An Adaptive Routing Protocol Based on QoS and Vehicular Density in Urban VANETs

Yongmei Sun, Shuyun Luo, Qijin Dai, and Yuefeng Ji

State Key Laboratory of Information Photonics and Optical Communications, Beijing University of Posts and Telecommunications, Beijing 100876, China

Correspondence should be addressed to Yongmei Sun; ymsun@bupt.edu.cn

Received 8 December 2014; Accepted 3 March 2015

Copyright © 2015 Yongmei Sun et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Multihop data delivery between vehicles is an important technique to support the implementation of vehicular ad hoc networks (VANETs). However, many inherent characteristics of VANETs (e.g., dynamic network topology) bring great challenges to the data delivery. In particular, dynamic topology and intermittent connectivity make it difficult to design an efficient and stable geographic routing protocol for different applications of VANETs. To solve this problem, the paper proposes an adaptive routing protocol based on QoS and vehicular density (ARP-QD) in urban VANETs environments. The basic idea is to find the best path for end-to-end data delivery, which can satisfy diverse QoS requirements by considering hop count and link duration simultaneously. To reduce the network overhead furthermore, ARP-QD adopts an adaptive neighbor discovery algorithm to obtain neighbors’ information based on local vehicular density. In addition, a recovery strategy with carry-and-forward is utilized when the routing path is disrupted. Numerical simulations show that the proposed ARP-QD has higher delivery ratio than two prominent routing protocols in VANETs, without giving large compromise on delivery delay. The adaptivity of ARP-QD is also analyzed.

1. Introduction

With the development of wireless technologies and dedicated short-range communication technologies, vehicular ad hoc networks (VANETs) have been paid increasing attention [1]. In vehicular settings, the availability of navigation system, global positioning system (GPS), and other sensors that can perceive the vehicle speed, location, and other useful information makes it possible to exploit many applications, such as intelligent transportation system (ITS) applications and infotainment applications [2, 3]. ITS applications include cooperative traffic monitoring, traffic control, blind crossing, collision prevention, nearby information services, and real-time detour route computation [4], which have attracted attention from many car manufacturers, research institutes, and national transportation departments. Vehicle communications [5, 6] are the basic foundation of the above applications of VANETs.

Unfortunately, the traditional wireless technologies cannot be applied for VANETs directly, since they have some inherent characteristics, such as dynamic radio environments and frequent topology changes, which cause the network disconnection from time to time. Due to high speeds of vehicular movements, link duration between two vehicles is hard to keep stable for a period of time. As communication relays or information broadcasters, the equipment of roadside-units (RSUs) can help improve the vehicle communications. However, the RSUs usually have high costs. Therefore, the dynamic network topology is the most critical issue in VANETs. In particular, it brings significant challenges for designing an efficient and stable geographic routing protocol.

The existing routing protocols lack the friendly adaptation to diverse QoS requirements of different applications. The objectives of the current routing protocols focus on either the fastest path with the minimum hop count or the most stable path with the longest link duration or connectivity but neglect the adaptive balance of the routing protocol with consideration of path efficiency and path stability. In this paper, we propose an adaptive routing protocol based on QoS and vehicular density (ARP-QD) over urban VANETs. It balances the path efficiency and path stability by an optimal forwarding algorithm and an adaptive neighbor discovery
algorithm with friendly adaptation to different QoS requirements and urban VANETs environments. The main intellectual contributions of this paper are summarized as follows.

(1) For describing the dynamic link quality in VANETs, we define two new metrics, named product of connectivity and distance (CDP) and segment selection weight (SSW), by considering the hop count and link duration simultaneously.

(2) We present an optimal forwarding algorithm based on CDP and SSW, which can obtain a qualified path to satisfy the diverse QoS requirements of different applications by balancing the path efficiency and path stability. As an essential part, a quick recovery strategy with carry-and-forward is also provided when the routing path is disrupted.

(3) To reduce the network overhead and improve resource usage, we propose an adaptive neighbor discovery algorithm to obtain the neighbors’ information based on local vehicular density.

(4) The extensive simulation results show that the proposed ARP-QD has higher delivery ratio than two prominent routing protocols in VANETs, without giving large compromise on the delivery delay. The adaptivity of ARP-QD is also analyzed.

The remainder of the paper is organized as follows. Section 2 briefly reviews related routing mechanisms proposed in VANETs and details the motivation of this paper. In Section 3, we design two metrics combining hop count and link duration for forwarding optimization. The adaptive neighbor discovery algorithm is also presented as well as the recovery strategy to improve the robustness of ARP-QD. Numerical simulations and the results are analyzed in Section 4. We conclude the paper and list some possible future works in Section 5.

2. Related Works

Generally, path efficiency and path stability are two important criterions in designing routing protocol for VANETs. To achieve high efficiency, the shortest (generally fastest) path with minimum hop count is usually selected as the best path. To pursue high stability, the path with the longest duration is considered as the best candidate. However, most of existing researches focus on either efficiency or stability. We review related works in both directions as follows.

Path Efficiency. One objective of a routing protocol in VANETs is to find an efficient (or a fast) path with the shortest number of hops for data delivery [5, 7–10]. Greedy Perimeter Stateless Routing (GPSR) algorithm uses the positions of routers and a packet’s destination to make packet forwarding decisions [7]. It chooses the nearest node to the destination as the next hop within communication range, which will increase the link loss because of high mobility and radio obstacles. Like GPSR, Geographic Source Routing (GSR) [8] is also a position based routing protocol. The weakness of GSR is not flexible to the sparse network, since GSR works on the foundation of end-to-end connectivity. Another similar method of GPSR is Greedy Perimeter Coordinator Routing (GPCR) [9], which assigns the routing decision to the nodes located at the street intersections and uses the greedy forwarding strategy to route the packet path between the street intersections. However, GPCR does not take the link connectivity into consideration to select the best path. An improved Greedy Traffic Aware Routing Protocol (GyTAR) has been presented in [5], which is based on the geographical intersection information to find robust and optimal routes within urban environments. In [10], a two-stage routing algorithm has been presented to find out the practically fastest route to a destination at a given departure time in terms of taxi drivers’ intelligence learned from a large number of historical taxi trajectories.

