Enhanced Perimeter Routing for Geographic Forwarding Protocols in Urban Vehicular Scenarios*

Kevin C. Lee[†], Jérôme Häerri[‡], Uichin Lee[†], Mario Gerla[†]

University of California[†]
Department of Computer Science
Los Angeles, CA 90095
{kclee,uclee,gerla}@cs.ucla.edu

University of Karlsruhe[‡]
Institute for Telematics
76131 Karlsruhe, Germany
haerri@tm.uni-karlsruhe.de

Abstract—Geographic stateless routing schemes such as GPSR have been widely adopted to routing in Vehicular Ad Hoc Networks (VANET). However, due to the particular urban topology and the non-uniform distribution of cars, the greedy routing mode often fails and needs a recovery strategy such as GPSR's perimeter mode to deliver data successfully to the destination. It has been shown that the cost of planarization, the non-uniform distribution of cars, and radio obstacles make GPSR's perimeter mode inefficient in urban configurations. Some enhancements have been proposed such as GPCR, which uses the concept of junction nodes to control the next road segments that packets should follow. However, the concept of junction nodes itself is problematic and hard to maintain in a dynamic urban environment.

In this paper, we describe GpsrJ+, a solution that further improves the packet delivery ratio of GPCR with minimal modification by predicting on which road segment its neighboring junction node will forward packets to. GpsrJ+ differs from GPCR as decisions about which road segment to turn does not need to be made by junction nodes. Moreover, GpsrJ+ does not need an expensive planarization strategy since it uses the natural planar feature of urban maps. Consequently, GpsrJ+ reduces the hop count used in the perimeter mode by as much as 200% compared to GPSR. It therefore allows geographic routing schemes to return to the greedy mode faster.

I. INTRODUCTION

With the sharp increase of personal and sport utility vehicles in the recent years, driving has not stopped from being more challenging and dangerous. Roads are saturated, safety distance and reasonable speeds are hardly respected, and drivers often lack enough attention. Without a clear signal of improvement in the near future, leading car manufacturers decided to jointly work with national government agencies in order to develop solutions aimed at helping drivers on the roads by anticipating hazardous events or avoiding bad traffic areas. One of the outcomes has been a novel type of wireless access called Wireless Access for Vehicular Environment (WAVE) [1] dedicated to vehicle-to-vehicle and vehicle-to-roadside communications. While the major objective has clearly been to improve the overall safety of vehicular traffic, promising traffic management solutions or on-board entertainment applications are also expected by the different bodies involved in this field (C2CCC [2], VII [3], CALM [4]).

When equipped with WAVE communication devices, cars and roadside units form a highly dynamic network called Vehicular Ad Hoc Network (VANET). While some safety scenarios mostly need point-to-point connectivity, it is expected from most of the scenarios developed for intelligent transportation systems (ITS) to benefit from a multi-hop connectivity. Current state of the art solution developed for Mobile Ad Hoc Networks (MANET) such as OLSR [5] or AODV [6] happen to be highly inappropriate for VANET due a specific mobility and higher velocity. However, location-based stateless routing schemes such as Greedy Perimeter Stateless Routing (GPSR) [7] are particularly efficient on highly dynamic networks. Using geo-localization information, packets are greedily forwarded to the vehicle bringing the maximum progress towards the destination. In some cases, there might not be such a vehicle and a recovery strategy, called *perimeter routing* in GPSR, is used to find an appropriate next relay.

The perimeter mode, and more generally any recovery strategy involved in geographic routing, guarantees a correct delivery if and only if the vehicular graph is planar. Significant research [8]–[12] has therefore been focused on building a planar vehicular graph, or to develop completely different forwarding schemes that do not need planar graphs in the recovery mode [13]. While these schemes work in general, they often involve significant overheads, which reduce the efficiency of location-based stateless routing schemes. Yet, by observing that urban maps naturally form a planar graph, planarization strategies are actually not necessary in VANETs. As it is expected from "VANET-ready" cars to be equipped with on-board topological maps, the use of maps in routing in VANET is as reasonable as the availability of GPS devices. In Lochert *et al.*'s GPCR [24], packets are simply forwarded on successive adjacent roads in the recovery mode until a new node bringing progress is found. However, critical decisions on the next direction to follow need to be made by vehicles located at intersections[1], as it is the only place where the general direction of the recovery mode on urban maps may be

[1]Vehicles located at intersections will be thereafter called "junction nodes."

changed. Junction nodes are accordingly critical to GPCR and missing a junction node will jeopardize packets' successful delivery.

In this paper, we propose to remove unnecessary junction-node awareness while keeping the efficient planarity of topological maps. The new schema which we call GpsrJ+, is a simple and intuitive scheme that does not need to be aware of junction nodes and predicts which road segment its neighboring junction node will take. By bypassing junction nodes to avoid wrong decisions, GpsrJ+ manages to increase packet delivery ratio of GPCR further and reduce the number of hops before returning to greedy mode by 200% compared to GPSR.

The rest of this paper is organized as follows: Section II provides a short background on GPSR and GPCR and more specifically on the recovery mode and the right hand rule. In Section III, we formally introduce the GpsrJ+ protocol and in Section IV, we provides simulation results. Finally, Section V describes other related work, while Section VI summarizes our contribution and presents future works.

II. BACKGROUND

In this section, we provide a short description of the Greedy Perimeter Stateless Routing (GPSR) algorithm and the Greedy Perimeter Coordinator Routing (GPCR) algorithm.

A. Greedy Perimeter Stateless Routing

The Greedy Perimeter Stateless Routing (GPSR) algorithm belongs to the category of position-based routing, where an intermediate node forwards a packet to an immediate neighbor which is geographically closer to the destination node. This approach is called greedy forwarding. For that matter, each node needs to be aware of its own position, the position of its neighbors as well as the position of the destination node. How positions are obtained or shared is outside the scope of this paper. We assume that each node is able to obtain its own position using GPS devices, exchange it with neighboring nodes by beacon messages and obtain the position of the destination node by a separate location service (see [16] for a summary of location services).

As position-based routing schemes are based on local information only, and due to non uniform distributions of nodes or to the existence of radio obstacles, it is possible that a packet reaches a local maximum with respect to the distance to the destination. In other words, a node cannot find a potential forwarder that is closer to the destination than itself (see Fig. 1). In order to escape from this 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 will be forwarded backward with respect to its distance to the destination until it reaches a node whose distance to the destination is closer and greedy mode may be resumed.

