ROXY: A VEHICULAR AD-HOC NETWORKS ROUTING PROTOCOL IN VEHICULAR CROWDED URBAN ENVIRONMENT

By

Mohamed Abubaker Ibrahim

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
in
Electronics and Communications Engineering

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT

2016

ROXY: A VEHICULAR AD-HOC NETWORKS ROUTING PROTOCOL IN VEHICULAR CROWDED URBAN ENVIRONMENT

By
Mohamed Abubaker Ibrahim

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
in
Electronics and Communications Engineering

Under the Supervision of

Prof. Dr. Hossam A. H. Fahmy

.....

Professor,
Electronics and Communications Engineering,
Faculty of Engineering, Cairo University

Dr. Omar A. Nasr

.....

Associate professor,
Electronics and Communications Engineering,
Faculty of Engineering, Cairo University

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT

2016

ROXY: A VEHICULAR AD-HOC NETWORKS ROUTING PROTOCOL IN VEHICULAR CROWDED URBAN ENVIRONMENT

By
Mohamed Abubaker Ibrahim

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
In
Electronics and Communications Engineering

Approved by the
Examining Committee

_____
Prof. Dr. Hossam A. H. Fahmy, Thesis Main Advisor

_____
Associate Prof. Dr. Tamer Abdel Mottalib Elbatt, Internal Examiner

_____
Prof. Dr. Salwa H. El-Ramly, External Examiner
(Electronics & Communication Eng. Department Ain Shams University)

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT

2016

Engineer's Name: Mohamed Abubaker Ibrahim
Date of Birth: 06/09/1981
Nationality: Egyptian
E-mail: e_m_apollo@yahoo.com
Phone: 01141099745
Address: 8 Jeddah St., Shooting Club, AL Doqi, Giza, Egypt
Registration Date: 01 / 10 / 2010
Awarding Date 2016
Degree: Master of Science
Department: Electronics and Communications Engineering

Supervisors: Prof. Dr. Hossam A. H. Fahmy
Dr. Omar Ahmed Nasr

Examiners: Prof. Dr. Hossam A. H. Fahmy (Thesis Main Advisor)
Dr. Tamer Abdel Mottalib Elbatt (Internal Examiner)
Prof. Dr. Salwa H. El-Ramly (External Examiner. Prof. Dr. in Electronics & Communication Eng. Department Ain Shams University)

Title of Thesis:

ROXY Ad-Hoc Network Routing Protocol in Vehicular urban Environment

Key Words:
VANET; Ad-Hoc; Routing protocol; Position base routing protocol; collision avoidance

Summary:

Vehicular Ad-hoc Network (VANET) is a wireless network between vehicles equipped with on-board units. VANET can be used in safety applications, where low latency is of high importance, or for entertainment and Internet access. VANET is a part of the overall vision of Intelligent Transportation Systems (ITS). The nature of the VANET network is dynamic due to the different speeds of vehicles on the roads, and their sudden change of directions/lanes. Therefore, packet routing algorithms in VANETs should take into consideration the fast topology changes. This is usually done by sending beacon packets by the vehicles to all other vehicles, and updating the routing table frequently. This is considered as a high overhead in VANET networks that tremendously decreases the effective throughput.

In this work, we introduce Receiver as prOXY (ROXY) VANET routing protocol to overcome the limitations of other protocols. ROXY employs the geographic routing scheme with opportunistic next hop selection to increase the overall network throughput in VANET networks. ROXY main objective is to solve the increasing congestion and collisions in heavy loaded network traffic. The improvement in VANET network is caused by tackling three issues. Firstly, we utilize the spatial resources by using two directed antennas on each vehicle, one on the front side, and one on the back side. The directed antennas are used to send and receive packets from two different directions simultaneously, which efficiently use the transmission medium and increase the overall throughput. Secondly, ROXY is a distributed routing algorithm, where the decision of next relay selection does not need communication with a central entity. Lastly, we reduce the transmission time delay through proposing a new collision avoidance algorithm that takes advantage of the directed antennas and the street topology in urban areas. Simulation results show the superiority of our algorithm in terms of overall network throughput compared to other algorithms.

Acknowledgements

First of all I must thank GOD for his great mercy supporting me to be what I have become.

I would like to thank my advisors, Prof. Dr. Hossam A. H. Fahmy and associate Prof. Omar Nasr for giving me the opportunity to work in a fruitful research environment and for their continuous guidance, patience, and support, as well as for their successful discussions and encouragements.

Most importantly, I do like to thank my family for their continuous support.

Table of Contents

List of Tables

List of Figures

List of Nomenclature

ACK	Acknowledge
CCH	Control Channel
CSIA	Current sender IP Address
CSMA/CA	Carrier sense multiple access with collision avoidance
CSMA/CD	Carrier sense multiple access with collision detection
CSP	Current sender position
CSU	Channel sender using
DHCP	Dynamic Host Configuration Protocol
DIA	Destination node IP Address
DOT/CP	Denial Of Transmission with Collision Prevention
DP	Destination node position
DSRC	Dedicated Short Range Communication
FNIA	FROM Node IP Address
In	Index
ITS	Intelligent transportation system
MANET	Mobile Ad hoc Network
NAS	Not Allowed to Send
NIA	New IP Address
Nix	New Index
PI	Packet Index
PTAIF	Packet Type the ACK Issued For
QoS	Quality of Service
RNIA	Receiver Node IP Address
ROXY	Receiver as Proxy
RREP	Route Reply
RREQ	Route Request
RSU	Road Side Unit
SCH	Service Channel
SIA	Source node IP Address
SP	Source node position
TA	Time to create and send

TRS	Time to Retransmit
TS	Time to fully send
UTC	Coordinated Universal Time
V2I	vehicle to infrastructure
V2V	vehicle to vehicle
VANET	Vehicle Ad-hoc Network

Abstract

Vehicular Ad-hoc Networks (VANET) is a wireless network between vehicles equipped with on-board units. VANET applications vary from safety applications, where low latency is of high importance, to entertainment and Internet access. VANET is a part of the overall vision of Intelligent Transportation Systems (ITS). Vehicles in VANET network have dynamic nature due to the fast speed variation of the vehicles on the roads, and their sudden change in directions/lanes. Therefore, packet routing algorithms in VANETs should take into consideration the fast topology changes. Usually this is achieved through sending beacon packets by the vehicles to all their neighbors, and updating the routing table frequently. This is considered as a high overhead in VANET networks that tremendously decreases the effective throughput.

In this work, we introduce Receiver as prOXY (ROXY) VANET routing protocol to overcome the limitations of other protocols. ROXY employs the geographic routing scheme with opportunistic next hop selection to increase the overall network throughput in VANET networks. ROXY main objective is to solve the increasing congestion and collisions in heavy load network traffic. ROXY improves the system by tackling three issues. Firstly, it utilizes the spatial resources by using two directed antennas on each vehicle, one on the front side, and one on the backside. The directed antennas are used to send and receive packets from two different directions simultaneously, which efficiently use the transmission medium and increases the overall throughput. Secondly, ROXY is a distributed routing algorithm, where the decision of next relay selection does not need communication with a central entity. Lastly, it reduces the transmission time delay through proposing a new collision avoidance algorithm that takes advantage of the directed antennas and the street topology in urban areas. Simulation results show the superiority of our algorithm in terms of overall network throughput compared to other algorithms.

Chapter 1. Introduction

Vehicular Ad-hoc Networks (VANETs) are wireless networks between vehicles moving or parking in the streets. Vehicles are equipped with onboard units and computerized control units in VANET. The onboard units can transmit to 300m range and they do not require mobile or stationary infrastructure to create the network for some applications. However, to have a much better services for the VANET customers, a good designed infrastructure is needed and spread through the streets in the shape of road side units (RSUs) to help leveling up the services and to serve as an Internet gateway. Packets going out of source transmission range need to be sent through multiple nodes in the network to reach the final destination.

VANETs are considered as a subclass of mobile ad hoc networks (MANETs) with some differences. The similarity between MANET and VANET are that in both networks the nodes are mobile, except for VANET, the vehicles are not moving randomly but they move in a controlled mobility manner, which makes their mobility pattern more predictable. The difference between the two networks is that VANET's vehicles move in various speeds from 20km/h to 60km/h in urban road and can exceed 130km/h on highways. Also the difference in nodes directions according in which side of the road they are moving on. All that leads to unstable connections and rapid changes in the network topology. Therefore, developing routing protocol, which is topology independent or at least can be adapted to the rapid changes in the network topology, is needed to achieve a decent VANET system

People spend 10 to 15% of their travel time in traffic jams [1]. VANET with a good designed infrastructure can leverage the Intelligent Transportation Systems (ITS) through the real time information gathered from the vehicles to achieve a better driving experience by advising drivers to change their travel path to a better one with no or less congestions. In addition, it can enhance the safety application to avoid accidents, which will save human lives and improve productivity. VANETs can be used to support different applications. For example:

- Accidents and hazards situation in the area can be reported to other users of VANET before reaching it, and the authority can be informed automatically e.g. the police, ambulance, or fire department to solve the situation as soon as possible,
- Sudden breaks in highways with the presence of fog or any sudden break downs for the vehicles will be reported fast to prevent accidents from happening,
- Traffic state information in the vehicles' traveling path (e.g. traffic congestion, blocked road for maintenance, or any sudden reason) can be gathered in real time and the system automatically update the path and inform the driver that a sudden issue occurred in the path chosen and a detour is advised,
- Empty parking spaces. Vehicles can reserve empty space even before they can reach their destination which is very helpful to the user to lower the congestion in the crowded zones,
- Reservation for restaurants, hotels, theater houses etc. While on the move, last minute change of plans, or if the person is new to the area, the reservation can be handled on the run through a simple application. It can reserve, lead the way to the best route, and reserve the place to park automatically without the need to waste time and effort,
- Internet access for streaming, communication, gaming, or any other service the Internet can offer.

To accomplish that, communication is needed between moving vehicles with each other's (V2V) and vehicles with the infrastructure (V2I) to be done as efficiently as possible with low time delay. Furthermore, RSUs deployment is expensive, and because of that, not all the vehicles will be in the RSUs transmission range. However, the vehicles should be connected to the RSUs, and this is accomplished through multi-hop transmission using a routing protocol designed for the heavy loaded VANET environment, to transmit the data from vehicles to the RSU and vice versa with low collisions and reduced time overhead.

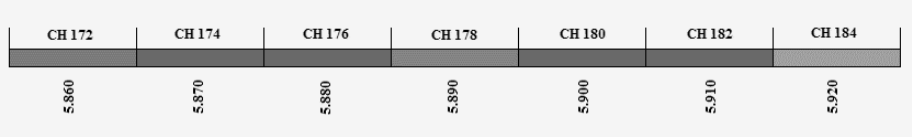


Figure 1.1: VANET channels frequencies available

The IEEE 802.11p/WAVE Wireless Access in Vehicular Environments [2] standard allocates seven channels (10MHz bandwidth per-channel centered around the frequency 5.89GHz) in the Dedicated Short Range Communication (DSRC) frequency band 5.85-5.925 (CH 172-CH 184) as shown in Figure 1.1. The seven channels are distributed as follows: one channel is called Control Channel (CCH) for sending emergency and beacons and the other six are Service Channels (SCH) [3] [4] [2] [5]. Some papers use channels in a different way: CH 172 and CH 184 are reserved for special purposes. CH 178 for CCH and the remaining four channels for SCH [6]. Others use CH 172 for safety transmission and CH 178 as CCH while CH 184 for long range high power transmission and the remaining four channels for SCH as in [7] and [8]. Vehicles equipped with single network devices must alternate between the CCH and SCH. The method used is to divide time into integer number of sync intervals with fixed length of 100ms. A sync interval is the sum of a CCH interval and a SCH interval. 50ms is used for each channel. To coordinate the access to these channels, vehicles must have time synchronization by using coordinated universal time (UTC). In addition to that, there is 2ms guard interval at the beginning of each interval for devices switching [9]. The network devices channels can be classified to four modes [6], as shown in Figure 1.2:

1. **Continuous access:** in this mode, the network devices access only the CCH to send and receive the emergency and control messages.
2. **Alternating access:** the network device alternates between the CCH and SCH in fixed interval.
3. **Immediate access:** In this mode, the devices must switch to the CCH every new sync interval and when the transmission on the CCH ends then the devices can switch back to the SCH.
4. **Extended access:** the network devices work in the SCH as long as there is no transmission on the CCH.

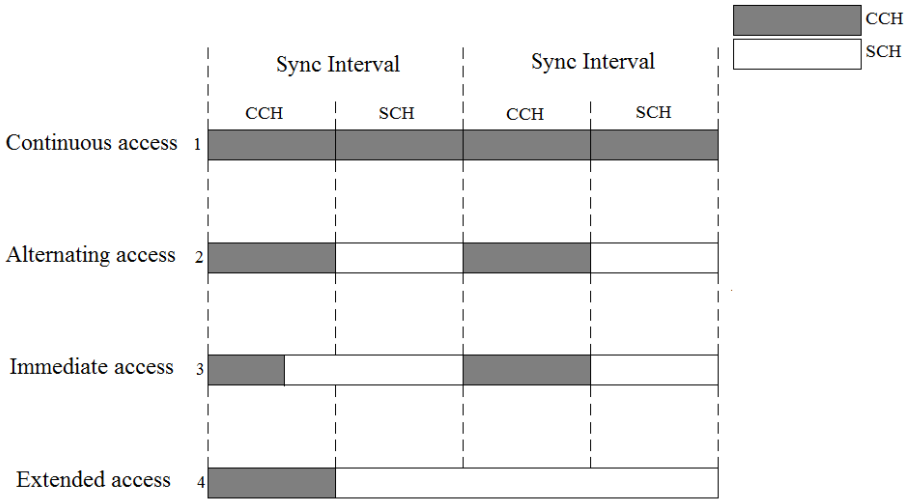


Figure 1.2: Channel Coordination [6]

The vehicles choose the service channel in one of two ways. Local selection is done by sensing the spectrum and allocating the idle or best channel for transmission [10]. The Decentralized Location Based Channel Access Protocol for vehicle to vehicle communication (DLCAP) [11] proposes that vehicles should allocate channels to use according to their location. The other way is a centralized way. This is done by leaving the allocation and management of channels to the RSU as in CMAC [12] and many other protocols [13]. In a crowded network traffic area, if a decentralized protocol is used, there will be a fairness issue. On the other hand, the centralized channel allocation protocols will increase the load on the system and affect the QoS.

1.1 Motivation

Due to the significant development in safety, efficiency, business, and entertainment applications and the amount of data the average person consumes, a good protocol to manage transmission is needed. Large number of applications if combined with good and well-designed VANET network can make life safer and more productive. For that, developing a routing protocol is needed to guarantee the delivery of the data packets from sources to destinations with low time overhead and reduced collisions with relatively acceptable data rates.

There is a need for a routing protocol that considers more than just routing the packets from source to destination. The protocol needs V2V transmission schemes that guarantee high packet delivery rates with low time delay to increase the overall throughput from source to destination, combined with a collision prevention protocol designed for VANET system in heavy load network.

The proposed routing protocol should satisfy the following:

- Minimizing the packet traffic by using the minimum number of packets required for successful delivery from sources to destinations
- Locating the destination without relying on information query or path discovery, which floods the network,

- Minimizing or removing, if possible, the collisions from the system by using an effective scheme in VANET system,
- Removing the dependency on beacons to gather information about the neighbor vehicles to route packets,
- Allowing emergency messages to have more time to be transmitted by reducing the need for beacons since they share the same channel,
- Increasing the time of the CCH and SCH and, removing the delay caused by the guard interval,
- Organize the arrangement for transmission and retransmission when collisions occur,
- Fairly distribute the channels between the nodes as much as possible.

1.2 Objectives

The objectives of this thesis is to introduce a routing protocol for VANET system with the following features:

- Minimize the number of packets used in the selection of the next hop node process in order to reduce the delays and collisions,
- Reduce the hop count by selecting the node closest to the destination,
- Reduce the time overhead to send a packet from source to destination,
- Increase the throughput,
- Minimize the collisions by proposing new MAC protocol,
- Test the protocol with a simulator for VANET.

1.3 Thesis Outline

The remainder of this thesis is organized as follows. Chapter 2 provides the background about VANET routing protocols and outlines the relevant related work in the literature. It also shows the problems in these protocols. Chapter 3 explains the proposed ROXY routing protocol. Chapter 4 outlines the simulator and the simulation setup used to validate the effectiveness of ROXY. It also provides a comparison with existing BRAVE routing protocol [14]. Chapter 5 shows the conclusion of this thesis.

Chapter 2. Related Works

Researchers paid great attention for VANET routing protocols in the last few years [15]. This is due to VANETs environment due to the rapid topology changes, and hybrid network architectures. In this chapter, we will go through some of the previous works about the previously introduced routing protocols. Routing protocols can be categorized into three types [16]:-

1. **Topology based routing:**

This type of routing protocols transmits packets through a pre-created and maintained multi-hop path consisting of vehicles and RSUs in the streets. This type of protocols are subdivided into two types

- **Proactive**

Such as OLSR [17] and DSDV [18], this class of protocols uses routing tables to forward packets. They broadcast and flood control packets among nodes in order to create and maintain paths between nodes, even though some of the paths may not be needed or might not be used. The next hop node is preselected to the destination since the paths are maintained all the time. After creating the paths, nodes do not need to waste time on sending any route request packets (RREQ) or route reply packets (RREP) messages to send packets. Instead, they use the route already established. However, these protocols have a major drawback in heavy network traffic in the path-creation phase. In addition, since there is interference, the routes are not stable, which will need a lot of maintenance and re-finding the routes. This needs flooding the system repeatedly with control packets that will affect the network efficiency and may lead to network failure.

- **Reactive or on-demand**

Such as AODV [19] and DSR [20], the protocols from this type create and maintain paths to a specific destination only when needed. The nodes that need to forward packets start by flooding the system with RREQ packets to find the best path to a specific destination. When the destination finds the best path, it will create RREP carrying the reverse path in the header of the packet and send it to the source. When the source receives the RREP, it will use the chosen path and send the data packet in the correct path.

2. **Position based routing**

Such as GPSR [21] and GPCR [22] protocols. This type chooses the next hop node in the direction of the destination. Nodes using this type of routing protocols transmit the packet to the neighbor node closest to the destination. To select a node, the sender needs to have a neighbor list created beforehand. This can be done by transmitting periodic HELLO messages (beacons) carrying the nodes ID, position, speed, direction, and any other information that help to facilitate the routing process. Depending on the source node position and the destination position, the source node decides which one of the nodes is the best to receive the packet as the next hop node. The position and ID of the destination is stored in the header of that packet as well as the ID and position of the chosen next hop node.

3. **Map based routing**

Such as GSR [23] takes advantage of the node's position and the knowledge of the area topology to find the shortest path to forward the packet through. A list of junctions is usually attached in the packet header, while the node-to-node transmission is done using greedy forwarding.

2.1 Routing Protocols

2.1.1 OLSR (Optimized Link State Routing protocol) [17]

This protocol uses a routing table created using topology control packets to forward packets to their final destination. The protocol defines the term "efficient" for transmission as transmitting packets to all nodes without duplications using multipoint relays MPRs. Each node selects a symmetric number of one hop neighbors as MPRs to relay their messages and they relay the node messages. The OLSR uses this technique to prevent unnecessary duplications. OLSR uses periodic hello messages to sense the one and two neighbor nodes and to create the MPRs. To have a full network topology map and create the routing table, all nodes with non-empty set of MPRs periodically create TC-messages (topology control messages) and spread them through the networks any node that needs to send a packet uses the information available in the routing table.

2.1.2 DSDV (Highly dynamic Destination-Sequenced Distance-Vector routing for mobile computers) [18]

This protocol uses an updated routing table to transmit the packets that contains all routes to all possible destinations. The nodes send periodic update messages to confirm their availability and the routes they have. Moreover, if there is new significant information available, they send immediate updates.

2.1.3 AODV (Ad hoc On demand Distance Vector) [19]

AODV uses route discovery to predefine the packet's path. When a source node needs to send packets to a destination node, it transmits a route request (RREQ) to all neighbor nodes. The receivers of the RREQ either send back a route for the destination if it was available or rebroadcast the RREQ. The receiver then records the first sender ID from the RREQ received to be the reverse path for the incoming RREP. When a RREQ packet arrives to the destination or a node has a route for the destination, a route reply packet (RREP) will be sent to the source node. If a failure occurred to an active link, this failure is reported immediately to the source node to activate a maintaining phase by propagating a new route discovery to find a new path. However, this method of route discovery creates time overhead.

