Mobile Netw Appl
DOI 10.1007/s11036-009-0181-6

# GeoDTN+Nav: Geographic DTN Routing with Navigator Prediction for Urban Vehicular Environments

**Pei-Chun Cheng · Kevin C. Lee · Mario Gerla · Jérôme Härri**



**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 packets by selecting relays with the best progress towards the destination or use a recovery mode in case such solutions fail. 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 GeoDTN+Nav, a hybrid geographic routing solution enhancing the standard greedy and recovery modes exploiting the vehicular mobility and onboard vehicular navigation systems to efficiently deliver packets even in partitioned networks. GeoDTN+Nav outperforms standard geographic routing protocols such as 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 to optimize such forwarding procedure. We finally evaluate the benefit of our approach first analytically and then with simulations. By using delay tolerant forwarding in sparse networks, GeoDTN+Nav greatly increases the packet delivery ratio of geographic routing protocols and provides comparable routing delay to benchmark DTN algorithms.

**Keywords** geographic routing · delay tolerant network · navigation interface · store-carry-forward · VANET

This paper is an extended version of the one selected as one of the best papers in ISVCS 2008.

The work is partially supported by the International Technology Alliance sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defense under agreement number W911NF-06-3-0001, and partially supported by the state of Baden-Wurttemberg, Tschira Foundation, PTV AG and INIT AG to the research group on Traffc Telematics.

P.-C. Cheng · K. C. Lee (✉) · M. Gerla
Computer Science Department, University of California, Los Angeles, CA 90095, USA
e-mail: kclee@cs.ucla.edu

P.-C. Cheng
e-mail: pccheng@cs.ucla.edu

M. Gerla
e-mail: gerla@cs.ucla.edu

J. Härri
Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany
e-mail: haerri@kit.edu

## 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.

Published online: 17 June 2009

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'. A particular 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" occurs. Several routing protocols have been proposed (GPSR [4], GPCR [9], VCLCR [7]) 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 always true. When junction nodes are missing, packets will be forwarded across junctions, causing possible routing loops. VCLCR attempts to solve this problem by detecting loops and removing cross links whenever possible. It greatly increases the packet delivery ratio compared to GPSR or GPCR.

Unfortunately, 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) [3]. DTN protocols such as [12, 13] employ such a store-carry-and-forward mechanism to forward packets yet at the cost of an increased routing delay.

Numbers of delay tolerant routing protocols exploiting different strategies to route packets have been developed. GeOpps [8] 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 [6] 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 [6], 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 a complete solution as shown in Table 1, called GeoDTN+Nav, that includes the greedy mode, the perimeter mode, and the DTN mode. In order to know when to use one of these modes, a network partition detection method is proposed to evaluate for each packet the correct forwarding method to use in order to guarantee a better packet delivery even in sparse or partitioned networks. We also introduce the *Virtual Navigation Interface* (VNI) which efficiently provides navigator predictions in order to choose the best delay tolerant forwarders. We analytically and simulatively measure the performance of our solution and illustrate how it outperforms GPSR and GPCR and manages to transmit information when they both fail. We also show the capability of GeoDTN+Nav using diverse and heterogenous information provided by VNI. The use of diverse navigational information greatly improves the packet delivery compared to single-metric DTN routing protocols like GeOpps [8]. In GeoDTN+Nav, geographic routing is employed for efficient and fast routing within network partitions, while DTN "data

**Table 1** Packet delivery concept

| Packet delivery | Connected/dense network | Sparse network |
|---|---|---|
| Geo-routing | Fast | No delivery (dropped) |
| DTN routing | Slow | Slow |
| GeoDTN+Nav | Fast | Slow |

mules" are used to ensure correct delivery between partitions yet at the cost of an increased delay.

The rest of the paper is organized as follows: In Section 3, we formally introduce the virtual navigation interface model. Section 4 describes 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. Section 2 provides a short discussion of the current efforts in geo-routing and delay tolerant forwarding. Section 7 discusses ways to deliver packets to moving vehicles as part of the future work. Finally, Section 8 concludes the paper.

## 2 Related work

We briefly describe two categories of routing protocols used in VANET, *geographic routing* and *delay-tolerant routing* since GeoDTN+Nav being a hybrid approach considering concepts from both categories cannot be compared solely with protocols of one category. For each category, we present related work to GeoDTN+Nav.

### 2.1 Geographic routing

*Greedy perimeter stateless routing* The Greedy Perimeter Stateless Routing (GPSR) [4] 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 the perimeter mode to recover from the local maximum.

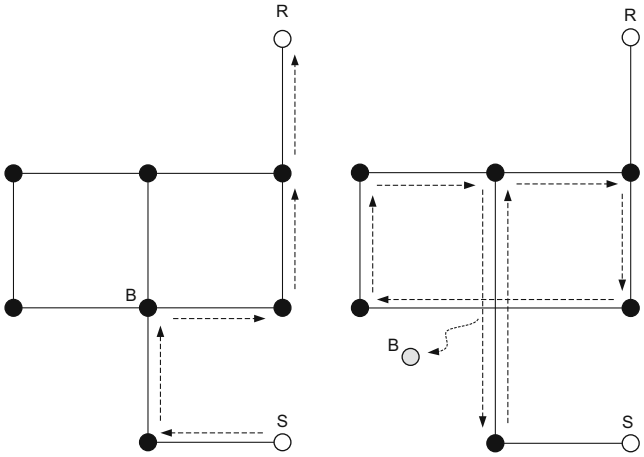
The idea of GPSR's perimeter mode is to forward packet by right-hand rule with the starting vector constraint. When a packet switches itself to the perimeter mode at an intermediate node $x$, it first draws a virtual vector from $x$ to destination node $D$. Node $x$ then forwards the packet to the first 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 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.

*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) [9] 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 1a and b illustrate an example. Originally $S$ forwards packets to $R$ along the dash line in Fig. 1a. If the junction node $B$ is missing, the packet will be forwarded cross the junction, goes back to $S$, and gets dropped shown in Fig. 1b.

*VANET cross link corrected routing protocol* Lee et al. [7] 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




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

(b) Junction B is empty (two cross links).

**Fig. 1** Cross-link causes routing loops (**a**, **b**)

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.

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 [5]. This cross-link-removal procedure is on-demand and the overhead is small. When a packet in the 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.

### 2.2 Delay-Tolerant Network (DTN) routing

This section presents only the DTN routing approaches relevant to GeoDTN+Nav. Readers can refer to [15] for an overview of the state of the art DTN routing protocols for different types of delay tolerant networks.

*Mobile Relay Protocol* (*MRP*) MRP [11] is a relay-based approach that is used in conjunction with traditional ad hoc routing protocol. A node would engage in traditional routing until a route to the destination is unobtainable. It then performs *controlled local broadcast* to its immediate neighbors. All nodes that receive the broadcast store the packet and enter into the relaying mode. Such nodes carry the packet until their buffer is full. When that happens, the relay-nodes would choose to relay the packet to a *single random* neighbor. Similar to MRP, GeoDTN+Nav combines traditional ad hoc routing and DTN routing. However, a GeoDTN+Nav node does not broadcast to its local neighbors in the DTN mode. Furthermore, the node constantly seeks the best neighbor to deliver to the destination since holding packets until the buffer is full or until the relay node meets the destination prolong the end-to-end delay.