Many recovery algorithms have been developed including GPSR [7], Compass [14], Face-1 and Face-2 [8], or GOAFR+ [15]. GPSR recovers from a local maximum using
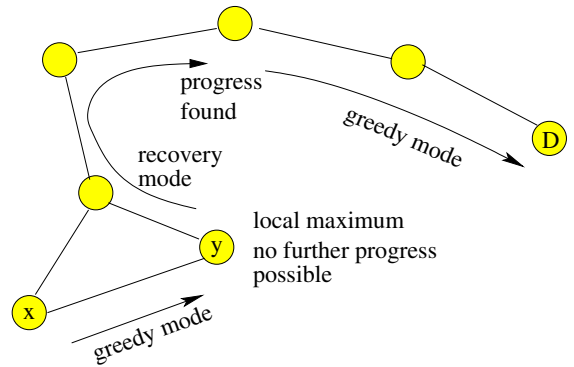

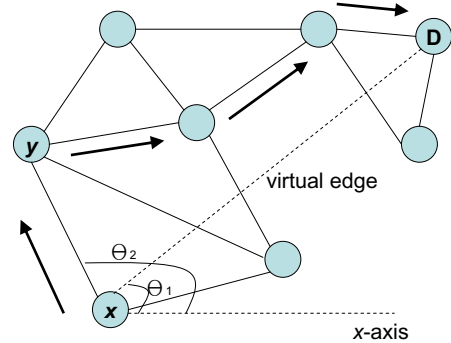Fig. 1. GPSR two-mode routing in presence of a local maximum.

Fig. 2. Routing by right-hand rule.

a *Perimeter mode*, where the *right-hand rule* (Fig. 2) is used. This rule states that when a node x first enters into the recovery mode, its next forwarding hop is the node that is sequentially counterclockwise to the virtual edge[2] formed by x and destination D. Afterwards, the next hop is sequentially counterclockwise to the edge formed by y and its previous node x (See [7] for more details). For obvious reasons, the right-hand rule requires that all edges are non-crossing. GPSR proposes either Relative Neighborhood Graph(RNG) [17] or Gabriel Graph(GG) [18] to get a planar network graph with no crossing edges, while another approach suggests the use of spanning trees or convex hulls [19]. The maintenance of planar graphs at each node introduces a significant overhead. While all nodes need to maintain the planar graph all the time, this information is only used by nodes facing the local minimum phenomenon. From this point of view, maintaining a planar graph for the recovery mode makes recovery modes more statefull than stateless.

In this paper, we relax the need for planarization by observing that we may extract a planar graph from an urban map at no extra cost. Moreover, we implemented the right-hand rule by comparing the angle θ_1 between the x-axis and the edge formed by the current node and its previous node (or destination depending on whether it first enters or continues in the perimeter mode) and the angle θ_2 between the x-axis and the edge formed by the current node and its neighbors. We choose the smallest angle θ_2 that is greater than θ_1. The neighbor forming θ_2 with the current node will be chosen to

---

[2]An edge is here defined as a wireless bi-directional link between two neighbor nodes. A virtual edge is a geometric construction connecting two nodes that are not in connection range
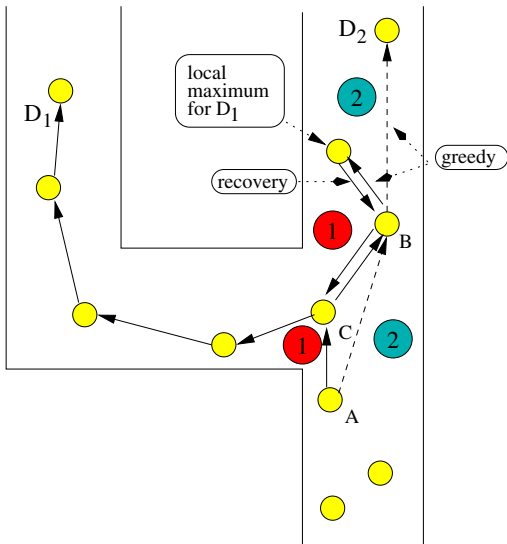
Fig. 3. Routing with GPCR and the critical junction node (coordinator); circle 1 indicates case 1 and circle 2 indicates case 2.

be the next hop.

B. Greedy Perimeter Coordinator Routing algorithm

Greedy Perimeter Coordinator Routing algorithm (GPCR) is an enhancement of the GPSR protocol. It is also based on the fact that streets and junctions naturally form a planar graph and thus does not require any planarization algorithm. Moreover, GPCR does not need an urban map. As GPSR, it contains two phases: a restricted greedy forwarding and a recovery phase. The restricted greedy forwarding part only uses nodes on the same road segment as potential relays, as building and other obstacles block radio signal between adjacent road segments. An important point is that, since junctions are the only places where routing decisions are made, packet must always be sent to a node that is at a junction. Forwarding a packet across a junction risks to bring GPCR to a local maximum. At junctions, a greedy decision is also made, and the neighboring node which brings the maximum progress towards the destination is chosen. If a local maximum is reached, the recovery mode is used.

When GPCR is in recovery mode, packets are backtracked in a greedy fashion (i.e. bringing maximum progress) to a junction node in order to find an alternate solution to return to the greedy mode. At the junction node, the right-hand rule is used to find the next road segment to forward the packets.

The major weak points of GPCR are threefold. First, junction nodes need to be determined and advertised, which might bring some overhead to the protocol. Second, recognizing a junction node, which is faulty in GPCR, is extremely crucial to avoid local maximums and consequent hop reduction. Third, even if the junction node detection algorithm is perfect, forwarding to a node at a junction is often unnecessary and counter-productive as most of junctions are not critical[3]. As shown in Fig. 3, where the destination node is D_1 and the routing starts from A, if a packet faces a critical junction

---

[3]A critical junction is a junction which leads to a local maximum if the greedy decision is maintained.

and that the protocol fails to provide a valid junction node, GPCR will surely encounter a local maximum (see case 1 in Fig. 3). Moreover, when a packet does not face a critical junction (the destination node is D_2 in Fig. 3), routing to a junction node is counter-productive as crossing the junction to the relay bringing the maximum distance would have been preferred (see case 2 in Fig. 3). In that perspective, it would be better if the observation of a critical junction be made by nodes before the junction. That is precisely what we propose to do in this paper with GpsrJ+.

III. GPSRJ+

GpsrJ+ is a position-based routing protocol which consists of two modes, yet using a special form of greedy forwarding. As obstacles (e.g., buildings) block radio signals, packets may only be greedily forwarded along road segments as close to the destination as possible. Accordingly, the major directional decisions are made at junctions. When packets reach a local maximum, a point at which there is no node closer to the destination, the node switches to GpsrJ+'s recovery mode.

In the recovery mode, packets are greedily backtracked along the perimeter of roads. It is not necessary to backforward in small steps through planarized links, first because the general direction of the right-hand rule always results in the opposite direction of where packets were going before recovery, and second because the objective is to come back as fast as possible to a junction. Unlike GPCR, where packets must be sent to a junction node since junction nodes coordinate the next forwarding direction, GpsrJ+ lets nodes that have junction nodes as their neighbors predict on which road segment its junction nodes would forward packets onto, and thus may safely overpass them if not needed. The prediction is based on the fact that the forwarding node knows all road segments on which its junction neighbors have neighbors. The road segment, on which neighbor nodes are, is extracted from the urban map using the neighbor's location. Finally, nodes incorporate this information in the modified beacon (see Section III-B) and broadcast it to the forwarding node that carries out the prediction.

