University Of Bahrain

College of Engineering

Department of Electrical & Electronics Engineering

Adaptive Broadcasting Schemes in Vehicular Ad-Hoc Networks

A Thesis Submitted in Partial Fulfillment of the Requirements for the Ph.D. Degree in Electrical Engineering

Prepared by:

Omar Bani Fayyad

20150041

Supervised by:

Mohab Mangoud

Professor

University of Bahrain

Kingdom of Bahrain

May 2023

UNIVERSITY OF BAHRAIN

College of Engineering



Department of Electrical & Electronics Engineering

Adaptive Broadcasting Schemes in Vehicular Ad-Hoc Networks

A Thesis Submitted in Partial Fulfillment of the Requirements for the Ph.D. Degree in Electrical Engineering

Prepared by:

Omar Bani Fayyad

20150041

Supervised by:

Mohab Mangoud

Professor

University of Bahrain

Kingdom of Bahrain

May 2023

Thesis Discussion Committee Decision

Defense committee's approval page

University of Bahrain

College of Engineering

Department of Electrical & Electronics Engineering

PhD Degree in Electrical Engineering

Student Omar Yousef Ali Bani Fayyad with ID number 20150041 defended his thesis which he fulfilled the requirement for the degree of PhD in Electrical & Electronics Engineering entitled Adaptive Broadcasting Schemes in Vehicular Ad-Hoc Networks on this day of May 2, 2023.

The committee recommends awarding PhD Degree with the Grade of: A

(Defense Committee)

Name		Signature
Prof. Mohab Mangoud	Supervisor	
Prof. Gheith Abandah	External Examiner	
Prof. Ebrahim Mattar	Internal Examiner	

Academic department's stamp

Declaration

I hereby declare that this work is the result of my own investigation and that it has not already been accepted in substance of any degree, nor is it being currently submitted for any degree.

Signature:  Name: Omar Bani Fayyad Date: 24/06/2023

Abstract

Vehicular ad-hoc network (VANET) is considered a promising solution in intelligent transport system (ITS) and smart cities (SC) to enhance road safety, traffic efficiency, driver, and passenger's comfort. Numerous studies have been conducted in this field to provide a reliable and bandwidth-efficient broadcast protocols for IEEE 802.11p VANET. The main problem that will be addressed in this thesis is the deficiency of the existing VANET broadcasting protocols and channel access schemes that leads to broadcast storm, message collisions, and consequently, a waste of channel bandwidth and decrease packet delivery ratio (PDR). VANET utilizes a traditional broadcasting protocol that employs flooding techniques to disseminate received messages to vehicles within communication range. Moreover, VANET uses a carrier sense multiple access with collision avoidance (CSMA/CA) to access channel resources, utilizing a slotted binary exponential backoff (BEB) mechanism. It is based on a random backoff value called contention window (CW) to access the channel. The CW value is doubled after the unsuccessful transmission and rest after successful transmission. The proper setting of CW value has a significant impact on the efficiency of VANET network.

This thesis proposes a fuzzy logic inference to address the issues of broadcast storms and high message collisions. The protocol intelligently selects the next hop relay vehicle, considering factors such distance, mobility, and received signal strength indication (RSSI).

The thesis has also proposed new and improved artificial intelligence (AI) based broadcasting protocols and channel access schemes to improve the performance of the BEB algorithm and enhance the overall performance of VANETs. Three deep reinforcement learning (DRL) algorithms are proposed, Q-learning, deep Q-network (DQN) and actor-critic. Q-learning is a simple and effective algorithm that can be used to learn the optimal action-value function for a given state. DQN is a more complex algorithm that uses a deep neural network to approximate the action-value function, and Actor-critic is a hybrid algorithm that combines elements of both Q-learning and policy-gradient methods. By using DRL, an intelligent learner can dynamically and independently control the CW value to achieve its objectives. To apply RL and DRL algorithms in VANET environment, a hybrid framework called NS3-gym is utilized.

The proposed algorithms were implemented, tested, and evaluated in NS3-Gym; fuzzy logic has significant improvement by reducing rebroadcast packets compared to the conventional approach. The proposed QL and DRL algorithms learns various VANET environments and enhance their performance compared to conventional channel access protocol.

The acquired new insights on the network performance of the proposed algorithms can offer precise guidelines for efficient designs of practical, reliable, fair V2V communication systems for sparse and dense environments. These results can potentially have a significant impact on a wide range of related VANET applications.

Dedication

To my parents and family

Acknowledgments

In the name of Allah, the most Gracious, the most Merciful. All praises to Allah for the strength for reaching this stage.

I would like to express my sincere thanks and gratitude to my supervisor Prof. Mohab Mangoud for his patience, encouragement, continuous support, dedication, and guidance through my PhD journey, as well as the time taken to teach me how to conduct research and writing thesis. His insightful reviews, comments, feedback helped me a lot to drive my research in the right direction starting from selecting the research topic along to evaluating the research results.

I would like to express my gratitude to the members of my thesis committee. Additionally, I extend my thanks to all the faculty members in the engineering college who imparted their knowledge and expertise to me during my PhD courses, as well as to my fellow PhD colleagues.

Special thanks to my parents, for supporting me through this journey. Your inspiration, blessing, encouragement, and continuous support have helped me to be who I am now. Most importantly, my wife and daughters Taj, Joud, Zeina and Dahab, warmest appreciation and thanks for their patience, encouragement, and sacrifices through this journey. They missed a lot of opportunities and time we could have had together.

Finally, my thanks and appreciations to all my brothers, sisters, relatives, friends, colleagues, who supported me during the journey.

Table of Contents

Thesis Discussion Committee Decision ..... **A**
Declaration ..... **A**
Abstract ..... **C**
Dedication ..... **E**
Acknowledgments ..... **F**
Table of Contents ..... **G**
List of Tables ..... **J**
List of Figures ..... **K**
Abbreviations ..... **N**
Chapter 1 Introduction **2**
1.1 Introduction 2
1.2 Research Problem 4
1.3 Research Questions 6
1.4 Research Objectives 6
1.5 Research Significance 8
1.6 Research Scope 8
1.7 Research Terminology 9
1.8 Dissertation Outline 10
Chapter 2 Theoretical Framework and Literature Review **12**
2.1 Introduction to Vehicular Ad-hoc Networks 12
2.1.1 DSRC Standards 13
2.1.2 IEEE 802.11P Medium Access Control 16
2.2 Reinforcement Learning 21
2.2.1 Reinforcement Learning in Markovian Environment 23
2.3 Deep Reinforcement Learning 25
2.3.1 Neural Networks 26
2.4 Literature Review 27
2.4.1 Multi Hop Broadcasting in VANET 27
2.4.2 IEEE802.11p MAC Protocol in Vehicular Ad Hoc Networks 30
2.4.3 Reinforcement Learning for Contention Window Adaptation 32
2.5 Conclusion 33

Chapter 3 Smart Fuzzy Inference-based Multi-hop Broadcast Protocol for VANET 35
3.1 Introduction 35
3.2 Developing Fuzzy Expert Protocol 37
3.3 Design of Fuzzy Logic Inference 39
3.3.1 Fuzzification 40
3.3.2 Elicit and construct Fuzzy rules 42
3.3.3 Defuzzification 43
3.4 Simulation and Evaluation 44
3.5 Conclusion 51
Chapter 4 Simulated Reinforcement Learning Environment for VANETs ... 54
4.1 Introduction 54
4.2 NS3-gym framework 55
4.3 Ns3-gym System Architecture 56
4.3.1 NS3 simulation 57
4.3.2 Open AI Gym 58
4.3.3 Ns3-Gym middleware 59
4.4 Ns3-gym implementation 60
4.4.1 Gateway Component 60
4.4.2 Proxy Component 69
4.5 A Q-Learning-Based Contention Window Adjustment in VANETs 72
4.5.1 Introduction 72
4.5.2 Q-Learning MAC Protocol Design 75
4.5.3 Performance Evaluation 80
4.6 Conclusion 90
Chapter 5 Optimizing Contention Window in VANET using DQN and Actor-Critic 94
5.1 Introduction 94
5.2 Deep Reinforcement Learning (DRL) in VANET 100
5.2.1 Applying DRL in VANETs 101
5.3 Performance Evaluation 122
5.3.1 Simulation Setup 122
5.3.2 Simulation Results 124

5.4 Conclusion 129
Chapter 6 Conclusion 132
6.1 Contribution and Conclusion 132
6.2 Recommendation and Future Work 134
References 135
Publications and Participations 145
الملخص 147

List of Tables

Table 3-1 Fuzzy rules considered for the proposed protocol 43
Table 3-2 Simulation Parameters 45
Table 3-3 Synthetic Highway scenario parameters 45
Table 4-1 Simulation parameters 81
Table 4-2 Q-Learning parameters 81
Table 5-1 VANET PHY and MAC parameters 122
Table 5-2 Road traffic parameters 122
Table 5-3 DRL parameters. 123

List of Figures

Figure 2-1: DSRC Frequency band 13
Figure 2-2 WAVE architecture 14
Figure 2-3 CSMA/CA in broadcast transmission 19
Figure 2-4 CSMA/CA in unicast transmission 20
Figure 2-5 The agent environment interaction in Reinforcement Learning 22
Figure 3-1 Block Diagram for fuzzy logic controller 40
Figure 3-2 Input Variable "Distance" 41
Figure 3-3 Input Variable "Mobility" 41
Figure 3-4 Input membership function for RSSI 42
Figure 3-5 output membership function (Relay fitness value) 42
Figure 3-6 Reachability versus number of vehicles for fuzzy based protocol, WAVE and DTM Protocol 47
Figure 3-7 Reachability versus number of hops count for fuzzy based protocol, WAVE and DTM Protocol 47
Figure 3-8 Reachability versus transmission range for fuzzy based protocol, WAVE and DTM Protocol 47
Figure 3-9 Byte Sent Per Covered node versus number of vehicles for Fuzzy based protocol, WAVE and DTM protocols 48
Figure 3-10 Byte Sent Per Covered node versus number of hops count for Fuzzy based protocol, WAVE and DTM protocols 49
Figure 3-11 Byte Sent Per Covered node versus transmission range for Fuzzy based protocol, WAVE and DTM protocols 49
Figure 3-12 Rebroadcast Per Covered node versus Number of vehicles for Fuzzy based protocol, WAVE and DTM protocols 50
Figure 3-13 Rebroadcast Per Covered node versus hops count for Fuzzy based protocol, WAVE and DTM protocols 50
Figure 3-14 Rebroadcast Per Covered node versus transmission range for Fuzzy based protocol, WAVE and DTM protocols 51
Figure 4-1 Figure 3.1 ns3-gym system architecture 57
Figure 4-2 Open Gym Interface 61
Figure 4-3 standard set of functions 62
Figure 4-4 Observation space function 63
Figure 4-5 Action space function 63
Figure 4-6 Nodes creation and simulation time 64
Figure 4-7 MyGetGameOver function 65
Figure 4-8 Main function 65
Figure 4-9 GetObservation function 66
Figure 4-10 MyGetReward function 67
Figure 4-11 MyExecuteAction function 68
Figure 4-12 SetCW function 69
Figure 4-13 Gym libraries 70

Figure 4-14 Simulation arguments 70
Figure 4-15 episodes iteration 72
Figure 4-16 Reinforcement learning based CW adjustment in V2V communication 77
Figure 4-17 System throughput versus vehicles number with packet size is 200 bytes ... 82
Figure 4-18 System throughput versus vehicles number with packet size is 500 bytes ... 83
Figure 4-19 Throughput versus different distance ranges with packet size is 200 bytes for 75 vehicles 84
Figure 4-20 Throughput versus different distance ranges with packet size is 200 bytes for 75 vehicles 84
Figure 4-21: Throughput versus different distance ranges with packet size is 200 bytes for 150 vehicles 85
Figure 4-22 throughput versus different distance ranges with packet size is 500 bytes for 25 vehicles 85
Figure 4-23 Throughput versus different distance ranges with packet size is 500 bytes for 75 vehicles 86
Figure 4-24 Throughput versus different distance ranges with packet size is 500 bytes for 150 vehicles 86
Figure 4-25 PDR versus different distance ranges with packet size is 200 bytes for 25 vehicles 87
Figure 4-26 PDR versus different distance ranges with packet size is 200 bytes for 75 vehicles 88
Figure 4-27 PDR versus different distance ranges with packet size is 200 bytes for 150 vehicles 88
Figure 4-28 PDR versus different distance ranges with packet size is 500 bytes for 25 vehicles 89
Figure 4-29 PDR versus different distance ranges with packet size is 500 bytes for 75 vehicles 89
Figure 4-30 PDR versus different distance ranges with packet size is 500 bytes for 150 vehicles 90
Figure 5-1 Example of exponential increase of CW in CSMA/CA. Taken from IEEE Std 802.11-2007. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (Physical) specification 96
Figure 5-2 Backoff Procedure in CSMA/CA. Taken from IEEE Std 802.11-2007. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (Physical) specification 97
Figure 5-3 CSMA/CA managing channel access in unicast transmission among Vehicles 98
Figure 5-4 Agent-environment interaction in MDP 102
Figure 5-5 VANET Topology 105
Figure 5-6 Simple DQN diagram 106
Figure 5-7 Q-learning tuple 111
Figure 5-8 Buffer Replay technique 111
Figure 5-9 NN architecture for DQN 113
Figure 5-10 the repeating module in RNN with single layer 114
Figure 5-11 The LSTM NN architecture 115

Figure 5-12 Stochastic policy with respect to state 116
Figure 5-13 Stochastic policy with respect to state (s) and action (a2) 117
Figure 5-14 Actor-Critic Architecture 118
Figure 5-15 VANET topology 119
Figure 5-16 Network Throughput in scenario 1 125
Figure 5-17 Instantaneous network throughput in scenario 1 for 50 vehicles 126
Figure 5-18 Chosen CW in scenario 1 for 30 vehicles 126
Figure 5-19 Network Throughput in scenario 2 127
Figure 5-20 Instantaneous Network Throughput in scenario 2 128
Figure 5-21 Chosen CW under scenario 2 for 30 vehicles 129

Abbreviations

ACK	Acknowledgment
AIFS	Arbitration Inter-Frame Spacing
API	Application Programming interface
BEB	Binary Exponential Backoff
BL	Bayesian Learning
BSS	Basic Service Set
CCH	Control Channel
CCHI	Control Channel Interval
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CW	Contention Window
DCF	Distributed Coordination function
DIFS	DCF Inter-Frame Space
DQN	Deep Q- Network
DRL	Deep reinforcement Learning
DSRC	Dedicate Short Range Communication
EDCA	Enhanced Distributed Channel Access
FNN	Feed Forward Neural Network
ICA	Independent Component Analysis
IFS	Inter-Frame Spacing
ITS	Intelligent Transport System
KNN	K-Nearest Neighbor
ML	Machine Learning
NN	Neural Network
NS3	Network Simulator 3
MANETs	Mobile Ad-hoc Networks

MAC	Medium Access Control
MDP	Markov Decision Process
POMDP	Partial Observable Markov Decision Process
OBU	On-Board Unit
OCB	Outside the Context of A BSS
PCA	Principal Component Analysis
PDR	Packet Delivery Ration
PHY	Physical Layer
QL	Quality Learning
QoS	Quality of Service
RL	Reinforcement Learning
RNN	Recurrent Neural Network
RSSI	Received Signal Strength Indicator
RSU	Road Side Units
SC	Smart Cities
SCH	Service Channel
SCHI	Service Channel Interval
SIFS	Short Interframe Space
SVM	Support Vector Machine
UTC	Coordinated Universal Time
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
V2X	Vehicle to Everything
VANETs	Vehicle Ad Hoc Networks
WAVE	Wireless Access Vehicular Environment
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
WSMP	WAVE Short Message Protocol

Chapter 1

Introduction

Chapter 1 Introduction

1.1 Introduction

The road accidents are becoming the major world economic and social problem. Many reports show that the number of deaths, injuries and total cost of accidents are increasing. The recent advances in integrated circuits, microelectronics (MEMS) and wireless communications, especially in wireless networks, enabled the development of new standards that help the people to make their life easier, safer, and comfortable by the ease of implementing the wireless technologies and their application.

Vehicle Ad-Hoc Network (VANET) is a subset of Mobile Ad-hoc Network (MANET) that provide a communication among mobile vehicles using a dedicated short-range communication (DSRC). IEEE 802.11p standard is produced to support wireless communication among vehicles to broadcast safety message, this standard referred as wireless access in vehicular environment (WAVE).

VANET is a new promising solution in wireless communication to enhance road safety, traffic efficiency, driver and passenger's comfort and many other smart applications by enabling vehicles to communicate with other vehicles directly forming Vehicle to Vehicle communication (V2V) or communicate with Roadside Unit (RSU) forming vehicle to infrastructure communication (V2I) (Olariu & Weigle, 2009) . Through these communications, vehicles exchange messages containing real time information about vehicles' position, speed, direction, acceleration...etc. This information will help the drivers to know the status of the network and predict vehicles' movement to take early action about any hazardous situation and

avoid them. Another example, in V2I the Road Side Unit (RSU) needs to advertise location-based services (gas station, restaurants, advertisements and so on). This needs to be broadcast to the nearby vehicles to inform the driver about their services. These features and others made VANETs a key pillar for developing smart cities (SC) and improving the development of the future autonomous vehicles.

Among different types of VANETs' applications, the road safety is considered the most urgent problem that needs to be addressed and researched to save humans' life and money. When an accident happens on a road, a safety or a warning message should be disseminated immediately to alert all the incoming and speedy vehicles that might be affected by this accident. Even the vehicles that seem to be very far from the accident, they will reach the danger very soon. In the highway, when you are driving in high speed, milliseconds will be a matter to avoid any hazardous or danger situation. For example, In March 2008, at least 4 people were killed and more than 300 injured from the shock of a 200-car pile-up on the Dubai-Abu Dhabi highway. Another huge car pile-up on the same highway in April 2011 involving 127 vehicles, causing at least one death and 61 injured according to Dubai's police. When the first accident occurred there should be a technique to disseminate alarms to all incoming vehicles about the accident, so driver can take an immediate decision to avoid such danger event and saving humans' life and money.

Broadcasting is considered the simple and suitable communication mechanism to disseminate warning messages across the network through flooding the message to all vehicles in the communication range. In Multi-hop broadcasting, the message broadcasting is extended to the received vehicles that have the responsibility to flood the messages as soon as it is

received to ensure the warning message will be received by all vehicles that might be affected. However, the traditional way of broadcasting where every single vehicle rebroadcasts the message is inefficient because of the following reasons.

1. In the dense network, the broadcast protocol will disseminate the same message over and over. This will waste network resources and makes broadcasting not scaled with high density network (Panichpapiboon & Pattara-atikom, 2012).
2. A broadcast storm occurs when there are many broadcast messages disseminating across the network (Wisitpongphan, et al., 2007).
3. There will be a high number of vehicles are contending for accessing wireless channel at the same time, this will result in a long waiting time at the MAC layer and consequently increase end-to-end delay time (Wu, Chen, Ji, Ohzahata, & Kato, 2015)
4. Message collision becomes more severe problem since large number of vehicles may rebroadcast at the same time. This will have a negative impact on the packet delivery ratio (Panichpapiboon & Pattara-atikom, 2012).
5. high vehicles mobility in VANETs and dynamic channel conditions cause frequent topology changes in VANETs. It is making difficult to efficiently disseminate messages among vehicles (Xiaolan, Zhi, Wenlong, & Mojtaba, 2017) (Wu, Chen, Ji, Ohzahata, & Kato, 2015).
6. Inefficient MAC access sharing protocol that does not adapt to the density of vehicular traffic in broadcast and unicast VANET communications (Du, Wang, & Zhang, 2020) (Karabulut, Shah, & Ilhan, 2018) (Rawat D. , Popescu, Yan, & Olariu, 2011).

1.2 Research Problem

VANETs Broadcasting is the process of sending a message to all the vehicles in a specific communication range. The purpose of broadcasting is to inform all the vehicles about a new event or certain action. Designing an efficient broadcasting protocol is the key factor for having a successful deployment for VANET services. Most of the VANETs services including road safety, traffic efficiency and infotainment services are heavily depending on broadcasting messages across the VANETs. For example, in V2V communication the vehicles must broadcast messages among the vehicles to inform them about any sudden event like vehicle collision, sudden brakes, etc.

Broadcasting requires every vehicle to rebroadcast messages to its neighbors once it receives them. in case of accidents or traffic jam, it is easy to cause a broadcast storm and increase number of messages' collision because of huge number of messages' rebroadcasting among vehicles which finally leads to network performance degradation in terms of bandwidth utilization, packets' delivery ratio, collision rate and transmission delay.

Also, VANET is using MAC channel access contention mechanism to send data. IEEE 802.11p employs CSMA/CA with BEB mechanism for contention-based access channel to reduce the collision probability among multiple contending nodes that trying to access the medium to send their data. CW is a parameter that is used by CSMA/ CA to randomize the time to access channel among vehicles that wish to transmit data to avoid accessing channel simultaneously which leads to messages' collision that will negatively affect network performance.

The research problem is to find an adaptive technique in VANET to solve the following issues:

- Broadcast Storm and message collisions.
- Reliability issues: broadcast protocol should disseminate broadcast message to all intended vehicles in that area.

- Network Throughput: enhance VANET's throughput.
- PDR: minimize message collision which will enhance PDR.
- Scalability issue: The adaptive broadcast protocol should efficiently deal with several situations and scenarios.

1.3 Research Questions

The research questions can be expressed as follows:

1. How to deliver the broadcast message to all vehicles that might be affected by road's conditions with less number of rebroadcasting?
2. How to avoid broadcast storm and message's collisions among vehicles in sparse and dense traffic conditions?
3. How to deal with packet loss in VANETs?
4. How to deal with temporarily network disconnection?
5. Is Reinforcement Learning (RL) considering the suitable technique to efficiently utilize the bandwidth and enhance VANET performance?

1.4 Research Objectives

The main goal of this project is to design a reliable, scalable and efficient protocols in VANETs that can enhance network's performance. There are many objectives to support this goal that are summarized as follows:

- Using fuzzy logic inference in order to choose next hop relay vehicles to make sure that the broadcast messages will be received by all vehicles that might be affected with less number of broadcasting.

- Mitigating broadcast storm and messages collision by reducing number of vehicles that responsible for messages rebroadcasting.
- To be scalable, able to deal with both sparse and dense networks.
- Providing an efficient way for dealing with packet loss with purpose of accurately receiving necessary information on time in safety applications.
- Using RL to adjust CW value in VANET communications to improve PDR in dense network, thus utilize the channel bandwidth efficiently. This will improve network's performance in terms of increasing throughput, save bandwidth and enhance PDR.

1.5 Research Significance

This research is important because it investigates the following genuine issues:

1. Designing an adaptive and efficient broadcasting protocol is the key factor for having a successful deployment for VANET services.
2. Reduce number of accidents on the road and consequently save humans live and money.
3. improving the development of the future autonomous vehicles and smart cities.
4. Fuzzy logic inference will be utilized to minimize number of vehicles that rebroadcast messages and consequently minimize redundant messages that will cause broadcast storm.
5. RL will be used to adjust CW value in CSMA/ CA protocol, help to optimize access channel contention mechanism. The outcome of this work would be an adaptive MAC protocol based on RL agents.

1.6 Research Scope

Although the mapping of subsurface features was enhanced exponentially in the last years. Still, the final products are vector or raster data, where the attributes, metadata, and other information (such as the accuracy of collected and analyzed data) might be done later by data entry or by the user manually. Besides, the traditional geophysical methods concentrate on the techniques and depend heavily on the operator and analysts' experience.

Therefore, the scope of this research is to develop an optimum approach for mapping the subsurface feature from Geomatics and Remote Sensing

perspectives, to help the user to know more about the available data and their reliability.

1.7 Research Terminology

CSMA/CA: a MAC layer protocol that regulate vehicles' access to the network channel for sending data.

Binary Exponential Backoff (BEB): a mechanism used in CSMA/CA protocol to randomize vehicles' access into the channel to avoid collision probability.

Contention Window (CW): a period of time that the vehicles have to wait before accessing the channel medium.

Reinforcement Learning: an optimization framework to learn how to control and solve complex problems. It is a branch of ML where an intelligent agent takes an action, and the environment returns observations or state of the system and rewards.

Q-learning is a classic reinforcement learning algorithm that works in any environment and learn the environment's behavior through trial-and-error interaction. It is a model free algorithm that works by estimating Q- value which is the value of future rewards if the Q-learning agent takes an action in particular state.

Deep Q-Network: a model-free reinforcement learning algorithm approximate Q-learning using Neural Network (NN) proposed by DeepMind (Mnih, et al., 2015).

1.8 Dissertation Outline

The thesis consists of six chapters. Chapter 1 outlines the introduction to the study. Chapter 2 discusses the theoretical framework and provides a literature review of IEEE 802.11p VANET network broadcasting, MAC layer protocol and Reinforcement Learning in VANET. Chapter 3 introduced fuzzy logic to select best relay vehicle to broadcast VANET messages. Chapter 4 introduces NS3-gym simulation along with Q-learning algorithm to adjust CW in VANET broadcasting scenario. Chapter 5 explained Deep Q-network and Actor-Critic network as deep reinforcement learning algorithms to be used to optimize CW in VANET. Finally, Chapter 6 provides the conclusions and recommendations.

Chapter 2

Theoretical Framework

and

Literature Review

Chapter 2 Theoretical Framework and Literature Review

This Chapter will provide a theoretical overview about VANETs in terms of standards, architecture, applications, communications models, and MAC layers protocols. Furthermore, it will provide an overview of the various types of ML utilized in the research, followed by a review of relevant literature related to the research topics.

2.1 Introduction to Vehicular Ad-hoc Networks

Wireless networks are classified into two types, infrastructure, and Ad-hoc wireless networks (Royer & Toh, 1999). The first type of wireless networks needs to build an infrastructure by setup base stations in specific location that provide connectivity to mobile nodes. While in mobile Ad-hoc Network (MANET), the mobile node will act as a mobile node and router at the same time for transmission without implementing base stations. VANET is a subset of MANET that provide a communication among mobile vehicles using a dedicated short-range communication (DSRC). VANETs are considered a promising ad hoc solution for roads' safety and non-safety applications in ITS and SC (Moloisane, Malekian, & Bogatinoska, 2017). In VANET, all vehicles are equipped with different types of sensors and On-Board Unit (OBU) to allow wireless communications among vehicles.

2.1.1 DSRC Standards

DSRC is a wireless communication technology used for ITS and SC. It is a short to medium range communication service licensed at 5.9 GHz with a 75-MHz spectrum, which is divided into seven 10-MHz channels and a 5-MHz guard band as shown in Figure 2-1 DSRC Frequency band Figure 2-1.

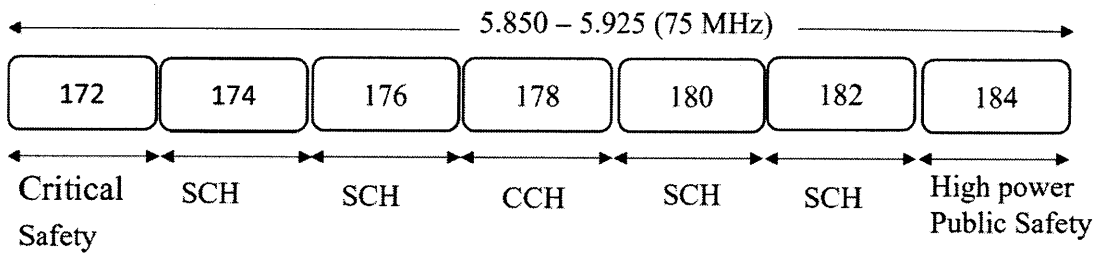


Figure 2-1 DSRC Frequency band

The control channel (CCH) will be used for safety applications, whereas the other six channels, called service channels (SCHs), will be used for infotainment or commercial applications to make this technology more cost effective (Hafeez, Zhao, Ma, & Mark, 2013). Under the DSRC, Wireless Access in Vehicular Environment (WAVE) is included to describe set of standards that govern VANET communication as shown in Figure 2-2.

The WAVE architecture is a set of standards that working in different layers governing and organizing communications among vehicles.

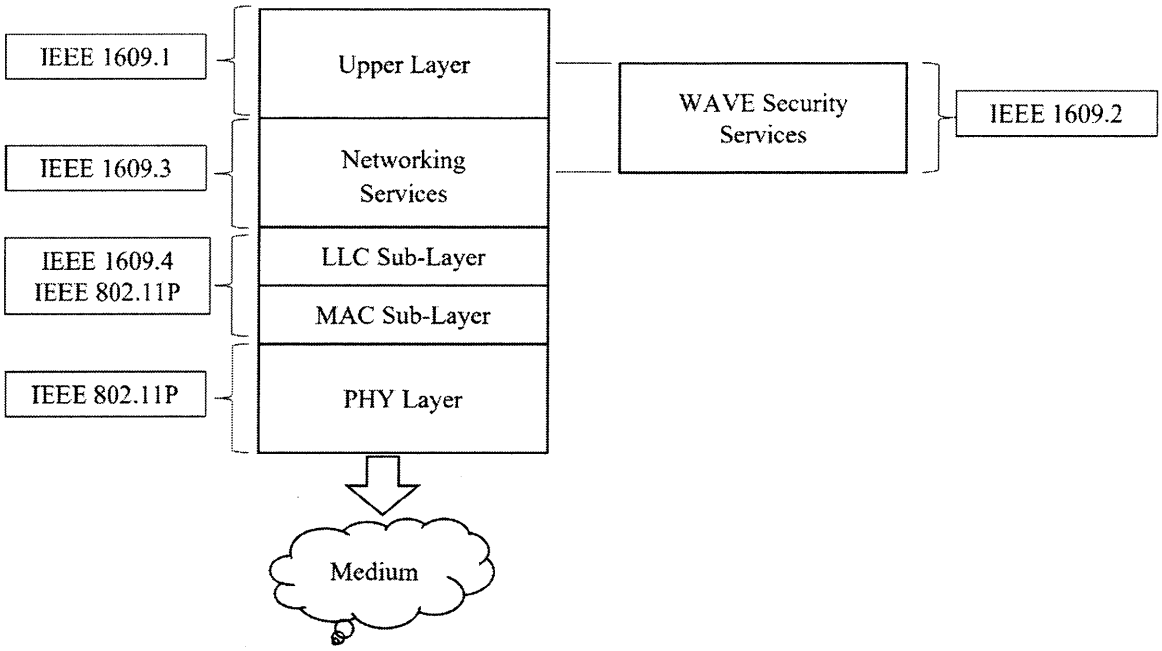


Figure 2-2 WAVE architecture

2.1.1.1 IEEE 802.11p

IEEE 802.11p is an amendment to Wireless Local Area Network (WLAN) standard and known as Wireless Access in Vehicular Environment (WAVE) that define physical MAC layers (IEEE 802.11p, 2010). It has a set of useful features for making communications among mobile vehicles in the vehicular environment where highly dynamic, rapid changes, low latency and operation in a reserved frequency range (Arena, Pau, & Severino, 2020). The operating frequency of IEEE 802.11p is 5.85 -5.925 GHz band. In this range, there is a channel called CCH that carries safety messages and beacons. The other six SCHs used to exchange safety and non-safety messages (Hassan, Vu, & Sakurai, 2011). The channel timing is 100ms long, consisting of two equal length intervals alternating between CCH Interval (CCHI) and SCH Interval (SCHI). During the CCHI, all vehicles tune to the CCH frequency for safety

and control messages and in the SCHI, all vehicles switch into SCH frequency. There is 4ms a guard time to consider radio switching delay and inaccuracy timing among WAVE vehicles devices (CHOE, et al., 2020). All vehicles use Coordinated Universal Time (UTC) for channels coordination utilizing global satellite system (Arena, Pau, & Severino, 2020) .