2.1.4 DSR (Dynamic Source Routing) [20]

In this protocol, if a source node needs to send a packet to a destination node, it broadcasts RREQ to all neighbor nodes to rebroadcast it again by them until it reaches the destination. Upon receiving the RREQ by the destination, the destination issues an RREP to send it back to the source node through the route it came through. If the destination received more than one instant of the same RREQ through different routes, it will send RREP through each one of those routes to the source so that it can choose the best route to use, and leave the other paths as alternative routes in case of failure.

2.1.5 GPSR (Greedy Perimeter Stateless Routing) [21]

This protocol is a greedy protocol that forwards the packet to the node closest to the destination. This protocol chooses the closest node to the destination from the neighbors list. The nodes create the neighbors list using the periodic beacons sent by all the nodes with random intervals to reduce the collisions.

To recover from a local maximum, GPSR uses the right hand scheme. When drawing two circles one centered at the destination with a radius equal the distance from the destination to the sending node (*D* to *x*) as shown in Figure 2.1. The second circle centered at *x* with radius equals to *x* transmission range if there was no nodes in the area created (marked as void in the figure) then the right hand scheme will take over.

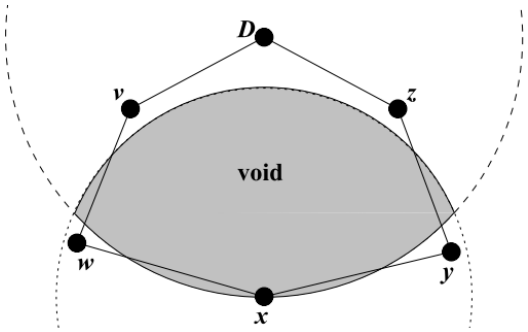


Figure 2.1: GPSR right hand protocol to recover from local maximum [21]

The scheme states that the next hop node is chosen from one of the one-hop neighbors in the counterclockwise direction from the line linking the sender to the destination. From Figure 2.1, the sender node *x* chooses a node from the one hop neighbor nodes that have the smallest angle created from the line *xD* (*D* is the destination) in the counterclockwise direction, in this example the next hop node is *w*. From node *w*, the next hop node will be *y*, but since the line *wy* crossing the line *xD*, this node will be ignored and the next node to be selected is the first node in the counterclockwise direction from the line *wx* that does not cross the line *xD*. From that, the node chosen is *v*, and finally the packet reaches *D* from the link *vD*.

2.1.6 BRAVE (Beacon-less Routing Algorithm for Vehicular Environments) [14]

This protocol forwards the packets to their destination through relaying them to the closest node to it, without the need for beacons. BRAVE goes through multiple packet transmissions to send one packet from sender to the next hop node. It starts with the sender broadcasting the packet to all the nodes in the transmission range carrying the position of the

final destination. All the receiver nodes check their distance to the destination and compare it with the distance from the sender to the destination. If their distance is greater than the sender distance, then they drop the packet and no further action is needed. If the distance is less, these nodes send RESPONSE packets to the sender storing their ID and position. The sender chooses the closest one to the destination and sends a SELECT packet containing the ID of the selected node. The selected node then sends an ACK packet acknowledging the reception and announcing the end of this transmission. The process keeps going until the packet reaches its destination.

2.1.7 GSR (Geographic Source Routing) [23]

This protocol forwards the packet using position based routing, with the help of a map, to choose the path the packet must take. The source node floods the network with *position request* packets for the destination. When the destination receives the position request packet, it sends a reply to the sender to inform it with the position. The source then computes the path the packet should take and store the list of junctions that create this path in the packet header.

2.1.8 ASAR (Adaptive State Aware Routing) [24]

This protocol assumes that there is fixed equipment in every junction along the roads, and they can connect with each other via cellular networks. Whenever a vehicle enters a road, it sends its ID to the fixed equipment so there will be a record of the number of vehicles in each section. When a source node has a packet to send to a destination, it first gets the destination position via Location Services Agreement and attach it in the packet header. Then, it forwards it to the fixed equipment. The fixed equipment chooses the best route according to the delay the streets respond with (number of cars in that street) and if there is a vehicle in the fixed equipment transmission range. Since the destination on the move, the packet may reach the destination position while the destination has already left that area. Hence, the sending node informs the destination node through cellular network, so that if the destination node leaves its location, it will inform the fixed equipment about its new location to be updated by the source.

2.1.9 BCRPV (Broadcast Control-Based Routing Protocol for Internet Access in VANETS) [25]

This protocol establishes routes on demand. It defines four types of vehicles in the system (1) ordinary vehicles (2) forwarder vehicles to rebroadcast RREQ (3) gateway vehicles acting as a gateway with both WLAN and WWAN (4) designated gateway to rebroadcast RREQ.

When a source has a packet to send to a destination with no pre-created route, the source broadcasts an RREQ to forwarders (two maximum). They are selected according to their number of neighboring gateways, and the distance from the RREQ sender.

They proposed a scheme to reduce the flooding created by the blind rebroadcasting of the RREQ. When a node receives an RREQ, it records the path the RREQ traversed. If the receiver does not have a route to the destination, it checks if the sender selected it to

retransmit the packet. The protocol chooses the relay node by classifying the nodes and deciding the nodes that will rebroadcast the RREQ as follows:

1. Designated gateway will rebroadcast the RREQ,
2. Gateway if there is no designated gateway selected,
3. Forwarder if there is no designated gateway or gateway selected,
4. Ordinary node if none of the above selected to rebroadcast the RREQ.

Upon the reception of the RREQ by the destination or an intermediate node, which has a fresh route to the destination, a RREP is created and sent back to the source using the reverse route created by the RREQ.

2.1.10 CAR (Connectivity-Aware Routing) [26]

CAR protocol uses periodic HELLO messages to enrich the neighbors list so it can have a fresh general view of the one-hop nodes. The interval of the beacons is adaptive to the number of neighbors to have an accurate neighbor list, i.e. the fewer the neighbors, the more frequent the HELLO messages sent.

CAR presents a new concept called *guards*. Guards represent temporary state information tied to the area the guard in. There are two types of guards: *standing guards* and *traveling guards*. Guard is kept alive by the node in the area and if a node has a guard, it can interfere with packets transmission by redirecting them or adding information to the packets. More than one guard can be in a node to fulfill different tasks and when a task is done, the guard can be removed even before its time to live expire.

When a source has a packet to a destination, it starts by locating its position using Preferred Group Broadcasting (PGB). A path discovery (PD) broadcast carries the node information to all neighbors to rebroadcast to the destination, every node retransmits the PD rewrite the *previous forwarder coordinates/velocity vector* with its own

Whenever a node near a crossroad or a curve-road receives a packet, an anchor is recorded in the packet header carrying the coordinates of the current and the previous forwarder nodes.

Upon the reception of a PD, the destination waits for some time in case more PD can arrive. When the time ends, the destination chooses the best path available from the PD received, and a route reply is created and sent back to the source using AGF [27] via the recorded anchor points. The source uses the information in the route reply to greedy forward the packets over the anchored path.

2.1.11 MRRP (Multipath Route Restoration Protocol) [28]

When a source needs to send a packet to a destination, it starts by sending RREQ to the neighbors in the destination direction and waits for the RREP. MRRP uses *advertised hop-count* mechanism, as in AOMDV [29], for the identification of multiple loop-free paths. For the route maintenance, the protocol uses similar scheme to CAMP [30], where the intermediate nodes calculate average queue length and send to the source node a Route Streamline Packet (RSP) if a threshold is exceeded. Hence, the source can choose the next best path from the routing table.

2.1.12 IDVR-PFM (Intersection Dispatch-based VANET Routing protocol with Parked vehicles Forwarding Mechanism) [15]

IDVR-PFM utilizes the mobile and parked vehicles to forward packets. The protocol assumes there is a fixed node in each intersection that is responsible for route selection and for the detection of the optimal direction for data forwarding. The vehicles maintain a neighbor list through the periodic transmission of beacon messages carrying the node information in it, e.g. time of sending the node ID, position, and type (fixed, parked, or mobile).

When a node has a packet to forward to a destination, the sender starts by searching in its neighbor list for the destination. If it was not there, it will search for a fixed node to forward the packet to it. If there was no fixed node in the neighbor list, the sender uses greedy forwarding to forward the packet in the direction of the destination. When the packet reaches a fixed node, it chooses the best path for the packet to be sent through it.

2.1.13 RBVT (Road-Based using Vehicular Traffic) [31]

This protocol presents two types of routing: reactive or on-demand (RBVT-R), and proactive (RBVT-P).

2.1.13.1 RBVT-R

When a source tries to find a route to a destination to send a packet through, it starts by flooding route discovery packets RD. When an intermediate node receives the packet, it checks before retransmitting it if the current receiver is in a different road segment than the last sender node (meaning the packet passed an intersection). If this was true, then the receiver node will append this intersection ID in the RD's header.

When the destination receives the RD packet, a route reply RR is created from the sequence of intersection IDs appended in the RD received, and it will be sent back to the source through that route. If another RD arrives to the destination from the same source but through another route, a new RR is created if and only if that new route was better than the old route created. In this case, it will send the new RR to the source to use. When the source node receives the RR, it will use that route to send its packet through the list of intersections delivered through the RR. The method to forward the packet inside the same road segment is the geographic forwarding.

If the destination or the source nodes changed their location (passed a new intersection) the new intersection is added or removed from the route, and the node creates a route update RU and forwards to the opposite destination.

2.1.13.2 RBVT-P

This type floods the network with periodic CPs, with a limited flooding frequency, to capture the road traffic view in real-time. Random numbers of vehicles is selected to generate CP packets independently according to some variable they set in the protocol (e.g. time interval since the last CP received or other parameters). When a CP passes an intersection, it appends its ID in its header to construct the network topology. Then an RU is created from

the road topology constructed by the CP, and it is disseminated in the region covered by that CP.

When a source wants to send a packet to a destination, it computes the shortest path from the routing table using reachable road segments and appends them in the packet header. If the receiver node has a fresh route to the destination, the intermediate node will update the route in the packet header. The CP scheme was inspired from the ad-hoc sensor networks protocols in [32] [33] to eliminate the HELLO packet transmission.

2.1.14 TREBOL (tree-based routing protocol) [34]

This protocol assumes that all the nodes have IPv6 addresses to use for routing. The routing protocol works by building a network tree to forward packets from the nodes to the Internet through the RSU and vice versa. The RSU sends periodical configuration messages (CM) to all nodes in its transmission range. When the node receives these CMs, it changes some fields in the packet with a back-off timer and save it to be transmitted. When the back-off timer expires, the CM will be retransmitted. Nodes that received the CM consider the sender node as a parent node (the next hop node). If a node has a CM to retransmit and receive with the same sequence number, it will drop the CM.

The back-off timer is set by the information attached to the CM by the RSU, e.g. preferred speed and distance and other variables to create a better tree. The tree is built and refreshed on a per data packet basis as part of the data packets forwarding process.

The downstream tree is built when a node sends a packet to the RSU. When a node sends a packet to the RSU, the node retransmitting this packet will be the parent node and it will register the children's IPv6 they own. When a response for a packet comes, the parent node will use the created downstream tree to relay it back to the child node.

2.1.15 IBR (Intersection-Based Routing Protocol) [35]

The purpose of this protocol is to find a path with the shortest packet delay. It uses the carry-and-forward scheme. For that, it needs to avoid vehicles moving in the direction opposite to the packet's transmission direction. In straight roads, the protocol adopts a greedy method, while in intersections the next relay hop depends on the packet's transmission and moving directions.

To avoid packet loss, when the packet reaches an intersection the destination around, a packet duplication will be created, and one of the duplicated packets will be sent to the road segment saved in the packet header, and the other one will be sent to the road segment the destination predicted to go through.

When an intermediate node approaches an intersection, it deals with one of the following four situations:

1. One vehicle in the next road segment. When the intermediate node exchanges its road segment table with neighbors in that intersection and discovers the presence of an existing reachable vehicle in next road segment, the packet will be forwarded to that vehicle.
2. No vehicle at the intersection. In this case, the node carries the packet and passes through the intersection.
3. The packet routing direction is different from the vehicle. In this case, the packet is forwarded to one of the waiting vehicles to hold until the best path is found or a new vehicle with the same direction of the packet route arrives.

4. Upon reaching the last intersection. A duplicated packet is created at the last intersection, one of them is sent to the road segment the destination knows to be there by the packet. In addition, the duplicated packet is sent to the road segment the destination is heading to. The packet time to live is either when it is received by the destination or when it reaches the end of that road segment.

2.2 Advantages and disadvantages

The three classes of routing protocols have their advantages and disadvantages according to the network situation and type of usages. We will pinpoint what we think it will affect the network traffic in the crowded area with heavy load transmission in the following subsections.

2.2.1 Topology based routing

The routing protocols that belong to this class have the advantage of fast transmission and low delay since all routes are available and ready to be used. In the case of the reactive type, there will be some delays to discover the route to be used.

However, in the VANET system, the topology is rapidly changing and the link maintenance is very expensive because of the flooding algorithms used to create and maintain these links, especially in the case of crowded and heavy transmission areas. Furthermore, the use of the traditional method of the TDMA and one network device will make this routing class unsuitable for the heavy transmission areas.

2.2.2 Position based routing

This class of routing is good in the way it does not load the system with the route request flooding and it is more suitable for the VANET systems.

The traditional method for choosing the next hop node, channel selection, network devices used, the TDMA, and especially the collision detection and avoidance (which is not mentioned in most of the proposed researches) need to be designed better to suite the VANET severe environment.

2.2.3 Map based routing

This class has the same advantages as the position base routing class, and in addition, the source chooses the streets the packets should traverse.

However, the source nodes do not know the streets' condition all the time. The lack of knowledge about the streets' condition may lead to packet transmission in disconnected streets (local maximum). The recovery will be either be carrying the packet by the last received node which will lead to high time delay, or retransmit it back to the last intersection to find a better route and that will increase the network traffic especially in crowded and heavy transmission areas. Moreover, if the scheme is made to transmit the state of the streets to the nodes that will introduce more time delay.

For that, we propose a routing protocol using the position based routing class, with new scheme for collision avoidance, and schemes to handle the next hop selection, channel selection, network devices, and a way to remove the TDMA used for the CCH and SCH usage.

2.3 Medium Access Control (MAC) protocols

In this section we will preview some of the MAC protocols used in wireless networks.

2.3.1 Carrier Sense Multiple Access CSMA:

Carrier Sense Multiple Access with Collision Detection CSMA/CD:

In this protocol, when a node has a packet to send, it senses the medium for some time slots. If it was idle, it will transmit the packet. If the medium is busy, it will back-off for a random time and retry in the same manner [36]. This protocol does not work efficiently with the hidden and exposed node problems.

Carrier Sense Multiple Access with Collision Avoidance CSMA/CA:

In this protocol, if a node needs to send a packet, it first senses the medium for some time slots. And if the medium is idle, the sender node will send a Request To Send packet (RTS) as an initiation for the handshake with the destination. If the destination mentioned in the RTS packet receives the RTS successfully, the receiver will sense the medium if there is any transmission or any cause that prevents the sender from sending at the receiver location. If the medium is clear, the receiver will send a Clear To Send (CTS) packet to the sender of the RTS. When the sender node receives the CTS, it will start sending the data packet. When the receiver successfully receives the data packet, it will reply with an Acknowledge packet (ACK) [36], and any node receiving the handshake packets will postponed their transmission until the whole process end. This protocol improves the problem of the hidden nodes problem existing in the CSMA/CD, but it maximizes the exposed node problem.

2.3.2 BTMA-NTS (Busy-Tone Multiple Access – Not-To-Send protocol) [37]

When a node has data to send to a destination, according to this scheme the sender will check if there is no receive Busy-Tone signal (BT_r). If the channel is idle , the sender will transmits an RTS to the destination, and will wait for a period of time even if it did sense the BT_r for its RTS from the destination. If the sender senses that the medium is busy, it will check if the medium is busy with RTS or PRE (preamble packet), it will wait until the transmission is complete before sending its RTS. If the medium isbusy with data packet and there is no BT_r , this means that the sender is an exposed node and will send its RTS. When the destination receives the RTS, it will turn the BT_r on indicating that it is ready to receive and informs its neighbors that it is busy receiving. When the sender wait time ends, and it senses the BT_r , it will start sending its data.

If a node sends an RTS before it receives the BT_r for previously sent RTS to the same receiver, and the receiver receives it, the receiver will send a Not-To-Send packet (NTS) for the second node sent the RTS to prevent it from further transmission. Figure 2.2 shows an example to BTMA-NTS

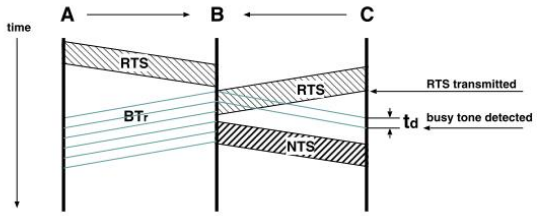


Figure 2.2: Example of BTMA-NTS protocol with A as a sender, B as a receiver, and C as the second RTS sender [37]

2.3.3 A Directional CSMA/CA Protocol for mmWave Wireless PANs [38]

This proposed protocol uses directional antenna with centralized CSMA/CA protocol. The protocol states that one node acts as piconet coordinator (PNC) to coordinate accessing the medium for the other nodes. Furthermore, nodes in their idle mode need to beam-forming with the PNC. When a node wants to transmit data to another node, it starts by sending a Target Request To Send packet (TRTS) to the PNC directly containing all the information about the sender, PNC, and the destination. When the PNC receives the TRTS, it replies with a Target Clear To Send packet (TCTS) in an omnidirectional manner to inform all the neighbors about the transmission. When the sender receives the TCTS, it steers its antenna towards the destination and starts transmitting the data. The destination will respond with ACK packets for the received data packets

2.3.4 MAC protocol for mobile ad hoc networks using directional antennas [39]

In this work, the authors use M number of directional antenna with a transmission angle of $(2\pi/M)$. When a node has a packet to send, it starts by sending an RTS in an omnidirectional manner. When the destination receives the RTS successfully, it replies with a CTS in an omnidirectional manner as well. When the sender receives the CTS successfully, it starts transmitting the data packet to the destination using the antenna facing the destination. In the meantime, the neighbors do not interfere with the transmission with the help of the information held in the RTS/CTS packets

2.3.5 DBTMA/DA (Dual Busy Tone Multiple Access with Directional Antenna)[40]

This protocol uses one channel divided into two sub-channels: one for the data channel and the second sub-channel is for the control channel. In addition to that, two busy-tones transmit and receive (BT_t and BT_r respectively). The protocol uses directional antenna with M number of antenna elements. When a node has a packet to transmit, it senses the medium for any BT_r, so it does not interfere with any ongoing received data. If it is clear, it sends an RTS to the destination. When the RTS is fully received, the destination checks if there is any BT_t, so the sender does not interfere with another sender. If it is clear, the destination will send a CTS directly to the sender and turns on the BT_r. When the CTS is received, the sender starts transmitting the data packet and turns on the BT_t. All the RTS, CTS, data packet, and the busy-tone use the directional antenna and no omnidirectional antennas are used.

2.4 Conclusion

In this chapter we have briefly described some of the previous work about routing protocols, which are classified into three categories: topology based, position based, and map based. We have listed some of their pros and cons, e.g. for the topology-based, the algorithm achieves fast packet forwarding, but it loads the system with RREQ and RREP etc. We have also surveyed the most related work in MAC protocols that can be used in the system. The previous MAC protocols are based on CSMA protocol. They all use RTS/CTS packets. Hence, the field is still open for more improvement to find an adaptive protocol to be deployed in real systems that have the advantage of fast packets forwarding with low control packets loading the system. Moreover, we will introduce a lower overhead MAC protocol that uses directional antennas for VANET systems. The control packets can be either route discovery packets or beacon packets. With that in mind, we have developed our protocol as will be discussed in the coming chapters.

Chapter 3. Receiver as Proxy Routing Protocol

Receiver as Proxy (ROXY) routing protocol for vehicular ad hoc networks is a position based routing with opportunistic next hop selection without the need for beacons. This protocol chooses the best direction to forward packets to its destination using the geographic locations of the sender node, source or intermediate node, and the destination node. For relaying a packet, nodes do not need information about their neighbors. The sender broadcasts the packet in the direction of the destination, and the receivers opportunistically choose one of them to be the next hop node without the need for beacons, which lower the time overhead needed to gather information to send the packet.