In short, most of the above researches regard the shortest path, but fail to concern the diverse QoS requirements of different applications. Some applications require more stable path for high delivery ratio, while the link connectivity between the current and farthest neighbor node is always most vulnerable, which may cause shorter link duration than other links. Hence, the above protocols are not suitable for applications which require high delivery ratio.

Path Stability. One of the simple but efficient methods to improve the path stability is to find the next hop with the longest link duration (or the most stable connectivity) [11–15]. A Receive On Most Stable Group-Path (ROMSGP) scheme [11] has been designed to choose the most stable path with the longest link expiration time. However, ROMSGP only broadcasts specific and well-defined packets, which will result in the loss of other packets. The goal of [12, 13] is to find the routing path with the least probability of network disconnection and avoid carry-and-forward delay. However, the links with good connectivity usually have short distance, which makes the selected paths include more hops and therefore brings longer delivery delay. A stable VANETs routing protocol [14] has been proposed to provide fast and reliable message delivery based on the real-time road vehicular density. However, the real-time update of density information incurs a large number of communication overheads, which results in its performance deterioration with the augment of network scale. An intersection-based geographical routing protocol has been proposed in [15], which aims to find the path with high connectivity probability and other QoS constraints.

In a word, all aforementioned researches mainly focus on the link connectivity and make less use of the geographical distance information among vehicles, such that the selected paths may have unnecessary loops, which causes longer delivery delay. Thus, the above protocols are not suitable for the applications which require low delivery delay.

Some researches, like [16, 17], take the link state and hop count into account. In [16], the authors have presented an Optimized Link State Routing (OLSR) algorithm to provide optimal routes. However, the link state is only used to obtain the neighbors’ information and OLSR provides the path with minimum hop count as the best path. Moreover, OLSR is a topology-based routing algorithm, which consumes a large amount of topology control messages. To improve GPSR,
[17] uses the vehicle speed and position to find relatively stable links, which is based on the forecast of the speed fluctuations. However, the above works failed to adaptively trade off the path efficiency and path stability for diverse QoS requirements in different scenarios and could not achieve the purpose of friendly communications.

To the best of our knowledge, there is no prior work that has thoroughly researched the adaptive routing protocol which can balance the path efficiency and stability based on diverse QoS requirements of different applications. In this paper, based on the information of intersection location, vehicle speed, and position, we take the hop count and link duration into consideration and propose a novel optimal forwarding algorithm to trade off the path efficiency and stability with friendly adaptation to different QoS requirements of applications. Furthermore, we present an adaptive neighbor discovery algorithm, which exploits different ways to acquire the neighbors’ information according to the local vehicular density. Based on the above two main algorithms, we build the adaptive routing protocol based on QoS and vehicular density (ARP-QD), which has higher delivery ratio and reasonable delivery delay.

3. The Proposed Adaptive Routing Protocol (ARP-QD)

In this section, we first introduce the system model used for urban VANETs. Then we present the optimal forwarding algorithm which adaptively balances the path efficiency and stability based on QoS requirements, as well as the adaptive neighbor discovery algorithm based on the real-time vehicular density. To improve the robustness of ARP-QD, the recovery strategy with carry-and-forward is adopted when the routing path is disrupted. Finally, an example is given to illustrate how the proposed ARP-QD works.

3.1. System Model. As shown in Figure 1, we consider a VANET road environment with intersections and segments within two intersections, which is a typical scenario in urban areas. The circle with the intersection ID inside denotes the intersection. \( \vec{v} \) and \( \vec{p} \) indicate the moving directions of the vehicle and the packet, respectively. The yellow arrow means the moving direction of vehicles on that road segment. The purple arrow with a right angle denotes the candidate path of the packet from the source node \( S \) to the destination \( D \). Vehicles move through the segments in the same or opposite direction, while, when moving into the intersection, they will find their neighbors moving in various directions.

Since the RSUs are costly, the paper focuses on the routing protocol for vehicle-to-vehicle (V2V) communications without RSUs. We assume that all vehicles are equipped with onboard navigation system and wireless communication capability as described in [18]. Each vehicle has a digital street map of the area using the onboard navigation system to determine the positions of its neighboring intersections. Meanwhile, it can acquire a landscape of the road environment, including the vehicular velocity and density on each road. The above information can be obtained through the commercialized applications [19]. Furthermore, through the periodic information exchange, each vehicle knows its neighbors’ information including the positions and velocities, which is maintained in its neighbor table. For easy illustration, we assume that all vehicles have the same transmission range. In addition, the location service can make the source node have the knowledge of destination position in real time. The above assumptions are the same as the previous works [4, 20, 21].

3.2. Optimal Forwarding Algorithm. As mentioned above, the real road environment contains two parts: intersections and segments within two intersections. Many vehicles, which are regarded as mobile nodes, move along the road as shown in Figure 1. We aim to find the best path hop by hop from the source node \( S \) which creates the packets to the destination \( D \). \( D \) can be the nearest Internet gateway or data collection center. Thus, we assume the destinations are always located in the intersections. The proposed ARP-QD is a geographic routing protocol including optimal forwarding decision, adaptive neighbor discovery, and robust route recovery. It selects the whole path hop by hop from \( S \) to \( D \), and each sender decides its next hop locally. It is easy to observe that a node traveling in the segment or intersection should use different tactics to calculate the metric to choose the next hop. For a node in the segment, it only chooses its next hop in the parallel directions, while for a node in the intersection it should first choose the next segment and then decide the next hop within the selected segment. Therefore, we define a new metric, that is, product of connectivity and distance (CDP), in two cases, respectively.