The physical layer of IEEE 802.11p is similar to IEEE 802.11a with some modifications (Saponara & Gagliardi, 2018) (Anwar, Kulkarni, Augustin, Franchi, & Fettweis, 2018) to support highly dynamic and mobile environment, low overhead and minimize delay. As a result, IEEE 802.11p allows the vehicles to transmit data frame without being a member of Basic Service Set (BSS) by defining a new mode called Outside the Context BSS (OCB). This mode enables vehicles to communicate directly over the wireless medium, avoiding latency associated BSS association, authentication, de-authentication, data confidentiality (IEEE 802.11p, 2010).

2.1.1.2 IEEE 1609 standards

The WAVE architecture as shown in Figure 2-2 consists of different sub-standards that provide services on the layers above physical layer. These standards together provide secure wireless communications among vehicles, resource management, channel switching and other application services.

- IEEE 1609.1: specifies the services of the WAVE resource manager application.
- IEEE 1609.2: defines secure communications among vehicles.

- IEEE 1609.3: Defines network layer services including addressing, routing and secure WAVE messages. It introduces WAVE Short Message Protocol (WSMP), to support sending safety messages that require high priority and less latency.
- IEEE 1609.4: used for multi-channel operations, where provide an enhancement to the IEEE 80.211 MAC layer to switch between CCH and SCH channels.

2.1.2 IEEE 802.11P Medium Access Control

The MAC protocol plays a critical role for having a reliable and an efficient communication channel, minimize packet collisions, Quality of service (QoS) provision and provide high throughput for safety and non-safety packet transmission (Dang, Hong, Lee, & Huh, 2014).

IEEE 802.11p standard uses Enhanced Distributed Channel access (EDCA) (IEEE 802.11 Working Group, 2005) which specifies the distributed contention-based channel access scheme to support different level of equality of services (QoS). EDCA is considered an enhancement variant of legacy Distributed Coordination function (DCF) scheme to provide prioritized services. Both schemes are relying on Carrier Sense Multiple Access with collision Avoidance (CSMA/CA) (Kim & Lee, 1999) with a slotted BEB mechanism for contention-based access channel to with the goal of minimize the collision probability among multiple contending nodes by allowing the nodes to transmit the data based on the channel status. The main differences between DCF and EDCA are the Inter-Frame Spacing (IFS) which is the time interval between data frames and the size of CW (Nguyen, et al., 2018) , EDCA is used an Arbitrary Interframe space (AIFS), while DCF interframe space (DIFS) is used in DCF scheme.

2.1.2.1 The CSMA/CA Algorithm

Figure 2-3 and Figure 2-4 (ETSI, 2019) show the flowchart for CSMA/CA algorithm in broadcast and unicast transmission, which works as follows:

- Once the vehicle has the frame ready in the queue, the vehicle senses the channel medium before commence transmission to make sure the medium is idle, and no-one is transmitting.
- If the vehicle finds the wireless medium is idle for DIFS/AIFS period, the vehicle waits for a random time before start transmitting the data. This random time called backoff which is based on BEB mechanism, defined by CW parameter. The BEB is introducing kind of asynchronization among vehicles when their AIFS/DIFS expired simultaneously. When the backoff timer reaches 0, the vehicle starts transmitting immediately.
- If the vehicle senses a busy channel before DIFS/AIFS period expiration, the vehicle defers its transmission until the channel is idle for the duration of DIFS/AIFS.
- In the unicast transmission, when the data received correctly, the transmitting vehicle waits for acknowledgment (ACK) message. If an ACK is received successfully by the transmitting vehicle, this is an indication for a successful transmission.
- If an ACK is not received by the transmitting vehicle in due time, the transmission will be considered failed and the vehicle starts over the CSMA/CA (Unless the maximum number of

retransmissions has been exceeded, the CW value will be incremented and starts the BEB again).

2.1.2.2 Binary Exponential Backoff Mechanism

The random timer generated by BEB is bounded by the CW value. The vehicle randomly draws an integer backoff from uniform distribution (0, CW), where the initial CW value equals CW_{min}, and starts count down the backoff timer before commencing the transmission. The backoff timer will be decremented only when the medium is free.

In the broadcast transmission as shown in Figure 2-3, where no ACK message is transmitted by the receiver, the backoff mechanism will be invoked only one time for packet transmission and the vehicle draws an integer backoff number from distribution (0, CW) where CW will be set to CW_{min} and it will not be incremented.

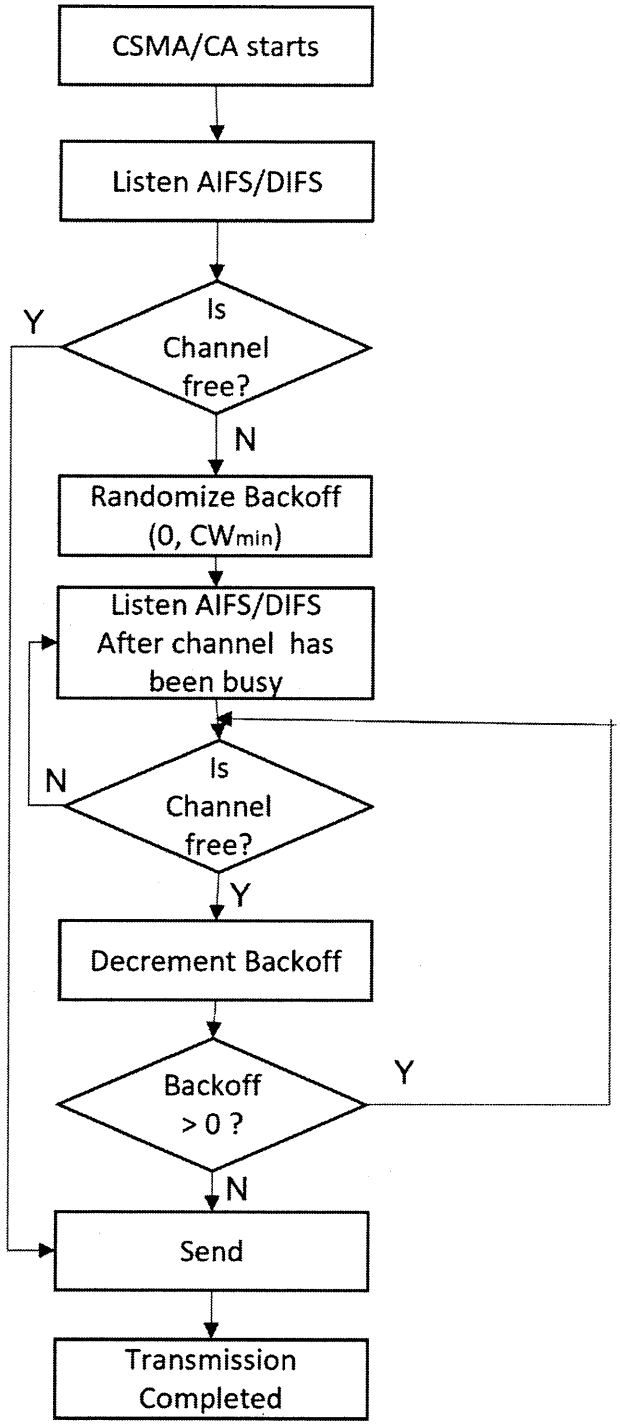


Figure 2-3 CSMA/CA in broadcast transmission

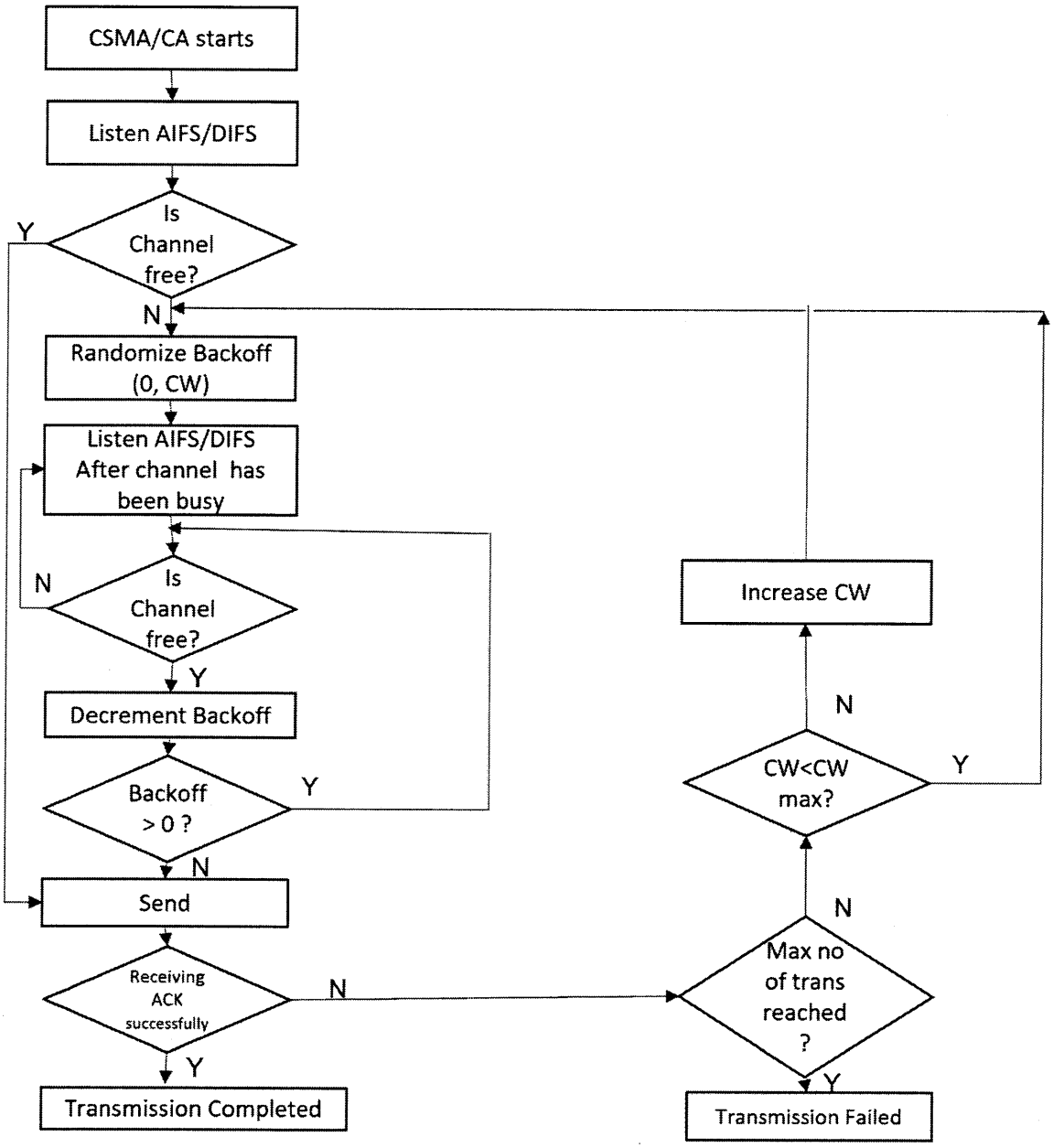


Figure 2-4 CSMA/CA in unicast transmission

In unicast transmission as shown in Figure 2-4, the vehicle draws randomly draws an integer backoff from uniform distribution (0, CW), where the initial CW value equals CW_{min}, and stars count down the backoff timers before transmitting the packet same as explained in the broadcast transmission. The BEB mechanism is adapting the CW value in the unicast

transmission, when the vehicle failed to transmit the message because of high traffic density, the BEB will double the CW value in the second attempt. The vehicle will draw randomly an integer backoff from new distribution (0, $2*CW_{min}$). every time the vehicle fails to transmit the packet, the CW value will be double until either reaching the CW_{max} or the maximum number of retransmissions has been exceeded.

2.2 Reinforcement Learning

RL is a branch of Machine Learning (ML) to learn control strategies and methodologies based on interaction with different types of environments. RL is a framework for learning the environment by direct interaction and build-up the knowledge about the system to use it to solve or optimize complex systems. It is a class of learning inspired by animal learning psychology (Sutton & Barto, 2018) and implemented on machine, this is what the animals do through trial and error, experience, or positive and negative feedback from environment interaction to form intelligent decision-making.

In RL as shown in Figure 2-5, the agent or the learner interacts with the environment or the world through a set of actions and observe how the environment reacts to different types of actions at each time step. Then, observe the state of the system at each time step and use this knowledge or experience to change actions trying to maximize the rewards, the current or the future rewards.

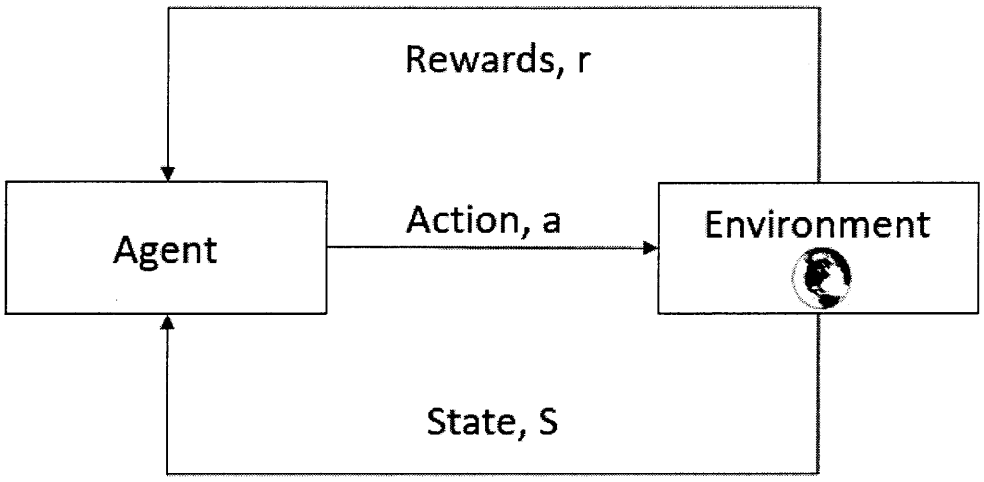


Figure 2-5 The agent environment interaction in Reinforcement Learning

RL is different from supervised learning, another ML branch where it is learning from a training dataset, labeled data provided by an experienced external supervisor. Each labeled training data set is a description of a state with a specification, the label, a specific action that the system should take, to identify the category or the class of the action. The objective of supervised learning is to elicit the system's responses, so it can act intelligently in situations or states do not present in the labeled training dataset. This kind of learning is important but not adequate to learn the environment by direct interaction, such complex systems or interactive learning systems, it is difficult to have sample training dataset for a particular system that represents all situations the agent can face. Different examples of supervised learning algorithms have been studied including K-Nearest Neighbor (KNN), Bayesian Learning (BL), regression models and Support Vector Machine (SVM) (Alpaydın, 2014).

Also, the RL is different from unsupervised learning, the third branch of ML with the objective of finding hidden structure of unlabeled dataset to classify unlabeled input data into groups by examining their similarities. Comparing to supervised learning, that learns mapping from input dataset to output results where the supervisor has the accurate data values. On the other side, the unsupervised learnings do not have this supervisor, the goal of unsupervised learning is to identify patterns among unlabeled datasets that can help the system decides what to do when encounters unknown input in the future. Different algorithms have been researched under unsupervised learning including, K- means clustering, Principal Component Analysis (PCA) and Independent Component Analysis (ICA) (Alpaydın, 2014). RL addresses the problem of CW adjustment by using an iterative learning approach. Through trial and error, The RL agent learns to optimize the selection of CW values based on the network conditions and performance objectives. By interacting with the environment and receiving feedback on selected actions, the agent gradually improves its decision-making process. This allows for dynamic and adaptive adjustment of the CW, leading to enhance performance and reduced contention in VANET.

2.2.1 Reinforcement Learning in Markovian Environment

2.2.1.1 Markov Decision Processes (MDP)

MDP (Sutton & Barto, 2018) is a mathematical framework to represent decision-making process in the RL agent. An MDP is defined is defined as a (S, A, R, S′) tuple where S denotes to set of possible states, A is the set of possible actions, R is the reward for selecting action, and S' or S_{t+1} is the next state that the environment will return because of an action A as shown in Figure 2-5.

2.2.1.2 Q-Learning algorithm

Q-learning is a widely used algorithm in RL for learning an optimal policy in an unknown environment. It is a model-free algorithm, meaning it does not require a model of the environment to learn from. Instead, it learns through trial and error, by iteratively updating estimates of the value of each state-action pair based on the rewards received from the environment. At a high level, Q-learning maintains an estimate of the optimal action-value function, which maps each state-action pair to the expected total reward obtained by taking that action from that state and following the optimal policy thereafter. The optimal policy is the one that maximizes the expected total reward. The algorithm works by updating the estimates of the action-value function using the Bellman equation, which expresses the optimal value of a state-action pair as the expected immediate reward plus the discounted value of the best next state-action pair. The updates are performed iteratively, based on the rewards received from the environment. Q-learning is often used in situations where the environment is complex and uncertain, and a model of the environment is not available. It has been applied to a wide range of problems, including robotics, game playing, and control systems.

The Quality Learning (QL) (Watkins C. J., 1989) algorithm is considered one of the innovation algorithms that was developed in the RL domain and defined (Sutton & Barto, 2018) as:

$$Q(S_t, A_t) \longleftarrow Q(S_t, A_t) + \alpha [R_{t+1} + \gamma * \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \quad (2.1)$$

where γ is the discount factor models the importance of future rewards compared to immediate rewards. α is representing the learning rate which

determines the weight given to new information compared to the existing Q-values. $Q(S_t, A_t)$ is a function of state (s) and action (a), it tells you what the quality is of being in a state (s) and taking action (a). The equation combines the current Q-values with the new information obtained from the reward and estimated maximum Q-value of the next state. By repeatedly updating the Q-value based on this equation over multiple iterations, the Q-learning algorithm converges to the optimal Q-values that maximize the cumulative rewards over time. The way of updating the quality function Q in eq 2.1, $Q(S_t, A_t)$ is representing an estimate of quality of being in state (s_t) and taking action (a_t) and this is not the perfect estimate since we are still learning. So, to calculate the future Q function at the next time step, we use $R_{t+1} + \gamma * \max a\ Q(S_{t+1}, a)$ as a target estimate minus old Q value, $Q(S_t, A_t)$ to get an error signal which used as feedback to update the Q function. Using $\max_a Q(S_{t+1}, a)$, we are maximizing over (a), it means taking some action (a) to get reward R_t but it does not have to be optimal action, so we can take some random actions to get R_t but when computing Q function of next state , we are going to optimize over action (a) to calculate the target estimate $R_{t+1} + \gamma * \max a\ Q(S_{t+1}, a)$.

2.3 Deep Reinforcement Learning

The classical RL algorithms like Q-learning stores the Q-values in tables and this approach has several disadvantages. First, using the table limits the algorithms with small number of actions and states. Real optimization problems have large state space, it is difficult to visit all possible space states to get the value for all action-state pairs. Secondly, the size of Q-table is limited because of hardware limitation.

To overcome classical RL limitation, Deep Reinforcement Learning (DRL) has been introduced utilizing Deep Neural Network as a function approximator. Using Neural Networks (NN) enables the agent to learn successful and efficient policies from high dimensional state spaces environment.

2.3.1 Neural Networks

Neural network is the heart of DRL, and the name is derived from human brain, simulating, or mimicking the structure of human's neurons aiming to learn from data. It is a class of machine learning used to model non-linear relationships in huge datasets. Neural network consisting of a sequence of node layers, the input layer, several hidden layers, and the output layer. Each node is representing neuron, which is considered the basic element in Neural networks connects to another neuron and associated with weight. The common model which used in neural network is perceptron. A perceptron is performing a mathematical operation that maps inputs to desired output. The concept of perceptron introduced in 1957 by Frank Rosenblatt (Rosenblatt, 1958) where it takes inputs (x_i), perform a mathematical operation utilizing activation function to combine input with associated weights (w_i), to obtain the output y. b represents bias, which is a real number. The activation function is a mathematical function that determine output in NN, transform the input value and pass the results to the next layer or to output layer for prediction. It is used to scale possible values of the internal states to appropriate interval for output values.

There are different NN architecture refer to how the neurons are connected to each other.

- **Feed-Forward NN (FNN)**: In this kind of NN, the information flows in one direction from input to output layer. No limitation on the number of layers in this model (López-Martín, Chavoya, & Meda-Campaña, 2013). Mainly used in computer vision, facial recognition and other applications.
- **Recurrent neural network (RNN)**: The information flows in both direction of input and output layers, forward and backward direction. it is appropriate learning model for handling sequential data such as speech recognition and language processing (Yu, Antonio, & Mora, 2022). It learns characteristics and features for time series data by using the internal memory of previous inputs in NN.

2.4 Literature Review

The significance of building an adaptive and efficient MAC protocols for VANET has been heavily studied by the researchers (Nguyen, et al., 2018) (Nguyen, et al., 2018) to enhance the performance of unicast and broadcast communications for different types for VANET applications including safety, traffic management, comfort driving and infotainment applications.

2.4.1 Multi Hop Broadcasting in VANET

In Multi-hop broadcasting, the broadcasting is extended to the received vehicles to flood the messages, to ensure that the warning message will be received by all vehicles that might be affected. However, the traditional way of broadcasting using flooding where every single vehicle rebroadcasts the message is inefficient because of the following reasons.

1. In the dense network, the broadcast protocol will disseminate the same message over and over. This will waste network resources and makes

broadcasting not scaled with high density network (Panichpapiboon & Pattara-atikom, 2012).

2. Message collision when large number of vehicles rebroadcast at the same time. This will have a negative impact on the packet delivery ratio (Panichpapiboon & Pattara-atikom, 2012).

3. A broadcast storm occurs when there are many broadcast messages disseminating across the network (Wisitpongphan N. , Tonguz, Parikh, Mudalige, & Bai, 2007) .

4. Vehicles are accessing wireless channel at the same time; this will result in a long waiting time at the MAC layer and consequently increase end-to-end delay time (Wu, Chen, Ji, Ohzahata, & Kato, 2015) .

5. High vehicles mobility in VANETs and rapid dynamic channel conditions that cause difficulty in disseminating the messages efficiently among vehicles (Tang, Geng, Chen, & Moharrer, 2017).

Much research has been conducted in VANETs to address broadcasting issues and find an optimum efficient and reliable protocol to disseminate messages across VANETs. The proposed broadcasting protocols can be classified into two main categories, the transmitter based and the receiver-based protocols. In the transmitter-based protocols, the transmitter vehicle is responsible for selecting the next forward relay vehicle and attach its address in the message. In the receiver-based protocols, once the message is received each vehicle must decide whether to rebroadcast the received message or not. (Slavik, Mahgoub, & Sibai, 2011), (Slavik, Mahgoub, & Alwakeel, 2014), (Limouchi, Mahgoub, & Alwakeel, 2016), (Chuang & Chen, 2013), (Miao, Cai, Luo, & Dong, 2016) have presented receiver based, the drawbacks of

these protocols are increased overhead and reduced bandwidth utilization efficiency and increased end to end delay. In (Slavik, Mahgoub, & Sibai, 2011) and (Slavik, Mahgoub, & Alwakeel, 2014) the distance to mean broadcast method that will be used for comparison in the result section, the author is using the vehicle positional information to estimate the coverage area. DTM estimates coverage by calculating the spatial mean of the neighboring vehicles from whom the message has been received. Each vehicle finds the distance to the spatial mean, then the vehicle will be able to rebroadcast the message if the calculated distance exceeds a predefined threshold. In (Limouchi, Mahgoub, & Alwakeel, 2016) , the author proposed a receiver based fuzzy broadcast protocol. The receiver vehicle will use fuzzy logic based on coverage, connectivity, and mobility factors to decide whether to rebroadcast or not. This protocol could end up with many vehicles rebroadcast the message and increase network overhead. In (Chuang & Chen, 2013), Density-aware emergency message extension protocol (DEEP) is proposed. The road is divided into segments and the vehicles in the segments that are further away from the source are given a shorter waiting time to rebroadcast the message. Considering only the distance factor to relay the message and ignoring the mobility and connectivity will result in losing the messages that negatively affect the network performance. In (Miao, Cai, Luo, & Dong, 2016), an optimized scheme is presented by considering the influence of distance factor and time delay. The authors used fuzzy logic to calculate and gain the output fuzzy sets of forwarding probability, then define forwarding probability values. Each node set timers to forward message.

In transmitter-based protocols, the transmitter will select next relay vehicle using vehicles' information in hello messages that are exchanged among the vehicles. In (Pekşen & Acarman, 2012), (Jiang, Guo, & Chen, 2008), (Sung & Lee, 2012) the farthest vehicle from the transmitter is

considered as the appropriate relay to rebroadcast the message and cover more area and mitigate send redundant messages. Considering distance factor can cover more area and minimize redundant messages but neglecting channel quality and mobility factors will increase number of lost messages and affecting protocol reliability. In (Rehman, Bourdoucen, & Ould-Khaoua, 2015), the link quality has been used to select next forwarder relay. A link quality estimation algorithm is exploited to recurrently quantify the quality of forward communication links. Based on the quality link, the transmitter vehicle finds out which neighbors have the good quality link and assigned them to rebroadcast the message. In (Zhu, Gao, & Foh, 2015), the velocity and distance have been considered to select next relay.

2.4.2 IEEE802.11p MAC Protocol in Vehicular Ad Hoc Networks

IEEE 802.11p employs EDCA scheme which is an enhanced version of DCF scheme for channel access management. EDCA and DCF using CSMA/CA with BEB mechanism to make sure that the vehicle that needs to transmit data, sense the channel medium to verify if it is free. The BEB is used to reduce probability of collision among vehicles by incrementing CW parameter based on the collision detected by the vehicles and receiving ACK message in the unicast transmission. While, in broadcast transmission, where no ack message is used, the CW will set to the minimum value CW_{min}. The importance of having an adaptive MAC protocol for VANET has been discussed thoroughly in the literature to enhance VANET performance for safety (Wang, Yang, Gu, & Shi, 2016) and non-safety application. Given the huge number of vehicles that share the medium, especially in urban streets, the CSMA/CA is not reliable and lead to frequent packet collisions among the

neighbors (Panichpapiboon & Pattara-atikom, 2012) (Miao, Djouani, Van Wyk, & Yskandar, 2013) (Li, Song, Lu, & Yue, 2020). CSMA/CA needs appropriate or adaptive CW mechanism depends on the network traffic density, since when you have high traffic density, a large CW value will be appropriate to avoid collision among contended vehicles. On the other side, a small CW value will be suitable for high density traffic to avoid waiting too long to send the data (Cailean, Cagneau, Chassagne, Popa, & Dimian, 2015) (Wu, Ohzahata, Ji, & Kato, 2014) (Wu, et al., 2018), and efficiently utilized the channel Bandwidth.

Several research had been evaluated and investigated the effect of CW on the VANET performance. In (Eichler, 2007) the small CW value will affect the network performance in high load scenarios, so the throughput is decrease while the delay is increasing significantly. The authors (Wang, Ahmed, Krishnamachari, & Psounis, 2008) had studied two CW adaptations schemes in V2I environment after concluding that the default CW setting does not perform well under highly dynamic VANET environment. In (Jang & Feng, 2010) proposed a detection-based MAC mechanism to detect network congestion and number of nodes to dynamically adjust contention CW. The authors in (Rawat D. , Popescu, Yan, & Olariu, 2011) proposed scheme that adjusts CW according to the instantaneous collision rate. In (Campolo, Vinel, Molinaro, & Koucheryavy, 2011), the author shows that the Packet delivery probability can be modeled as a function of contention window size and number of vehicles, and it is affected negatively when the vehicles' number is increasing. In addition, the author in (Harigovindan, Babu, & Jacob, 2012) addressed the unfairness problem exists among vehicles with different velocities in V2I network, it proves that smaller CW will benefit the vehicles that moving fast and positively affect the packet delivery performance. In (

Kloiber, Härri, Strang, & Sand, 2014), the author suggests that increasing CW to reduce the probability of simultaneous transmissions and increase packet delivery for message broadcasting which is more delay- tolerant. Furthermore, (Wu , et al., 2018) a swarming approach has been proposed to adapt CW for optimizing the one hop-delay in inter-platoon V2V communications. In summary and based on the research, there is no CW value that is suitable for all VANET conditions and environments, and that can be a problem in unicast and broadcast communication which will negatively affect the network performance.

2.4.3 Reinforcement Learning for Contention Window Adaptation

Significant effort has been done in both academic and industrial research on using RL and DRL algorithms for adapting CW in vehicular wireless MAC layer to improve the broadcast and unicast performance of safety and non-safety messages for V2V and V2I communication environments. A QL based MAC protocol has been proposed in (Wu, Ohzahata, Ji, & Kato, 2014) for delay sensitive VANET applications. QL algorithm is used to adjust the CW size to avoid message collision. The broadcast aspect is not considered in this work, it is only considered V2V unicast communication. A QL algorithm with collective contention estimation for V2V broadcasting is proposed in (Pressas, Ali, Tian, & Nekovee, 2017) (Pressas, Sheng, Ali, & Tian, 2019). The protocol does not interfere with safety broadcast messages. The author in (Choe, Choi, Ahn, Park, & Ahn, 2020) proposed DQN-based MAC algorithm for V2V safety packet broadcast for infrastructure-less congested VANET.

2.5 Conclusion

This chapter has provided a theoretical overview of the VANET network and discussed different VANET architecture, standards, applications, Communication types and thoroughly the MAC channel access scheme. The CSMA/CA MAC is the default channel access protocol for communication under IEEE 802.11p which is inadequate for highly dynamic and congested environment. The CW parameter is considered the critical factor that affecting CSMA/CA performance and needs to be adjusted to suite different VANET environment conditions. According to the literature, The CW value should be adaptable depending on various environment's factors. Traditional approaches for MAC access sharing are not suitable especially in dense environment. Thus. Much research has been focused on utilizing ML and RL to solve the MAC channel access problem.

Chapter 3

Smart Fuzzy Inference-based Multi-hop Broadcast Protocol for VANET

Chapter 3 Smart Fuzzy Inference-based Multi-hop Broadcast Protocol for VANET

3.1 Introduction

Vehicular Ad-Hoc Network (VANET) is a subset of Mobile Ad-hoc Network (MANET) that provide a communication among mobile vehicles using a dedicated short-range communication (DSRC). IEEE 802.11p standard is produced to support wireless communication among vehicles to broadcast safety and other type of messages, this standard referred as wireless access in vehicular environment (WAVE). VANET is a new promising solution in wireless communication that is developed to enhance road safety, traffic efficiency, driver and passenger's comfort and many other smart applications. It includes enabling vehicles to communicate with other vehicles directly forming vehicle to vehicle communication (V2V) or communicate with roadside unit (RSU) forming vehicle to infrastructure communication (V2I) (Olariu & Weigle, 2009). Through these communications, vehicles exchange messages containing real time information about vehicles' position, speed, direction, acceleration, etc. This information will help the drivers to know the status of the network and predict vehicles' movement to take early action about any hazardous situation and avoid them. Another example, in V2I the Roadside Unit (RSU) needs to advertise location-based services (gas station, restaurants, advertisements and so on). This needs to be broadcast to the nearby vehicles to inform the driver about their services.