*Context Aware Routing* (*CAR*) CAR [10] integrates *synchronous* and *asynchronous* mechanisms for message delivery. A synchronous message delivery mechanism is characterized by a contemporaneous path between the current node and the destination; whereas, an asynchronous message delivery mechanism does not have such a path. The concept is similar to the hybrid approach adopted by GeoDTN+Nav where a node switches to the DTN mode when its scoring function indicates network disconnectivity. More importantly, during asynchronous message delivery, a node relays to another node with the highest probability of reaching the destination by the evaluation and prediction of the *context* information. An utility function similar to GeoDTN+Nav's scoring function is used. However, CAR did consider weights of each contextual parameter (e.g., rate change of connectivity, battery life, etc.) dynamically. Since CAR uses DSDV for traditional ad hoc routing, it introduces prediction to reduce the overhead of dissemination of routing table. CAR provides another framework of utilizing the contextual information with dynamic-weight consideration geared towards sensor networks and prediction geared towards proactive routing.

*Model Based Routing* (*MBR*) Chen et al. [1] presents a model based routing that takes advantage of the predictable node moments along a highway. Authors have verified the hypothesis that the motion of vehicles on a highway can contribute to successful message delivery, provided that messages can be relayed and stored temporarily at moving nodes while waiting for opportunities to be forwarded further. As a result,

GeoDTN+Nav takes node movements into consideration when computing the next forwarding node in the DTN mode.

*GeOpps* GeOpps [8] is a delay tolerant routing algorithm that exploits the availability of information from a navigation system (NS). Such navigation system includes a GPS device, maps, and the function to calculate a suggested route from current position to a requested destination. In GeOpps, each vehicle equipped with an navigation system communicates with one another and obtains information to perform efficient and accurate route computation.

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 Fig. 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 $NP_c$. The weakness of the approach is that the scheme assumes all vehicles have a navigation system and the navigation system provides the same transmission format and content. The assumption is not true in reality. As a natural consequence of the design, GeOpps does not utilize heterogenous information from devices other than the navigation system and misses opportunities of finding a better forwarder. Furthermore, since it is a DTN routing protocol and packets tend to be held by nodes whose NP is closer to the destination than another node whose NP is further yet it is on the connected path to the destination, thus, nodes generally experience higher delay in GeOpps than in GeoDTN+Nav. For example, Fig. 3 shows a two-lane road segment with traffic in opposite directions. A sender node $S$ is sending packets to a curb-side destination $D$. Among $S$'s neighbors $A$ and $B$, $A$ has the closest NP to the destination $D$. Thus, in GeOpps, the packet would be delegated to $A$. Since no other nodes have closer NP to the destination, the packets would remain stored at A and only delivered to the destination when $A$ meets $D$. On the contrary, in GeoDTN+Nav, the packet would be first forwarded using geographic routing and successfully delivered at $D$ since there is a connected path from $S$ to $D$. Given that radio propagation is much faster than vehicle movement, we can expect that GeoDTN+Nav has lower latency in other similar scenarios.

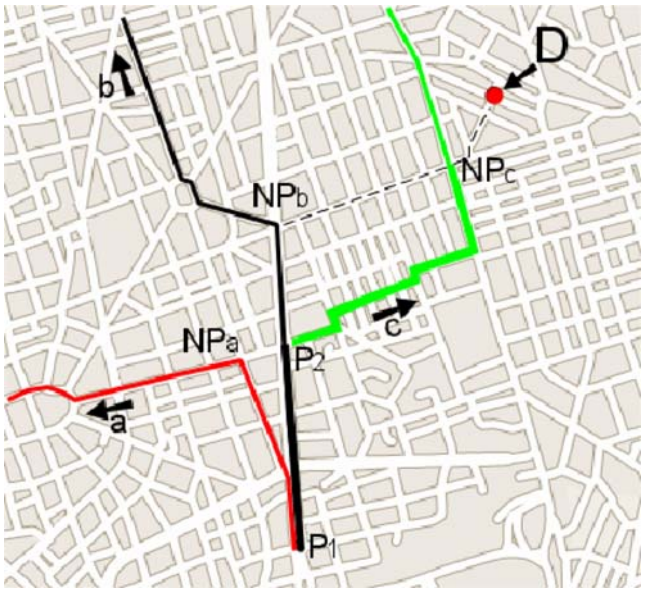



**Fig. 2** GeOpps Neighbor Selection, where their routes are evaluated with respect to the potential "Nearest Poin" (NPx) to the destination D

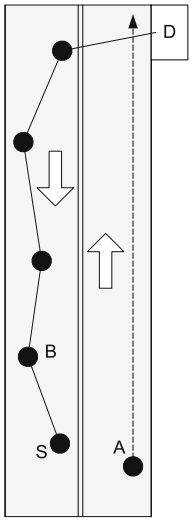



**Fig. 3** GeOpps Problem, where node A is chosen as DTN mule whereas a geo-routing would have used the connected graph and selected node B for a faster progress

## 3 Virtual navigation interface framework

The goal of the Virtual Navigation Interface (VNI) is to help discover neighboring vehicles that can 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 make a better decision.

In [8], 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 with on-board navigation systems. In addition, modern 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 classes of vehicles, if these latter vehicles are even equipped with such devices. For example, road identification can differ from one navigation system to another. The map encoding of a road on one navigation system may define a road as one separated by junctions; whereas, the map encoding of a road on another navigation system may define a road naturally from the name of the road.

In GeoDTN+Nav, we adopt a more relaxed and generalized assumption and provide a unified framework for the different kinds 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 Fig. 4 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

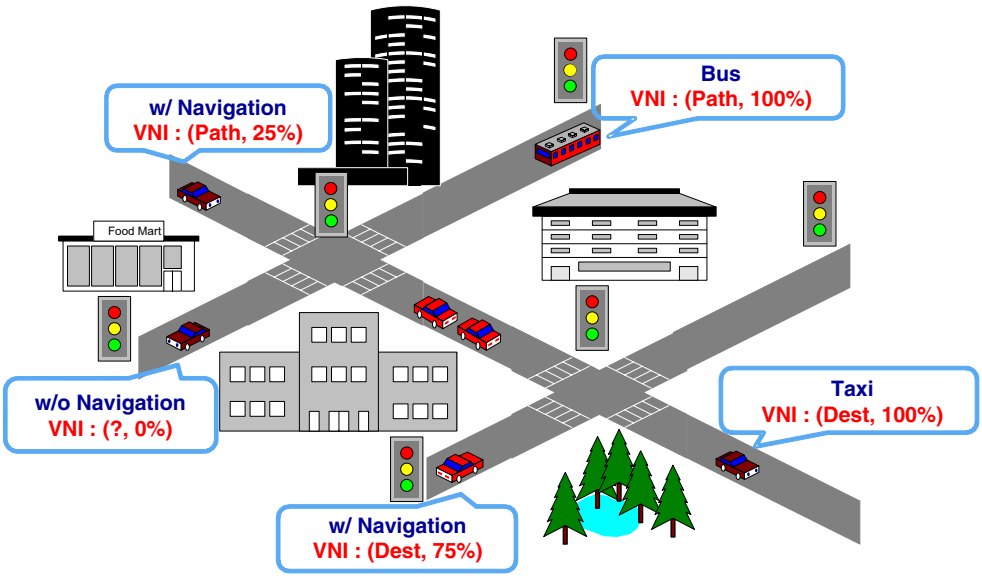



**Fig. 4** Categories of vehicular route pattern and VNI example

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:

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 2.

