GeoDTN+Nav: A Hybrid Geographic and DTN Routing with Navigation Assistance in Urban Vehicular Networks[*]

P.-C. Cheng[‡], J.-T. Weng[‡], L.-C. Tung[‡], K. C. Lee[‡], M. Gerla[‡], J. Härri[†]

Computer Science Department[‡]
University of California, Los Angeles
Los Angeles, CA 90095
{pccheng, jtweng, lctung, kclee, gerla}@cs.ucla.edu

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

ABSTRACT

Position-based routing has proven to be well suited for highly dynamic environment such as Vehicular Ad Hoc Networks (VANET) due to its simplicity. Greedy Perimeter Stateless Routing (GPSR) and Greedy Perimeter Coordinator Routing (GPCR) both use greedy algorithms to forward packet and try to find a route by the right-hand rule in perimeter mode when it encounters a local maximum. These protocols could forward packets efficiently given that the underlying network is fully connected. However, the dynamic nature of vehicular network, such as vehicle density, traffic pattern, and radio obstacles could create unconnected networks partitions.

To this end, we propose a hybrid geographic routing solution GeoDTN+Nav, an extension of VANET Cross Link Corrected Routing (VCLCR), which improves VCLCR by exploiting the vehicular mobility and on-board vehicular navigation systems. GeoDTN+Nav outperforms GPSR and GPCR because it is able to estimate network partitions and then improves partitions reachability by using a store-carry-forward procedure when necessary. We propose a virtual navigation interface (VNI) to provide generalized route information for the delay tolerant forwarding. We finally evaluate the benefit of our approach first analytically and then with simulations. By using delay tolerant forwarding, GeoDTN+Nav greatly increases the packet delivery ratio in a sparse network.

Keywords

Geographic routing, sparse, partition, store-carry-forward, delay tolerant network, VANET

1. INTRODUCTION

Vehicular Ad Hoc Networks (VANET), a particular instance of Mobile Ad Hoc Networks (MANET), are a particular kind of networks, where vehicles or transportation infrastructures equipped with transmission capabilities are interconnected to form a network. The topology created by vehicles is usually very dynamic and significantly non uniformly distributed. In order to transfer information on that kind of networks, standards MANET routing algorithms are not appropriate. The other particularity of VANET is the availability of navigation systems, thanks to which each vehicle may be aware of its geographic location as well as its neighbors'. Another kind of routing approach, called *Geographic Routing* becomes possible, where packets are forwarded to destination simply by choosing a neighbor which is geographically closer to the destination.

Although geographic routing is a promising method in VANET, it also has limitations. Due to the non uniform topology distribution, a node may not be able to find a neighbor closer to the destination than itself, a situation called a"local maximum". Several routing protocols have been proposed (GPSR [3], GPCR [8], VCLCR [6]) to solve this problem. GPSR introduces a perimeter mode to extract packets from local maxima by planarizing the network and forwarding packets around the obstacle. This solution has been proved to be suboptimal in VANET first as the planarization procedure is complex and second as it also forces a packet to progress in small steps. GPCR suppresses planarization by assuming that urban street maps naturally form planar graphs. Each road segment is an edge of a planar graph while nodes at junctions are vertices. Routing decisions are made only at junctions; between junctions, packets are simply forwarded to next junction. The limitation of GPCR is that it assumes that the junction nodes always exist. But in reality, it is not true. When junction nodes are missing, packets will be forwarded across junctions, causing possible routing loops. VCLCR solves this problem by detecting loops and removing cross links whenever possible. It greatly increases the packet delivery ratio compared to GPSR or GPCR.

Even if VCLCR can detect routing loops and remove cross links, packets can still be dropped due to network disconnection or partitions. Indeed, in case of sparse VANETs or when vehicles in a VANET are significantly aggregated at junctions, network partitions occur and none of the previously described solution is able to deliver packets across partitions. However, vehicles mobility patterns may help to recover from this situation by letting a vehicle carry packets to a different partition. If sufficient vehicles are moving between network partitions, then packets can be delivered even if the network is disconnected. This is the idea behind the concept of Delay Tolerant Networks (DTN) [2]. DTN protocols such as [10, 9] employ such a store-carry-and-forward mechanism to forward packets.

Numbers of delay tolerant routing protocols exploiting different strategies to route packets have been developed. GeOpps [7] takes advantage of the vehicles' navigation system suggested routes to select vehicles that are likely to move closer to the final destination of a packet. It calculates the shortest distance from packet's destination to the vehicles' path, and

---

estimates the arrival of time of a packet to destination. During the travel of vehicles, if there is another vehicle that has a shorter estimated arrival time, the packet will be forwarded to that vehicle. The process repeats until the packet reaches destination. MoVe [5] uses the motion vector of a node to take forwarding decisions. The motion vector represents a node's current moving direction. MoVe chooses the neighbor which has the shortest distance to destination. The shortest distance to destination is calculated as the distance from destination to the extending line of the motion vector. A variante is MoVe-Lookahead [5], which uses the next waypoint, i.e. points where vehicles change their directions, instead motion vectors to calculate the shortest distance.

All of these routing algorithms lack an integrated protocol to combine both the efficient position-based routing for connected partitions and delay tolerant forwarding for routing between partitions. In this paper, we propose such a complete solution, called GeoDTN+Nav, that includes a greedy mode, a perimeter mode, and DTN mode. In order to know when to use each of these modes, a network partition method is proposed that evaluates for each packet the correct forwarding method to use in order to guarantee a high packet delivery even in sparse or partitioned networks. We also introduce a Virtual Navigation Interface (VNI) which efficiently provides mobility patterns in order to choose the best delay tolerant forwarders. We analytically and simulatively analyze the performance of our solution and illustrate how it outperforms GPSR and GPCR, and manages to transmit information when they both fail.

The rest of the paper is organized as follows: Section 2 provides a short discussion of the current efforts in geo-routing and delay tolerant forwarding. In Section 3, we formally introduce the virtual navigation interface model. In Section 4, we describe the GeoDTN+Nav algorithm and illustrate its properties. Section 5 presents a simplified analytical model to evaluate the performance of GeoDTN+Nav. Section 6 presents the synthetic and realistic simulative evaluation of GeoDTN+Nav. Finally, Section 7 concludes the paper.

2. GEO-ROUTING IN VANET

In this section, we briefly describe Greedy Perimeter Stateless Routing (GPSR), Greedy Perimeter Coordinator Routing (GPCR), and VANET Cross Link Corrected Routing (VCLCR).

2.1 Greedy Perimeter Stateless Routing

The Greedy Perimeter Stateless Routing (GPSR) [3] is a routing protocol that uses the positions of wireless node and the destination location of the packet to decide the forwarding decision. In GPSR, intermediate nodes only maintain the location of their neighbor nodes rather than routing metrics, which makes the protocol stateless. GPSR has two modes: greedy forwarding mode and perimeter mode.

In a network using GPSR as routing protocol, when an intermediate node receives a packet, it will forward the packet to the neighbor that is geographically closest to the destination node. This approach is called greedy forwarding. If an intermediate has no other neighbors closer to the destination than itself, this intermediate node is the local maximum node for this packet and the packet will switch to perimeter mode to recover from the local maximum.

