BRT: Bus-based Routing Technique in Urban Vehicular Networks

Noureddine Chaib, Omar Sami Oubbati, Omar Sami Oubbati, Mohamed Lahcen Bensaad, Abderrahmane Lakas, Pascal Lorenz

BRT: Bus-based Routing Technique in Urban Vehicular Networks

Noureddine Chaib, *Senior Member, IEEE*, Omar Sami Oubbati, *Member, IEEE*, Mohamed Lahcen Bensaad, *Member, IEEE*, Abderrahmane Lakas, *Member, IEEE*, Pascal Lorenz, *Senior Member, IEEE*, Abbas Jamalipour, *Fellow, IEEE*,

***Abstract*—Routing data in Vehicular Ad hoc Networks is still a challenging topic. The unpredictable mobility of nodes renders routing of data packets over optimal paths not always possible. Therefore, there is a need to enhance the routing service. *Bus Rapid Transit* systems, consisting of buses characterized by a regular mobility pattern, can be a good candidate for building a backbone to tackle the problem of uncontrolled mobility of nodes and to select appropriate routing paths for data delivery. For this purpose, we propose a new routing scheme called Bus-based Routing Technique (BRT) which exploits the periodic and predictable movement of buses to learn the required time (the temporal distance) for each data transmission to Road-Side-Units (RSUs) through a dedicated bus-based backbone. Indeed, BRT comprises two phases: (i) Learning process which should be carried out, basically, one time to allow buses to build routing tables entries and expect the delay for routing data packets over buses, (ii) Data delivery process which exploits the pre-learned temporal distances to route data packets through the bus backbone towards an RSU (backbone mode). BRT uses other types of vehicles to boost the routing of data packets and also provides a maintenance procedure to deal with unexpected situations like a missing nexthop bus, which allows BRT to continue routing data packets. Simulation results show that BRT provides good performance results in terms of delivery ratio and end-to-end delay.**

***Index Terms*—VANETs; Routing; Backbone; Learning process; Bus.**

I. Introduction

In Vehicular Ad hoc Networks (VANETs), the vehicle to infrastructure communication has two main purposes: First, it allows RSUs (RoadSide Units) to warn vehicles about dangerous situations. Second, it is required to report to infrastructures information about road and traffic conditions, as well as other data originated from vehicles embedded sensors [1]. Unfortunately, reporting these data to infrastructures can only be achieved if RSUs are within vehicles range, using paid services provided from a telecommunication operator, or using unreliable multihop communications. The last option seems to be more interesting than others regarding budget and its feasibility. However, paths to RSUs over direct multihop communications, are not always available due to the limited number of deployed RSUs and to the dynamic and disconnected nature of the topology [2].

To increase the packet delivery ratio in vehicular environments with a limited number of RSUs and low-density of vehicles, the technique of store-carry and forward strategy has been proposed and used in an important number of routing schemes [3]–[10]. It buffers data packets until it finds paths over time [11]. The problem with most of the existing solutions is that they rely on chances to find paths to RSUs over vehicles with unpredictable mobility [12]. Thus, we cannot expect the end-to-end delay. In fact, most of VANET applications require delivering data to intended destinations within an acceptable bound of delays [13].

The best solution might comprise a dedicated backbone that provides data relaying services for VANET applications with the lowest budget requirements. A backbone can be built using existing nodes in VANETs that are characterized by a periodic mobility pattern, and the access to this backbone should be available everywhere to maximize the backbone service scale [2]. To this day, only one specific system is characterized by a known and a periodic mobility pattern, which is the *Bus Rapid Transit* system. It is now available in many cities over the world. The regularity of this system can be ensured by reserving dedicated lanes for buses, specific traffic policies such as giving buses higher priority at intersections. The regularity of the schedule of buses can be rendered more accurate using smart systems of self-driving buses [14].

The current work aims to present a new protocol, called BRT (Bus-based Routing Technique), to deliver data packets to an RSU. We focus on the application of data collection where vehicles collect data and send it to an RSU. Unlike the previous bus-based routing schemes, BRT takes full advantage of the schedules of buses. Our challenge is to select the best combinations of buses and other types of vehicles to optimize packet delivery ratios and end-to-end delays. This protocol requires two phases. The first phase is the learning process which should be carried out once initially to learn the near-optimal paths and build routing entries taking into account the existing permanent obstructions. The main purpose of the learning process is to measure the required time to deliver data packets from buses to RSUs all over time and to build routing entries that minimize this metric. The second phase is the data delivery process. It provides the routing of data packets by switching between the following modes:

N. Chaib, O.S. Oubbati, and M.L. Bensaad are with the Computer Science and mathematics Laboratory, University of Laghouat, BP 37G, Ghardaïa Road, Laghouat 03000, Algeria. E-mail: {n.chaib,s.oubbati,l.bensaad}@lagh-univ.dz.

A. Lakas is with College of Information Technology, United Arab Emirates University, United Arab Emirates. E-mail: alakas@uaeu.ac.ae.

P. Lorenz is with University of Haute Alsace, France. E-mail: pascal.lorenz@uha.fr.

A. Jamalipour is with School of Electrical and Information Engineering, University of Sydney, Camperdown NSW 2006, Australia. E-mail: a.jamalipour@ieee.org.

TABLE I: Features comparison of bus-based routing protocols.

Features	Bus-based routing							Our protocol
	CBS	Vela	UBTS	MI-VANET	BUS-VANET	BTSC	MIBR	
	Ref. [15]	Ref. [16]	Ref. [17]	Ref. [18]	Ref. [19]	Ref. [20]	Ref. [21]	
Routing involving buses	√	√	√	√	√	√	√	√
Bounded End-to-end Delay estimation	×	√	×	×	√	×	×	√
Store-carry and forward	√	√	√	√	√	√	√	√
Predefined Paths	×	×	×	×	×	×	×	√
Epidimic	√	×	√	×	×	×	×	×
Basic Technique	Buses Community	Probabilistic Graph	Neural network	Mobile Infrastructure Routing	Two Tiers architecture	Probability of path consistency	Buses Density Estimation	Temporal distance

- *The backbone mode:* it is performed only by buses which exploit the routing tables that are constructed during the learning process.
- *The FFG (Fast Forwarding to Gateway) mode:* the purpose of this mode is to accelerate the routing process. The keystone of our solution is to incorporate other types of vehicles in the data delivery path to extend the communication range of buses. Using FFG, a bus can detect and communicate (using a multihop communication and without a store-carry and forward) with a faraway another bus which acts as a transit gateway (or even an RSU if possible) that minimizes the temporal distance to an RSU. In fact, BRT is designed to select a transit gateway with the lowest temporal distance within *n* hops. In this way, it takes advantage of other types of vehicle to boost the data delivery to an RSU.

The rest of the paper is organized as follows: We start by presenting the related works in Section II. Then, in Section III, we provide an overview of our proposed protocol BRT. Next, we describe the details of its phases (learning process, data delivery process). In Section IV, we evaluate its performance through simulations. Finally, we conclude our paper and discuss future works in Section V.

II. Related Work

In the literature, there are a large number of existing VANET routing solutions. These solutions can be classified into two main categories according to the types of nodes constituting the routing paths, which are: (i) Vehicle-based routing and (ii) Bus-based routing.

A. Vehicle-based routing protocols

In this category, we summarize the most relevant solutions based on different techniques.

Among the approaches that are often used to address the routing issue in VANETs is the greedy forwarding strategy. It consists in bringing the data packet closer to the destination in each step using geographical position information of other nodes. Typical greedy position based routing schemes are GSR [22] and GPCR [23].

In [24], data packets are broadcasted, and a timer based technique is used to minimize packets collision and duplication. Even though this protocol delivers data packets within short delays, it incurs a high overhead caused by the broadcast process.

In [25], the network is divided into multiple moving zones based on vehicle movement information. This approach needs a load balancing by avoiding sending all data packets to a predefined set of nodes. Moreover, the management zone requires an extensive exchange of control packets.

UVAR (UAV-Assisted VANET Routing Protocol) [26] takes advantage of the existing unmanned aerial vehicles hovering over the area to route data packets. The major drawback of UVAR is the limited energy restriction of UAVs, which is crucial for the functioning of the protocol.

In [27], the route selection is based on statistics about traffic density information. In the case of poorly dense networks, data packets might never be delivered to the target destination since the vehicles' movements are not predictable. Moreover, the density statistics cannot guarantee accurate and correct paths.

B. Bus-based routing protocols

To better shape our routing protocol, a set of bus-based routing protocols is described in this subsection.

In [15], the CBS routing technique has been proposed to deliver data to RSUs, buses, and specific locations. The proposed scheme is inspired by social networks to build community graphs which are required to derive the backbone graph. It should be stressed that the backbone construction is a one-off operation which is done offline using the GPS reports of buses. To increase the packet delivery ratio and save the carrying time of buses, the authors proposed to send duplicate copies of the same message.