The VANETs features made IEEE 802.11P standard to play an important role in improving ITS that is considered a key pillar of improving the development of the future autonomous vehicles and developing SC. Among different types of VANETs' applications, the road safety is considered the

most urgent problem that needs to be addressed and researched to save humans' life and money. When an accident happens on a road, a safety or a warning message should be disseminated immediately to alert all the incoming and speedy vehicles that might be affected by this accident. Even the vehicles that seem to be very far from the accident, they will reach the danger very soon. In the highway, when you are driving in high speed, milliseconds will be a matter to avoid any hazardous or danger situation. For example, When the first accident occurred, there should be a technique to disseminate alarms to all incoming vehicles about the accident, so driver can take an immediate decision to avoid such danger event and saving humans' life and money.

In Vehicular Ad-Hoc Networks (VANETs), effective and efficient communication plays a important role in enabling various applications such as traffic management, collision avoidance, and information dissemination. Broadcasting, the process of disseminating information from one vehicle to all nearby vehicles, is a fundamental mechanism in VANETs for exchanging safety warnings, traffic updates, and other critical messages.

However, the broadcasting process in VANETs presents unique challenges due to the dynamic nature of the vehicular environment. Factors such as varying traffic densities, changing road conditions, and diverse driving behaviors introduce uncertainties and complexities that can impact the effectiveness of broadcasting schemes.

To address these challenges, the utilization of fuzzy logic emerges as a promising approach in minimizing VANET broadcasting. Fuzzy logic offers a flexible and adaptive framework for decision-making and information processing, capable of capturing the inherent uncertainties and vagueness in the vehicular environment.

3.2 Developing Fuzzy Expert Protocol

In order to deliver the broadcast message to all vehicles that might be affected by road's condition with a smaller number of rebroadcasting, fuzzy inference system will be used to select the suitable relay candidate to rebroadcast the message. Before broadcasting a message, a sender vehicle needs to select the next relay and attach its address in the message. Upon receiving the broadcast message, the vehicle that is selected as a relay will only be able to rebroadcast the message.

Fuzzy logic is a smart tool that used to address decision making process based on different information inputs. Fuzzy logic is an approach for computing based on the "degrees of truth" rather than usual "true or false" Boolean logic used in modern computer systems. Fuzzy logic tries to mimic the human brain in the way to take a decision based on some vague input information.

The importance of employing fuzzy logic in minimizing VANET broadcasting lies in its ability to optimize the dissemination of messages while considering the dynamic nature of the network. By leveraging fuzzy logic techniques, broadcasting schemes can adaptively adjust transmission parameters based on real-time factors such as traffic density, vehicle speeds, and road conditions. This adaptability ensures that the broadcasting process remains efficient and effective even in the face of unpredictable variations in the environment.

Furthermore, fuzzy logic allows for intelligent decision-making in VANET broadcasting by considering multiple input variables simultaneously.

Linguistic terms and fuzzy rules can be defined to capture the relevance and priority of different factors, enabling a context-aware approach to message dissemination. This intelligence leads to better resource utilization, reduced channel congestion, and improved overall network performance.

The fuzzy inference system is proposed to choose the suitable relay vehicle to rebroadcast the message. The fuzzy controller in the sender vehicle is utilizing the attached information distance, mobility and RSSI with hello messages that periodically exchanged among neighbors. Each vehicle builds up a detailed information database for all neighbors.

- Distance: Based on the received hello messages, the sender vehicle calculates distance to its neighbors. The longer the distance between sender and receiver the more chance to be selected as a relay, it helps to extend coverage area and decrease redundant messages. However, it does not mean that the furthest neighbor is the best relay to rebroadcast the message taking into consideration the high speed and poor signal strength of the vehicle affect the relay selection.
- Mobility: Upon receiving the hello messages, the sender vehicle calculates vehicles mobility for its neighbors. A lower mobility vehicle the more appropriate vehicle desired to be selected to rebroadcast the message. Lower mobility makes the vehicle more stable because it may increase network connectivity comparing to high mobility vehicle that change the position instantly and becomes out of the range of intended neighbors that are supposed to receive broadcast message from that relay.
- RSSI: Upon receiving the hello messages, the sender vehicle calculates vehicles' signal strength for its neighbors. The vehicle with strong signal

strength becomes appropriate to be selected as relay to get more reliable connection and make sure to transmit message to all neighbors.

3.3 Design of Fuzzy Logic Inference

Each sender vehicle builds a fuzzy logic controller (FLC) for selecting next forwarder relay according to fuzzy decisions that are taken based on fuzzy rules for three input parameters distance, mobility and RSSI. The basic structure for FLC consists of four components as shown in Figure 3-1. (1) Fuzzification which is the mapping of the crisp values into a fuzzy set represented by membership functions (MFs). (2) Rule evaluation which takes the fuzzified inputs and applies them to the fuzzy rules. In other word, the rules use the input membership values as weighting factors to determine their influence on the final output by computing the logical product of the membership weights for each active rule. Aggregation of the rule outputs, which is the process of unification of the outputs of all rules. (3) Defuzzification is the last step in fuzzy inference system which converts fuzzy output into crisp value.

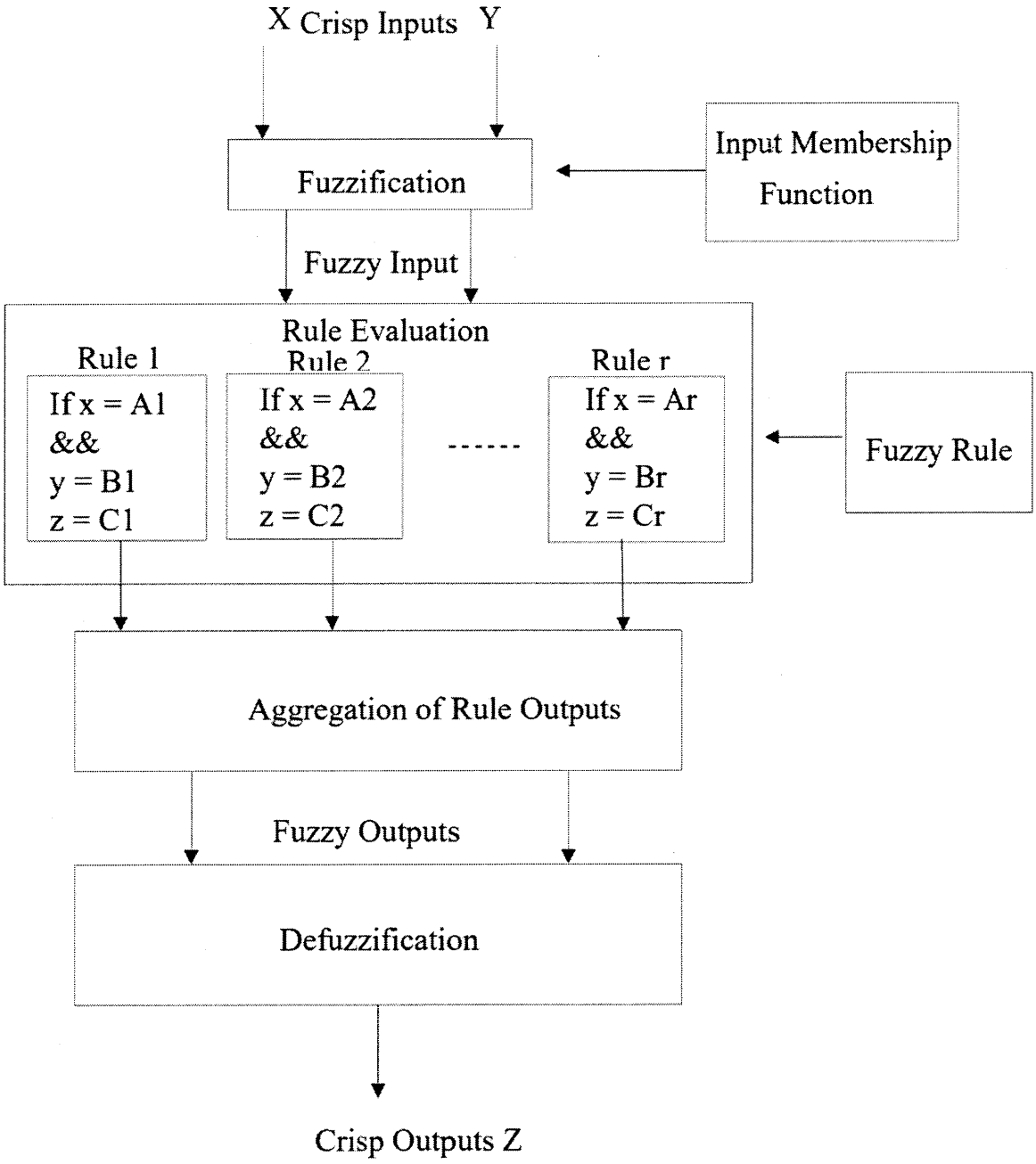


Figure 3-1 Block Diagram for fuzzy logic controller

3.3.1 Fuzzification

The process of mapping of the crisp values into a fuzzy set represented by membership functions (MFs). The FLC has three linguistic variables: the distance from sender, Vehicle's Mobility, RSSI and relay fitness value for

each neighbour. Fuzzy sets can have a variety of shapes, but for simplifying the process of computation, triangular / trapezoidal membership functions are assumed for mapping the input values into the respective linguistic sets. Figure 2 to 4 show the fuzzy sets for all linguistic variables used in FLC.

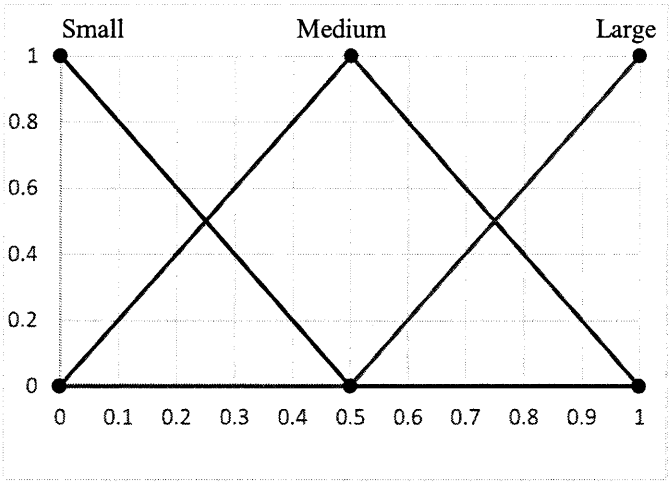


Figure 3-2 Input Variable "Distance"

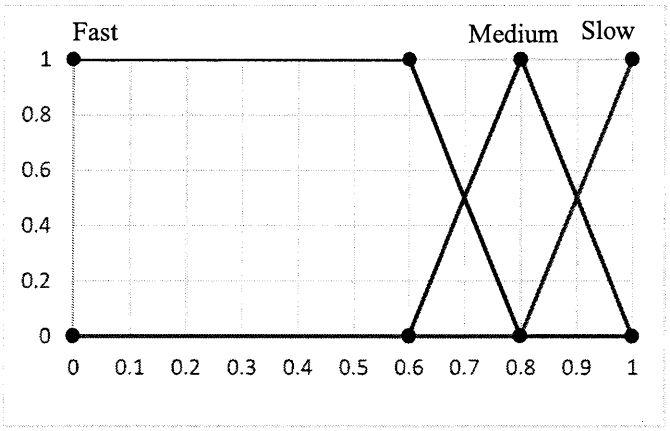


Figure 3-3 Input Variable "Mobility"

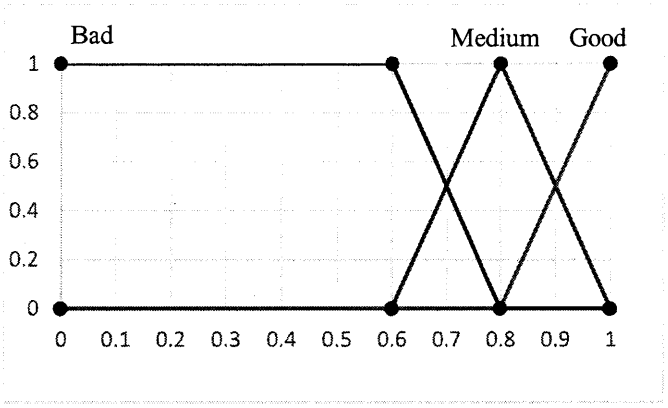


Figure 3-4 Input membership function for RSSI

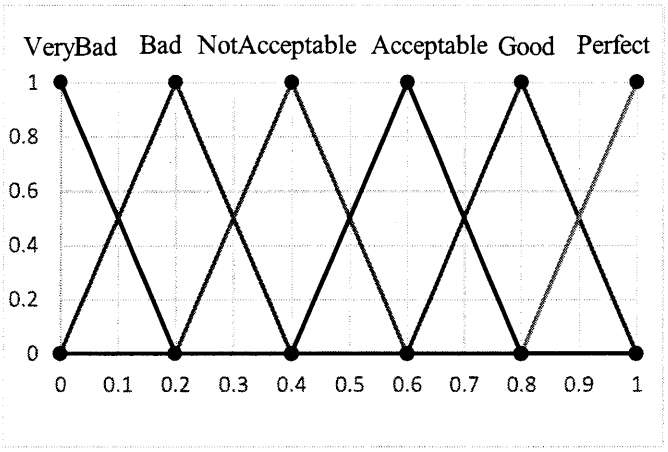


Figure 3-5 Output membership function (Relay fitness value)

3.3.2 Elicit and construct Fuzzy rules

As mentioned earlier, the fuzzy inference system tries to mimic the cognitive capability of human brain based on the experience, knowledge accumulation, etc. The experience of the human controller is usually expressed as some linguistic "IF- THEN" rules that state in what situation(s) which action(s) should be taken. So, the rules are chosen to get minimum overshoot and faster response. Some of the rules applied in the designed FLC are:

1. If (Distance is large) AND (Mobility is Slow) AND (RSSI is Good) then (Output is perfect)

2. If (Distance is Med) AND (Mobility is fast) AND (RSSI is Good) then (Output is bad)

3. If (Distance is Small) AND (Mobility is Slow) AND (RSSI is bad) then (Output is bad)

Table 3-1 shows the rules included in FLC, we only show part of the rules, the total rules can refer to supplemental material. Based on these rules, the FLC includes all the expected situations that would possibly occur.

3.3.3 Defuzzification

The process of converting fuzzy output into crisp value. The output membership function is presented in Figure 3-5. There are many approaches to achieve defuzzification, here we used centre of area approach [17] as equation 3.1.

$$u = \frac{\sum_i m(x_i)x_i}{\sum_i m(x_i)} \qquad (3.1)$$

Table (3-1): Fuzzy rules considered for the proposed protocol

	Distance	Mobility	RSSI	Output
Rule 1	**Large**	**Slow**	**Good**	**Perfect**
Rule 2	Large	Slow	Medium	Good
Rule 3	Large	Slow	Bad	Not acceptable

Rule 4	Large	Medium	Good	Acceptable
Rule 5	Large	Medium	Medium	Good
Rule 6	Large	Medium	Bad	Acceptable
Rule 7	Large	Fast	Good	Bad
...				

Where xi is a crisp value in universe X, $m(xi)$ is a membership value and u is a crisp output. The approach utilized in this chapter involves the application of the Mamdani fuzzy inference method (Negnevitsky, 2005) to design FLC in order to select next forwarder relay. FLC is configured step by step to explain each part of fuzzy logic to select next forwarder relay.

After defining fuzzy sets and constructing the fuzzy rule, the defuzzification engine is used to encode them to calculate the appropriate Relay based on fuzzy rules and the linguistic value for Fuzzy inputs.

3.4 Simulation and Evaluation

The network model is simulated using network simulator NS-3 (version 3.29) to evaluate our proposed protocol's performance and compare it with other protocols. The simulation's parameters and synthetic highway scenario parameters are shown in Table 3-2 and 3-3. The vehicle's mobility is simulated using NS-3 model which support synthetic highway scenario and the geographical position is based on NS-3 constant speed and rectangle position model. The VANET simulation is implemented in a rectangular area in a 3000 ×300 meters to simulate highway environment. All vehicles have the same transmission range, and each vehicle can receive the messages with the transmission range. The road width is negligible compared to transmission range. All vehicles are equipped with VANETs and GPS capabilities that enable them to know their position, speed, direction, and road ID and can

update the information using hello messages among neighboring vehicles. The following metrics are evaluated to compare our proposed protocol with others.

Reachability: The percentage of vehicles receiving a broadcast message to the total number of vehicles in the network. It is an indicator for protocol's reliability.

Byte sent per vehicle: Total number of bytes sent per vehicle that received the message, it is an indicator for measuring the protocol's efficiency.

Rebroadcast per covered vehicle: The ratio of message retransmissions to the number of received messages.

Table (3-2): Simulation Parameters

Application Layer	Basic safety Massage (BSM)
Routing protocol	AODV
MAC Layer	WAVE IEEE802.11P
Transmission Range	300m
Antenna Type	Omni directional
Topology	3000 × 300 meters
Data Rate	12Mbps
Packet size	200 Byte
Propagation Model	Tow-Ray Ground Loss Model
Simulation Time	300s

Table 3-3: Synthetic Highway scenario parameters

Speed	20 – 35 m/s
X- coordinate	3000m
Y- coordinate	300m
Number of Nodes	50, 100, 150, 200, 250, 300
Mobility Model	Constant speed Model

Fuzzy logic was compared with WAVE standard and DTM (Slavik, Mahgoub, & Sibai, The distance-to-mean broadcast method for vehicular wireless communication systems, 2001). As we mentioned earlier that the WAVE standard using flooding technique, this requires every receiver vehicle to rebroadcast messages to all vehicles in the range. In DTM scheme, it calculates the spatial mean of the receiver transmitting neighbors and finds the vehicle's distance to that spatial mean. The neighbors will take a decision to rebroadcast if the calculated distance exceeds the predefined threshold.

Figure 3-6, Figure 3-7, and Figure 3-8 show the reachability of the proposed fuzzy based protocol in compare with WAVE and DTM methods for various number of vehicles, hops count and transmission range in highway scenario. As can be seen, All the protocols have achieved 95% reachability for different traffic densities, the fuzzy scheme has outperformed the other schemes especially during high traffic density where the number of vehicles is increasing.

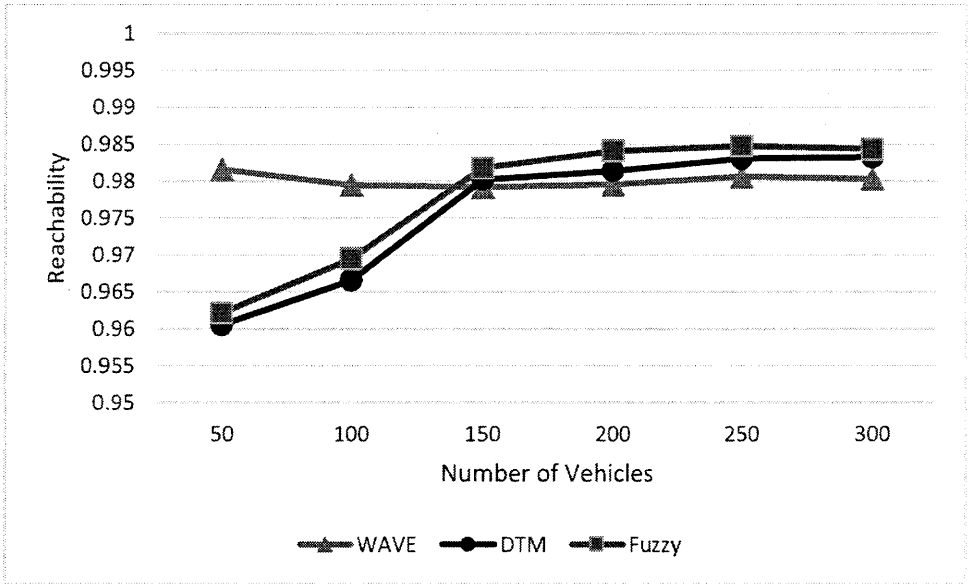


Figure 3-6 Reachability versus number of vehicles for fuzzy based protocol, WAVE and DTM Protocol

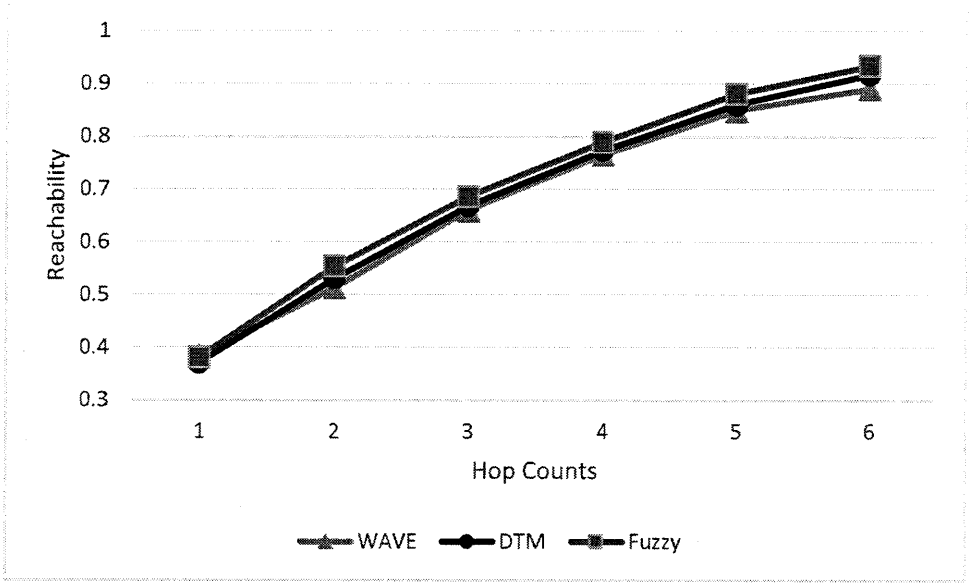


Figure 3-7 Reachability versus number of hops count for fuzzy based protocol, WAVE and DTM Protocol

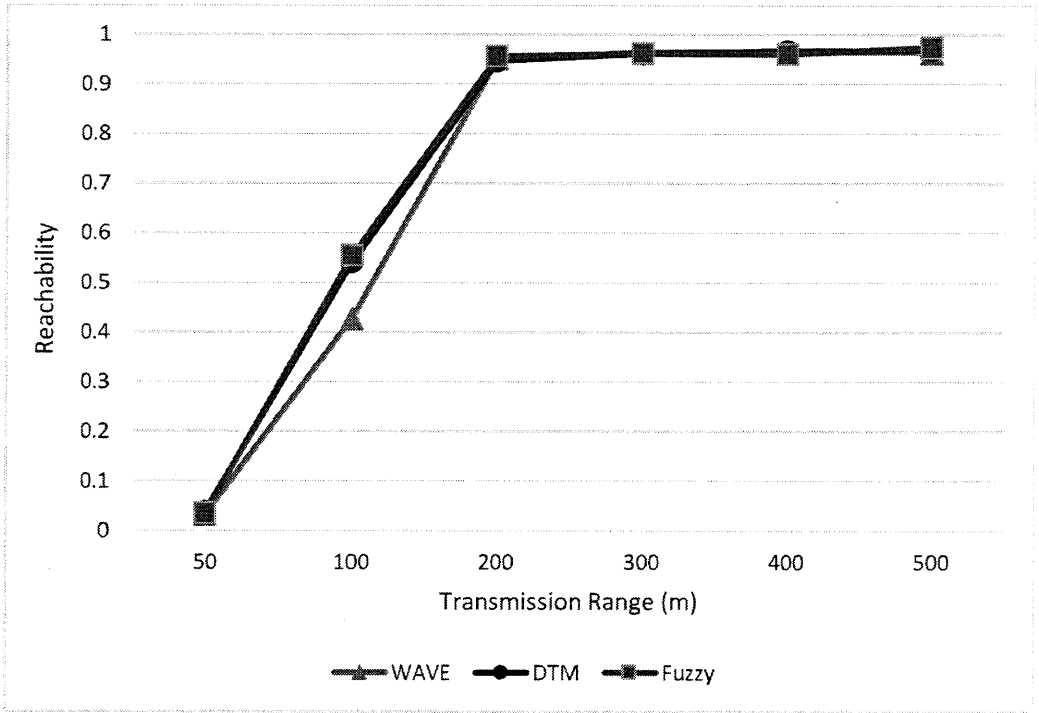


Figure 3-8 Reachability versus transmission range for fuzzy based protocol, WAVE and DTM Protocol

Figure 3-9, Figure 3-10, and Figure 3-11 demonstrates bandwidth utilization in terms of Bytes sent per covered vehicles. It is noticed that the fuzzy scheme has lower bandwidth utilization for number of vehicles' densities compare to other schemes.

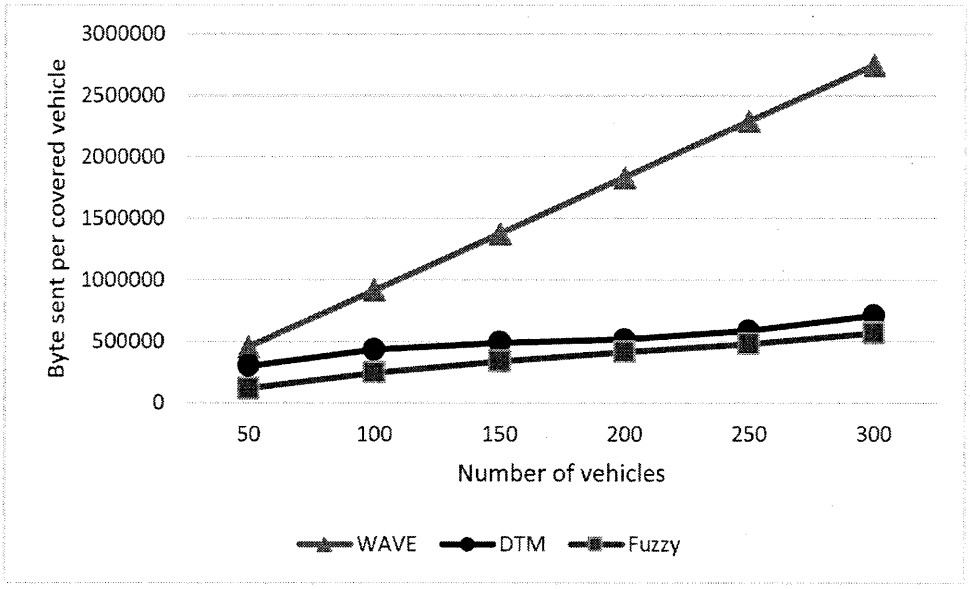


Figure 3-9 Byte Sent Per Covered node versus number of vehicles for Fuzzy based protocol, WAVE and DTM protocols

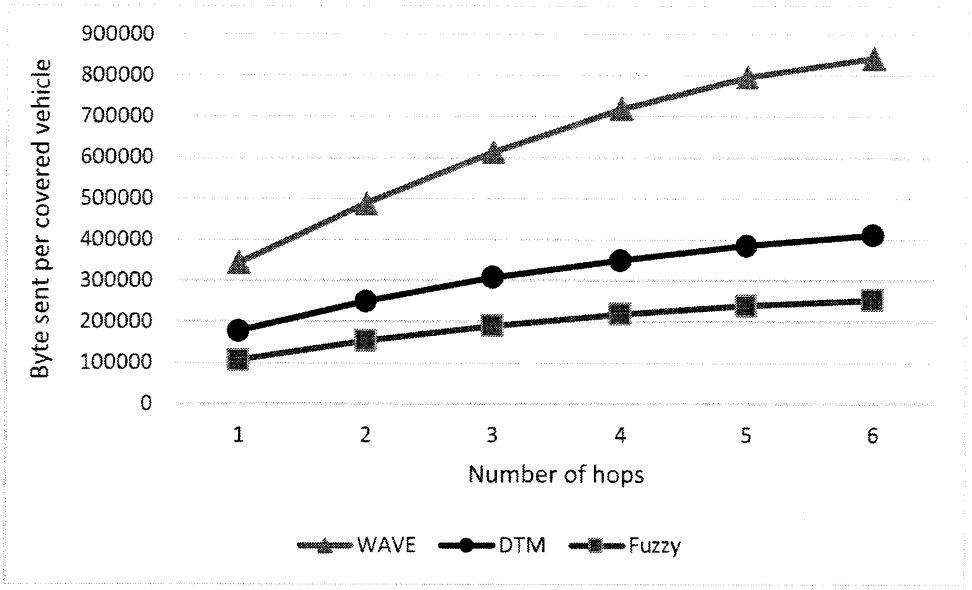


Figure 3-10 Byte Sent Per Covered node versus number of hops count for Fuzzy based protocol, WAVE and DTM protocols

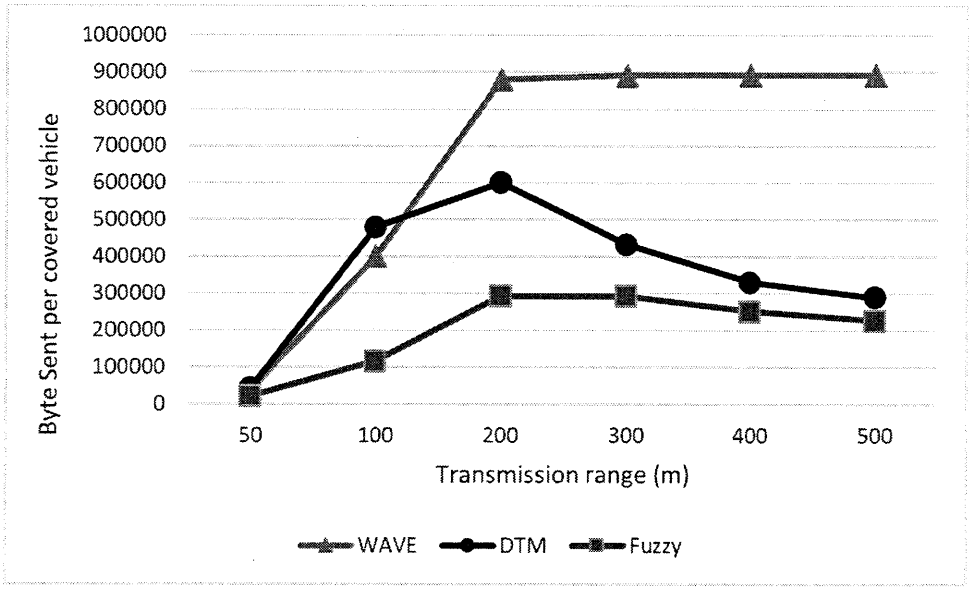


Figure 3-11 Byte Sent Per Covered node versus transmission range for Fuzzy based protocol, WAVE and DTM protocols

According to the Figure 3-12, Figure 3-13 and Figure 3-14 the fuzzy scheme significantly decreases the amount of rebroadcasting across different traffic densities.

