GeoCross	✓	✓	✓	✓	✓	

[a] LICL stands for loop-inducing cross-links.

1. Cross-link detection and removal should be a non-intrusive procedure. There should not be any control messages generated as a result of detecting and removing cross-links.
2. Cross-link removal should be done only when necessary. Cross-links are removed only if the removal facilitates packets to make advancement towards the destination.
3. No state information should be kept in nodes as nodes are constantly moving. However, we relax this by keeping *minimal* state information with timeout at strategic nodes to benefit traffic flow from the same source to the same destination. We describe this in more detail in Section 5.5.

With the above three desiderata, GeoCross bears the feature of dynamic loop detection by detecting and removing cross-links while keeping minimal routing information in nodes as it routes. Table 1 shows the comparisons between CLDP, LCR, and GeoCross.

4. Cross-link frequency

To motivate the desire to find a solution to cross-link problem in VANETS, quantifying cross-link frequency is essential. If cross-links do not occur frequently, remedies to conventional geographic routing protocols are not necessary. This section presents cross-link frequency obtained from realistic mobility traces.

Fig. 7 shows the cross-link frequency of 640 mobility traces generated by VanetMobisim [13] against the number of nodes ranging from 25 to 100 in the network. For each node, there are 40 mobility traces. Each mobility trace is 100-s long, with a warm-up period of 90 s in the beginning. The network is 1500 m by 1500 m, 5 × 5 Manhattan grid. The frequency of cross-links is calculated by the number of junctions whose intersecting roads are filled with cars yet the junctions themselves are *not* filled divided by the total number of junctions whose intersecting roads are filled with cars. Nodes that are in the 15-m radius of the junction is considered to be at the junction. The cross-link frequency is sampled every 10-s interval. The cross-link for a particular interval and a particular network size is the average of 40 mobility traces. The 11 intervals of these cross-link frequencies are then averaged together to obtained the grand average cross-link frequency for that particular network size.

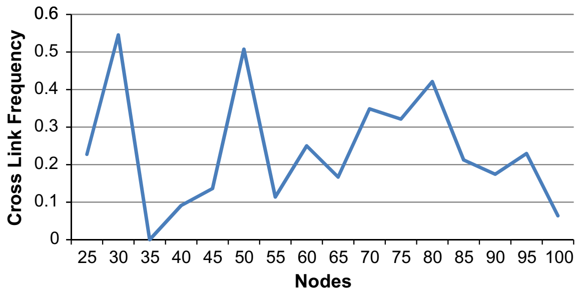


Fig. 7. Cross-link frequency vs. network size.

There are a few observations that can be made from the experiment. First, while cross-link frequency does decrease as the number of nodes increases in the network, the presence of cross-links does not disappear. Observe that there is a bottom at network size 55 where the cross-link frequency is 17%. However, at network size 80, the cross-link frequency jumps up to a whopping 42% although this is not as high as when network size is 30 or 50. So although the overall trend is downward, indicating a gradual decrease in cross-links, there is sometimes a sudden jump in the occurrence of cross-links due to the mobility of the nodes and their instantaneous placement. The other observation of high standard deviation from the average across intervals for all network sizes shows the vehicle movement greatly affects the cross-link frequency. A 10-s interval is enough to alter the network completely from a cross-link free network to a network full of cross-links. A close inspection of some of the sampled intervals reveals cross-link frequency as high as 100%. Given that the speed of packet transmission is faster than the node mobility, there might be a period of time when packets experience high cross-links that they are forced to be dropped. Redirecting the packets onto a different path in the face of cross-links by a cross-link removal algorithm suggests a better way of handling this situation than simply dropping the packets and waiting for nodes to move to a cross-link free configuration.

5. GeoCross algorithm

GeoCross is a geographic routing protocol that consists of greedy forwarding and perimeter forwarding modes. In GeoCross, we utilize street maps as natural planar graphs. The basic idea behind GeoCross is that when a node receives packets forwarded in perimeter mode, it first checks if there is a loop (i.e., the packet comes back to itself) by looking at the *Probe* field. If there is a loop, then the node further checks the probe to determine if there is any cross-link. If so, the node forwards the packet according to the loop indicated by the packet so that the detected cross can eventually be removed. If the loop has an *adjacent cross-link*, a link such that a node is on either end of the link, the node determines which cross-link is going to be removed and record the removed cross-link in *UnRoutable* (*UR*) field. Future forwarding nodes will look at the *UR* field and forward to nodes that are not on unroutable links in the *UR* field of the packet. At the same time, the original recorded loop in the *Probe* field is also truncated. If there are multiple cross-links in the loop, the cross-links will be removed one by one as the packet keeps looping back to the same node. In conclusion, as long as a packet forms a loop by coming back to any node on the previously visited road and the node detects that there is a cross-link in the loop, it will forward the packet to the same loop indicated in the packet so that the adjacent cross-link can eventually be found and removed. We describe our basic assumptions, GeoCross algorithm, GeoCross examples, and GeoCross optimizations here in this section.

5.1. Basic assumptions and terminology

GeoCross has the following definition and assumptions:

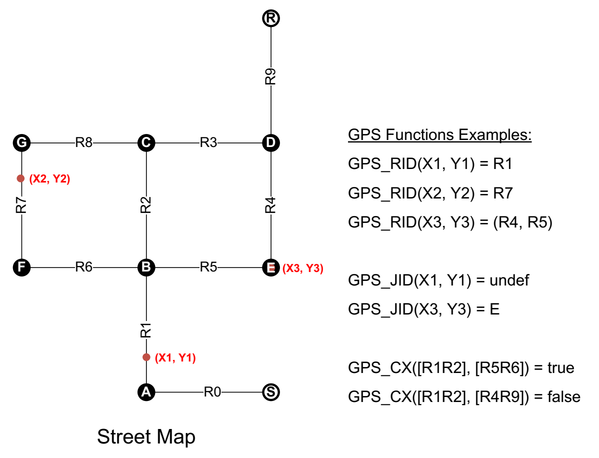


Fig. 8. GeoCross basics.

1. A road segment is defined as being formed by exactly two unique junctions. Furthermore, each road segment and junction is uniquely numbered, denoted as Road Segment ID and Junction ID. For example, in Fig. 8, 10 road segments $(R_0, R_1, \ldots, R_9)$ are connected by seven junctions $(A, B, \ldots, I)$. In reality, the same road can be separated by more than two unique junctions. Each of these segments can be given a unique identifier to conform to our definition.
2. GeoCross assumes that a GPS service is given. This service provides three primitive functions: map coordinates to road segment ids (*GPS_RID*), map coordinates to junction ids (*GPS_JID*), and check whether two pairs of a tuple of road segments are intersecting (*GPS_CX*). For example, in Fig. 8, $GPS\_RID(X_1, Y_1)$ returns R_1, $GPS\_JID(X_3, Y_3)$ returns E, and $GPS\_CX([R_1R_2], [R_5R_6])$ returns true. We believe this assumption is reasonable because more and more cars are equipped with on-board navigation systems, and navigation systems themselves should already have some internal road numbering and coordinate mapping mechanisms.
3. The paper assumes that nodes on different cross-road segments cannot detect one another because of radio obstacles such as buildings. However, if one road segment is an extension of another road segment, either horizontally or vertically, nodes may detect each other. For example, in Fig. 8, Node 1 on R_1 can hear Node 2 on R_2 because R_1 is an extension of R_2 in the vertical direction.