If the resulting next hop is on a road segment that shares the same x or y coordinate as the coordinates of forwarding node's junction neighbors, the forwarding node can simply forward the packets to such a next hop and may save one hop. However, if the resulting hop is on a road segment that does not share the same x and y coordinates as coordinates of forwarding node's junction neighbors, the forwarding node's next hop must be its junction neighbor. In summary, GpsrJ+ further enhances GPCR by taking fewer hops to the destination, while keeping the same route traversal and the same high delivery ratio as GPCR over GPSR.

A. Basic Assumptions

Before we describe GpsrJ+ formally, we make four assumptions. First, we assume a road segment to be an edge formed by two points, intersecting or not. For example, in Fig. 4, two edges that overlap each other form four segments with five
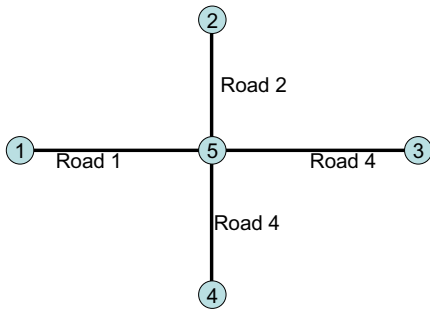
Fig. 4. Five nodes with four road segments.

Fig. 5. Enhanced beacon format and modified neighbor list.

points, one of which is an intersecting point. This assumption naturally gives us a planar graph out of a city map.

Second, similar to [24], we assume that nodes on different road segments cannot detect one another because of radio obstacles. However, if one road segment is an extension, either horizontally or vertically, of another road segment, nodes may detect each other.

Third, unlike the geo-routing protocol presented by [24], we assume the map of a city is given. We believe this is a reasonable assumption as more and more cars are equipped with an on-board navigation system. Consequently, each node knows its location and the road segment it is on. Each node also knows whether it is a junction node. Lastly, in the scope of this work, we assume that the city map is in the form of grids, that is, there are no turning or diagonal straight roads for the prediction to make sense. When there are turning or diagonal straight roads, packets must go through junction nodes. Therefore, no prediction can take place at all. We argue that this assumption does not limit GpsrJ+ to grid scenarios which do happen frequently in cities, but rather complement GPCR when grid scenarios happen. We yet plan to extend GpsrJ+ to handle such configuration in future work.

B. Enhanced Beacon

In addition to node's position in the beacon, each node also broadcasts the road segments that its neighbors are on. Since each node is equipped with a navigation system, it is easy to extract the road segments on which its neighbors are, given their locations. In the neighbor list, each node therefore has its neighbor's location and the associated road segments on which its neighbor's neighbors are. The size of the enhanced beacon is bounded by the number of roads a junction node can have. In a grid network, this is at most four. In most of the city scenarios, this number is also trivially small. As GPCR also needs to transmit a flag bit per intersection node, the size of the enhanced beacon is only increased by 1 bit in Grid areas compared to GPCR. The storage cost of each node is each one hop neighbor's location and its two-hop neighbors' road IDs. The cost is the same as before because the number of roads stored is bounded by a constant.

Note that nodes on the same road segment, such as Node A in Fig. 5, will broadcast the same road ID. Nodes that are near the junction, such as Node B in the same figure, will only broadcast its road ID, 2 (not all the other roads that the junction neighbor also shares on) for its junction neighbor. The
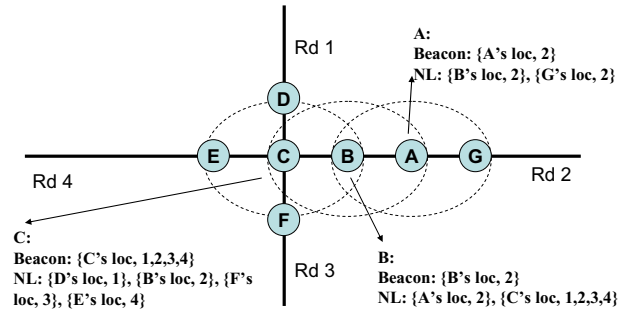
nodes that are at the junction, such as Node C, will broadcast their locations and the road segment IDs on which they have neighbors. Fig. 5 shows the beacon and the neighboring table of Node A, B, and C.

C. Repaired Strategy

In GPCR, when a packet hits a local maximum, the corresponding node switches to recovery mode and greedily backwards the packet to the closest junction. In [23], [24], authors make a good observation that the only direction a packet may take is the reversed direction as far as its radio range allows in a multi-hop fashion along a rectilinear road segment. According to [23], [24], since a packet can possibly take a new direction if it is forwarded to a junction node, the packet has to be forwarded to such a node. The reason that a node in a junction can forward the packet onto a different road segment is it receives beacons from nodes in the other segments. By the right-hand rule that makes sequentially counterclockwise turn with respect to the line formed by the current node and the junction node, it can yield a road segment that is not the extension of the road segment that the current node is on. If the packet is not forwarded to the junction node, it will not be routed to a different road segment. Consequently, the route traversed in the recovery mode will not be the same as the route in GPSR's perimeter mode. Therefore, there is no guarantee that the packet will eventually reach the destination if it is connected with the rest of nodes.