3.2.1. Metric Design in the Segment Case. Two seemingly contradictory, yet related, objectives of routing performance exist: improving the path efficiency with less hop count and improving the path stability with longer link duration. In general, the longer the link distance is, the smaller the hop count is. In contrast, the shorter the link distance is, the more stable the link is. We aim to design a novel metric for selecting the best next hop on the road segment, which can balance the requirements of the path efficiency and stability. We first
consider the one lane case and later show that the case of multiple lanes has the same result. In the one-lane case, we just consider that all vehicles drive in the same direction, and the result in the opposite direction can be easily induced in the same way. To formally design the metric, that is, CDP, the notations used in the following analysis are described in Notations.

We regard a neighbor node $n$ with $PL_n < PL_s$ as a candidate neighbor. Note that the path length means the distance along the selected roads.

First, we discuss the case of one lane. As depicted in Figure 2(a), we can obtain

$$R_n + L_n = R. \quad (1)$$

Since $PL_n < PL_s$,

$$R_n = PL_s - PL_n > 0. \quad (2)$$

It can be observed that the neighbor node $n$, which is closest to the destination, has the largest $R_n$.

We define $T_n$ in (3) to denote the link connection duration time between candidate neighbor node $n$ and the sender $s$:

$$T_n = \begin{cases} \frac{L_n}{v_n - v_s}, & v_n > v_s, \\ K, & v_n = v_s, \\ \frac{(2R - L_n)}{(v_s - v_n)}, & v_n < v_s, \end{cases} \quad (3)$$

where $K$ is a default constant set by the VANETs system. In order to improve the path efficiency and stability, we prefer to choose the neighbor node with the largest product of $T_n$ and $R_n$ as the next hop. Hence, the basic CDP of a neighbor node $n$ is defined as

$$CDP_n^b = R_n T_n = \begin{cases} \frac{R_n L_n}{v_n - v_s}, & v_n > v_s, \\ R_n K, & v_n = v_s, \\ \frac{R_n (2R - L_n)}{(v_s - v_n)}, & v_n < v_s. \end{cases} \quad (4)$$

As we can see, the CDP value depends on the relative speed and distance between the sender $s$ and candidate neighbor node $n$. Indeed, for a given lane with some nodes, the CDP function combines the factors of the distance from $n$ to $s$ and the link connection duration. Since larger $R_n$ means less hop count and larger $T_n$ means longer link duration, larger CDP is preferred. The node with the largest CDP among the candidates is selected to be the next hop. Figure 2(a) shows an example of vehicles driving on one lane. In this scenario, once the sender $s$ obtains the information of neighbors’ positions and velocities, it computes the CDP value of each neighboring vehicle. Considering its path length to the destination and the link duration with $s$, neighboring vehicle 1 (i.e., node 1) is assumed to get the maximum value of CDP. It is then chosen as the next hop. Note that if there are multiple neighbor nodes with the same largest CDP, $s$ will randomly pick up one as the next hop.

Then, we will discuss the case of multiple lanes as depicted in Figure 2(b). The relation is changed as shown in the following:

$$Q_s + L_n = \sqrt{R^2 - (kl)^2}. \quad (5)$$

Here it is assumed that a sender $s$ drives in lane 2 and the candidate neighbor node $n$ drives in lane $k + 2$, where $k$ indicates the number of interval lanes. Although transmission range $R$ is more than 100 m, (1) is usually less than 3 m. We can get $Q_n = R_s$ and $\sqrt{R^2 - (kl)^2} \approx R$. Consequently, (5) can be simplified to

$$R_n + L_n = R. \quad (6)$$

No matter where the vehicles drive in one lane or multiple lanes, their basic CDP can be calculated by (4).

Next, we modify the definition of CDP to satisfy diverse QoS requirements of different applications. In this paper, two prominent QoS requirements, that is, delivery delay and delivery ratio, are considered. For real-time applications such as video on demand, which require high priority on delivery delay, they need to find the efficient path with minimum hop count, while, for other applications such as file transmissions, which require the reliable transmission with high delivery ratio, they need to find the stable path with longest link duration. We use adaptive factors $\alpha$ and $\beta$ to represent diverse QoS requirements of different applications, where $\alpha$ implies the priority weight of hop count, while $\beta$ means the importance of link duration under the condition of $\alpha + \beta = 1$.

To friendly adapt to diverse QoS requirements of different applications, $\alpha$ and $\beta$ will be set according to the application requirements on delivery delay and delivery ratio. It is easily obtained that the larger $\alpha$ makes higher priority on delivery delay, which requires finding a path with shorter hop count. Therefore, the advanced CDP is defined as

$$CDP_n^a = R_n T_n^\alpha = \begin{cases} \frac{R_n^\alpha L_n}{(v_n - v_s)}, & v_n > v_s, \\ R_n^\alpha K, & v_n = v_s, \\ \frac{R_n^\alpha (2R - L_n)}{(v_s - v_n)}, & v_n < v_s. \end{cases} \quad (7)$$

From (1) and (7), we can obtain an optimal value of $CDP_n^a$ among different neighbors.

ARP-QD will select the one with the maximum $CDP_n^a$ among all candidate neighbor nodes as the best next hop. In conclusion, using the metric $CDP_n^a$ defined in (7), ARP-QD can friendly adapt to diverse QoS requirements when packets are delivered on the segment areas.

3.2.2. Metric Design in the Intersection Case. This part discusses how to design a new metric expression for the best next hop selection in the intersection by taking hop count and link duration into consideration. There are two stages for the sender in the intersection to choose the best next hop. First, the sender needs to choose which segment the packet will be
delivered. Then, based on the selected segment, the sender computes the best next hop located in that segment.