Terms used throughout the paper are in Table 2.

5.2. Greedy forwarding mode

In greedy forwarding mode, GeoCross would always forward packets greedily towards the junction that is closer to the destination. At junctions, a greedy decision is made to determine which neighbor brings the maximum progress towards the destination. If a local maximum is reached, the recovery mode, that is, the perimeter forwarding is used.

Table 2
Terminology.

Term	Definition
Empty junction	A junction in which no node resides
(Missing-junction) link	A link such that packets travel directly from one road segment to another because they are connected by an empty junction. It is characterized by $[R_{from}R_{to}]$
Symmetric link	A link $[R_{to}R_{from}]$ such that $[R_{from}R_{to}]$ also exists
Cross-link	A link that has been crossed by another link
Removable cross-link	A cross-link that can be removed so as not to cause network partition
Adjacent cross-link	A cross-link $[R_{from}R_{to}]$ such that a node is on either one of the adjacent roads

5.3. Perimeter forwarding mode

Having shown that empty junctions might cause cross-links that induce routing loops, we describe GeoCross implementation in perimeter forwarding to detect and remove cross-links to improve packet delivery. Unlike CLDP and LCR, GeoCross is on demand with no additional probing message, making it ideally suited for vehicular networks. GeoCross piggybacks three fields *Probe*, *UR*, and *VF* in data packets:

1. *Probe*: *Probe* records the roads and junctions that a packet has traveled. For example, if a packet has traversed Junction *A*, Road 1, Junction *B*, and Road 2, the *Probe* field would record the path as "$A,\ R_1,\ B,\ R_2$". It is important to make it clear that we only store distinct road/junction IDs instead of individual node ID. If there is more than one vehicle at the same junction, they have the same junction ID as far as GeoCross is concerned.
2. *UR* (unroutable roads): *UR* records road segments that are unroutable. The format is the same as the format of missing-junction links as $[R_{from}R_{to}]$. When nodes make a routing decision, they would ignore all unroutable road segments. This effect is, to some extent, similar to negative source routing.
3. *VF* (visited faces): *VF* records the first road that has been visited in the current face by the source node. This field makes sure that the source node visits *all* faces so as to *try* to find a complete solution. Fig. 9 illustrates a packet format.

The probe field stores the road ID, junction ID, and missing-junction link. The design choice is one of necessity although ways to reduce the probe size is discussed in Section 5.5. The reason for recording the road ID (i.e., R_1, R_2, etc.) is that non-junction nodes can detect a loop-inducing cross-link and take an earlier action. For example, as soon as the packet comes back to node *I* in Fig. 6b, node *I* can make an immediate action to forward along the same loop with cross-link $[R_5R_6]$ removed; this decision to forward along the same loop need not be made at junction *A*. The similar reason applies to recording junction ID so that nodes at a junction can detect a loop-inducing cross-link and take an earlier action. Since there can be more than one node at a junction and since a junction ID best characterizes such a node, we therefore put junction ID in the probe. The missing-junction link in the probe allows us to determine what candidate links to consider for cross-

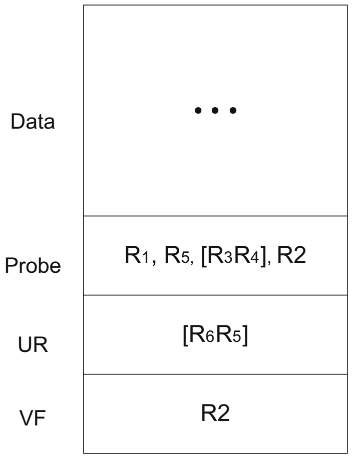


Fig. 9. Packet format.

links and for removal. Therefore, it is necessary to record it in the probe as well.

The procedure for GeoCross when a node receives a packet is listed as follows. We only outline our algorithm for perimeter forwarding:

Algorithm. PerimeterForward

```
input: currentHop, destination, Probe, UR, VF, node x that
   first enters into the perimeter mode
output: nextHop
1  if $\mathrm{DIST}(currentHop, destination) < \mathrm{DIST}(x, destination)$
2    then switch to Greedy Forwarding Mode
3      return currentHop
4  if currentHop not at junction
5    then nmax let the furthest potential forwarder
     along the current road in the same packet
     forwarding direction
6      if nmax exists
7        then return nmax
8        else mmax ← the furthest Potential Forwarder
         along the current road in the opposite
         direction to the current packet forwarding
         direction
9          return mmax
10   else PF ← ordered list of potential forwarders by
     the right-hand rule
```

```
11      $forced\_nextHop \leftarrow$ call
        $CheckProcessLoop(Probe, UR)$
12      if $forced\_nextHop$ exists
13        then $PF \leftarrow \{forced\_nextHop, PF\}$
14      for $nextHop \in PF$
15        do $roadLink$ let road of $currentHop$ to road of
           the next hop
16          $nextroad$ let the road $nextHop$ is on
17          if $(roadLink \notin UR)$ and $(nextRoad \notin VF)$
18            then return $nextHop$
19          if $currentHop$ is source node
20            then $VF \leftarrow \{VF, nextRoad\}$
21      return NULL
```

Algorithm. CheckProcessLoop

```
input: Probe, UR
output: nextHop
1  $loop \leftarrow$ a routing loop enclosed by two junction IDs
   or roads in $Probe$
2  $rclink \leftarrow$ set of cross-links in $Probe$
3  $aclink \leftarrow$ set of adjacent cross-links in $Probe$
4  if $loop$ exists
5    then if $rclink$ exists
6      then if $aclink$ exists
7        then $dlink \leftarrow$ pick one adjacent cross-link
8          $UR \leftarrow \{UR, dlink\}$
9          $Probe \leftarrow \{Probe - loop\}$
10         $forced\_nextHop \leftarrow$ furthest neighbor on the
           next road or the next junction node
           indicated in the loop
11         return $forced\_nextHop$
12     else return NULL
13   else return NULL
```

From line 2 to 3, the current node first checks if it is closer to the destination than the first node x that enters into the perimeter mode. If it is, the node switches to the greedy mode and returns itself. If it is not closer, the node first determines whether it is a junction node by the GPS function, *GPS_JID*. If it is not a junction node, according to line 4–9, it looks for *nmax* in the current forwarding direction. If it is not found, *mmax* is the furthest node in the *opposite* forwarding direction. Note that *nmax* is only up to the next road junction.