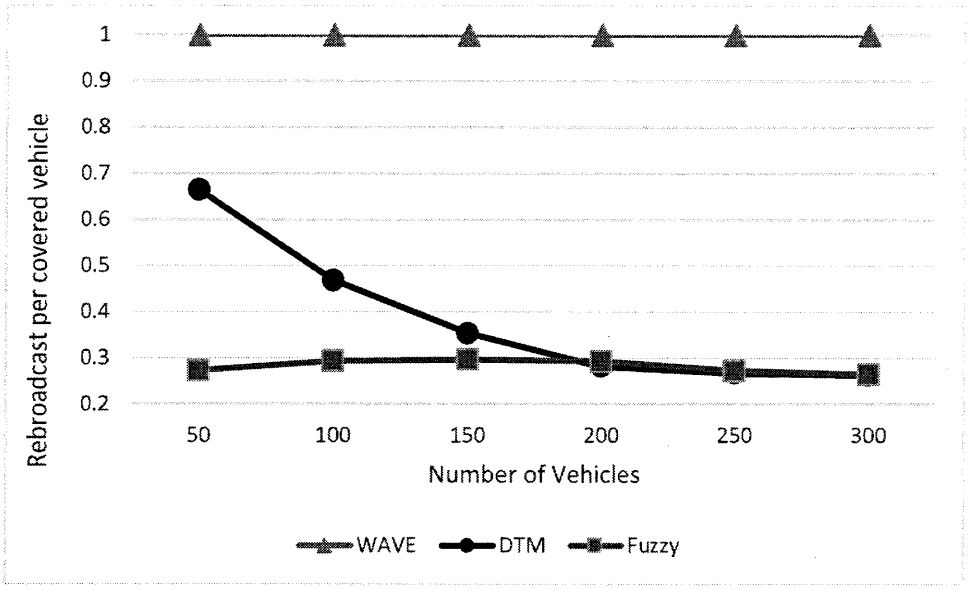


Figure 3-12 Rebroadcast Per Covered node versus Number of vehicles for Fuzzy based protocol, WAVE and DTM protocols

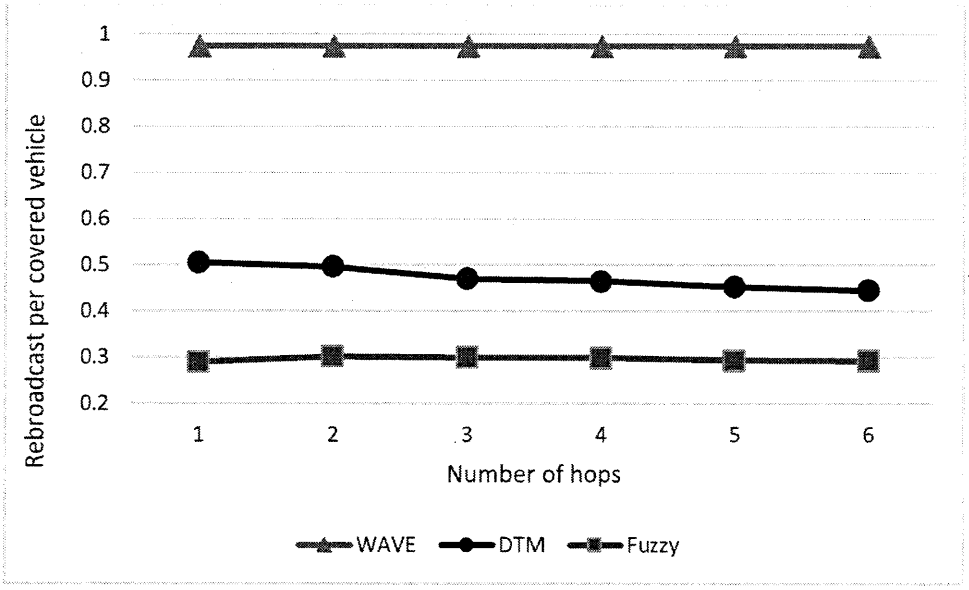


Figure 3-13 Rebroadcast Per Covered node versus hops count for Fuzzy based protocol, WAVE and DTM protocols

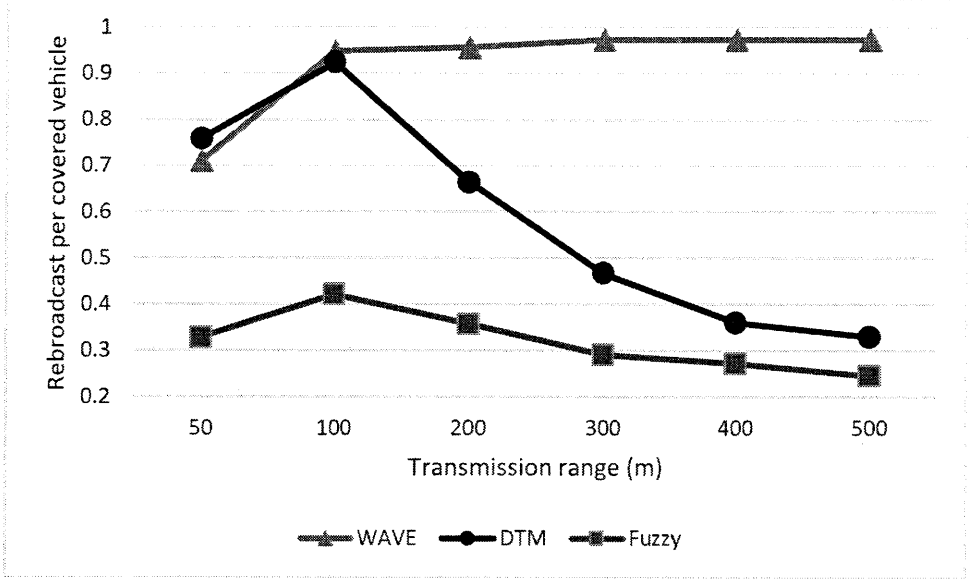


Figure 3-14 Rebroadcast Per Covered node versus transmission range for Fuzzy based protocol, WAVE and DTM protocols

3.5 Conclusion

In this chapter, a novel fuzzy logic inference protocol was proposed for VANET multi-hop broadcast networks. The main objective of the protocol was to minimize the number of redundant messages by intelligently selecting the most suitable relay vehicle for message broadcasting. This was achieved by considering three important characteristics of vehicles: distance, mobility, and RSSI. Through computer network simulations, the performance of the proposed protocol was evaluated. The results indicated that the protocol effectively reduced redundancy in message broadcasting and improved the overall network performance. The intelligent selection of relay vehicles based on fuzzy logic inference proved to be a promising approach for optimizing message dissemination in VANETs. Looking ahead, there are several avenues for future work that can further enhance the network performance. One

recommendation is to explore the possibility of considering multiple relays per hop. By allowing multiple relays, the protocol can increase reliability and resilience against relay failures. This enhancement would ensure a more robust communication framework in VANETs. Additionally, it is essential to evaluate the proposed protocol in various scenarios, including dense and sparse modes. Different environmental conditions and traffic densities can significantly impact the performance of the protocol. Therefore, conducting evaluations in different scenarios will provide a comprehensive understanding of its effectiveness and adaptability.

Chapter 4

Simulated Reinforcement Learning Environment for VANETs.

Chapter 4 Simulated Reinforcement Learning Environment for VANETs

4.1 Introduction

The current and future demands on computer networks especially wireless communication technologies are rapidly increasing. The future generation of wireless technologies are considered very necessary to support emerging required services in all business sectors including but not limited to military, Intelligent transportation system (ITS), educations, finance, health-care services, gaming industry, etc. The future expansion of wireless communication technologies has opened an issue of how to design, implement, test, and evaluate such technologies before deploying them into a real-life environment. Designing and implementing a real-life model for new protocols or algorithms in wireless communication networks is considered very costly, time consuming and high risky in most of the scenarios (Mehta, Sulatan, & Kwak, 2010) (Khan, Bilal, & Othman, 2012). Due to the reasons mentioned above, the developers and researchers were actively seeking a virtual environment that would allow them to test and verify the newly designed protocols and algorithms under various conditions before deploying them into hardware.

Simulation was the most cost-effective solution that helps the researchers and developers to model, test and verify wireless communications protocols before real time implementation. It is widely used for the development of new communication architecture and network protocols (Piotr & Piotr, 2014).

VANET studies frequently employ simulation tools, such as NS3, to simulate vehicle movements, wireless communications, and protocol actions. NS3 is equipped with models, protocols, and a mobility simulator that allow

for VANET simulations and experiments. Our goal is to integrate VANET and ML by integrating NS3 with OpenAI Gym. This integration aims to merge the two fields and create a bridge between them through APIs. VANET researchers can use the interfaces that is compatible with common RL frameworks to improve their protocols.

In this chapter the integration of RL OpenAI gym (Brockman, et al., 2016) framework and NS3 (Riley & Henderson, 2010) is presented. The hybrid framework ns3-gym (Gawłowicz & Zubow, 2018) is the first framework for RL research in computer networks and wireless communication which used mainly in Wi-Fi networks. The NS3-Gym framework enables researchers to apply RL techniques to VANET problems. This framework provides a practical example of how learning techniques can be utilized in VANET research. It allows researchers to define custom state representations, action spaces, and rewards functions tailored to specific VANET research questions. By utilizing the NS3-Gym framework, researchers can explore various RL methods, such as DRL, and apply them to VANET problems. This approach can lead to the development of intelligent and adaptive solutions that improve the performance, efficiency, and reliability of vehicular communication systems in real-world scenarios.

The following is a summary of my key contributions:

- Integration of the NS3 VANET scenarios with OpenAI Gym, which provides a straightforward means of formulating and testing RL solutions in the field of ITS.
- Implementation of RL through the training of an agent to tackle VANET problems.

4.2 NS3-gym framework

Ns3-gym (Gawłowicz & Zubow, 2018) introduced by Gawłowicz is considered a benchmarking simulator for networking researcher and developers that has developed based on two well-known and trustworthy simulations in the research community, namely, ns3 and OpenAI Gym. It is a powerful simulator that facilitates the job of researchers in networking and wireless communication field by finding an environment that integrate RL with network simulator. Ns3-gym can be used to apply RL in a wide range of networking and wireless communication technologies by utilizing well-known RL python libraries like TensorFlow (Abadi, et al., 2015) and Keras (Chollet, 2015) with tested models of ns3 under one framework. In the recent years, there was a booming in the usage of RL applications, and it will play a main role for solving such complex problems and optimizing algorithms by utilizing an agent with no prior knowledge to learn making decisions by interacting with environment and facing different situations to take best actions. This kind of integration opens new opportunities for developing innovative algorithms that utilize RL and DRL in networking and wireless communication using a unified framework that combine networking models with RL and DRL-based algorithms.

4.3 Ns3-gym System Architecture

Figure 4.1 shows the architecture of Ns3-gym, it consists of different components. The two major components are the ns3 simulator and Open AI gym. Ns3 is used to design and build VANET networks while the Open AI gym is interfacing with ns3 to collect all numerical values, process them using different RL algorithms and get the optimum parameters back to ns3 to achieve the optimum network performance. Following sections 4.3.1, 4.3.2 and 4.3.3 will describe each software blocks.

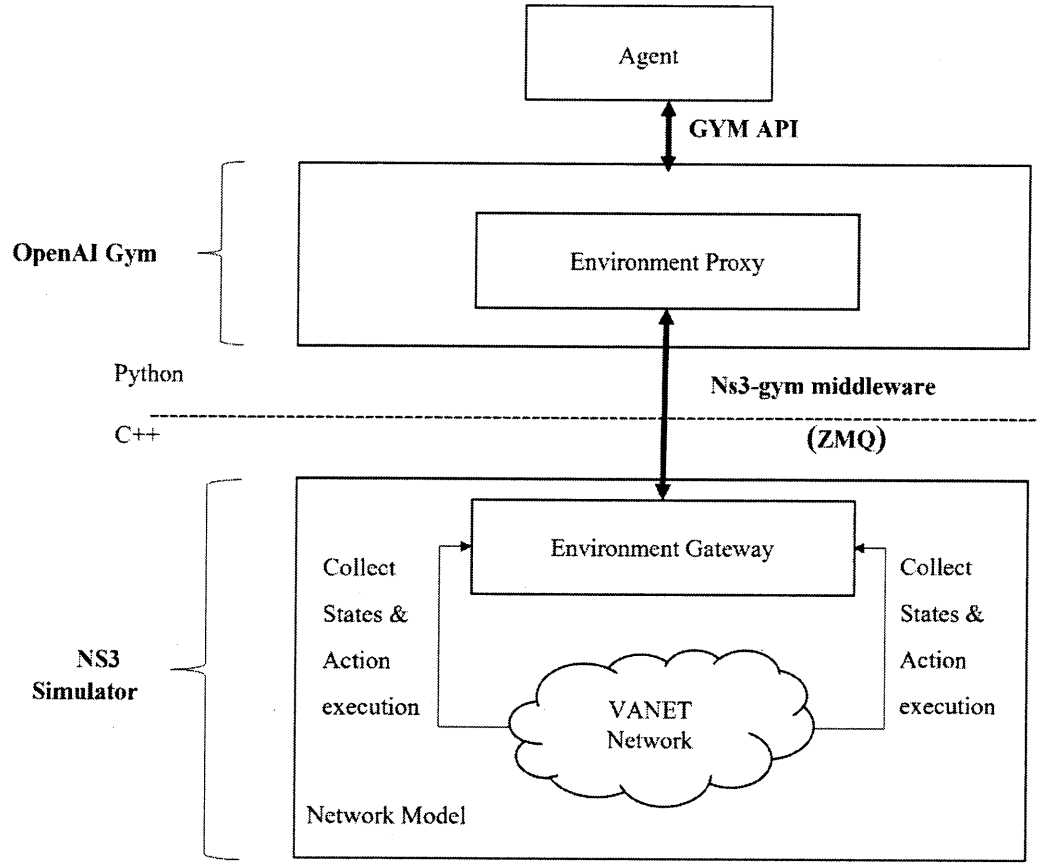


Figure 4-1 NS3-gym system architecture

4.3.1 NS3 simulation

Ns3 is considered the main component in the NS3-gym system architecture since it is used to design and implement VANET topology network. This network will be used as an environment where the RL agent interacts with the environment and get the observations and react with specific action. Ns3 is a discrete event simulator (DES) and it is written in C++ and python and supported by different modern operating systems. It is primarily used for research in educational field and becomes a popular simulation in networking and wireless communication community research. It is widely

accepted in the networking community because of it is results that are close to the real environment (Khan, Bilal, & Othman, 2012) and its ability to interact with external systems (Gawłowicz & Zubow, 2018), application and libraries. Ns3 can model OSI layers from physical to application layer in networking and communications standards.

In ns3 network scenarios, any network's event can be modeled and represented as a state for RL agent in the learning phase. Where an agent can achieve it is goal by directly interacting with and learning from its surrounding environment. The agent does not have information to decide if it is close to its goal, however, the agent observes the surrounding environment to maximize the numerical rewards in an MDP by taking actions. The process of interaction between the agent and the environment will continue to enable the agent to learn the environment and react properly through selected actions.

4.3.2 Open AI Gym

Open AI gym (Brockman, et al., 2016) is an open-source toolkit created by OpenAI for developing and implementing RL algorithms and accelerate RL research, using this toolkit, the RL agent will learn to accomplish different and complex problems. It provides Application Programming interface (API) that enable agent to interact with any learning environment that can be modeled or formalized as MDP (Otterlo & Wiering , 2012) or POMDP (Monahan G. E., 1982). OpenAI gym provides different types of environments like Atari games, Toy Text, classic Control, Robotics and Box2D. In our implementation, the environment is written and defined inside the ns3 simulation scenario. It is a framework that provides a variety of environments and agents to implement reinforcement learning algorithms, and

provides the foundation to regularly train, evaluate, and compare RL models. The OpenAI Gym consists of:

- A library that provides a variety of environments and agent collections.
- A service or API layer to compare different models as well as access the environment's variables such as state or action.

In an iterative process, the agent takes an action in its current state, to which the environment responds with a new state and reward. The agent would then choose its next action based on the environment's response. The Gym environment abstracts its implementation as well as its operations with the agent using interfaces. Those interfaces provide standard operations that we can use to interact with the environment.

To integrate OpenAI Gym and ns3 network simulator, we are using the ns3-gym framework that allows an ns-3 simulation scenario to be represented as an environment in Gym and transfers the state and actions between the simulation and the agent.

4.3.3 Ns3-Gym middleware

Ns3-gym middleware provides a full range of required communications and connectivity between ns3 simulation and Open AI gym. It is responsible for passing RL control and observed parameters including states and actions between ns3 environment and agent. ns3-gym middleware consists of two main sub-components, the environment gateway that resides in ns3 and the environment proxy that resides in Open AI gym. The environment gateway is a C++ component that starts the simulation and collects the environment states as a numerical value that have been formalized in MDP and pass them to the

environment proxy. Also, it is responsible to receive formalized actions from the proxy and pass them to ns3. On the other hand, the proxy is a python component receiving the formalized states and pass them towards the agent through gym API. This process of passing environment parameters is accomplished by using a messaging library called Zero Message Queue (ZMQ) (ZeroMQ, 2022), it supports common messaging patterns (publish/subscribe, request/reply, client/server, etc) over transport channel (TCP, multicast, WebSocket and many others). Libzmq library has been used in the implementation, it is written in C++ and compatible with ubuntu 18.04 as operating system that is used in this work implementation with gcc 4.4 compiler (Pfeifer & Law, 2022).

4.4 Ns3-gym implementation

Ns3-gym is a toolkit consists of two components (the gateway component written in C++ and the proxy component written in python), being add-ons to the NS3 and OpenAI Gym framework. In the following sections, ns3-gym implementation steps will be explained.

4.4.1 Gateway Component

The gateway will implement an interface called openGymInterface within the ns3-gym framework, in order to use the ns-3 simulation script as a Gym environment. The first step is to instantiate this interface as shown in Figure 4-2.

Creating open Gym Interface

```
1:  // OpenGym Env
```

```
2:  Ptr<OpenGymInterface>openGyminterface = CreateObject<OpenGymInterface> openGymPort);
```

Figure 4-2 Open Gym Interface

The Open Gym interface requires the implementation of a standard set of functions which are:

- GetObservationSpace: which is called once to define the observation space either as a discrete number, a box with bounded limits, a tuple, or a dictionary.
- GetActionSpace: which is called once to define the action space.
- GetObservations: which collects values from the ns3 simulation environment.
- GetReward: which returns the reward achieved by the last action.
- GetGameOver: which checks a boolean value to reset the environment based on simulation ending or predefined conditions.
- GetExtraInfo: which returns more information about current state.
- ExecuteAction: which executes the action sent by the agent in the simulation.

Once, we have our own implementation of above functions required by the interface, they are set as the callback functions as shown in Figure 4-3 on the same openGymInterface object that was created.

Implementation of a standard set of functions

```
1:  // OpenGym Env
2:  Ptr<OpenGymInterface> openGymInterface = CreateObject<OpenGymInterface> (openGymPort);
3:  openGymInterface->SetGetActionSpaceCb( MakeCallback (&MyGetActionSpace) );
4:  openGymInterface->SetGetObservationSpaceCb( MakeCallback (&MyGetObservationSpace) );
5:  openGymInterface->SetGetGameOverCb( MakeCallback (&MyGetGameOver) );
6:  openGymInterface->SetGetObservationCb( MakeCallback (&MyGetObservation) );
```

```
7:  openGymInterface->SetGetRewardCb( MakeCallback (&MyGetReward) );
8:  openGymInterface->SetGetExtraInfoCb( MakeCallback (&MyGetExtraInfo) );
9:  openGymInterface->SetExecuteActionsCb( MakeCallback (&MyExecuteActions) );
10: Simulator::Schedule (Seconds(0.0), &ScheduleNextStateRead, envStepTime, openGymInterface);
```

Figure 4-3 standard set of functions

The MyGetObservationSpace function creates a box space in OpenGym. The OpenGymBoxSpace requires a standard set of parameters that define a vector of size that matches the number of nodes return by GetNNodes() function. The values in that vector are defined as unsigned integer32 numbers and are bounded between 0 and 100 that represent number of packets in the queue as shown in Figure 4-4. The space will be used to observe the queue length of all nodes available in the network.

Creating open Gym Interface

```
1:  /*
2:  Define observation space
3:  */
4:  Ptr<OpenGymSpace> MyGetObservationSpace(void)
5:  {
6:      uint32_t nodeNum = NodeList::GetNNodes ();
7:      float low = 0.0;
8:      float high = 100.0;
9:      std::vector<uint32_t> shape = {nodeNum,};
10:     std::string dtype = TypeNameGet<uint32_t> ();
11:     Ptr<OpenGymBoxSpace> space = CreateObject<OpenGymBoxSpace> (low, high, shape, dtype);
12:     NS_LOG_UNCOND ("MyGetObservationSpace: " << space);
13:     return space;
```

```
14: }
```

Figure 4-4 Observation space function

Similarly, MyGetActionSpace creates a similar box space in OpenGym to the one created in MyGetObservationSpace function. The action space vector size is also equal to the number of nodes and the actions are encoded as a numerical value bounded between 0 and 100, which each value represents a specific action as shown in Figure 4-5.

Action Space

```
1.      /*
2.       Define action space
3.      */
4. Ptr<OpenGymSpace> MyGetActionSpace(void)
5. {
6.      uint32_t nodeNum = NodeList::GetNNodes ();
7.     float low = 0.0;
8.    float high = 6.0;
9.    std::vector<uint32_t> shape = {nodeNum,};
10    std::string dtype = TypeNameGet<uint32_t> ();
11.   Ptr<OpenGymBoxSpace> space = CreateObject<OpenGymBoxSpace> (low, high, shape, dtype);
12.   NS_LOG_UNCOND ("MyGetActionSpace: " << space);
13.   return space;
14. }
```

Figure 4-5 Action space function

The number of nodes is defined as a variable in the main function as shown in Figure 4-6.

Nodes creation and simulation time

```
1:   int
2.   main (int argc, char *argv[])
3.   {
4.  //LogComponentEnable ("Txop", LOG_LEVEL_DEBUG);
5. uint32_t m_nNodes = 20; ///< number of nodes
5.         double m_TotalSimTime = 60/0; /// total sim time
6:       double simulationTime = m_TotalSimTime;  //seconds
```

Figure 4-6 Nodes creation and simulation time

The MyGetGameOver functions, as shown in Figure 4-7 allows us to specify any predefined conditions at which the environment will reset. In our implementation, there are no predefined conditions and thus the Boolean value is always set to false, such that the environment would only reset when it reaches the end of the simulation.

MyGetGameOver Function

```
1:   /*
2. Define game over condition
3.   */
4. bool MyGetGameOver(void)
5. {
```

```
6.  bool isGameOver = false;
7.  NS_LOG_UNCOND ("MyGetGameOver: " << isGameOver);
8.  return isGameOver;
9. }
```

Figure 4-7 MyGetGameOver function

Total simulation time is specified as a global variable in the main function. Reaching this time will automatically end the simulation and trigger the GameOver function. The simulation time which would trigger the environment reset is defined in the simulationTime inside the main function as shown in Figure 4-8.

Main function

```
1:  int
2.  main (int argc, char *argv[])
3.  {
4.      //LogComponentEnable ("Txop", LOG_LEVEL_DEBUG);
5.      double m_TotalSimTime = 60/0; /// total sim time
6:     double simulationTime = m_TotalSimTime;  //seconds
```

Figure 4-8 Main function

The implementation of MyGetObservation function, as shown in Figure 4-9, creates an OpenGym Box container object. The OpenGymBoxContainer also uses a box with bounding limit that match the definition of the observation space object created earlier in Figure 4-4. It uses a vector with the same size as the current number of nodes to create a box in which it collects the observed variables values at each node.

GetObservation function

```
1: /*
2. Collect observations
3. */
4. Ptr<OpenGymDataContainer> MyGetObservation(void)
5. {
6.       uint32_t nodeNum = NodeList::GetNNodes ();
7.       std::vector<uint32_t> shape = {nodeNum,};
8.      Ptr<OpenGymBoxContainer<uint32_t> > box = CreateObject<OpenGymBoxContainer<uint32_t>
9.      >(shape);
10.     for (NodeList::Iterator i = NodeList::Begin (); i != NodeList::End (); ++i) {
11.        Ptr<Node> node = *i;
12.        Ptr<WifiMacQueue> queue = GetQueue (node);
13.        uint32_t value = queue->GetNPackets();
14.        box->AddValue(value);
15. }
16  NS_LOG_UNCOND ("MyGetObservation: " << box);
17. return box;
18. }
```

Figure 4-9 GetObservation function

The function loops through each, then calls GetQueue function passing the node object. The GetQueue function, as shown in snippet 8, uses the node device's attributes to return the queue. The queue is then used to get the number of packets using the GetNPackets function, and then add it to the OpenGym box container. This process is repeated for each node.

The MyGetReward function is implemented to get the reward from the last executed action as shown in Figure 4-10. Each time a packet is received by the client, the value of the g_rxPktNum get incremented. The reward is

calculated by subtracting the g_rxPktNum variable from the number of the last received packet using lastValue variable.

Creating open Gym Interface

```
1:  /*
2: Define reward function
3:*/
4: float MyGetReward(void)
5: {
6:    static float lastValue = 0.0;
7:    float reward = g_rxPktNum - lastValue;
8:    lastValue = g_rxPktNum;
9:    return reward;
10: }
```

Figure 4-10 MyGetReward function

MyExecuteActions function also uses an OpenGym Box container object which would get its vector properties from the action data container sent as numerical values that are mapped to their corresponding action as shown in Figure 4-11. The function then retrieves the current number of nodes and loops through them to execute an action (by calling SetCW function in Figure 4-12) on each node.

MyExcuteAction function

```
1:  /*
2: Execute received actions
3: */
4: bool MyExecuteActions(Ptr<OpenGymDataContainer> action)
5: {
```

```
6: NS_LOG_UNCOND ("MyExecuteActions: " << action);
7:  Ptr<OpenGymBoxContainer<uint32_t> > box = DynamicCast<OpenGymBoxContainer<uint32_t>
8:>(action);
9: std::vector<uint32_t> actionVector = box->GetData();

10: uint32_t nodeNum = NodeList::GetNNodes ();
11:  for (uint32_t i=0; i<nodeNum; i++)
12:  {
13:   Ptr<Node> node = NodeList::GetNode(i);
14:   uint32_t cwSize = actionVector.at(i);
15:  SetCw(node, cwSize, cwSize);
16:  }
17:   return true;
18: }
```

Figure 4-11 MyExecuteAction function

SetCW function

```
1: bool SetCw(Ptr<Node> node, uint32_t cwMinValue=0, uint32_t cwMaxValue=0)
2: {
3:     Ptr<NetDevice> dev = node->GetDevice (0);
4:     Ptr<WifiNetDevice> wifi_dev = DynamicCast<WifiNetDevice> (dev);
5:     Ptr<WifiMac> wifi_mac = wifi_dev->GetMac ();
6:     Ptr<RegularWifiMac> rmac = DynamicCast<RegularWifiMac> (wifi_mac);
7:     PointerValue ptr;
8:     rmac->GetAttribute ("Txop", ptr);
9:     Ptr<Txop> txop = ptr.Get<Txop> ();
10:    // if both set to the same value then we have uniform backoff?
11:    if (cwMinValue != 0) {
```

```
12:       NS_LOG_DEBUG ("Set CW min: " << cwMinValue);
13:       txop->SetMinCw(cwMinValue);
14:   }
15:   if (cwMaxValue != 0) {
16:  NS_LOG_DEBUG ("Set CW max: " << cwMaxValue);
17:  txop->SetMaxCw(cwMaxValue);
18:   }
19:   return true;
20: }
```

Figure 4-12 SetCW function

4.4.2 Proxy Component

The proxy component is implemented in Python as the part of the middleware that sits between the OpenAI Gym agent and the gateway controlling the environment. It receives information and function calls from the agent uses a zmq to send them to the ns3-gym framework to execute the make and step functions in the environment.

The Python imports gym library to allow the communication with the gateway component as shown in Figure 4-13.

Gym libraries

```
1:  #!/usr/bin/env python3
2:  # -*- coding: utf-8 -*-
3:  import time
4:  import gym
5:  import os
6:  import numpy as np
7:  from ns3gym import ns3env
```

```
8:  import matplotlib.pyplot as plt
```

Figure 4-13 Gym libraries

Although the environment has been fully defined in ns3 simulation script, we utilized the arguments of the Ns3Env constructor to pass the step time, start simulation flag, simulation seed, and simulation arguments as parameters as shown in Figure 4-14.

Simulation arguments

```
1:  # Environment initialization
2:  port = 5556
3:  simTime = 10 # seconds
4:  startSim = True
5:  stepTime = 0.005  # seconds
6:  seed = 0
7:  simArgs = {"--simTime": simTime,
8:                  "--testArg": 123,
9:                  "--distance": 20}
10: debug = False
11: env = ns3env.Ns3Env(port=port, stepTime=stepTime, startSim=startSim, simSeed=seed, simArgs=simArgs, debug=debug)
```

Figure 4-14 Simulation arguments

The code then iterates through a number of episodes specified within the script (as shown in Figure 4-15– Line 2), collects the environment state and current observation, then defines the next action.

An inner loop will get the current state, calculates the action array based on Numpy argmax function, then calls environment Step function passing the action array as a parameter, then adds the reward to the total reward variable being calculated for each episode. This loop continues until the done flag - indicating MyGameOver function returned true based on a condition, or the simulation time was over- is set to True. (Figure 4-15– Line 10 - 23). The episode reward is stored in Rewards array to be used later for plotting (Figure 4-15 – Line 25).

episodes iteration

```
1:  # Episodes
2:  for episode in range(episodes):
3:      # Refresh state
4:      state = env.reset()

5:      #state = calculate_q_diff(state)
6:      done = False
7:      t_reward = 0
8:      i = 0
9:      # Run episode
10:     while True:
11:         if done:
12:             break
13:         i += 1
14:         current = state
15:         for n in range(5):
16:             action[n] = np.argmax(Q[n, current[n], :] + np.random.randn(1, 100) * (1 / float(episode
+ 1)))

17:         state, reward, done, info = env.step(action)
```

```
18:          #state = calculate_q_diff(state)
19:          #print('state:', state, reward, done)
20           #print("action:", action)

21:          t_reward += reward
22:          for n in range(5):
23:              Q[n, current[n], action[n]] += alpha * (reward + discount * np.max(Q[n, state[n], :]) - Q[n,
current[n], action[n]])

24:    print("Total reward:", t_reward)
25:    rewards.append(t_reward)
26:    iterations.append(i)
```

Figure 4-15 episodes iteration

Once the simulation is over and all episode iterations are done, the environment is closed, and the results are then plotted using matplotlib library as shown in Figure 4-15. The total rewards array is split into chunks of size 5, then the average reward is calculated to be plotted for each episode.

4.5 A Q-Learning-Based Contention Window Adjustment in VANETs

4.5.1 Introduction

Vehicular-to-Everything (V2X) communications have a great potential to play a vital role for developing Intelligent Transportation System (ITS) where vehicles exchange messages with their surroundings and with each other to enhance roads' safety, efficient traffic management, and infotainment. V2X has different types of communication, including Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), Vehicle-to-Network (V2N), and Vehicle-to-Pedestrian (V2P), with different applications that necessary to support vehicular network to work. For V2V safety applications including but not limited to a Cooperative Collision Warning (CCW), Emergency Electronic Brake Light (EEBL), Cooperative Adaptive Cruise Control (CACC), periodic messages are broadcasting containing various real state information such as vehicles' speed, acceleration, direction, position, size and others to the surrounding vehicles known as Basic Safety Messages (BSM) based on SAE J2735 (SAE J2735 DSRC Message Set, 2009). During high traffic densities with high numbers of vehicles in a specific geographical range, would create an enormous amount of BSM traffic that will be broadcasting at the same time from vehicles within distance limited communication rage, it will increase load on wireless channel access, causing packets collision, longer delay, network throughput reduction and decreasing packet delivery ratio.