VANET networks are allocated seven channels to use, one for emergency and control messages CCH and six channels for service messages SCH. Nodes send and receive messages to neighbor nodes in the area through the CCH and SCH using the same network device. Nodes' net-device use time multiplexing with interval useful to the algorithm in use, usually use 100ms intervals subdivided to 50ms to use the CCH and 50ms for the SCH. The selection of the SCH is done either, locally by listening to the medium and choosing the channel not in use, or centrally by requesting channels from the RSU or central node. We assume ROXY use three devices to separate the SCH from the CCH. Furthermore, two of the three net-devices use directed antennas that spatially separate the transmission area to increase the throughput, and the RSU use directed antenna as well according to its location in the street.

We consider a system consisting of multiple vehicles moving in streets equipped with onboard units. Each vehicle is addressed through a unique IP address. We assume all the vehicles know their own location through GPS, and that they have an updated digital map of the city with full information about the RSUs. This information includes the RSU GPS location, its IP address, and the number of network devices. We mainly consider the case of crowded streets. Vehicles dynamically acquire IP addresses while moving from an area to another. We assume that vehicles can communicate with each other, and with the RSU through multi-hop transmission. Vehicles can send their packets to other vehicles in the network, or to the RSU and vice versa.

In the next subsections, we introduce the basic algorithms used by ROXY to minimize the time required for a packet transmission between a source node and a destination node.

3.1 On-board units

The on-board units are wireless network devices used by the vehicle to send and receive packets to and from the RSUs or other vehicles equipped with on-board units. We assume that all the nodes in the network are equipped with an on-bored units. These devices transmit in a range limited to 300m.

In ROXY, we assume that the on-board units consist of three network devices with different types of antennas according to the channel it uses as shown in Figure 3.1. We will describe them according to the antenna type they use

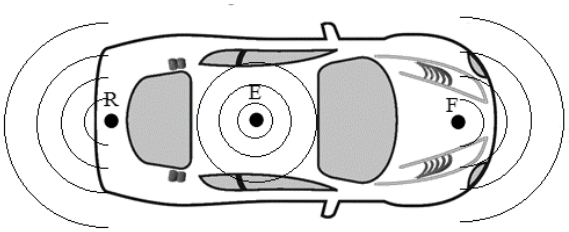


Figure 3.1: Antenna placement on the vehicle. R is the rear directional antenna, F is the directional front antenna, and E is the omnidirectional antenna for the CCH

1. **Omnidirectional antenna:**

We assume that each vehicle is equipped with one network device that uses an omnidirectional antenna for only sending and receiving the emergency and control messages (beacons) through the CCH. The packets that will be sent through this device will carry in its header the IP describing the node itself.

ROXY is beacon-less protocol so there is no need to use them to create a routing table. We do not use the beacons in our test simulation.

2. **Directional antennas:**

We assume that all vehicles are equipped with two network devices that use a 180º directional antennas to send and receive through the same SCH. One network device is directed towards the front side of the vehicle. It is designated to communicate with other nodes (vehicles or RSUs) in front of the vehicle. A second directional antenna exists at the rear, for communicating with the nodes in the back of the vehicle. The packets that will be sent through these two devices carry the same source IP since the IP describes the node not the devices. The three net-devices are separated either by frequency or spatially.

By using these settings, the vehicles can send and receive through the CCH and the SCH simultaneously without using time multiplexing. By using the two directional antennas for the SCH, we can send and receive in two different directions simultaneously on the same channel, which can increase the throughput significantly if used efficiently.

The RSUs are equipped with more than one network device to connect more efficiently with the VANET nodes. The number of network devices of each RSU depends on the street topology. If the RSU is in the middle of a straight street it will be equipped with two sets of network devices, each set contains devices that can use all the seven channels allocated by the IEE 802.11p simultaneously. Each network device points towards one direction of the street: one to the right and the other to the left as shown in Figure 3.2. This is done for spatially separating the transmission in the street, and to remove the interference at the RSU done by the vehicles transmissions on the right side from those vehicles on the left.

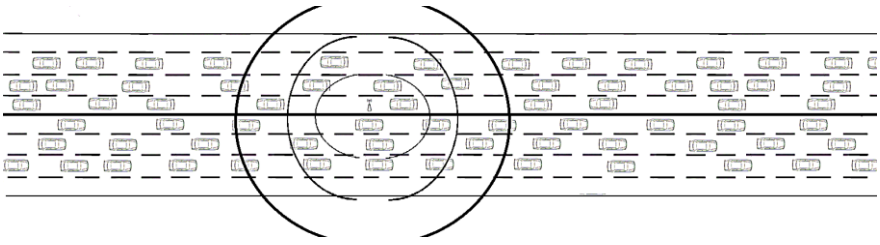


Figure 3.2: RSU's network devices transmission shape in straight streets

In case the RSU was at an intersection, it will be equipped with network devices sets equal to the number of streets the intersection connects. Since the intersections in crowded situations have the most heavy network traffic, it is suitable to use a separate network device for each street to separate the communications, lessen the collision caused by hidden nodes, and to increase the throughput.

3.2 Channels

We use a centralized approach to allocate SCHs for the nodes in the system. This is done indirectly through the handshaking process with the RSU. When sending the IP Address request packet, the RSU sends an index field inside the IP Address reply packet. This index is created in a way to be unique for the nodes in the same area. The RSUs use a range of integer numbers from 0 to 1023 to be sent as indices. When a node receives the new index, it uses equation (3.1) to choose one of the six SCH available (leaving the zero channel as CCH). This gives fairness in distributing the channels to the nodes. Since the range of the index pool is large, no two nodes will have the same index in their transmission range.

$$channel = (Index \ \% \ 6) + 1 \tag{3.1}$$

3.3 IP Acquiring and Hand-over

VANET vehicles go from RSU to another RSU transmission range during each vehicle's journey. To join the new RSU network, the first step the vehicle takes is to request for a new IP address belonging to the upcoming RSU network. ROXY protocol uses the following setups and steps to guarantee a fast and efficient assignment of the IP addresses to different nodes:

1. Each RSU is assigned a unique set of IP addresses that will be distributed among all nodes associated with it,
2. Each node is assigned the IP address by the RSU associated with it,
3. When a vehicle is in the midway between two RSUs, it sends an IP request to the target RSU through multi-hop transmission,
4. When the RSU receives the IP request packet, it gets a new IP address from its DHCP pool to assign it to the approaching vehicle. It also assigns an index that starts from 0 to 1023. The pointer to the current index is reset to 0 when it reaches its maximum to ensure no two nodes in the same area having the same index,
5. The old IP address that was assigned to the node will be returned to its old RSU,
6. The new RSU communicates with the old RSU to direct any down streaming packets going to this node to the new RSU,
7. The new IP and index are sent from the new RSU through a packet to the incoming node.

Step (7) comes after step (6) to ensure that the downstream packets are redirected to the new RSU to avoid any packet loss in the down-stream. This is similar to handover philosophy of "make before break".

3.4 Collision Avoidance

The major challenge in random access wireless communication is collisions. Finding an algorithm to prevent or reduce them is necessary, especially in VANET environment to reduce human casualties in case of emergencies. If a collision occurs during packet transmission, it will cause time overhead to the transmitting node and any node was waiting the medium to be idle to transmit. If the collisions kept occurring, it might cause failure to time critical applications, or even stop the whole routing system if not handled properly. For that, a good collision avoidance protocol is needed.

Many efforts were done in the field of collision detection and avoidance. Two major algorithms exists: (1) Carrier sense multiple access with collision detection (CSMA/CD) (2) Carrier sense multiple access with collision avoidance (CSMA/CA) which show good result in MANET and VANET unicast. It can help avoid some of the collisions but still it cannot solve the *hidden node* problem as shown in Figure 3.3.

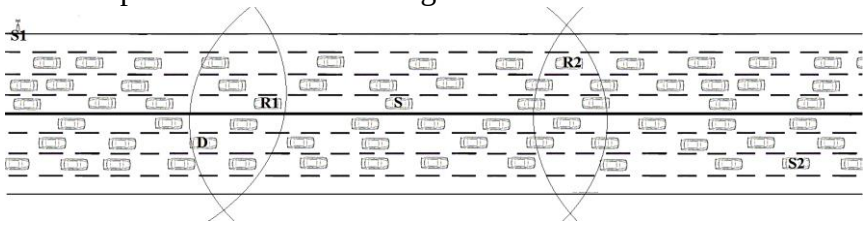


Figure 3.3: CSMA/CA hidden node problem. S wants to send to D since the channel is idle, S started sending RTS to D causing collision to R1 and R2 currently receiving from S1 and S2

In this thesis, we propose new collision avoidance scheme called *Denial Of Transmission with Collision Prevention* (DOT/CP). In most cases, the position of the packet's destination is in the front or in the rear of the sending node. We came up with DOT/CP by taking advantage of the two directed antennas and their ability to send two different packets simultaneously. The algorithm works to prevent collisions from happening by denying the nodes' right to send in a specific direction while another node is sending in it.

DOT/CP algorithm works as follow:

1. Sense the channel, and check if it is idle,
2. Send the packet from the network device in the direction of the destination. Note that only the direction of the final destination is needed not the location of the intermediate nodes in the multi-hop network,
3. In the same time, send a (NAS) packet in the opposite direction using the other network device and directed antenna, carrying the time needed to send the packet.
4. When a network device in any intermediate node receives the packet, it will set its *Allow To Send* flag (ATS) to "false" (this means that the node cannot send using that network device). It will also set "time to free the device", which is the time required to change ATS to "true", equals to the time needed to send the NAS packet sent in step (3). This will prevent collisions to the NAS packet to ensure successful reception by the nodes.
5. When a NAS packet is received by nodes, their ATS is set to "false" with "time to free the device" equals to the time needed to send the data packet successfully attached in the NAS packet.

DOT/CP has less time overhead than CSMA/CA. Since CSMA/CA needs to go through four stages (1) Sense the channel, and check if the channel is idle, (2) Send Request to Send

(RTS) to the target node, (3) Wait for receiving a Clear to Send (CTS). In case of failure, go to (2). Otherwise, continue, (4) Send the packet. While DOT/CP go only through two stages (1) Sense the channel, and check if the channel is idle, (2) send the packet and the NAS packet. Even in case the RTS or the CTS did not cause collision to a hidden node, the DOT/CP guarantee to give better result in case of time overhead for the VANET network while using our onboard unit setup.

This method lowers the overhead needed to send the packet as it does not use any CTS/RTS packets. It also avoids the hidden node problem. We can explain that using the scenario in Figure 3.3. In this scenario, S1 is sending to R1, which received the packets through the front network device. S2 is also sending to R2, which received the packets through the rear net-device. If S is going to send a data packet to D, it will start by sensing the medium. Because the coverage area of S1 and S2 does not reach S, it will report an empty channel. S is going to send the packet through its front net-device. In the meantime, it will send a NAS packet through the rear device. R2 will receive the NAS packet, and will not send any packets through its front device until time to free the device expires. Note that the RSU does not follow this DOT/CP protocol. Simulations show that by sending the packets all the time by the RSU the average delay is reduced.

3.5 Packet types and structures

ROXY protocol uses different types of packets in requesting IP addresses and applications, IP reply, application reply, and packets used in the proposed collision prevention algorithm. We will describe these packets in details in the coming subsections.

3.5.1 Not Allowed to Send (NAS) packet

The purpose of NAS packet is to prevent the receiver nodes from transmitting for an amount of time specified in the NAS packet header. This packet type plays a major role in the proposed collision avoidance algorithm. When a node wants to send a packet, it uses NAS packet to inform the nodes that received this packet that they are not allowed to send until the sender node finishes sending his packet. NAS packet structure consists of a header only. It does not have a body, as shown in Table 3.1.

Packet Type is an integer number that represents the NAS packet type to differentiate NAS packets from other packet types. *Time* is a real number set by the sender that represents the period of time the receivers' network devices that received the NAS packet must stop transmitting. This will help to prevent interference with the routing process and to reduce collisions.

Table 3.1: NAS packet structure

Packet Type
Time

We structured the NAS packet in small size (64bit) so it will not hold the medium for long period. Since the goal of the algorithm is to lower the time overhead, and since the packet does not need to convey any information about the sender or the details of the routing process

except of the time needed to prevent the nodes from sending, we omitted the sender IP or any other unnecessary information.

3.5.2 Acknowledge (ACK) packet

In ROXY protocol, ACK packet is an important type of packets for the next hop selection process, and it holds the highest priority in the packet transmission order. Nodes create ACK packet for two reasons. The first reason is to inform the sender node, either vehicle or RSU, that its packet has been fully received and it can either remove the packet from the send-buffer or move it to the temp-buffer for later retransmission in case the reply didn't come before the max waiting time expires, which indicates a failure attempt. The second reason is to inform the other nodes that the ACK packet sender has chosen itself as the next hop sender, namely "the source node proxy", for the packet in concern. In this case, all other candidates in the selection process can drop the packet.

Table 3.2: ACK packet structure

Packet Type	PTAIF
SIA	
DIA	
PI	
FNIA	
RNIA	

Table 3.2 shows the detailed structure of the ACK packet. *Packet Type* is an integer number that represents the ACK packet type. *PTAIF* (Packet Type the ACK Issued For) is an integer number that represents the type of the packet the node sending the ACK packet for. *SIA* (Source IP Address) is the IP of the node created the packet this ACK issued for. *DIA* (Destination IP Address) the IP address of the destination saved in the packet's header that the ACK packet responds to. *PI* (Packet Index) is the index of the packet the ACK is issued-for. *FNIA* (FROM Node IP Address) is the IP address of the node sending the packet that the ACK is issued for. *RNIA* (Receiver Node IP Address) is the IP of the node currently sending the ACK packet.

ACK packet length is 192 bit in size. It consists of a header with no body. We use the main characteristics that can distinguish packets from each other, and use them to lower the time it needs for transmission.

3.5.3 IP Request packets

This packet type is created from nodes that need IP addresses, or when they reach a position, where they need to switch from one RSU to another RSU to reduce the hop count. Vehicles along their traveling path go through different RSUs transmission ranges. Hence, they need to start their conversation with the incoming RSU by first requesting a new IP from that RSU to join its network. After that, the node can start sending and receiving from and to that RSU as part of its network.

The structure of the IP request packet is shown in Table 3.3. The size of the IP request packet is 256bit and it consists of only a header with no body because there is no need to convey any data to the RSU the node heading. The description of the header parameters is shown in Table 3.4.

The flag parameter contains (1) *Routing is needed* field for informing the receivers if the packet destination is out of the transmission range and needs routing or not, (2) *ACK needed* to inform the receivers that ACK packet is to be expected from the chosen node (destination or next hop node).

Table 3.3: IP request packet structure

Packet Type	Flags	CSU	Hop Count
API			
SIA			
SP			
DIA			
DP			
CSIA			
CSP			

Table 3.4: IP request header parameters description

Parameters	Description
Packet Type	Integer number representing the IP request packet type
Flags	flags used in the routing process
CSU	Integer number that represents the channel sender used when sending the IP request
Hop Count	Hop count to the current state
PI	Packet Index
SIA	Source node IP Address (the IP of the node sending the request
SP	Source node Position when this packet was sent
DIA	Destination (RSU) IP Address
DP	Destination (RSU) Position known by the source
CSIA	Current sender IP Address (source or intermediate node)
CSP	Current sender position (source or intermediate node)

3.5.4 IP reply packets

As a reply for the IP request, the RSU creates and sends back the IP reply packet with the information needed. The IP reply packet contains the new IP received from the DHCP associated with the RSU, and a unique index designated by the RSU to the node. This index ranges from 0 to 1023. The pointer to the current index keeps rounding every time it reaches the maximum range. The index is the RSU way to inform the nodes which channel they will use and, how much back-off delay they should use when a delay is needed to send a packet in

case of collisions, or when the medium was busy with another node to minimize the collisions.

Table 3.5 describes the structure the IP reply packet used, which is 320bit in size. It consists of a header only with no body. It contains the same parameters as the IP request except for the new IP and index fields that the RSU assigned to the incoming node. As for the source, this time it will be the RSU, and the destination is the node requested the IP. The destination position field is filled with the IP request received before.

Table 3.5: IP reply packet structure

Packet Type	Flags	CSU	Hop Count
API			
SIA			
SP			
DIA			
DP			
CSIA			
CSP			
NIA			
NIx			

Table 3.6: IP reply parameters description

Parameters	**Description**
Packet Type	Integer number representing the IP request packet type
Flags	flags used in the routing process
CSU	Integer number representing the channel sender used when sending the IP request
Hop Count	Hop count to the current state
PI	Packet index
SIA	Source node (RSU) IP Address
SP	Source node (RSU) position
DIA	Destination (vehicle) IP Address
DP	Destination (vehicle) position known from the IP request packet
CSIA	Current sender IP Address (RSU or intermediate node)
CSP	Current sender position (RSU or intermediate node)
NIA	New IP Address from the RSU DHCP
NIx	New Index from the RSU to the node

3.5.5 Application Request packet

In the scope of this thesis, the application represents any service the node can ask for. This ranges from local information through the VANET system to entertainment from the Internet. During the simulations at the end of the thesis, we assumed that any node receives a new IP would create and send an application request to the RSU the IP reply came from. In addition, we assumed that the application request packet sent to a RSU is requesting for ten application reply packets.

The structure for the application request is shown in Table 3.7. The header size is 256 bits, and a body of 200kbit carrying information to the RSU about the service needed to be sent back to the vehicle. The application packet's parameters description is shown in Table 3.8.

Table 3.7: Application packet structure

Packet Type	Flags	CSU	Hop Count
API			
SIA			
SP			
DIA			
DP			
CSIA			
CSP			
DATA			

Table 3.8: Application packet parameter description

Parameters	Description
Packet Type	Integer number representing the application request packet type
Flags	flags used in the routing process
CSU	Integer number representing the channel which the sender used when it sent the application request
Hop Count	Hop count to the current state
PI	Packet index
SIA	Source node IP Address
SP	Source node position when this packet was sent
DIA	Destination (RSU) IP Address
DP	Destination (RSU) position known by the source
CSIA	Current sender IP Address (source or intermediate node)
CSP	Current sender position (source or intermediate node)
DATA	Is 200kbit worth of data

3.5.6 Application reply packet

The RSU respond to the Application request packet by creating a bundle of ten application reply packets (represent anything like picture, email etc. of total size 2Mbit to test the protocol), and saving them in a buffer to send them to the node requesting the service. The RSU uses Round Robin to select a packet from the different buffers responding to different nodes application requests.

The structure of the packet's header is the same as the application request in Table 3.7 and Table 3.8 and size of the packet is the same as well 200kbit, but the body of the packet contains the information the node requested from the RSU.

3.5.7 Packets priorities

The packets in ROXY are prioritized in order to send the time tolerant packets after the time critical ones. The packets are served according to their priorities, not on a first come first served basis. For example, ACK packets should have the highest priority in the transmission buffer in order to speed up the next hop selection process and decrease the time overhead. Table 3.9 shows the packets priorities used. These priorities are for the packets sent only by the directional antennas, and not the emergency or control messages. Note that NAS packets does not need priority value since it is created to be sent immediately.

Table 3.9: Packets types and their priorities

Packet type	priority
Application	0
IP Request & IP Reply	1
ACK	2

3.6 ROXY Routing Protocol

ROXY's objective is to deliver the packets on node-to-node level as fast as possible with as low collisions as possible. To achieve such goal, the protocol must consider more than one issue to reduce the collisions, we use the DOT/CP scheme previously introduced, and manage the transmission efficiently. ROXY with the MAC layer proposed, work with the following issues to enhance the network system

1. Reducing the time overhead by reducing the hop count. This can be achieved by relaying the packet to the closest one hop node to the destination,
2. Reducing the time lost in choosing the next hop node centrally, and letting the receivers opportunistically select the closest node to the destination,
3. Increasing the throughput by using multiple network devices to separate CCH from the SCH and using directed antennas to separate the transmission area spatially,
4. Removing the dependency of the beacon messages in the routing process, which improves the emergency messages transmission efficiency.

The main advantage of ROXY is that it does not need to create and maintain a path for the packets. Creating a path needs to flood the network with path discovery messages, which will affect the overall throughput greatly, especially in the crowded network traffic scenario. For

that, we proposed this algorithm to cope with the heavy loaded network by minimizing the packet transmission, delay overhead, and collision to a minimum.

ROXY sending nodes do not need to know the IPs or locations of the neighbors to route packets. Creating neighbor table to use for choosing the next hop node will need periodic beacon messages to collect the one hop neighbor information. Beacons share the same frequency channel used to send emergency messages CCH, so depending on beacons will mean that it will compete with the emergency messages that might lead to human casualties. For removing the dependency on beacons, ROXY is a distributed algorithm, i.e. the next hop node selection is not central determined by the source node but the receiver nodes elect the next hop node.