If the current node is a junction node, it first creates an ordered list *PF* of forwarders based on the right-hand rule in line 10. In other words, the node that makes the first counterclockwise turn from the edge formed by the current node and the previous node (or the virtual edge between the current node x and destination if x is the node that first enters in the perimeter mode) will be the first node in this set. The second node will be the node that makes the second counterclockwise turn, so on and so forth. Subroutine *CheckProcessLoop* is called to add *forced_nexthop* in *PF*.

In *CheckProcessLoop*, the node first looks for a probe enclosed by its junction IDs (in the form of *A*, *B*, *C*, etc.) or roads (in the form of $R1, [R_1R_2]$, etc.) in the probe in line 1. It then determines whether there are any cross-links in the probe in line 2. Next, a node can determine whether it is adjacent to the cross-link. If there is a cross-link and the node is adjacent to it, it would decide whether its adjacent link or its crossing link to be placed in *UR* in line 8. GeoCross follow CLDP in determining whether a cross-link is removable to avoid partitioning the network. In particular, GeoCross only removes the link $[R_{from}R_{to}]$ tuple only if $[R_{to}R_{from}]$, its symmetric link, does not exist within the loop. The presence of a symmetric link indicates that the link is traveled in both directions; thus, removing it can partition the network. When two separate cross-links do not have symmetric links, GeoCross removes either one of them arbitrarily. The probe is truncated by deleting the loop that is enclosed by the junction ID. *forced_nexthop* will be assigned the furthest neighbor on the road that follows the junction ID in the loop or a node at the next junction ID. *forced_nexthop* will force the packet to travel the original route so that loops as a result of cross-links can be resolved right away. This affords the packet the opportunity to get to the destination faster. Observe that it does not matter whether the cross-link is adjacent to the current node, as long as there is a cross-link in a local loop, cross-link will be identified and *forced_nexthop* will be assigned in line 11.

Back to *PerimeterForward*, lines 12–13 add *forced_nexthop* to *PF* if *forced_nexthop* exists. From lines 14–21, a candidate *nextHop* is chosen based on what is in the *PF*, which is an ordered list of nodes by the right-hand rule. If the road link formed by the current hop and the candidate nexthop is not in *UR* and the edge formed by *currentHop* and *nextHop* is not in *VF*, we can return *nextHop*. Lines 19–20 add *nextRoad* to *VF* if *currentHop* is the source node. This ensures that if the packet ever comes back to the source node, it can try to visit other faces to try to obtain a complete solution. Note that if all the faces are visited by the source node, *PerimeterForward* would return NULL. This is consistent with the condition of dropping the packets.

5.4. GeoCross routing examples

We demonstrate how GeoCross works in the following examples. Note that the main purpose of perimeter forwarding is to recover the packet from a local maximum. For simplicity, we assume all packets in our examples are forwarded in perimeter mode and the destination in our examples is along the route that does not loop back to the source.

5.4.1. Routing loop scenario

The first example investigates the same cross-link problem discussed in Section 3. In Fig. 6b, nodes A, B, C, D, E, F, G, H, R and S represent junction nodes with node A missing. Edges R_1 to R_{11} represent road segments. Since there are no cars at junction A, $[R_1R_2]$ becomes a cross-link to $[R_5R_6]$, and vice versa.

Now a packet is sent from junction S to B through road R_1 and R_2. By the right-hand rule, the packet will be forwarded to junctions C, D, E, F, and then back to B again.

The path recorded in the *Probe* is "S, I, R_1, $[R_1R_2]$, R_2, B, R_3, C, R_4, D, R_5, $[R_5R_6]$, R_6, E, R_7, F, R_8, B". Here B will detect there is a loop because its own ID appears twice in the *Probe*. So it checks its "local loop", which is "B, R_3, C, R_4, D, R_5, $[R_5R_6]$, R_6, E, R_7, F, R_8, B" and finds that there is no cross-link pair in the local loop. Thus, it simply forwards packets to the next hop by the right-hand rule. The next hop is I in this case.

Now I will also detect a routing loop and a cross-link pair $[R_1R_2]$, $[R_5R_6]$ in its local loop, "$S, I, R_1, [R_1R_2], R_2, B, R_3, C, R_4, D, R_5, [R_5R_6], R_6, E, R_7, F, R_8, B, R_2, [R_2R_1], R_1, I$". Since $[R_1R_2]$ is an adjacent link for I, it is responsible for removing either $[R_1R_2]$ or $[R_5R_6]$ and adding the offending link to *UR*. Since $[R_1R_2]$ and its symmetric link $[R_2R_1]$ appear inside the loop, removing $[R_1R_2]$ will cause a network partition. However, $[R_5R_6]$ does not have a symmetric link. Hence, it is a removable cross-link. I then places $[R_5R_6]$ in the *UR*, removes its local loop from the *Probe*, and forwards the packet to the node that is on the next road or the next junction in its loop. Note that the right-hand rule is not followed here because a removed cross-link has changed the network topology. Therefore, the packet is forwarded to this path again without the offending cross-link. The packet then goes to junction D through path "B, R_3, C, R_4, D". D finds that $[R_5R_6]$ is marked as unroutable path in the *UR* field, so it forwards the packet back to G (Fig. 6b). Now G will forward the packet to H and then to destination R.

Note that this example shows the advantage of dynamic loop detection. Before the packet loops back to S, I detects that there is a loop and reroute the packet with the cross-link removed. In fact, dynamic loop detection finds a route no slower than LCR. We give its proof below:

Property 1. *Dynamic loop detection finds a route no slower than LCR.*

Proof. We denote the first link of this loop L_0. A loop can come back to the source either through L_0 or through a link other than L_0. If the loop that comes back to the source is through L_0, there will be a loop formed by all nodes before the source. These nodes will trigger GeoCross to traverse the loop with cross-links removed. In this case, GeoCross finds a route faster than LCR because loop detection takes place at "downstream" nodes before the packet loops back to the source. If the loop that comes back to the source is not through L_0, GeoCross and LCR will take care of loop-inducing cross-links the same way. Therefore, the packet will be routed to the destination at the same time. □

5.4.2. Non-loop-inducing cross-links

We demonstrate that GeoCross can still deliver even though the current loop does not have any cross-links. In Fig. 10, node S tries to send packets to node D. The packet travels $S \rightarrow A \rightarrow B \rightarrow C$ face and records R_1 as the first edge of the current face in *VF*. When the packet loops back to S, it detects no cross-link in the current face traversal and tries to route to another face whose first edge is not in *VF*. Based on the right-hand rule, S will try to route the packet to the face whose first edge is R_2 while adding R_2 to *VF*. Since the packet is delivered to D successfully on the new face, GeoCross is never triggered.