**Segment Selection.** Obviously, the segment selection for a sender in the intersection is to find the best next intersection. Candidate intersections are defined as the adjacent intersections whose path lengths are shorter than the current intersection. The mobile vehicles moving along the roads are formalized to form a mobile ad hoc vehicular network. To find an efficient routing path, we prefer to choose the one with long link duration, we tend to choose the neighbors in the vehicle carrying the packet until it connects to another vehicle, but the vehicle's moving speed is significantly slower than that of wireless communications. Thus, we aim to find an efficient routing path, we prefer to choose the slower than that of wireless communications. Thus, we aim to find an efficient routing path, we prefer to choose the slowest among all candidate intersections as the best next intersection.

**Next Hop Selection.** Once the next segment is selected, the direction of packet delivery is determined. In the following we give the process to select the next hop among all candidate intersections, which can be classified into two cases.

1. $\theta = 0$: in this case, the sender's moving direction is the same as the next hop's. Hence, we can use the same method to select the best next hop as that used in the segment case.
2. $\theta \neq 0$: in this case, $R_n + L_n \neq R$. We need to obtain new CDP equations. As shown in Figure 3, we assume that both the sender $s$ and candidate neighbor node $n$ are moving in constant speed, which are noted as $v_s$ and $v_n$, respectively. Using the cosine law, we can obtain the equation as follows:

$$R^2 = (R_n + v_n T_n)^2 + (v_s T_n)^2 - 2 v_n T_n (R_n + v_n T_n) \cos \theta.$$  

(10)

From (10), we can compute $T_n$ as follows:

$$T_n = \frac{R_n (v_s \cos \theta - v_n)}{v_n^2 + v_s^2 - 2 v_s v_n \cos \theta} + \frac{\sqrt{R_n^2 (v_s^2 + v_n^2 - 2 v_s v_n \cos \theta) - R_n^2 v_n^2 \sin^2 \theta}}{v_n^2 + v_s^2 - 2 v_s v_n \cos \theta},$$  

(11)

where $PL_{sj}$ indicates the path length of packet delivery from the sender $s$ to the destination through the intersection $j$. $PL$ is the summed length of paths through all candidate intersections, formally shown as $PL = \sum PL_{sj}$, where $j$ represents the ID of candidate intersections. $PL_{sj}$ is divided by PL for normalization. For the sender in a given intersection, PL is fixed and the path with smaller $PL_{sj}$ is preferred. In order to satisfy diverse QoS requirements of different applications, we also use the adaptive factors $\alpha$ and $\beta$ to represent the weight of hop count and link duration, respectively, in (9). ARP-QD will select the one with the minimum SSW among all candidate intersections as the best next intersection.
where \( R_n \) is the Euclidean distance from the sender \( s \) to its candidate neighbor \( n \) and \( \theta \) is the angle between candidate segment and movement direction of the current sender. Hence, the basic CDP is defined as

\[
\text{CDP}_n^b = R_n \frac{R_n(v_s \cos \theta - v_n)}{v_n^2 + v_s^2 - 2v_nv_s \cos \theta} + \sqrt{R^2 \left( \frac{v_n^2 + v_s^2 - 2v_nv_s \cos \theta}{v_n^2 + v_s^2 - 2v_nv_s \cos \theta} \right)^\beta}.
\]

(12)

Accordingly, the advanced CDP is obtained as

\[
\text{CDP}_n^a = R_n \frac{R_n(v_s \cos \theta - v_n)}{v_n^2 + v_s^2 - 2v_nv_s \cos \theta} + \sqrt{R^2 \left( \frac{v_n^2 + v_s^2 - 2v_nv_s \cos \theta}{v_n^2 + v_s^2 - 2v_nv_s \cos \theta} \right)^\beta}.
\]

(13)

The sender \( s \) chooses its candidate neighbor with the maximum \( \text{CDP}_n^a \) in (13) as the best next hop, which is located in the selected segment.

3.2.3. Optimal Forwarding Algorithm. In this part, we present a novel optimal forwarding algorithm, as described in Algorithm 1, to choose the best next hop for multihop packet delivery. The best next hop is selected from the sender's neighbor list, which is obtained by neighbor discovery algorithm (described in Section 3.3). Note that neighbor list contains the information of neighbors’ IDs and \( \text{CDP}_n^a \) values, while neighbor table is composed of neighbors' IDs, velocities, and positions. Each \( \text{CDP}_n^a \) value in the neighbor list is computed by (7) or (13) using the information in the neighbor table.

As mentioned above, there are two cases to analyze the next hop selection. On the one hand, when the sender \( s \) is moving along a road segment, it will choose the candidate neighbor with the maximum \( \text{CDP}_n^a \) value, from its neighbor list, as the best next hop. On the other hand, when \( s \) approaches an intersection, it needs to firstly find the best next intersection with the minimum SSW and then choose the best next hop located in the selected segment.

To find the best next intersection (or segment), \( s \) needs to get the information of which intersection is connected with the current intersection. Hence, \( s \) broadcasts a beacon packet, which contains a connectivity probe request (CP_REQ) and its own information as shown in Figure 4. CP_REQ includes the current intersection ID, source and destination of the data, request time, and expired time. It is used to probe the connectivity of each candidate intersection, which is indicated by \( U_j \). If a candidate intersection \( j \) is connected to the current intersection by mobile nodes (i.e., vehicles) moving between the current intersection and candidate intersection \( j \), the sender \( s \) will receive a responding packet from its neighbor node before the expired time; then \( U_j = 1 \); otherwise \( U_j = 0 \). The responding packet contains a connectivity probe reply (CP_REP) and neighbor's information as shown in Figure 4. CP_REP includes the candidate intersection ID, source and destination of the data, reply time, and expired time. Neighbor's information includes its ID, velocity, and position used for calculation of SSW and \( \text{CDP}_n^a \). The beacon will be dropped if the expired time is over. Based on the received responding packets, \( s \) calculates values of SSW according to (9) for all candidate intersections and then picks out the candidate intersection with the minimum SSW as the next intersection. Thus, the next delivery segment is selected accordingly. Finally, to find the best next hop, \( s \) chooses the candidate neighbor with the maximum \( \text{CDP}_n^a \) value as the best next hop from its neighbor list.
Input: The information of sender $s$ and destination $D$
Output: The next hop of the delivered packet

1. if $s$ approaches the intersection then
2. Broadcast a beacon packet with CP_REQ to each candidate intersection and active the Time 1 (expired time).
3. repeat
4. Receive responding packets with CP_REP and neighbors’ information
5. until Timer 1 expires
6. if $s$ receives a responding packet with CP_REP and neighbor’s information from intersection $j$ then
7. $U_j = 1$.
8. else
9. $U_j = 0$.
10. Compute SSW of each candidate intersection.
11. Select the next intersection with the minimum SSW.
12. Select the next hop with the maximum $CDP_n$ based on (7) or (13).
13. else
14. Choose the next hop with the maximum $CDP_n$ according to (7).