### 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 Fig. 5, 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 preconfigured 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 Fig. 4, we installed VNI on every vehicle:

- 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

**Table 2** Categories of vehicular route pattern

| Categories | Examples |
|---|---|
| Deterministic (fixed) route | Metro bus, metro train, campus shuttle |
| Deterministic (fixed) destination | Taxi, van pool |
| Probabilistic (expected) route/destination | Navigation system guided vehicles |
| Unknown | Non-random movement |

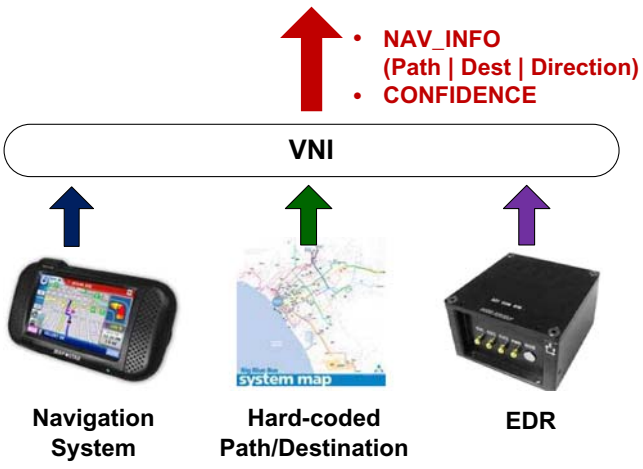



**Fig. 5** Virtual navigation interface

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 the greedy mode, and the second mode is the 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 the greedy mode, and then by the 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 6 illustrates the transition diagram between these three modes.

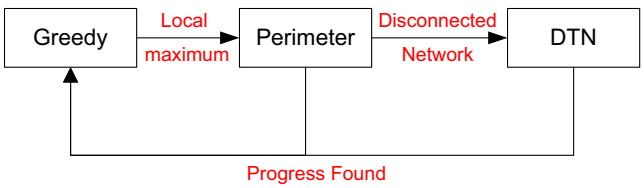



**Fig. 6** Switch between greedy, perimeter, and DTN mode

Two questions arise in this scheme: Exactly when should we switch to DTN mode, and when to switch back to the greedy mode. For the former, we will use a cost function and a threshold related to a network partition detection and to the quality of nodes mobility pattern between partitions. For the latter, similar to the recovery mode, we will return to the 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

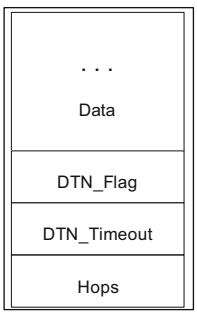
In GeoDTN+Nav, the default greedy forwarding strategy is the same as the restrictive greedy forwarding in GPCR, where packets are always forwarded between junction nodes as junctions are the only places where a node can make significant routing decisions. This remains true even if a current forwarding 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 default 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 Fig. 7:

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.

**Fig. 7** Packet format



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 the perimeter mode.
   GeoDTN+Nav uses this information to determine if the network is disconnected. This field can be replaced or augmented if future work adopts 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 the 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 less favored relay to carry a packet.

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

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

where:

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

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 the 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 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 the perimeter mode before switching to DTN mode. If a packet goes back to the point where it entered the 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 Fig. 8:

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 the 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 the 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 Fig. 9, 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 Fig. 9, 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:

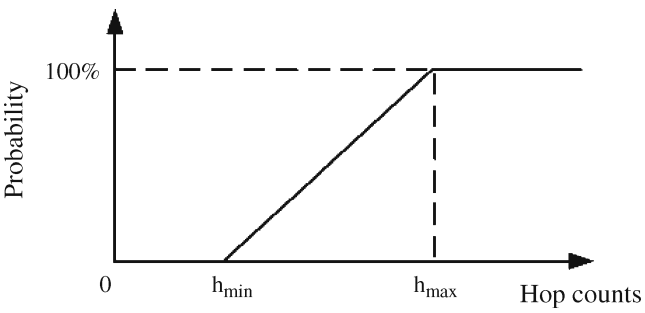



**Fig. 8** Function P(h)

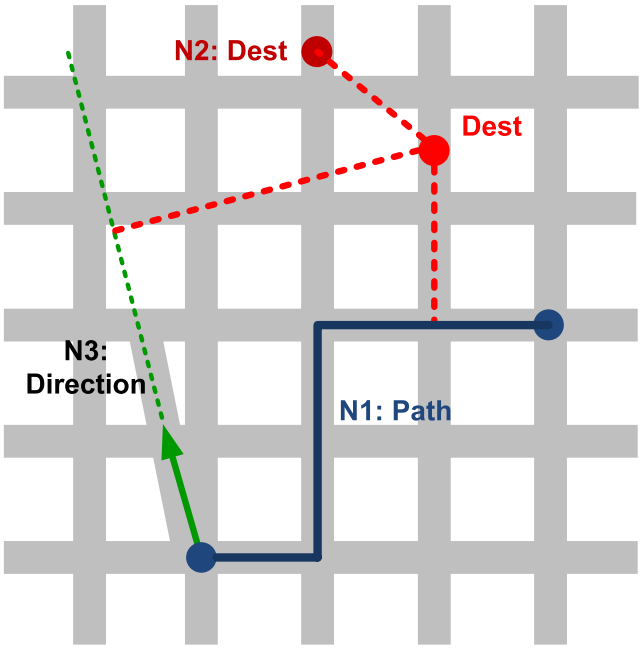



**Fig. 9** Calculate $Q(N_i)$

**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** $\theta$ equals 0
3. **then return** 1
4. **else return** $\frac{1}{abs(\theta)}$

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 the greedy mode, until it reaches a local maximum.
3. Then it switches to the perimeter mode and record its own location $e0$ and its $dest$ in the packet header.

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

   (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 Eq. 1.
   (e) If one of the scores is greater than $S_{thresh}$, 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 the 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. Increase the hop count. If the hop count reaches the TTL and there is no node with a score greater than $S_{thresh}$, 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.

Note that, in GeoDTN+Nav, each node makes intelligent decision based on the navigation information which may not always be reliable. For example, even buses with fixed routes might detour to another route because of temporary road construction. However, notice that GeoDTN+Nav only exploits DTN forwarding to recover network separation. A detoured vehicle might carry packets to another connected network. Still, it is true that a detoured vehicle would strictly move away from the destination so the packets have to be eventually dropped. In this case, VNI would automatically decrease this vehicle's confidence value, which in turn makes this vehicle less favorable in the future DTN forwarding.

Lastly, we would like to emphasize that GeoDTN+Nav is a distributed routing protocol, which can not guarantee the packet delivery. Depending on application requirements, upper layer protocols may implement their own mechanisms to acknowledge end-to-end packet delivery. However, because of the dynamic and evolving nature of vehicle networks, we believe that GeoDTN+Nav is more suitable for connectionless message-based applications.

### 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 the 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 the perimeter mode and each node has already collected navigation information broadcasted by the VNI installed on its neighbors.

#### *4.4.1 Example 1: greedy to DTN*

Assume weight parameters $\alpha$, $\beta$, and $\gamma$ are 1, and the threshold $S_{thresh}$ is 0.25. Also suppose that a packet has traversed 8 hops in the 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 the 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 the 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 the greedy mode. Once it has reached that a point, the packet is forwarded to destination in the greedy mode again (Fig. 10).