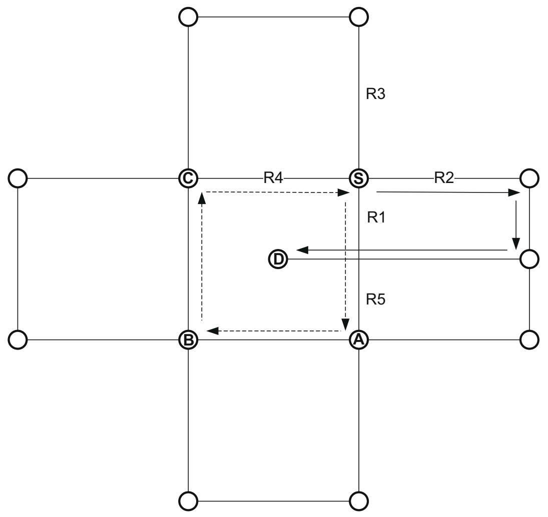


Fig. 10. The dotted arrows form the first face traversal; the solid arrows form the second face traversal.

Note that GeoCross attempts to reach a *complete* solution by visiting neighboring faces originated from the source. Completeness says that if there is a solution, the algorithm must find it. However, GeoCross is not complete because each face walk originated from the source is determined from the right-hand rule and the right-hand rule does not explore other branching links along the walk. As shown in Fig. 11, S can never detect $[R_1R_2]$ and $[R_3R_4]$ cross-links when the packet loops back. The packet never explores the branching road at node A.

GeoCross can be made complete by doing a recursive traversal similar to LCR. The idea would be to visit the neighboring faces of the current non-cross-link looping face recursively. If any one of the neighboring faces is also a non-cross-link looping face, all the neighboring links on the non-cross-link looping face will also be visited. This recursion stops until at least one face traversal reaches the destination. The destination then initiates some kind of signaling to stop all recursive packets trying to reach it to avoid flooding. The destination also notifies the source so that future packets would be sent through this route to avoid repeated recursions experienced by future packets.

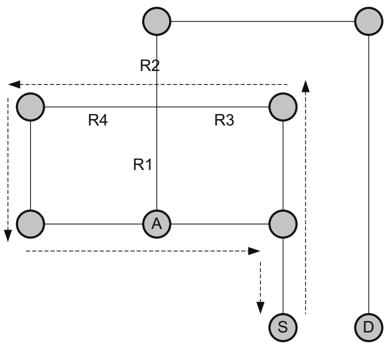


Fig. 11. S loops back, finding no cross-links. It drops the packet because there is no other face to try.

In the worst case when the destination is disconnected, the packet is duplicated at every node and the network becomes flooded.

Because of the observation that "on random graphs, we have never found the need to recurse more than one level to make progress towards the destination" by [12] and because of the dynamic nature of the VANET, the cost of a complete solution outweighs its benefit. We therefore choose to keep GeoCross light-weight and simple without pursuing a complete solution.

5.5. GeoCross optimizations

We propose two optimizations for GeoCross from our observations and analysis from the simulation runs.

5.5.1. Time optimization – by cache

In Fig. 6b, it is not too hard to see that every packet originated from S will loop back to H, get stamped with the same unroutable link in their UR, and get forwarded to the destination again without link $[R_5R_6]$. This redundancy can be eliminated by caching the unroutable link at a node (in our example, it is node H) where the link is first stamped into the UR. Future packets can then be stamped with this information in their UR from the cache a priori to avoid looping back unnecessarily. Cache is uniquely identified with each source–destination pair because the virtual edge formed by the source and destination can affect the outcome of the loop and the node at which cache is stored. In addition, to consider mobility, cache expires after a certain timeout. We implement GeoCross + CACHE and show the results in Section 6.

5.5.2. Space optimization – by recording cross-link only

We begin the motivation for space optimization by first showing that the storage overhead is on the order of n^2 given a $n \times n$ grid.

Property 2. *Given a $n \times n$ grid, where n is the number of junctions per side, GeoCross's storage overhead is $O(n^2)$.*

Proof. Consider the *Probe* field in the data packet since its cost dominates *UR* and *VF* fields. Recall that the *Probe* field stores the junction IDs, road IDs, and links. In a $n \times n$ grid, there are n junctions per row. Since there are n rows, there are n^2 junction IDs. In a $n \times n$ grid, there are $(n-1)$ roads per row and $(n-1)$ roads per column. Since there are n rows and n columns, the total number of roads is $n(n-1)+n(n-1)$ roads or $2n(n-1)$ roads. Lastly, in a $n \times n$ grid, cross-links cannot appear along the edges of the grid because links do not cross on the grid's perimeter. Therefore, it is enough to count the number of junctions *within* the grid. For each of this junction, the absence of a node produces two cross-links. Since there are $(n-2)(n-2)$ junctions within the grid, there are $2(n-2)(n-2)$ cross-links. We can sum up all these quantities and subtract the number of missing junction IDs that cause cross-links (there are $(n-1)(n-2)$ of them). Hence, the worst-case storage overhead would then be $n^2+2n(n-1)+2(n-2)^2-(n-2)^2$ or $O(n^2)$. □

The cost for storing roads, links, and junction IDs can be tremendously high for a GeoCross packet. The cost, as Property 2 shows, grows quadratically with the number of roads the packet travels in the network. One way to reduce this cost is to record only the missing junction ID and a direction. When a node forwards past a missing junction, it would record the missing junction ID and a direction in the form of a tuple (North, South) matched most closely from a predefined permutation of cardinal orientations (North, South, East, and West) and their derived orientations (NE, NW, SE, and SW). If the missing junction is passed more than once, the new direction will be appended to the existing direction. A cross-link can be recognized as one where there is more than one direction associated with the junction ID. A removable cross-link can be identified as one where there is no opposite direction; that is, the cross-link is not traveled in both directions.

When the packet loops back to the source, it can then determine the removable cross-link and place the offending cross-link in the form of junction ID and direction in the *UR* field of the packet. When the packet arrives at a node that is about to cross the same junction in the same direction as noted in the *UR* field, the node would switch to a node on a different link to avoid looping back to the source. The storage cost, in the new scheme, does not depend on the number of roads traveled but on the number of cross-links seen. However, the drawback of the new scheme is that loops can only be detected by the source, not by nodes along the way back to the source. Because nodes along the way are not able to dynamically make decisions in forwarding packets with cross-links removed, the new scheme can potentially incur longer end-to-end delay. In the new scheme, we trade time for space. We plan to verify the new scheme's correctness and compare its latency with the original GeoCross in the future.

6. Performance evaluation

In this section, we evaluate GeoCross and GeoCross + Cache by comparing them with GPSR and GPCR. Our objective is to show that GeoCross improves performance significantly compared to GPSR and GPCR. Since LCR TinyOS implementation to Qualnet is not readily available, we compare GeoCross to the *optimal* flooding protocol.