BTSC (Bus Trajectory-based Street-Centric routing algorithm) [20] uses the probability of bus appearance on streets to make routing decisions. The BTSC selects the best path with a higher density of buses and a lower probability of transmission direction deviating from the routing path. To decrease end-to-end delay, BTSC uses the ant colony optimization based strategy to find a reliable and steady multi-hop link.

In [18], the proposed scheme MI-VANET considers public transports as mobile infrastructures to provide networking services to other vehicles. The major drawback of this scheme is the use of dedicated centralized infrastructure for managing backbone services.

BUS-VANET [19] combines the existing infrastructures on the roads (*e.g.,* RSUs) with buses to forward data packets to the closest road infrastructure. Then, if the infrastructure has the required information about the destination vehicle, the packets are directly sent to the destination. As a drawback of this technique, the initialization of the discovery process is required each time the position of the target infrastructure is unknown.

A neural network based routing approach built on top of an Urban Bus Transportation System (UBTS) has been proposed in [17]. It uses a multi-graph of predicted journeys to the destinations and a specific control algorithm to improve the performance of the system.

In [16], a geocast routing mechanism named Vela has been proposed. It analyzes and mines spatialtemporal patterns about buses. This information helps to build an efficient probabilistic graph model and to make routing decisions with the best possible quality-of-service levels for data delivery requests.

In [21], road segments based routing scheme called MIBR(Mobile Infrastructure Based Routing) has been proposed. The basic idea of this scheme is to select a road with many buses because it is often a prosperous area. In MIBR, data packets will be routed between vehicles. However, buses are given higher priority to become the nexthops in some situations.

TABLE I provides a brief comparison between the previously discussed bus-based routing protocols.

III. Bus-based Routing Technique in Urban Vehicular Networks

Our proposed routing system BRT (Bus-based Routing Technique) comprises two phases:

1) An initial phase called learning process that should be carried out once by buses to build routing tables and to learn the temporal distances to reach a gateway (over the near-optimal path).
2) An exploitation phase called data delivery process in which two types of network participants exist, as depicted in Fig. 1, PBs (Public Buses) constituting the *Bus Rapid Transit* system and OTVs (Other Types of Vehicles). PBs are characterized by a regular and frequent schedule where a shift of the schedule is rare.

TABLE II summarizes the used notations in this paper.

Let us consider the illustrating scenario depicted in Fig 1. The vehicle v_1 wants to send a data packet to the gateway. To that effect, it should send this data packet to the bus b_1. The latter, in turn, forwards the packet to the nexthop b_2, which will carry it until it reaches the gateway. The data packet has been successfully delivered because the bus movements and their paths are predictable, and this knowledge has to be exploited.

TABLE II: Summary of notations.

Notation	Definition
t_i	Instant of time
$\phi_{i,j}(t)$	Reception time of a message sent at t by the bus i to the bus j
$\phi_i(t)$	It equals $\phi_{i,RSU}(t)$
$d_{i,j}(t)$	Temporal distance from the bus i to the bus j at t
$d_i(t)$	It equals $d_{i,RSU}(t)$
LC_i	Local Clock of the bus i
$Update\_time$	It records the expiration time of the FFG mode variables
$t_{reception}$	Message reception time
nh	Number of hops to the gateway according to the OTV message sender
rnh	Saved value of the number of hops to the gateway
L	Remaining time for two buses to stay neighbors
TDA	Temporal Distance Advertisement message
OTV	Other Types of Vehicles
PB	Public Bus
FFG	BRT mode (Fast Forwarding to Gateway)
TD	Temporal Distance
TGT	A bus (reachable via the FFG mode) that can serve as a Transit Gateway and has the lowest temporal distance within n hops
$Gateway$	The RSU that can receive a message first.
TG_ID	Gateway or Transit gateway identifier
TG_v	TG_ID advertized by the vehicle v
PDR	Packet Delivery Ratio
EED	End-to-End Delay

The mobility of OTVs is to a large extent unpredictable and uncontrollable from a store-carry and forward viewpoint [28]. Thus, we propose to avoid using the store-carry and forward strategy for OTVs, as much as possible (except for the isolated OTV scenario which will be described later in the paper). OTVs provide a facility of forwarding to the gateway (an *RSU*) or to TGT (Transit Gateway, which is a public bus within n hops that has the lowest temporal distance to the gateway), if they are accessible through a multihop protocol (without store-carry and forward). The selected path over OTVs should be the near-fastest one. Routing of data packets through OTVs is called FFG (Fast Forwarding to Gateway) routing mode. It should be stressed that this mode prioritizes forwarding to a gateway if this option is available. Second, it tries to find a

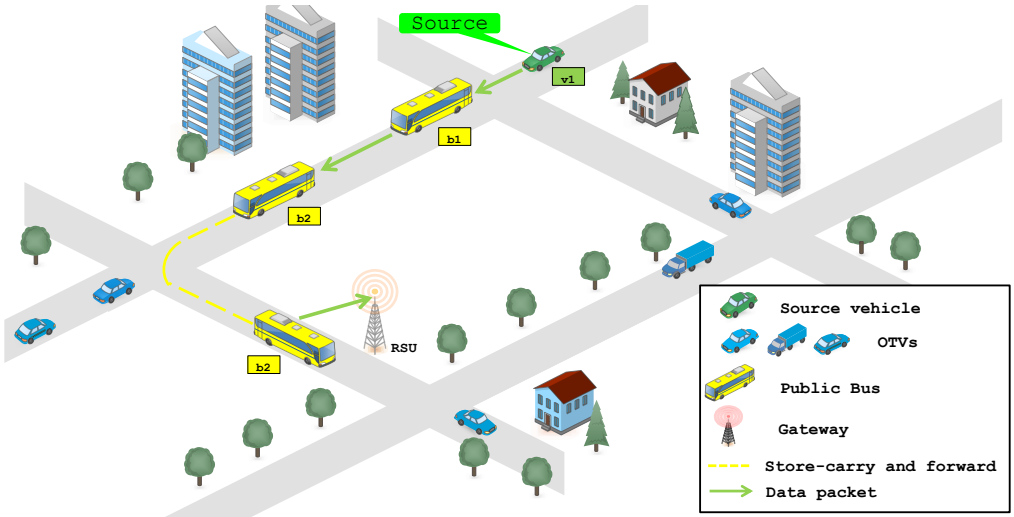


Fig. 1: BRT Architecture.

path to a TGT to minimize the required time for delivering messages to the gateway.

PBs use the store-carry and forward strategy, and they constitute the building block of the backbone. They take advantage of their periodic mobility pattern to learn how to relay data packets faster. Therefore, PBs should pass by a learning process to know their temporal distances to the gateway and the nexthop PB to select. The routing of data packets merely based on backbone buses is called **backbone mode**. It is worth noting that a bus should switch to FFG mode, whenever a gateway or a TGT with a lower temporal distance can be reached through a multi-hop protocol over OTVs.

BRT allows the fast forwarding packets to the gateway, in which intermediate relays switch between the Backbone and FFG mode. The backbone mode is only available for buses. In this mode, buses use the temporal distance table to forward data packets to another adequate nexthop bus that can relay them to a gateway within a known maximum bound of delay. This process comprises only buses of the *Bus Rapid Transit* system and takes advantage of their frequent mobility pattern to know when to forward the data packet, to which bus and the time required to reach the gateway based only on the backbone. The FFG mode aims to boost the forwarding to a gateway or to relay a data packet to a bus which acts as a TGT. The latter is a bus with the lowest temporal distance within n hops (n is a predefined threshold, and setting its value will be discussed later).

The details of BRT will be explained in the following subsections. First, we describe how buses can build the temporal distance based routing tables during the learning process. Next, we describe how the backbone mode is carried out by buses and how they use these tables. After that, we present the FFG mode and how OTVs can reach the gateway and TGTs using a multihop protocol. Finally, we show how to perform the maintenance procedure when a bus is out of schedule.

A. Learning process

Buses will carry out the learning process when BRT is deployed, or after a long time when the *Bus Rapid Transit* system network has been radically changed. This process aims to build BRT tables that are required for making routing decisions, which comprise information about the temporal distances to an RSU and the nexthop bus to select. To that effect, we assume the presence of a central base station that is connected to RSUs, which is responsible for building BRT tables during the learning process. Moreover, we assume a periodic schedule of buses (*i.e.,* the case of a bus out of schedule is rare). Each period of time T is divided into equivalent intervals $[0, t_1[$, $[t_1, t_2[$, $\ldots$, $[t_n, t_{n+1}[$, where all the clocks of buses are synchronized.

Before describing how to get temporal distances based routing tables and for a better understanding, we need to simplify the operation of getting temporal distances to an RSU. To that effect, we first show the learning process for one bus during one period of *Bus Rapid Transit* system. Afterward, we give the generalized process to get this information and how to build these tables for all buses.