ROXY algorithm is described by the node's sending or receiving state. The coming subsection will describe in details the algorithm steps with these different nodes states.

3.6.1 Source node state

As a source node, the first step is to create the packet and fill its header with the correct information to route it successfully. The header of the packets that have a single destination must contain a packet type and ID to prevent duplication. It will also carry the IP address and position of the destination. In vehicles case, the destination is mostly one of the RSUs. The position and IP address for the destination RSU can be obtained from the digital map that the vehicles have. In the case when the RSU is the source, the IP address and position of the destination are found in the request packet's header.

After the packet creation, the node goes through several preparations and checks to start sending. First, the node needs to figure out which direction the packet needs to be sent through. After the direction have been calculated, the node saves the packet to the buffer associated with the net-device corresponding to that direction. The node then needs to go through different checks to start the sending process. Before the packet sending, the node creates a NAS packet as part of the collision prevention process to be sent with the packet in the same instant but from different devices. Then, the packet is sent in the direction of the destination and the NAS packet is sent in the opposite direction to inform the nodes to refrain from sending until the sending process is finished to prevent any collision from occurring. Figure 3.4 shows the source node process.

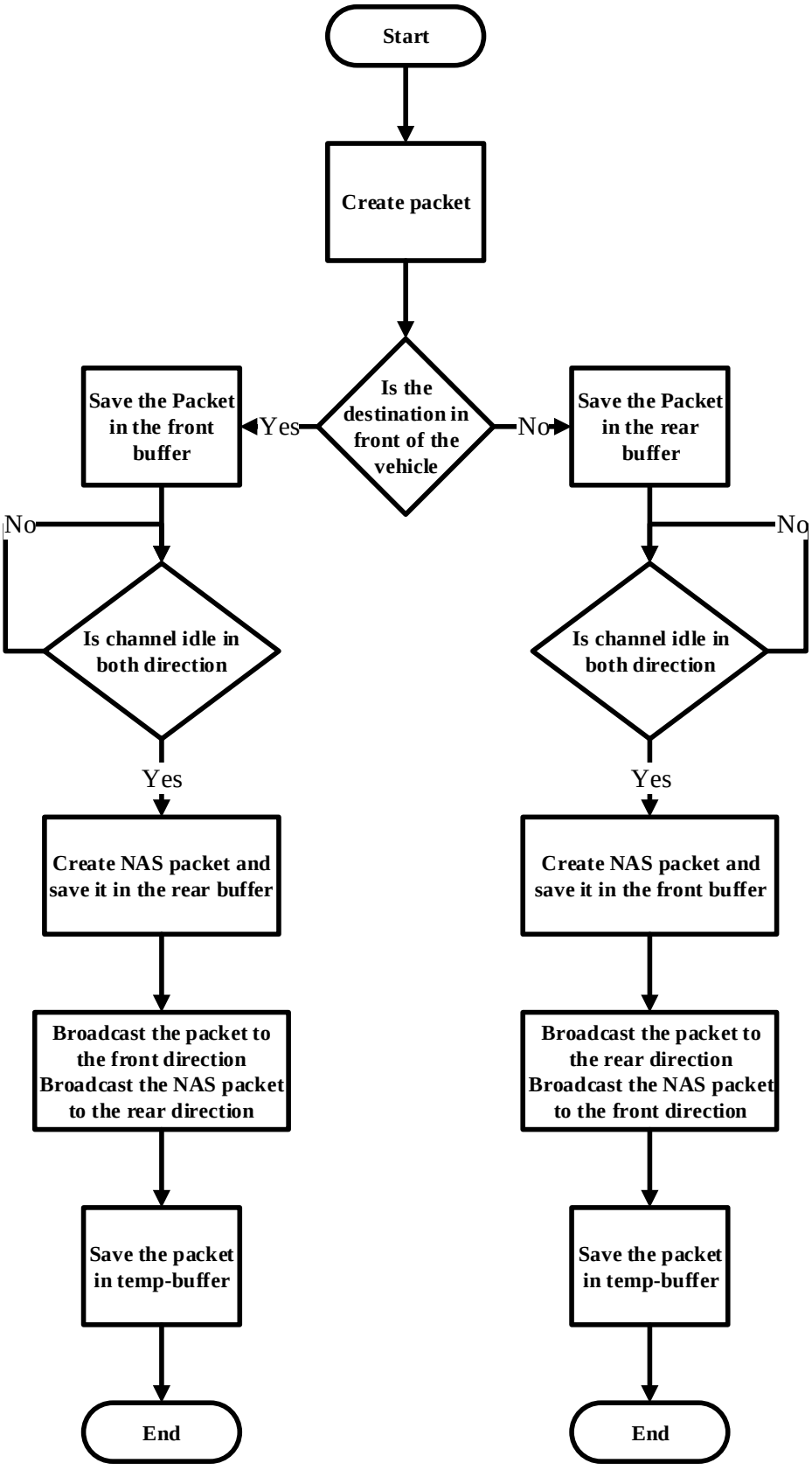


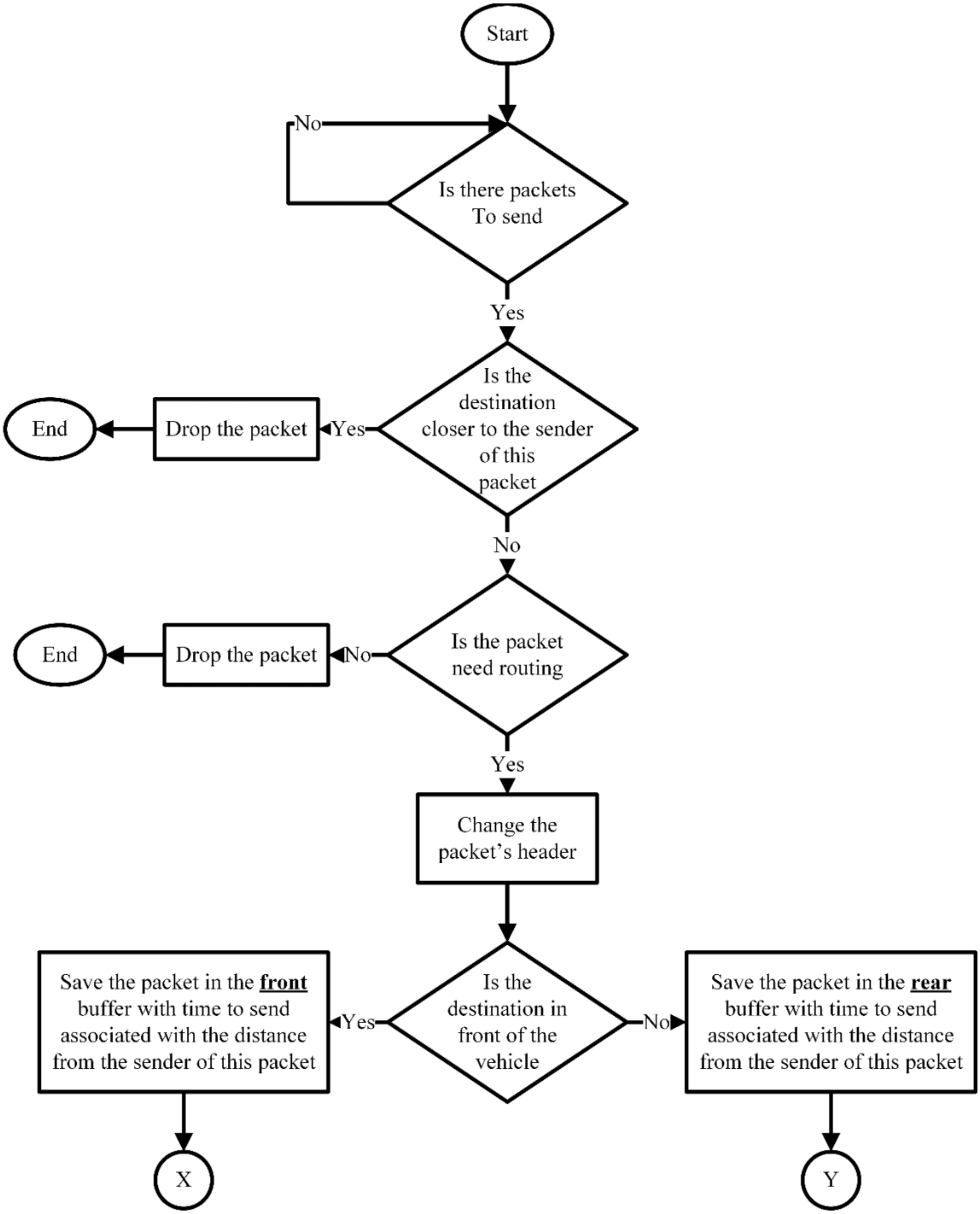
Figure 3.4: Souce node transmission algorithm

Now we will list the detailed steps of the source node algorithm the flow chart summarized:

1. Create a packet to be sent,
2. Check the position of the destination relative to the source in order to save the packet according to the transmission direction in the appropriate buffer. If the destination is on front of the source, then save it in the buffer associated with the front net-device, and vice versa,
3. Tag the packet with the appropriate priority level, as shown in Table 3.9 The buffers contains two fields for each storing unit, one stores the packet, and the second field stores the time this packet should be sent in. the second field is where the protocol tag the "time to send the packet" in. The packets tagged with "time to send the packet". Note that, the buffer does not work in a first come first served basis. The device first checks the "time to send the packet" field, and among the packets that should be sent at the current time instant only the packet with higher priority will be sent first,
4. Check if there is a packet in buffers (front and rear) to send. If false go to 4 if true go to 5,
5. Get the packet with the highest priority to be sent from the packets that has their "time to send the packet" value equals to or less than the current time,
6. Check if the channel is idle in both directions to send the packet and the NAS packet. If false go to 5, if true go to 7,
7. Create NAS packet carrying the "time to free the device" required to transmit the packet,
8. Send the packet through the device in the direction of the destination. In the same instant, send the NAS packet through the device in the opposite direction,
9. Save a copy of the packet in the temp-buffer to resend later in case no ACK received. The time for resending the packet is calculated through the following equation

$$T_{RS} = T_S + T_A + \big(10\big((In \,\%\, 30) + 1\big) + 1000\big) \tag{3.2}$$

Where T_{RS} retransmission time, T_S the time to fully send the packet, T_A time to create and send the ACK packet by the receivers, and In is the node index received with the IP reply packet from the RSU to lower the collision. The last term guarantees that if there were two nodes trying to transmit at the same instant and caused a collision they will retransmit at two different time instants to avoid the collision reoccurrence.


Start
No
Is there packets To send
Yes
Is the destination closer to the sender of this packet
Yes
Drop the packet
End
No
Is the packet need routing
No
Drop the packet
End
Yes
Change the packet's header
Is the destination in front of the vehicle
Yes
Save the packet in the front buffer with time to send associated with the distance from the sender of this packet
X
No
Save the packet in the rear buffer with time to send associated with the distance from the sender of this packet
Y

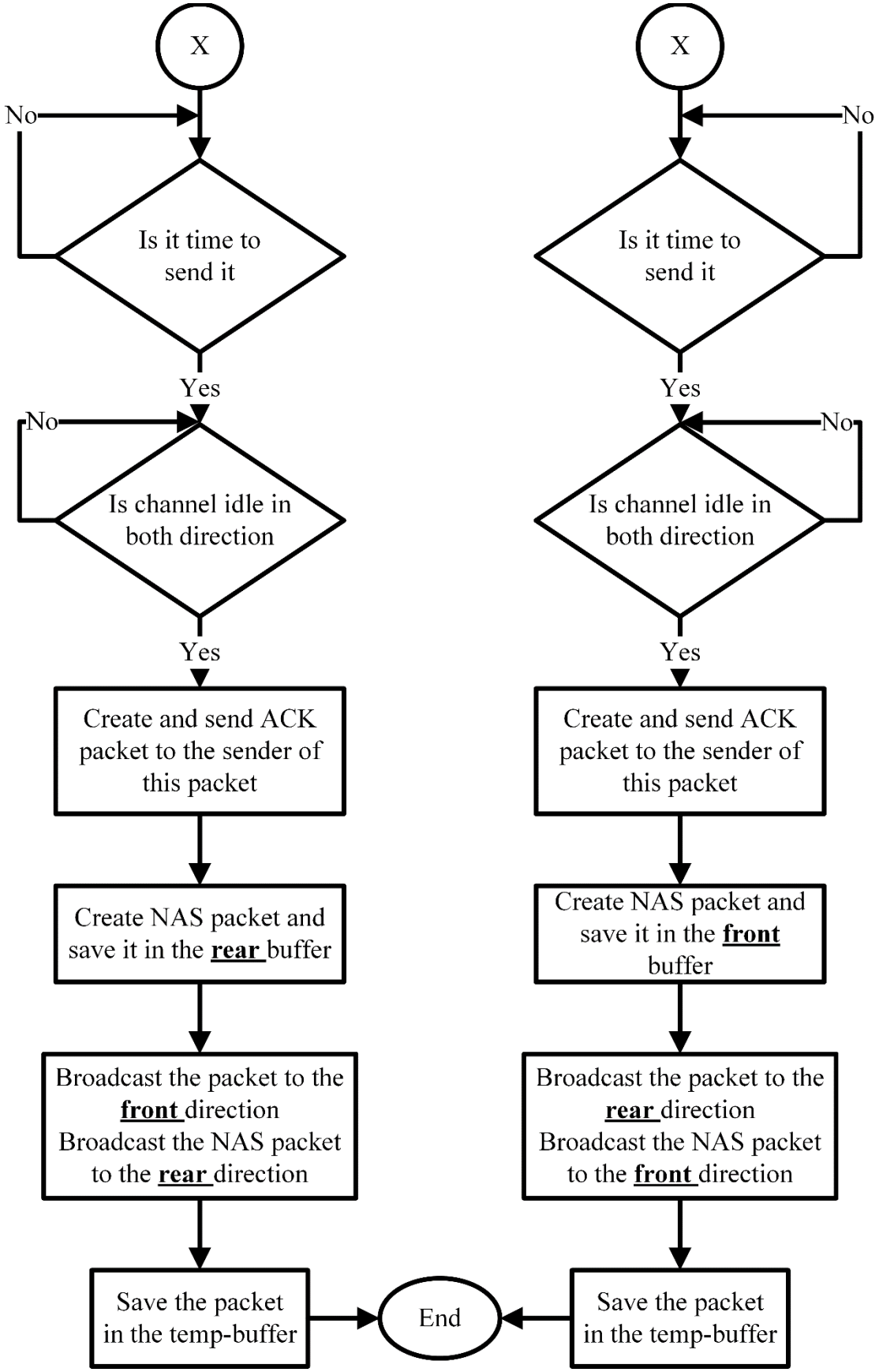


Figure 3.5: Intermediate node transmission algorithm

3.6.2 Intermediate node state

The intermediate node transmits in a similar manner as the source node Figure 3.5. After the intermediate node receives the packet and sends an ACK packet to the sender node (source or another intermediate node), it needs to choose which device to use to have the best results in sending the packet. If there was any failure in the transmission, it will make sure to retransmit it until the destination replies or another node acts as the new next hop node. This intermediate node assured the sender node that it would be its proxy to send this packet through the agreement done by the ACK packet sent to the source.

The intermediate node needs to prepare the packet before saving it in the buffer to send it later. The nodes need to change the information in the header as preparation for the next transmission process by changing the IP and position of the *current sender node* fields to the IP and position of this node since it will be the current sender to the receivers also the *hop count* needs to be incremented. The flags need to be handled with care since it can terminate the whole routing process by filling the field *routing needed* and *ACK needed* field with a wrong value since it can lead to infinite retransmission by the sender and packet dropping by the receiver.

The detailed process the intermediate node uses to transmit a received packet is as follow:

1. Check if the destination is closer than the intermediate node to the sender. If true, then drop the packet, if false go to 2. Note that the destination location is tagged in the packet header,
2. Check the routing flag in the packet. If it does not need routing, drop the packet, if it needs routing go to 3,
3. Change the *current sender IP* and *current sender position* fields in the packet's header with the intermediate node IP and position, and increase the *hop count* by one,
4. Save the packet in the appropriate buffer (front or rear buffer) according to the destination direction,
5. Set the priority and "time to send packet" tag. The "time to send packet" value is set proportionally to the distance between the intermediate and sender node using the following equation

$$T_S = T_C + (Tx_R - D_{SR}) \times 2 + (In \,\%\, 20) \tag{3.3}$$

Where T_S transmission time, T_C current time, Tx_R is the transmission range, D_{SR} distance between the sender (source or another intermediate node) and the receiving node, and *In* is the node index.

6. Create an ACK packet to be sent to the sender with the same delay the packet saved with. After receiving the packet to be retransmitted, the receiver must send an ACK to the sender node to inform it that the packet found a new proxy to send it to the destination,

Since the "time to send packet" proportional to the distance between the sender and the receiver this will guarantee that the farthest node from the sender will send the ACK first. Hence, this will reduce the overall number of hops between the source and destination

7. Check if there is a packet to be sent in this instance, if true go to 8, else stay in 7,
8. Broadcast the ACK in the direction of the sender, and broadcast NAS packet in the opposite direction informing the nodes to wait until the ACK packet is fully sent,
9. For broadcasting the packet in the direction of the destination, we use the same method as the "source node protocol" starting from step 4 to the end.

3.6.3 Destination node state

The end of a routing process is at the destination node, which does not have computational load. The destination nodes algorithm is about sending the ACK packet as fast as possible to confirm the reception of the packet by its destination. The process of the destination node's state protocol is listed below:

1. Check if the node is the final destination; if true continue
2. Consume the packet ,
3. Create ACK packet and save it in buffer to send with delay zero.

3.6.4 ACK packet receiving state

The reception confirmation of any packet is done by sending an ACK packet corresponding to the received packet to the sender. The concept in ROXY protocol is to move the source virtually to be one hop distance from the destination, which is achieved by making every relay node as a proxy for the source in the routing transmission path. The relay nodes guarantee that they will successfully send the packet closer to the destination through the best route possible, and it will carry the packet and retransmit it until a new intermediate node is found or the node reaches the destination (in case of empty street). If the intermediate node receives a retransmission of the same packet it will update the source node position and keep searching for a route for it.

If a node is receiving an ACK packet, it will do the following:

1. Check if the node is the final destination of this ACK. If false, go to 2. If true, one of the following actions will be taken:

A. If the node is not the original creator of the packet, then drop the packet associated with the received ACK.

B. If the node is the "source" of the packet, then check the type of the packet. If it is a request packet, then change the scheduled retransmission time for this packet to guarantee reaching its final destination. If it is a reply packet, then drop the packet from the temp-buffer.

2. Check if the node has a copy of the packet associated with the received ACK, and drop it from the buffer.

Step 2 is the scheme the protocol takes advantage of to make the next hop selection decision locally made by the receivers. In intermediate node state algorithm, the sending time of the ACK packet is decided through the inverse of the distance from the intermediate node to the sender node. Therefore, the farthest intermediate node from the sender node will be the one that sends the ACK packet faster than the rest of the other candidates. When the receiver nodes receive the ACK packet, they will drop the received packet associated with this sent ACK packet, and the ACK packet sender has chosen itself as the next hop node without the sender node interference and with minimal packet sending.

3.7 ROXY protocol advantages and disadvantages

Advantages:

- ROXY nodes do not need to have the full map of the vehicles instead it needs only a map for the fixed locations of the RSUs. Hence, there is no need for beacon messages carrying information about all the nodes to all other nodes as required by many other protocols. With this improvement, we can reduce the time overhead lost collecting fresh information to calculate the best next hop relay node since it is done locally. Furthermore, since the routing is beacon independent, there will be no need for massive sending for the beacons that will affect the performance of the emergency messages.
- There are no route discovery procedures in ROXY. Hence, this reduces the end-to-end delay. The major concern is to reduce the time overhead. Hence, we choose the position based routing since it removes the time wasted in finding a list of nodes to relay the packets through. Also because of the rapid changes in topology caused by vehicle's speed and direction variations, the transmission's paths are not stable and suffer from disconnections causing reduced performance. Furthermore, in most of the cases, destinations of the vehicles' packets will be the RSUs. This will cause a very high load on the chosen vehicles in the discovered path while the other nodes will be idle. In ROXY, each vehicle decides for itself if it is good to be in the selection process.
- The NAS message reduces the collision probability because it prevents other nodes from sending, and avoids the hidden nodes problem while there is a node transmitting. This is one of the main improvements in ROXY since it reduces the time wasted in preparing the medium to send a packet by using the minimum number of packets needed. It was proven by the simulation the superiority of the DOT/CP over the CSMA/CD by preventing the node from interfering with the sending. In addition, using the directed antenna combined with DOT/CP solved the hidden node problem.
- ROXY reduces the overall number of hops between source and destination by setting the time to send the ACK message according to equation (3.3). This will also reduce the end-to-end delay required to send the packet.
- Distributed next hop selection and using only one packet and an ACK. This contributes in reducing the delay.