6.1. Experimental setup

We based our experiments on Qualnet simulator 3.95 with IEEE 802.11b DCF as the MAC with a transmission rate of 2 Mbps and transmission range of 250 m. We assume that nodes on different roads cannot talk to each other because of obstacles (trees, buildings, etc.) unless two roads share the same extension in either the horizontal or vertical direction. We also assume nodes lie on the same level; that is, there are no overpasses or bridges.

The traces were generated by VanetMobiSim [13], an open source and freely available realistic vehicular traffic generator for network simulators. VanetMobiSim's functionalities are decomposed into macro- and micro-mobility features of a vehicular environment to produce realistic urban mobility traces. The macro-mobility part is composed of motion constraints and a traffic generator, while the micro-mobility part controls cars' acceleration and deceleration in order to keep a safe inter-distance and avoid accidents and overlapping.

The urban topology is a realistic 1000 m by 1000 m Washington, DC map from U.S. Census Bureau's Topologically Integrated Geographic Encoding and Referencing (TIGER) database [4]. All intersections are controlled by stop signs and all road segments contain speed limitations. Unless specified differently, all roads have a single lane and a speed limit of 15 m/s (54 km/h). Finally, the micro-mobility is controlled by the IDM-IM4, an extension to the Intelligent Driver Model (IDM) considering intersections.

With the aforementioned setup, we run three experiments. The first experiment is a static scenario of 100 nodes. The placement of nodes is such that we expose routing protocols to high occurrence of cross-links. We repeat the experiment for 100 runs. Each run consists of 10 random src–dest pairs using constant bit rate (CBR), an UDP-based packet generation application.

In the second experiment, we manually insert random nodes in increment of 20's into the previous static scenario to see the effect of intersection "filling" on cross-links. We repeat this experiment 20 runs for each 20-node increment insertion for a total of 100 runs. Each run consists of 30 random src–dest pairs using CBR packet generation application.

In the third experiment, we generate 20 random layouts of cars on the Washington, DC map for node density ranging from 100 to 200 in increment of 25. Unlike the previous two experiments, these layouts do not favor one routing protocol over the other. The difference then is the impact of loop-inducing cross-links in random scenarios.

In each experiment, we compare the packet delivery ratio (PDR), hop count, and latency across four different routing protocols: GPSR, GPCR, GeoCross(+Cache), and Optimal Flooding. The comparison to the optimal flooding protocol provides a benchmark as to how far away GeoCross is in its performance to a global knowledge aware protocol.

Due to the broadcast nature of a flooding protocol, interference as a result of hidden terminal problem degrades packet delivery. To counter this problem, we take the packet delivery ratio of a pair of source and destination to be unity if *at least* one packet arrives from the source to the destination. The redefinition of the packet delivery ratio for the optimal flooding then concerns more about whether there is end-to-end connectivity than about whether the channel is reliable *in addition to* end-to-end connectivity.

6.2. Experiment results

6.2.1. Pathologic scenario

Fig. 12 shows the packet delivery ratio of GPSR, GPCR, GeoCross, GeoCross + Cache, and Optimal for each run up to 100 runs. An average delivery ratio (PDR) is obtained for the 10 random src–dest pairs per each run. A quick glance shows that GeoCross and GeoCross + Cache display a higher PDR than GPCR and GPSR since GeoCross and GeoCross + Cache remove cross-links that result in routing loops for most src–dest pairs. The oscillation explains the

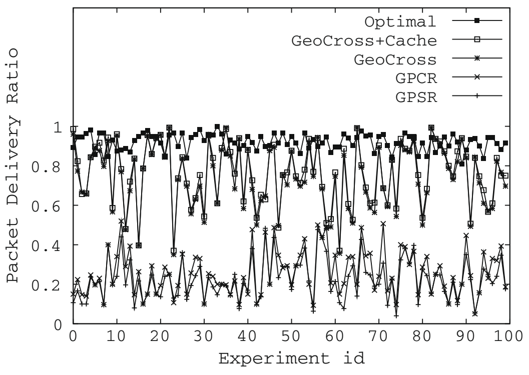


Fig. 12. PDR for pathologic scenario.

random selection of src–dest pairs. Some src–dest pairs are unreachable; therefore, it explains the low PDR for all routing protocols (for example in run 24). Some src–dest pairs are only reachable with cross-links removed; therefore, it explains the GeoCross's and GPCR's opposing heights in PDR (for example in run 10). The figure also shows that without GeoCross or GeoCross + Cache, the highest PDR that GPCR can deliver is about 52% (at run 11). For GPSR, the highest PDR is even less because of the high hop counts from planarization.

Fig. 13 shows the results of 100 runs of the experiment sorted by the delivery ratio of GeoCross in ascending order. One obvious observation is that GeoCross + Cache further improves GeoCross. The caching reduces the number of hops that each packet has to travel (evident also in Fig. 14b); therefore, it results in slightly higher PDR. The other observation is that there is a subtle downward trend (minus the outliers) of PDR for GPSR and GPCR for an upward trend of PDR for GeoCross(+Cache). The opposing, diverging trends indicate that cross-link is the culprit for low packet delivery. The trends also confirm that removal of cross-links is beneficial in improving packet delivery.

The comparison between GeoCross and Optimal Flooding in Fig. 13 indicates that there are connected source–destination pairs for which GeoCross is unable to deliver. The reason is that unlike Optimal Flooding, GeoCross explores path in a greedy way and switches to recovery when the greedy mode fails. By design, GeoCross may yield a

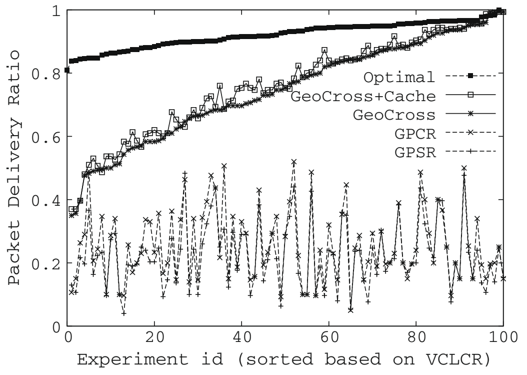


Fig. 13. PDR for pathologic scenario, sorted by GeoCross' PDR.