#### *4.4.2 Example 2: DTN to greedy*

The second example, depicted in Fig. 11, illustrates the condition for a node to switch from DTN mode to the greedy mode. In this example, a packet first hit the local

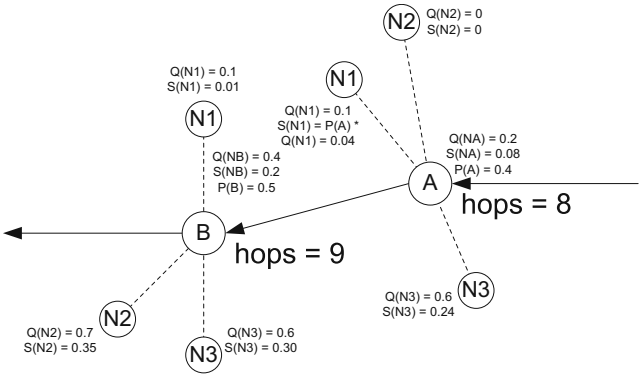



**Fig. 10** Routing example 1

maximum at point *e*0 where it switched to the 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 the 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 *e*0 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 *e*0 to *Dest*. Therefore, if a neighbor is located at that point, node *A* is able to switch back to the 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.

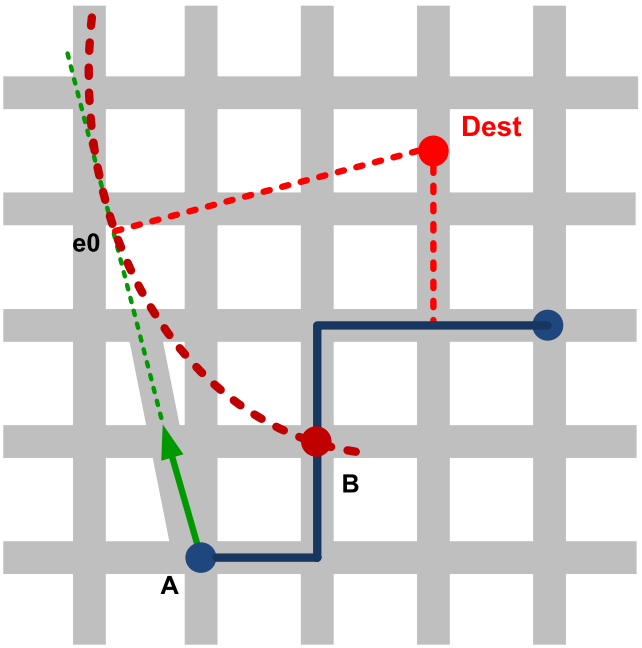



**Fig. 11** Routing example 2

## 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 networks, 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 the perimeter mode. If the perimeter mode designed by GPCR 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 3 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 probability of a disconnect network after a packet is forwarded for $x$ hops, and $S_{thresh}$ 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 of switching to DTN mode after $x$ hops.

The other variable $G_x$ is defined as follows: When a packet is forwarded in the perimeter mode at $x$th hop, the probability that it switches to the greedy mode at next hop is $G_x$ and the probability that it remains in the 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

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

**Table 3** Notation in analytical model

| | |
|---|---|
| $G_x$ | The probability of switching to the greedy mode after x hops in the perimeter mode |
| $N_x^R$ | The expected number of type R neighboring nodes after x hops in the 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 $Q(Nx) = k$ |
| $S_{thresh}$ | The threshold of switching to DTN |

is the probability that a packet can switch back to the greedy mode after it enters the perimeter mode. Here, $G_0 = 0$; that is, the probability of switching to the greedy mode after $x$ hops in the perimeter mode is 0. Intuitively, $G_0 = 0$ says if a node has not tried to forward in the perimeter mode, it should try at least one hop before considering switching to the greedy mode. Since the inability to switch back to the greedy mode means the network is either disconnected, or the packet loops in the perimeter mode, the probability that network is disconnected is at *most*

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

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 the 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.

The definition of $T_x$ is similar to $G_x$. When a packet is on its $x$th hop in the 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:

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

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:

$$F_x^R(S_{thresh}) = \left( \int_0^1 \int_0^{S_{thresh}/\alpha\beta\gamma z P(x)} P_Q^x(y) dy dz \right)^{N_x^R}$$

Therefore, $T_x$ can be written as follows:

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

$F_x^R$ 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 the perimeter mode.

Figure 12 demonstrates our framework about this analytical model. When a packet is forwarded in the perimeter mode, it has three choices. If any of its neighbors is closer to destination than the starting point in the 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 the perimeter mode. Given $T_x$ and $G_x$, the probability that a packet can successfully exit the perimeter mode in $x$ hops is:

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

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

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

while:

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

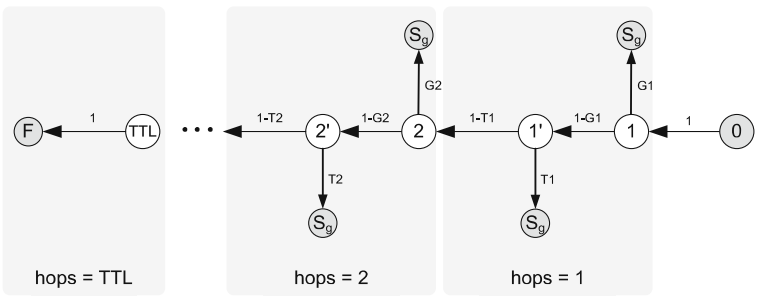



**Fig. 12** Markov Chain for the perimeter mode forwarding

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

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

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

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:

$$\sum_{k=1}^{x} G_xI_xD_g + \sum_{k=1}^{x}(1-G_x)T_xI_xD_d$$

Figures 13 and 14 illustrate the PDR vs. latency tradeoff as a function of the probability of switching to DTN $T_x$, where the probability of switching to the 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 Fig. 13, 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, Fig. 14 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

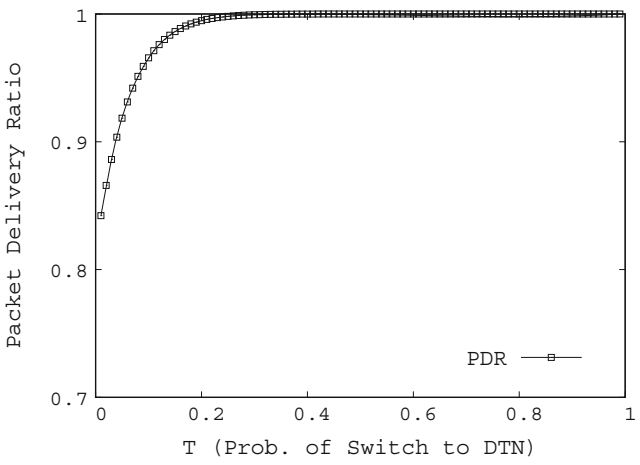



**Fig. 13** Probability vs. PDR

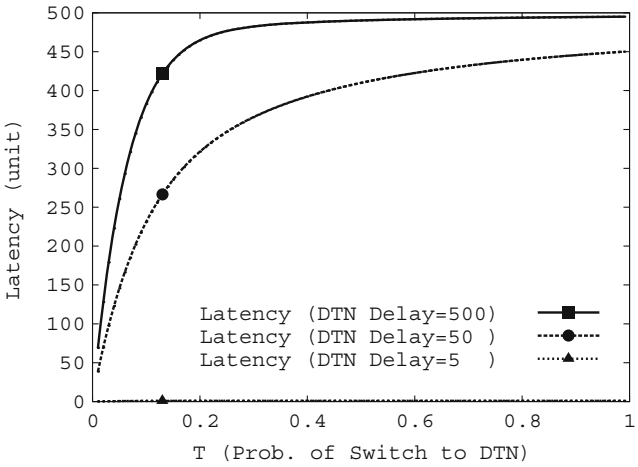



**Fig. 14** Probability vs. latency

in the perimeter mode should switch to DTN for a probability ranging from $a$ to $b$, we have:

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

Thus,

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

which leads to:

$$\prod_{\forall R} F_x^R(m)$$

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

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

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

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:

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

Figure 14 demonstrates another aspect of our analytical model. Suppose there are three different types of $R$ nodes, and each of them has different DTN delay, $D_d$ of 5, 50, 500, respectively. If the network application

wishes to have a latency lower than 200 s 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 s. In this case the packet delivery ratio becomes the only constraint that is of concern. Once the probability of switching to DTN is obtained, $S_{thresh}$ can be calculated by the method mentioned above.