1) One bus learning process: Before describing the basics of the learning process, let us give the formal definition of the temporal distance.

Definition 1. The temporal distance between two nodes x and y is the time needed for a packet to reach the node y after leaving the node x. It can be calculated as follows:

$$d_{x,y}(t) = \phi_{x,y}(t) - t \tag{1}$$

Where t is the time of sending the packet by node x and $\phi_{x,y}(t)$ is the time of receiving the packet by the node y.

Definition 2. The temporal distance of a bus x is the temporal distance between this bus and the nearest RSU ($d_{x,RSU}(t)$), and is denoted by $d_x(t)$.

To know the temporal distance of a bus i at any instance of time, this bus should maintain a Local Clock LC_i which counts time from 0 to T. This clock is ticking periodically. At each tick, the bus i should send a learning message (*i.e.,* to its one-hop neighbors) that comprises the current value of its local clock as well as its identifier i. The neighboring bus *i+1*, in turn, should locally maintain records about the bus i comprising its identifier, the latest received information about LC_i (*i.e.,* the highest received value of LC_i), the update time, and the identifier of the previous bus that provided such information (*i.e.,* the previous bus in this scenario is i. This means that the bus *i+1* may serve as a nexthop for the bus i) so that the base station, at the end of the learning process, would be able to indicate the nexthop of each bus to reach an RSU over the fastest path. It is worth noting that we can easily get the minimum time required for delivering a message sent at LC_i from the bus i to the neighboring bus *i+1* (the temporal distance), using the following equation:

$$d_{i,i+1}(LC_i) = \phi_{i,i+1}(LC_i) - LC_i \tag{2}$$

Let us consider that the neighbor *i+1* maintains the identifier i, the latest received value (the highest received value) of LC_i, the identifier of the previous (the bus that provided the latest update which is, in this case, the bus i) and rebroadcasts this information using its periodic learning message to neighbors. In turn, the bus *i+2* (which is only a neighbor to the bus *i+1*) will proceed similarly to the bus *i+1*. Notice that *i+2* can also get the required time for delivering a message sent at LC_i from the bus i to the bus *i+2* (the message from the bus i to *i+2* is the value of LC_i which also represents also the sending time), using the following equation:

$$d_{i,i+2}(LC_i) = \phi_{i,i+2}(LC_i) - LC_i \tag{3}$$

It should be stressed that during the propagation of the learning message information, buses should ignore learning messages from neighbors if they advertise old values about LC_i. Finally, if the process is repeatedly carried out by buses (in the same line), the information: the identifier i and the value of LC_i, that are originated from the bus i will be received by an RSU, if there is a path over time towards that RSU. Moreover, the RSU that received the information (which is sent at LC_i) first at the

time $\phi_{i,RSU}(t)$, has received it via the fastest path. Therefore, the final temporal distance of the bus i at LC_i is as follows:

$$d_i(LC_i) = \phi_{i,RSU}(LC_i) - LC_i \quad (4)$$

Comparing equation (4) with (1), we can notice that equation (4) is only an instance of the equation (1) where:

$$d_{i,RSU}(LC_i) = d_i(LC_i), \text{ see Definition 2} \quad (5)$$

The base station, at the end of the learning process, should have received from all buses updates information so that it can determine the reverse path for each first reception of LC_i. Therefore, the base station will be able to know the nexthop to reach RSU and its corresponding temporal distance at each time t. It should be stressed that the base station (before building routing entries) should trace the sequence of buses constituting the path to reach to the RSU. Even though this path is the fastest found one, it should first filter it from loops (loop filtering process). During one period T, the base station can receive information about i as shown in the following temporal distances table which is ordered according to the sending time values (*see* TABLE III).

TABLE III: Temporal Distance Based Routing table of bus i.

Sending time	Nexthop	Temporal distances
t'_1	N_1	$d_i(t'_1)$
t'_2	N_2	$d_i(t'_2)$
...	...	...
t'_{n-1}	N_{n-1}	$d_i(t'_{n-1})$
t'_n	N_n	$d_i(t'_n)$

It is worth noting that the base station considers only the first reception of the information t'_j. Thus, if a message sent by the bus i at t'_j, it would be received at $t'_j + d_i(t'_j)$ via the fastest path. Another important result is that the learning process is performed in the real world environment, which means that these paths have successfully transmitted the information to the gateway in the presence of permanent obstacles. Thus, we can conclude that BRT is aware of permanent obstacles like buildings.

For a period T=1 week and considering that nexthops change every one second, the number of entries constituting the previous table would be 604800 where 12 bytes are required for each entry. Hence, the required memory for each bus to store the temporal distance based routing table is about 7 MB.

TO guarantee stable paths, we can apply the learning process to only a subset of the neighbors' tables that comprises only buses that will stay within range for a time longer than a predefined threshold th (it is the minimum contact duration). Therefore, we need to expect this time based on information about the speed and the driving direction of buses, and forcing some neighbors to ignore learning messages if the lifetime of their shared link is too small. There are several ways to estimate the Mobility-Based Link Lifetime [29], [30]. The simplest way to estimate the remaining time of two nodes to stay connected consists in using the technique provided in [30]. Assuming the two buses that we would like to estimate the lifetime of their link are a and b. Let also (x_a,y_a) be the geographic coordinates of a and (x_b,y_b) be that of the bus b. Also, let v_a, v_b be the speeds, and θ_a, θ_b be the directions of buses a and b, respectively. Then, the remaining time for the two buses to stay neighbors is:

$$L = \frac{-(AB + CD) + \sqrt{(A^2 + C^2)R^2 - (AD - BC)^2}}{A^2 + C^2} \quad (6)$$

Where,
R is the range of the bus.

$$A = v_a \cos\theta_a - v_b \cos\theta_b$$

$$C = v_a \sin\theta_a - v_b \sin\theta_b$$

$$B = x_a - x_b$$

$$D = y_a - y_b$$

Note that when $v_a = v_b$ and $\theta_a = \theta_b$, L becomes ∞. The other option is to perform another dedicated learning process to predict unstable neighbors. It is also worth pointing out that increasing the value of the threshold th will filter out more candidate nexthops and increase the stability of the backbone.

2) Generalized learning process: We need to generalize the learning process to consider all the buses, and the learning message should have the format depicted in Fig. 2. The learning message format comprises several fields. The *Msg type* field should be equal to 0 to indicate that this is a BRT learning message. The *BRT version* field indicates the used BRT version. The *Bus ID* field is the identifier of the sending bus. The *Vector clocks* field is a vector of elements that allows each bus to record the latest values of LCs of other buses. This vector is initially set to $+\infty$ (*i.e.,* the highest value). The *Error code* is a one-byte size field which will be used to get notified about details of learning process errors. The *Flag* is a one-byte field used for indicating an abnormal situation about this bus affecting the learning performance. The default value is 0. If one bus sets this flag to 1, it means that the learning process is not successful, and thus it should broadcast a failure message to all the learning process participants which they should stop the learning process, and they set their flag to 255.

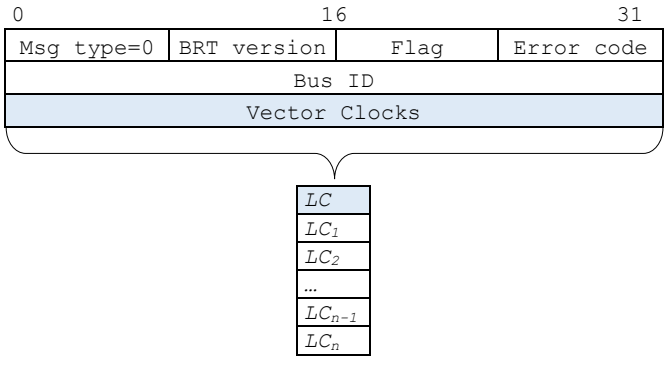


Fig. 2: Learning message format.

During the learning process, all the n buses, in the system, broadcast their learning messages to one-hop neighbors. Thus,

the complexity can be expressed as $O(n)$. Moreover, the size of the learning message depends on the number of buses in the network. Therefore, for big cities having a high number of buses, this process can be carried out within multiple rounds. In each round, buses can exchange vector clocks regarding a subset of buses. Moreover, buses may send learning messages only in the case of updates. In general, performing the learning process in big cities can be tricky due to the difficulties of finding an appropriate time for the learning process. Thus, the complexity of the learning process may increase depending on the affected zones.