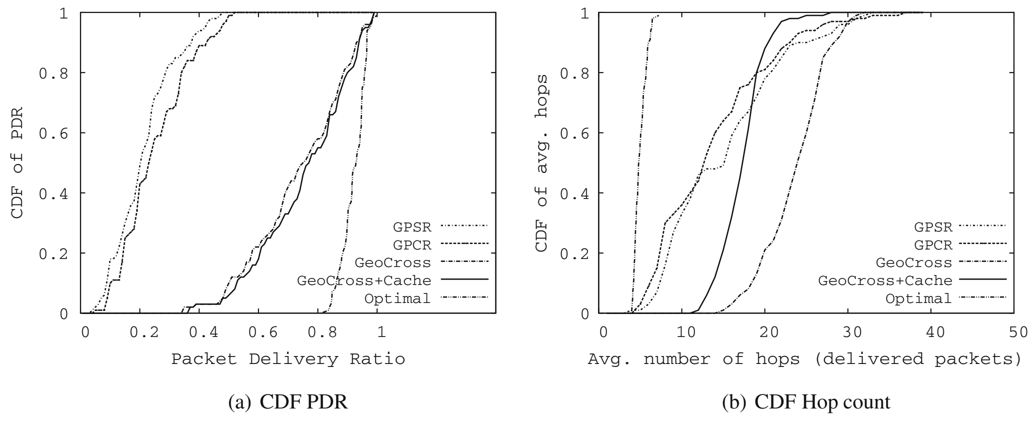


Fig. 14. Measurements for static scenario.

suboptimal path. Suboptimal path exploration gives rise to high packet hop count and consequently low packet delivery. For experiment run with ID less than 5, it can be said that there might not be cross-links at all since the difference between GeoCross and GPCR/GPSR is minimal. In conclusion, it can be deduced that there are no cross-link at all for these runs and the gap in delivery ratio between GeoCross and Optimal Flooding is due to a problem inherent to geographic routing in general. For this reason, GeoCross is comparable to LCR.

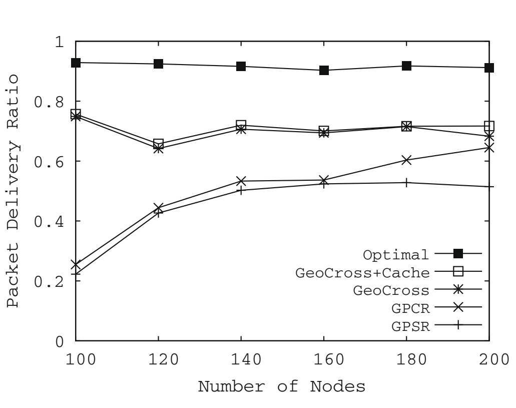
Fig. 14a and b show the cumulative distribution function (CDF) of PDR and hop count. Fig. 14a indicates that about 90% of the random src–dest pairs deliver successfully 50%+ of the time in GeoCross and GeoCross + Cache while close to 0% of the random src–dest pairs deliver successfully 50%+ of the time in GPCR and GPSR. In the comparison between GeoCross with the Optimal Flooding, only 50% of the pairs for GeoCross versus 100% of the pairs for Optimal Flooding experience 80%+ packet delivery for the same reason described above.

Fig. 14b shows the CDF of hop counts of successful packets for each protocol. At 20 hops, there are 81% of src–dest pairs for GPCR, 78% for GPSR, 90% for GeoCross + Cache, and 20% for GeoCross. GPCR's hop counts are less than GPSR's because GPCR is "greedy" in both greedy and perimeter mode. GPCR's and GPSR's hop counts are less than GeoCross + Cache's and GeoCross's because the destinations that cannot be successfully delivered by GPCR and GPSR contribute zero to the average hop count. In other words, since most src–dest pairs are not reachable because of cross-links, their contribution (0 hop count) to the average hop count only lowers it. The same src–dest pairs that are unreachable are reachable in GeoCross and GeoCross + Cache with nonzero hop count. Thus, they contribute to higher average hop count. We also see that caching dramatically improves the average hop count as 90% of src–dest pairs take 20 or less hops in GeoCross + Cache and only 20% of src–dest pairs take 20 or less hops in GeoCross. Comparing with the Optimal Flooding which possesses global knowledge and has hop count lying between 6 and 8, expectedly, GeoCross produces higher hop count because local maximum is unavoidable and the recovery mode is expensive.

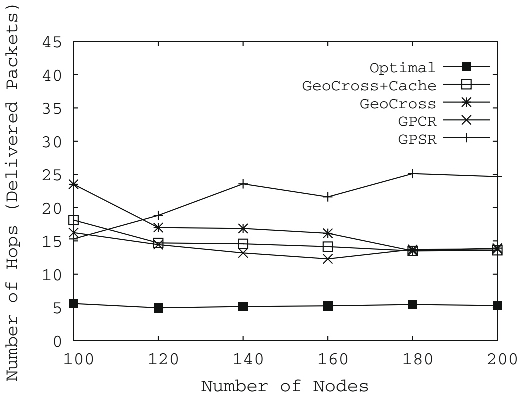
6.2.2. Pathologic scenario with node insertions

Fig. 15a–c show the PDR, hop count, and latency of the five routing protocols in the pathologic experiment where cross-links are manually introduced. Several topologies are created with nodes ranging from 100 to 200 nodes. Fig. 15a, the PDR of GeoCross and GeoCross + Cache remains steadily high and the PDR of GPSR and GPCR increases as the number of nodes increases. This outcome is expected since the denser the roads, the more likely it is that junctions are nonempty – this clearly favors GPSR. In fact, while GeoCross and GeoCross + Cache delivers packets with or without junction nodes, the presence of junction nodes greatly helps GPCR and GPSR. Fig. 15b shows the hop count of all five routing protocols. Once again, the lower hop count with 100 nodes for GPCR and GPSR than for GeoCross and GeoCross + Cache is because most src–dest pairs fail to communicate with each other in the presence of cross-links and result in zero hop count contribution. Consequently, they lower the average number of hops for GPCR and GPSR. But as the number of nodes increases, the gap becomes smaller. Starting from 180 nodes onwards, GPCR, GeoCross, and GeoCross + Cache result in the same hop count. As the number of nodes increases, however, GPSR becomes increasingly inefficient. Node planarization prevents packets from making large "strides" to the destinations. Once again, we see the advantage of caching unroutable links to reduce the average hop count.

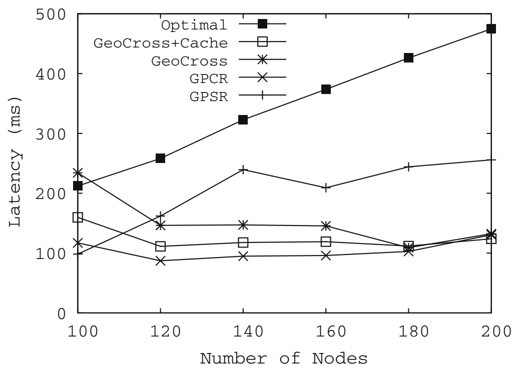
Fig. 15c shows the latency of all five routing protocols. As the number of nodes increases, latency of Optimal Flooding and GPSR increases. The latency of Optimal Flooding increases because broadcasting is not so scalable. Where there is an increasing number of nodes in the network, there is a broadcast storm. Broadcast from many nodes interfere with one another, causing many broadcast to be dropped. Thus, the end-to-end latency increases with increasing number of nodes. As for GPSR, when packets are in perimeter mode, they make many small steps towards the destination. Thus as the number of nodes increases, the latency increases, the increasing hop count also reflects this. For GeoCross(+Cache) and GPCR, the increasing number of nodes increases the connectivity to the destination. Because of their large strides towards the destination in



(a) PDR



(b) Hop count



(c) Latency

Fig. 15. Measurements for pathologic scenarios.

the perimeter mode, they all benefit from the increasing node density. As a result, their latency decreases.