Algorithm 1: The optimal forwarding algorithm.

| CP_REQ | Node information |
|--------|------------------|
| src dst| Intersection ID | REQ time | Expired time | ID | Velocity | Position |

(a) Beacon packet

| CP_REP | Neighbor information |
|--------|-----------------------|
| src dst| Intersection ID | REP time | Expired time | ID | Velocity | Position |

(b) Responding packet

Figure 4: Packet format.

3.3. Adaptive Neighbor Discovery Algorithm. The neighbor list of each node is updated at fixed intervals to keep neighbors’ information in real time, which is the precondition of the optimal forwarding algorithm. Vehicular density has a tremendous impact on the network performance, and high density incurs serious congestion during the update process of neighbors’ information. In other words, heavy periodic beacons for neighbor discovery will decrease the average throughput of network, which causes negative influence on the end-to-end data delivery. In this section, we aim to design an adaptive neighbor discovery algorithm based on the vehicular density to obtain the neighbor list.

The proposed neighbor discovery algorithm can adaptively reduce the communication overhead according to the local vehicular density, which is defined as the number of nodes in transmission range of node $i$, denoted as $d_i$. We set a density threshold $d_{th}$ to evaluate the local vehicular density $d_i$. The basic principle of the adaptive neighbor discovery algorithm is to choose a centralized way to discover neighbors and update neighbor list when $d_i$ is lower than $d_{th}$, while using a distributed fashion on the opposite. The detailed process of neighbor discovery is illustrated in Algorithm 2.

In the centralized way, node $i$ first broadcasts a start beacon to request all neighbors’ information. Next each neighbor answers to the beacon with the information of its own position and velocity. Based on the neighbors’ information of positions and velocities, node $i$ can compute $CDP_n$ value of each neighbor $n$ by (7) or (13). Thus the neighbor table and neighbor list of node $i$ are updated. The optimal forwarding algorithm will select the best next hop from this neighbor list, as mentioned in Section 3.2. Since the destination of all neighbors’ answers is node $i$, they adopt distributed coordination function in IEEE 802.11 to avoid the transmission collision. Request to send (RTS) and clear to send (CTS) control frames are used to reserve channel access and to minimize the amount of wasted bandwidth when collision occurs [22]. Since $d_i$ is lower than $d_{th}$, such a centralized way for neighbor discovery will not result in heavy communication overheads.

In the distributed fashion, we propose a receiver-based approach for neighbor discovery. Node $i$ broadcasts a start beacon that informs its neighbors about its position and velocity. Each receiver computes its own $CDP_n$ value by (7) or (13). In order to reduce the communication overhead, it can only answer to the beacon after a waiting time based on its $CDP_n$ value by a uniform rule as defined in the following:

$$T_n = \frac{T^*}{CDP^*_n},$$

where $T^*$, set by the VANETs system, is a time parameter to control the relation between $CDP_n$ value and waiting time of receiver $n$. $n$ means the node which can receive the start beacon from node $i$, which is node $i$’s neighbor. The waiting time of neighbor $n$ is inverse correlation with the value of $CDP_n$ calculated by (7) or (13). It is easily observed that the neighbor with the maximum $CDP_n$ has the smallest waiting time; therefore it will answer to node $i$ at the first time. Once node $i$ hears this answer, it will broadcast a stop message to all neighbors to terminate the current neighbor discovery. Thus the neighbor list of node $i$ will have only one node. If node $i$ has not received any answers before the expired time, its neighbor list will be empty at current time. The optimal forwarding algorithm will select the best next hop from this
neighbor list, as mentioned in Section 3.2. Since $d_1$ is higher than $d_{th}$, such a distributed way for neighbor discovery will significantly reduce the communication overheads.

This adaptive neighbor discovery algorithm requires each node to previously know the local vehicular density, which is easily to be obtained by the current commercial applications [19], as mentioned before. Intuitively, this adaptive approach will increase the average data delivery ratio by reducing the communication overheads during the neighbor discovery in dense networks, while decreasing the delay by reducing the waiting time in sparse networks.

Remark. The adaptive neighbor discovery algorithm still works when the update of neighbor list is triggered by the forwarding event.

### 3.4. Routing Path Recovery Strategy

In the dynamic wireless environment, it is inevitable that the routing path fails or breaks. Once a selected link breaks, a local recovery procedure takes place. To improve the robustness of ARP-QD, the adopted recovery strategy is based on the idea of carry-and-forward [23]. The sender which detects the broken link will explore the one-hop neighbors to find a backup link. If the sender has no one-hop neighbor, it will carry the packet until another node moves its transmission range to transfer the packet. Furthermore, such a carry-and-forward strategy guarantees loop-free routing and avoids endlessly forwarding loop by marking the previous hops.

### 3.5. A Walk-Through Example

The whole ARP-QD contains the two main novel algorithms proposed above: optimal forwarding algorithm and adaptive neighbor discovery algorithm. In order to improve the robustness of ARP-QD, the carry-and-forward strategy for routing path recovery is also complemented. We use the following example, depicted in Figure 5, to illustrate how ARP-QD works. According to the QoS requirement of certain application, the adaptation factors $\alpha$ and $\beta$ are set for computation of $CDP^*_n$ and $SSW_j$. With the help of onboard GPS, navigation system, and digital map, the source node $S$ can obtain the position of destination. The dotted parallel lines denote the transmission range of the source node $S$. We assume all nodes have the same transmission range.