The idea of GPSR's perimeter mode is to forward packet by right-hand rule with the starting vector constraint. When a packet switches itself to perimeter mode at an intermediate node x, it first draws a virtual vector from x to destination node D. Node x then forwards the packet to the first edge counterclockwise about x from the vector. (An edge here is defined as a bi-direction feasible transmission pair between two wireless nodes.) Then GPSR always finds the next hop by the right-hand rule – the next forwarding edge should be
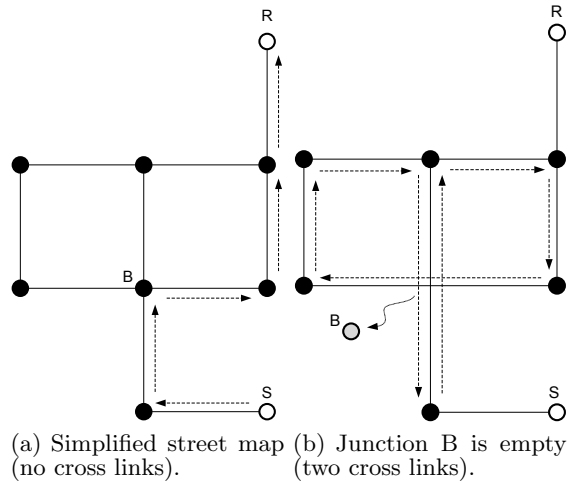

(a) Simplified street map (no cross links). (b) Junction B is empty (two cross links).

Figure 1: Cross-link causes routing loops.

the first edge counterclockwise from the previous edge without crossing the starting vector. When a packet is forwarded to a node which is closer to D than x, it switches back to greedy forwarding mode. Otherwise, when it loops back to x, the packet will be dropped. The perimeter mode of GPSR must be applied on a planar graph, or the crosslink may cause routing loops. GPSR proposes two schemes to construct a planar graph. However, issues such as obstacles and asymmetric radio range cause planar graphs unable to be formed correctly. Many later works have proposed geographic routing without the requirement of planar graphs.

2.2 Greedy Perimeter Coordinator Routing

Two methods are proposed in GPSR to construct planar graph: Relative Neighborhood Graph (RNG) and Gabriel Graph (GG). However, it is impossible to construct a planar graph in VANET, because the network topology is always changing. Each time when nodes move, a new planar graph has to be constructed. Greedy Perimeter Coordinator Routing (GPCR) [8] solves the planarization problem by exploiting the urban street map that naturally forms a planar graph. Each road segment forms the edge in network topology, and the junctions of roads form the vertices. In GPCR's greedy mode, a node forwards packets until it reaches a node at a junction. The junction node forwards packets by choosing one neighbor which has the shortest distance to destination. In the perimeter mode, junction nodes forward packets to the next hop by applying right-hand rule. Non-junction nodes forward packets until it reaches a junction node.

GPCR assumes that there is always a node at a junction. But this assumption does not always hold. If the junction node is missing, the network topology may not be planar any more. The packet will be forwarded across junctions. This causes routing loops and packet's dropping. Figure 1(a) and 1(b) illustrate an example. Originally S forwards packets to R along the dash line in 1(a). If the junction node B is missing, the packet will be forwarded cross the junction, goes back to S, and gets dropped shown in Figure 1(b).

2.3 VANET Cross Link Corrected Routing Protocol

Lee *et al.* [6] proposed VANET Cross Link Corrected Routing (VCLCR), a geographic routing solution that improves GPCR by removing cross links induced by perimeter traversal GPCR algorithm. The concept is to use the loop back packet as a crosslink detection probe. When a packet is forwarded by perimeter mode, it records the path information in the packet. When the packet routes back to the perimeter mode's starting point, it checks the path it traverses and sees if there is a routing loop and cross links.
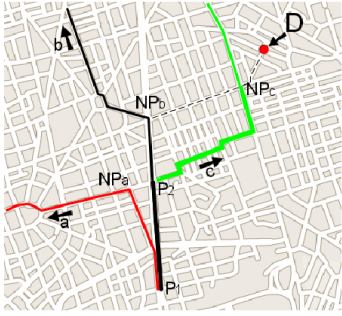
Figure 2: GeOpps Neighbor Selection

More specifically, when a node receives a packet and discovers that there is a loop, it checks the traversal history and sees if it has traversed through any cross link. If not, it indicates there is no available path to the destination and the packet will be dropped. Otherwise, the packet will be forwarded again by right-hand rule. In addition, one of the neighboring links that is crossed and only traversed once will be removed. The reason that links traversed twice will not be removed is because it may disconnect the graph [4]. This cross-link-removal procedure is on-demand and the overhead is small. When a packet in perimeter mode is forwarded to any node that is closer to the destination node than the perimeter mode's starting node, the packet will switch back to greedy forwarding mode and reset its path information.

When the packet is forwarding on a path without cross link, VCLCR performs the same as GPCR. By eliminating loops in packets paths, VCLCR increases the packet delivery rate and also reduces failed hops compared to GPCR.

2.4 GeOpps

GeOpps [7] is a delay tolerant routing algorithm that exploits the availability of information from the navigation system (NS). A navigation system includes a GPS device, maps, and the function to calculate a suggested route from current position to a requested destination. GeOpps have vehicles communicate with neighbor's navigation system and use the obtained information to perform efficient and accurate delay tolerant network.

A NS is assumed to have the ability to calculate the route to a given destination and to estimate the required time to a given destination. When a vehicle wants to deliver a data packet, it broadcasts the destination of it. The one-hop neighbors of the packet holder will calculate the "Nearest Point" (NP). Since every vehicle using NS has a suggested path, the NP is the location that is the location on the path which is geographically closest to the destination. For example, in Figure 2, paths a, b and c are the different suggested paths of three vehicles. Their NPs to the destination D is marked as NP_a, NP_b, and NP_c.

3. VIRTUAL NAVIGATION INTERFACE FRAMEWORK

The goal of the Virtual Navigation Interface (VNI) is to look for neighboring vehicles that can help deliver packets in partitioned networks. Without any prior information, randomly choosing a neighbor to carry a packet might not be appropriate because this neighbor might move farther away from the destination. Yet, with external knowledge of neighbors' path or destination information, we could take a better decision.

In [7], GeOpps assumes that vehicles are equipped with navigation systems that contain geographical locations. Hence it makes carrier decision based on which neighbor can deliver the packet quicker/closer to its destination. This assumption might be valid since more and more cars are equipped
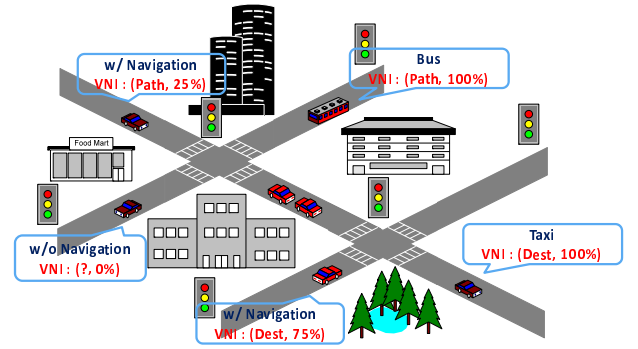

Figure 3: Categories of vehicular route pattern and VNI example.

with on-board navigation systems. In addition, many applications, such as route suggestion based on real-time traffic and proximity based advertisement, may encourage the deployment of navigation systems. However, this assumption neglects the heterogeneity of vehicles. Indeed, although the content of GPS information has been standardized, the content and transmission format of navigation information is not and may differ between different class of vehicles, if these latter vehicles are even equipped with such devices.