In the basic learning process without optimization, each bus should periodically send a learning message to one-hop neighbors. In turn, the size of control messages per second can be expressed as follows:

$$S_{learning} = S \times f \tag{7}$$

Where S is the size of the learning message, and f is the learning message frequency. Based on the learning message format (see Fig. 2), S can be expressed as follows:

$$S = 8 + 4 \times n \tag{8}$$

Where n is the number of buses. For $n = 200$ and $f = 10$, each bus would send 8080 bytes per second. BRT is basically designed to route data packets towards RSUs. However, buses can extract more information from learning messages during the learning process. The locally maintained records by a bus x during the learning process at a given time t, provide information about the temporal distance from other buses towards x, the corresponding previous hops and update times. Thus, a bus can build Temporal Distance Based Routing table to reach any bus at any time. In this case, the backbone can relay data packets towards any bus. Based on the fact that it is possible to know the geographic position of any bus at any time t, the last solution can be easily extended to allow delivering data packets towards any specific location. In the rest of the paper, the details are all about routing towards an RSU.

B. Data Delivery Process

In this subsection, we provide the details of BRT modes and the maintenance procedure.

1) Backbone mode: in this mode, we describe how buses use the temporal distances based routing tables to route data packets. Moreover, we also show how they extract their temporal distances and advertise it to neighbors.

- **Calculating temporal distances and packets routing:** at the end of the learning process, each bus x should have its own temporal distances based routing table, so that it can determine, at any given time t, its temporal distance and nexthop to reach an RSU (*see* TABLE IV).
  Let x be a bus willing to know its own temporal distance at t. It should use the TABLE IV so that it can find an entry e with the lowest sending time that equals to $t'_e > t$ and its corresponding temporal distance $d_x(t'_e)$. The temporal distance of x at t, can be determined as follows:
  $$d_x(t) = d_x(t'_e) + t'_e - t \tag{9}$$

TABLE IV: Temporal Distance Based Routing table of bus x.

Entry number #	Sending time	Nexthop	Temporal distances
...	...	...	...
$e-1$	t'_{e-1}	$nexthop_{e-1}$	$d_x(t'_{e-1})$
e	t'_e	$nexthop_e$	$d_x(t'_e)$
$e+1$	t'_{e+1}	$nexthop_{e+1}$	$d_x(t'_{e+1})$
...	...	...	...

  In each neighbors' table update, if there is a data packet to forward, the forwarding process should look for the appropriate entry, as indicated previously. The data packet should be forwarded to the corresponding nexthop whenever it joins the set of neighbors, without waiting. In fact, the sending time t'_e in TABLE IV, corresponds to the last instant of time at which a message sent from x to reach the gateway at $\phi_x(t'_e)$ via the nexthop_e. Thus, it is required to send the data packet whenever the corresponding nexthop is a neighbor and before t'_e. The abnormal situations will be treated in the maintenance procedure.

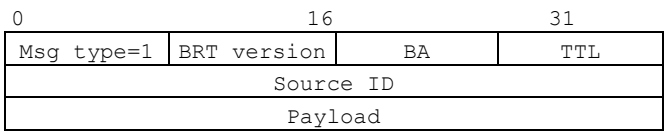


Fig. 3: BRT Data Packet format.

  Fig. 3 presents the BRT data packet format. It includes several fields such as, the *Msg type* field that should be set to 1 to indicate that is a BRT data packet. The *BRT version* field equals to 1. The *BA* (Backbone avoidance) field where its default value is 0. If it is set to 1, it means this packet should not be forwarded according to the backbone mode. *TTL* is the maximum allowed number of hops to reach the destination. The *Source ID* field includes the identifier of the source vehicle. The *Payload* field contains the data to be forwarded to the gateway.
  Routing data packets through the backbone mode is automatic and primarily based on sending time and nexthop records. If the FFG cannot provide a better route, the data packets are stored in the routing queue, waiting for the nexthop indicated by the temporal distances' routing table.
- **Bus status and temporal distance advertisement:** for better functioning, each bus should advertise information about it to its neighbors, particularly, the bus status and the local temporal distance. The bus status allows other BRT participants to avoid considering it as a member of the backbone, if the bus is out of schedule (*i.e.,* it can be considered as an OTV participant if the bus is out of schedule), whereas the advertised local temporal distance is useful for the FFG mode.
  To advertise the previous information, each bus should periodically broadcast to its neighbors a special BRT message called TDA (Temporal Distance Advertisement) message. According to Fig. 4, a TDA message comprises

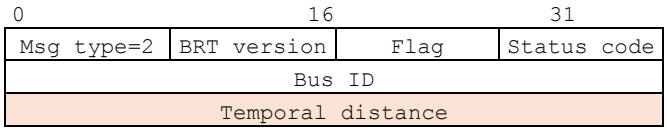


Fig. 4: TDA message format.

a set of fields. The *Msg type* field equals to 2 to indicate that this message is a TDA message. The *BRT version* field is the same, as shown in the previous section. The *Flag* field is set to 1 to indicate that this bus is under an abnormal situation. Thus other BRT participants, should not deal with it as a valid TGT. *Bus ID* is a four bytes field which is used to identify the bus. The *status code* field is used by the maintenance procedure to provide details about the abnormal situations of the bus. The *Temporal distance* field indicates the temporal distance of the sender to RSU.

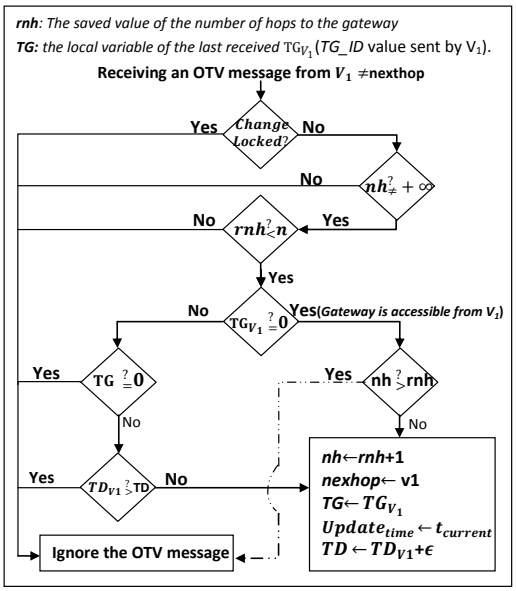


Fig. 5: The FFG mode procedure.

2) Maintenance procedure: this procedure aims to deal with abnormal situations in our system. The process can deal only with two main issues: (i) a bus that is out of schedule and (ii) integrating a new bus.

- ***Out of schedule case:*** even though we assumed that the *Bus Rapid Transit* system is frequent and delays in schedule are rare. We propose the maintenance procedure to alleviate the negative impact of the problem of a missing nexthop. Whenever a bus m is in an abnormal situation (*i.e.,* not following the normal schedule of the learning process), all the calculated temporal distances based on routes comprising m as an intermediate hop are all erroneous. Moreover, the corresponding buses cannot know that their temporal distances are erroneous within some intervals. A bus cannot ensure that selecting another bus would lead certainly to a valid path over the backbone. In fact, the learning process has not considered this situation. To that effect, we need to add to the data packet a specific field BA (Backbone Avoidance) and setting its value to 1 if the bus fails to find the nexthop in the backbone mode. That means that the data packet should not be relayed via the backbone. Another important countermeasure that should be performed by a delayed or out of schedule bus, consists in changing its flag value to 1. Other BRT participants (OTVs and PBs) should ignore this bus as a valid backbone member. To that effect, only FFG mode is available for routing such data packets. Due to the limits of the FFG mode in low-density scenarios, other vehicular routing protocols can be used for routing data packets with a backbone avoidance mode instead of the OTV. To that effect, data packets can still reach a gateway.
- ***A new bus added to BRT:*** any new bus x added to the system and has not participated in the learning process, would not be able to route data packets since it has no routing table. Moreover, the pre-calculated temporal distances of other buses and their corresponding routes have not considered it for finding the near fastest routes during the learning process. However, this bus would be able to build over time its routing table by saving interaction time with neighboring buses having lower temporal distances and would consider them as nexthops, and the bus x should add small duration ϵ to each temporal distance before adding it to its routing table that will be constructed. ϵ represents the expected time for relaying the data packet to a neighboring bus.

The default value for a new bus is set to 2 in the flag field (*c.f.,* Fig. 4). Moreover, it is possible to build its temporal distance based routing table after one period, just by recording the lowest temporal distance of neighboring buses over time. The table is considered valid if there was no encountered bus with a flag equal to 1.