1. The source node is in the segment area, and the local vehicular density around $S$ is smaller than the certain density threshold $d_{th}$. Thus, $S$ exploits the centralized way to discover neighbors and compute the $CDP^*_j$ value of each candidate neighbor. After collecting the $CDP^*_n$ information, $S$ chooses the one with maximum $CDP^*_n$ as the best next hop, which is $s_1$ in this case. For the same way, $s_2$ is selected to be the best next hop of $s_1$.

2. The sender $s_2$ approaches the intersection $I_5$. First, $s_2$ needs to choose the best next intersection with the minimum value of $SSW$. $s_2$ broadcasts a beacon with CP_REQ to request the information of connectivity from the current intersection to all candidate intersections. It aims to find the intersection with the shortest path length $PL_{s_j}$, the least direction change angle $\theta$, and the connectivity with the current intersection. In this example, when $s_2$ traveling to $I_2$ arrives at $I_5$, it has two candidate intersections, that is, $I_4$ and $I_6$. Note that $\vec{v}$ and $\vec{p}$ are the moving directions of sender $s_3$ and the delivered packet, respectively. $s_2$ computes $SSW_4$ and $SSW_6$ according to (9). It is easy to get that $U_4 = 1$ and $U_6 = 0$ because there are no vehicles between intersections $I_2$ and $I_6$. Hence, $s_2$ chooses $I_4$ with the minimum $SSW$ as its next intersection. Next, $s_2$ selects its best next hop. Assume that the local vehicular density of $s_2$ is smaller than the density threshold $d_{th}$, $s_2$ uses the centralized way to compute the $CDP^*_n$ values of candidate neighbors located in the selected segment. Assuming that $s_3$ has the maximum $CDP^*_n$, it is selected to be the next hop.

![Figure 5: A walk-through example.](image-url)
(3) The local vehicular density of $s_3$ is higher than the density threshold $d_{th}$; therefore $s_3$ adapts to the distributed fashion to discover neighbors. $s_3$ broadcasts a start beacon to inform its neighbors about its position and velocity. Each neighbor which received the beacon will compute its unique waiting time for sending answer to $s_3$ based on (14). In this case, we assume $s_3$ is the one which first replies and then $s_3$ is selected to be the next hop of $s_3$.

(4) If the link fails when $s_3$ is sending packets, the recovery mode of routing path is active. $s_3$ will notice its neighbors and find a backup link from its current neighbors. If $s_3$ has no neighbor to deliver packets, it will carry them until some appropriate nodes move into its transmission range. The following process of packet forwarding is the same as the above illustrated until the packet is delivered to its destination $D$.

4. Performance Evaluation

To evaluate the performance of the proposed ARP-QD, we simulate the protocol on a variety of data transmission rates and network densities. To compare the performance of ARP-QD with the previous works in VANETs routing, we also simulate basic GPSR [7], which aims to find a path with minimum hop count, and ROMSGP [11] which can guarantee a high level of stable communication to some extent. Note that most of geographic VANET routing protocols are based on GPSR with little differences in essence. ROMSGP is a classical stable VANET routing protocol for comparison.

4.1. Simulation Environment. We simulate ARP-QD in the vehicle traffic model using the standard NS2 simulator [24], which offers full simulation of the IEEE 802.11 physical and MAC layers. In our simulation, network size is set to be 50, 100, 150, 200, and 250 nodes with 802.11 WaveLAN radios. The assumptions are that all vehicles have the same transmission range of 250 m and all packets have the same size of 512 bytes. We simulate 20 constant bit rate (CBR) traffic flows to destinations, and sources and destinations are picked up randomly. The transmission rate of each CBR flow is set to be 0.5, 1.0, 1.5, 2.0, 2.5, and 3.0 packets per second (p/s). Each simulation lasts for 1000 seconds. Table 1 summarizes the key parameters in the simulation.

4.2. Mobility Model. The mobility model has a great impact on the studied protocol behavior in the simulation and the corresponding results [25]. For evaluating protocol performance accurately in such a complex and dynamic vehicular environment, we use VanetMobiSim [25] to initially place nodes uniformly at random and generate the random movement of the nodes within a $10 \times 10$ km$^2$ rectangular region with a maximum speed of 30 m/s. Figure 5 shows the simulation scenario, including 9 intersections and 12 road segments. We assume that a road segment composes two lanes without traffic signals. When a node approaches the intersection, it will randomly choose a road segment to turn its direction without pause.

| Parameter               | Value       |
|-------------------------|-------------|
| Number of lanes         | 2           |
| Number of nodes         | 50, 100, 150, 200, 250 |
| Velocity                | 10–30 m/s   |
| Simulation duration     | 1000 s      |
| Simulation area         | $10 \times 10$ km$^2$ |
| Channel capacity        | 2 Mbps      |
| Wireless communication range | 250 m     |
| Mac protocol            | 802.11 DCF  |
| Beacon interval         | 0.5 s       |
| Data packet size        | 512 bytes   |
| CBR rate                | 0.5, 1.0, 1.5, 2.0, 2.5, 3.0 p/s |
| Routing protocol        | ARP-QD, GPSR, ROMSGP |

4.3. Simulation Results. We focus mainly on the performance of delivery ratio and delivery delay in the simulation. (1) Delivery ratio is measured as the ratio of the number of successfully delivered data packets to the total number of transmitted data packets. The packet will be dropped when it fails to be delivered, without retransmission rule. (2) Delivery delay is measured as the average time elapsed from sending the packet by the source node to receiving it by the destination. Without loss of generality, we first fix the adaptive weight factors ($\alpha, \beta$) at (0.5, 0.5) to evaluate the impact of transmission rate and network density. Next, we fix the transmission rate at 1.5 p/s and the number of nodes at 150 to observe the impact of adaptive weight factors $\alpha$ and $\beta$.