In GeoDTN+Nav, we adopt a more relaxed and generalized assumption and provide a framework for the different kind of navigation information available. We assume that every car is equipped with a *Virtual Navigation Interface* (VNI). We describe the assumption and model of VNI in the following sections.

3.1 Vehicle Mobility Categories

In this section, we present a scenario that motivates the idea of virtual navigation interface. As Figure 3 shows, different kinds of vehicles together create a vehicular ad hoc network. These vehicles move based on different patterns:

- Bus, train: These vehicles' movement is strictly restricted by a predefined route. For a given bus, its destination, path to the destination, and schedule are given in advance. For these vehicles, they do not require navigation systems, but they would move based on a deterministic route,

- Taxi, Van pool: Unlike previous ones, these vehicles do not move along a fixed route. However, no matter how different the routes are, they would eventually arrive at a predefined destination. For example, a taxi driver may dynamically choose a different path to avoid traffic, but he should still drive passengers to their destination,

- Vehicles equipped with Navigation Systems: Privately owned vehicles might be equipped with navigation systems. These vehicles are expected to follow the route suggested by navigation systems because navigation systems usually suggest shortest routes, or simply because drivers may not know the route to their destinations. However, it is also possible that drivers do not follow the suggested route or they may change the destination during their travel. Therefore, these vehicles introduce extra uncertainties in its movement pattern,

- Vehicles not equipped with Navigation Systems: Privately owned vehicles also might not be equipped with navigation systems, and therefore they are not capable of providing their route information. However, these vehicles still do not move randomly. For example, vehicles are expected to maintain their direction along a road before they arrive at the next junction. It is not likely that vehicles would move back and forth irrationally.

Based on vehicles movement pattern discussed above, we categorize vehicles into four broad categories:

Categories	Examples
Deterministic (Fixed) Route	Big Blue Bus, UCLA Shuttle, Metro Train
Deterministic (Fixed) Destination	Taxi, UCLA Van Pool
Probabilistic (Expected) Route/Destination	Navigation system guided vehicles
Unknown	Random movement

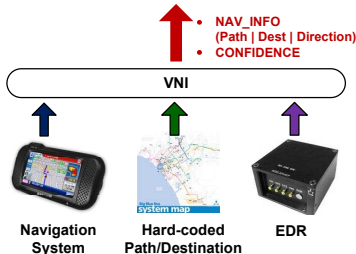**Table 1: Categories of Vehicular Route Pattern.**

Figure 4: Virtual Navigation Interface

1. Deterministic (Fixed) Route: Vehicles move strictly along preconfigured routes. These vehicles will not deviate away from their routes. Also, the moving direction of vehicles can be derived from their routes,

2. Deterministic (Fixed) Destination: Vehicles move strictly toward a preconfigured destination. However, it is possible that vehicles take different routes to reach the destination. A coarse-grain moving direction can also be obtained,

3. Probabilistic (Expected) Route / Destination: Vehicles may move based on suggested routes or destinations. They are allowed to change their route or destination discretionarily,

4. Unknown: Vehicles could not provide information about their route, but they do not move randomly either.

Categories of vehicles and examples are summarized in Table 1.

3.2 Virtual Navigation Interface (VNI) Design

We have already discussed different categories of vehicles in the previous section. In order to provide a consistent and generalized view of different vehicles in our routing decision, we assume VNI is installed on every vehicle. VNI is a lightweight wrapper interface that interacts with underlying vehicular components. It provides two kinds of primitive information:

1. Route_info: Route_info represents the vehicle's route information. Note that route information may either consist of detailed path, destination, or the direction of vehicles, depending on the types underlying data sources. As in Figure 4, VNI might be able to retrieve detailed path information from a navigation system while it may only retrieve vehicle's direction from an Event Data Recorder (EDR). In addition, VNI can also retrieve pre-configured route information.

2. Confidence: Confidence indicates the probability that the vehicle's movement would abide by the given route information. More specifically, confidence with 0% means that the vehicle move completely in random while confidence with 100% means that the vehicle move strictly based on its route information. This confidence information can be configured or derived from vehicles' movement history.

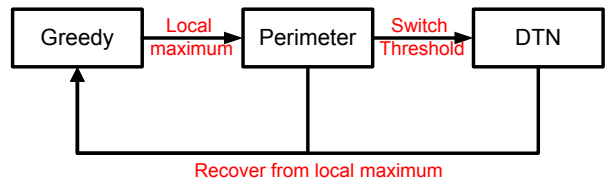For example, in Figure 3, we installed VNI on every vehicle:

Figure 5: Switch between greedy, perimeter, and DTN mode.

- VNI on buses would broadcast two-tuple information ($Path, 100\%$) because buses move deterministically along its preconfigured route.

- VNI on taxis would broadcast ($Dest, 100\%$) because taxis move deterministically toward its destination.

- VNI on vehicles with navigation systems would broadcast ($Path/Dest, P\%$) depending on what information the VNI can obtain from the underlying navigation system.

- VNI on vehicles without navigation systems might broadcast ($?, 0\%$) because VNI cannot obtain enough route information, or it might broadcast ($Dir, P\%$), if VNI is able to estimate vehicles' moving direction.

Based on the unified information provided by VNI, every vehicle now can collect navigation information from its neighbors and make routing decision accordingly. Note that this generic information advertised by VNI is independent from our GeoDTN+Nav protocol. It can also be used by other routing protocols serving different purposes. However, in this paper, we focus on using information provided by VNI to choose a neighbor which can potentially carry packets across disconnected networks.

4. GEODTN+NAV ALGORITHM

Traditionally, geo-routing routes packets in two modes: the first mode is greedy mode, and the second mode is perimeter mode. In greedy mode, a packet is forwarded to destination greedily by choosing a neighbor which has a bigger progress to destination among all the neighbors. However, due to obstacles the packet can arrive at a local maximum where there is no neighbor closer to the destination than itself. In this case, the perimeter mode is applied to extract packets from local maxima and to eventually return to the greedy mode. After a planarization process, packets are forwarded around the obstacle towards destination. In this way, the packet delivery is guaranteed as long as the network is connected.

However, the assumption that the network is connected may not always be true. Due to the mobile characteristics of VANET, it is common that the network is disconnected or partitioned, particularly in sparse networks. The greedy and perimeter modes are not sufficient in VANET. Therefore, we introduce the third mode: DTN (Delay Tolerate Network) mode, which can deliver packets even if the network is disconnected or partitioned by taking advantage of the mobility of vehicles in VANET. Unlike the common belief that mobility harms routing in VANET, we specifically count on it in this work to improve routing.

In short, packets are forwarded first forwarded in greedy mode, and then by perimeter mode when a packet hits a local maximum. If the perimeter mode also fails, it finally switches to the DTN mode and relies on mobility to deliver packets. Figure 5 illustrates the transition diagram between these three modes.