3) FFG mode: this mode allows BRT participants to detect a gateway or a TGT that is reachable via n hops without waiting. To that effect, OTVs broadcast OTV messages periodically to their one-hop neighbors. The format of these messages is illustrated in Fig. 7. An OTV message comprises several fields. The *Msg type* field which is equal to 3 for an OTV message. The *BRT version* field is set to 1 for all BRT messages. The *OTV_ID* field indicates the identifier of the OTV. The *TG_ID* field equals to 0 if a gateway is reachable, it equals to the ID of the accessible TGT if the gateway is not reachable (always within n hops). Finally, the *Temporal Distance* that corresponds to the *TG_ID*. If a neighboring OTV advertises a lower temporal distance, the data packet should be forwarded to this OTV. Each OTV node should maintain local variables (FFG variables) about the number of hops nh to reach a gateway or a TGT if possible, the corresponding temporal distance (advertised by the gateway or the TGT), the nexthop OTV, and the update time. These variables should be updated upon receiving an OTV message according to Fig. 5, and mapped to its advertised OTV message fields so that information about the path will

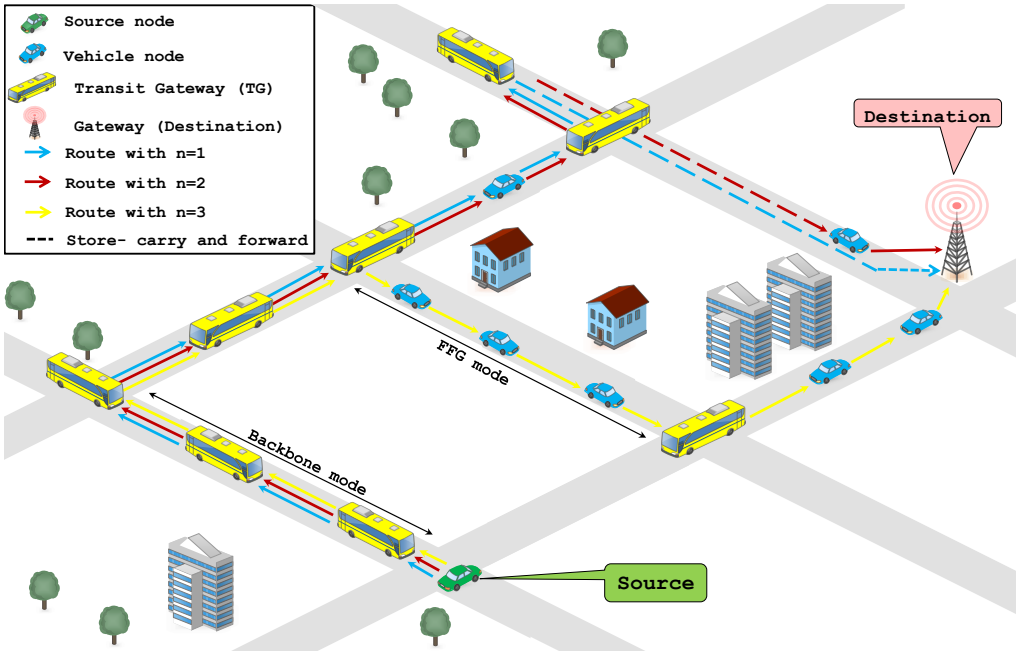


Fig. 6: Impact of n on BRT path.

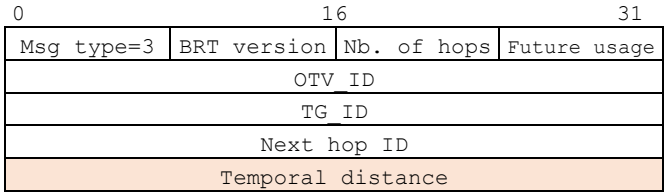


Fig. 7: OTV message format.

be propagated across n-hop neighbors. In the case of receiving an OTV message from a nexthop, these local variables should be directly updated. In turn, an OTV should ignore OTV messages from others if it is their nexthop to avoid creating loops.

Selecting a lower value for n implies that data packets would not be routed to the gateway much farther than the backbone buses. To learn more about the impact of the value of n on BRT routing paths *see* Fig. 6. Indeed, if $n = 1$, the routing path should not include two consecutive OTVs. If $n = 2$, there would be at most two consecutive OTVs. Thus, the higher n is the longer OTVs sequence within the routing path will be. In high-density situations and most of the time, there is no risk to select high values for n. Only one bad scenario (*i.e.,* isolated OTV scenario), low-density situation, in which an OTV enters an isolated zone before transmitting the data packet to a gateway or a TGT. It should be stressed that in this scenario the vehicle carrying the data packets should exceptionally carry the data packet until finding a path to a gateway or a TGT according to the proposed FFG forwarding scheme.

If we would like to avoid or reduce chances of such situations, we can use the technique of expecting the lifetime of link showed in subsection III-A, to reduce chances that vehicles use the store-carry and forward strategy. Remember that OTV paths are generally unpredictable, especially at intersections. In the case of the previous scenario, the OTV exceptionally uses the store-carry and forward waiting for an appropriate update of its FFG variables (*i.e.,* the gateway or the TGT will be accessible through multi-hop protocol).

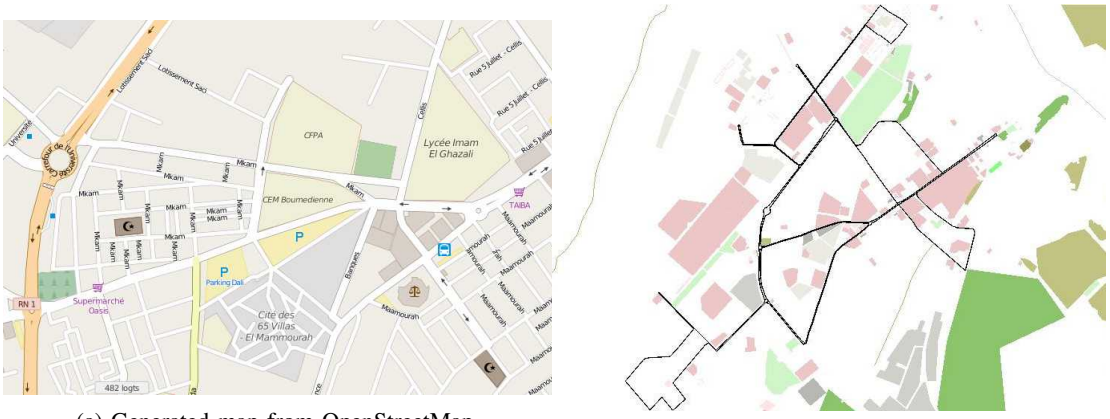
The initial values for these variables are as follows:

$$\begin{cases} nh \leftarrow +\infty \\ TG \leftarrow FFFFFFFF \\ Nexthop \leftarrow FFFFFFFF \\ TD \leftarrow +\infty \\ Update\_time = t_{current} \end{cases}$$

If no update has occurred to these variables after a predefined period, they are reset to their initial values, and it means that no gateway or TGT is accessible within n hops. Subsequently, if an OTV receives a message, with $TG_{ID} = +\infty$, from a neighbor that is its nexthop towards the RSU, it should instantly reset the previous values, and send an OTV message without waiting to advertise the new values. Changes of these values should be locked temporarily to ensure the propagation of the previous information.

In contrast, in the case of accessible gateway $TG\_ID = 0$ and $nh \leq n$, and in the case of accessible TGT $TG\_ID > 0$ and $TD \neq +\infty$.

After presenting BRT functions, its details are summarized in Fig. 9. It shows the process of switching between its modes and how to deliver data packets.



(a) Generated map from OpenStreetMap.

(b) Buses lines' paths.

Fig. 8: Map of the simulation area (N 33°47' 51.5" E 2°51' 58.9").

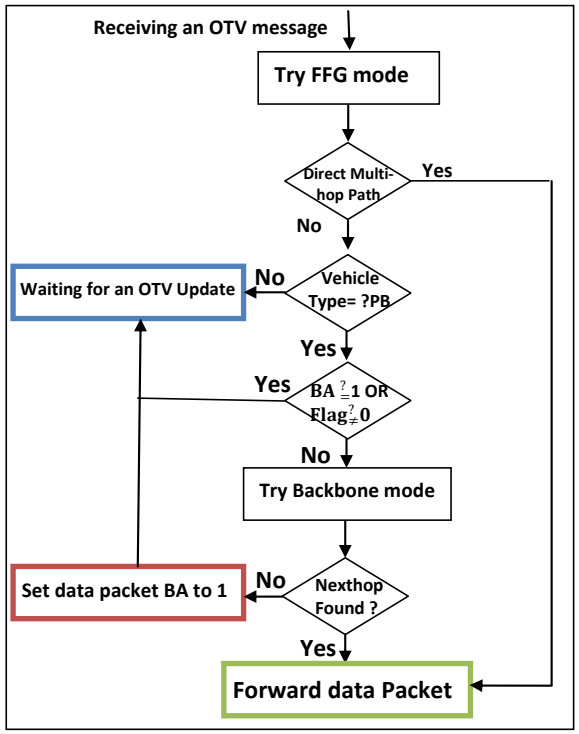


Fig. 9: Packets forwarding.

IV. Performance Evaluation

To evaluate the performance of BRT, we have considered NS-2 [31] as a simulation tool. The topology is based on a real-world map (*c.f.,* Fig. 8) generated from OpenStreetMap [32]. The mobility traces are generated by SUMO [33], [34] based on bus routes which are depicted in Fig. 8(b). The maximum number of roadside units is three. They are installed randomly on the network on bus paths intersections (RSUs can be deployed efficiently considering specific techniques existing in the literature [35]). Other simulation parameters are summarized in TABLE V.