To send BSM traffic in V2V channel communications, V2X uses a Media Access Control (MAC) protocol that is based on a Carrier Sense Multiple Access protocol with Collision avoidance (CSMA/CA) with Binary Backoff Exponential (BEB) mechanism for contention-based access channel that decides when the vehicles can access channel medium to minimize the interference or collision probability among contending vehicles to increase Packet Delivery Ratio (PDR). In CSMA/CA a vehicle wishes to transmit data

should start listen to the channel medium before transmission and if the channel is free for a predefined period called Arbitrary Interframe space (AIFS, the duration varies according to vehicles priority), the vehicle can start transmitting the data. If the channel is detected busy during channel listening period, the vehicle will invoke exponential backoff algorithm. The vehicle shall postpone the transmission by generating random backoff time by drawing an integer from uniform distribution (0, CW), where is CW is the current contention window which initially set to minimum value, CW_{min} (CW refers to number of waited slots, and for 10Mhz channel the slot time is 16μs) and then decrease CW timer when the channel is detected free. Once the timer is reaching 0 and no transmission is detected in the channel, the vehicle can start the transmission, otherwise the backoff will be invoked and the CW will be doubled for the next transmission. This process will be continued until receiving acknowledgement (ACK) message denoting that the data have been received successfully or by reaching maximum CW_{max} value. In other words, for every attempt to send packets, the CW value will be doubled from initial value CW_{min} until reaching CW_{max}.. In broadcast mode where no ACK message is sent by the receiver, the backoff algorithm will be invoked only one time during first listening of the channel, therefore the CW will be set to the minimum value, CWmin, and it will never be doubled (ETSI, Intelligent Transport Systems (ITS);European profile standard for the physical and medium access, 2009), the choice of backoff values will be (0, CWmin). when CWmin is small, the vehicles will have high opportunity for not waiting too many time slots (chosen CW) to access the channel medium for sending their data. This scenario is good in the sparse mode where small number of vehicles are contending to access the channel and the backoff waiting time will be less and this will enable the vehicles to send their data faster. But in dense scenario where there are many vehicles like an urban environment or traffic jam, using

small CW_{min} will increase the probability of two or more vehicles to choose same CW and start transmitting simultaneously leading to packet collisions and bandwidth wastage.

It is obvious that the CW value is significantly can affect the performance of VANET (Reinders, Eenennaam, Karagiannis, & Heijenk, 2011) (Pressas, Ali, Tian, & Nekovee, 2017) where large number of vehicles could be contending among each other to send the BSM traffic. In this chapter the focus will be in the V2V communication where the BSM traffic will be disseminated among the vehicles and the backoff algorithm will be invoked only one time to transmit the packets using CW_{min} as a minimum and maximum CW value. Choosing CW_{min} value every time as an initial CW for broadcasting BSM traffic for all vehicles will increase the probability of selecting same CW by multiple vehicles and start transmitting their data simultaneously. As a result, multiple packets' collision will happen and this will decrease PDR, increase delay, and decrease network throughput.

The primary goal of this chapter is a mechanism that learns how to adjust CW value in V2V communication while broadcasting BSM traffic to improve PDR in dense network, thus utilize the channel bandwidth efficiently. The Q-Learning algorithm will be used because of it is ability adjust the CW value through trial-and-error interaction with the environment, with no prior knowledge or following analytical models. It is targeting reliable PDR and enhance network throughput

4.5.2 Q-Learning MAC Protocol Design

Q-Learning algorithm (Watkins & Dayan, 1992) is a classic reinforcement learning algorithm that works in any environment and learn the

environment's behavior through trial-and-error interaction. It is based on a smart agent that has the capability of interactive learning by taking an action that is most beneficial in each observed state to maximize a numerical value called reward following partially observable Markov decision process (POMDP). POMDP is mathematical framework to represent optimization decision-making process of Q-learning agent and it is suitable for modeling channel access problem in V2V communication (CHOE, et al., 2020) since vehicles cannot observe perfect status information of other vehicles. The POMDP model that represents channel access problem in V2V communication to adjust CW value in backoff algorithm consists of (s, a, r, s') tuple where (s) denotes to environment state, (a) for action, (r) for reward numerical value and (s′) for next environment state after taking the action. Figure 4-16 shows the reinforcement learning structure based on adjusting CW, where each vehicle in the network act as an independent agent, the agent observes the environment states and decide to take action based on its experience interacting with the environment. After taking the action on the environment, a reward and next state will feedback to agent. Through interactions between the agent and the environment, the agent can make prediction of rewards from past learning experience and take actions that maximize rewards. Q-learning system has several elements or components that need to be defined to solve channel access problem in V2V.

- Agent: Every vehicle is considered as an independent agent.
- State (S): Number of packets in vehicle queue which represent the network load. The queue size set to 100 packets.

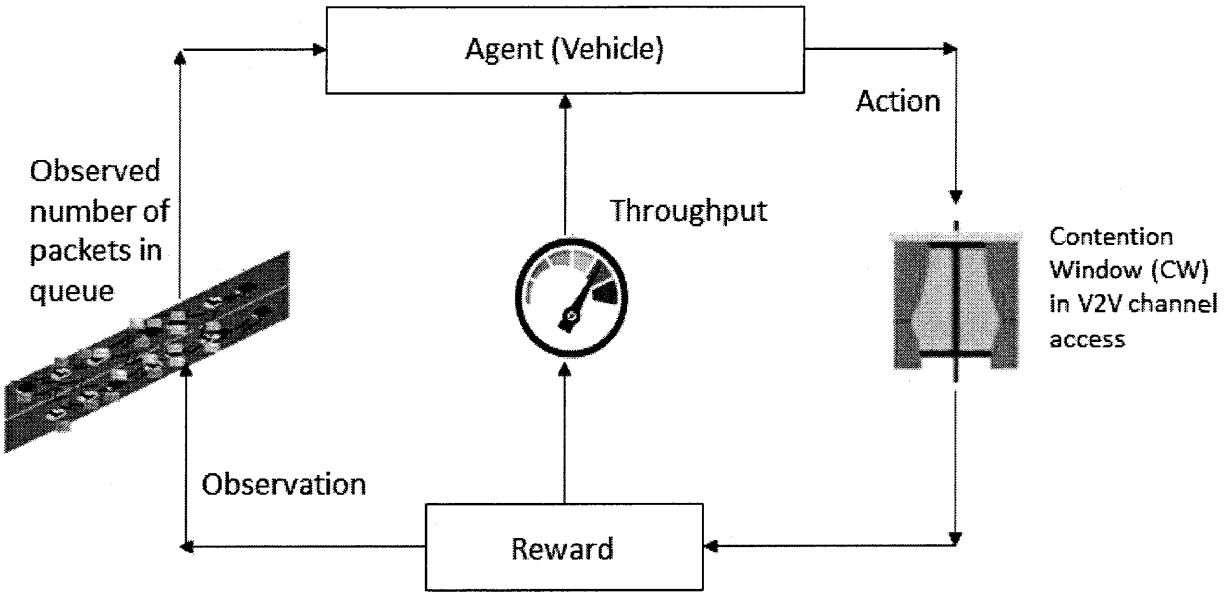


Figure 4-16 Reinforcement learning based CW adjustment in V2V communication

- **Action (A)**

  Set CW value following backoff algorithm in V2V broadcast mode the vehicle has 7 actions [15, 31, 63, 127, 255, 512, 1023] to select. will act as CW_{min} and CW_{max} will be set to same value (in V2V broadcast communication the CW value stays constant even when packet collision happens)

- **Policy (π)**

  The learning agent's behavior when interacting with the environment to take specific action (a_t) in a specific state (s_t). It is like a mapping state with action (π: s $\rightarrow$ a).

- **Reward (r)**

  The number of packets received at the flow's destination during last step interval. On each time step, the environment sends to the agent a value called reward, the main goal of the agent is to maximize total

rewards over the long run. The reward signal is very important to Q-learning agent, since the agent sometimes decides to change the policy based on the reward, if the current taken action by the agent is resulting a low reward, then the agent has to change its policy and try other action.

- **Q- value function or Quality function**

  Q (s, a): Quality of State/Action pair.

  Q-function is the joint quality or value of taking an action (a) given a state (s). The Q-function is an indicator what is good in the end, it is the total amount of reward an agent can expect to accumulate over the future starting from that state (Sutton & Barto, 2018).

The primary aim of RL is to determine the action that maximizes the policy for every state. To accomplish this objective, The Q-learning is used to select the optimal policy by learning the optimal Q-values for each state-action pair. The Q-learning algorithm updates the Q-values for each state-action pair iteratively using the Bellman equation until the Q-function reaches the optimal Q-function Q* as shown in equation 4.1.

$$Q^*(s,a) = E\left[r_{t+1} + \gamma \max Q\left(s_{t+1}, a_{t+1}\right)\right] \qquad (4.1)$$

In Q-learning, the agent builds a table called Q-table of Q (S, A), where S is the set of all possible states and A is the set of all possible actions. The Q-table has columns represent set of actions and rows represent set of states and filled with Q-value Q (s, a) which represent rewards (R) for moving from state (s) to next state (s') by taking action

(a) as shown in equation 4.2 where u represents the states in the system s = {1,…, u} and v represents actions per state a = { 1,…,v}.

$$(Q) = \begin{pmatrix} Q(1,1) & Q(1,2) & \dots & Q(1,v) \\ Q(2,1) & Q(2,2) & \dots & Q(2,v) \\ \dots & \dots & \dots & \dots \\ Q(u,1) & Q(u,2) & \dots & Q(u,v) \end{pmatrix} \qquad (4.2)$$

At each time step (t), the agent selects action (a_t) at state (s_t) and get the reward value (r_t) based on the reward function, then observe the next state (s_{t+1}). Using the Q-learning experience (s_t, a_t, r_t, s_{t+1}) to update the Q-table records Q (s_t, a_t), the Q-learning algorithm uses equation 4.3 to update Q-table with Q (s_t, a_t).

$$Q^{new}(s_t, a_t) \longleftarrow Q^{old}(s_t, a_t) + \alpha\,[r_t + \gamma * \max_{at+1} Q(s_{t+1}, a_{t+1}) - Q^{old}(s_t, a_t)] \qquad (4.3)$$

During the Q-learning process, the Q-table updates Q values for all states when a reward is received from the environment following algorithm 4.1. In equation 4.3, α is the learning rate which quantifies to what extent the new Q (s_t, a_t) will overwrite the old Q (s_t, a_t), $\alpha \in [0,1]$, α=0 means the agent is not learning any new information. γ is discount factor that models the importance of future rewards compare to the immediate rewards, $\gamma \in [0,1]$, $\gamma = 0$ means the agent considers only the current reward while larger γ means the agent is considering the future rewards in computing Q-function.

Algorithm 4.1: Update Q-table

1 Initialize $Q(s,a) = 0$ for all $s \in S$, $a \in A$

2 Loop for each episode:

3 Initialize S

4 Loop for each step $t = 0,1,\ldots$ of episode:

5 Choose A from S using policy $\boldsymbol{\pi(s, a) = \arg_a \max Q(s,a)}$

6 Take action A, observe R, S'

7 $Q(S_t, a_t) \leftarrow Q(S_t, a_t) + \alpha[r_t + \gamma * \max_a Q(S_{t+1}, a) - Q(S_t, a_t)]$

8 $S \leftarrow S'$

9 until S is terminal

4.5.3 Performance Evaluation

This section presents simulation experiments for evaluating the Q-learning algorithm to adjust CW value in V2V channel access in VANET. The Q-learning algorithm compared to V2V traditional broadcasting communication in a freeway scenario under diverse traffic density with various BSM packet size and different packets generation rate.

4.5.3.1 Simulation Setup

Ns3-gym (Gawłowicz & Zubow, 2019) used to conducts simulation in highway scenario. VANET simulation environment are shown in Table 4-1. We consider a 300 x 1500 highway, where vehicles are moving based on random way-point mobility model with a speed of 20 m/s. Vehicles will

broadcast 200, and 500 bytes messages 100 times per seconds at 6Mbps as a data rate.

Table 4-1: Simulation parameters

Parameters	Value
Simulation time	60s
Topology	300 * 1500m
Channel Frequency	5.89GHz
Channel Bandwidth	10Mhz
Transmission rate	250
MAC	WAVE IEEE802.11P
Data rate	6Mbps
BSM Packet size	200, 500B
WAVE interval	0.01s
Propagation model	Nakagami model
Mobility model	Synthetic highway scenario (RWP)

Table 4-2 shows Q-learning parameters that used in the simulation.

Table 4-2: Q-Learning parameters

Parameters	Value
Learning Rate α	0.75
Discount Rate γ	0.95
Episodes	100
Step time	0.005

4.5.3.2 Simulation results

The Q-learning algorithm was compared with the traditional broadcasting scheme used in V2V communication to broadcast BSM messages to the Surrounding vehicles. The simulation used different packet size, different number of nodes and different distance ranges to evaluate traditional broadcasting V2V scheme and Q-Learning in terms of network throughput and Packet Delivery ratio (PDR) as performance metrics.

Figure 4-17 and Figure 4-18 show the total system throughput for transmitting BSM packet size 200 and 500 Byte respectively compared to various number of vehicles.

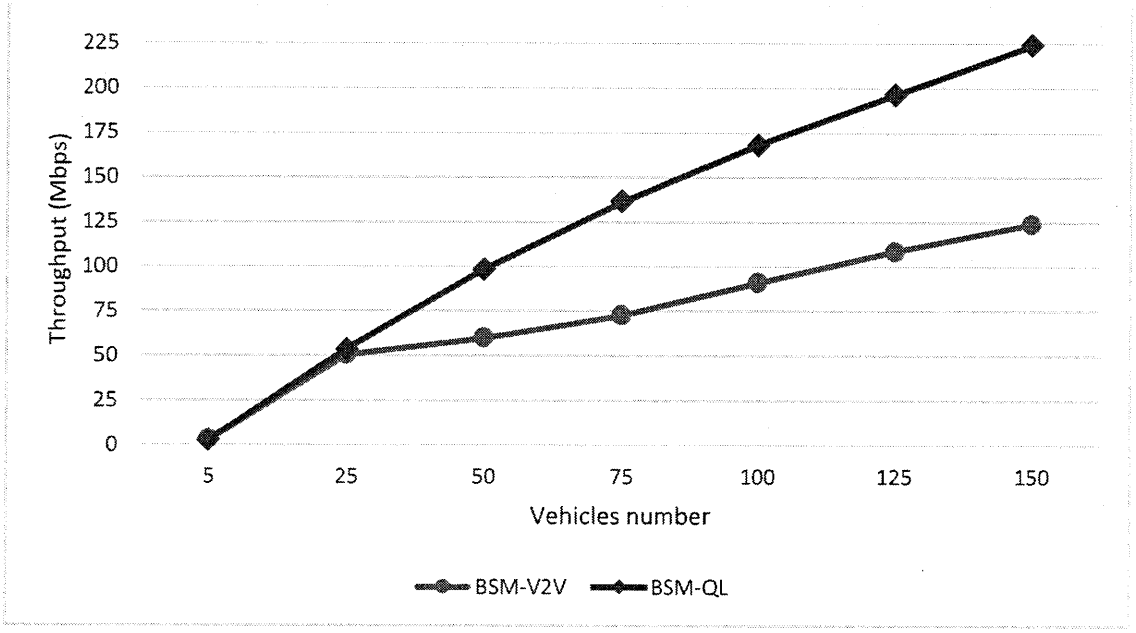


Figure 4-17 System throughput versus vehicles number with packet size is 200 bytes

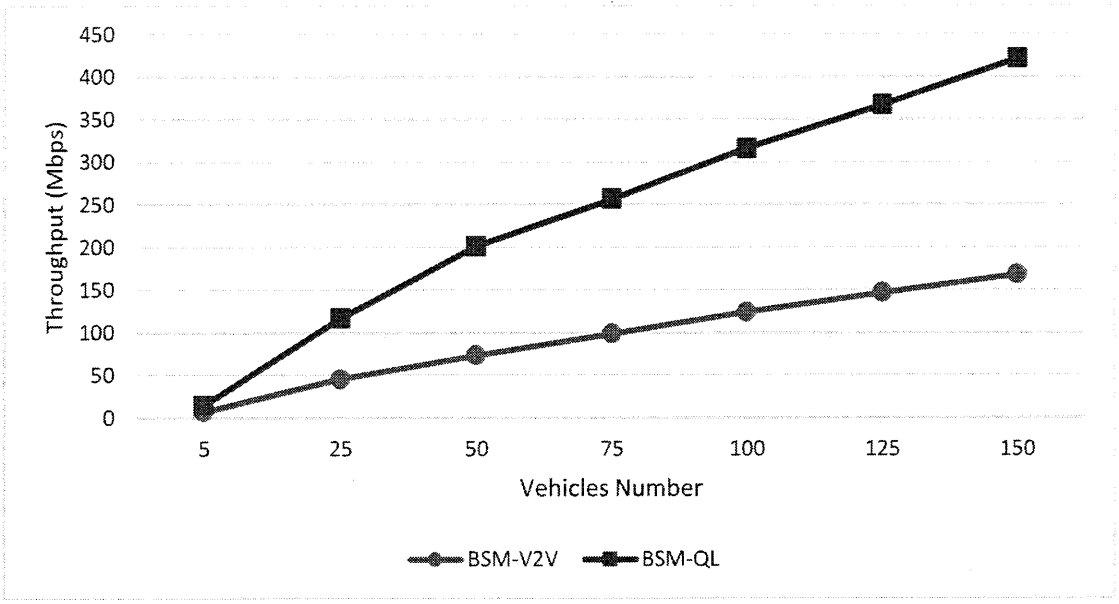


Figure 4-18 System throughput versus vehicles number with packet size is 500 bytes

Figures 4-17 and 4-18 show the comparison of the proposed Q-Learning algorithm with the traditional broadcasting BSM scheme with fixed CW value (CW_{min}) in terms of throughput. The throughput for traditional scheme is decreasing as the number of vehicles increases in different packet size, while the Q-Learning scheme achieves better throughput especially when the number of vehicles is increasing. Q-learning is adjusting CW value based on the number of packets in the queue to minimize probability of collisions while broadcasting BSM traffic compared to the traditional scheme that keeps fixed CW value for broadcasting the BSM traffic.

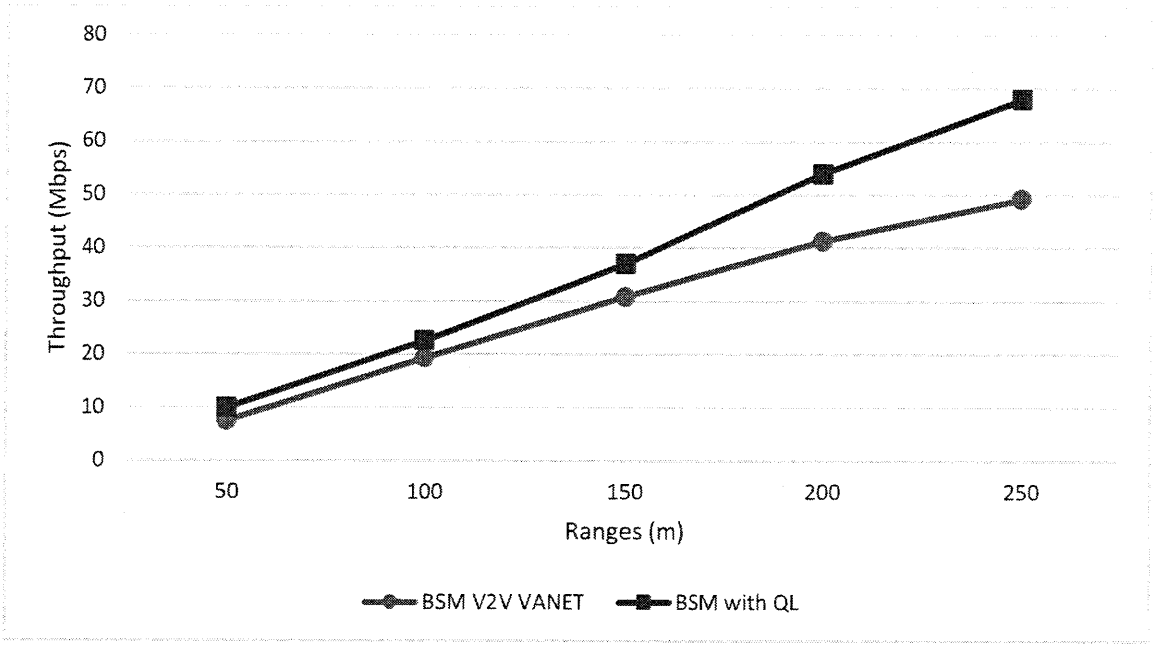


Figure 4-19 Throughput versus different distance ranges with packet size is 200 bytes for 75 vehicles

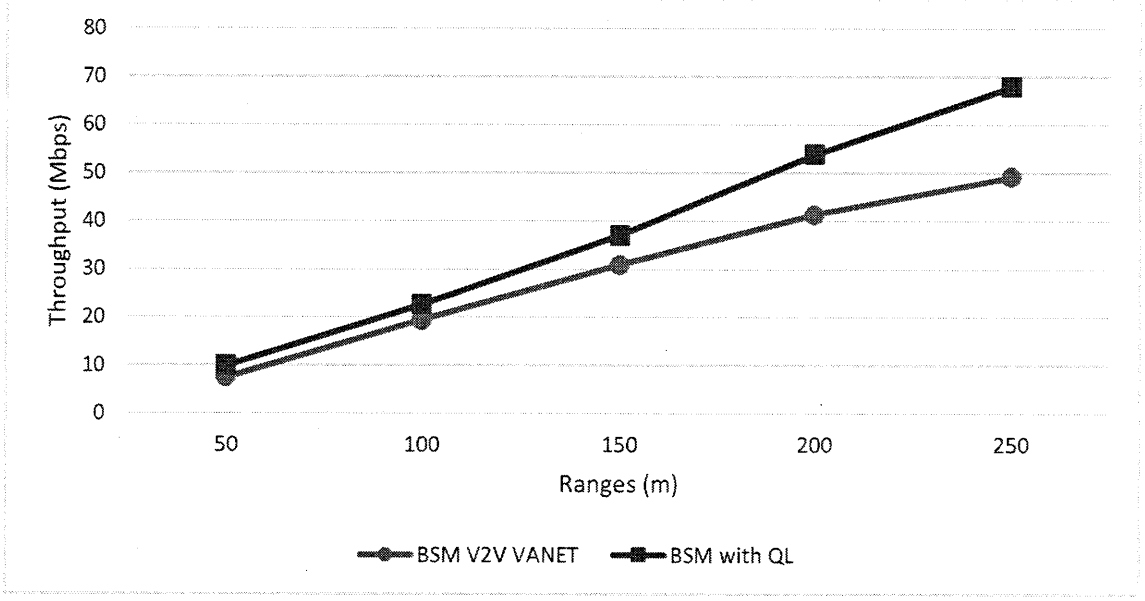


Figure 4-20 Throughput versus different distance ranges with packet size is 200 bytes for 75 vehicles

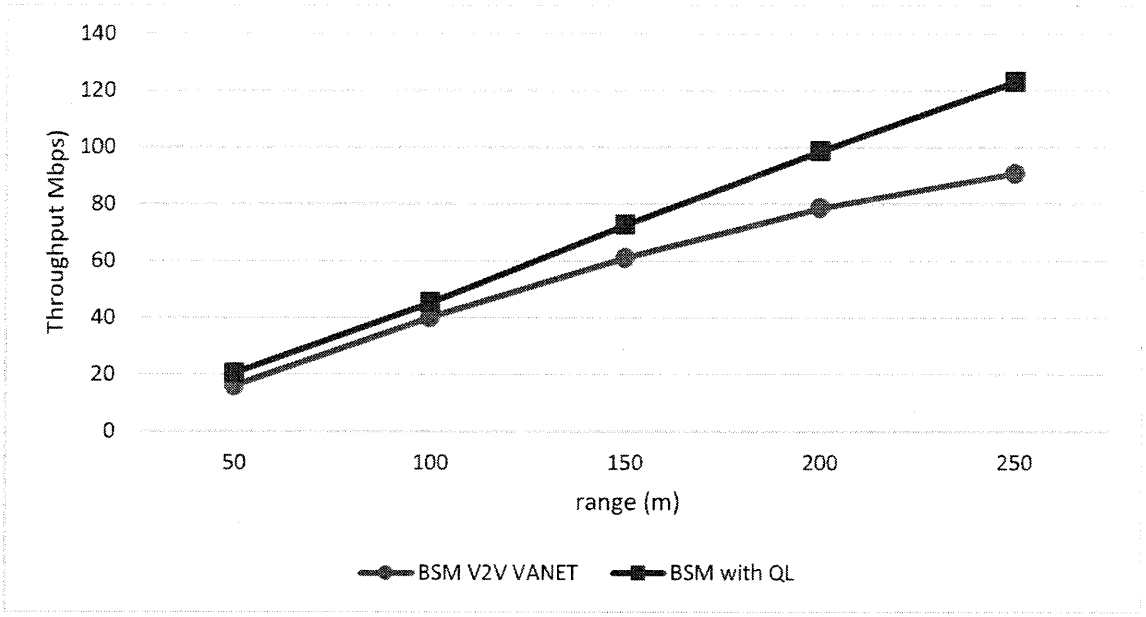


Figure 4-21 Throughput versus different distance ranges with packet size is 200 bytes for 150 vehicles

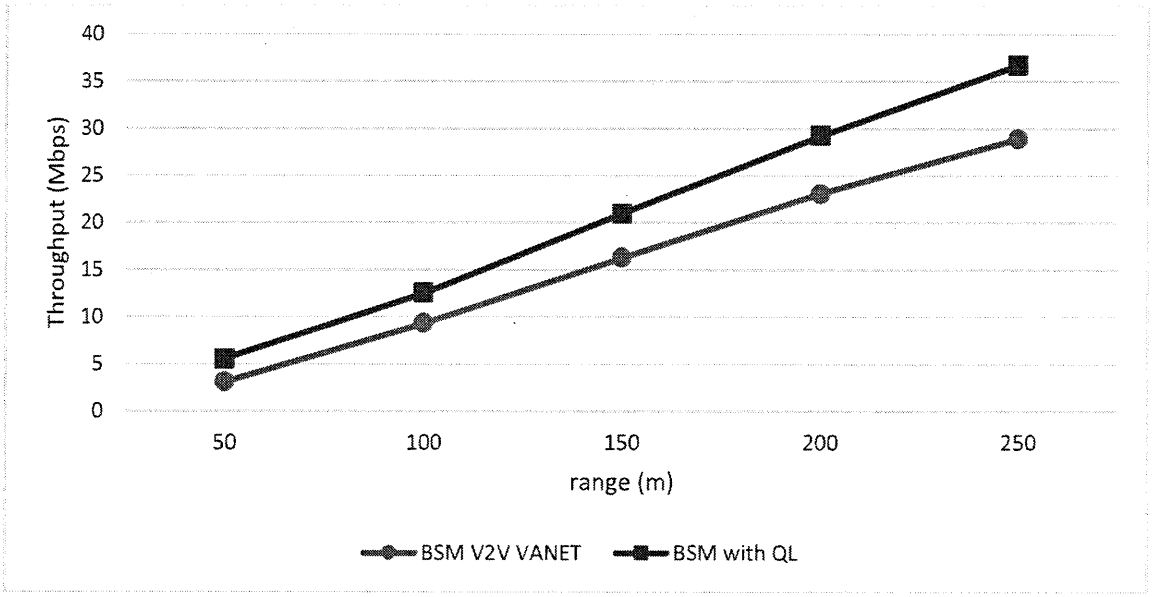


Figure 4-22 Throughput versus different distance ranges with packet size is 500 bytes for 25 vehicles

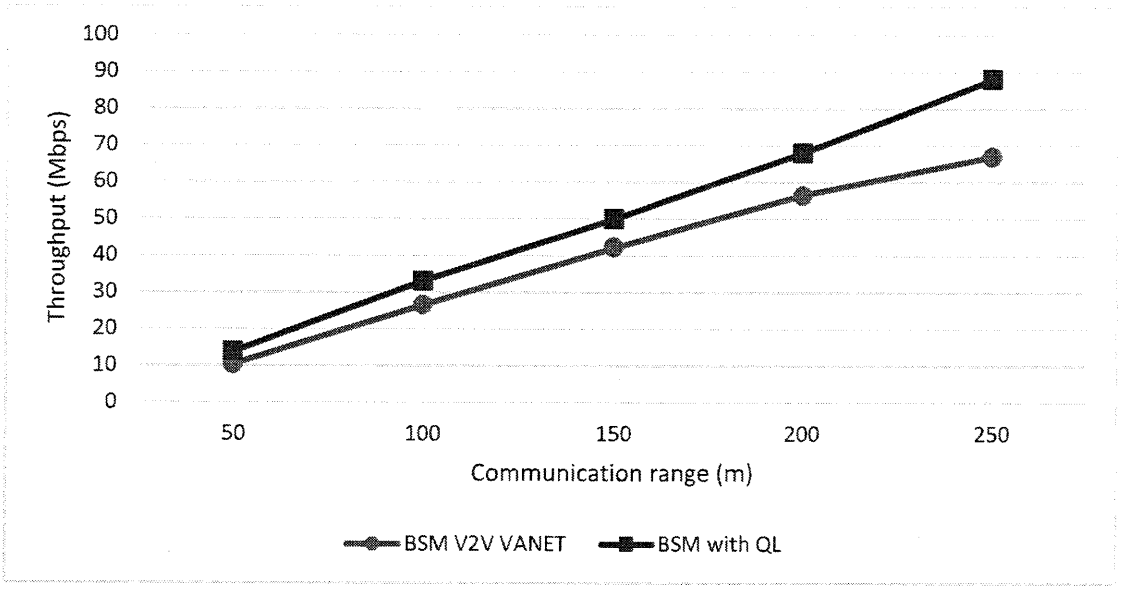


Figure 4-23 Throughput versus different distance ranges with packet size is 500 bytes for 75 vehicles

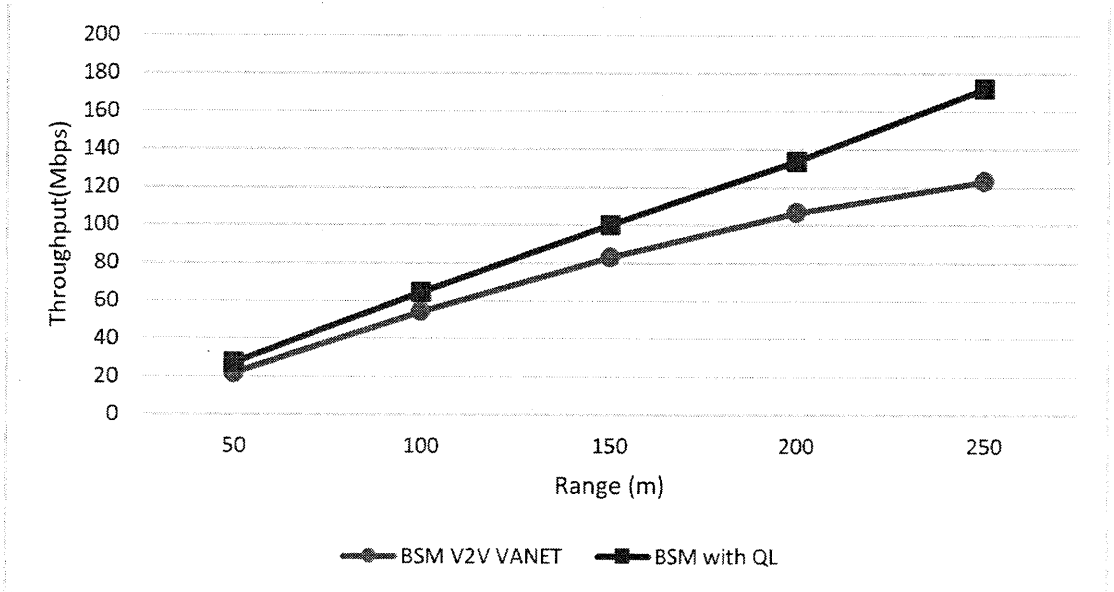


Figure 4-24 Throughput versus different distance ranges with packet size is 500 bytes for 150 vehicles

Figures 4-19 – 4-24 show a comparison of proposed Q-Learning with traditional broadcasting scheme in terms of throughput for 200 and 500-byte packet size. The horizontal axis represents the distance ranges, while the vertical axis represents the throughput. The Q-learning has outperformed the traditional scheme especially when the broadcasting traffic is increased in term of vehicles or number of contending vehicles to broadcast BSM traffic.