Two questions arise in this scheme: Exactly when should we switch to DTN mode, and when to switch back to greedy mode. For the former, we will use a cost function and a threshold related to a network partition detection and to the
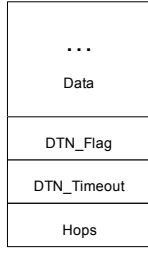
Figure 6: Packet format.

quality of nodes mobility pattern between partitions. For the latter, similar to the recovery mode, we will return to greedy mode when a relay with better progress than the one that triggered the DTN mode is found. We will discuss the details in Section 4.3.

4.1 Restricted Greedy Forwarding

In GeoDTN+Nav, the default greedy forwarding strategy is the same as the restrictive greedy forwarding in GPCR, where packets are always forwarded between junction nodes as junctions are the only places where a node can make significant routing decisions. This remains true even if a current forwarding node can greedily forward packets beyond a junction. At junctions, a greedy decision is made to determine which road direction should be taken that can bring the maximum progress towards the destination. If a local maximum is reached, the recovery mode, called the perimeter forwarding, is used.

4.2 Perimeter Forwarding

In GeoDTN+Nav, the defuelt recovery mode is the same as VCLCR's. The goal of VCLCR in perimeter forwarding is to detect and remove cross links created by the lack of junction nodes to improve packet delivery. For GeoDTN+Nav, in order to support delay tolerant forwarding, we piggyback the following extra fields in data packets as shown in Figure 6:

1. DTN_Flag: the DTN_flag indicates whether or not this packet can be forwarded by delay tolerant mode. Applications that do not require on-time delivery can enable this flag to improve packet deliver probability.

2. DTN_Timeout: Applications specify packets' tolerated delay. Based on this information, nodes buffer and carry DTN packets can flush packets that are already expired or decide which packet to delete based on buffer management policy.

3. Hop_Count: The field records the number of hops that a packet has been forwarded in perimeter mode. GeoDTN+Nav uses this information to determine if the network is disconnected. This field can be replaced or augmented if future works adopt other means to measure network connectivity.

The basic idea behind GeoDTN+Nav is that in the perimeter forwarding mode, nodes keep suspecting whether the network is disconnected based on how many hops the packet has traveled in the perimeter mode. Every node also monitors its neighbors' navigation information. Based on the connectivity and navigation information, a *switch score* is calculated for each neighbor. A packet would be switched to DTN mode only when the switch score is beyond a certain predefined threshold and the DTN_flag is set.

For all neighbors, if no switch score is beyond the threshold, the packet would be forwarded based on conventional perimeter forwarding and increment the hops by one.

4.3 DTN Forwarding

With DTN forwarding, the first question to address is when we should switch to DTN mode. Two factors need to be considered: network disconnections and delivery quality of nodes storing a packet. Determining network disconnectivity is not an easy task; in fact, there is no way to know whether the network is connected or not unless we have the complete information of network topology. Moreover, even if we have the complete network topology information, any decision is only valid at the time of the evaluation because the topology is changing all the time. Thus, what we can do is to take a good guess. We propose to base this decision on the hop count, as an increasing hop count in perimeter mode could mean the network is partitioned.

The delivery quality of nodes carrying a packet is the second criterion to determine whether we should use DTN forwarding or not. If there is a good neighbor that has a mobility pattern that will bring the packet closer to destination, we rely on it to deliver the packet. By a good neighbor, we mean a neighbor which has a path, destination, or direction towards the destination with high confidence. For example, a bus may have paths in NVI because its route is well-known, and may have high confidence because it seldom changes such route. A taxi may not transmit its path but its destination because it only knows the destination where customers want to go, and the confidence associated to that destination is low as real traffic condition may alter it.

Network disconnectivity and the delivery quality only are not enough to define a good neighbor. We also have to consider the neighbor' moving direction. For example, a bus may have good delivery quality because it has a fixed route closer to destination but it is moving away from it, which makes it a bad potential relay to carry a packet.

Combined the three factors, we derive the "score function" S as follows:

$$S(N_i) = \alpha P(h) \times \beta Q(N_i) \times \gamma Dir(N_i) \qquad (1)$$

where:

$$
\begin{array}{rcl}
S(N_i) & : & \text{Switching score of } N_i \\
P(h) & : & \text{Probability that the network is disconnected} \\
 & & \text{(range from 0 to 1)} \\
Q(N_i) & : & \text{Delivery quality of } N_i \text{ in DTN mode} \\
 & & \text{(range from 0 to 1)} \\
Dir(N_i) & : & \text{Direction of } N_i \text{ (range from 0 to 1)} \\
\alpha, \beta, \gamma & : & \text{System parameters} \\
N_i & : & \text{a neighbor of current node } i \\
h & : & \text{hop counts that the packet has traversed in} \\
 & & \text{perimeter mode.}
\end{array}
$$

The function $P(h)$ represents the probability that the network is disconnected, as measured by hop counts. The larger the hop counts, the higher the probability that the network is disconnected. We use Algorithm P to calculate function $P(h)$:

Algorithm P

Input: Current hop count h, first edge traversed in perimeter mode $e0$

Output: Probability that the network is disconnected

1. $nextHop \leftarrow$ perimeter forwarding by right-hand rule from current node
2. $nextEdge \leftarrow$ current node to $nextHop$
3. **if** $nextEdge$ equals $e0$
4.   **then return** 1
5.   **else return** $\frac{max(0, h - h_{min})}{h_{max} - h_{min}}$

In Algorithm P, h_{max} is the maximum hops for which we assume the network is connected. After this hop count, $P(h)$ equals to 1, which means the network is disconnected. In our algorithm, h_{max} equals TTL. h_{min} is the minimum hop
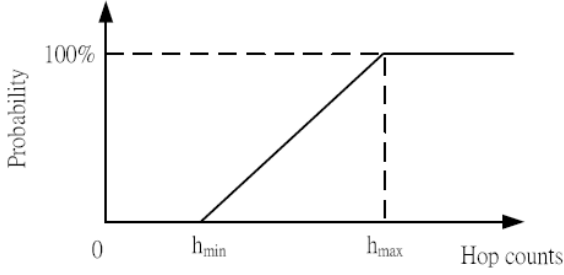
Figure 7: Function P(h)

counts that we will switch to DTN mode, i.e., we will only apply DTN forwarding after the packet has been forwarded more than h_{min}. The reason for this is that the perimeter forwarding mode is more efficient than relying on mobile vehicles to deliver packets. We therefore want to further try several hops in perimeter mode before switching DTN mode. If a packet goes back to the point where it entered perimeter mode (i.e., $e0$), Algorithm P will return 1 because we simply assume that the network is partitioned. The relationship between hop counts and $P(h)$ is illustrated in Figure 7:

The function $Q(N_i)$ represents the delivery quality of neighbor N_i. We use Algorithm Q to compute the delivery quality of each neighbor:

Algorithm Q
Input: Neighbor's location (nl), neighbor's confidence (c), the destination ($dest$), the node that enters perimeter mode (L_f)
Output: Neighbor's delivery quality $\in \mathbb{R}$
1. $D \leftarrow Dist(dest, L_f)$
2. $d \leftarrow Dist(dest, nl)$
3. **return** $(\frac{max(0, D-d)}{D})c$

In Algorithm Q, D is the distance between the destination and the location of the node that switched to perimeter mode. d is the distance between the destination and the location information *Nav-info* of neighbors broadcasted in beacon packets. If *Nav-info* contains the path, then d is the distance from packet's destination to the closest road segment on this path. If *Nav-info* contains the neighbor's destination, then d is the distance from packet's destination to neighbor's destination. If *Nav-info* contains the direction, then d is the perpendicular distance from packet's destination to the extending line of the direction. For example, in Figure 8, the packet is now at node C. There are three neighbors of current node, $N1$, $N2$ and $N3$. For $N1$, $d = d_1$. For $N2$, $d = d_2$. For $N3$, $d = d_2$. Using Algorithm Q, we obtain the delivery quality of each node.