GpsrJ+ takes note of the fact that packets do not necessarily have to go through the junction nodes that might forward packets to the same node as non-junction nodes would. In other words, there are cases where a node can forward a packet passed the junction node and arrives at a node A as it were forwarded to the junction node which eventually forwards to A. We first describe the six cases where the recovery mode can possibly occur, and explain when forwarding from the junction node and non-junction node results in packet's being in the same node. We argue that these cases are exhaustive because there is no other way that a packet can be forwarded while it is in perimeter mode. We then describe the GpsrJ+ algorithm in the next section. We suppose that the destination node is Node D and a packet has arrived in Node A in greedy mode, hit a local maximum, and then switched to the recovery mode. Node A can choose to forward to either Node B or Node C. We indicate the forwarding direction by an arrow and the radio range of A by a dotted oval.
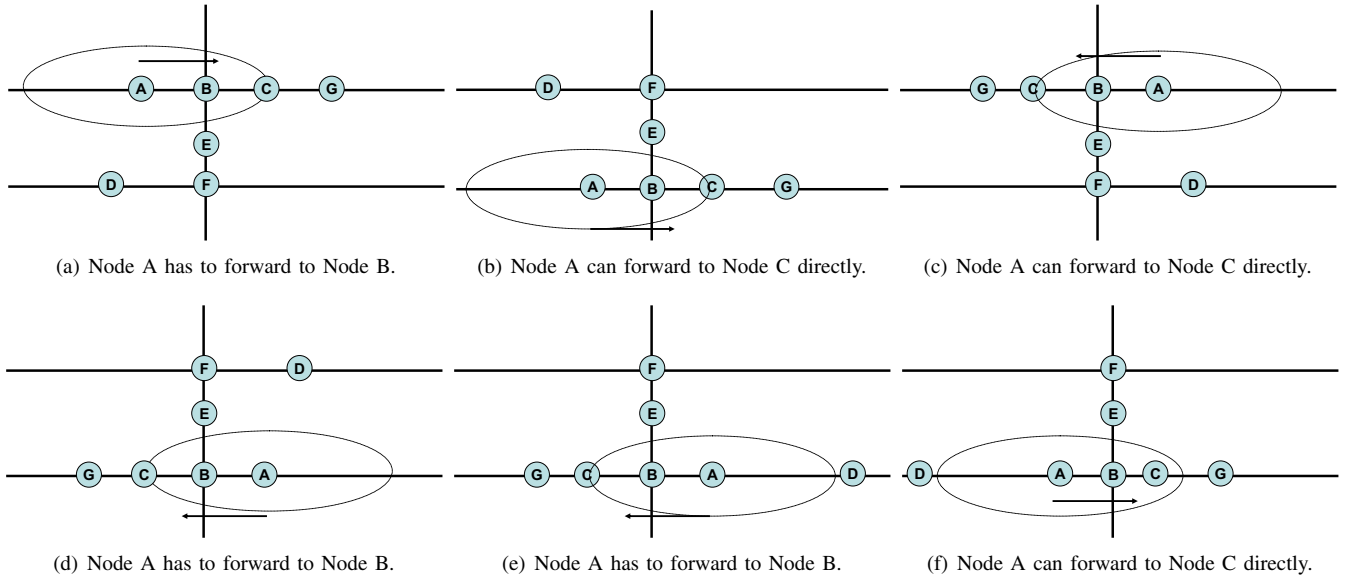
(a) Node A has to forward to Node B.   (b) Node A can forward to Node C directly.   (c) Node A can forward to Node C directly.

(d) Node A has to forward to Node B.   (e) Node A has to forward to Node B.   (f) Node A can forward to Node C directly.

Fig. 6. Six possible cases.

In Fig. 6(a), if Node A forwards to Node B, Node B will forward to Node E and Node E will switch to greedy mode. In this scenario, Node A has to forward the packet to Node B or the packet gets forwarded all the way to the east end of the axis. By missing junction Node E, we are not following the route dictated by perimeter routing in GPSR. Therefore, it is not guaranteed that the packet will be delivered to the destination.

In Fig. 6(b), if Node A forwards to Node B, Node B will forward to Node C. Hence, Node A can safely forward the packet to Node C without passing through Node B. Unlike the previous scenario, we save ourselves one hop by directly forwarding to Node C.

If Node A forwards to Node B, Node B will forward to Node C in Fig. 6(c). Hence, Node A can safely forward the packet to Node C without passing through Node B. Similar to the previous scenario, we save ourselves one hop by directly forwarding to Node C.

If Node A forwards to Node B in Fig. 6(d), Node B will forward to Node E and Node E will switch to greedy mode. In this scenario, Node A has to forward the packet to Node B or the packet gets forwarded all the way to the west end of the axis. By missing junction Node E, we are not following the route dictated by perimeter routing in GPSR. Therefore, it is not guaranteed that the packet will be delivered to the destination.

In Fig. 6(e), the packet reaches Node A and cannot be forwarded further to Node D since it is outside of Node A's radio range. If Node A forwards to Node B, the packet will then be forwarded to Node E. But if it is forwarded to Node C, it will be forwarded to Node F and continue toward the west end of the axis. Therefore, in this scenario, the packet has to be forwarded to Node B.

In Fig. 6(f), similar to the previous scenario, the packet reaches Node A and cannot be forwarded further to Node D

since it is outside of Node A's radio range. Node A can safely forward to Node C because what has been forwarded to Node B will eventually be forwarded to Node C according to the right-hand rule.

D. GpsrJ+ Algorithm

Three out of six scenarios above need not be forwarded to a junction node in the recovery mode for the forwarding decision to be made. GpsrJ+ exploits the fact that a node can determine whether it should forward to its furthest neighbor or junction neighbor by noticing the road segment its junction neighbors' neighbors are on. In other words, each node will beacon its coordinates and road segments that its neighbors are on. In Fig. 7, for example, Node A has Node B and Node C as its neighbors. Node B will beacon its location as well as Road1, Road2, Road3 to Node A. Similarly, Node C will beacon its location as well as Road2 to Node A. Based on the road segments it receives from its neighbors, in particular its junction neighbors, Node A pre-computes the forwarding segment that its next hop is on from the perspective of Node B according to the right-hand rule. If the pre-computed forwarding segment is the same as the road segment that Node A's furthest neighbor is on, Node A will forward to its furthest neighbor C; otherwise, it will forward to its junction neighbor B. Since GpsrJ+ predicts where a packet will be forwarded ahead of time in the vicinity of a junction node, it will always stay ahead of GPCR in its route traversal. Since both output the same route traversal as GPSR, packets will be delivered successfully to the destination if it is connected.

We list the major steps of GpsrJ+'s recovery mode:

1) Find all potential forwarders, PF.
2) Compute the node with maximum progress, $nmax$.
3) If the set PF does not contain any junction node, forward to the node with maximum progress, $nmax$.
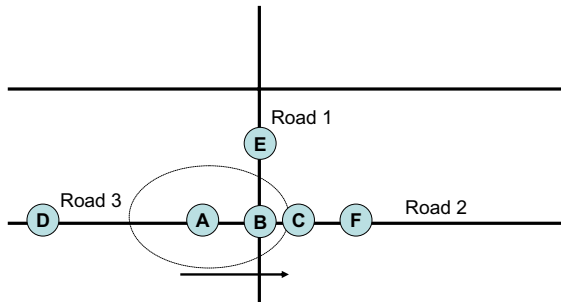
Fig. 7. GpsrJ+ example; Node A will forward to Node C directly.

Return $nmax$.

4) If the set PF contains a junction node, compute road segment center and choose the one according to right-hand rule with respect to the current node's location.

5) If the road segment center does not have the same x and y coordinates as coordinates of $nmax$, forward to the junction node.

6) Else, forward to $nmax$ (save one hop).