6.2.3. Random scenarios

Fig. 16 shows the PDR of GPCR, GeoCross, and GeoCross + Cache in scenarios where nodes are placed randomly on roads from 100 nodes to 200 nodes. PDR for each routing protocol is averaged over 20 runs. We observe that the PDR is extremely low (in the Optimal Flooding, the highest PDR of 33% occurs in node density 200) even though nodes should be well connected in a small topology of 1000 m by 1000 m. However, in reality, the connectivity between nodes is almost non-existent for two reasons. First, nodes on different roads cannot communicate with each other because of building obstacles. The only nodes that can hear nodes from neighboring roads are junction nodes. Second, nodes that happen to lie in a circle of some radius (25 m) to be categorized as junction nodes are too few. Without these nodes, packets cannot make transition from one road segment to another. Most of the packets that are delivered are ones that either src–dest pairs lie on the same road segment or src–dest pairs are connected with intermediate junction nodes. Given that most of the nodes are disconnected, the PDR is therefore low.

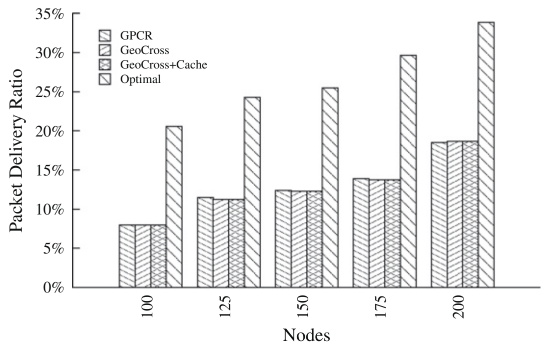


Fig. 16. PDR for random scenarios.

While the random scenarios do not provide evidence that GeoCross benefits packet delivery in general, they do not discredit it, either. It is for a fact that once a packet gets into a loop in perimeter mode, GPSR and GPCR will give up and drop the packet but GeoCross will *try* to deliver it. Since most of these random graphs are not connected, all of the routing protocols yield such a poor performance. For the ones that are connected, however, cross-link is not the cause for low packet delivery. The cause for low packet delivery is more intrinsically related to geographic routing that produces higher hop count in general than to environmental pathology of cross-links. A more meaningful comparison would then be to generate a *connected* graph which these routing protocols run on. Such a connected graph needs to be created carefully so that it makes perimeter mode forwarding more frequent than greedy mode forwarding. Moreover, it should consider obstacles and takes into account of the notion of junction nodes. We plan to study the effect of cross-link removal in different densities of VANET *connected* graphs.

Another future work is to incorporate GeoCross with some Delay Tolerant Network (DTN) routing protocol to take care of intermittent connectivity. A proposal is to look into a hybrid approach that switches between Geographic routing assisted by GeoCross and DTN routing based on some scoring function. The scoring function helps to determine the "connectedness" of the network and allows the hybrid routing protocol to act appropriately. Because of the dynamic nature of VANETs, a flexible routing protocol is thus needed to improve the end-to-end performance.

7. Conclusion

In this paper, we propose GeoCross, a simple, lightweight, yet novel, event-driven geographic routing protocol that removes cross-links dynamically to avoid routing loops in urban Vehicular Ad Hoc Networks (VANETs). GeoCross' salient feature of dynamic loop detection makes its suitable for highly mobile VANET. Simulations using realistic city map with building blocking model have shown that in pathologic cases, GeoCross's packet delivery ratio (PDR) is consistently higher than GPSR's and GPCR's. We also demonstrate that it is possible to reduce hop count and improves the PDR further by caching an unroutable link in nodes where it is first determined. Finally, we propose a scheme to compress GeoCross packet efficiently by recording only a vector that contains the cross-link with its direction. Our future work includes implementing GeoCross with space optimization, evaluating GeoCross and other geographic routing protocols on connected VANET graphs, comparing LCR and GeoCross in VANETs, and incorporating GeoCross to some Delay Tolerant Network approach for intermittent connectivity.

Acknowledgements

We express our deepest gratitude to Jui-Ting Weng and Lung-Chih Tung for their initial contribution to the paper. Without their help, the paper would not have been possible.