As mentioned before, $Q(Ni)$ may not be enough to define a "good" neighbor; we also need to consider the moving direction. For example, in Figure 8, for current node C, neighbor $N1$ has a path which has shortest distance to destination. $N1$ is definitely a good choice to forward packet in comparison to $N2$ and $N3$ in this case. But what if $N1$ is moving away from the destination at that time? Obviously it is not a good choice to carry the packet. It may be better to choose a neighbor that is moving toward the destination rather than moving away. Therefore, we add the third function, $Dir(Ni)$, in our score function:

Algorithm Dir
Input: Neighbor's direction (ndr), the destination ($dest$), the current node's location ($curLoc$)
Output: Neighbor's direction quality $\in \mathbb{R}$
1. $\theta \leftarrow$ the angle formed by the vector formed by ndr and the vector formed by $dest$ and $curLoc$
2. **if** θ equals 0
3.   **then return** 1
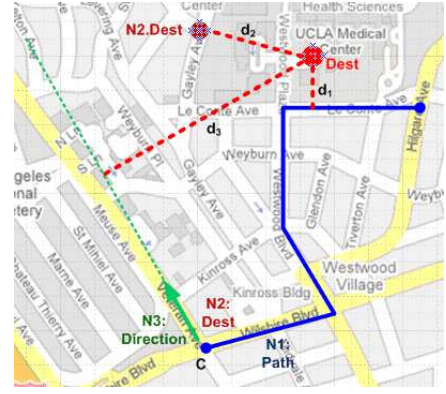4.   **else return** $\frac{1}{abs(\theta)}$

Figure 8: Calculate $Q(N_i)$

Here is the complete algorithm using all of the three modes:

1. Every node periodically broadcast two-tuple navigation information by VNI: <Nav-info, Confidence>.

2. A packet is forwarded in greedy mode, until it reaches a local maximum.

3. Then it switches to perimeter mode and record its own location $e0$ and its *dest* in the packet header.

4. At each hop in perimeter mode, do the following things:

   (a) Use $P(h)$ to calculate the probability of network disconnectivity.

   (b) Use $Q(N_i)$ to calculate the delivery quality of each of its neighbors as well as itself.

   (c) Use $Dir(N_i)$ to calculate the direction quality of each neighbor.

   (d) Calculate the global score for each node by using Equation 1.

   (e) If one of the scores is greater than S_{thres}, forward the packet to the respective node and switch to DTN mode. The packet will be stored and carried by that node until it can switch to greedy mode. If there are multiple nodes that have greater scores than S_{thresh}, choose the node with highest score and forward the packet to it.

5. If the hop count reaches the TTL and there is no node with a score greater than S_{thres}, drop the packet.

We have described an architecture that integrates three modes (greedy, perimeter, DTN) in VANET in order of delivery for sparse or partitioned networks. The "score function" here is an example that takes into account of the network disconnectivity and delivery quality of nodes carrying a packet. A better function can be derived from a careful analysis of traffic patterns and forwarding policy, which we let to future work. We describe how we can efficiently set S_{thresh} in Section 5.

4.4 GeoDTN+Nav Routing with VNI Examples

After having described the VNI and the GeoDTN+Nav routing protocol, we now demonstrate their joint functionalities in two examples. We emphasize that the main purpose of switching from perimeter mode to DTN mode is to virtually connect network partitions and improve the delivery ratio, while switching from DTN mode back to greedy is to improve delivery delay in connected partitions. For simplicity, we assume all packets in our examples are already in perimeter mode and each node has already collected navigation information broadcasted by the VNI installed on its neighbors.
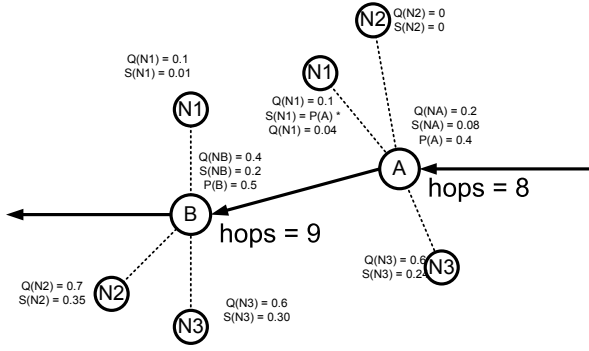
Figure 9: Routing example 1.

4.4.1 Example 1: Greedy to DTN

Assume weight parameters α, β, and γ are 1, and the threshold S_{thres} is 0.25. Also suppose that a packet has traversed 8 hops in perimeter mode up to node A. Node A has three neighbors, N1, N2, and N3. While the packet arrives at node A, node A calculates the probability of network disconnectivity by applying Algorithm P and obtains $P(8) = 0.4$. Note that Algorithm P depends only on the hop counts that has been traversed in perimeter mode. At the same time, node A calculates the delivery quality of its neighbors, also including itself, in order to know if they could bring the packet to the targeted network partition in DTN mode. It finally computes the "score function" S by multiplying $P(h)$, $Q(N_i)$, and $Dir(N_i)$. At this time, none of its neighbors including itself has a higher score than S_{thresh}, so the packet will remain in perimeter mode to the next hop. The above process repeats in node B, but now two neighbors $N2$ and $N3$ have greater scores than S_{thresh}. Node B therefore switches to DTN mode, and chooses the neighbor with the greatest score to carry the packet, node $N2$ in this case. $N2$ will buffer the packet until it reaches a point where it can switch back to greedy mode. Once it has reached that a point, the packet is forwarded to destination in greedy mode again.

4.4.2 Example 2: DTN to Greedy

The second example, depicted in Figure 10, illustrates the condition for a node to switch from DTN mode to greedy mode. In this example, a packet first hit the local maximum at point $e0$ where it switched to perimeter mode, before switching to DTN mode at node A. Node A periodically checks on the packets in its buffer to decide whether there is a packet that can be forwarded again in greedy mode. In order to switch a packet's forwarding mode back to greedy, Node A needs to find a neighbor closer to the destination than the node $e0$ where the initial local maximum was. As node A moves to a point B, it detects that its distance to $Dest$ is smaller than the distance from $e0$ to $Dest$. Therefore, if a neighbor is located at that point, node A is able to switch back to greedy mode. If there is not such a neighbor, the packet will stay in Node A until it finds an applicable neighbor to forward packets, as it should never switch back to greedy until it has reached the network partition possibly containing the destination node.