Disadvantages:

- The extra load caused by the map updating. However, the update frequency can be low
- Extra cost for using more network devices,
- The retransmission in case of collision will cost more time than in CSMA/CA and equal or less than the delay if used CSMA/CD. However, the MAC used in the proposed protocol causes rare collisions.

3.8 Conclusion

This chapter describes the protocol we developed with all its features, covering the physical, MAC, and network layers. In the physical layer, we used multi-network devices, two of them use directed antenna. We also covered the channels distribution, IP addresses acquiring and hand-over, collision avoidance, and data packets types and structures. We have also described the developed routing protocol used by different nodes e.g. source nodes, intermediate nodes, destination nodes and the scenario of the ACK packet reception. At the end of the chapter, we listed the advantages and disadvantages of ROXY protocol.

Chapter 4. Simulators

In this chapter, we will discuss the tools used in testing the protocol. To test the protocol, two simulators were needed: SUMO [41] to simulate and generate the model of vehicles in an urban environment, and a network simulator we developed that will be discussed in details in further sections. In addition, two tools were used: one to transform SUMO output files to a form that the network simulator can handle, and the other to extract the results from the trace files created by the network simulator.

For the tools we created, we used Java Object Oriented language and Eclipse IDE to write, test, debug and run the simulator.

4.1 SUMO (Simulation of Urban Mobility)

SUMO is an open source microscopic road traffic simulator to manually create, automatically create, or export the road map and to generate vehicles to use in the mobility simulation. The roads details, from the number of lanes, traffic lights positions, streets' movement direction, road expansion or contraction, and many other options can be optimized to make the roads more realistic. SUMO can also customize different details of vehicles, and can create groups of vehicles to be used in mobility scenarios.

After creating the map and vehicles, the last setup we have done was to describe the movement of the vehicles along the created streets. We used "flow" type of movement. In this type, we create an xml file, each line in the xml file represents a mobility flow and it contains the flow id, start and end of movement, the maximum speed the vehicles in this flow should use, and the number of vehicles in it.

When the simulation is over, we get an xml trace file containing the mobility of each vehicle according to time steps specified in the run command option.

4.2 Transforming Tool

After completing work with SUMO tool, we export the xml trace file to our transforming tool to transform the single trace file to multiple text files, each file describes the mobility of a single vehicle. We choose the text file format to have an easy and fast readable file, to use in testing along with the network simulator trace and debug files. In addition, if there is any need to modify the mobility files of a single vehicle in the future, e.g. removing a vehicle, shortening its traveling path or any other change, it can be easily done.

```
1  package MobilityCreater;
2
3  import java.io.IOException;
8
9  public class MobTest {
10     public static void main (String[] args) throws IOException,
11                       SAXException, ParserConfigurationException{
12         Mobility mob = new Mobility("exout1440.xml");
13         doNothing(mob);
14     }
15
16     private static void doNothing(Mobility mob) {
17
18
19     }
20 }
```

Figure 4.1: Mobility transforming tool's interface

The user interface is simple and limited to only inserting the xml file name. Figure 4.1 shows the interface code in Eclipse IDE window. The part "exout1440.xml" is the file we generated from SUMO and Figure 4.2 shows a snapshot of the files generated by our transforming tool. Each one of these files represents a vehicle's mobility, direction, and position as a function of to time.

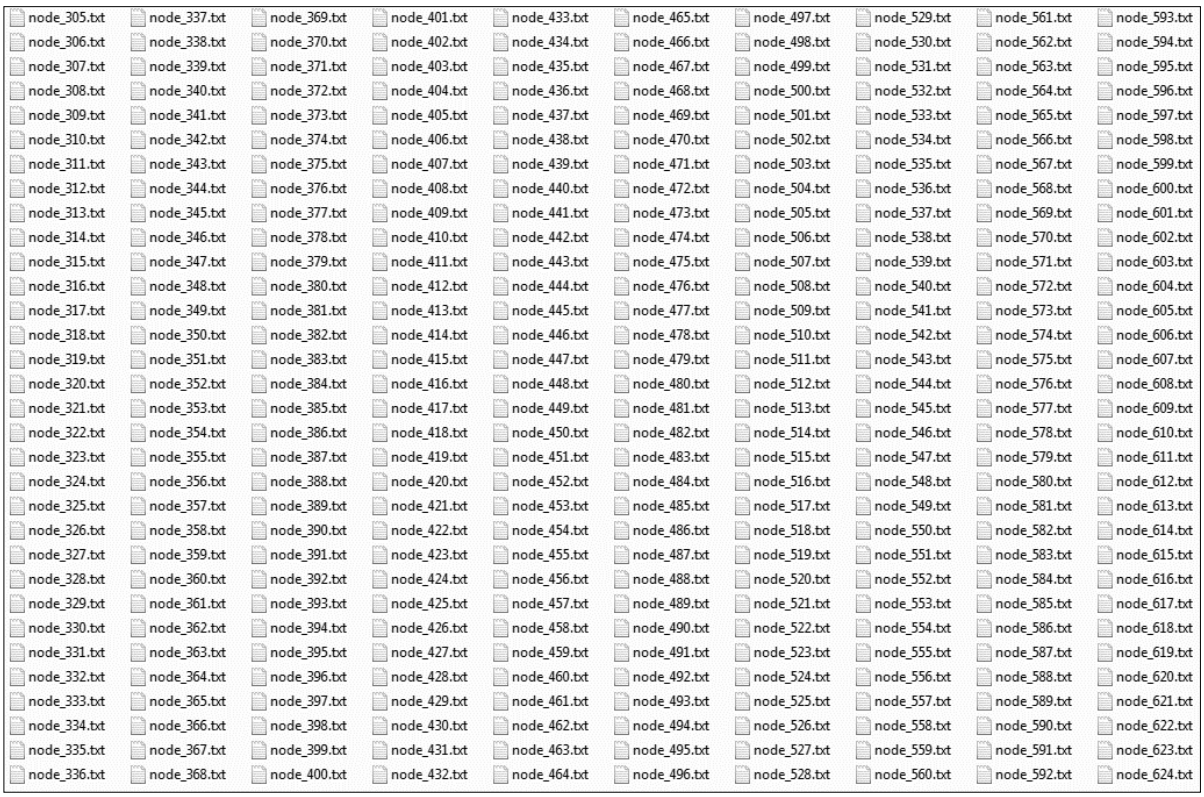


Figure 4.2: Part of the file generated by the transforming tool

4.3 Network Simulator

We developed a new network simulator using Java Object Oriented language using Eclipse IDE as the working environment. The simulator contains all the features needed to test and compare ROXY with BRAVE protocols e.g. directed Wi-Fi antenna, DHCP, the ability to obtain the IP address and change it in the run time, detailed debug system, action log system, etc. The following sections will describe the main methods and processes used in the simulator.

4.3.1 Simulator interface

The interface of the simulator in Figure 4.3 is used to insert the parameters for the current run and the log type. The first four methods are used to call invoking the LOG class to specify the way the log file is needed. There are two types of logging system. *Debug log*: this type of log system saves in a text file a log about what method is running and when it leaves this method in addition to all the actions done by the nodes. *Action log*: this type of log system saves the action done by the nodes and the RSU. The Action log system is useful to trace all transmissions, collisions, or errors carried by the nodes to help analyses and debug. The Debug system do the same as the Action system and in addition to that, it is useful to trace any bugs in the simulator.

```
1 public class RoxySimulator{
2
3⊖    public static void main (String[] args){
4         LOG.setDebugState(false);
5         LOG.setDebugStepByStepState(false);
6         LOG.setActionLogState(false);
7         LOG.setActionLogStepByStepState(false);
8         Simulator.setBeaconState(false);
9         Simulator.setTimeAddingCheckNeeded(true);
10        Simulator.startTime(1460.4);
11        Simulator.endTime(3513.1);
12        Simulator.NumberOfNodes(1440);
13        Simulator sim = new Simulator();
14        SimulatorApplicationContainer app = new SimulatorApplicationContainer(1, 1440, 1, 50.4, 10);
15        Simulator.getApplicationContainer().add(app);
16
17        sim.start();
18        LOG.setDebugState(false);
19        LOG.setDebugStepByStepState(false);
20        LOG.setActionLogState(false);
21        LOG.setActionLogStepByStepState(false);
22    }
```

Figure 4.3: Simulator interface code

The node's position in the mobility trace files must be read according to time, and since SUMO starts streaming the vehicles in the map vehicle by vehicle according to the number of vehicles in the scenario and the time between every successive vehicles. For that, we need to define the start and end time in the network simulator to prevent the simulator from starting with empty streets. For that the methods *startTime* and *endTime* are made.

The method *NumberOfNodes* is to define the number of nodes the simulator should create. It also adds all the vehicles variables needed from different classes (e.g. IP address, MAC address, network devices etc.). Then we create an instance of *SimulatorApplicationContainer* class to specify the number of nodes that request application from the RSU, the parameters the constructor takes are:

1. The starting node the constructor should install the application in,
2. The total number of nodes to install the application in,
3. The step between nodes to install the application on,
4. The starting time the nodes should start sending request for application from the RSU, if the node is ready to send request (received IP from the RSU the node its region),
5. The number of packets the node will request from the RSU.

The start method will invoke the simulator class to start simulating the protocol and keep going until the simulation end.

4.3.2 Start method

Figure 4.4 shows a flow chart of the start method. The start method starts by creating the nodes and RSUs and integrating the application inside the nodes. After this initialization, a loop starts using the time as its step value. The time used is not a static step interval, but instead, we used event-based simulator. The next time step used is reserved by the nodes to execute an event, e.g. send an IP request, receiving a packet etc.

The simulator reserves a static tic to update the node position. We defined this time tic with the value of one millisecond. When this tic arrives, the simulator goes to every node instance and read the text file corresponding to this specific node, and updates the position variable of that node.

In each round of the loop, the simulator checks all the nodes and RSUs for events that need handling. The first method invoked by the start method that interacts with the nodes is the *creatIpRequestAndAddThemToBufferToBeSentLater*. This method checks the nodes if they need new IP address, if true the method creates IP address request then saves it in the sending-buffer. The second method to execute is *checkForRelayPacketsToBeConsumed* for the RSUs. This method checks if any node received a packet successfully or a collision occurred, and handle the event accordingly. Then, a similar method is executed for the nodes as well. Then the start method invokes three other methods to check if there is an ACK or any other type of packets to be sent from send-buffer or temp-buffer for both nodes and RSUs. The simulator keeps looping until the end-time specified in the simulator interface is reached.

4.3.3 IP address request packet creation

The process of creating IP address and saving it in the sending-buffer is done using two methods. Figure 4.5 shows the process for a single node. The first method the simulator uses to check and prepare for creating IP address packet is *setVariablesForIpRequests*. This method checks all the nodes for one of two cases. The first is if they need IP address (the node just entered the simulation and needs IP). The second case is when the nodes have an IP but reached the zone the coming RSU. If one of these cases is true, the method set all the node variables to need new IP address. According to the node's position, if the node is in RSU range, it will set the variable responsible for the type of sending to send the request directly to the RSU so no multi-hop process is needed. If the node is out of the RSU range, the variable will be set to use routing process.

When the start method invokes the create IP request method, the start method checks the nodes' variable dedicated for the IP request part set in the *setVariablesForIpRequests* method. If the node needs IP, the method will create an IP request packet and fill the header using the nodes positions and the information in the variables responsible for the type, and save the IP request packet in the buffer according to the protocol used.

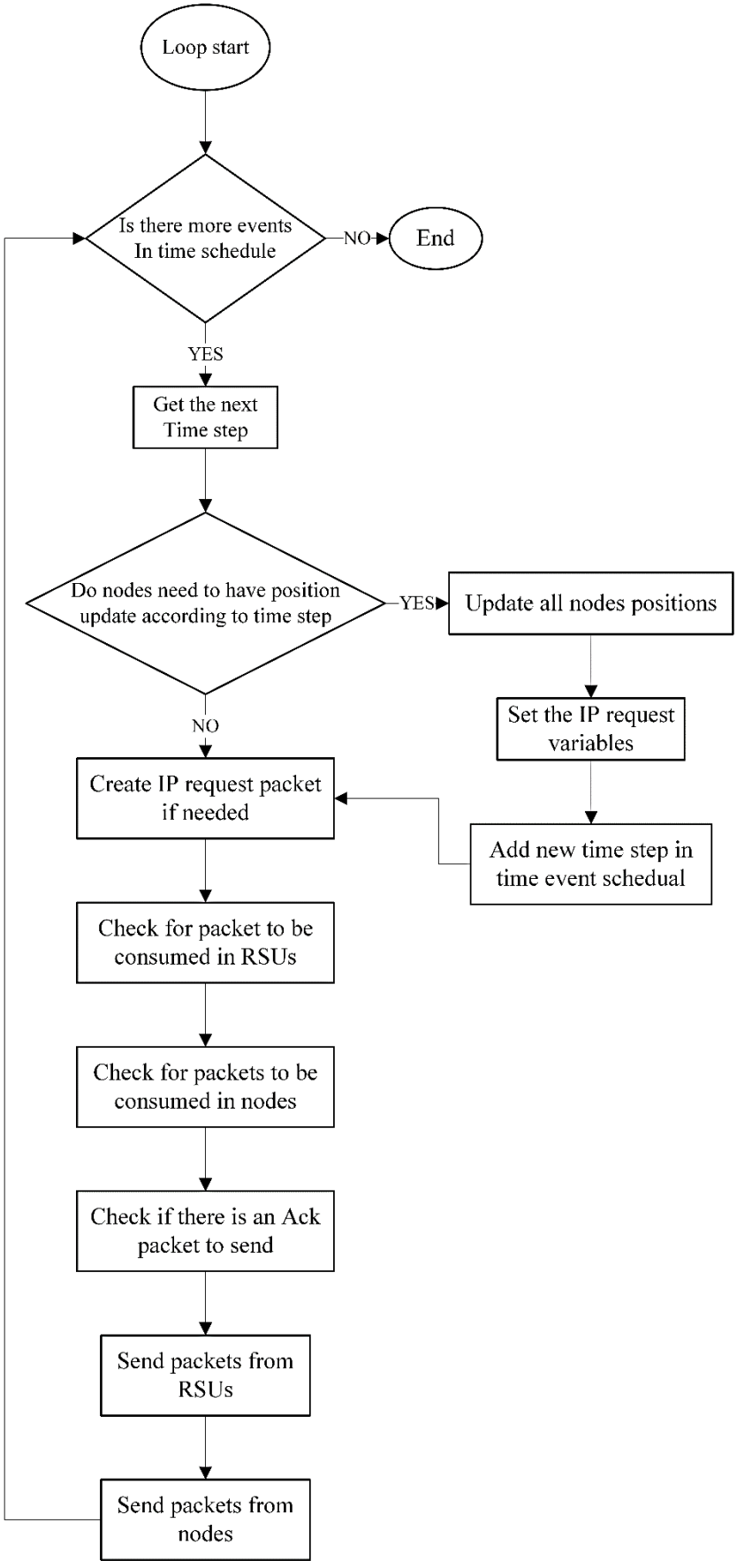


Figure 4.4: Main process of the start method

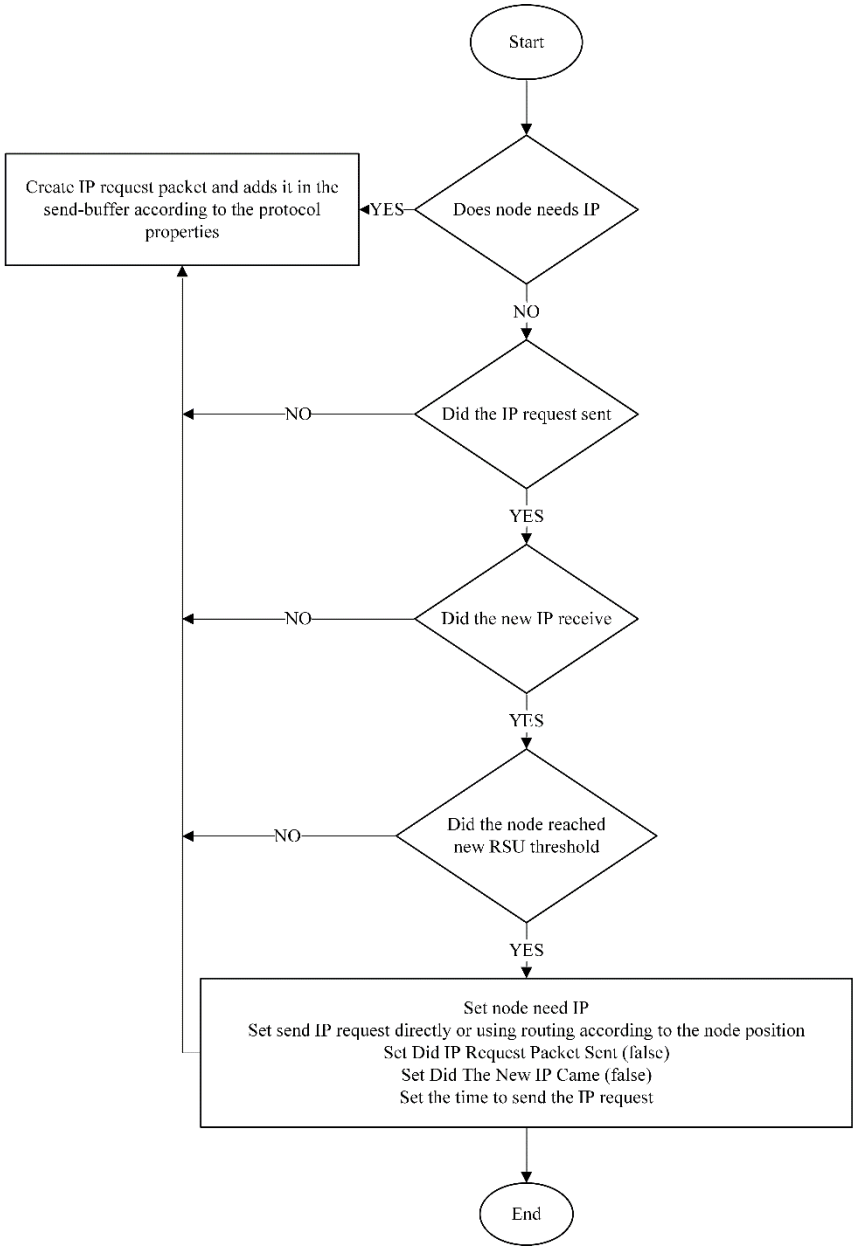


Figure 4.5: IP request creating process

4.3.4 Transmission method

This method searches in the sending buffer for packets to send according to sending time and priority, then sends the packet with respect to the protocol used in Figure 4.6. The first step done in sending packet process for both ROXY and BRAVE is to loop through all the nodes and check if there are packets to send in the send-buffer. If any is found, it selects from the packets that their time-to-send is equal to or less than the current time. Then from these packets, the method chooses the packet with the highest priority according to the used protocol's priority system.

After selecting a packet, sending preparation checks are made. The method calls another method *checkIfNodeIsReadyToSendUsingRearDevice* to check the following

1. Is the node we are checking is still in the simulator,

2. Is the medium is clear to send in,
3. Is the network device busy to send,
4. Is the network device allowed to send,

This method returns two values. The first value this method returns is a Boolean value representing the state of the network device if ready or not. If the device is not ready, the second value come in use, which is a double value, representing the time the device will be free in. If the node is not ready to send, the method changes the sending time to the new one, returns it to the send-buffer, and continues to the next node.

To send the packet, the method creates a delivery package to be sent to the receivers containing

- The Packet,
- The device ID used to be freed when the transmission ends,
- Transmission ending time to inform the receiver when it is ready to consume the packet,
- The ID of the sending node to free its device

The method checks if the node is ready to send in the opposite direction to send the NAS packet through. Then it creates a new NAS packet and inserts it in a sending package, and uses the same procedure to send the normal packet.