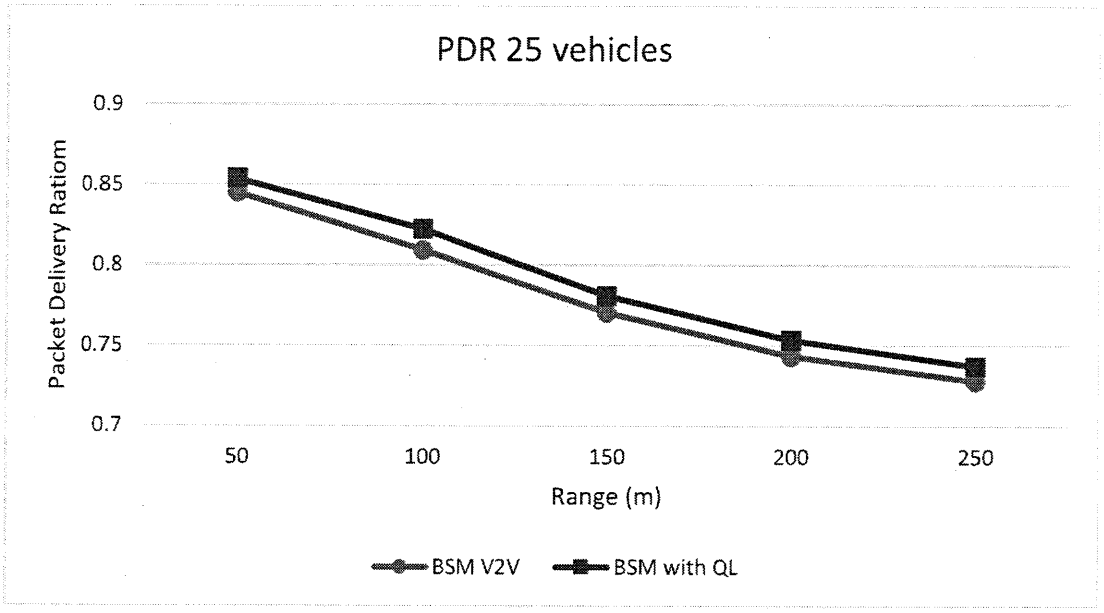


Figure 4-25 PDR versus different distance ranges with packet size is 200 bytes for 25 vehicles

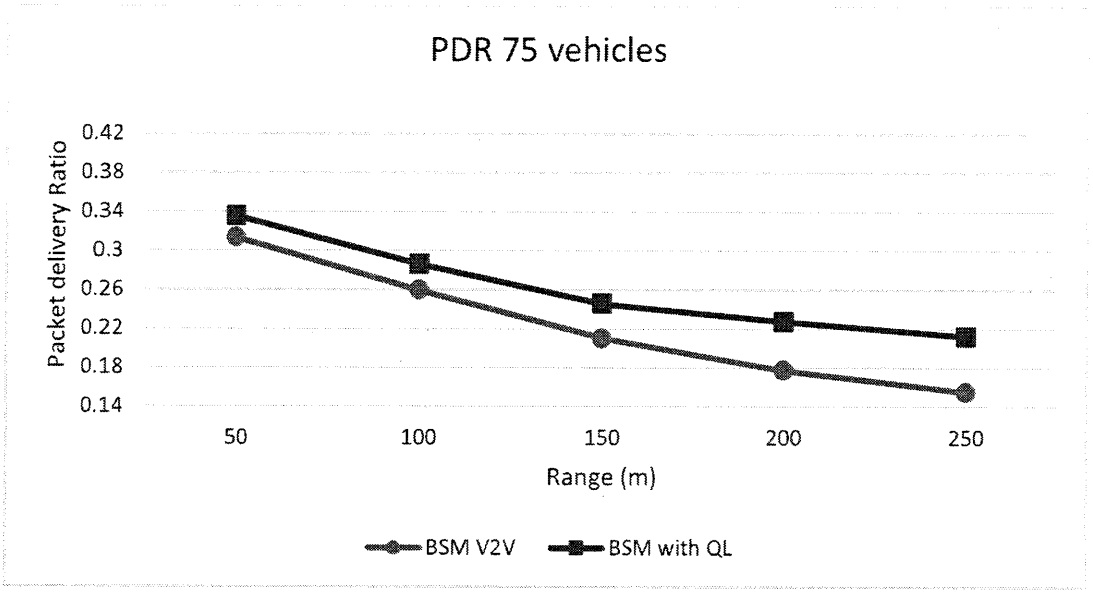


Figure 4-26 PDR versus different distance ranges with packet size is 200 bytes for 75 vehicles

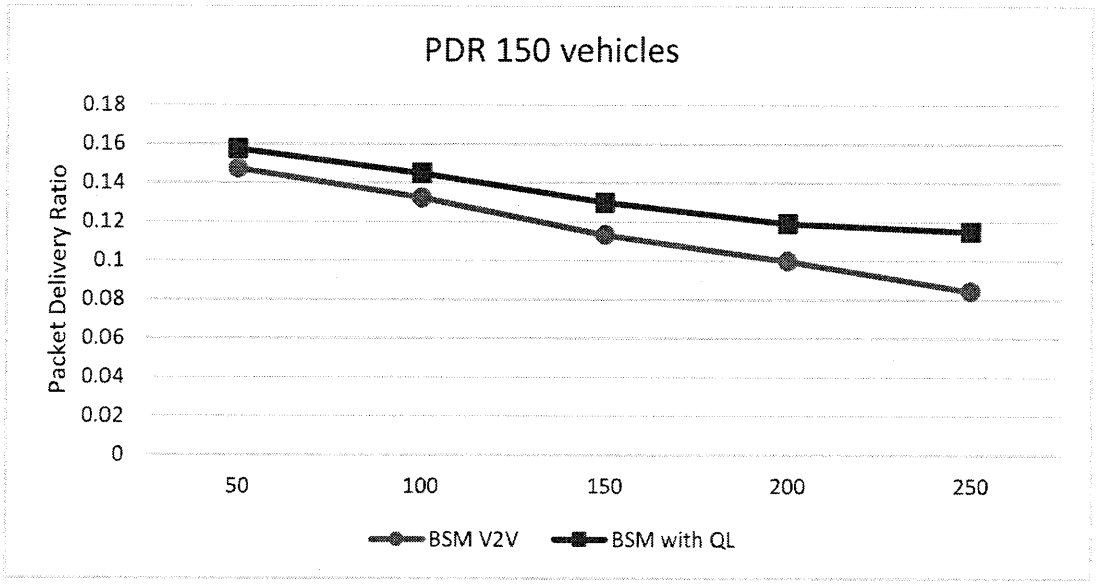


Figure 4-27 PDR versus different distance ranges with packet size is 200 bytes for 150 vehicles

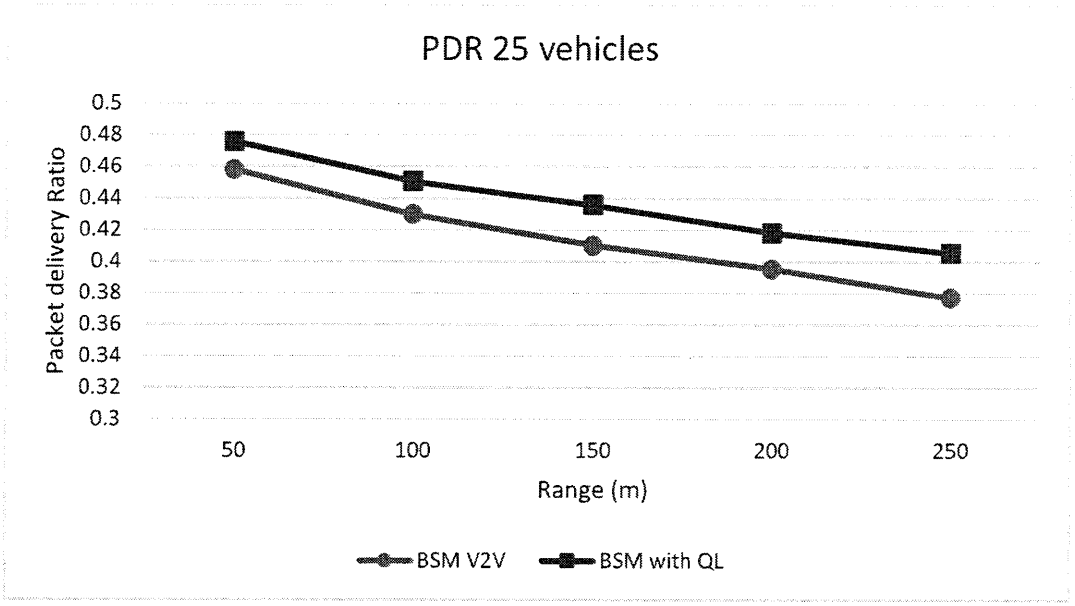


Figure 4-28 PDR versus different distance ranges with packet size is 500 bytes for 25 vehicles

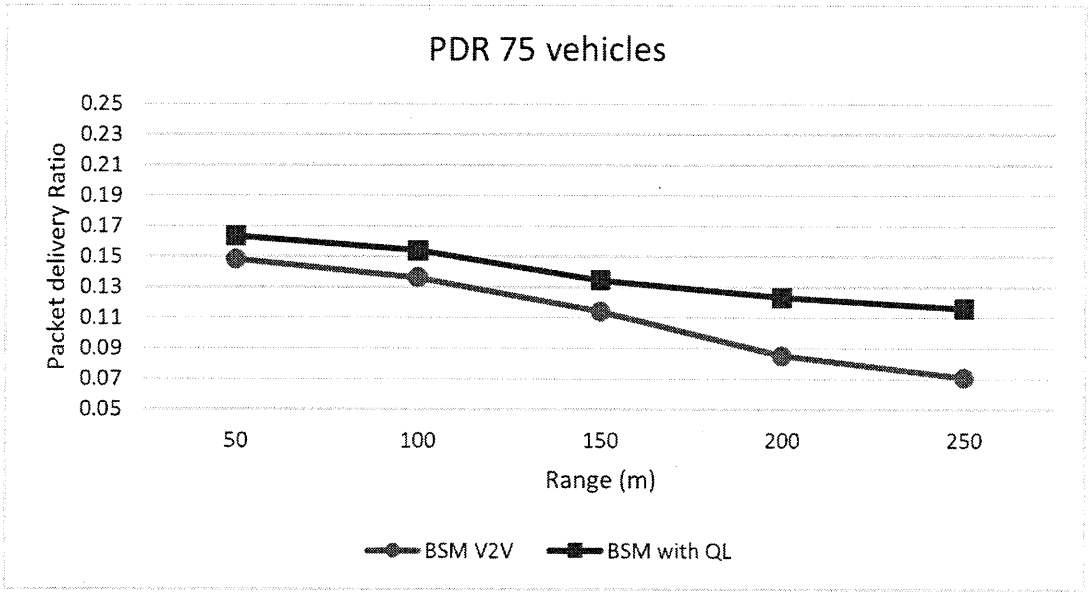


Figure 4-29 PDR versus different distance ranges with packet size is 500 bytes for 75 vehicles

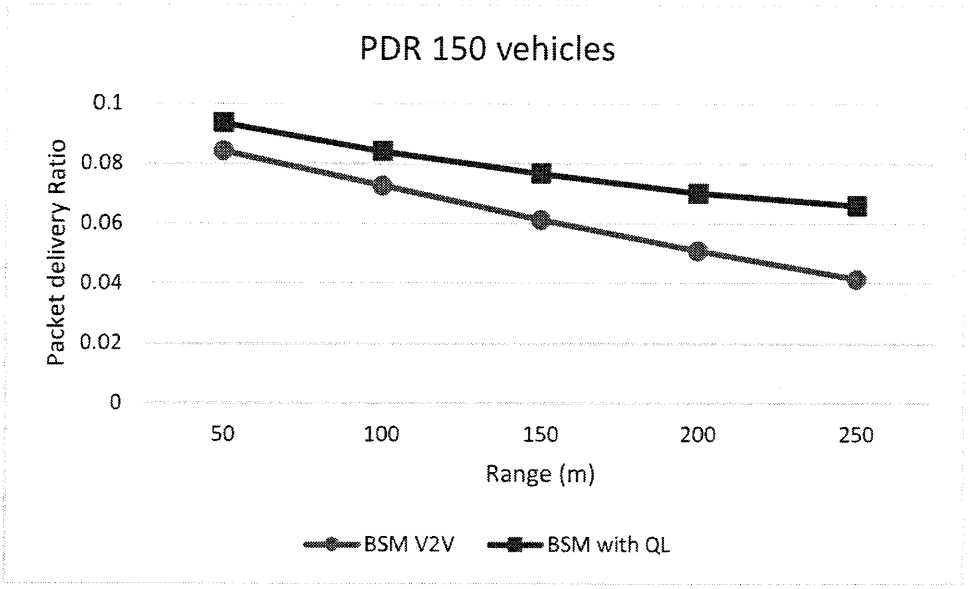


Figure 4-30 PDR versus different distance ranges with packet size is 500 bytes for 150 vehicles

Figures 4-25 – 4-30 show a comparison of proposed Q-Learning with traditional broadcasting scheme in terms of PDR for 200 and 500 packet size. The horizontal axis represents the distance ranges that vehicles can cover, while the vertical axis represents the PDR. It is obvious that the Q-learning has achieved higher PDR compared to traditional broadcasting scheme especially in the high traffic since the Q-learning is adjusting CW value and as a result the collision rate will be decreased compared to traditional scheme.

4.6 Conclusion

This chapter presents the utilization of Q-learning for enhancing the VANET throughput and PDR by adjusting the CW value during the broadcasting of BSM traffic in V2V communication. The traditional broadcasting scheme, which employs a fixed CW value, is compared to the proposed QL approach. Simulations are conducted to demonstrate the

effectiveness of using QL in V2V communication for broadcasting BSM traffic. By adapting the CW value using Q-learning, the proposed approach aims to minimize packet collisions and drops in congested V2V communications, surpassing the performance of the traditional scheme based on CSMA/CA. Particularly in high-traffic scenarios, the QL approach proves to be superior, enabling the network to achieve a relatively reliable PDR and high throughput. The simulation results validate the effectiveness of QL in enhancing the performance of V2V communication for broadcasting BSM traffic.

By integrating RL into the VANET environment, the agent can directly interact with the environment and learn about the system without prior information. This allows the agent to solve or optimize complex VANET systems. The results demonstrate the effectiveness of RL when compared to traditional broadcasting schemes commonly used in VANET scenarios. RL enables the agent to learn and adapt its behavior over time, leading to improved performance and better system optimization in VANET environments.

Chapter 5

Optimizing Contention Window in VANET using DQN and Actor-Critic

Chapter 5 Optimizing Contention Window in VANET using DQN and Actor-Critic

5.1 Introduction

VANET is considered as a promising solution to meet the intelligent transportation systems' requirements by implementing different types of applications that improve roads safety, enhance future transportation systems, traffic efficiency and increase driver and passengers' comfort. The IEEE 802.11p is an approved amendment to IEEE 802.11 standard that defines medium access control (MAC) and physical layers (PHY) specifications of the wireless access for Wireless Access in Vehicular Environments (WAVE) protocol stack.

IEEE 802.11p standard adopts Enhanced Distributed Channel Access (EDCA) which specifies the distributed contention-based channel Access scheme similar to EDCA in IEEE 802.11e (IEEE 802.11 Working Group, 2005) to support different level of equality of services (QoS). EDCA is considered an enhancement variant of legacy Distributed Coordination function (DCF) scheme to provide prioritized services. Both schemes are relying on Carrier Sense Multiple Access with collision Avoidance (CSMA/CA) (Kim & Lee, 1999) with a slotted Binary Exponential Backoff (BEB) mechanism for contention-based access channel to reduce the collision probability among multiple contending nodes that trying to access the medium to send their frames. The main differences between DCF and EDCA are the Inter-Frame Spacing (IFS) which is the time interval between data frames and the size of Contention window (CW) (Nguyen, et al., 2018), EDCA is used an Arbitrary Interframe space (AIFS), while DCF interframe space (DIFS) is used in DCF scheme.

According to (802.11 WG - Wireless LAN Working Group, 2007) For a vehicle to transmit date shall invoke the Carrier Sense (CS) mechanism where it needs to sense the wireless medium (WM) to determine if another vehicle is transmitting. If the medium is determined to be an idle for an AIFS period, the vehicle can procced with its transmission. If the medium is sensed to be busy, the vehicle shall defer the transmission by generating random backoff time using Backoff procedure as Equation 5.1.

$$BT = Rand\ (n) * SlotTime \tag{5.1}$$

Where $Rand\ (n)$ = Random integer is selected between 0 to CW, where CW is an integer within the range of values of the PHY characteristics CW minimum (CWmin) and CW maximum (CWmax), where

$$CWmin \leq \text{CW} \leq CWmax$$

In EDCA for our traffic, the CWmin and CWmax will be 15 and 1023.

Slot Time = The time of one slot taken from PHY characteristics which is equal to 16μs in EDCA scheme.

The CW value shall take the initial value of CWmin. The set of CW values shall be sequentially ascending integer powers of two, minus 1 as shown in equation 5.2, beginning with CWmin value, and continuing up until it reaches the CWmax as shown in Figure 5-1. The CW's exponential increment shall be continued every time an unsuccessful attempt to transmit the frame, until the CW reaches CWmax. The chosen CW value shall be

rested to CWmin after a successful unicast frame transmission and receiving the acknowledgment (ACK) frame, or after exceeding retry limit based on the standard.

$$CW = 2^x - 1 \tag{5.2}$$

Where x is the backoff exponent, it is written in format of (4 + a) to satisfy the CW range [15 - 1023], where a is an integer from zero to 6.

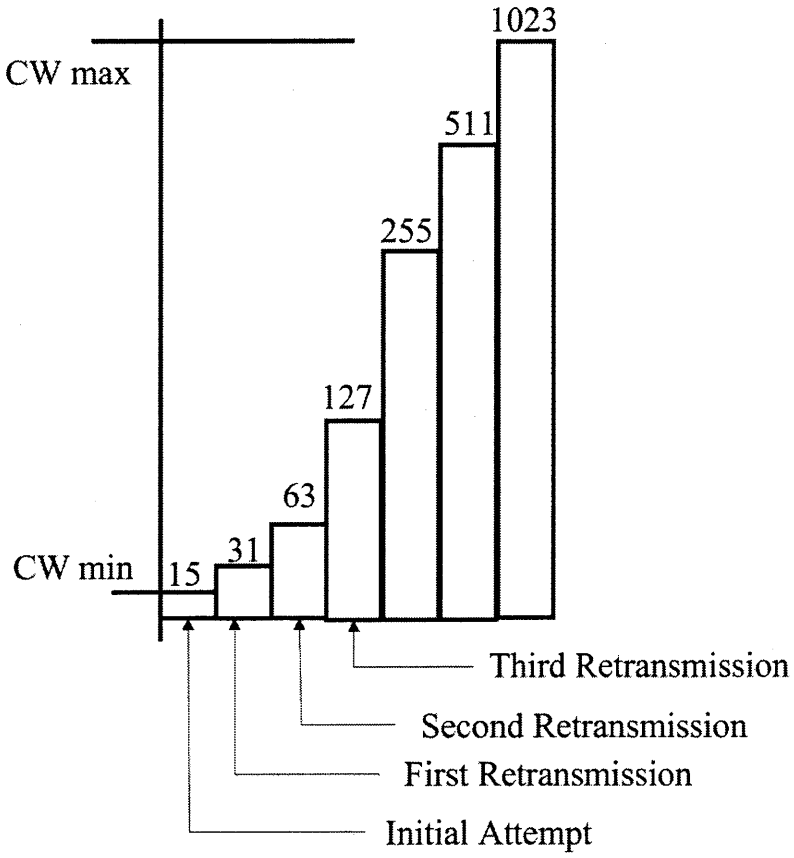


Figure 5-1 Example of exponential increase of CW in CSMA/CA. Taken from IEEE Std 802.11-2007. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (Physical) specification

When the vehicle senses the WM to be busy, it will begin the backoff procedure as shown in Figure 5-2, the vehicle sets the backoff time based on

equation 5.2 by selecting a random value from 0 to the chosen CW value. If the medium is sensed to be idle for an AIFS duration, then the backoff procedure shall decrement its backoff time by slot time. Transmission shall start when the backoff time reaches zero.

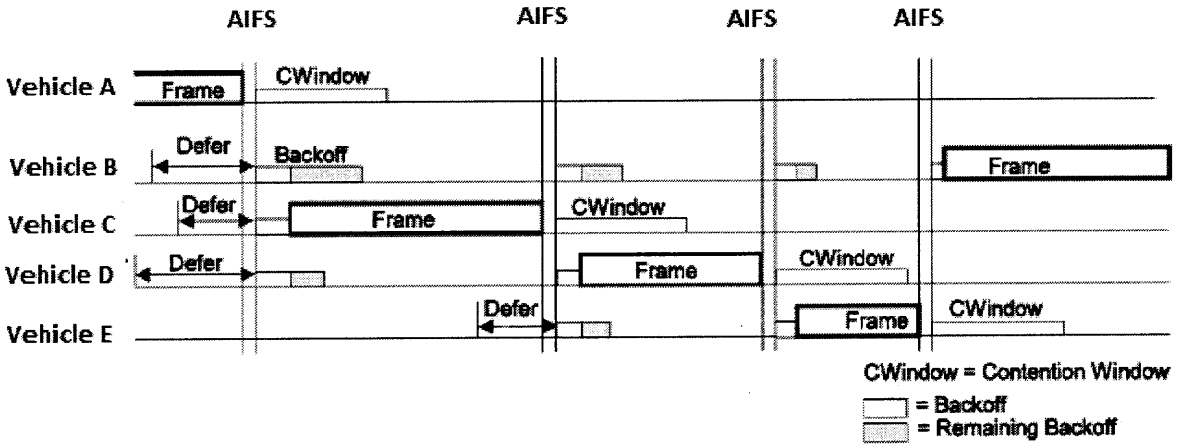


Figure 5-2 Backoff Procedure in CSMA/CA. Taken from IEEE Std 802.11-2007. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (Physical) specification

For example, let's say that vehicle A has data to be sent and received with Vehicle B in unicast form as shown in figure 5-3. First, the vehicle A senses the WM to check the availability of the channel medium and needs to wait for AIFS duration and then, select the Random backoff from 0 to CW which is initially set to 15. The backoff timer decremented until reaches 0, if the medium was sensed idle, the vehicle will start sending the frame. If the medium was sensed busy, Vehicle A will exponentially increase CW to be 31 and selects the random backoff in the range of [0 – 31], this process will continue until receiving successful ack or reaches the retry limit.

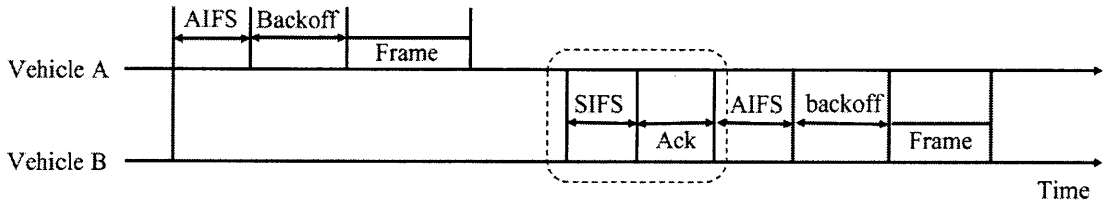


Figure 5-3 CSMA/CA managing channel access in unicast transmission among Vehicles

A small CW value will enable the vehicles to transmit their data quickly without waiting long time when the channel sensed to be an idle for backoff time. This scenario of selecting a small CW is preferable in the sparse VANET environment, where the number of contending vehicles are small consequently, it will not miss transmission opportunities. Selecting a small CW will affect the network performance positively in the sparse environment, since the vehicles will not wait longer than needed, then delay time will be less and will enhance bandwidth utilization. But in the dense environment where many vehicles are trying to access the channel for transmission, the small CW will increase the probability of multiple vehicles select same CW value after they are finding the channel idle and trying to transmit at the same time, in the result, collisions will occur and leads to bandwidth wastage.

In Addition to what precedes, CSMA/CA bringing unfairness problem in the channel contention (Li, Tang, & Chang, 2009) (Wang & Giannakis, 2005) (Chetoui & Bouabdallah, 2007). Vehicles cannot get access to the WM equally during dense environment due to that each vehicle relies on its own direct experience in estimating the congestion, which often leads to asymmetric views. The fairness problem arises when using CSMA/CA mechanism under high traffic conditions where the vehicles that sent their data successfully, they will reset the CW to CWmin and give them high chance to

access the WM. Other vehicles who have experienced collisions will keep increasing CW value exponentially, thus reducing their chance to access the WM and significantly decreasing VANET throughput.

Consequently, an efficient mechanism is required to dynamically adjust the CW value based on different VANET environments to overcome the network collision and fairness problem. The proper setting of CW value has a significant effect on VANET's performance, especially in dense environment where many vehicles in same location are trying to access the channel for data transmission. The optimal CW value depends on various factors such as, number of vehicles in the same vicinity, network traffic load, vehicles' mobility, roads' topology, application latency tolerance, and QoS.

Unfortunately, the channel access mechanism used in VANET is not efficient and scalable to keep stable network throughput for different traffic load. The smaller the CW, the higher probability to access the channel and start the transmission. However, due to the nature of VANET network where at any times the number of vehicles can increase significantly, small CW can cause multiple vehicles to select same backoff time and start sending at same time and ending with collision and affect bandwidth utilization. On the other hand, A high CW reduce the probability of vehicles to select same backoff time, consequently, reducing the probability of collision and increase throughput. But in some scenarios, high CW will increase waiting time and vehicles miss the opportunity to send their data and wasting channel bandwidth.

The main objective of this chapter is to adjust the CW in VANETs to enhance VANET's data delivery in congested network, eliminate fairness problem among vehicles, thus efficiently utilize the channel bandwidth. DRL have been used for this purpose because of their ability to select optimal CW value by interacting directly with VANET environment, without any prior

knowledge of the environment. DRL is a branch of ML that is suitable for improving VANET performance since it deals with intelligent agents or learners that can achieve their goal by interacting with the environment to maximize rewards. In this chapter two DRL algorithms have been used, the Deep Q-network (DQN) learning (Mnih, et al., 2015) and Actor-critic learning (Silver, et al., 2014) are used to propose intelligent DRL based channel access mechanism in VANET.

5.2 Deep Reinforcement Learning (DRL) in VANET

RL is an optimization framework to learn how to control and solve complex problems. RL doesn't depend on analytical model to solve the problem, it is not a data-driven approach where it requires training datasets, labelled or unlabeled, for the purpose of solving or building a system model. RL is a branch of ML where an intelligent agent takes an action, and the environment returns observations or state of the system and rewards. The goal of the agent is to determine the best action to take, usually RL is described in terms of the agent interacting with the previously unknown environment, trying to maximize overall or total reward. Through extensive trainings, interacting with the environment, trial and error approach, the agent learns the best action to take in every state of the environment to maximize numerical reward. While RL has achieved success story in various domains (Riedmiller, Gabel, Hafner, & Lange, 2009) (Diuk, Cohen, & Littman, 2008), its applicability has been limited to domains that fully observed, limited state spaces (Mnih, et al., 2015). In this chapter, we are going to use DRL to overcome such limitations in RL where it utilizes the deep neural networks (Bengio, 2009) (Krizhevsky, Sutskever, & Hinton, 2017) (Bellemare, Naddaf, Veness, & Bowling, 2013), to develop an artificial agent that can learn

successful and efficient policies from high dimensional state spaces environment. Over time, the agent has the ability for developing intuitive human-like behavior. In DRL, the agent's policy that employs to determine the next action based on the current state is deep Neural Network (NN) that requires training. Two DRL algorithms have been used in this research, DQN and Actor-critic algorithms.

5.2.1 Applying DRL in VANETs

In order to apply DRL in VANETs, we are going to implement DQN and Actor-critic algorithms. Both algorithms are using the agent or learner who interact with environment (VANETs environment), selecting action (appropriate CW value) to maximize the reward (VANETs' throughput) and observing the next state after selecting action. In this scenario, the agent does not have any information on which action (selecting CW value) to take, it has to learn which action to select that maximize reward. Moreover, the selected action usually will affect not only the immediate reward but also the future states, and consequently, all subsequent rewards. This setup required a framework for modeling decision- making process in a particular situation when the outcomes are unpredictable. The Markov Decision Process (MDP) will be used to formalize the sequential decision- making process.

5.2.1.1 Markov Decision Process

MDP (Sutton & Barto, 2018) is a mathematical framework to represent decision-making process in the DRL agent. MDP in VANET channel access problem is defined (S, A, R, S′) where S stands for set of possible states, A is the set of possible actions, R is the reward for selecting action, and S′ or S_{t+1}

is the next state that the environment will return because of an action A as shown in Figure 5-4. At each time step t = 1, 2, 3 …, the agent observes the current state $s_t \in S$, choose an action $a_t \in A$, then receives r_t and observes the next state $s_{t+1} \in S$.

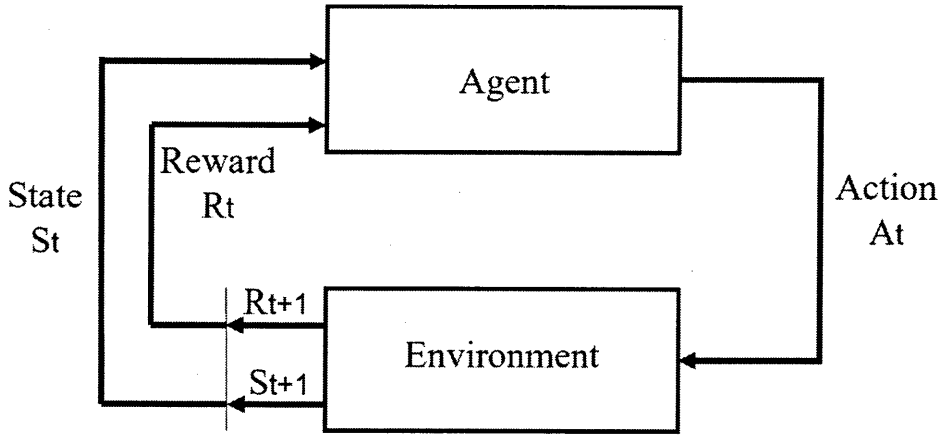


Figure 5-4 Agent-environment interaction in MDP.

5.2.1.2 Partially Observable MDP (POMDP)

In the previous section, MDP was explained as a mathematical representation for VANET channel access problem, However the MDP model is not sufficient for such problem since the agent is not fully aware of all vehicles' status information. The agent can partially observe the VANET information (CHOE, et al., 2020). The abovementioned reasons make the

POMDP (Monahan G. , 1982) (Spaan, 2012) a good choice for representing VANET channel access problem.