5. PARAMETERS ANALYTICAL EVALUATION

In Section 4, the algorithm for GeoDTN+Nav has been proposed in order to improve packet delivery in sparse or partitioned networks for delay tolerant applications. Due to the large set of parameters, the performance of GeoDTN+Nav may be hard to evaluate. This section proposes to study the proper settings for weight values and utility functions through an analytical model. For dense and connected net-
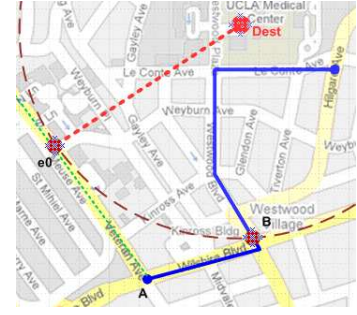

Figure 10: Routing example 2.

Table 2: Notation in Analytical Model

G_x	The probability of switching to greedy mode after x hops in perimeter mode
N_x^R	The expected number of type R neighboring nodes after x hops in perimeter mode.
$P(x)$	The probability that network is disconnected after x hops.
$Q(N_x)$	The delivery quality function for traffic type R in DTN mode
$P_Q^x(k)$	The probability mass function for $Q(Nx) = k$
S_{thres}	The threshold of switching to DTN

works, greedy forwarding is able to efficiently deliver packets with low delay. On the contrary, for sparse or partitioned networks, it has a high chance to fail and must switch to perimeter mode. If the perimeter mode designed by VCLR cannot successfully return to greedy, then we have the option to switch to the DTN mode. This action however trades an improved delivery ratio for a significantly increased delivery delay. So, when tuning the settings for GeoDTN+Nav, we need to consider the application requirements between delivery ratio and it respective delay. Table 2 displays the notation of our analysis.

In Section 4.3, $Q(N_x)$ is introduced as the delivery quality function for neighboring node after x hops, $P(x)$ is introduced as the disconnect probability of a network after a packet is forwarded for x hops, and S_{thres} is the threshold of scoring function to switch to DTN mode. These values together with weight parameters are the controllable setting for GeoDTN+Nav. They are used to calculate T_x, which is the probability to switch to DTN mode after x hops.

The other variable G_x is defined as follows: When a packet is forwarded in perimeter mode at xth hop, the probability that it switches to greedy mode at next hop is G_x and the probability that it remains in perimeter mode is $1 - G_x$. Here we assume that this variable is obtained by simulation or real network experiment. By definition it is clear that

$$\sum_{i=1}^{\infty} \{G_i \prod_{x=0}^{i-1} (1 - G_x)\}$$

is the probability that a packet can switch back to greedy mode after it enters the perimeter mode. Since the inability to switch back to greedy mode means the network is either disconnected, or the packet loops in perimeter mode. Therefore, the probability that network is disconnected is at most $1 - \sum_{i=1}^{\infty} \{G_i \prod_{x=0}^{i-1} (1 - G_x)\}$.

Another variable that can be parameterized by measuring topology is N_x^R. N_x^R stands for the expected number of type R nodes that can be used for DTN mode when a packet is traversed in perimeter mode on x_{th} hop. Each type R represents different kinds of vehicular category, such as taxis, buses, trains, or cars. Note that N_x^R includes the node that currently holds the packet, because if the packet owner itself finds out that the threshold is exceeded, it will change to DTN mode as well.
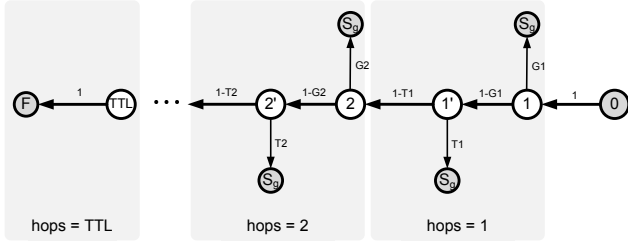
Figure 11: Markov Chain for perimeter forwarding.

The definition of T_x is similar to G_x. When a packet is on its xth hop in perimeter mode, it has a T_x probability to switch to DTN mode on itself and any of its neighboring nodes. For a type R node, if the forwarding packet is switched to DTN mode at this node, this node must satisfy the condition:

$$S_{thres} \leq \alpha P(x)\beta Q(N_x)\gamma Dir(N_i).$$

Furthermore, it is reasonable to assume that $Dir(N_i)$ is a uniform distribution between 0 to 1. Therefore, after x hops, the probability that none of the neighboring nodes in type R satisfies this constraint is:

$$F_x^R(S_{thres}) = (\int_0^1 \int_0^{S_{thres}/\alpha\beta\gamma zP(x)} P_Q^x(y)dydz)^{N_x^R}$$

Therefore, T_x can be written as follows:

$$T_x = 1 - \prod_{\forall R} F_x^R(S_{thres})$$

F_x^R represents the probability that all type R nodes within the delivery range of a current packet location do not have their scoring function S exceeding the threshold forcing the packet to stay in perimeter mode.

Figure 11 demonstrates our framework about this analytical mode. When a packet is forwarded in perimeter mode, it has three choices. If any of its neighbors is closer to destination than the starting point in perimeter mode, it will switch to greedy mode; if any of its neighbor or itself exceeds the DTN threshold, it will switch to DTN mode; if none of the previous case happens, it will remain in perimeter mode. Given T_x and G_x, the probability that a packet can successfully exits perimeter mode in x hops is:

$$
\begin{aligned}
G_1 &+ (1-G_1)T_1 + (1-G_1)(1-T_1)G_2 + \dots \\
&+ G_x\prod_{i=1}^{x-1}(1-G_i)(1-T_i) + T_x[\prod_{i=1}^{x-1}(1-G_i)(1-T_i)](1-G_x) \\
&= 1 - \prod_{n=1}^{x}(1-G_n)(1-T_n)
\end{aligned}
$$

Furthermore, the probability that a packet switches to greedy mode in x hops is the odd terms of the equation, which is:

$$\sum_{k=1}^{x} G_k I_k,$$

while:

$$I_k = \begin{cases} \prod_{i=1}^{k-1}(1-G_i)(1-T_i), & k > 1, k \in \mathbb{N} \\ 1, & k = 1 \end{cases}$$

Similarly, the probability that a packet switches to DTN mode in x hops is the even terms of the equation, which is:

$$\sum_{k=1}^{x}(1-G_x)T_x I_x,$$

This model allows us to predict the probability that a packet will switch to DTN whenever it enters the perimeter mode.
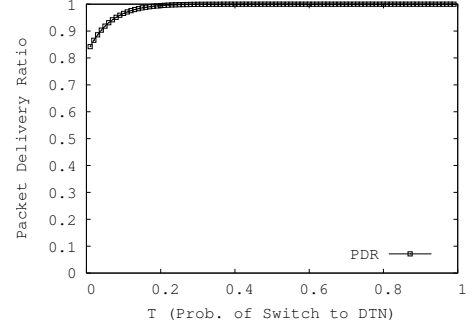

Figure 12: Probability vs. PDR

To understand how DTN affects the average packet latency, suppose that the hop-wise delivery delay in DTN mode is D_d, while that of the joint greedy and perimeter mode is D_g. The total average delay of packet delivery becomes:

$$\sum_{k=1}^{x} G_x I_x D_g + \sum_{k=1}^{x}(1-G_x)T_x I_x D_d$$

Figure 12 and 13 illustrate the PDR vs. latency tradeoff as a function of the probability of switching to DTN T_x, where the probability of switching to greedy mode G_x is fixed for each x, where $D_g = 5$, $G_x = 0.1$ and where a packet is dropped after 16 hops. As depicted on Figure 12, the packet delivery ratio increases when the probability to switch to DTN goes up, if the application only requires a minimum packet delivery ratio, it can be mapped directly to the probability to switch to DTN mode. Similarly, Figure 13 provides an upper bound for the probability to switch to DTN when there is a maximum average delay constraint. In the above example, if the packet wishes to have 90% delivery ratio and a delay less than 400, the probability of switching to DTN should be set between 0.1 and 0.4 by adjusting the threshold.