## 6 Performance evaluation

In this section, we are going to evaluate the performance of GeoDTN+Nav in two different scenarios: a synthetic one that aims at illustrating the concept of GeoDTN+Nav and a realistic one that consists of a real urban map and realistic vehicular mobility. Table 4 indicates the parameters we used in our simulations.

GeoDTN+Nav is compared against a randomized DTN routing scheme [14], $Rand - DTN$. RandDTN works as follows. At each beacon interval, a node forwards the packet it is carrying with probability $p$. When $p = 0$, RandDTN is reduced to direct transmission scheme where packets reach the destination only when the source node meets the destination node. When $p = 1$, a node *always* considers its neighbors to forward the packet. To avoid the packet from being forwarded to *any* node, thus reducing progress towards the destination, we modify RandDTN so that the node would forward to its neighbor whose final destination is closest to the destination of the packet. If such a neighbor does not exist, the node would simply store and carry the packet until the next beacon interval. Moreover, in this paper, we set $p = 0.5$ for generality.

### 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 Fig. 15, 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 300 m, and the transmission range is 300 m. According to the topology, each packet sent from the source to the destination will reach a local maximum and switch to the perimeter mode.

**Table 4** Simulation parameters

| Parameter | Value |
|---|---|
| Network simulator | Qualnet 3.95 |
| Mobility simulator | VanetMobisim |
| CBR rate | 64 Bytes/s |
| 802.11b rate | 2 Mbps |
| 802.11b transmission power | 15.0 dBm |
| TX range | 300 m |
| Avg vehicle speed | 50 km/h |
| Simulation runs | 30 |
| Confidence interval | 95% |
| Propagation | Free space with inter-road radio blocking model |

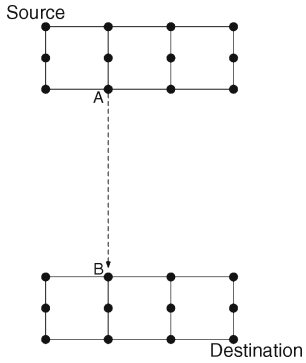
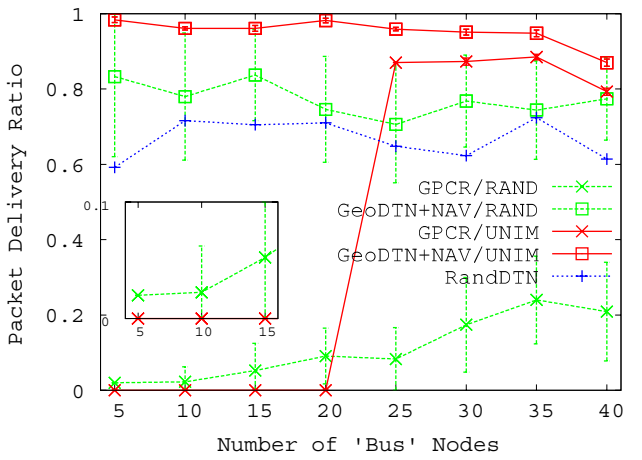



**Fig. 15** Synthetic topology

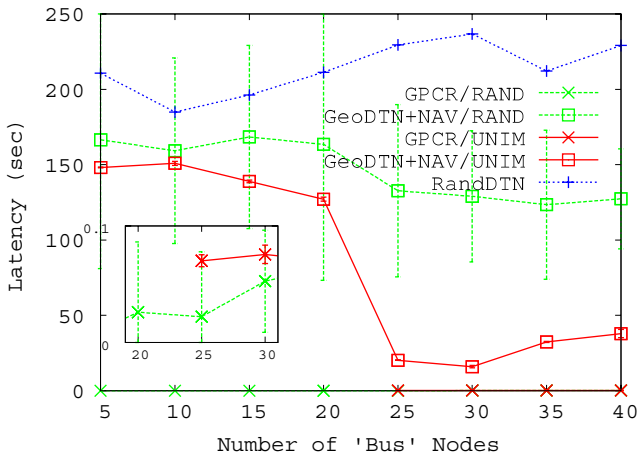
We place 'Bus' nodes at location A. 'Bus' nodes move towards location B at a speed of 50 km/h. We manipulate the number of bus nodes as well as their departure pattern in order to study the virtual connectivity between the two partitions. More precisely, we compare two departure patterns: a *uniform 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.

Simulations are ran with constant bit rate UDP traffic with packet size of 64 bytes and compare GeoDTN+Nav with GPCR and RandDTN in packet delivery ratio, latency, and hop count. For brevity, we only show results of RandDTN for random departure. We set $\alpha = \beta = \gamma = 1$. Based on the topology, packets are transferred from one partition to the next by buses. Therefore, the delay experienced in the DTN mode is the time it takes for a bus to arrive from one end of the partition to another. We set $D_d = 450$ s.[1] Furthermore, because the end-to-end latency is mostly attributed to the delay in the DTN mode, it is set to $D_d$ of 450 s. We also configure the probability of switching to the DTN mode to approximately 90%. Based on these values and according to our study in Section 5, the parameter $S_{thresh}$ is computed to be 0.1. Note that this value of $S_{thresh}$ will also be used in the rest of this paper as we would like to keep a similar DTN latency. The results of the synthetic scenario are depicted in Fig. 16.

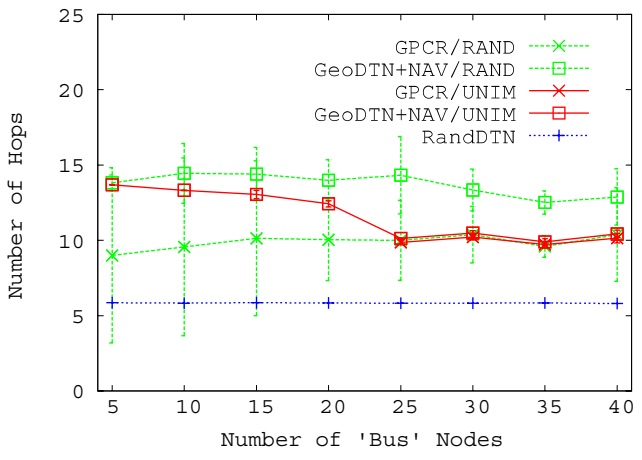
[1]Since bus arrival varies depending on its departure pattern—random or uniform, $D_d$ ranges anywhere from 150 s to 450 s. $D_d$ is set to 450 s conservatively.




(a) PDR




(b) Latency




(c) Hop count

**Fig. 16** Performance evaluation for the synthetic scenario (**a**–**c**)

In Fig. 16a, 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 delivery 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. Note that RandDTN can also achieve over 60% delivery ratio. This is because, in this simple synthetic network configuration, there is sufficiently high probability for source nodes to meet 'Bus' nodes, which can help deliver packets to the destination.

Now considering the second metric set, Fig. 16b and c show the average number of hops and latency a delivered packet travels. We may clearly see the trade-off with GeoDTN+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. Last, we can also see that RandDTN has higher latency. This is because RandDTN is a pure DTN forwarding protocol which relies on nodes' physical movement to carry packets. This is expected to be much slower than wireless communication.