The list omits computation of L_p and $e0$ which are the node first entering the recovery mode and the first edge traversed on the current face, respectively, in GPSR. Step 1 finds the neighbors that satisfy the right-hand rule with respect to the line formed by the current and the previous node. In Step 2, we forward to the node with the maximum progress if none of these potential forwards is a junction node. Based on the aforementioned assumption of a map and definition of a beacon, a node knows whether its neighbor is a junction node or not simply by checking if its x and y coordinates are within a certain width from the center of a junction. If there is more than one junction node, we randomly pick one. In Step 3, we then compute the road segment that the next hop is on by the right-hand rule from the junction node to each road segment's center. We pick the road segment center that satisfies the right-hand rule first. In other words, we choose the the road segment center such that it forms the smallest angle θ_2 with the junction node and θ_2 is bigger than the angle θ_1 formed by the current node and the junction node (consistent to our implementation of the right-hand rule describe in Section II-A). Road segment centers can be computed by realizing the coordinates of the road segment's two ends from the map. If the road segment center has the same x or y coordinate as coordinates of the node with the maximum progress, forward to such a node, otherwise, forward to the junction node.

In the case where there is no junction node, GpsrJ+ forwarding behavior is the same as GPCR's. But in a city environment where there are traffic lights and roads with high and low traffic volume, the enhancement of GpsrJ+ can be noticeable.

E. Special Cases

We need to take care of two special cases that are likely to occur. The first case requires us to compare the distance of L_p (the location where the node first entered the perimeter node; see [7] for details) and the furthest node. If the distance

of L_p to the destination is greater than the distance of the junction node to the destination, GpsrJ+ will forward to the junction node; otherwise, it will forward to the furthest node. The reason to forward to the junction node if the distance of L_p to the destination is greater than the distance of the junction node to the destination is that when the packet arrives at the junction node, it will switch to greedy mode as it makes progress towards the destination. If the node can reach the destination directly, it will cost us one hop to forward to the furthest node.

Consider Fig. 8(a) where Node A tries to forward to Node D. Node A hits a local maximum, switches to perimeter mode, and forwards to Node B. Assume that Node B can hear Node C and E; and Node C can hear Node D. At Node B, GpsrJ+ will forward to E. However, if the packet is forwarded to C, Node C will switch to greedy mode because its distance to Node D is smaller than the distance (L_p) between Node A and Node D. In order to save us one hop in this special case, we will compare the distance between Node D and Node F to the distance (L_p) between Node D and Node A. If it is smaller, Node F will be the next hop; otherwise, Node C will be the next hop.

The second special case happens when a node's radio range covers more than one road segment (Fig. 8(b)). The furthest node by the right-hand rule should not exceed such a road segment. If it does, the current node is forwarding way too far. The resulting path reversal will not yield the same traversal as GPSR.

Consider the following example in Fig. 8(b). Node A hits a local maximum as it tries to forward to D. It then switches to the recovery mode and can either forward to its furthest neighbor F or its junction node B. According to GpsrJ+, since junction node B does not have neighbors on either of its two vertical segments Road 1 and Road 2, it will forward to node F. By forwarding to F, however, it misses the junction node C and its next hop E. As a result, it generates a different path traversal from GPCR.

The problem here is that GpsrJ+ considers A's furthest node F beyond A's second closest junction node C (A's first closest junction node is B). At A, had GpsrJ+ considered the furthest node from itself up to its second closest junction node, it would have forwarded correctly to E. Therefore, the node with the maximum progress should be up to and including the current node's second closest junction node. If the second closest node does not exist, it is simply the furthest node.

A future work is to further optimize GpsrJ+ by considering junction nodes over more than one road segment. The multiple-road segment prediction is very practical in an urban environment where there are lots of branching roads from the main road segment and these intersections that lead to branching roads are all within the radio range of a forwarding node. We summarize the major steps of GpsrJ+'s recovery mode with consideration of these two special cases:

1) Find all potential forwarders, PF.

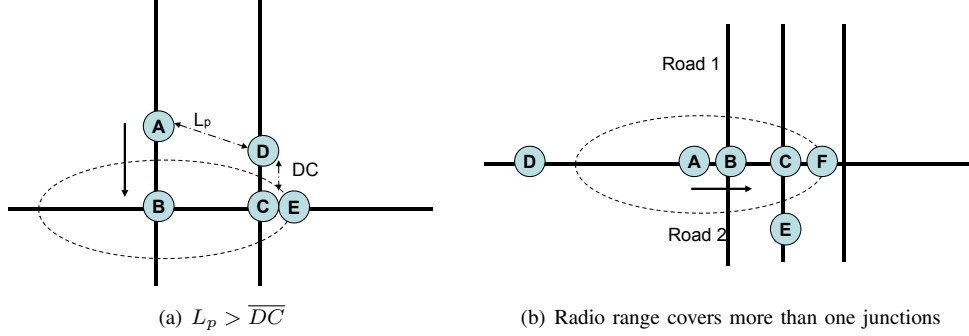2) Compute the node with maximum progress $nmax$ subject to the rule that it does not exceed the second unique

(a) $L_p > \overline{DC}$ (b) Radio range covers more than one junctions

Fig. 8. Two special cases

junction from the first one closest to the current node.

3) If the set PF does not contain any junction node, forward to the node with maximum progress, $nmax$. Return $nmax$.

4) If the set PF contains a junction node, compute road segment center and choose the one according to right-hand rule with respect to the current node's location.

5) If the road segment center does not have the same x and y coordinates as coordinates of $nmax$, forward to the junction node.

6) Else:
   a) If `Distance`(junction node, destination) < `Distance`(L_p, destination), forward to junction nodeforward to nmax (save one hop).
   b) Else, forward to $nmax$ (save one hop).

IV. PERFORMANCE EVALUATION

In this section, we evaluate GpsrJ+ by comparing it with GPSR and GPCR. Our objective is to show that GpsrJ+ improved recovery strategy brings significant results compared to the benchmark GPSR and also to GPCR, yet without the cost of computing and maintaining junction nodes. First we describe our experimental setup and then provide simulation results.

A. Experimental Setup

We based our simulations on Qualnet 2.95 with 75 nodes up to 175 nodes, with a 25-node increment. We use IEEE 802.11b DCF as the MAC with a transmission rate of 2Mbps and transmission range of 371m. The default transmission range yields on average 2 to 5 neighbors for non-junction nodes and 4 to 10 neighbors for junction nodes for node density between 75 and 175. These settings well guarantee a fully connected network.

The mobility traces were generated by VanetMobiSim [27], 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.
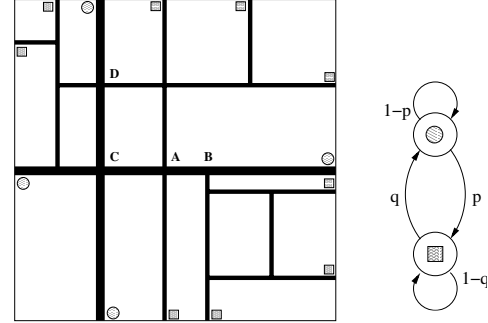

Fig. 9. City section map and activity chain.