The simulator uses a text file for each street to represent the seven channels medium to write in it the sending information. In order for the nodes to know the channel's state if it is clear to send or not, there should be an entity representing the transmission medium. Hence, we used a text file for each street. Whenever a node starts to send, it calls *RoutingProtocol* class and invokes the *sendPacket* method to open the text file for the street it is in, and writes the following:

1. Transmission starting and ending time,
2. Transmission start and end position,
3. Transmission direction,
4. Sending node ID
5. Channel used in transmission,
6. Edge (street side) the node in.

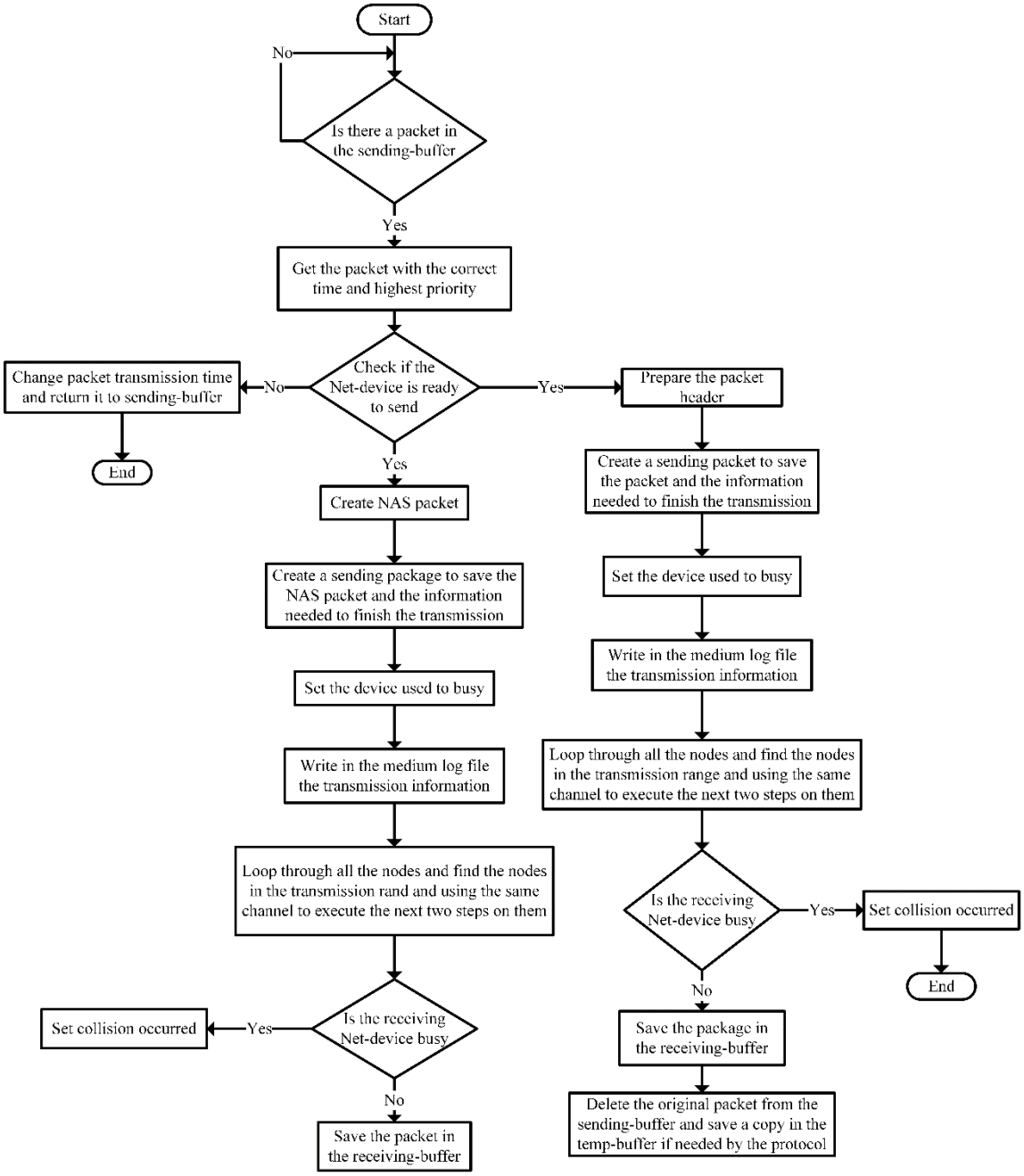


Figure 4.6: Packet sending process

After checking the node for readiness, it calls the method *sendPacket* to transmit the packet in the medium in the *Node* class. This method sets the network device to busy, and sets the time it will free the device in. Then, it calls the *sendPacket* in the routing protocol to write the sending information in the medium log file.

Now after everything is set, the method starts delivering the packet to the receivers. This part of the method starts by a loop to check all the nodes in the simulation about the following:

1. Is the node in the simulation,
2. Is the node in the same street as the sender is,
3. Is the node using the same channel as the sender,

4. Is the network device associated with this transmission clear to receive. If the device is busy receiving from another source, the method will set collision occurred in this device, and both packets will be dropped. The collision will be registered to the collision tracing system,
5. Is the node in the transmission range.

The first four checks are done by a separated method called *isNodeGoodToReceive* for reusability.

If all these checks were true, then the method saves the transmitted package in its receiving-buffer to be consumed when the transmission ends, and sets the node variables as follow:

1. Set the device associated with the transmission busy,
2. Set the time to free the device to the time to finish the transmission,
3. Set the device not allowed to send and the time to allow sending,

When the transmission is over, the original packet is either removed from the send-buffer, or removed from the send-buffer to the temp-buffer if needed further retransmission according to the protocol used.

4.3.5 CheckForRelayPacketsToBeConsumed method

This method is responsible for checking if any node has a packet in the receiving buffer and needs to be processed. The start method calls this method every time tic to check if there is a RSU or a node with a packet in its receive-buffer to handle the packet received, and to drop the packet if there is a collision occurred.

The method starts by checking if the node's network device suffered from a collision. The method checks if there were a collision occurred to the network device to drop the packet received from the receive-buffer. The method will set the device to (clear-to-receive) state only when the transmission or the collision is finished.

The next step is to import and process the packet received according to the protocol used to drop it, retransmit it, consume it and change the node's IP address, update the trace system, create ACK or RESPOND packet, or create another type of packets. Then clear the receive-buffer and change the network state to clear to receive.

4.3.6 Tracing system

We created five types of tracing systems for debugging and result outputs.

4.3.6.1 LOG trace system

To be sure the simulator is running correctly and the outputs are reliable we used the *LOG* class with two type of logging system. One to state all the actions done by the simulator. This includes what method we entered or returned from, and any action done in the method running, and any action done by the nodes and RSUs we see it needs to trace. The second one is the *ACTION* log system that traces the actions done by the nodes and RSUs we see it needs to trace. The output of the log system is saved in a text file. Figure 4.7 shows the code used for the debug type in the LOG class.

```
34      public static void DEBUG(String str)  {
35          BufferedReader stdin = new BufferedReader(new InputStreamReader(System.in));
36          if(LOG.isDebugOn()){
37              PrintWriter outPutFile = openTheFile("C:\\traceSystem\\Log\\debug_file.txt");
38              outPutFile.println(str);
39              System.out.println(str);
40              outPutFile.close();
41              if(LOG.isDebugStepByStepOn()){
42                  try {
43                      stdin.readLine();
44                  } catch (IOException e) {
45                      System.out.println("error in log method");
46                      e.printStackTrace();
47                  }
48              }
49          }
50      }
```

Figure 4.7: Debug method in LOG class

4.3.6.2 IP trace system

To trace the IP addresses received by the nodes and the IP request packets sent to receive this new IP, we created a trace system to trace only the IP acquisition. Whenever a node sends an IP request packet, it increments a counter for the number of IP requests sent. After the new IP is received, a process starts to create a text file with the following information:

1. The node's ID,
2. The time the first IP request is sent,
3. The time the new IP is received,
4. The delay to receive the new IP,
5. The number of retransmissions of the IP request.

The text file named with the ID of the node in a folder named "*IpRequest*" for easy access and reading.

4.3.6.3 Collision trace system

This tracing system saves a line of text in the collision trace file containing the following:

1. The node ID causing the collision,
2. The node ID having the collision,
3. The device suffered the collision for ROXY protocol,
4. The time the collision occurred in.

Every time a collision occurs, the collision trace method in the LOG class opens the collision file in the collision folder and writes the collision information in it.

4.3.6.4 Application trace system

Application tracing system creates two types of output files: one is for every node named after the node ID and its contents is shown in Figure 4.8.

node 1190 start time 1.4604384869199998E9 number of packets needed 10
paket id 0 last packet came time 0.0 this packet came time 1.4620046553600001E9 with delay from last packet 1566168.4400002956
paket id 1 last packet came time 1.4620046553600001E9 this packet came time 1.4620714893600001E9 with delay from last packet 66834.0
paket id 2 last packet came time 1.4620714893600001E9 this packet came time 1.4621383233600001E9 with delay from last packet 66834.0
paket id 3 last packet came time 1.4621383233600001E9 this packet came time 1.4622051573600001E9 with delay from last packet 66834.0
paket id 4 last packet came time 1.4622051573600001E9 this packet came time 1.4622719836200001E9 with delay from last packet 66826.25999999046
paket id 5 last packet came time 1.4622719836200001E9 this packet came time 1.4623388106200001E9 with delay from last packet 66827.0
paket id 6 last packet came time 1.4623388106200001E9 this packet came time 1.4624056356200001E9 with delay from last packet 66825.0
paket id 7 last packet came time 1.4624056356200001E9 this packet came time 1.4624724696200001E9 with delay from last packet 66834.0
paket id 8 last packet came time 1.4624724696200001E9 this packet came time 1.46253929448E9 with delay from last packet 66824.8599998951
paket id 9 last packet came time 1.46253929448E9 this packet came time 1.4626061206200001E9 with delay from last packet 66826.1400001049
Application fully recived and its full trace deta is
Start Time = 1.4604384869199998E9
End Time = 1.4626061206200001E9
total time = 2167633.700000286
Average delay between packet = 73904.57470363384

Figure 4.8: Application trace file for a specific node

The other output file contains two columns: one representing the number of application packets requested by the nodes from the RSUs, the second column represents the number of application packets received by the nodes from the RSUs.

4.3.6.5 Total IP delay trace system

The output file of this system contains three columns: one represents the overall IP request sent, the second column represents the number of IP addresses received by the nodes, and the last column is to show the average time delay suffered from all the nodes received IP address.

4.4 Result extracting tool

The last set of tools we used are number of small programs to go through the trace files and extract the average time delay for IP acquisition and application packet receiving, and bitrate experienced by the nodes.

Chapter 5. Simulations & Results

In this chapter, we will discuss the results obtained using the tools described in the previous chapter. We will start by stating the scenario used in simulation and the parameters used, and lastly we will state the results and compare it to BRAVE routing protocol [14].

5.1 Simulation of Vehicle's Mobility

We created a simple map consisting of one 2000m two way street. Then we feed the map with six vehicles streams, each one inserts 100 vehicles in the street to make it 600 in total. The streams of vehicles will be inserted gradually in the system. Hence, we start our network simulation on 133 second from the start of the streams feed. The RSUs will be installed in both intersections. Figure 5.1 shows the map and the density of vehicles in different positions.

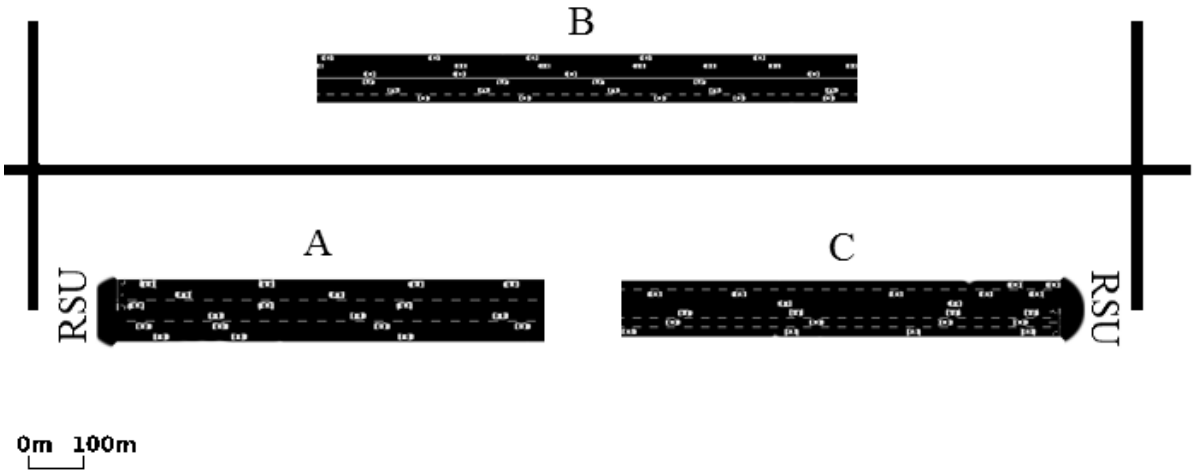


Figure 5.1: The simulator environment topology used. A, B, C are snapshots of the left, middle, and right sides of the street respectively

5.2 Simulation of the Network

Packets are transmitted in the network when the vehicles moving in the street reache steady state. We considered the worst-case scenario, where all the vehicles need IP addresses. The vehicles ask for IP addresses whenever they enter a threshold position for a new RSU (in this case, the midpoint between consecutive RSU's). The simulation included 600 vehicles. We tested the cases where (10, 50, 150, 250, and 600) vehicles request application packets from the RSUs after obtaining the IP address. We assume that the nodes request 10 applications packets from the RSUs after obtaining their IPs. We simulate the algorithm and

see its efficiency against these densities. We assume that the RSUs can use the seven channels simultaneously.

We assume that vehicles are equipped with two directed antennas and an Omni-directional antenna for the CCH, and all the vehicles and RSUs have a unified transmission range of 300m. Table 5.1 summarizes the used simulation parameters.

We made two types of tests. The first type is for ROXY alone to show the effect of incrementing the application requests and reply transmissions on the collisions, number of packets received successfully by the destinations, and time delays. The second type is to compare our work with BRAVE routing protocol.

Table 5.1: list of simulation parameters

Parameter	**Value**
Application Packet Size	200kb
Number of Vehicles Request Application	10, 50, 150, 250, and 600
Active Vehicles	1146
Transmission range	300m
Transmission rate	6 Mbps

We choose BRAVE routing protocol for the similarity in using geographic forwarding, opportunistic next hop selection, and beacon-less routing scheme. We will examine the average time delay for the IP acquiring, the application packets received, average time delay for acquiring all the application packets needed, number of collisions, and the average bit rate the vehicle actually experiences while receiving the application packets.

5.3 Results

The results will be listed in two parts, one for ROXY alone and the second part will be the comparison between ROXY and BRAVE protocols.

5.3.1 ROXY results

We will study the effect of increasing application packets requests on the system performance. The result will be divided to three categories IP addresses, application packets acquisition, and number of collisions

5.3.1.1 IP address results

In this part, we will show the number of nodes that successfully received IP addresses directly or by using routing scheme, and the time delay they suffered to receive them.

Figure 5.2 shows the number of vehicles that received IP addresses from the RSUs directly without the need for routing since these vehicles are in the RSU transmission range for the five different runs.

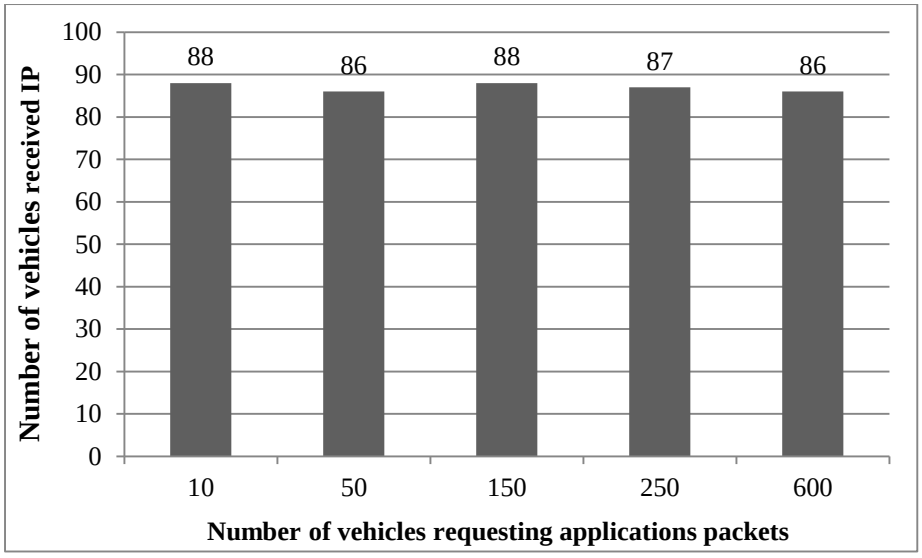


Figure 5.2: Number of vehicles received IP addresses directly from the RSU

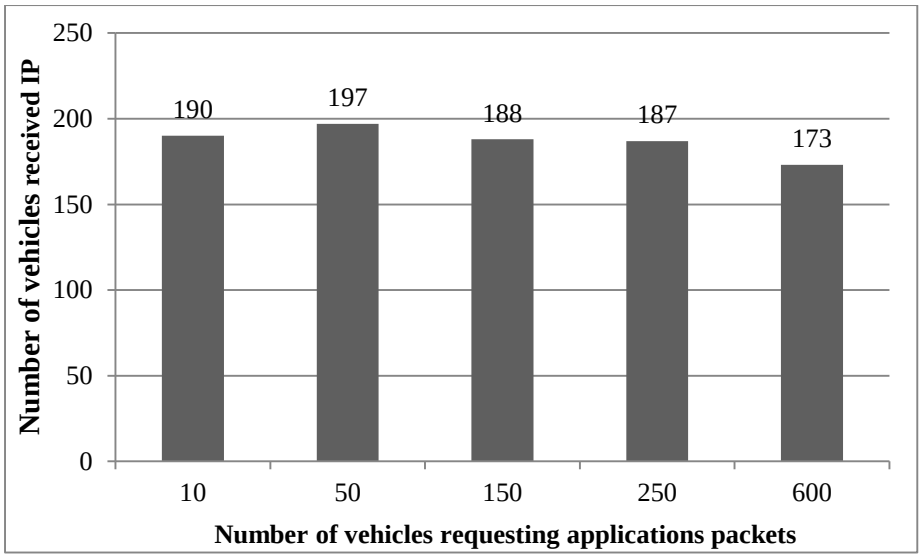


Figure 5.3: Number of vehicles received IP addresses using routing from the RSU

Figure 5.3 shows the number of vehicles that received IP addresses using multi-hop scheme since they are out of the RSU range. In both figures, we noticed the expected decrease in IP addresses received for the same duration because of the increase in transmission of application packets. We didn't use a scheme to distribute and install the application request in vehicles according to their location in the map but according to their id, that lead to increase the transmission density in one place which in turn increased the delay slightly, which explains the abnormality in the second run.

Figure 5.4 shows the average time delay the node experiences to acquire the new IP addresses they request from the RSUs directly without the need for routing for the different runs related to Figure 5.2.