## 6.2 Realistic scenario

In this particular experiment setup, realistic vehicular mobility traces have been generated using the Intelligent Driver Model with Intersection Management (IDM-IM) by VanetMobiSim [2], 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 300 m. The urban topology employed in this paper is a realistic 1500 m by 4000 m 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, and 3) hop count. We also show in the figures the 95% confidence interval.

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 pairs are randomly distributed. In Fig. 17a, 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 partitioned network. We may also see in Fig. 17a 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 Fig. 17c, GPSR and GPCR only successfully deliver packets when the source and destination nodes are one hop away, which also results in low latency. In Fig. 17b and c, as the number of buses increases,

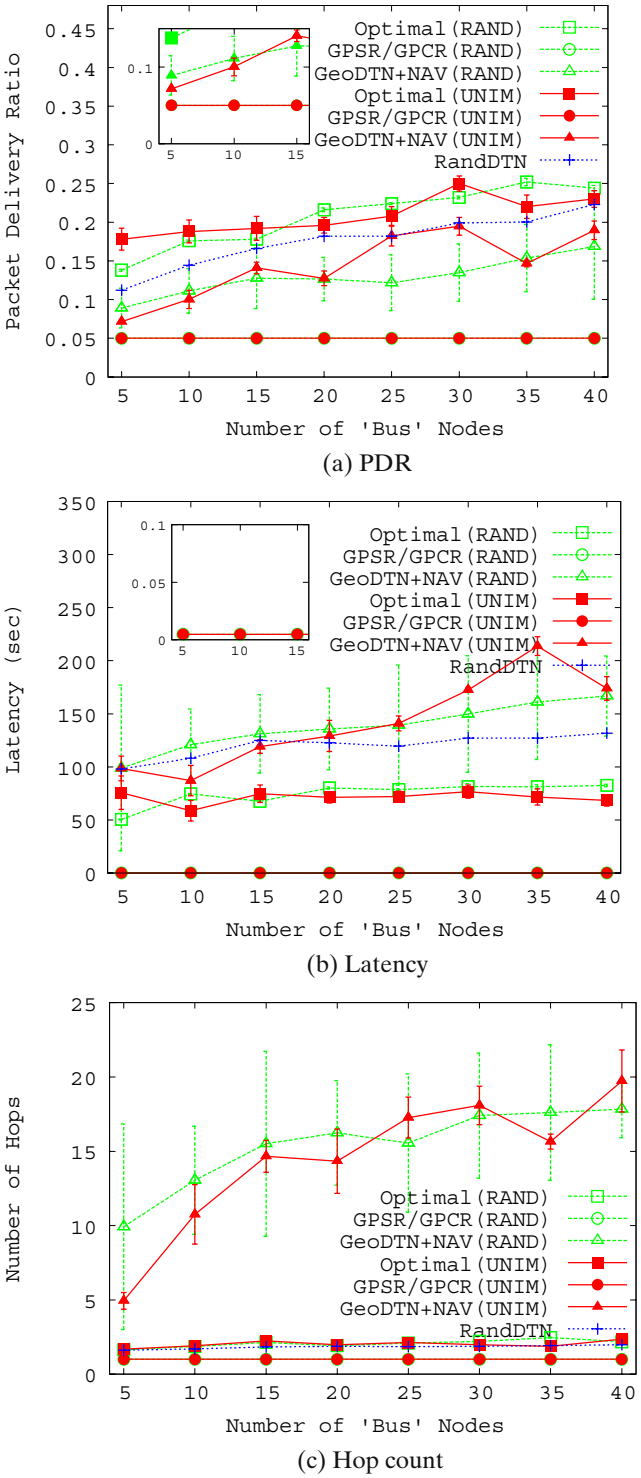



**Fig. 17** Performance evaluation for the realistic scenario (**a**–**c**)

GeoDTN+Nav's hop count and latency increase. This is GeoDTN+Nav's fundamental tradeoff between packets' forwarding latency and delivery ratio.

Last, note that RandDTN achieves slightly better PDR and lower latency than GeoDTN+Nav. This is

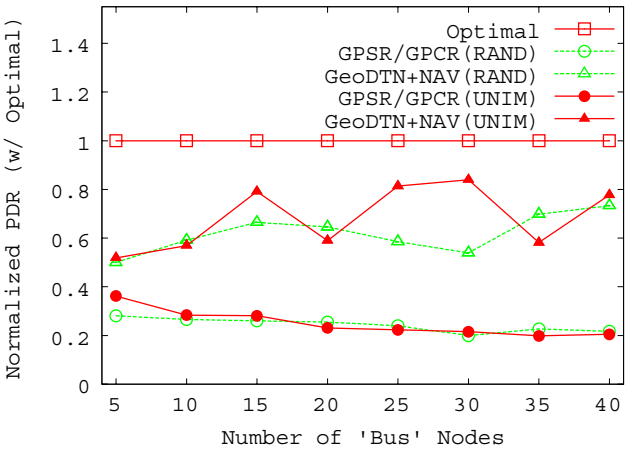
because of the intelligent feature of GeoDTN+Nav: whenever the packet is carried across partitioned network, GeoDTN+Nav would try to switch back to geographic routing. However, in such a sparse network, GeoDTN+Nav is likely to fall back to DTN mode again, which increases the latency and might also decrease the PDR. However, owing to the dynamic and evolving nature of vehicular networks, we expect that this hybrid geo-rouing and DTN forwarding nature of GeoDTN+Nav could yield better performance in general.

### *6.2.1 Benchmarking with optimal routing protocol*

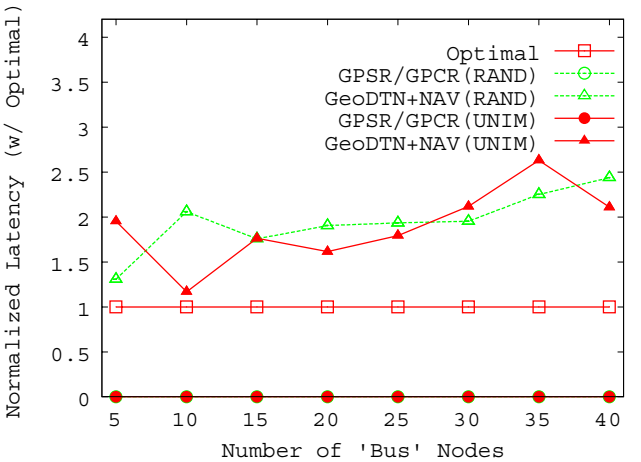
A unique feature of GeoDTN+Nav is its hybrid routing feature. Based on the "score function", GeoDTN+Nav make intelligent decisions on switching between position-based routing and delay tolerant routing. Hence, the performance of GeoDTN+Nav highly depends on the correctness of the score function. In this section, we further evaluate GeoDTN+Nav by benchmarking it with an optimal unicast routing protocol.

Here, we define an imaginary optimal unicast routing protocol as a protocol that can always forward packets to the destination node with the fewest hops or lowest latency. In other words, an optimal unicast routing protocol acts as an oracle which forsees all node encounters in the future and construct a shortest routing path beforehand. Since such a protocol is not practical, in this simulation, we use a augmented flooding protocol as an approximation. In the modified flooding protocol, when a node receives a packet, it buffers the packet and copies to every new node it encounters onward. For multiple copies of the same packet, we only record the latency and hop counts of the first copy which successfully arrives at the destination. The idea is that if an optimal protocol does exist, it would unicast the packet exactly following the traversing path of this first packet copy. In addition, we limit the buffer size on each node to 20 packets, which is derived from the average buffer usage of GeoDTN+Nav.[2] Notice that, compared with any other unicast protocols, this flooding protocol guarantees the highest packet deliver rate because of its broadcast nature. It also guarantees the lowest latency because we measure the latency based on the first arrived packet. However, it does not necessarily guarantee the lowest hop counts, since this first packet would traverse multiple low-latency hops rather than fewer high-latency hops.