The urban topology employed in this paper is an user-defined Manhattan-grid of 1500m by 1500m as illustrated in Fig. 9. 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), except for the roads represented with thicker lines, which allow a maximum speed of 20 m/s (72 km/h). Vehicles travel between entry/exit points at borders, identified with circles and squares, crossing the city section according to the fastest path to their destination.

The trips generation scheme is activity-based and the relative transition probability matrix describes a simple activity chain depicted in Fig. 9. As also shown in Fig. 9, the states denote the class of the selected destination: a round for the entry/exit points of high-speed roads, a square for the entry/exit points of normal-speed roads. The paths between entry/exit points are computed based on a speed-based shortest path cost function. Finally, the micro-mobility is controlled by the IDM-IM[4], an extension to the Intelligent Driver Model (IDM) considering intersections. We have set the different IDM-IM parameters to suitable real world values (see Table I).

Param	Description	Value
a	Maximum Comfortable Acceleration	$0.9m/s^2$
b	Maximum Comfortable Deceleration	$0.5m/s^2$
l	Vehicle Length	5m
s_{com}	Minimum Congestion Distance	2m
t	Safe headway time	1.5s
b_{sav}	Maximum safety deceleration	$4m/s^2$

TABLE I
VEHICULAR MICRO-MOBILITY MODEL PARAMETERS

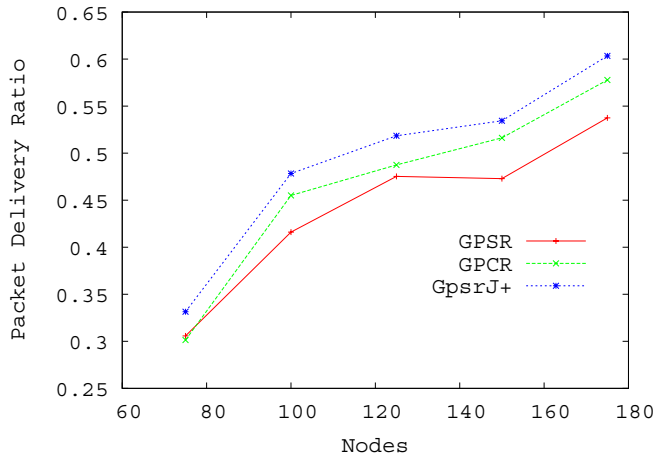[4]Intelligent Driver Model with Intersection Management

Fig. 10. PDR among GPSR, GPCR, and GpsrJ+

We collected mobility traces of 75, 100, 125, 150, and 175 nodes, each of them considered at steady state. For each node density, there were 10 simulation runs with different sets of 5 random source-destination pairs that used CBR for data traffic generation. In the next section, we show performance evaluation metrics of the delivery ratio and hop count. For a realistic radio propagation emulation, blocking radio obstacles have been added between different road segments if they do not share the same horizontal or vertical coordinates.

B. Experiment Results

A quick glance at the results shows that, as the density of the network increases and the network becomes more connected, the likelihood that a random pair of source and destination reaches each other increases. This explains the upward trend of PDR shown in Fig. 10. Since nodes are more connected, the frequency of getting into a local maximum decreases. Most of the packets can be forwarded in greedy mode. Unlike the recovery mode where packets are forwarded along a perimeter, the greedy mode takes fewer hops. The downward trend in hop count for a packet is observed in Fig. 11(a) and the corresponding downward trend in packet's end-to-end latency is observed in Fig. 11(c). Fig. 12 shows the fraction of times a packet is in greedy mode and perimeter mode, respectively. When nodes' reachability improves due to the increasing density, the greedy mode becomes sufficient to take care of most of source-destination pairs. Therefore, the greedy mode increases with the density. Correspondingly, the fraction of times a packet is in perimeter mode decreases.

In GPSR, we notice a spike in hop count at node density 125. We have repeated 10 more runs of 125 nodes with different sets of 5 random source-destination pairs, each using different seeds. However, the spike still remains. We reason that the increase is due to the fact that most source-destination pairs are still not connected because of a lack of enough junction nodes guiding packets to the destination. As a result, the perimeter forwarding still dominates the greedy forwarding, and the increase in the number of nodes reduces the hop distance and increases the number of hops. This explains the sudden increase in hop count. As more nodes are introduced,

they become connected and the greedy forwarding gradually dominates the perimeter forwarding. Since most forwarding is done in greedy, it explains the sudden decrease in hop count. The disadvantage of planarization of nodes that reduces hop distance and thus increase hop count is clearly shown. In contrast to GPSR's hop count, GpsrJ+'s hop count is not affected by the increasing density of the network. The hop count does not fluctuate much and stays lower between 6 and 8.

Figure 10 shows the packet delivery ratio (PDR) between GPSR, GPCR, and GpsrJ+. Clearly, taking aggressive hops in the recovery mode along the perimeter improves the PDR. This is further verified by the fewer hops GpsrJ+ needs compared to GPSR as shown in Fig. 11(a). A higher number of hops implies an increased probability of channel contention; therefore, there is a higher probability that a packet gets dropped along the way. Although GPCR and GpsrJ+ stop at each junction node in greedy mode, this is not sufficient to increase the hop count dramatically. The total hop count of GPCR and GpsrJ+ is still lower than that of GPSR. At node 125 for example, the hop count of GPSR (16.93) is twice as high as that of GpsrJ+ (8.20). Fig. 11(b) shows the number of hops a packet experiences before being dropped. GPSR's failed hop is twice as much as GPCR and GpsrJ+. This is consistent with what has just been mentioned that planarization of nodes produces too many hops. The undeliverable packets, as a result of disconnections between the source and destination, engage in perimeter forwarding most of the time and explore all possible perimeters in a limited way caused by planarization. Since more nodes are involved in forwarding, there is a lot of resource wastage. The situation worsens for undeliverable packets as they create a loop and the same route formed by the same nodes in the same many hops is visited again. In summary, the inefficiency of node planarization strategies in urban vehicular scenarios to forward packets in perimeter mode not only affects the delivery ratio, it also impacts the hop count and network resources as packets stay longer in the network before being dropped.