4.3.1. Delivery Ratio. The number of nodes is set to 150 when we study the impact of transmission rate, while the transmission rate is fixed at 1.5 p/s when we focus on the impact of network density. Figures 6 and 7 show the delivery ratio with respect to varied transmission rate and the number of nodes, respectively. The two figures show that the proposed ARP-QD has higher delivery ratio compared with that of GPSR and ROMSGP in all cases. The reason is that ARP-QD considers the whole path based on the SSW metric, while GPSR works on the vehicle-by-vehicle forwarding and ROMSGP makes the vehicles with the same moving direction into groups, which only considers the local segment, rather than the whole path. Another reason is that the adaptive neighbor discovery algorithm reduces the communication overheads. Furthermore, from Figure 6 we can see that the delivery ratio of ARP-QD does not change much as the transmission rate is increased, while that of GPSR and ROMSGP deteriorates. This comes from the fact that the routing paths found by ARP-QD are more tolerant to the high network load due to the adaptive neighbor discovery algorithm. The main reason is that the adaptive neighbor discovery algorithm largely reduces the beacon cost to require neighbors’ information, which reserves more bandwidth for data delivery. Thus, the network load is still tolerable when the transmission rate rises up to 3.0 p/s. From Figure 7 we can observe that the delivery ratio increases with the rise of the number of nodes but decreases when the number of nodes goes up to 200.
The reason is that before the number of nodes reaches 150 or other values less than 200, the increased network density becomes higher than the density threshold and the enhanced connectivity and the reduced communication overheads during the neighbor discovery procedure improve the delivery ratio. With the continuous increase of node density, the overheads increase for updating all nodes’ neighbor list. Thus the performance of delivery ratio diminishes.

4.3.2. Delivery Delay. As shown in Figure 8, the delay of ARP-QD is the same as that of GPSR but is lower than that of ROMSGP at lower transmission rate. That is because the collisions are rare to happen when the transmission rate is lower and ROMSGP tends to choose the path with more hops for stability. However, when the transmission rate increases, the performance of ARP-QD deteriorates in terms of delivery delay. That is because high transmission rate makes the sender fail to find a backup neighbor quickly; when the link breaks, the time of carry-and-forward procedure prolongs the delivery delay. In brief, ARP-QD is not suitable for the applications with high QoS requirement on delivery delay when the network load is higher. Figure 9 shows that the delay of all protocols decreases along with the increase of the number of nodes. The reason is that packets can be delivered quickly with less caching time when the network density is high. Moreover, ARP-QD only has little difference on the delivery delay compared with the other two protocols when the number of nodes increases, which means ARP-QD does not give high compromise on the delivery delay.

4.3.3. The Impact of Adaptive Factor $\alpha$. In order to evaluate the impact of weight factors $\alpha$ and $\beta$ for different QoS
requirements, we obtain the simulation results of delivery ratio and delivery delay when $\alpha$ increases from 0.1 to 0.9 with the interval of 0.2. In Figures 10 and 11 where the transmission rate is set to 1.5 p/s, the four curves represent different number of nodes. In Figures 12 and 13 where the number of nodes is set to 150, the six curves represent different transmission rate.

From Figures 10, 11, 12, and 13, we can draw the conclusion that the delivery ratio declines, while the delivery delay goes down along with the increase of $\alpha$. That is because the link efficiency has larger weight and the link stability has smaller weight accordingly when the factor $\alpha$ is increased. The link has more probability to break down along with the rise of $\alpha$; thus the delivery ratio turns worse. At the same time, the number of hops for each path is decreased with the higher requirements on link efficiency; thus the delivery delay is improved. In addition, from Figure 12, we can observe that the delivery ratio of ARP-QD does not vary much when the transmission rate changes, which is the same as that observed in Figure 6. Figure 13 shows that the delivery delay of ARP-QD varies much when the network density changes as analyzed in Figure 8. These results show that the weight factor $\alpha$ can adaptively satisfy the diverse QoS requirements of different applications.

5. Conclusion

In this paper, the proposed adaptive routing protocol based on QoS and vehicular density (ARP-QD) is capable of
finding a fast and reliable path for end-to-end data delivery within urban VANETs environments according to diverse QoS requirements of different applications. To reduce the communication overheads furthermore, ARP-QD adopts the adaptive fashion to obtain the neighbors’ information based on local vehicular density and recovers quickly when the routes are disrupted. Numerical simulations showed that ARP-QD has a higher delivery ratio than GPSR and ROMSGP, without giving large compromise on the delivery delay. In the future, we shall take the real data trace into consideration to validate ARP-QD protocol and combine the link correlations to estimate link quality.

**Notations**

- \( n \): The neighbor node
- \( s \): The current sender
- \( S \): The source node
- \( D \): The destination
- \( R \): The vehicular transmission range
- \( l \): The distance of two adjacent lanes
- \( R_{n} \): The Euclidean distance between the neighbor node \( n \) and the sender \( s \)
- \( L_{n} \): The distance that the neighbor node \( n \) moves out the transmission range of the sender \( s \)
- \( Q_{n} \): The projection of \( R_{n} \) in the road direction
- \( PL_{n} \): The path length between the neighbor node \( n \) and the destination \( D \)
- \( PL_{s} \): The path length between the sender \( s \) and the destination \( D \)
- \( v_{n} \): Velocity of the neighbor node \( n \)
- \( v_{s} \): Velocity of the current sender \( s \)
- \( T_{n} \): The link duration time between the neighbor node \( n \) and the sender \( s \).

**Conflict of Interests**

The authors declare that there is no conflict of interests regarding the publication of this paper.

**Acknowledgment**

This research was supported by Major Program of National Natural Science Foundation of China (no. 6190114).