The next step is therefore to be able to compute a threshold satisfying the probability bounds. If a packet in perimeter mode should switch to DTN for a probability ranging from a to b, we have:

$$T_x = a < 1 - \prod_{\forall R} F_x^R(S_{thres}) < b.$$

Thus,

$$1 - b < \prod_{\forall R} F_x^R(S_{thres}) < 1 - a.$$

which leads to:

$$\prod_{\forall R} F_x^R(m) = \prod_{\forall R}(\int_0^1 \int_0^{mk/z} P_Q^x(y)dydz)^{N_x^R}$$

$$= [(\int_0^{mk} P_Q^x(y)dy) - \int_0^1 zP_Q^x(mk/z)(-mk/z^2)]^{N_x^R}$$

$$= [(\int_0^{mk} P_Q^x(y)dy) + \int_0^1 mkP_Q^x(mk/z)/zdz]^{N_x^R}$$

given $k = 1/(\alpha\beta\gamma P(x))$

In conclusion, in order to have T_x bounded between a and b, the threshold bound satisfies:

$$1 - b < [(\int_0^{mk} P_Q^x(y)dy) + \int_0^1 mkP_Q^x(mk/z)/zdz]^{N_x^R} < 1 - a$$

Figure 13 demonstrates another aspect of our analytical model. Suppose there are three different types of R nodes, and each of them has different DTN delays. If three different types of vehicles lead to a delay D_d of 5, 50, 500. If the network application wishes to have a latency lower than 200
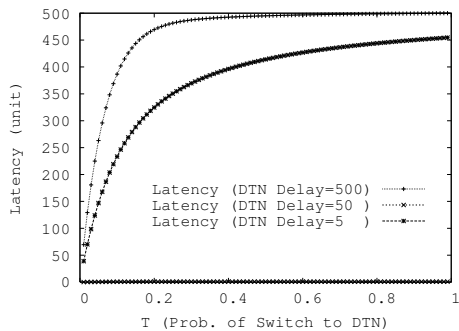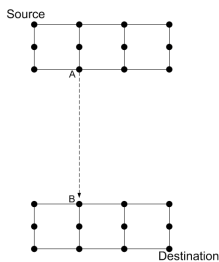
Figure 13: Probability vs. Latency

Figure 14: Synthetic Topology

seconds with DTN delay = 500, the probability of switching to DTN should be 0.5. Moreover, if the DTN delay is 50 or 5, the packet can always be delivered within 200 seconds. In this case the packet delivery ratio becomes the only constraint that is of concern. Once the probability of switching to DTN is obtained, S_{thres} can be calculated by the mentioned method above.

6. PERFORMANCE EVALUATION

Our simulations are based on Qualnet 3.95 with IEEE 802.11b DCF as the MAC with a transmission rate of 2Mbps. We evaluate our routing protocol in both synthetic and realistic scenarios.

6.1 Synthetic Scenario

We evaluate GeoDTN+Nav on a synthetic topology to show it is able to improve packet delivery ratio by delay tolerant forwarding. In Figure 14, nodes are placed so that they create two separate partitions. Obstacles are placed between different road segments if they do not share the same horizontal or vertical coordinates. The length of each edge is 300m, and the transmission range is 350m. According to the topology, each packet sent from the source to the destination will reach a local maximum and switch to the perimeter mode.

We place 'Bus' nodes at location A. 'Bus' nodes move towards location B at a speed of 50km/hour. We manipulate the number of bus nodes as well as their departure pattern in order to study the virtual connectivity between the two partitions. More precisely, we compare two departure patterns: a *uniform pattern*, in which a bus departure time is uniformly distributed throughout the whole simulation time; and the Random pattern, in which each bus node randomly departs.

We run simulations with constant bit rate UDP traffic with packet size 1460 bytes and compare GeoDTN+Nav with GPCR in packet delivery ratio, latency, and hop count. With $\alpha = \beta = 1$ and $S_{thresh} = 0.1$, the simulation results are depicted in Figure 15.

In Figure 15(a), for *uniform* departure configuration, when the number of 'Bus' nodes is small and due to the two network partitions, GPCR cannot deliver any packet to the destination. However, GeoDTN+Nav can achieve over 90% delivery ratio because packets are carried by 'Bus' nodes between the different partitioned. As the number of 'Bus' nodes increases, GeoDTN+Nav obviously maintains a steady high packet de-

livery ratio. On the contrary, GPCR has a sharp PDR jump with between 20 and 25 nodes. The reason is that the increasing number of 'Bus' nodes uniformly spread over the edge AB eventually reconnects the two partitions and allows GPCR to successfully deliver packets.

Similarly to uniform departure, GPCR is also not able to deliver any packet to the destination for the *random* departure configuration when the number of 'Bus' nodes is small. On the contrary, GeoDTN+Nav can still achieve around 80% delivery ratio. As 'Bus' nodes randomly depart, there is a chance that not a single 'Bus' node is available when GeoDTN+Nav needs it, which explains the 10% drop in delivery ratio between the uniform and random departures. For GPCR, due to random 'Bus' node departures, even an increasing number of 'Bus' nodes is never able to fully reconnect the two partitions. It yet increases the probability to find such configuration and explains the linear increase of the GPCR delivery ratio with the number of 'Bus' nodes.

Now considering the second metric set, Figures 15(b) and 15(c) show the average number of hops and latency a delivered packet travels. We may clearly see the tradeoff with Geo-DTN+Nav's high packet delivery ratio, as the number of hops and delivery delay are significantly higher. We however argue that the hop count and latency of GPCR remains steadily low as it can only deliver packets when the network is connected. When the number of 'Bus' nodes increases, the probability that a packet can be delivered by GeoDTN+Nav but solely based on the greedy mode also increases. Therefore, the hop count and the latency of packet delivery decrease accordingly.

6.2 Realistic Scenario

In this different experiment, realistic vehicular mobility traces have been generated using the Intelligent Driver Model with Intersection Management (IDM-IM) by VanetMobiSim [1], an open source and freely available realistic vehicular traffic generator for network simulators. The mobility scheme is based on a sequence of activities (home, work, shopping, etc..) described by a relative transition probability matrix. The unified transmission range is 350m. The urban topology employed in this paper is a realistic 1500m by 4000m Oakland area from U.S. Census Bureau's Topologically Integrated Geographic Encoding and Referencing (TIGER) database. 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).

We generate mobility traces for 50 nodes and introduce extra 'Bus' nodes. We manipulate the number of bus nodes as well as their departure patterns. In each simulation, 20 random source nodes send data to a fixed destination node using constant bit rate (CBR), a UDP-based packet generation application. To emulate radio propagation in urban area, blocking radio obstacles have been placed between different road segments if they do not share the same horizontal or vertical coordinates. In each experiment, we compare GPSR, GPCR and GeoDTN+Nav for the following metrics: 1)packet delivery ratio (PDR), 2)latency, 3)hop count.