Figure 10 also shows that GpsrJ+ possesses a higher PDR than GPCR thanks to prediction. The slight increase in hop count and latency in Fig. 11(a) and Fig. 11(c), respectively, is the result of packets that do *not* get delivered to the destination and thus do *not* contribute to GPCR's hop count and latency. We reason that the smoother decrease in hop count in GpsrJ+ compared to GPCR is due to the fact that nodes do *not* necessarily have to go through junction nodes, which might be heavily used for forwarding in GPCR. Consequently, the interference and collision of multiple packet transmission cause packets either to be dropped or to be forwarded on a longer route. The slight increase in failed hops in GpsrJ+ compared to GPCR in Fig. 11(b) illustrates a longer expectancy of packets as GpsrJ+ makes a better effort to deliver them. Once again, the ability not to rely on junction nodes that get flooded with traffic prolongs the life expectancy of a packet before it gets dropped. The improved PDR in GpsrJ+ also brings in the advantage of the fraction of times a packet travels

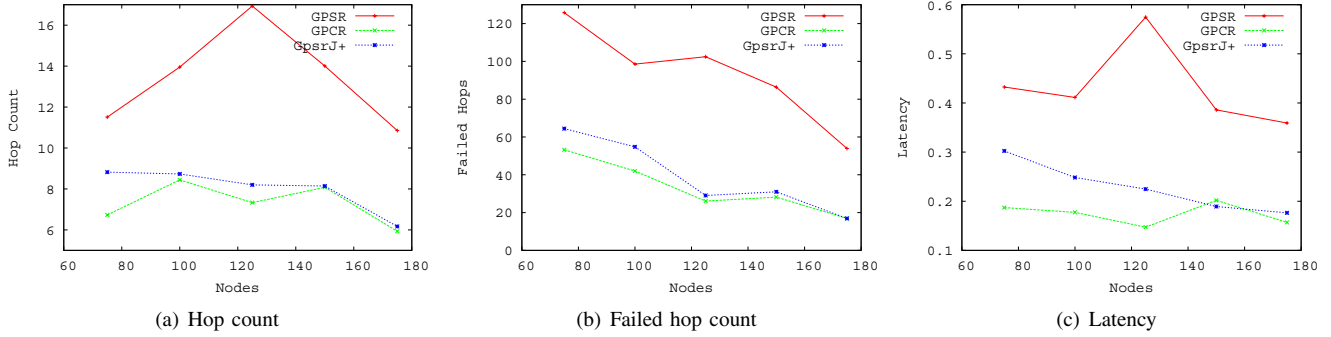(a) Hop count   (b) Failed hop count   (c) Latency

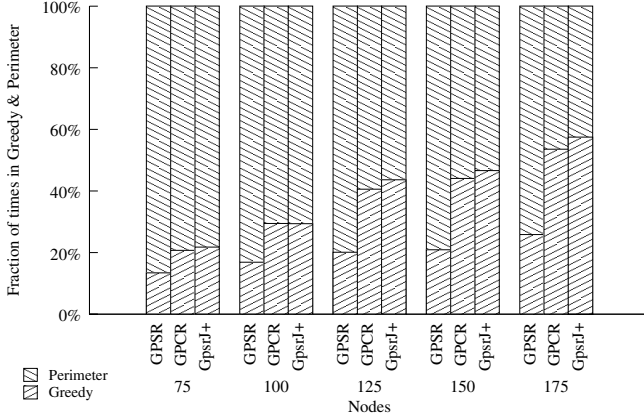Fig. 11. Hop count and latency among GPSR, GPCR, and GpsrJ+

Fig. 12. Fraction of times in Greedy and Perimeter.

in greedy mode. Fig. 12 indicates that GpsrJ+ is in greedy mode a higher fraction of time than GPCR, and implies that GpsrJ+ minimizes times a packet gets into a local maximum and maximizes times a packet gets out of a local maximum.

V. RELATED WORK

Naumov *et al.* [20] incorporated a velocity vector of speed and direction in the beacon to improve GPSR by accurately determining the location of a destination. The velocity vector information relies on the location service which lets each node know the other nodes' moving directions and speeds. It also relies on packets' being stamped with travel time. By the velocity vector and the packets' travel time, each node can approximate the distance and the direction of the destination and choose the best next forwarding hop. The approach is tightly coupled with the location service. Thus, there is too much overhead in updating and querying the velocity vector. A longer period of updating the velocity vector from the location can make the routing as inaccurate as GPSR without such an optimization.

Naumov *et al.* [21] also introduced AODV with preferred group broadcasting (PGB) that reduces control message overhead by adaptive beaconing based on the number of nearby neighbors. They also introduce the concept of "guards" to track the position of a destination. A node that is marked as a guard can redirect the packet or add information to the packet to be delivered to the destination. CAR is a non-positioned-based routing that provides a scalable low overhead and locates destinations without a location service.

Blazevic *et al.* [22] talked about using anchors to guide geodesic packet forwarding as a replacement to perimeter forwarding in the recovery mode of a position-based routing. However, finding anchors is usually expensive. Since destinations change frequently in a highly mobile network, anchors need to be computed dynamically before routing. The overhead associated with determining anchors dynamically is RREQ flooding to find DSR-like paths to the destination. In an urban environment, we believe a more "light-weight" approach like GpsrJ+ is enough to forward around voids.

Lochert *et al.* [23] described reactive location service (RLS) to learn the current location of a destination. RLS is coupled with Geographic Source Routing (GSR), a position-based routing presented in the work. The accurate position of a destination is supposed to improve the delivery ratio of GSR. However, RLS cannot avoid the 'broadcast storm problem' without a specific treatment of it. Unlike our work, they seek to improve their routing protocol by introducing location service. We seek to improve routing by focusing on how to improve the existing position-based protocol.

Lochert *et al.* [24] proposed GPCR, a solution that does not rely on planarization of nodes by taking note of the fact that an urban map naturally forms a planar graph. In the recovery mode, nodes simply forward along a road in the direction produced by the right-hand rule [7] until hitting nodes at a junction. The junction nodes will determine the next hop and the new direction of the perimeter to forward packets to. We enhance GPCR by arguing that nodes do not necessarily need to stop at each junction node. GpsrJ+ improves GPCR further in delivery ratio and hop count. Since a city map is given, GpsrJ+ does not depend on the heuristics of low density in a junction to determine junction nodes.

At the same time as [24], another approach, called Context-Aware Routing (CAR), has been proposed by Dumitrescu *et al.* [25]. CAR is a two phases scheme, which first uses global context information such as road topology, traffic information or roadside access points to compute the least-cost routing trajectory. The objective is to proactively avoid to reach a local maximum. Then, in the second step, a packet is forwarded with the least deviation from the intended trajectory, yet with intermediate nodes using local context information, such as location, speed and driving direction of neighboring cars, to improve the greedy mode. No recovery mode has been proposed in this work. As this work only improves

the greedy phase of GPSR, it is significantly different from the objective of this work. However, CAR's improved greedy phase and GpsrJ+ enhanced recovery phase could be jointly tested in order to evaluate the general improvement of this joint approach.