Two main environments have been considered: (i) PBs environment and (ii) PBs+OTVs environment. In the PBs environment, the bus-based routing protocol BUS-VANET [19] has been considered to analyze the performance of BRT (we have considered two variants of BRT according to the th value) in the absence of OTVs. In the PBs+OTVs environment, we have considered another bus-based routing protocol which is MIBR [21]. This protocol employs both buses and vehicles to route data packets. Thus, it is a good candidate to ensure a fair comparison with BRT. Moreover, in this environment BRT is expected to employ the OTV mode. To that effect, we have also defined two variants of BRT according to the value of the parameter n of the OTV mode.

In the first scenario, the overhead is negligible in the PBs environment for both BRT and BUS-VANET. In fact, most of the overhead of BUS-VANET is caused by the presence of vehicles (registration and updating reports). Moreover, BRT employs only the backbone mode in which no control packets are present. To that effect, we focus on the overhead analysis in the PBs+OTVs environment. We have also studied the impact of schedule shifts on the performance of BRT to see how the OTV can deal with abnormal situations.

The simulation results are expressed in terms of the packet delivery ratio (PDR), the End-to-End delay (EED), and the average number of hops. Each point in the obtained results depicts the mean of 50 runs with a confidence interval of 95%.

TABLE V: Simulation parameters.

	Parameter	Value
PHY & MAC	Frequency Band	5.9 GHz
	Pt_	3.57382
	RXThresh_	3.652e-10
	Path loss model	TwoRayGround
	PHY model	IEEE 802.11p
	Bitrate	6 Mbit/s
Scenario	Area size	$5 \times 5\ km^2$
	Simulation time	900 s
	Number of buses	[15, 95]
	Number of OTVs	40
	Bus speed v_{max}	15m/s
	OTV speed v_{max}	15m/s
	Mobility generator	SUMO [33]
Routing	Communication range of Buses	$\approx$ 400 m and 500 m
	Communication range of OTVs	$\approx$ 400 m
	Data size	1 KB
	Number of packets senders	10
	Evaluation metrics	PDR, EED, and HOP

A. PBs environment

In this sub-section, the simulation environment includes only public buses (*i.e.,* no OTVs are present on the roads and the range of PBs is set to 400 meters) to have an overview of the effect of the backbone mode on BRT performance. Figs. 10(a) and 10(b) depict the average number of hops in terms of the number of buses. We can see that for situations of low bus density, both BRT variants have largely optimized the number of hops compared to the BUS-VANET due to its learning process. This allows BRT to transmit data packets over near-optimal paths. That is, BRT forwards data packets only when it is required based on the learning process. However, the performance gap between BRT variants and BUS-VANET decreases when the number of buses increases in the simulation environment. This can be explained by the fact that BRT increases the number of hops in the presence of a high number of buses to optimize the path.

In the same figures, we can see also that BRT with th=40s value has a slightly better performance than the other variant because increasing th would reduce the number of candidates buses to be selected as nexthops.

Figs. 10(c) and 10(d) depict the average end-to-end delay in terms of the number of buses using only the backbone mode. We can see that the threshold th has considerably affected the performance of the backbone mode of BRT. Reducing its value would increase the number of considered buses during the learning process. Therefore, it might increase the number of available paths and enhance their quality. In contrast, lower values for th would affect BRT and make it more sensitive to shifts of the schedule of buses. In the same figures, we can see that the BRT variant with a lower th value presents the best results. However, BUS-VANET outperforms BRT with $th = 40s$ in the case of low-density of buses because it cannot find better paths considering only a limited set of buses.

B. PBs and OTVs environment

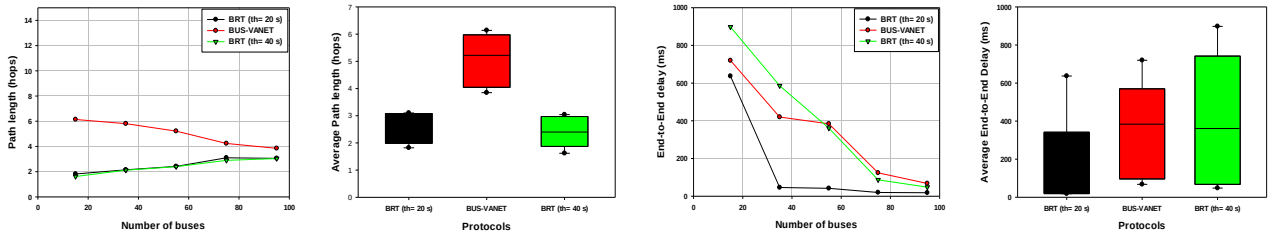
In this sub-section, we include both categories of vehicles PBs and OTVs in the simulation environment, and we extend the range of PBs to 500 meters. This environment would provide an idea about the overall performance of BRT when it combines its modes.

Figs. 11(a) and 11(e) depict the PDR in terms of the number of buses. In this scenario, the number of OTVs is only 40. It should be stressed that the number of OTVs is usually much higher. Thus, increasing this number would hide the performance of the backbone mode, and the FFG mode would be frequently used. We can see that both variants of BRT outperform MIBR because BRT theoretically grantee packets delivery if there is an available path during the learning process. The gap of performance is large in low-density situations because the number of available paths would be minimal, and thanks to the learning process BRT can always find near-optimal paths.

Figs. 11(b) and 11(f) depict the average path length (average number of hops) in terms of the number of buses. Considering the scenario of Fig. 10(a), we can see that in the presence of additional vehicles the average number of hops has increased in both BRT variants because our scheme would not carry data packets that should be forwarded without waiting if adequate nexthops are available. In fact, BRT has been designed to switch to the FFG mode as much as possible. It should switch to this mode whenever there are nodes having lower temporal distances and are accessible using OTVs as relays. Particularly, BRT with $n = 3$ presents the lowest average hops because this variant of BRT excludes paths having more than three consecutive intermediate vehicles.

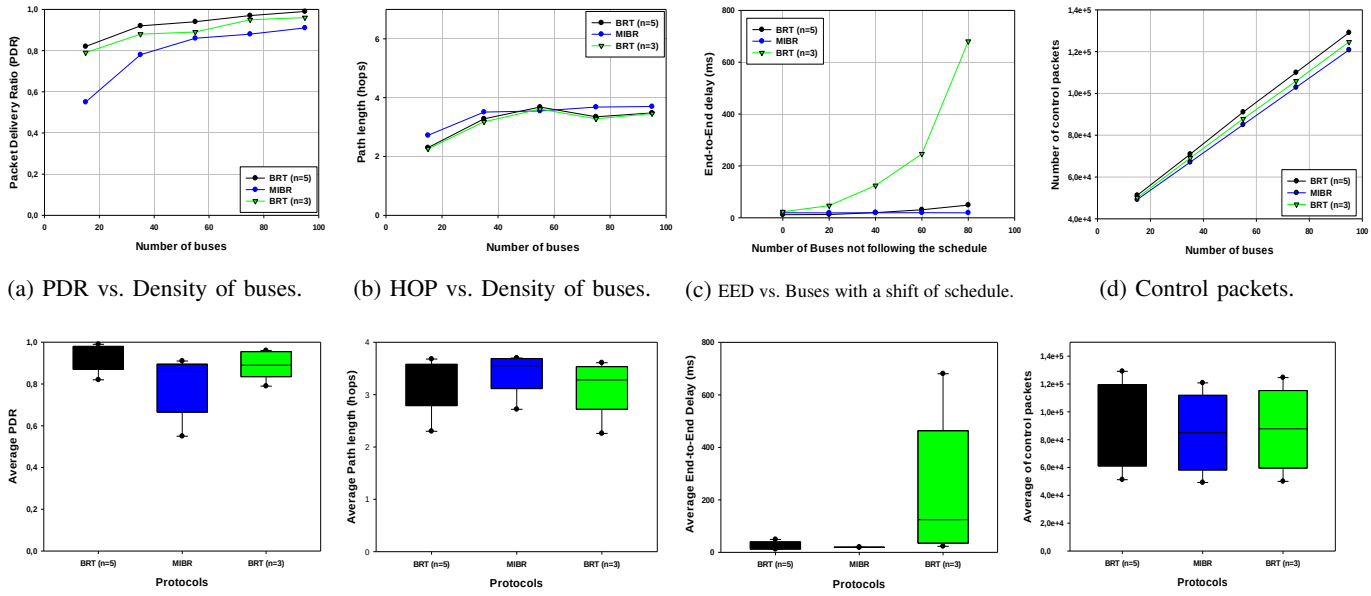
Figs. 11(c) and 11(g) present a very challenging scenario in which we would like to study the degradation of performance of BRT if there is a shift in the schedule of some buses. These figures depict the EED in terms of buses not following the schedule (the shift is greater than the value th that has been considered during the learning process, which is 20 seconds). We can see that BRT provides the best results if less than 40 buses are not following the normal schedule. Increasing the threshold n to 5 allowed the OTV mode to alleviate the impact of the shift of schedule because more vehicles and alternative routes would be considered to deliver data packets. In contrast, MIBR has not much affected by shifts of schedule due to its strategy of forwarding that is not dependent on the schedule.