**References**

[1] H. Hartenstein and K. P. Laberteaux, “A tutorial survey on vehicular ad hoc networks,” IEEE Communications Magazine, vol. 46, no. 6, pp. 164–171, 2008.

[2] C. Xu, F. Zhao, J. Guan, H. Zhang, and G.-M. Muntean, “QoE-driven user-centric vod services in urban multi-homed P2P-based vehicular networks,” IEEE Transactions on Vehicular Technology, vol. 62, no. 5, pp. 2273–2289, 2013.

[3] C. Xu, T. Liu, J. Guan, H. Zhang, and G.-M. Muntean, “CMT-QA: quality-aware adaptive concurrent multipath data transfer in heterogeneous wireless networks,” IEEE Transactions on Mobile Computing, vol. 12, no. 11, pp. 2193–2205, 2013.

[4] F. Li and Y. Wang, “Routing in vehicular ad hoc networks: a survey,” IEEE Vehicular Technology Magazine, vol. 2, no. 2, pp. 12–22, 2007.

[5] M. Jerbi, S.-M. Senouci, T. Rasheed, and Y. Ghamri-Doudane, “Towards efficient geographic routing in urban vehicular networks,” IEEE Transactions on Vehicular Technology, vol. 58, no. 9, pp. 5048–5059, 2009.

[6] D. Lee, Y.-H. Kim, and H. Lee, “Route prediction based vehicular mobility management scheme for VANET,” International Journal of Distributed Sensor Networks, vol. 2014, Article ID 679789, 9 pages, 2014.

[7] B. Karp and H. T. Kung, “GPSR: greedy perimeter stateless routing for wireless networks,” in Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM ’00), pp. 243–254, ACM, August 2000.

[8] C. Lochert, H. Hartenstein, J. Tian, H. Fussler, D. Hermann, and M. Mauve, “A routing strategy for vehicular ad hoc networks in city environments,” in Proceedings of the IEEE Intelligent Vehicles Symposium, pp. 156–161, IEEE, Columbus, Ohio, USA, June 2003.

[9] C. Lochert, M. Mauve, H. Füßler, and H. Hartenstein, “Geographic routing in city scenarios,” ACM SIGMOBILE Mobile Computing and Communications Review, vol. 9, no. 1, pp. 69–72, 2005.

[10] J. Yuan, Y. Zheng, C. Zhang et al., “T-drive: driving directions based on taxi trajectories,” in Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS ’10), pp. 99–108, ACM, San Jose, Calif, USA, November 2010.

[11] T. Taleb, E. Sakhaee, A. Jamalipour, K. Hashimoto, N. Kato, and Y. Nemoto, “A stable routing protocol to support its services in vanet networks,” IEEE Transactions on Vehicular Technology, vol. 56, no. 6, pp. 3337–3347, 2007.

[12] C. Rezende, R. W. Pazzi, and A. Boukerche, “Enhancing path stability towards the provision of multimedia support in vehicular ad hoc networks,” in Proceedings of the IEEE International Conference on Communications (ICC ’10), pp. 1–5, IEEE, 2010.

[13] Q. Yang, A. Lim, and P. Agrawal, “Connectivity aware routing in vehicular networks,” in Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC ’08), pp. 2218–2223, IEEE, April 2008.

[14] H. Yu, S. Ahn, and J. Yoo, ”A stable routing protocol for vehicles in urban environments,” International Journal of Distributed Sensor Networks, vol. 2013, Article ID 759261, 9 pages, 2013.

[15] H. Saleet, R. Langar, K. Naik, R. Boutaba, A. Nayak, and N. Goel, “Intersection-based geographical routing protocol for VANETs: a proposal and analysis,” IEEE Transactions on Vehicular Technology, vol. 60, no. 9, pp. 4560–4574, 2011.

[16] T. Clausen, P. Jacquet, C. Adjih et al., Optimized Link State Routing Protocol (OLSR), RFC, 2003.

[17] H. Huang and S. Zhang, “A routing algorithm based on dynamic forecast of vehicle speed and position in vanet,” International Journal of Distributed Sensor Networks, vol. 2013, Article ID 390795, 9 pages, 2013.

[18] H. Saleet, O. Basir, R. Langar, and R. Boutaba, “Region-based location-service-management protocol for VANETs,” IEEE Transactions on Vehicular Technology, vol. 59, no. 2, pp. 917–931, 2010.

[19] Digital map from mapmechanics, http://www.allmapdata.com/.
[20] Z. Mo, H. Zhu, K. Makk, and N. Pissinou, “MURU: a multi-hop routing protocol for urban vehicular ad hoc networks,” in *Proceedings of the 3rd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous ’06)*, pp. 1–8, July 2006.

[21] R. N. Murty, G. Mainland, I. Rose et al., “Citysense: an urban-scale wireless sensor network and testbed,” in *Proceedings of the IEEE Conference on Technologies for Homeland Security*, pp. 583–588, IEEE, 2008.

[22] B. P. Crow, I. Widjaja, J. G. Kim, and P. T. Sakai, “IEEE 802.11 wireless local area networks,” *IEEE Communications Magazine*, vol. 35, no. 9, pp. 116–126, 1997.

[23] J. A. Davis, A. H. Fagg, and B. N. Levine, “Wearable computers as packet transport mechanisms in highly-partitioned Ad-Hoc networks,” in *Proceedings of the 5th International Symposium on Wearable Computers (ISWC ’01)*, pp. 141–148, IEEE, October 2001.

[24] The Network Simulator, “The vint project. The ucb/lbnijvint network simulator-ns (version 2),” http://mash.cs.berkeley.edu/ns.

[25] J. Härri, F. Filali, C. Bonnet, and M. Fiore, “Vanetmobisim: generating realistic mobility patterns for vanets,” in *Proceedings of the 3rd ACM International Workshop on Vehicular Ad Hoc Networks (VANET ’06)*, pp. 96–97, ACM, September 2006.