References

[1] 802.11p. <http://grouper.ieee.org/groups/802/11/Reports/tgp_update.htm>.
[2] dash. <http://www.dash.net/>.
[3] ITS Standards. <http://www.standards.its.dot.gov/about.asp>.
[4] Tiger, tiger/line and tiger-related products. U.S. Census Bureau. <http://www.census.gov/geo/www/tiger/>.
[5] P. Bose, P. Morin, I. Stojmenovic, J. Urrutia, Routing with guaranteed delivery in ad hoc wireless networks, Wireless Networks 7 (6) (2001) 609–616.
[6] P. Bose, P. Morin, I. Stojmenović, J. Urrutia, Routing with guaranteed delivery in ad hoc wireless networks, Wireless Networks 7 (6) (2001) 609–616.
[7] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, L. Viennot, Optimized link state routing protocol for ad hoc networks, in: Multi Topic Conference, IEEE INMIC 2001. Technology for the 21st Century, Proceedings IEEE International, 2001, pp. 62–68.
[8] R. Flury, R. Wattenhofer, MLS: an efficient location service for mobile ad hoc networks, in: MobiHoc '06: Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing, New York, NY, USA, 2006, pp. 226–237.
[9] H. Füßler, M. Käsemann, M. Mauve, H. Hartenstein, J. Widmer, Contention-based forwarding for mobile ad-hoc networks, Ad Hoc Networks 1 (4) (2003) 351–369.
[10] H. Füßler, H. Hartenstein, M. Mauve, W. Effelsberg, J. Widmer, Contention-based forwarding for street scenarios, in: 1st International Workshop in Intelligent Transportation (WIT 2004), Hamburg, Germany.
[11] K.R. Gabriel, R. Sokal, A new statistical approach to geographic variation analysis, Systematic Zoology 18 (1969) 231–268.
[12] Y.-J.K.R. Govindan, B. Karp, S. Shenker, Lazy cross-link removal for geographic routing, in: SenSys '06: Proceedings of the 4th International Conference on Embedded Networked Sensor Systems, New York, NY, USA, 2006, pp. 112–124.
[13] J. Härri, F. Filali, C. Bonnet, M. Fiore, Vanetmobisim: generating realistic mobility patterns for VANETs, in: VANET '06: Proceedings of the 3rd International Workshop on Vehicular Ad Hoc Networks, New York, NY, USA, 2006, pp. 96–97.
[14] H. Hartenstein, B. Bochow, A. Ebner, M. Lott, M. Radimirsch, D. Vollmer, Position-aware ad hoc wireless networks for inter-vehicle communications: the fleetnet project, in: MobiHoc '01: Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing, New York, NY, USA, 2001, pp. 259–262.
[15] D.B. Johnson, D.A. Maltz, Dynamic source routing in ad hoc wireless networks, in: Imielinski, Korth (Eds.), Mobile Computing, vol. 353, Kluwer Academic Publishers, 1996.
[16] R.G.K. Seada, A. Helmy, On the effect of localization errors on geographic face routing on sensor networks, in: Proceedings of the Third IEEE International Workshop on Information Processing in Sensor Networks, 2004.
[17] B. Karp, Challenges in geographic routing: sparse networks, obstacles, and traffic provisioning, in: Presentation at DIMACS Workshop on Pervasive Networking, Piscataway, NJ, May 2001.
[18] B. Karp, Geographic Routing for Wireless Networks, PhD thesis, Harvard University, 2000.
[19] B. Karp, H.T. Kung, GPSR: greedy perimeter stateless routing for wireless networks, in: Mobile Computing and Networking, 2000, pp. 243–254.
[20] Y.-J. Kim, R. Govindan, B. Karp, S. Shenker, Geographic routing made practical, in: NSDI'05: Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation, Berkeley, CA, USA, USENIX Association, 2005, pp. 217–230.
[21] Y.-J. Kim, R. Govindan, B. Karp, S. Shenker, On the pitfalls of geographic face routing, in: DIALM-POMC '05: Proceedings of the 2005 Joint Workshop on Foundations of Mobile Computing, New York, NY, USA, 2005, pp. 34–43.
[22] E. Kranakis, H. Singh, J. Urrutia, Compass routing on geometric networks, in: Proceedings of the 11th Canadian Conference on Computational Geometry, Vancouver, August 1999, pp. 51–54.
[23] F. Kuhn, R. Wattenhofer, Y. Zhang, A. Zollinger, Geometric ad-hoc routing: of theory and practice, in: PODC '03: Proceedings of the Twenty-second Annual Symposium on Principles of Distributed Computing, New York, NY, USA, 2003, pp. 63–72.
[24] K.C. Lee, J. Haerri, U. Lee, M. Gerla, Enhanced perimeter routing for geographic forwarding protocols in urban vehicular scenarios, 2007 IEEE Globecom Workshops, 26–30 November 2007, pp. 1–10.
[25] J. Li, J. Jannotti, D.S.J.D. Couto, D.R. Karger, R. Morris, A scalable location service for geographic ad hoc routing, in: MobiCom '00: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, New York, NY, USA, 2000, pp. 120–130.
[26] C. Lochert, H. Hartenstein, J. Tian, H. Fussler, D. Hermann, M. Mauve, A routing strategy for vehicular ad hoc networks in city environments, in: Proceedings of IEEE Intelligent Vehicles Symposium (IV2003), 9–11 June 2003, pp. 156–161.
[27] C. Lochert, M. Mauve, H. Füssler, H. Hartenstein, Geographic routing in city scenarios, SIGMOBILE Mobile Computing and Communications Review 9 (1) (2005) 69–72.
[28] V. Naumov, R. Baumann, T. Gross, An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces, in: MobiHoc '06: Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing, New York, NY, USA, 2006, pp. 108–119.
[29] G. Pei, M. Gerla, X. Hong, Lanmar: landmark routing for large scale wireless ad hoc networks with group mobility, 2000.
[30] G. Pei, M. Gerla, X. Hong, C.-C. Chiang, Wireless hierarchical routing protocol with group mobility (WHIRL), Technical Report 990006, UCLA, 25, 1999.
[31] C.E. Perkins, E.M. Royer, Ad hoc on-demand distance vector routing, in: Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, February 2005, pp. 90–100.

[32] G. Toussaint, The relative neighborhood graph of a finite planar set, Pattern Recognition 12 (1980) 231–268.
[33] Y. Yu, G.-H. Lu, Z.-L. Zhang, Enhancing location service scalability with high-grade, in: 2004 IEEE International Conference on Mobile Ad-hoc and Sensor Systems, 25–27 October 2004, pp. 164–173.
[34] James Bernsen, D. Manivannan, Review: unicast routing protocols for vehicular ad hoc networks: a critical comparison and classification, Pervasive and Mobile Computing 5 (1) (2009).

Kevin C. Lee is currently a Ph.D. student in the Computer Science Department at the University of California, Los Angeles. He received B.S. in Computer Science Engineering and B.A. in Mathematics at the University of Pennsylvania in 2002. He also received M.S. in Computer Science at the Carnegie Mellon University in 2004. Kevin has published several papers in the fields of vehicular ad hoc networks ranging from developing and running a peer-to-peer application on the real vehicular testbed, designing an optimized geographic routing protocol in urban scenarios, developing a light-weight loop-free geographic routing protocol, and proposing link-state routing based on density. He has also written technical reports in network process migration in the area of network system programming and in classification in the area of artificial intelligence. His current research interests include vehicular ad hoc network routing, theoretical analysis of computer networks, WiMax networks, QoS, and mobile applications for vehicular ad hoc networks.

Pei-chun Cheng has been pursuing the Ph.D. in Computer Science at University of California, Los Angeles, since 2007. He received his B.S. and M.S. degrees from National Taiwan University in 2000 and 2002. His research interests includes Internet topology, next generation routing architectures and vehicular ad hoc network routing protocols.

Mario Gerla, Professor, UCLA, Computer Science Dept. Dr. Gerla received his Engineering degree from the Politecnico di Milano, Italy, in 1966 and the M.S. and Ph.D. degrees from UCLA in 1970 and 1973. He became IEEE Fellow in 2002. At UCLA, he was part of a small team that developed the early ARPANET protocols under the guidance of Prof. Leonard Kleinrock. He worked at Network Analysis Corporation, New York, from 1973 to 1976, transferring the ARPANET technology to several Government and Commercial Networks. He joined the Faculty of the Computer Science Department at UCLA in 1976, where he is now Professor. At UCLA he has designed and implemented some of the most popular and cited network protocols for ad hoc wireless networks including distributed clustering, multicast (ODMRP and CODECast) and transport (TCP Westwood) under DARPA and NSF grants. He has lead the $12M, 6-year ONR MINUTEMAN project, designing the next generation scalable airborne Internet for tactical and homeland defense scenarios. He is now leading two advanced wireless network projects under ARMY and IBM funding. In the commercial network scenario, with NSF and Industry sponsorship, he has led the development of vehicular communications for safe navigation, urban sensing and location awareness. A parallel research activity covers personal P2P communications including cooperative, networked medical monitoring (see www.cs.ucla.edu/NRL for recent publications).