Figs. 11(d) and 11(h) present the number of control packets. Notice that most of the control packets are periodic hello messages, especially the MIBR protocol. We have reduced the simulation interval to 90 seconds to avoid having large numbers. In these figures, we can see that MIBR present a lower overhead slightly comparing to BRT variants. This can be explained by the fact that the mobility of vehicles provokes vehicles to increase the rate of control packets. Furthermore, the variant of BRT with $n = 5$, which provided the best results



(a) HOP vs. Density of buses. (b) Average HOP. (c) EED vs. Density of buses. (d) Average EED.

Fig. 10: Simulation results in PBs environment (Number of $OTVs = 0$).



(a) PDR vs. Density of buses. (b) HOP vs. Density of buses. (c) EED vs. Buses with a shift of schedule. (d) Control packets.

(e) Average PDR. (f) Average number of hops. (g) Average EED. (h) Average of control packets.

Fig. 11: Simulation results in PBs/OTVs environment (Number of $OTVs = 40$).

in terms of PDR and EED, has generated the highest overhead due to extra OTV messages caused by the mobility of vehicles.

V. Conclusion

Mobility of network participants in vehicular networks constitutes a challenge for designing routing protocols. In fact, the mobility of OTVs is to a large extent unpredictable. Thus it is challenging to select the appropriate nexthop. To that effect, we have designed a new routing scheme called BRT. It classifies the network participants into two categories BPs and OTVs. BPs have a predictable and a periodic mobility pattern, and thus they can learn the nexthop to select after performing an adequate learning process to build routing entries (backbone mode). However, OTVs are used as relays to boost data packets relaying between BPs or directly to the gateway (FFG mode), especially in high-density situations of OTVs. We can also notice that all the obtained simulation results show that BRT outperforms other schemes in the presence of a small number of buses. In such conditions, the backbone is often used and takes advantage of the learning process.

An important scenario to be considered is the case of a bus that detects an out of the schedule nexthop (based on the routing table). This bus should switch the data packet to the backbone avoidance mode. In our paper, we have considered routing such data packet according to the FFG mode. However, other routing concepts existing in the literature can be used. Such concept allows the network to continue functioning in exceptional circumstances.

BRT provides the routing service towards RSU. However, bidirectional communications between RSU and vehicles are sometimes needed. Such service requires low latency. Hence, the backbone mode, which uses the store-carry and forward, cannot be considered. In this case, an extended FFG mode can be used to solve the problem.

As future work, we will consider a dynamic learning process, in which the routes are automatically optimized according to potential changes in the schedule of buses. We also plan to propose an extension of the current work to propose how to deal with the learning process during activities that affect the schedule of buses. Moreover, we present solutions to reduce the overhead according to the network traffic load.

References

[1] Y. Ni, J. He, L. Cai, and Y. Bo, "Data Uploading in Hybrid V2V/V2I Vehicular Networks: Modeling and Cooperative Strategy," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4602–4614, 2018.

[2] D. Kim, Y. Velasco, W. Wang, R. Uma, R. Hussain, and S. Lee, "A new comprehensive RSU installation strategy for cost-efficient VANET deployment," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 5, pp. 4200–4211, 2017.

[3] Y. Wang, Y. Liu, J. Zhang, H. Ye, and Z. Tan, "Cooperative store–carry–forward scheme for intermittently connected vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 777–784, 2017.

[4] O. S. Oubbati, A. Lakas, F. Zhou, M. Güneş, and M. B. Yagoubi, "A survey on position-based routing protocols for Flying Ad hoc Networks (FANETs)," *Vehicular Communications*, vol. 10, pp. 29–56, 2017.

[5] R. I. Meneguette, A. Boukerche, F. A. Silva, L. Villas, L. B. Ruiz, and A. A. Loureiro, "A novel self-adaptive content delivery protocol for vehicular networks," *Ad Hoc Networks*, vol. 73, pp. 1–13, 2018.

[6] T. D. Nguyen, T.-V. Le, and H.-A. Pham, "Novel store–carry–forward scheme for message dissemination in vehicular ad-hoc networks," *ICT Express*, vol. 3, no. 4, pp. 193–198, 2017.

[7] W. Fawaz, R. Atallah, C. Assi, and M. Khabbaz, "Unmanned aerial vehicles as store-carry-forward nodes for vehicular networks," *IEEE Access*, vol. 5, pp. 23 710–23 718, 2017.

[8] E. Ahmed and H. Gharavi, "Cooperative vehicular networking: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 3, pp. 996–1014, 2018.

[9] O. S. Oubbati, A. Lakas, N. Lagraa, and M. B. Yagoubi, "UVAR: An intersection UAV-assisted VANET routing protocol," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2016, pp. 1–6.

[10] O. S. Oubbati, N. Lagraa, A. Lakas, and M. B. Yagoubi, "IRTIV: Intelligent routing protocol using real time traffic information in urban vehicular environment," in *Proceedings of the 6th International Conference on New Technologies, Mobility and Security (NTMS)*. IEEE, 2014, pp. 1–4.

[11] J. A. Dias, J. J. Rodrigues, N. Kumar, V. Korotaev, and G. Han, "REMA: A REsource MAnagement tool to improve the performance of vehicular delay-tolerant networks," *Vehicular Communications*, vol. 9, pp. 135–143, 2017.

[12] A. Benslimane, T. Taleb, and R. Sivaraj, "Dynamic clustering-based adaptive mobile gateway management in integrated VANET 3G heterogeneous wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 3, pp. 559–570, 2011.

[13] S. Boussoufa-Lahlah, F. Semchedine, and L. Bouallouche-Medjkoune, "Geographic routing protocols for Vehicular Ad hoc NETworks (VANETs): A survey," *Vehicular Communications*, vol. 11, pp. 20–31, 2018.

[14] H. Montes, C. Salinas, R. Fernández, and M. Armada, "An Experimental Platform for Autonomous Bus Development," *Applied Sciences*, vol. 7, no. 11, p. 1131, 2017.

[15] F. Zhang, H. Liu, Y.-W. Leung, X. Chu, and B. Jin, "CBS: Community-Based Bus System as Routing Backbone for Vehicular Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 16, no. 8, pp. 2132–2146, 2017.

[16] F. Zhang, B. Jin, Z. Wang, H. Liu, J. Hu, and L. Zhang, "On geocasting over urban bus-based networks by mining trajectories," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1734–1747, 2016.

[17] F. R. Segundo, E. S. e Silva, and J.-M. Farines, "A DTN routing strategy based on neural networks for urban bus transportation system," *Journal of Network and Computer Applications*, vol. 64, pp. 216–228, 2016.

[18] J. Luo, X. Gu, T. Zhao, and W. Yan, "MI-VANET: A new mobile infrastructure based VANET architecture for urban environment," in *Proceedings of the 72nd Vehicular Technology Conference Fall (VTC 2010-Fall)*. IEEE, 2010, pp. 1–5.

[19] X. Jiang and D. H. Du, "BUS-VANET: a bus vehicular network integrated with traffic infrastructure," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, no. 2, pp. 47–57, 2015.

[20] G. Sun, Y. Zhang, D. Liao, H. Yu, X. Du, and M. Guizani, "Bus Trajectory-Based Street-Centric Routing for Message Delivery in Urban Vehicular Ad hoc Networks," *IEEE Transactions on Vehicular Technology*, pp. 1–14, 2018.

[21] J. Luo, X. Gu, T. Zhao, and W. Yan, "A mobile infrastructure based vanet routing protocol in the urban environment," in *2010 International Conference on Communications and Mobile Computing*, 2010, pp. 432–437.

[22] 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 Intelligent vehicles symposium*. IEEE, 2003, pp. 156–161.

[23] 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.

[24] K. C. Lee, U. Lee, and M. Gerla, "TO-GO: TOpology-assist geo-opportunistic routing in urban vehicular grids," in *Proceedings of the sixth international conference on Wireless On-Demand Network Systems and Services, (WONS)*. IEEE, 2009, pp. 11–18.

[25] D. Lin, J. Kang, A. Squicciarini, Y. Wu, S. Gurung, and O. Tonguz, "MoZo: a moving zone based routing protocol using pure V2V communication in VANETs," *IEEE Transactions on Mobile Computing*, vol. 16, no. 5, pp. 1357–1370, 2017.