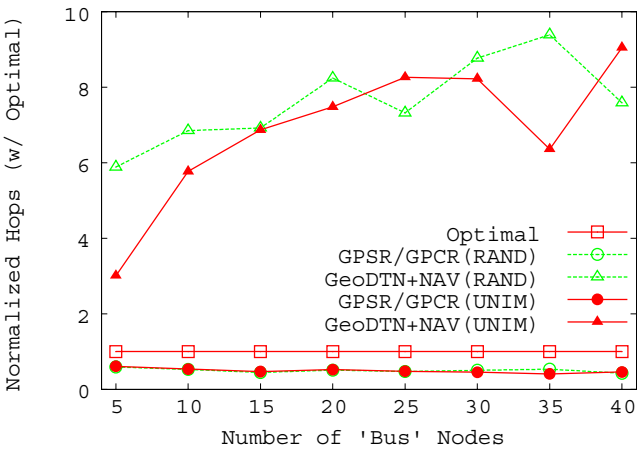
The result is shown in Fig. 17a. The optimal (flooding) protocol yields better PDR than GeoDTN+Nav as




(a) Normalized PDR.




(b) Normalized Latency.




(c) Normalized Hop count.

**Fig. 18** Performance evaluation for the normalized realistic scenario (**a–c**)

[2]The average buffer usage of GeoDTN+Nav is derived from the grand average queue size on each node throughout all simulations.

**Table 5** Summary of normalized performance evaluation with respect to "optimal routing"

| Protocols | Min | Max |
|---|---|---|
| GeoDTN+Nav | 50% | 84% |
| GPSR/GPCR | 19% | 36% |

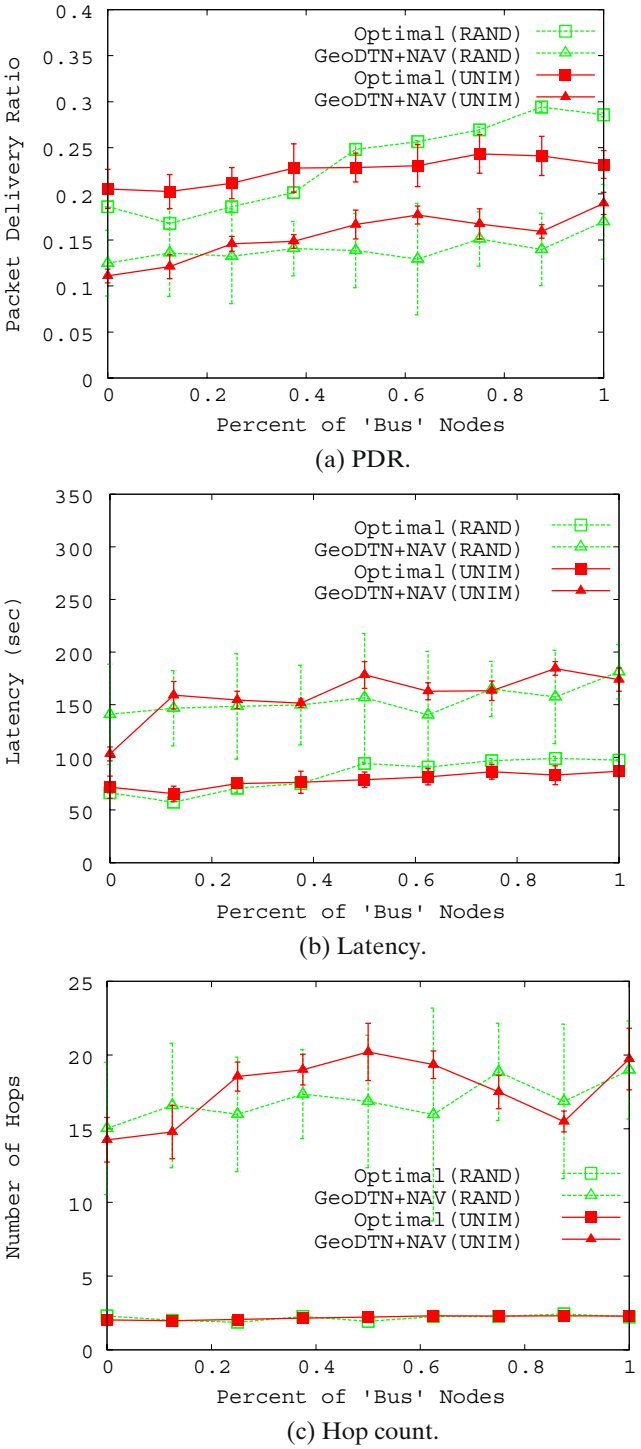
expected. In fact, we can take the result of the flooding protocol as the upper bound of any unicast routing protocols under the same configuration. Figure 18a shows the PDR normalized with the PDR of the flooding protocol. We can see that GeoDTN+Nav can achieve up to 80% of the highest possible PDR, as opposed to GPSR/GPCR, which only achieves around 30%. The downward trend of GPSR/GPCR's normalized PDR reflects the fact that the increasing number of buses helps flooding and GeoDTN+Nav but not enough to help GPSR/GPCR. The normalized performance is summarized in Table 5.

Figures 17b and 18b show the actual and normalized latency. Notice that the deliver latency of GeoDTN+Nav is twice as large as the optimal latency. This is because whenever a node in GeoDTN+Nav makes a wrong switching decision, it might miss the bus node and has to wait for the next encounter of another bus node, which inadvertently increases the end-to-end latency.

Finally, Figs. 17c and 18c show the actual and normalized hop count. Both GPSR and GPCR possess lower hop count and latency than the optimal protocol because most of the successful delivery for GPSR and GPCR are one hop away. The optimal routing's higher PDR verifies this claim: packets that have to be delivered cross partitions are simply dropped by GPSR and GPCR; whereas, the optimal routing would store the same packets and continue broadcasting them until they reach their destination. However, the optimal routing's latency and hop count are lower than GeoDTN+Nav's as GeoDTN+Nav packets will have to travel for more hops before switching to DTN mode. This is because we use a simple scoring function to guide the mode change in GeoDTN+Nav. The scoring function can be improved to choose a better neighbors or to remain longer in DTN mode, thereby reducing the number of hops or latency. We leave this tunable parameter as future work.

### *6.2.2 Heterogeneity*

In this section, we introduce 'Taxi' nodes as well as 'Bus' nodes. We fix the total number of "Data Mules" (Buses and Taxis). We first let all data mules be taxis, then we gradually replace taxis with bus nodes by 5-node increment up to 40 bus nodes and 0 taxi nodes. For taxi nodes, the VNI only broadcasts its destination coordinates. When there are only taxi nodes in the




(a) PDR.

(b) Latency.

(c) Hop count.