Initially, because the node density is low and the connectivity is limited by obstacles therefore creating a large number of network partitions, the packet delivery ratio is very low for all protocols. This 'realistic' scenario is more challenging than the 'synthetic' one for GeoDTN+Nav, as 'Bus' nodes do not specifically connect two partitions and source-destination nodes are randomly distributed. In Figure 16(a), as the number of buses increases, GeoDTN+Nav's PDR increases accordingly, first because nodes have a higher probability to meet and delegate packets to 'Bus' nodes, but also as 'Bus' nodes have a higher chance to connect the corresponding partitions. However, without a DTN mode, GPCR and GPSR remain unable to efficiently transport packets in such a par-
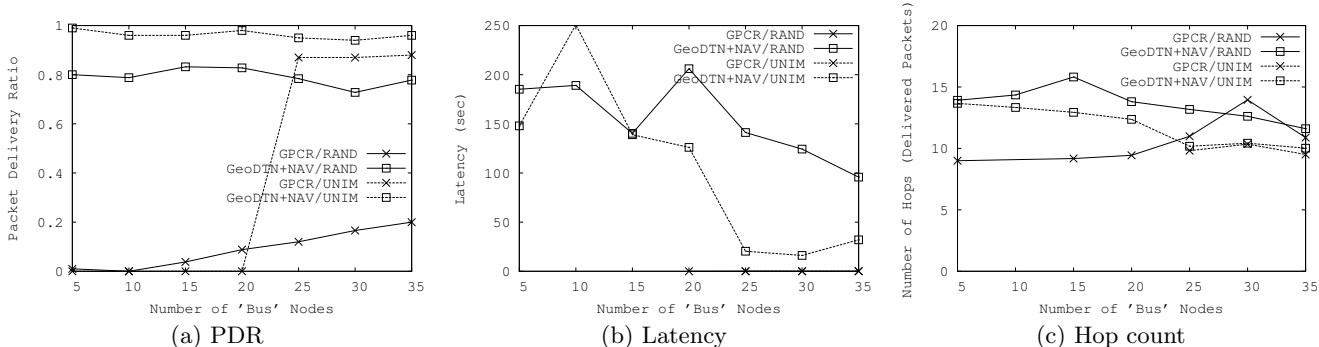
(a) PDR (b) Latency (c) Hop count

Figure 15: Measurements for synthetic scenario.
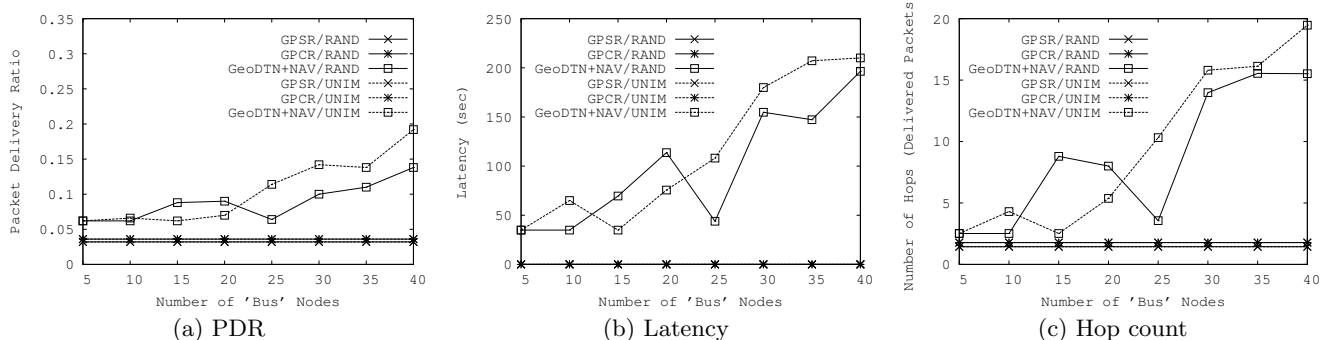

(a) PDR (b) Latency (c) Hop count

Figure 16: Measurements for realistic scenario.

titioned network. We may also see in Figure 16(a) that the uniform departure pattern also yields to a better PDR than the random one.

However, unlike the synthetic experiment described in the previous section, GPSR's and GPCR's PDR remain low even though the number of buses increases. For random source-destination pairs, the relatively low number of 'Bus' nodes is not sufficient to connect the different partitions. In fact, as it may be observed in Figure 16(c), GPSR and GPCR only successfully deliver packets when the source node and destination nodes are one hop away. In Figure 16(b) and 16(c), as the number of buses increases, GeoDTN+Nav's hops and latency increase. This is GeoDTN+Nav's fundamental tradeoff between packets' forwarding latency and delivery ratio.

7. CONCLUSIONS

In this paper, we proposed a hybrid Geo-DTN routing solution called *GeoDTN+Nav* for delay tolerant applications that improves geo-routing for sparse or partitioned networks by exploiting the vehicular mobility and on-board vehicular navigation systems to carry packets between partitions. Geo-DTN+Nav outperforms GPCR and GPSR in packet delivery ratio as it improves the graph reachability by using delay tolerant store-carry-forward solution to mitigate the impact of intermittent connectivity. The tradeoff is however an increased delivery delay. In order to evaluate this tradeoff and set optimal parameters, we conducted an analytical study of GeoDTN+Nav. Finally, in order to efficiently choose potential nodes to carry packets between partitions, we proposed a generic Virtual Navigation Interface (VNI) which provides generalized navigation information even when vehicles are not equipped with navigation systems. VNI is independent from GeoDTN+Nav and can be used by other routing protocols serving different purposes. In conclusion, we have presented an efficient and complete routing system for sparse and partitioned vehicular environments based on vehicular mobility that manages to deliver packets that other solutions do not.

8. REFERENCES

[1] J. Härri, F. Filali, C. Bonnet, and M. Fiore. VanetMobiSim: generating realistic mobility patterns for vanets. In *VANET '06: Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, pages 96–97, Los Angeles, CA, USA, 2006. ACM.

[2] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 34(4):145–158, 2004.

[3] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Mobile Computing and Networking*, pages 243–254, 2000.

[4] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. Geographic routing made practical. In *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*, pages 217–230, Berkeley, CA, USA, 2005. USENIX Association.

[5] J. LeBrun, C.-N. Chuah, D. Ghosal, and M. Zhang. Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks. *Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st*, 4:2289–2293 Vol. 4, 30 May-1 June 2005.

[6] K. C. Lee, P.-C. Cheng, J.-T. Weng, L.-C. Tung, and M. Gerla. VCLCR: A practical geographic routing protocol in urban scenarios. Technical Report 080009, UCLA, 4732 Boelter Hall, Los Angeles, CA 90095, March 2008.

[7] I. Leontiadis and C. Mascolo. GeOpps: Geographical opportunistic routing for vehicular networks. *World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE International Symposium on a*, pages 1–6, 18-21 June 2007.

[8] C. Lochert, M. Mauve, H. Füssler, and H. Hartenstein. Geographic routing in city scenarios. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(1):69–72, 2005.

[9] J. Ott and D. Kutscher. A disconnection-tolerant transport for drive-thru internet environments. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 3:1849–1862, March 2005.

[10] R. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modeling a three-tier architecture for sparse sensor networks, 2003.