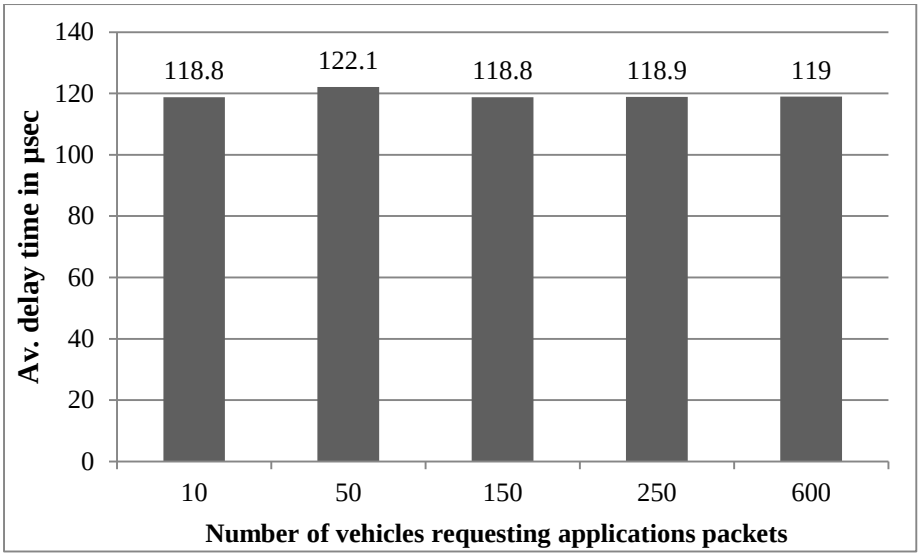


Figure 5.4: Average time delay the node experienced while directly receiving the IP addresses

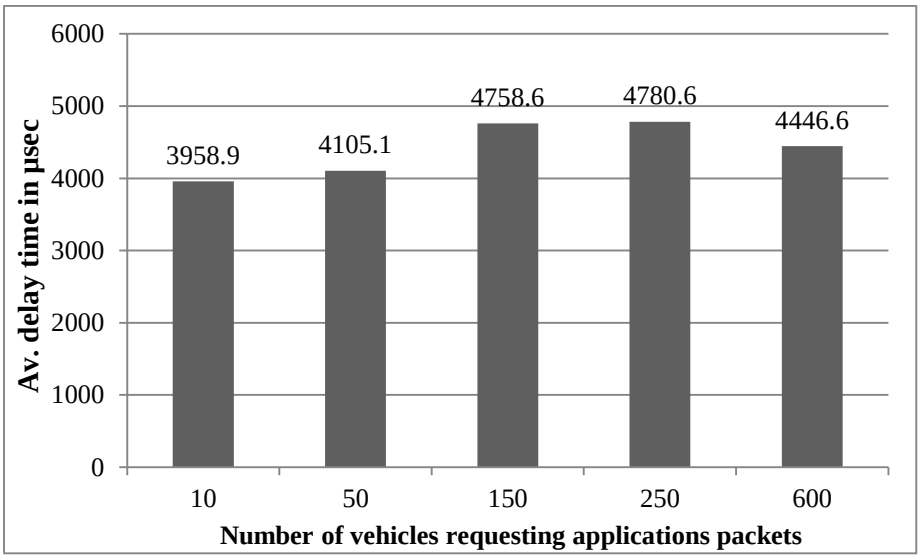


Figure 5.5: Average time delay the node experienced while receiving the IP addresses using routing

The nodes outside the RSUs transmission range received their IP addresses through routing mentioned in Figure 5.3, suffered from larger time delays than the vehicles received them directly, as shown in Figure 5.5.

5.3.1.2 Application results

Here we will show the number of application packets received through routing or directly from the RSUs, the average time delay they suffered from to receive the application packets, and the data bit-rate the vehicles experienced while receiving the application packets.

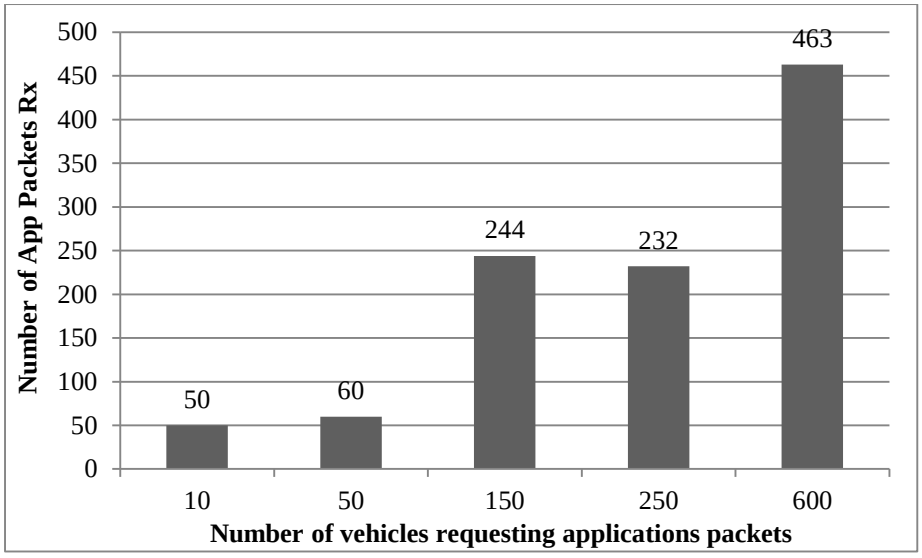


Figure 5.6: Number of application packets received by the vehicles directly

First, the number of application packets received directly from the RSUs for the run duration we used across the five runs is shown in Figure 5.6. Figure 5.7 shows the number of vehicles that received application packets using multi-hop. The first column equals zero because all the nodes requesting application in that run are inside the RSUs transmission range.

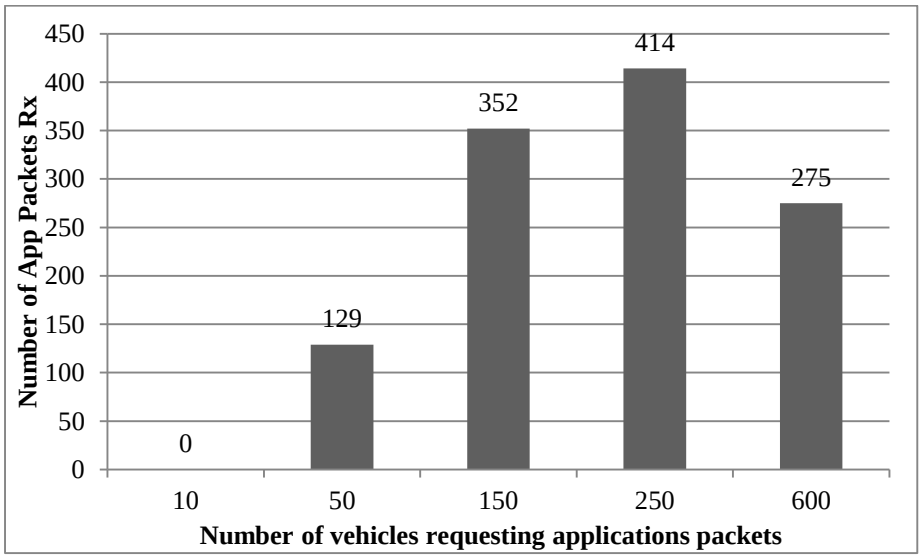


Figure 5.7: Number of application packets received by the vehicles using routing

Figure 5.8 and Figure 5.9 show the average time delay the vehicles suffered from to receive their application packets related to Figure 5.6 and Figure 5.7 respectively. As expected the time delay increases with the increase of transmission density in the system. For the first column in Figure 5.9, it is not available since there are no packets received using routing to calculate delay for it.

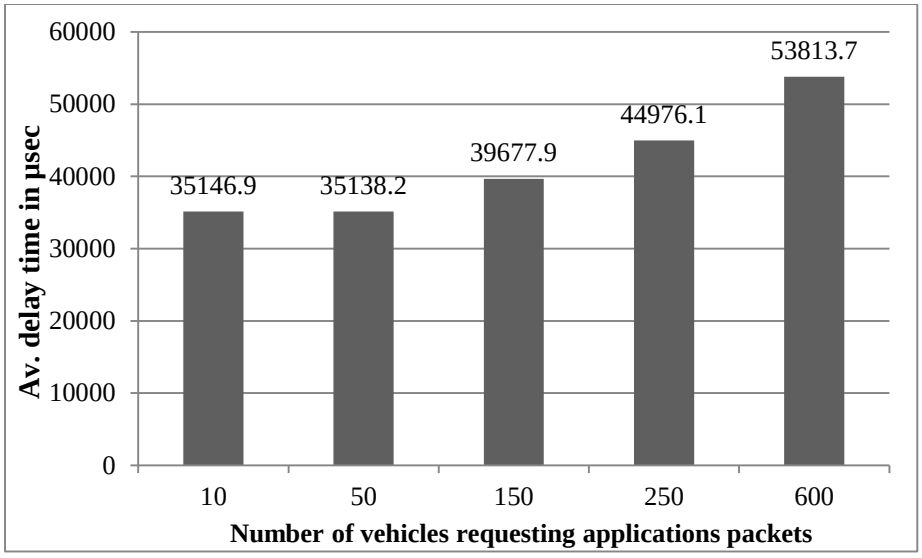


Figure 5.8: Average time delay to receive the application packets directly from the RSUs

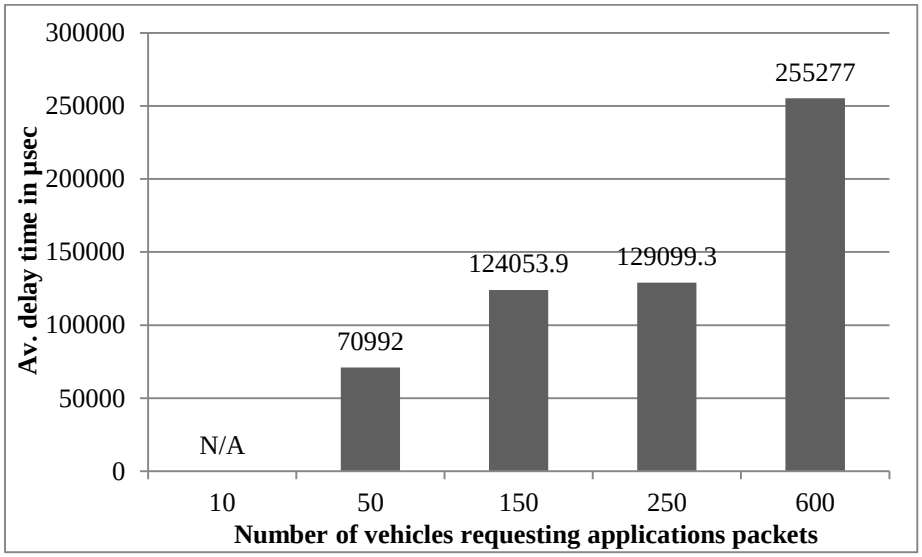


Figure 5.9: Average time delay to receive the application packets using routing from the RSUs

The last part of the application results section is to show the average data bit-rate the vehicle experienced while receiving their application packets directly and using routing scheme shown respectively in Figure 5.10 and Figure 5.11. The bit rate used in the test simulation is 6 Mbps.

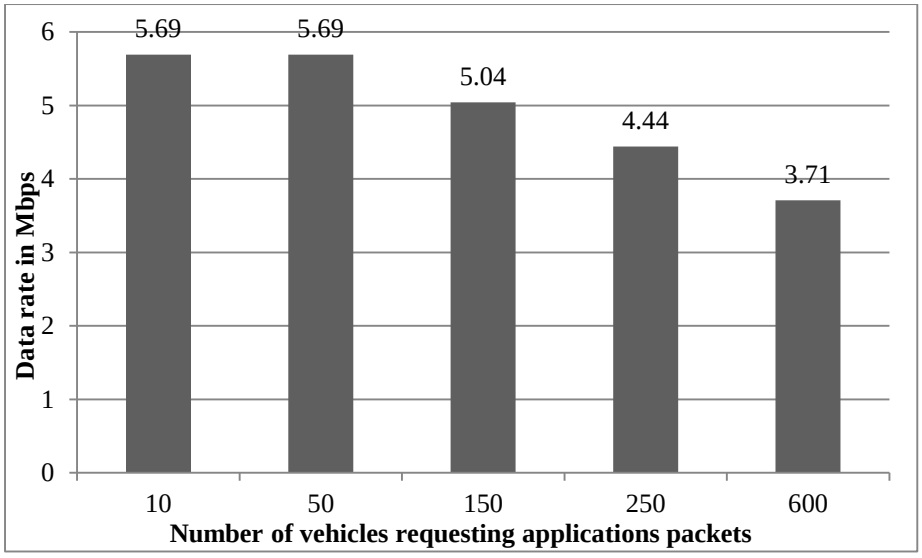


Figure 5.10: Data bit-rate vehicles experiences receiving the application packets directly

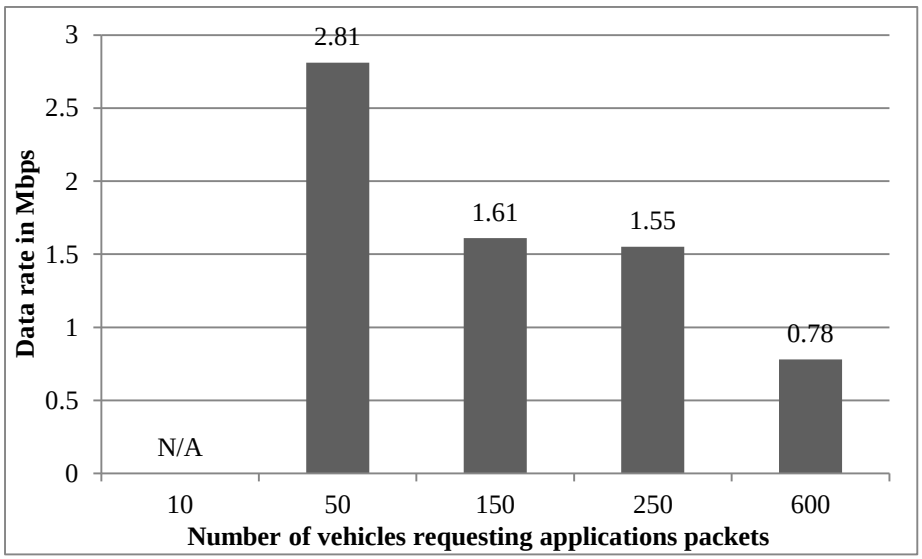


Figure 5.11: Data bit-rate vehicles experiences receiving the application packets using routing

5.3.1.3 Collision results

The last test result we acquired is the number of collisions occurred during the test run of the simulation which is shown in Figure 5.12. After studying collisions occurred, most of the collisions are grouped in two types: first type is made by the RSUs to the nodes inside the RSU's range receiving a packet from a node inside the RSU's range intended to the nodes outside the RSU's range which we permitted since it does not affect the transmission system and it gives a slight improvement to the RSU transmission time delay. The second type of collisions is caused by nodes changed the channel they used according to the new IP address received, and this node did not receive the present NAS packet, and since the node have packets in its send-buffer it starts sending the packet and NAS packet as stated in the protocol which caused this type of collisions.

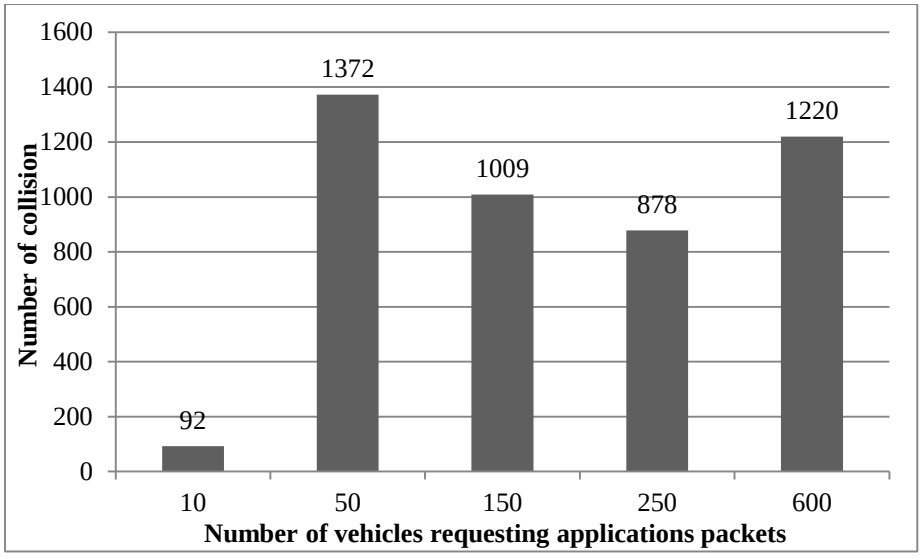


Figure 5.12: Collision occurred in the simulation

5.3.2 ROXY and BRAVE comparison

In this section, we will compare ROXY protocol to a BRAVE routing protocol due to the similarity between them. We will show the results in three categories: IP, application, and collision results.

5.3.2.1 IP addresses results

In this section, we will compare the number of IP addresses received to show the improvement using ROXY and the new collision avoidance protocol (DOT/CP). From Figure 5.13 we notice the improvement compared to BRAVE protocol.

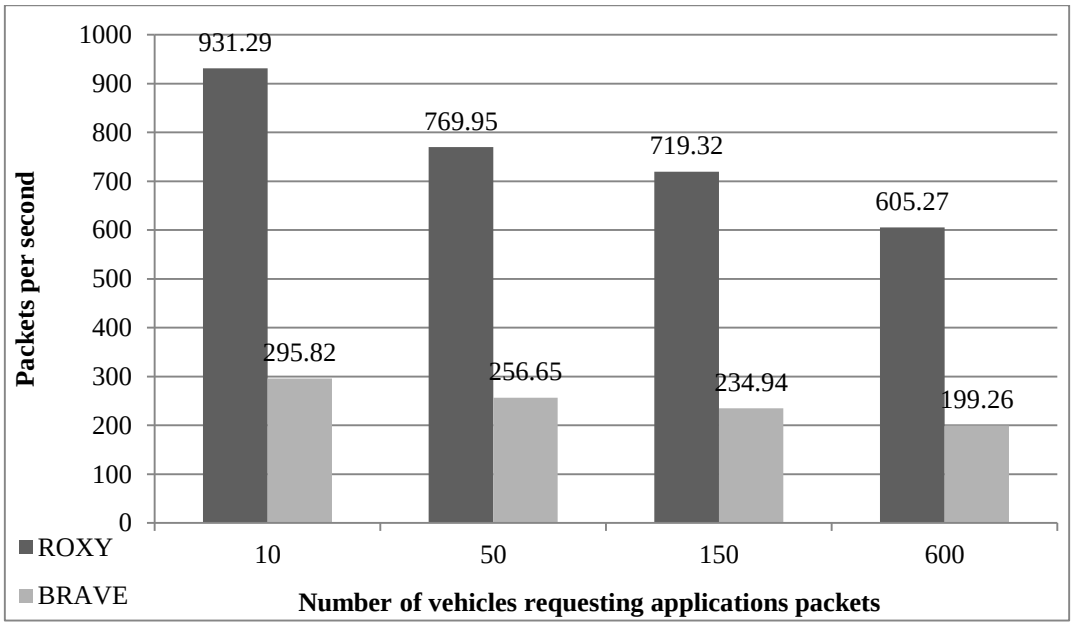


Figure 5.13: Number of IP packets per-second the vehicles received

5.3.2.2 Application results

For the application, we will compare the data bit-rate the application reply packet were received with. Table 5.2 shows the data bit-rate the nodes received the application reply packets with and the improvements compared to BRAVE protocol. The (N/A) part in first run (10) all the vehicles ordered applications where inside the RSUs transmission range, while in the remaining runs BRAVE protocol failed to receive application reply packets due to the high collision rate in the multi-hop reception case.

Table 5.2: The data bit-rate the vehicles received the application reply packets with

Run	Protocol	Data bitrate (Mbps)	
		Directly	Using routing
10	**ROXY**	5.69	N/A
	BRAVE	3.28	N/A
50	**ROXY**	5.69	2.81
	BRAVE	3.73	N/A
150	**ROXY**	5.04	1.61
	BRAVE	2.46	N/A
600	**ROXY**	3.71	0.78
	BRAVE	1.24	N/A

5.3.2.3 Collision results

This section shows the improvement in the number of collisions of the new collision avoidance compared to CSMA/CD. Figure 5.14 shows huge difference in the number of collisions occurred for both protocols, which shows that the normal methods to detect collisions are not good enough to meet the emergency time limitation, and it can lead to system failure.

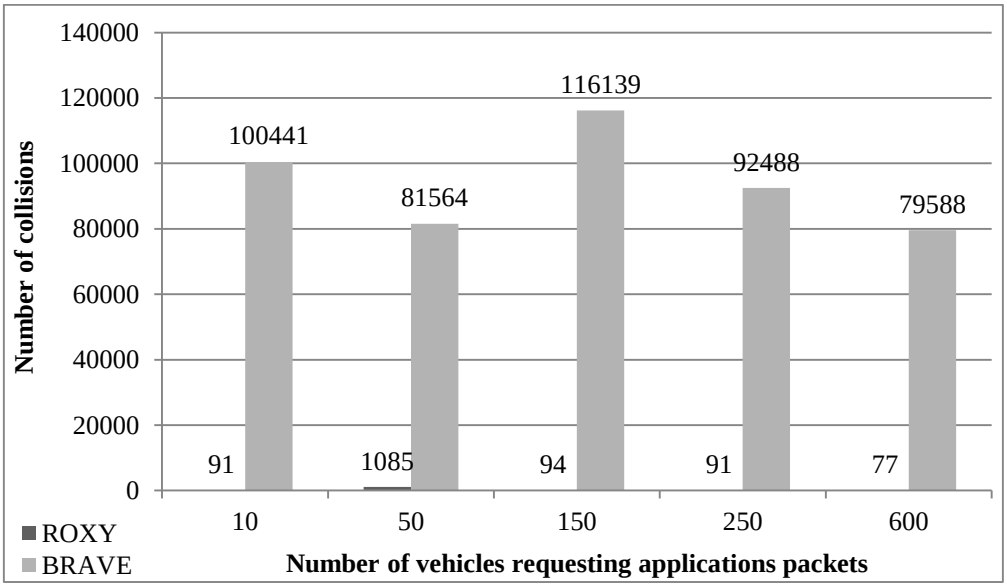


Figure 5.14: Number of collision occurred during the different simulation runs

5.4 Conclusion

This chapter lists the results extracted from simulating ROXY protocol and BRAVE protocol in a loaded system to prove the improvement gained by using ROXY. The results show that the protocol can stay functional even with the increase of the network traffic load, and the major reduction in the number of collisions compared to the results gained using CSMA/CD. We have also shown the improvement in the network throughput by using ROXY.