[26] O. S. Oubbati, A. Lakas, F. Zhou, M. Güneş, N. Lagraa, and M. B. Yagoubi, "Intelligent UAV-Assisted Routing Protocol for Urban VANETs," *Computer communications*, vol. 107, pp. 93–111, 2017.

[27] Q. Yang, A. Lim, S. Li, J. Fang, and P. Agrawal, "ACAR: Adaptive connectivity aware routing for vehicular ad hoc networks in city scenarios," *Mobile Networks and Applications*, vol. 15, no. 1, pp. 36–60, 2010.

[28] M. Tsuru, M. Takai, S. Kaneda, R. A. TSIORY *et al.*, "Towards Practical Store-Carry-Forward Networking: Examples and Issues," *IEICE Transactions on Communications*, vol. E100.B, no. 1, pp. 2–10, 2017.

[29] N. Alsharif and X. Shen, "*i* CAR-II: Infrastructure-Based Connectivity Aware Routing in Vehicular Networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 5, pp. 4231–4244, 2017.

[30] W. Su, S.-J. Lee, and M. Gerla, "Mobility prediction and routing in ad hoc wireless networks," *International Journal of Network Management*, vol. 11, no. 1, pp. 3–30, 2001.

[31] K. Fall and K. Varadhan, "The network simulator (NS-2)," *URL: http://www. isi. edu/nsnam/ns*, 2007.

[32] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, 2008.

[33] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "SUMO–simulation of urban mobility: an overview," in *Proceedings of the Third International Conference on Advances in System Simulation (SIMUL 2011)*, 2011.

[34] G. Sun, Y. Zhang, D. Liao, H. Yu, X. Du, and M. Guizani, "Bus Trajectory-Based Street-Centric Routing for Message Delivery in Urban Vehicular Ad hoc Networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 8, pp. 7550–7563, 2018.

[35] A. Bazzi, B. M. Masini, A. Zanella, and G. Pasolini, "IEEE 802.11 p for cellular offloading in vehicular sensor networks," *Computer Communications*, vol. 60, pp. 97–108, 2015.

Noureddine Chaib received his PhD degree in computer science from the University of Laghouat and serves as treasurer for IEEE Algeria Section. He is currently an associate professor in the Computer Science Department, the Chief Information Officer of the University of Laghouat. He is a member of the Computer Science and Mathematics Lab (LIM) and a member of IEEE Vehicular Technology and the Intelligent Transportation Systems Societies. He received the degree of engineering in computer science from University of Laghouat, in 2007, and the degree of Master of engineering in computer science from the University of Batna, in 2011. His research interests include Security, Privacy, Mobile and Vehicular networks and other networking topics.

Omar Sami Oubbati is an Associate Professor at the Electronics department, University of Laghouat, Algeria and a Research Assistant in the Computer Science and Mathematics Lab (LIM) at the same university. He received his degree of Engineer (2010), M.Sc. in Computer Engineering (2011), M.Sc. degree (2014), and a PhD in Computer Science (2018). From Oct. 2016 to Oct. 2017, he was a Visiting Student with the Laboratory of Computer Science, University of Avignon, France. His main research interests are in Flying and Vehicular ad hoc networks, Visible light communications, Energy efficiency and Internet of Things (IoT). He is a reviewer in many international journals and a TPC member in many international conferences. He is a member of the IEEE and IEEE Communications Society.

Mohamed Lahcen Bensaad is an associate professor of computer science at Laghouat University (Algeria), the same university where he received his engineering degree in June 2003 with a first class reward and a scholarship to study Master in China. He received his Master's degree from Hunan University (China) in June 2006. Upon his return from China to Algeria, in September 2006, he was appointed by the Ministry of Higher Education and Scientific Research as an assistant professor at Laghouat University. He continued his studies in this university where he received his doctorate in information security in March 2014. In addition to teaching and supervising Master's students, Dr. Bensaad served as a member of the Scientific Council of the Science Faculty and currently he is in charge of the Master's Program of Networking and Distributed Systems and he is a member of the Laboratory of Mathematics and Computer Science. His research interests include Information Security, Internet of Things and Machine Learning.

Abderrahmane Lakas received his MS (1990) and PhD (1996) in Computer Systems from the University of Paris VI, Paris, France. He joined the College of Information Technology, UAE University in 2003. He is teaching various courses on computer networks and network security. Dr. Lakas had many years of industrial experience holding various technical positions in telecommunication companies such as Netrake (Plano, Texas, 2002), Nortel (Ottawa, 2000) and Newbridge (Ottawa, 1998). He spent two years (94-96) as a Research Associate at the University of Lancaster, UK. Dr. Lakas has been conducting research in the areas of network design and performance, voice over IP, quality of service and wireless networks. He is a member of the IEEE and IEEE Communications Society. Dr. Lakas is in the editorial board of Journal of Communications (Actapress), and Journal of Computer Systems, Networks, and Communications (Hindawi). He is serving on the technical program committees of many international conferences GLOBCOM, ICC, VTC, etc.

Pascal Lorenz received his M.Sc. (1990) and Ph.D. (1994) from the University of Nancy, France. Between 1990 and 1995 he was a research engineer at WorldFIP Europe and at Alcatel-Alsthom. He is a professor at the University of Haute-Alsace, France, since 1995. His research interests include QoS, wireless networks and high-speed networks. He is the author/co-author of 3 books, 3 patents and 200 international publications in refereed journals and conferences. He was Technical Editor of the IEEE Communications Magazine Editorial Board (2000-2006), IEEE Networks Magazine since 2015, IEEE Transactions on Vehicular Technology since 2017, Chair of IEEE ComSoc France (2014-2018), Financial chair of IEEE France (2017-2019), Chair of Vertical Issues in Communication Systems Technical Committee Cluster (2008-2009), Chair of the Communications Systems Integration and Modeling Technical Committee (2003-2009), Chair of the Communications Software Technical Committee (2008-2010) and Chair of the Technical Committee on Information Infrastructure and Networking (2016-2017). He has served as Co-Program Chair of IEEE WCNC'2012 and ICC'2004, Executive Vice-Chair of ICC'2017, Panel sessions co-chair for Globecom'16, tutorial chair of VTC'2013 Spring and WCNC'2010, track chair of PIMRC'2012 and WCNC'2014, symposium Co-Chair at Globecom 2007-2011, ICC 2008-2010, ICC'2014 and '2016. He has served as Co-Guest Editor for special issues of IEEE Communications Magazine, Networks Magazine, Wireless Communications Magazine, Telecommunications Systems and LNCS. He is associate Editor for International Journal of Communication Systems (IJCS-Wiley), Journal on Security and Communication Networks (SCN-Wiley) and International Journal of Business Data Communications and Networking, Journal of Network and Computer Applications (JNCA-Elsevier). He is a senior member of the IEEE, IARIA fellow and member of many international program committees. He has organized many conferences, chaired several technical sessions and gave tutorials at major international conferences. He was IEEE ComSoc Distinguished Lecturer Tour during 2013-2014.

Abbas Jamalipour is the Professor of Ubiquitous Mobile Networking at the University of Sydney, Australia, and holds a PhD in Electrical Engineering from Nagoya University, Japan. He is a Fellow of the Institute of Electrical, Information, and Communication Engineers (IEICE) and the Institution of Engineers Australia, an ACM Professional Member, and an IEEE Distinguished Lecturer. He has authored six technical books, eleven book chapters, over 450 technical papers, and five patents, all in the area of wireless communications. He disseminated the fundamental concepts of the next generation networks and convergence networks as well as the integration of WLAN and cellular networks. He was the Editor-in-Chief IEEE Wireless Communications (2006-08), Vice President-Conferences (2012-13) and a member of Board of Governors of the IEEE Communications Society, and has been an editor for several journals. He has held positions of the Chair of the Communication Switching and Routing and the Satellite and Space Communications Technical Committees and Vice Director of the Asia Pacific Board, in ComSoc. He was a General Chair or Technical Program Chair for a number of conferences, including IEEE ICC, GLOBECOM, WCNC and PIMRC. Dr. Jamalipour is an elected member of the Board of Governors (2014-16 and 2017-19), Executive Vice-President, Chair of Fellow Evaluation Committee, and the Editor-in-Chief of the Mobile World, IEEE Vehicular Technology Society, a voting member of the IEEE Communications Society GITC and IEEE WCNC Steering Committee, and a member of the ComSoc Education Board, and Conference Boards. He is the recipient of a number of prestigious awards such as the 2016 IEEE ComSoc Distinguished Technical Achievement Award in Communications Switching and Routing, the 2010 Royal Academy of Engineering UK Distinguished Fellowship, the 2006 IEEE ComSoc Distinguished Contribution to Satellite Communications Award, 2010 IEEE ComSoc Harold Sobol Award, and the 2006 IEEE ComSoc Best Tutorial Paper Award, and ten best paper awards. He is one of the most cited researchers in the field of mobile, cellular, and satellite networks with over 10 000 citations.