**Fig. 19** Performance evaluation for the heterogenous scenario. *X*-axis indicates the percent of 'bus' nodes out of 40 data mules. *x*-axis = 0 may emulate GeOpps while larger x-axis values show the benefits of GeoDTN+Nav

network, GeoDTN+Nav approximates the behavior of GeOpps given that the routes of taxis are known. As there are more buses in the network, GeoDTN+Nav enjoys the diversity of options in forwarding packets. Figure 19a shows that as the number of buses increases, the PDR increases, indicating the delivery quality of buses is better than the taxis'. By further investigation, we found that even though some taxi's destination is near the packet's destination, from the networking point of view, the taxi's destination and the packet's destination are disconnected. There are also taxis which may have a destination that is far from the packet's destination even though they pass the packet's destination. By only looking at the destination coordinates may lead to suboptimal decisions.

The results shed light on the advantage of GeoDTN+Nav over GeOpps which relies only on taxis to deliver packets[3] and yields lower PDR of 13% and 12% than GeoDTN+Nav's in random and uniform bus departure, respectively. By incorporating simply a few bus nodes, GeoDTN+Nav is able to use both types of vehicles to deliver packets successfully.

## 7 Future work

### 7.1 Moving destination

Moving destination is a common issue in geographic routing, especially for vehicular networks in which nodes move with comparatively higher speed. For routing protocols that only have knowledge of the positional information, a best effort solution to moving destination is either to drop the packet or query the latest destination's location and forward the packet towards destination's latest location again. A more aggressive solution may even allow immediate nodes to query and update the location of packet's destination. However, these solutions inadvertently introduce additional query delay and network traffic.

In GeoDTN+Nav, we propose solutions exploiting the navigation information. Two approaches are considered:

1. Passive Tracking: In this approach, packets are first forwarded to the destination location. If the destination node has moved away, these packets are forwarded following the destination node's moving trajectory to try to catch up with the moving vehicle.
2. Active Predicting: In this approach, the destination node's moving path is encoded in the packet. Based on this information, node's moving speed, and the time it takes to forward the packet to the node at the old location, intermediate nodes can predict and recalibrate destination's location as they participate in forwarding the packet. The packet then will eventually meet at where the moving vehicle is at. This approach is different from the solution described above because now intermediate nodes do not have to do excessive location queries.

In this paper, we focus on developing an integrated routing architecture and assume destination is static. We address the problem of moving destinations in our future work.

### 7.2 Privacy issue

There may also be a privacy issue in VNI since it could possibly expose users' private commuting behavior. However, notice that VNI is designed for a wide range of vehicles. Some vehicles, especially of public transportation, have much lower privacy concern than others. Still, we can consider the following approaches to preserve privacy:

1. Exchange Partial Information: In this approach, VNI only exchange partial navigation information. For example, instead of giving out the whole navigation path, VNI may choose to only advertise a path segment. The idea here is that VNI simply exchange less information than it has.
2. Introduce Noise: In this approach, VNI can artificially introduce controlled errors in the navigation information. For example, a vehicle can advertise a bogus destination which is miles away from its real destination and reduce the confidence accordingly. This would reduce the probability that other vehicles handover packets to this vehicle to protect this vehicle's privacy.

Notice that there is a tradeoff between privacy and the routing performance. In GeoDTN+Nav, we are assuming a cooperative environment in which users are willing to exchange private information in return of better packet deliver. If this is not the case, then GeoDTN+Nav would simply fall back to the conventional geographical routing. Moreover, GeoDTN+Nav only exploits navigation information to recover for the

[3] GeOpps cannot consider buses because buses do not have navigation systems and therefore cannot provide information to GeOpps.

network separation, even partial or noise information might be sufficient for this purpose. We would address this issue in our future work.

Last but not least, in this paper, we focus on applications that can tolerate some amount of delay. Future work also includes improving the delivery guarantee for real-time applications by resorting to alternative routing solutions, such as different communication technologies (satellites) or roadside infrastructure.

## 8 Conclusions

In this paper, we proposed a hybrid Geo-DTN routing solution called *GeoDTN+Nav*, which incorporate the strength of both Geo-routing and DTN forwarding.

First, for sparse or partitioned networks, GeoDTN+Nav improves the packet delivery for delay tolerant applications by exploiting the vehicular mobility and on-board vehicular navigation systems to carry packets between partitions. GeoDTN+Nav outperforms conventional Geo-routing, such as 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. Then, 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.

Second, for dense or connected networks, GeoDTN+Nav simply falls back to Geo-routing and deliver packets through radio propagation, This is much faster than pure DTN approaches, which deliver packets through physical node mobility.

In conclusion, we have presented an efficient hybrid routing framework for both partitioned and connected vehicular environments based on vehicular mobility that manages to deliver packets.

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



## References

1. Chen ZD, Kung H, Vlah D (2001) Ad hoc relay wireless networks over moving vehicles on highways. In: MobiHoc '01: proceedings of the 2nd ACM international symposium on mobile ad hoc networking & computing. ACM, New York, pp 247–250
2. Härri J, Filali F, Bonnet C, Fiore M (2006) VanetMobiSim: generating realistic mobility patterns for vanets. In: VANET '06: proceedings of the 3rd international workshop on vehicular ad hoc networks. ACM, Los Angeles, pp 96–97
3. Jain S, Fall K, Patra R (2004) Routing in a delay tolerant network. SIGCOMM Comput Commun Rev 34(4):145–158
4. Karp B, Kung HT (2000) GPSR: greedy perimeter stateless routing for wireless networks. In: Mobile computing and networking, pp 243–254
5. Kim Y-J, Govindan R, Karp B, Shenker S (2005) Geographic routing made practical. In: NSDI'05: proceedings of the 2nd conference on symposium on networked systems design & implementation. USENIX Association, Berkeley, pp 217–230
6. LeBrun J, Chuah C-N, Ghosal D, Zhang M (2005) Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks. In: Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st, vol 4, 30 May–1 June 2005, pp 2289–2293
7. Lee KC, Cheng P-C, Weng J-T, Tung L-C, Gerla M (2008) VCLCR: a practical geographic routing protocol in urban scenarios. Technical report 080009, UCLA, 4732 Boelter Hall, Los Angeles, CA 90095, March 2008
8. Leontiadis I, Mascolo C (2007) GeOpps: geographical opportunistic routing for vehicular networks. In: World of wireless, mobile and multimedia networks, 2007. WoWMoM 2007. IEEE international symposium on a, 18–21 June 2007, pp 1–6
9. Lochert C, Mauve M, Füssler H, Hartenstein H (2005) Geographic routing in city scenarios. SIGMOBILE Mob Comput Commun Rev 9(1):69–72
10. Musolesi M, Hailes S, Mascolo C (2005) Adaptive routing for intermittently connected mobile ad hoc networks. In: WOWMOM '05: proceedings of the sixth IEEE international symposium on world of wireless mobile and multimedia networks. IEEE Computer Society, Washington, DC, pp 183–189
11. Nain D, Petigara N, Balakrishnan H (2004) Integrated routing and storage for messaging applications in mobile ad hoc networks. Mob Netw Appl 9(6):595–604
12. Ott J, Kutscher D (2005) A disconnection-tolerant transport for drive-thru internet environments. In: INFOCOM 2005. 24th annual joint conference of the IEEE computer and communications societies. Proceedings IEEE, vol 3, pp 1849–1862

13. Shah R, Roy S, Jain S, Brunette W (2003) Data mules: modeling a three-tier architecture for sparse sensor networks. http://www.ee.washington.edu/research/funlab/Publications/2003/data_mules.pdf
14. Spyropoulos T, Psounis K, Raghavendra CS (2004) Single-copy routing in intermittently connected mobile networks. In: Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on, October 2004, pp 235–244
15. Zhang Z (2006) Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. IEEE Commun Surv Tutor 8(1):24–37