Chapter 6. Conclusion and Future Work

In this thesis, we proposed “Receiver as Proxy” routing protocol (ROXY) for Vehicular Ad-hoc Networks (VANETs) in high densities urban areas. ROXY is a beacon free protocol. We developed algorithms in ROXY to effectively use the available resources e.g. the transmission medium and the channels. ROXY reduces the time overhead while sending packets, and it increases the throughput compared to other protocols, especially in high density scenarios.

We show how choosing the next hop in an efficient way can improve the system overall efficiency. We worked on lowering the load on the sending node by creating a self-organized system without the need for a predetermined next hop node. The receivers elect the least cost node in a distributed fashion. In this work, cost is represented as the hop count, which can be mapped to the time delay required in relaying packets.

The thesis proposed a new collision avoidance protocol called DOT/CP *(Denial Of Transmission with Collision Prevention)*, which is suitable for the directional antennas installed in the front and back of each vehicle. The new protocol proved, through simulations, its ability to reduce the collision significantly.

To simulate the protocol and test its efficiency, we created a new VANET simulator using JAVA to evaluate this work. The new simulator works smoothly with SUMO vehicles mobility tool.

We compared our work with previous beacon free routing protocol called BRAVE routing protocol and the simulation shows large improvements in average time delay for both the IP and application packets. Moreover, it outperformed BRAVE in the number of collisions occurred, and the actual amount of vehicle received IP and application packets.

To gain more benefits further improvement to ROXY will increase the efficiency. During the development of ROXY protocol we focus on technics to lower the overheads in crowded areas for the SCH, and did not focused on the low density or local maximum problems. More studies and researches is needed to get better results in these cases e.g. the RSU announce the streets that suffer from disconnections, the nodes in these streets do not accept to receive the packets while receiving them in the intersections and leave the nodes in other streets to be the proxy for it, or by adding hybrid connections technics for streets with low densities or suffering from local maximum. Further simulation is needed to gain more results for more complex scenarios in large maps and various situations e.g. accidents or blocked roads during runtime.

REFERENCES

[1] P. S. Nithya Darisini and N. S. Kumari, "A survey of routing protocols for VANET in urban scenarios," in *Pattern Recognition, Informatics and Medical Engineering (PRIME), 2013 International Conference on*, 2013, pp. 464–467.

[2] J. R. Gallardo, D. Makrakis, and H. T. Mouftah, "Performance analysis of the EDCA medium access mechanism over the control channel of an IEEE 802.11 p WAVE vehicular network," in *Communications, 2009. ICC'09. IEEE International Conference on*, 2009, pp. 1–6.

[3] C. Campolo and A. Molinaro, "On vehicle-to-roadside communications in 802.11p/WAVE VANETs," in *2011 IEEE Wireless Communications and Networking Conference (WCNC)*, 2011, pp. 1010–1015.

[4] C. Campolo and A. Molinaro, "Improving V2R connectivity to provide ITS applications in IEEE 802.11 p/WAVE VANETs," in *Telecommunications (ICT), 2011 18th International Conference on*, 2011, pp. 476–481.

[5] M. S. Almalag, "TDMA slot reservation in cluster-based vanets," Old Dominion University, 2013.

[6] Y. Du, L. Zhang, Y. Feng, Z. Ren, and Z. Wang, "Performance analysis and enhancement of IEEE 802.11 p/1609 protocol family in vehicular environments," in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, 2010, pp. 1085–1090.

[7] S. Eichler, "Performance evaluation of the IEEE 802.11 p WAVE communication standard," in *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*, 2007, pp. 2199–2203.

[8] D. Jiang and L. Delgrossi, "IEEE 802.11 p: Towards an international standard for wireless access in vehicular environments," in *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE*, 2008, pp. 2036–2040.

[9] "IEEE Standard for Wireless Access in Vehicular Environments (WAVE)–Multi-channel Operation," *IEEE Std 16094-2010 Revis. IEEE Std 16094-2006*, pp. 1–89, Feb. 2011.

[10] J. Jia, Q. Zhang, and X. Shen, "HC-MAC: A hardware-constrained cognitive MAC for efficient spectrum management," *Sel. Areas Commun. IEEE J. On*, vol. 26, no. 1, pp. 106–117, 2008.

[11] S. Katragadda, C. N. S. Ganesh Murthy, R. Rao, S. Mohan Kumar, and R. Sachin, "A decentralized location-based channel access protocol for inter-vehicle communication," in *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, 2003, vol. 3, pp. 1831–1835.

[12] N. Chandra Rathore, S. Verma, and G. S. Tomar, "CMAC: a cluster based MAC protocol for VANETs," in *Computer Information Systems and Industrial Management Applications (CISIM), 2010 International Conference on*, 2010, pp. 563–568.

[13] R. S. Tomar and S. Verma, "RSU centric channel allocation in Vehicular Ad-hoc Networks," in *Wireless Communication and Sensor Networks (WCSN), 2010 Sixth International Conference on*, 2010, pp. 1–6.

[14] P. M. Ruiz, V. Cabrera, J. A. Martinez, and F. J. Ros, "Brave: Beacon-less routing algorithm for vehicular environments," in *Mobile Adhoc and Sensor Systems (MASS), 2010 IEEE 7th International Conference on*, 2010, pp. 709–714.

[15] H. Shi, C. Ma, L. Chen, and Z. Ding, "IDVR-PFM: A connectivity-oriented VANET routing protocol in urban scenarios," in *Intelligent Control and Information Processing (ICICIP), 2013 Fourth International Conference on*, 2013, pp. 541–544.

[16] M. Rani and N. S. Gill, "Comparative Study of Various VANET Routing Protocols," *IJCSMS Int. J. Comput. Sci. Manag. Stud. Spec. Issue Of*, vol. 12, 2012.

[17] T. Clausen, P. Jacquet, C. Adjih, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized link state routing protocol (OLSR)," 2003.

[18] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *ACM SIGCOMM Computer Communication Review*, 1994, vol. 24, pp. 234–244.

[19] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA'99. Second IEEE Workshop on*, 1999, pp. 90–100.

[20] D. B. Johnson and D. A. Maltz, “Dynamic source routing in ad hoc wireless networks,” in *Mobile computing*, Springer, 1996, pp. 153–181.
[21] 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*, 2000, pp. 243–254.
[22] C. Lochert, M. Mauve, H. Füßler, and H. Hartenstein, “Geographic routing in city scenarios,” *ACM SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 9, no. 1, pp. 69–72, 2005.
[23] C. Lochert, H. Hartenstein, J. Tian, H. Fussler, D. Hermann, and M. Mauve, “A routing strategy for vehicular ad hoc networks in city environments,” in *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, 2003, pp. 156–161.
[24] H. Qin, C. Yu, L. Guo, and Y. Zhou, “Adaptive state aware routing protocol for vehicle ad hoc network in urban scenarios,” in *2013 10th IEEE International Conference on Control and Automation (ICCA)*, 2013, pp. 1376–1381.
[25] A. A. Ba, A. Hafid, and J. Drissi, “Broadcast Control-Based Routing Protocol for Internet Access in VANETS,” in *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International*, 2011, pp. 1766–1771.
[26] V. Naumov and T. R. Gross, “Connectivity-Aware Routing (CAR) in Vehicular Ad-hoc Networks.,” in *InfoCOM*, 2007, vol. 26, pp. 1919–1927.
[27] V. Naumov, R. Baumann, and T. Gross, “An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces,” in *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, 2006, pp. 108–119.
[28] J. Jayavel, R. Venkatesan, and D. Vinoth, “EFFICIENT MULTIPATH ROUTING PROTOCOL FOR VANET USING PATH RESTORATION,” *Int. J. Eng. Technol. 0975-4024*, vol. 5, no. 5, 2013.
[29] M. K. Marina and S. R. Das, “On-demand multipath distance vector routing in ad hoc networks,” in *Network Protocols, 2001. Ninth International Conference on*, 2001, pp. 14–23.
[30] B. L. Raviteja, B. Annappa, and M. P. Tahiliani, “CAMP: congestion adaptive multipath routing protocol for VANETs,” in *Advanced Computing, Networking and Security*, Springer, 2012, pp. 318–327.
[31] J. Nzouonta, N. Rajgure, G. Wang, and C. Borcea, “VANET routing on city roads using real-time vehicular traffic information,” *Veh. Technol. IEEE Trans. On*, vol. 58, no. 7, pp. 3609–3626, 2009.
[32] H. Fü\s sler, J. Widmer, M. Käsemann, M. Mauve, and H. Hartenstein, “Contention-based forwarding for mobile ad hoc networks,” *Ad Hoc Netw.*, vol. 1, no. 4, pp. 351–369, 2003.
[33] M. Chawla, N. Goel, K. Kalaichelvan, A. Nayak, and I. Stojmenovic, “Beaconless position based routing with guaranteed delivery for wireless ad-hoc and sensor networks,” in *Ad-Hoc Networking*, Springer, 2006, pp. 61–70.
[34] M. Gramaglia, M. Calderon, and C. J. Bernardos, “TREBOL: Tree-based routing and address autoconfiguration for vehicle-to-internet communications,” in *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, 2011, pp. 1–5.
[35] L.-D. Chou, J.-Y. Yang, Y.-C. Hsieh, D.-C. Chang, and C.-F. Tung, “Intersection-based routing protocol for VANETs,” *Wirel. Pers. Commun.*, vol. 60, no. 1, pp. 105–124, 2011.
[36] H. Menouar, F. Filali, and M. Lenardi, “A survey and qualitative analysis of MAC protocols for vehicular ad hoc networks,” *Wirel. Commun. IEEE*, vol. 13, no. 5, pp. 30–35, 2006.
[37] M. X. Cheng, Y. Ma, and Y. Wang, “Improving channel throughput of WLANs and ad hoc networks using explicit denial of requests,” in *Global Telecommunications Conference, 2006. GLOBECOM’06. IEEE*, 2006.
[38] M. X. Gong, R. Stacey, D. Akhmetov, and S. Mao, “A directional CSMA/CA protocol for mmWave wireless PANs,” in *Wireless Communications and Networking Conference (WCNC), 2010 IEEE*, 2010, pp. 1–6.
[39] A. Nasipuri, S. Ye, J. You, and R. E. Hiromoto, “A MAC protocol for mobile ad hoc networks using directional antennas,” in *Wireless Communications and Networking Confernce, 2000. WCNC. 2000 IEEE*, 2000, vol. 3, pp. 1214–1219.

[40] Z. Huang, C.-C. Shen, C. Srisathapornphat, and C. Jaikaeo, “A busy-tone based directional MAC protocol for ad hoc networks,” in *MILCOM 2002. Proceedings*, 2002, vol. 2, pp. 1233–1238.
[41] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, “Recent Development and Applications of SUMO - Simulation of Urban MObility,” *Int. J. Adv. Syst. Meas.*, vol. 5, no. 3&4, pp. 128–138, Dec. 2012.

ملخص الرسالة

الشبكات المخصصه للمركبات (VANET) هي شبكات لاسلكيه بين المركبات المجهزة بوحدات ارسال واستقبال لاسلكيه ووحده معالجه الكترونية. تطبيقات الشبكات المخصصه للمركبات تتنوع من تطبيقات الامان حيث ان سرعة تسليم معلوماتها من اهم ما يميز ويحدد هذا النوع من التطبيقات، الى تطبيقات الالعاب والتسليه والولوج الى الانترنت. الشبكات المخصصه للمركبات هي جزء من الرؤية الشاملة لأنظمة النقل الذكية (ITS). المركبات في الشبكات المخصصه للمركبات لها طبيعه حركه ديناميكيه بسبب اختلافات السرع بينهم والتغير المفاجي في اتجاه الحركه، ولهاذا الخوارزميات المستعمله لتوجيه وايصال الرسائل والحزم يجب ان تاخذ التغير السريع لبنيه الشبكه بنظر الاعتبار اثناء تطويرها. عادتاً هذا ينجز عن طريق ارسال رسائل ارشاد من المركبات الى كل المركبات المحيطه بها بشكل دوري، هذه الرسائل الارشاد تحتوي معلومات عن المركبه كمكانها وسرعه المركبه الحاليه واتجاه السير واي معلومه اضافيه ممكن استخدامها في خوارزميه توجيه الرسائل، كل المركبات تجدد المعلومات لديها عن المركبات المحيطه عن طريق خزنها في جدول خاص بالمركبات المجاوره. ولاكن استخدام هذه الرسائل الارشاديه بشكل دوري سوف يخفض من كفائة الشبكه ويزيد الوقت الازم لايصال الانواع الاخرى من الرسائل كرسائل الامان التي تعتبر الوقت عامل حاسم وايضا تخفض عدد الرسائل الكليه المستلمه المقاسه بوحده الزمن.

في هذا البحث نقدم برتوكول توجيه للشبكات المخصصه للمركبات "المستلم كوكيل" (ROXY) للتغلب على المحددات التي واجهت ابحاث اخرى. المستلم كوكيل يستعمل نظام التوجيه الجغرافي مع نظام اختيار القفزه التاليه بشكل انتهازي لزياده عدد الرسائل الكليه المستلمه المقاسه بوحده الزمن في الشبكات المخصصه للمركبات. الحدف الاساسي للمستلم كوكيل هو حل مشكله الاختناقات المتزايده في الشبكه وتصادمات الرسائل في حاله الشوارع المكتضه بالمركبات التي تستخدم الشبكات المخصصه للمركبات.

المستلم كوكيل يحسن النظام عن طريق معالجة ثلاثة قضايا. الاولى هي الانتفاع من الموارد المكانيه عن طريق استخدام هوائيين موجهيين في كل مركبه، احدهم موجه الى امام المركبه والاخر في الاتجاه الخلفي للمركبه. هذان الهوائيان يستخدمان لارسال الرسائل في اتجاهيين متعاكسيين في نفس الوقت وبهذا نستخدم وسط الانتقال بشكل اكثر فعاليه وزياده عدد الرسائل الكليه المستلمه المقاسه بوحده الزمن. الثانيه اختار المركبه التي سوف تقوم بتولي مهمه اعادا ارسال الرساله يكون بين المركبات المستلمه داخليا بدون تدخل المركبه المرسله او كيان مركزي. الثالثه والاخيره تقليل الوقت اللازم لارسال الرسائل عن طريق اقتراح خوارزميه جديده لتجنب اصطدام الرسائل التي تستفيد من الهوائيان الموجهان وشكل الشوارع في المدن. نتائج المحاكات تظهر تفوق هذا البحث في زياده عدد الرسائل الكليه المستلمه المقاسه بوحده الزمن والتقليل الكبير لعدد تصادمات الرسائل.

مهندس: محمد أبوبكر إبراهيم

تاريخ الميلاد: ٦/٩/١٩٨١

الجنسية: مصري

تاريخ التسجيل: ١/١٠/٢٠١٠

تاريخ المنح: ٢٠١٦

القسم: هندسه الالكترونيات و الاتصالات الكهربية

الدرجة: ماجستير العلوم

المشرفون: أ.د. حسام علي حسن فهمي
د. عمر احمد نصر

الممتحنون: أ.د. حسام علي حسن فهمي (المشرف الرئيسي)
د. تامر عبد المطلب البط (الممتحن الداخلي)
أ.د. سلوى حسين عبد الفتاح الرملي (الممتحن الخارجي استاذ بهندسة عين شمس)

عنوان الرسالة: **روكسي طريقة تلقائية لتوجيه البيانات في شبكات المركبات في طرق المدن المزدحمة**

الكلمات الدالة: الشبكات المخصصه للمركبات؛ بروتوكول اختيار المسار؛ اختيار المسار من مركبه الى مركبه؛ اختيار المسار على اساس الموقع؛ تفادي التصادمات

ملخص الرسالة: في هذا البحث نقدم برتوكول توجيه للشبكات المخصصه للمركبات "المستلم كوكيل" (ROXY) للتغلب على المحددات التي واجهت ابحاث اخرى. المستلم كوكيل يستعمل نظام التوجيه الجغرافي مع نظام اختيار القفزه التاليه بشكل انتهازي لزياده عدد الرسائل الكليه المستلمه المقاسه بوحده الزمن في الشبكات المخصصه للمركبات. الهدف الاساسي للمستلم كوكيل هو حل مشكله الاختناقات المتزايده في الشبكه وتصادمات الرسائل في حاله الشوارع المكتظه بالمركبات التي تستخدم الشبكات المخصصه للمركبات.
المستلم كوكيل يحسن النظام عن طريق معالجة ثلاثة قضايا. الاولى هي الانتفاع من الموارد المكانيه عن طريق استخدام هوائيين موجهيين في كل مركبه، احدهم موجه الى امام المركبه والاخر في الاتجاه الخلفي للمركبه. هذان الهوائيان يستخدمان لارسال الرسائل في اتجاهيين متعاكسيين في نفس الوقت وبهذا نستخدم وسط الانتقال بشكل اكثر فعاليه وزياده عدد الرسائل الكليه المستلمه المقاسه بوحده الزمن. الثانيه اختار المركبه التي سوف تقوم بتولي مهمه اعادا ارسال الرساله يكون بين المركبات المستلمه داخليا بدون تدخل المركبه المرسله او كيان مركزي. الثالثه والاخيره تقليل الوقت اللازم لارسال الرسائل عن طريق اقتراح خوارزميه جديده لتجنب اصطدام الرسائل التي تستفيد من الهوائيان الموجهان وشكل الشوارع في المدن. نتائج المحاكات تظهر تفوق هذا البحث في زياده عدد الرسائل الكليه المستلمه المقاسه بوحده الزمن والتقليل الكبير لعدد تصادمات الرسائل.

روكسي طريقة تلقائية لتوجيه البيانات في شبكات المركبات في طرق المدن المزدحمة

إعداد

محمد أبوبكر إبراهيم

رسالة مقدمة إلى كلية الهندسة – جامعة القاهرة

كجزء من متطلبات الحصول على درجة ماجستير العلوم

في

هندسه الالكترونيات و الاتصالات الكهربية

يعتمد من لجنة الممتحنين:

_____

الأستاذ الدكتور: حسام علي حسن فهمي المشرف الرئيسي

_____

دكتور : تامر عبد المطلب البط الممتحن الداخلي

_____

الأستاذة الدكتورة: سلوى حسين عبد الفتاح الرملي الممتحن الخارجي

(استاذة بهندسة عين شمس)

كليـة الهندســة – جامعـة القاهـرة

الجيـزة – جمهوريـة مصـر العربيـة

٢٠١٦

روكسي طريقة تلقائية لتوجيه البيانات في شبكات المركبات في طرق المدن المزدحمة

إعداد

محمد أبوبكر إبراهيم

رسالة مقدمة إلى كلية الهندسة – جامعة القاهرة

كجزء من متطلبات الحصول على درجة ماجستير العلوم

في

هندسه الالكترونيات و الاتصالات الكهربية

تحت إشراف

أ.د. حسام علي حسن فهمي

.....

أستاذ دكتور

هندسه الالكترونيات و الاتصالات الكهربية

كلية الهندسة – جامعة القاهرة

د. عمر احمد نصر

.....

أستاذ مساعد

هندسه الالكترونيات و الاتصالات الكهربية

كلية الهندسة – جامعة القاهرة

كليـة الهندســة – جامعـة القاهـرة

الجيـزة – جمهوريـة مصـر العربيـة

٢٠١٦

روكسي طريقة تلقائية لتوجيه البيانات في شبكات المركبات في طرق المدن المزدحمة

إعداد

محمد أبوبكر إبراهيم

رسالة مقدمة إلى كلية الهندسة – جامعة القاهرة

كجزء من متطلبات الحصول على درجة ماجستير العلوم

في

هندسه الالكترونيات و الاتصالات الكهربية

كليـة الهندســة – جامعـة القاهـرة

الجيـزة – جمهوريـة مصـر العربيـة

٢٠١٦