The model POMDP such as VANET channel access problem is defined as (S, A, P, R, O, γ) tuple, where:

- S define all possible states.
- A define all possible actions.
- P define the probability of transition from one state to another.
- R define reward function.
- O define possible observations.
- γ define the discount factor.

In the context of POMDP in VANET channel access problem, the following introduces POMDP model's elements for VANET channel access problem to adjust CW value.

- Agent: It is located at the Road Side Unit (RSU) where all the vehicles connected to it. The RSU overseas all VANETs information through beacon frames. It has global view of the network that enable centralized control over all the vehicles. Also, the RSU is equipped with required resources that enable it to handle all DRL computational process.
- State (s) $\in$ S: The agent observe the features of VANETs' environment represented by the status of all vehicles connected into the network.
- Action (a) $\in$ A: The action function that adjusts or sets the CW value. Referring to equation 5.2, The backoff exponent (x) is taking the format 4+a to satisfy CW lower and upper bound [15 – 1023] based the CSMA/CA standard. In this case, the set of all possible action A are defined as A $\in$ [0 - 6]. Each action (a) is taken by the agent to set CW value in (s) state, the agent receives (r) as numerical reward and finds itself in a new state (s′ $\in$ S) with probability P(s′| s, a) which is the

probability of being in new state (s′) given previous state (s) and action (a).

- Reward (r): The reward function is represented by VANETs throughput, the number of received bits per seconds r $\in$ [0 - 1]. The throughput is a key performance indicator about network that can be observed by the RSU. Since reward in DRL should be a real number between 0 and 1, we normalize network throughput by dividing received bits per seconds by the expected maximum throughput.
- Observation (o): It is defined as the current collision rate $P_{collision}$ observed by the agent using equation 5.3.

$$\text{P collision} = \frac{Pt - Pr}{\text{Pt}} \tag{5.3}$$

Where Pt is the number of transmitted packets and Pr is the number of received packets.

The collision rate observations are done in a predefined interaction period to reflect network performance of setting CW value. As we discussed before, the setting of CW value has a direct impact on collision rate in the network. When the CW value is small and multiple vehicles are trying to access the channel and transmit their data, the collision rate will increase since there is a high chance that multiple vehicles will select same backoff timer. In order for the agent to calculate collision rate in the network, since RSU is taking part of all transmission in the network, the agent knows the RSU's Pt and needs to obtain Pt from each vehicle which can piggybacked in the packets. But the agent requires the Pr, which is available through number of sent and received acknowledgment packets. This scenario is considering

uplink traffic initiated from the vehicles toward the RSU as shown in Figure 5-5.

- γ discount factor: Models the importance of future rewards compare to the immediate rewards.

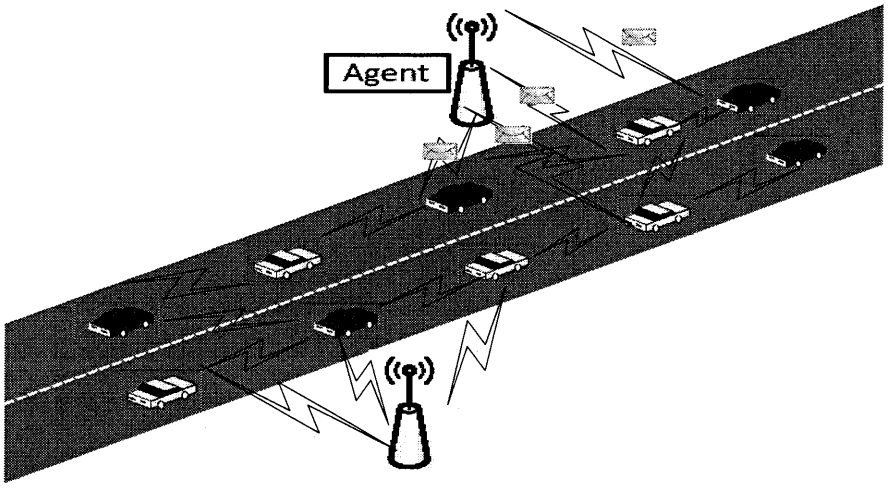


Figure 5-5 VANET Topology

5.2.1.3 Deep Q-Network Learning Algorithm

Due to its name, at the heart of the agent is a deep neural network that act as a function approximator where the states passed as inputs for NN and it produces a vector of action-values with the maximum value indicating which action to take as shown in Figure 5-6.

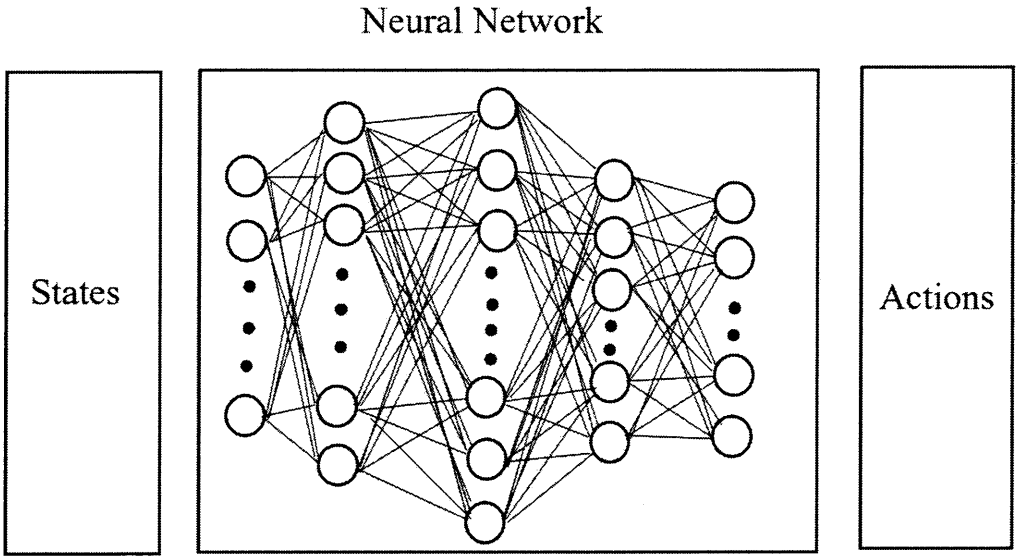


Figure 5-6 Simple DQN diagram

In DRL, the main goal of the agent is formalized in terms of a signal called reward (r), passing from the environment to the agent at each time step t. The agent's goal is to interact with the environment and selecting action that can maximize the total amount of rewards or expected cumulative rewards. This does not mean only the immediate rewards, but total future rewards. So, it is critical that the formulated rewards indicate or has a direct relationship with what we want to accomplish.

As we said earlier that the agent's goal is to maximize expected cumulative rewards, the question now, how to define the cumulative rewards. From looking at some time step (t), it is important to know that the rewards for all previous time steps (r_1, r_2, ...r_t) are already have been decided as they are in the past. Only future rewards (r_{t+1}, r_{t+2}, r_{t+3}, ...r_T where T is the final time step) are under the agent control, so we refer to the sum of rewards from next time (t) onwards as a return at time step (t) and denoted as Gt as shown in equation 5.4.

$$G_t = r_{t+1} + r_{t+2} + r_{t+3} + r_{t+4} + \ldots + r_T \quad (5.4)$$

The agent is trying to select action (a_t) to maximize expected G_t. the agent can predict what the future reward is likely to be, it has to depends on the prediction or an estimate. Since the future reward depends on prediction or estimate, it makes more sense to consider that it is not entirely clear what the future holds, especially the agent is still learning. So, does the current rewards carry the same weight as future reward?

It would be more sense to value the rewards that come sooner more highly since these rewards are more predictable compared to the far future rewards. This situation motivates the idea of discounted return (Sutton & Barto, 2018). RL makes a standard assumption (Mnih, et al., 2015) that future rewards are discounted by a factor of γ per time step and define the future discounted return at time (t) as equation 5.5.

$$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \ldots + \gamma^{T-t-1} r_T = \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k \quad (5.5)$$

Where $\gamma \in [0,1]$, In this concept, we define new function called action-value function for policy π $Q_\pi(s,a)$ equation 5.6 which is the value of taking action (a) in a state (s) under a policy π.

$$Q_\pi(s,a) = E_\pi [G_t \mid S_t = s, A_t = a] \quad (5.6)$$

$Q_\pi(s,a)$ is defined as the expected return of state (s) and action (a), when the agent starts in state (s_t) and then selects action (a_t) following the policy π to selects its action for all future time steps.

This bring us to the optimal action-value function Q*(s,a) as the maximum expected return following optimal policy π^* as shown in equation 5.7,8 and 9.

$$Q^*(s,a) = \max_\pi E\,[G_t \mid S_t = s, A_t = a, \pi] \tag{5.7}$$

$$Q^*(s,a) = \max_\pi E\,[r_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a, \pi] \tag{5.8}$$

$$Q^*(s,a) = \max_\pi E\,[r_{t+1} + \gamma\, Q^*(S_{t+1}) \mid S_t = s, A_t = a, \pi] \tag{5.9}$$

Equation 5.9 is a form of the bellman optimality equation for Q*. The Bellman optimality equation for Q* as shown in equation 5.10.

$$Q^*(s,a) = E\,[r_{t+1} + \gamma \max_{a'} Q^*(S_{t+1},a') \mid S_t = s, A_t = a] \tag{5.10}$$

By (Mnih, et al., 2015), the optimal action- value function Q* follow bellman equation based on the following intuition: if the optimal value Q*(s',a') of the sequences s' at the next time step was known for all actions a', then the optimal policy is to choose a' that maximize expected value of $r + \gamma Q^*(s',a')$ equation 5.11.

$$Q^*(s,a) = E\,[r + \gamma \max_{a'} Q^*(s',a') \,|s,a] \qquad (5.11)$$

Traditional RL estimates the action-value function by using bellman equation by iterative update, so the Q value converge to Q* as number of iterations approach ∞. This approach is impractical (Mnih, et al., 2015) because the Q function is estimated separately for each sequence. Instead, the Q function can be parameterized by NN weights θ, so instead of iterating over the states, and the states might be big, the Q function will be parameterized by lower dimensional set of parameters θ and then optimize over θ to get better Q function as equation 5.12.

$$Q^*(s,a) \approx Q(s,a,\theta) \text{ function approximator with weights } \theta \qquad (5.12)$$

The NN can be trained by adjusting weights θ_i at iteration i to reduce mean square error (MSE) in Bellman equation, where the optimal action-value function Q* is replaced with function approximator $y = r + \gamma \max_{a'} Q(s',a'; \theta_i)$. Now, the MSE is represented by loss function which is the expectation of the square of the difference error as shown in equation 5.13 and the NN is going to use stochastic gradient descent algorithm and backpropagation to optimize the weights θ to give the best possible Q function that minimize loss function.

$$L_i(\theta_i) = E_{s,a,r}\,[(E_{s'}\,[y|s,a] - Q(s,a;\theta_i))^2] \qquad (5.13)$$

Where:

$$y = r + \gamma \max_{a'} Q(s',a';\theta_i). \qquad (5.14)$$

Unlike the traditional RL setup where only one Q-value is produced at a time, the DQN is designed to take states as input to NN and produce a Q-value for every possible action as actions vector as output in a single forward pass. Use this vector to take an action stochastically or by choosing the action with the maximum Q-value. These inputs and outputs transformation support a powerful NN architecture. The input states (the collision rates observation) are processed by Recurrent NN (RNN), to represents the optimal action-value function Q*.

Training NN requires a huge data, and it is not guaranteed to converge into optimal action-value function Q*, and the NN might oscillate or diverge because of the high correlation between taken actions and environments states. This could lead to unstable and inefficient strategy or policy. To overcome these issues, the experience replay technique has been used.

The idea of experience replay is to make more efficient use of the observed experiences. Consider the basic Q- learning algorithm where the agent interacts with the environment, at each time step it obtain a state (s_t), action (a_t), reward (r_{t+1}) and next state (S_{t+1}) tuple, so learn from tuple and delete it and moving to the next tuple in the next step as shown in figure 5-7.

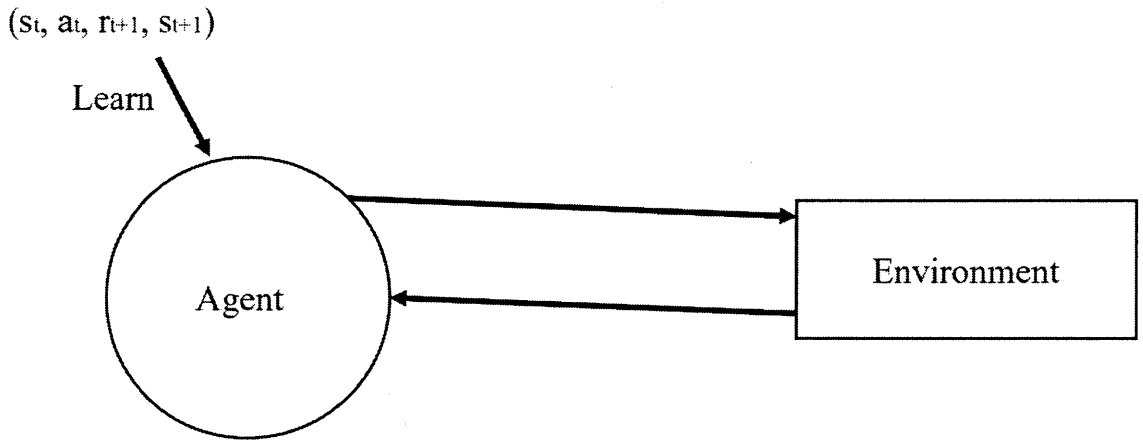


Figure 5-7 Q-learning tuple

The process of deleting these tuples seems wasting for one good resource of learning, the agent can use these tuples to learn more from those experience tuples if we save then in a buffer let's say. Another important benefit of these tuples is that some states are rarely to experience when the agent interact with the environment, and some actions are costly in some situation and difficult to repeat them, so it would be good idea to reuse such experiences. This is the idea behind replay buffer, the agent stores each experience tuple (s_t, a_t, r_{t+1}, s_{t+1}) in the replay buffer as the agent interacting with the environment and then sample a small batch (b) of tuples from the buffer for learning process. As a result, the agent is able to learn from individual tuple multiple times and make a better use of agent's experience as shown in Figure 5-8.

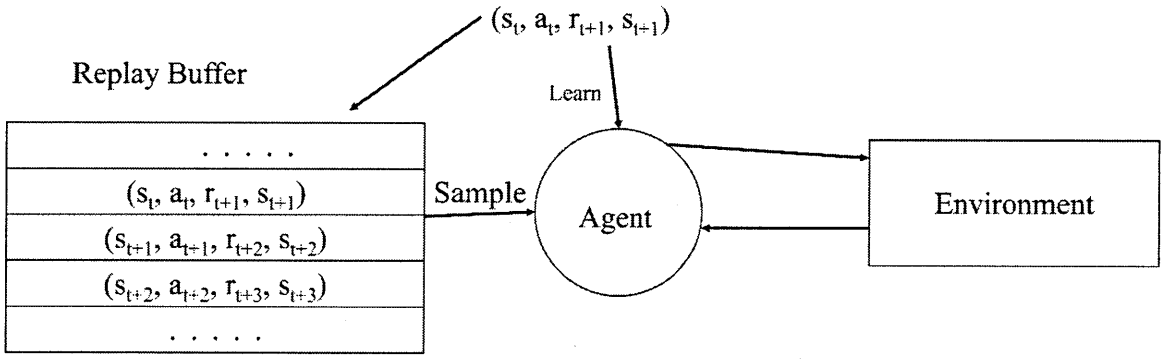


Figure 5-8 Buffer Replay technique

Another advantage of using the replay buffer to overcome correlation between states sequences. Every action (a_t) is affecting the next state (s_{t+1}), which means the sequence of experience tuples can be highly correlated and

diverge the learning process. Using experience replay, the agent can randomly sample from the buffer, to avoid sampling sequential tuples. Random sampling helps break the correlation and prevents action-value from oscillating or diverging.

The NN architecture is using one recurrent Long Short-term memory (LSTM) layer, followed by two dense layers with Rectified Linear Unit (ReLU) activation resulting in a $8 \times 128 \times 64$ and one fully connected output layer that produce the vector of action value as shown in Figure 5-9.

RNN has been used because of its ability to solve problem of the sequence dependency. It is used in time -series prediction to identify correlations among data. RNN mimics human brain where it helps to predict results of sequential data by memorizing some previous values. For example, standard NN takes the inputs and multiply them with random weights, and add bias, transformed with an activation function to get the output. Then, NN used backpropagation process from output layer to the input layer going through each layer to adjust and optimize weights of neurons.

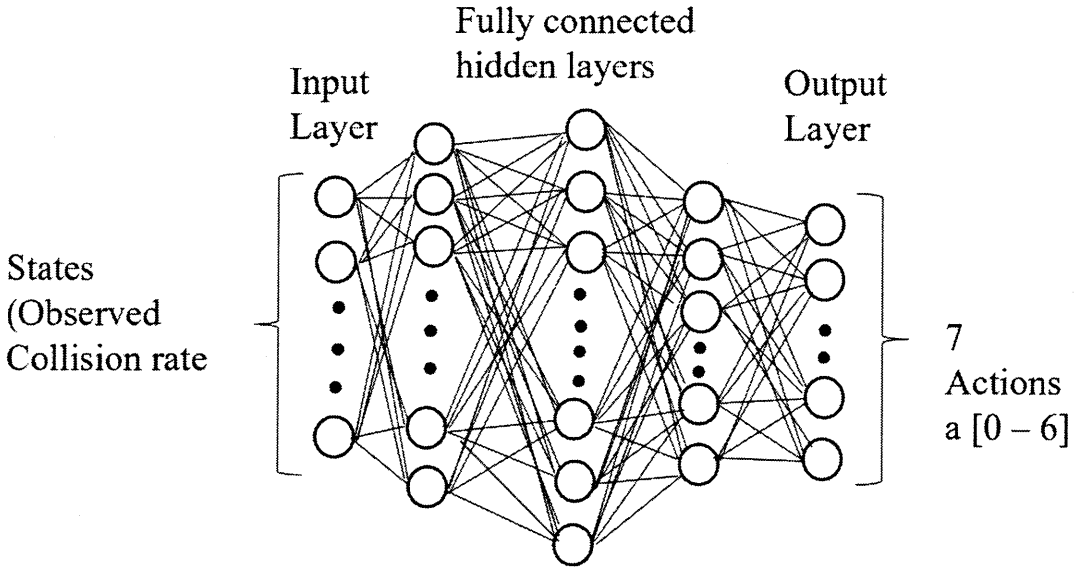


Figure 5-9 NN architecture for DQN

backpropagation process adjusts the neurons' weights by calculating gradient to minimize error between predicted values and the observed values. The optimization process is repeated iteratively until the error is minimized.

In case of sequence dependency. The standard NN will not help since there is no memory to remember the previous values and take them in the consideration. In other word, it processes the data input independently and does not concern the values' dependency or order. Therefore, standard NN is not suitable for to solve a problem with sequence dependences or time-series problem.

In the CW optimization problem, memorizing previous collision rate can affect the next state, for example when you have a dense environment and many vehicles are trying to access the wireless channel, the vehicle that successfully choose appropriate CW value and send the data, and trying again to access the channel, should consider the previous data input to help in taking appropriate decision.

RNN is a type of NN that can solve such problem with sequence dependency, it consists of sequential input layers, hidden layers, time steps and sequential output layers. The output of layers is calculated using inputs of the neurons, weights and most importantly the previous data point. RNN has solved the sequence dependency problem by considering the previous data point but RNN as the standard NN prone to a problem called gradient exploding or vanishing. The problem comes from using gradient descent algorithm in the back propagation process to optimize NN weights. As number

of hidden layers increase, gradient becomes very small and the NN weights will hardly change, as a result it will slow down the learning process.

To solve the vanishing problem faced by RNN, LSTM (Hochreiter & Schmidhuber, 1997) has been proposed which is considered a variant of RNN consisting of multiple cells. All RNNs have the form of a chain of repeating modules of NN with simple structure containing a single tanh layer as shown in Figure 5-10 where x_t is the t^{th} time step input, and M_t represents the memory of the current cell while M_{t-1} represents the memory of the previous cell. LSTM has repeating structure like the RNN but instead of single layer, there are four layers as shown in Figure 5.11.

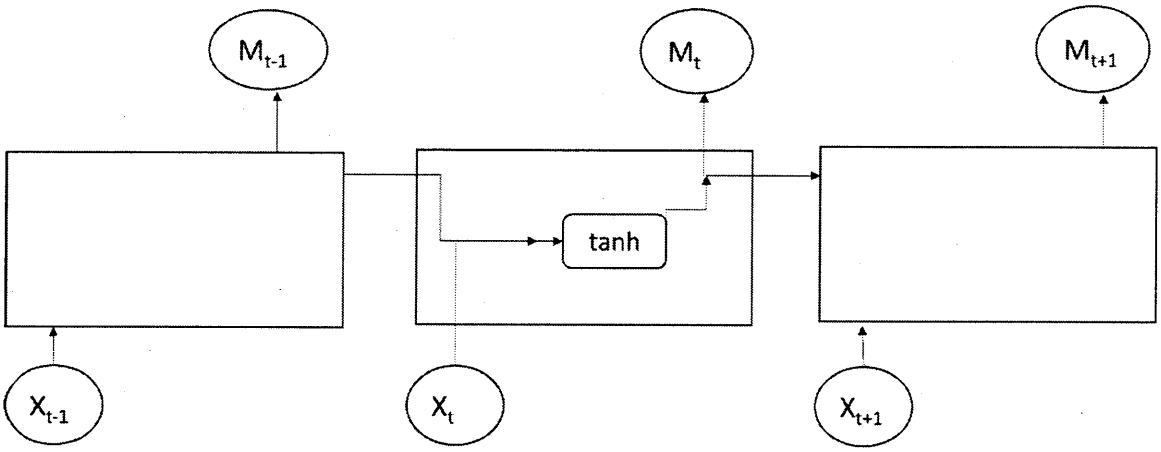


Figure 5-10 the repeating module in RNN with single layer

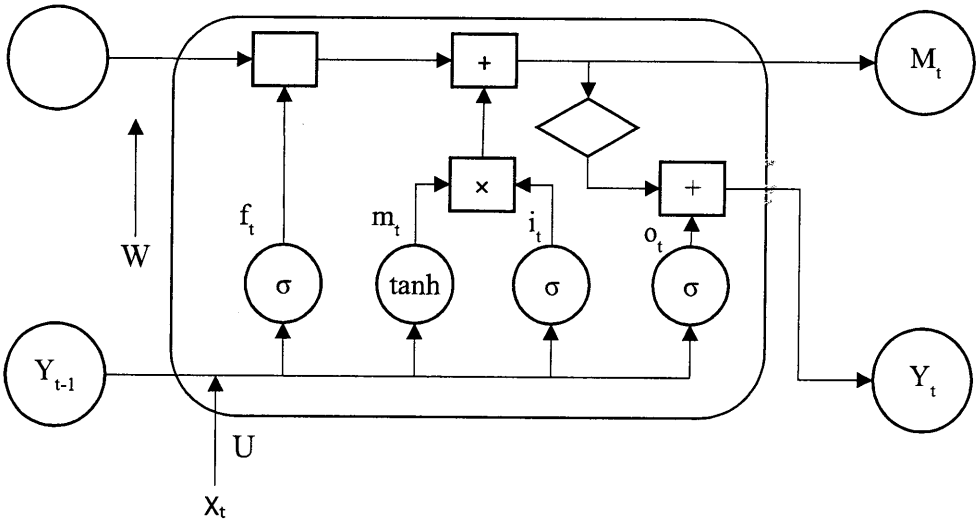


Figure 5-11 The LSTM NN architecture

Each LSTM cell is composed of an input gate (i_t), an output gate (o_t), a forget gate (f_t) and memory (m_t). The additional component in the LSTM neural network is memory. These cells are the elementary units for layers of the neural network. The memory of the LSTM neural network comes from the cells of the hidden units. The cell memorizes the values of the unit for an arbitrary period of time. The forget gate and input gate apply the sigmoid (represented as σ) activation function and the activation function for memory is tanh. M_{t-1} represents the memory from the previous cell and M_t represents the memory of the current cell. Similarly, the Y_{t-1} is the output from the previous cell and Y_t represents the output of the current cell. The symbol '×' illustrates the elementwise multiplication and the symbol '+' represents the elementwise addition. X_t is the t^{th} timestep input to the cell. U and W are the weight vectors for the input gate, output gate and the forget gate. The output of these gates are the vectors that are computed by applying the weights and corresponding activation functions on the input for every timestep. Each cell generates a memory and an output. The memory can be utilized or forgotten

by the next cell depending on the values from the activation function of the forget gate as depicted in Figure 5-11.

5.2.1.4 Actor-Critic Learning Algorithm

While DQN can solve high dimensional spaces problem, but it can solve only discrete action spaces. DQN is not suitable for continuous action spaces. In real- life RL applications the states and actions space are in continuous not discrete form, making the approach of saving value functions or policies for each state or state-action pair is impossible. RL algorithm used function approximators for value function and policy to cover full range of states and actions (Silver, et al., 2014).

Actor-Critic algorithm is at intersection of value-based method like DQN and policy-based method like REINFORCE. Value-based method calculate Q function $Q(s,a) \approx Q^*(s,a)$ to find optimal policy $\pi(q(s,a)) \approx \pi^*$. Policy based is trying to find the optimal policy $\pi \approx \pi^*$ by directly mapping state to probability of taking that action in a given state $\{a \sim \pi(s,a) \approx \mathbb{P}(a|s)\}$ and choose an action based on the probability distribution. For example, the stochastic policy takes a state (s) and return the probability distribution of the actions as shown in figure 5-12.

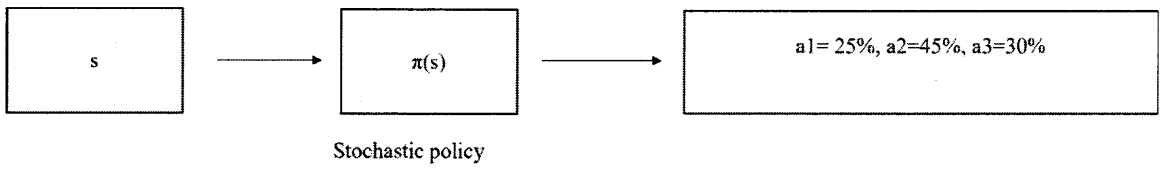


Figure 5-12 Stochastic policy with respect to state

In other notation of stochastic policy in which it takes the state (s) and action (a2) and return the probability of taken that action in state (s) as shown in Figure 5-13.

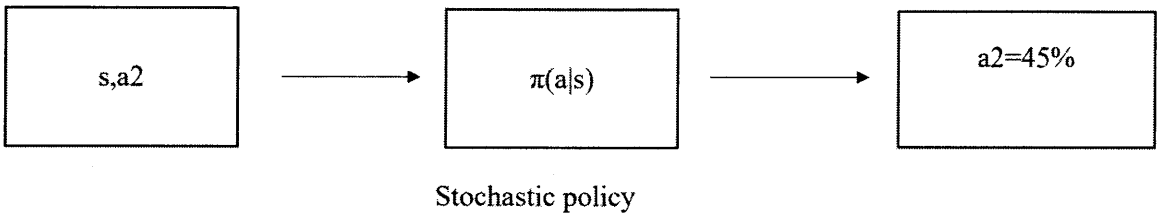


Figure 5-13 Stochastic policy with respect to state (s) and action (a2)

The Actor-Critic agent is using function approximator to learn policy and value function. The idea of Actor-Critic reinforcement learning is to take the best of both methods value-based and policy-based learning. So, in an Actor-Critic algorithm architecture two learners or two NN will be used. The Actor learner is trying to learn a good policy and the Critic is critiquing that policy $\pi(s,a;\theta)$ based on its estimate of the $Q(s,a;\theta)$ function.

In this research the Actor-Critic is using Deep Deterministic Policy Gradient (DDPG) algorithm for reinforcement learning with continuous actions. DDPG used two NN, one for the Actor and one for the Critic as shown in Figure 5-14.

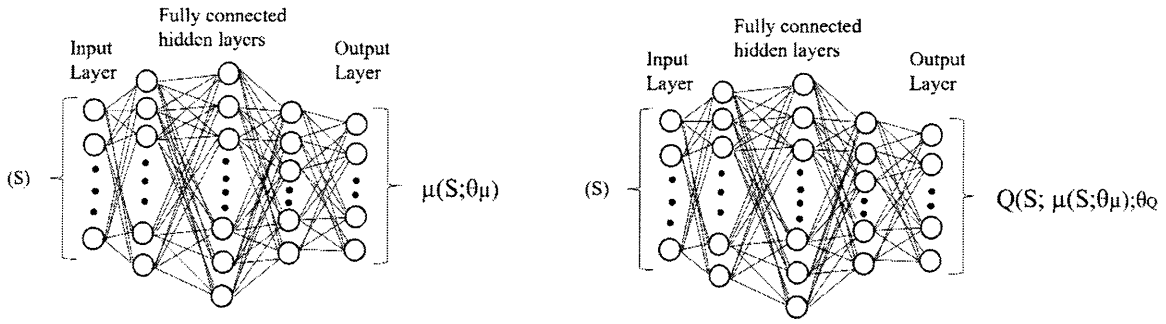


Figure 5-14 Actor-Critic Architecture

In DDPG the policy parameterized by the NN weights θ and we optimize over θ to maximize the future rewards. NN uses backpropagation to compute gradients and optimize weights θ by computing the gradients based on the cumulative expected future rewards using equation 5.15 taken from (Silver, et al., 2014).

$$\theta^{K+1} = \theta^{K} + \alpha \nabla\theta \, R_{\sum,\theta} \qquad (5.15)$$

Where α is the learning rate, $\nabla\theta \, R_{\sum,\theta}$ gradient of cumulative expected rewards, θ^{K} is the old weight θ and θ^{K+1} is the new weights θ.

5.2.1.5 VANET scenarios

Two VANET network scenarios will be discussed during the implementation phase.

Scenario 1: All Vehicles connected at the same time.

In this Scenario, all vehicles joined the network and start sending the data with RSU. This scenario is trying to simulate when many vehicles will be in

the RSU's vicinity joining and start sending data like stopping at the traffic light area as shown in Figure 5-5.

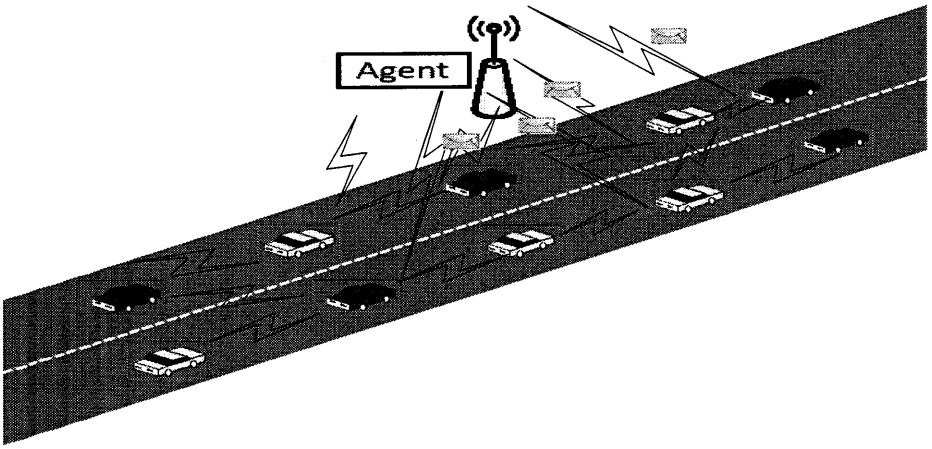


Figure 5-15 VANET topology

Scenario 2: All Vehicles gradually connected.