Ma *et al.* [26] presented a path pruning algorithm that exploits the channel listening capability to reduce the number of hops in perimeter mode. Each node keeps states of what its next hop and the packet's current hop count are. If the node hears the same packet transmitted with a bigger hop count from its neighbor, it recognizes a shortcut from such a neighbor and transmits future packets instead direclty to it. Since the approach requires a planarization of nodes, it differs from GPCR and GpsrJ+ that use the underlying map as the planar graph. In GPCR and GpsrJ+, the perimeter forwarding is greedy (i.e., the furthest node that is along the road segment) in between junctions. Because of the greedy approach in the perimeter mode, there are not so many redundant paths that can be pruned. Near junctions, GpsrJ+ further determines whether it can pass through its junction neighbor if it is not at a critical junction. The hop saving at junction nodes happens in the first pass in GpsrJ+ as opposed to the second pass in [26]. Furthermore, unlike the path pruning algorithm, GpsrJ+ does not require each node to keep states. Since [26] requires that each not listen all the time, it might affect nodes in transmitting data concurrently.

VI. CONCLUSION

In this paper, we have introduced GpsrJ+ as an intuitive predictive scheme that improves the recovery strategy of geographic forwarding algorithms. Unlike GPSR, it does not require an expensive planarization strategy and makes more efficient routing decisions at road junctions. Unlike GPCR, GpsrJ+ only forwards packets to nodes in road junctions if and only if the forwarding decision changes with respect to the general forwarding direction of the recovery mode; otherwise, packets are allowed to progress across the intersection with the maximum progress, saving the protocol many hops. The prediction requires little modification to GPCR. We have shown that GpsrJ+ improves GPCR and GPSR in packet delivery ratio, and improves the hop count of GPSR by as much as 200%.

Future works encompass three areas. First, we plan to extend GpsrJ+ so that it can make prediction beyond one junction. We indeed believe multiple-junction prediction in GpsrJ+ is practical as urban environments have lots of branching roads. Second, to further improve delivery ratio, we plan to incorporate prediction in the greedy mode as well as in the perimeter mode by determining whether the furthest node of the junction node lies on the same road segment as the furthest node of the current forwarding node. Third, we plan to run GpsrJ+ on realistic city maps that are not necessarily grids. Realistic roads follow a more complex trajectory; thus, a simple line trajectory equation will not be sufficient to define a road segment.

REFERENCES

[1] "Wireless Acess for Vehicular Environment (WAVE)," http://www.standards.its.dot.gov/fact_sheet.asp?f=80
[2] "Car to Car Communication Consortium (C2CCC)," http://www.car-to-car.org
[3] "The Vehicle Infrastructure Integration (VII) Initiative," http://www.vehicle-infrastructure.org
[4] "Continous Air Interface for Long and Medium Interface (CALM)," http://www.calm.hu
[5] IETF, "Optimized Link State Routing," RFC 3626, http://www.ietf.org/rfc/rfc3626.txt
[6] IETF, "Ad Hoc On demand Distance Vector Routing," RFC 3561, http://www.ietf.org/rfc/rfc3561.txt
[7] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in Proc. ACM/IEEE MOBICOM00, Boston, Massachusetts, USA, 2000, pp. 243254.
[8] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," in Proc. ACM DIAL M99, Aug 1999, pp. 4855.
[9] Y.-J. Kim, R. Govindan, B. Karp and S. Shenker, "Geographic Routing Made Practical," Proceedings of the 2nd NSDI 2005, Boston, MA.
[10] Kim, Y. J., Govindan, R., KARP, B., Shenker, S.,"On the Pitfalls of Geographic Routing," in Proc. of the 3rd International Workshop on DIALM-POMC (Cologne, Germany, September 2005).
[11] H. Frey , I. Stojmenovic, "On delivery guarantees of face and combined greedy-face routing in ad hoc and sensor networks," Proceedings of the 12th annual international conference on Mobile computing and networking, September 23-29, 2006, Los Angeles, CA, USA.
[12] Y. Kim, R. Govindan, B. Karp, S. Shenker, "Lazy cross-link removal for geographic routing," Proceedings of the 4th international conference on Embedded networked sensor systems, October 31-November 03, 2006, Boulder, Colorado, USA.
[13] B. Leong, B. Liskov, R. Morris, "Geographic routing without planarization," NSDI'06: Proceedings of the 3rd conference on 3rd Symposium on Networked Systems Design & Implementation, 2006, San Jose, CA, USA.
[14] Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia, "Compass routing on geometric networks," in *Proceedings of the 11th Canadian Conference on Computational Geometry*, pp. 51-54, 1999.
[15] Fabian Kuhn, Roger Wattenhofer, Yan Zhong, and Aaron Zollinger. "Geometric Ad-hoc Routing: Of Theory and Practice," *In 23rd ACM Symposium on Principles of Distributed Computing (PODC'03)*, July 2003.
[16] T. Camp, J. Boleng, J. Wilcox, "Location Information Services in mobile Ad Hoc Networks," in *Proc. of the IEEE International Conference on Communications (ICC)*, pp. 3318-3324, New York, 2002.
[17] G. Toussaint. "The Relative Neighborhood Graph of a Finite Planar Set," in *Pattern Recognition 12*, Vol. 4, pp. 261-268, 1980.
[18] K.R. Gabriel and R. R. Sokal, "A new statistical approach to geographic variation analysis," in *Systematic Zoology*, vol. 18, pp. 259-278, 1969.
[19] B. Leong, B. Liskov, R. Morris, "Geographic routing without planarization" in *Proc. of the 3rd Conference on 3rd Symposium on Networked Systems Design and Implementation*, San Jose, CA, May 2006.
[20] V. Naumov, R. Baumann, and T. Gross, "An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces," in Proc. ACM MOBIHOC06, 2006, pp. 108119.
[21] V. Naumov, T. Gross, "Connectivity-Aware Routing (CAR) in Vehicular Ad Hoc Networks," INFOCOM, May 2007.
[22] L. Blazevic, S. Giordano, and J.-Y. L. Boudec, "Self organized terminode routing," Cluster Computing, vol. 5, no. 2, pp. 205218, 2002.
[23] C. Lochert, H. Hartenstein, J. Tian, H. Fussler, D. Hermann, and M. Mauve, "A routing strategy for vehicular ad hoc networks in city environments," in Proc. IEEE IV03, Jun 2003, pp. 156161.
[24] C. Lochert, M. Mauve, H. Fussler, and H. Hartenstein, "Geographic routing in city scenarios," ACM SIGMOBILE05, vol. 9, no. 1, pp. 69 72, 2005.
[25] V. Dumitrescu and J. Guo, "Context Assisted Routing Protocols for Inter-Vehicle Wireless Communication," in *Proc. of the 2005 IEEE Intelligent Vehicles Symposium (IV'05)*, pp. 594-600, June 2005.
[26] X. Ma, M. Sun, X. Liu, G. Zhao, "Improving Geographical Routing for Wireless Networks with an Efficient Path Pruning Algorithm," 3rd Annual IEEE SECON, 2006.
[27] J. Härri, M. Fiore, F. Filali, and C. Bonnet, "VanetMobiSim: generating realistic mobility patterns for VANETs," Proceedings of the 3rd international workshop on vehicular ad hoc networks, 2006, Los Angeles, CA, USA