In the second scenario, the vehicles are gradually increased and connected to the RSU starting from 5 vehicles to maximum number of vehicles. The scenario is designed to evaluate algorithm under different environment traffic density.

5.2.1.6 DRL for adjusting CW

The DRL process consists of three stages, the first stage where the vehicles start accessing the channel and sending their data. This stage is considering a warming-up stage.

The second stage is the training stage where the agent (RSU) is taking a decision regarding CW values for vehicles following procedures in Algorithm 1.

The preprocessing for collision rate (P collision) means to calculate mean (μ) and standard deviation (σ) of collision rate for length h using sliding time window of a fixed size and stride (step size) that will be used in Recurrent neural network (RNN)and interpreted as a time series. This process will give more in-depth understanding of relationship between agent action and network congestion.

To mitigate exploration-exploitation issue, DRL utilize the ε-greedy (Sutton & Barto, 2018) algorithm to balance between exploitation where the agent needs to choose the effective action who look working good at present and exploration where the agent needs to explore new action that may work well and maximize the rewards in the long run.

After training stage, the agent will choose the best possible action and it will be considered as fully trained. The agent is ready to be implemented in the network.

Algorithm 1: DRL CW optimization

---

1 Initialize observation buffer B

2 Initialize D as replay buffer (Record all interactions between agent and environment to be used for mini batch sampling)

3 Initialize θ as agent weights for NN

4 Initialize s as states

5 Initialize action function

6 Define Pt, Pr as number of transmit and received packets

7 Define Δt as interaction time to calculate throughput

8 Initialize CW to 15

9 **for** step t = 0,1,... **do**

10 **If** last_update +Δt ≤ current_update

11 Preprocess (B) calculate mean (μ) and standard deviation (σ) of collision rate of length h

12 Action (a) from A_θ Pass through NN

13 $CW = 2^{4+a} - 1$

14 **If** train **then**

15 Throughput = Pr/Δt

16 r = normalize (throughput)

17 D.push (s, a, r, s′)

18 b sample minibatch from D

19 Optimization on θ based b

20 **end if**

21 **end if**

22 **end for**

In DRL process, there are some parameters that need to be configured. Starting with γ discount factor ($0 \leq \gamma \leq 1$). According to the discount factor, the agent tries to select actions so the sum of discounted rewards it receives over the future is maximized. It determines how much the agent is care about immediate rewards compare to future rewards. If $\gamma = 0$, the agent will be care about actions that give immediate rewards. If $\gamma = 1$, the agent will evaluate the actions based on sum total of all future rewards. Second parameter is the learning rate α ($0 < \alpha < 1$), it controls how much to change the model to the estimated error or to what extent the new information will overwrite the old information. The third parameter is the replay buffer which records all the interaction between the agent and the environment.

5.3 Performance Evaluation

5.3.1 Simulation Setup

DRL has been implemented in ns3-gym (Gawłowicz & Zubow, 2019) , it is a framework that integrates reinforcement learning toolkit the OpenAI Gym and ns-3 simulator to implement DRL VANET scenario. Pytorch and Tensorflow have been used to implement NN in the agent. The important simulation's parameters are listed in Table 5-1.

Table 5-1: VANET PHY and MAC parameters

Parameters	Value
Simulation time	60s
Channel Frequency	5.89GHz
Channel Bandwidth	10Mhz
Transmission rate R	250
Data rate	6Mbps
Packet size	1500B
CW max	1023
CW min	15
Propagation loss model	LogDistance Propagation
Mobility model	Constant Position Mobility

Table 5-2: Road traffic parameters

Parameters	Value
Number of Vehicles	10, 20, 30, 40,50
Velocity	Up to 20 m/s
Road dimensions	1000m *200m

Table 5-2 shows the traffic parameters for the simulation. We consider different number of vehicles to simulate traffic congestion.

Table 5-3: DRL parameters

Parameters	Value
History length h	300
Interaction period Δt	10ms
Learning rate	0.0004
Batch size	32
discount factor γ	0.7
Replay buffer D	18000
Timing window size	150
Stride	75

Table 5-3 shows the DRL hyperparameters which have been selected through lengthy simulation to provide a better performance for both algorithms.

The RNN long short-term memory layer (LSTM) design architecture was used for both algorithms, one RNN LSTM followed by 2 dense layers in 8×128×64. For each experiment, a total of 15 episodes of 60-second simulations were conducted. The first 14 episodes were dedicated to the learning phase, during which the DRL algorithms were trained and allowed to learn from the environment. The final episode represented the operational phase, where the trained algorithms were tested and evaluated based on their performance in the VANET network.

5.3.2 Simulation Results

The simulation results present the comparison among standard channel access method denoted by VANET, Channel access using DQN algorithm denoted by DQN, and Channel access using Actor-Critic algorithm denoted by Actor-Critic. Two VANET network deployment scenarios have been implemented, the first deployment scenario when all vehicles join the network at once and start sending their data. The second one, Vehicles are gradually joined the network starting from 5 vehicles and increased one by one until reach maximum number of vehicles. The idea behind the scenario to evaluate our algorithms in different traffic density trying to simulate real life scenario. where in some cases. Also, the second scenario simulates the situation where vehicles are joining RSU gradually.

5.3.2.1 Scenario 1: All Vehicles connected at the same time

The results indicate that standard VANET performance deteriorates in dense environments where multiple vehicles contend for channel access. However, in the case of DQN and Actor-Critic network algorithms, the throughput performance is significantly higher compared to standard VANET, as shown in Figure 5-16.

The improvement achieved by DRL algorithms over standard VANET ranges from 4% for 10 vehicles to approximately 28% for 50 vehicles. These results are expected because DRL algorithms dynamically adjust the CW value based on observed collision rates, whereas VANET relies on traditional exponential backoff channel access. The utilization of DRL algorithms allows for more efficient channel access and reduces collisions, leading to enhanced throughput performance in dense VANET environments. The observed

performance gains highlight the effectiveness of DRL algorithms in optimizing VANET systems compared to traditional approaches.

Figure 5-17 depicts the Instantaneous network throughput in scenario 1 with 50 vehicles. The results demonstrate that DRL algorithms exhibit improved throughput compared to standard VANET practices. Figure 5-18, the CW values selected by DRL algorithms are displayed over a duration of 15 episodes for 30 nodes. It is notable that the Actor-Critic algorithm converges to a stable CW value after approximately 9 or 10 episodes. This stability indicates that the algorithm has learned and determined an optimal CW value for efficient channel access in the given scenario.

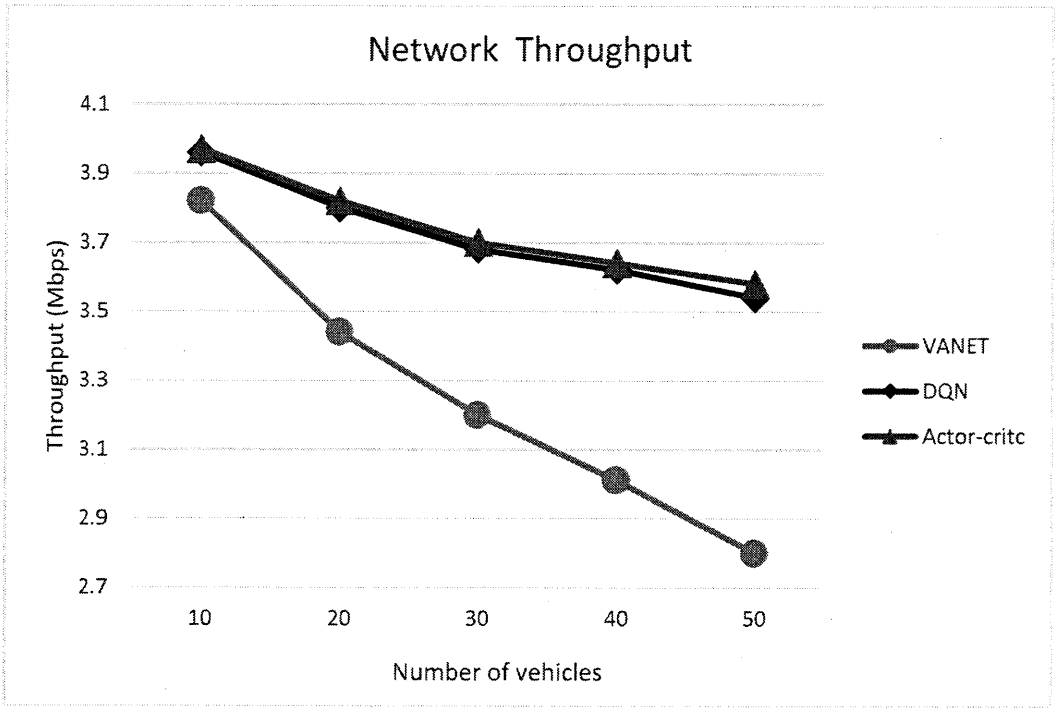


Figure 5-16 Network Throughput in scenario 1

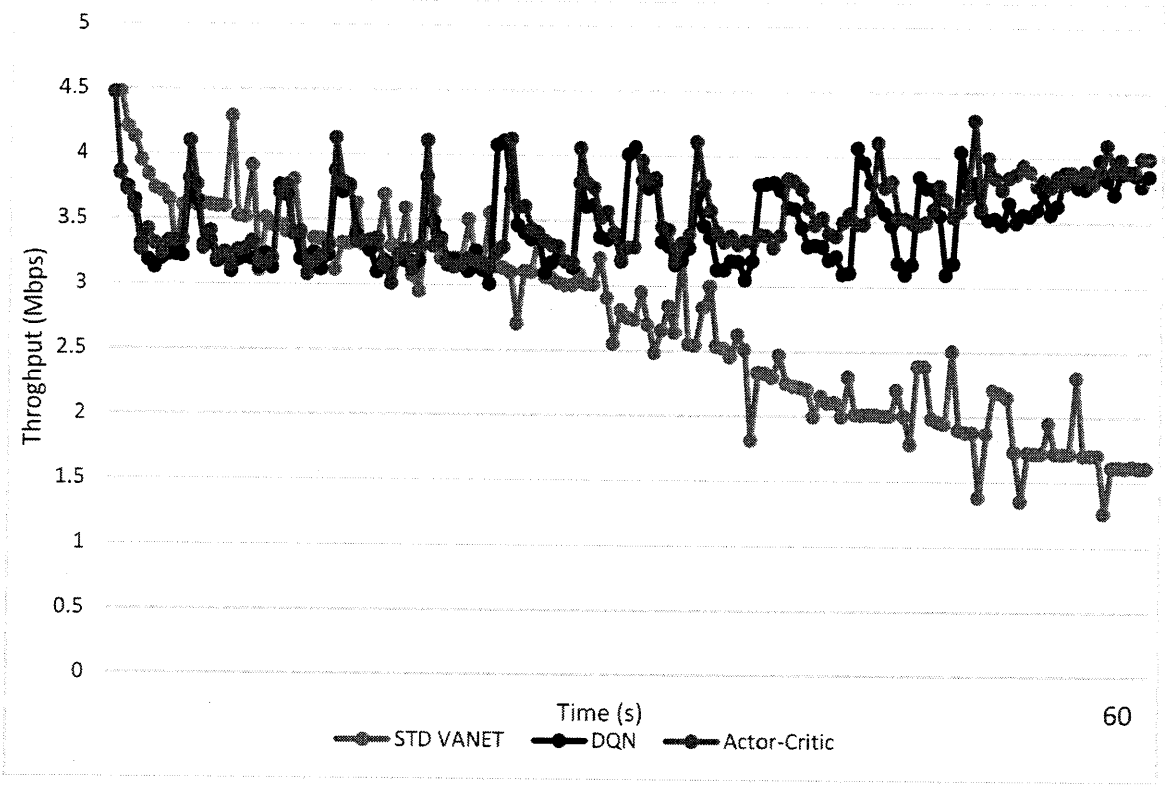


Figure 5-17 Instantaneous network throughput in scenario 1 for 50 vehicles

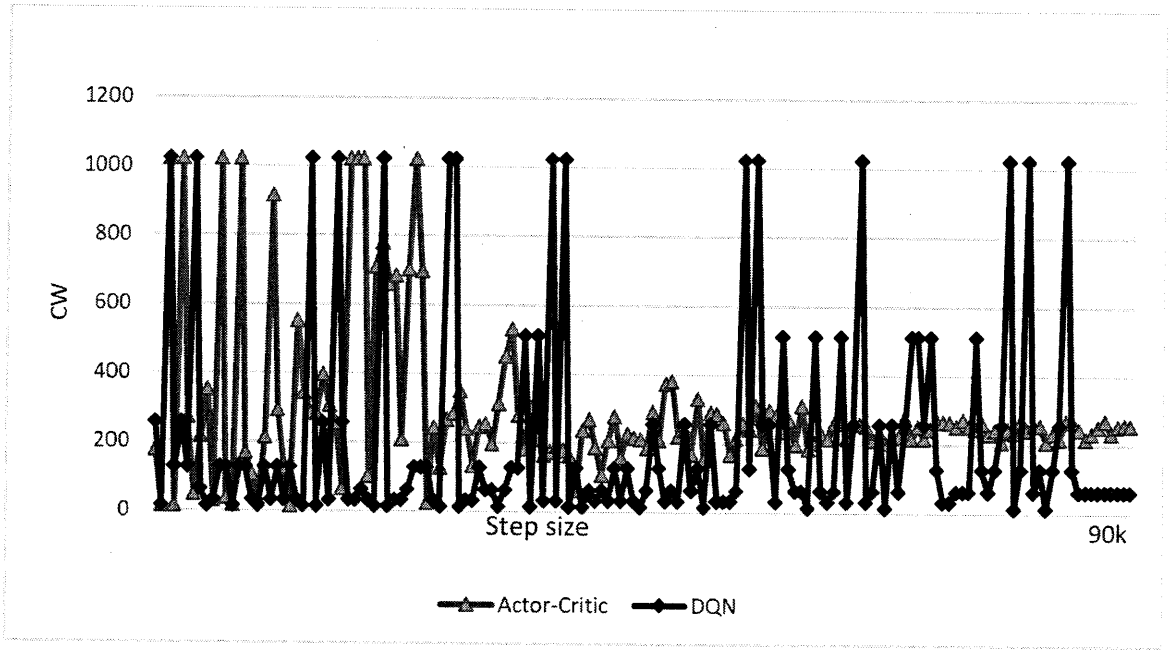


Figure 5-18 Chosen CW in scenario 1 for 30 vehicles

5.3.2.2 Scenario 2: All Vehicles gradually connected

Figure 5-19 shows the network throughput under this scenario. In DQN and Actor-Critic network, the performance in terms of throughput is higher compared to standard VANET.

The improvement between standard VANET and DRL algorithms range from 0.01% for 10 vehicles to around 17% for 50 vehicles. Actually, this scenario shows improvement in the VANET because vehicles are joining one by one which make the standard accommodate the traffic, but still the DRL algorithms outperform the standard.

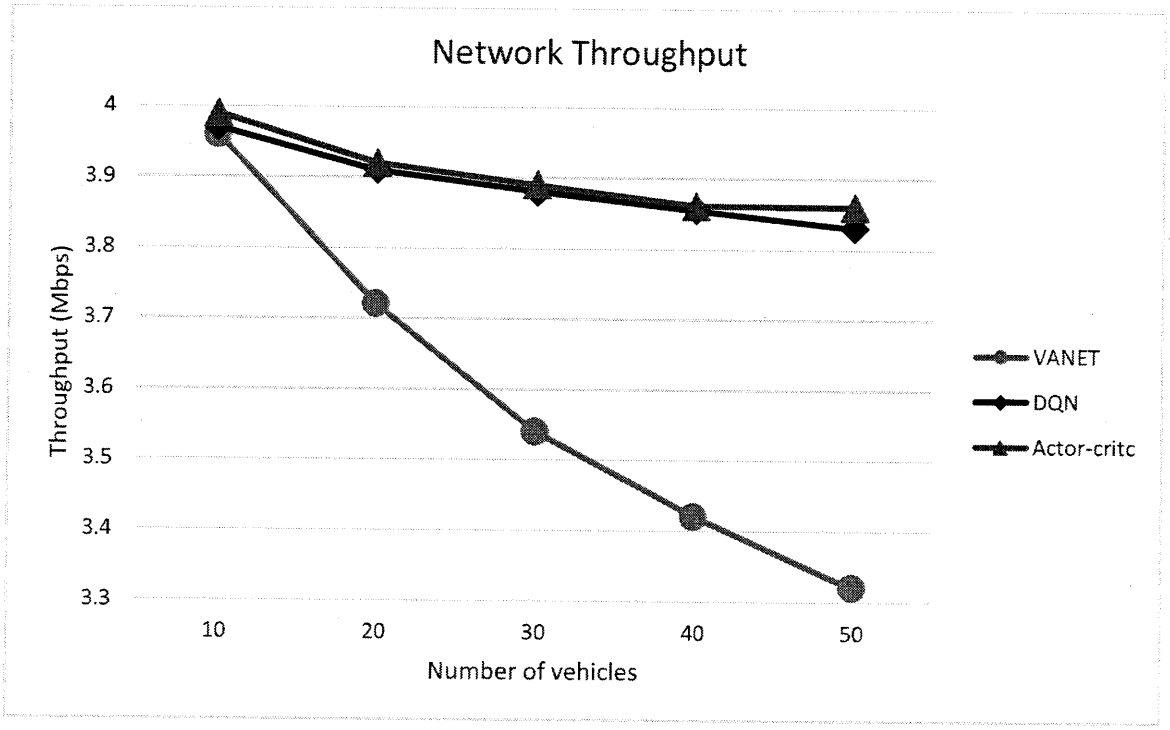


Figure 5-19 Network Throughput in scenario 2

Figure 5-20 shows the instantaneous throughput under scenario 2 for 50 nodes. The standard seems to start the simulation in good performance with light traffic density but when the number of vehicles increased, it decreases

the throughput compared to DRL algorithms. This degradation in throughput for standard because of high competition among vehicles to access the channel and not adjusting the CW accordingly.

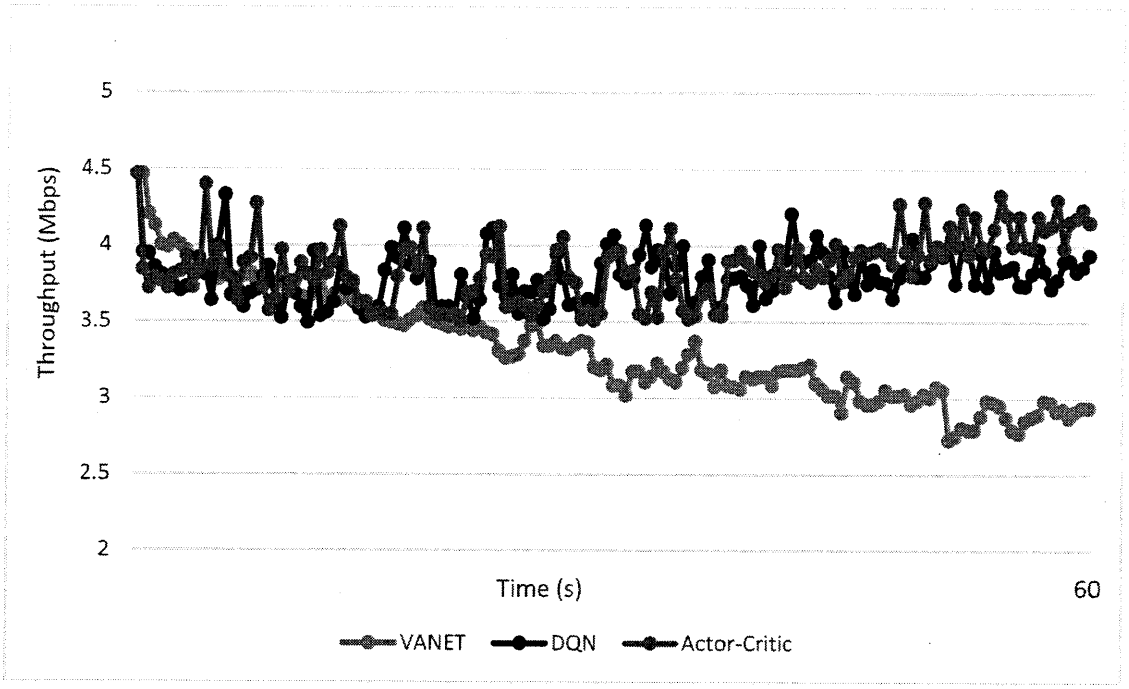


Figure 5-20 Instantaneous Network Throughput in scenario 2

Figure 5-21 is showing the CW value that have been selected by DRL algorithms during 15 episodes duration for 50 nodes. The Actor-critic algorithm is converged to stable CW value better than DQN.

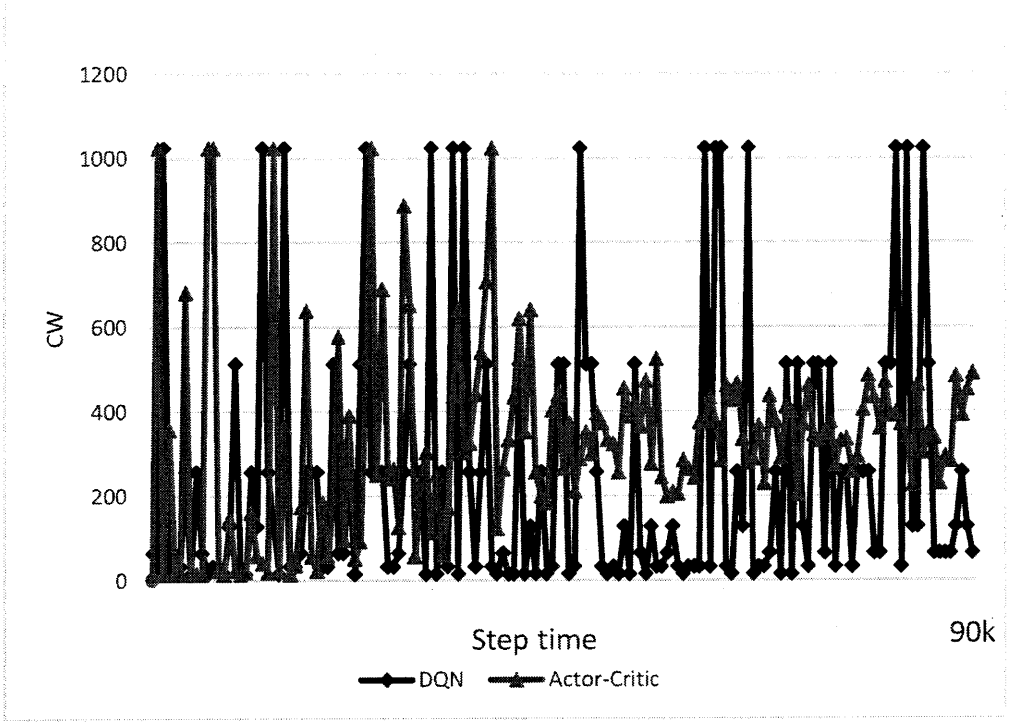


Figure 5-21 Chosen CW under scenario 2 for 30 vehicles

5.4 Conclusion

In this chapter, two DRL algorithms, namely DQN and Actor-Critic, have been introduced to adjust the CW value in various VANET network scenarios. The simulations conducted in the thesis have demonstrated the applicability of DRL in VANET networks for optimizing the CW value.

When compared to the traditional VANET scheme based on CSMA/CA, the proposed DRL algorithms offer advantages in terms of minimizing packet collisions and drops in congested VANET communications. This is achieved by dynamically adjusting the CW value during VANET communication. The simulation results have provided evidence that both DRL algorithms outperform the traditional scheme in terms of performance.

Overall, the chapter highlights the potential of DRL algorithms to enhance VANET networks by optimizing the CW value, leading to improved communication efficiency and reduced packet loss in congested scenarios.

Chapter 6

Conclusion

Chapter 6 Conclusion

6.1 Contribution and Conclusion

In modern times, VANETs are a crucial element of ITS and smart cities that have the capacity to facilitate a diverse set of safety and non-safety applications by enabling vehicles to communicate with other vehicles directly or through RSUs. Through these communications, vehicles exchange messages containing real time information about vehicles. This information can aid drivers in understanding the state of the network and predicting the movements of vehicles. This, in turn, enables them to take preemptive measures regarding any dangerous situations and avoid them. Most of VANETs applications are utilizing traditional broadcasting protocol as a routing protocol to disseminate information generated by vehicles.

Our research commenced by examining the conventional broadcasting protocol as a routing mechanism for disseminating messages among vehicles. We also analyzed critical aspects of forwarding routing mechanisms based on node assessment, including transmitter and receiver-based protocols to select best relay vehicle that can disseminate broadcast message with minimal number of retransmissions in multi-hop VANET networks. Additionally, we investigated the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) algorithm, along with Binary Exponential Backoff (BEB) utilized by vehicles to access the communication channel and transmit messages. Furthermore, we analyzed the impact of the traditional mechanism for selecting CW (Contention Window) value on network performance.

In chapter 3, we have proposed a fuzzy logic inference assisted multi-hop broadcasting relay selection protocol to select next hop relay vehicle. This is done to ensure that all vehicles potentially impacted by a situation receive

the broadcast message with the minimum number of transmission necessary. To select the optimal relay, three important characteristics of the vehicles have been considered: distance, mobility, and RSSI. Our proposal has demonstrated excellent performance in terms of reachability, bytes sent, and rebroadcast per covered vehicle, as illustrated in the results presented in Chapter 3.

In Chapter 4, we explored one of the major challenges in enhancing VANET performance: the efficient allocation of resources at the MAC layer in VANET networks. VANETs employ the CSMA/CA algorithm, which utilizes the BEB method for channel access. However, this approach blindly increases or decreases CW values after a collision or successful transmission. To tackle performance degradation in VANETs, particularly in dense environments, we proposed a contention-based protocol for Vehicle-to-Vehicle (V2V) communication utilizing reinforcement learning. Our proposal involves utilizing AI technique to enhance VANET performance through new broadcasting protocol and channel access scheme. Specifically, we have incorporated the Q-learning algorithm to adjust the BEB CW values in dense V2V broadcasting environments. The Q-learning algorithm is based on a smart agent that can learn interactively by taking the most beneficial action in each state to maximize a numerical reward. Our simulations demonstrate that Q-learning can effectively improve V2V communication for broadcasting BSM traffic by minimizing collisions and dropped packets in congested V2V communications through the adjustment of CW values. Compared to traditional broadcasting schemes based on CSMA/CA, Q-learning outperforms by achieving relatively reliable PDR and high throughput, particularly under high traffic conditions, as illustrated in the results presented in Chapter 4. In addition to incorporating the Q-learning algorithm, our proposal includes the integration of the RL OpenAI gym framework and NS3

to contribute to the field of ITS. This integration serves as a practical example of how RL methods can be applied to VANET problems. Our key contributions include integrating NS3 VANET scenarios with OpenAI Gym, which allows for a straightforward means of formulating and testing RL solutions in the ITS field. We have also implemented RL through the training of an agent to address VANET problems.

In chapter 5, we have continued to address deficiency of selecting CW values of CSMA/CA based BEB algorithm in unicast V2I communications. We propose two advanced DRL algorithms, namely the DQN and actor-critic algorithms, in dense network environments. These algorithms are implemented and compared to conventional standards. We designed and implemented simulation scenarios to demonstrate the efficiency of our proposed protocols. The results show that our proposed protocols can scale well and adapt to different network scenarios, achieving high network performance compared to the traditional protocol.

6.2 Recommendation and Future Work

Our work suggests several recommendations for future research. Firstly, it would be beneficial to formulate a mathematical model for the proposed solutions, which could aid in the analysis and optimization of the protocols. Additionally, future work should consider different parameters that can affect VANET performance as factors in designing RL and DRL algorithms. Furthermore, it is important to evaluate the proposed algorithms in a more realistic VANET environment to ensure their effectiveness in practical scenarios. These future research directions can contribute to the ongoing development and improvement of VANETs.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., . . . Zheng, X. (2015). TensorFlow:Large-Scale Machine Learning on Heterogeneous Distributed Systems. https://www.tensorflow.org/about/bib.

Alpaydı n, E. (2014). Introduction to Machine Learning, Third Edition. Cambridge: The MIT press.

Anwar, W., Kulkarni, K., Augustin, T., Franchi, N., & Fettweis, G. (2018). PHY Abstraction Techniques for IEEE 802.11p and LTE-V2V: Applications and Analysis. 2018 IEEE Globecom Workshops (GC Wkshps). Abu Dhabi, United Arab Emirates: IEEE.

Arena, F., Pau, G., & Severino, A. (2020). A Review on IEEE 802.11p for Intelligent Transportation Systems. Journal of Sensor and Actuator Networks.

Bellemare, M., Naddaf, Y., Veness, J., & Bowling, M. (2013). The arcade learning environment: an evaluation platform for general agents. Journal of Artificial Intelligence Research, 253–279.

Bengio, Y. (2009). Learning Deep Architectures for AI. Foundations and Trends® in Machine Learning, 1–127.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Wojciech, Z. (2016). OpenAI Gym. arXiv.

Cailean, A.-M., Cagneau, B., Chassagne, L., Popa, V., & Dimian, M. (2015). A survey on the usage of DSRC and VLC in communication-based vehicle safety applications. 2014 IEEE 21st Symposium on Communications and Vehicular Technology in the Benelux (SCVT). Delft, Netherlands: IEEE.

Campolo, C., Vinel, A., Molinaro, A., & Koucheryavy, Y. (2011). Modeling Broadcasting in IEEE 802.11p/WAVE Vehicular Networks. IEEE Communications Letters, 199 - 201.

Chetoui, Y., & Bouabdallah, N. (2007). Adjustment mechanism for the IEEE 802.11 contention window: An efficient bandwidth sharing scheme. ELSEVIER Computer Communications, 2686-2695.

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., . . . Zheng, X. (2015). TensorFlow:Large-Scale Machine Learning on Heterogeneous Distributed Systems. https://www.tensorflow.org/about/bib.

Alpaydın, E. (2014). Introduction to Machine Learning, Third Edition. Cambridge: The MIT press.

Anwar, W., Kulkarni, K., Augustin, T., Franchi, N., & Fettweis, G. (2018). PHY Abstraction Techniques for IEEE 802.11p and LTE-V2V: Applications and Analysis. 2018 IEEE Globecom Workshops (GC Wkshps). Abu Dhabi, United Arab Emirates: IEEE.

Arena, F., Pau, G., & Severino, A. (2020). A Review on IEEE 802.11p for Intelligent Transportation Systems. Journal of Sensor and Actuator Networks.

Bellemare, M., Naddaf, Y., Veness, J., & Bowling, M. (2013). The arcade learning environment: an evaluation platform for general agents. Journal of Artificial Intelligence Research, 253–279.

Bengio, Y. (2009). Learning Deep Architectures for AI. Foundations and Trends® in Machine Learning, 1–127.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Wojciech, Z. (2016). OpenAI Gym. arXiv.

Cailean, A.-M., Cagneau, B., Chassagne, L., Popa, V., & Dimian, M. (2015). A survey on the usage of DSRC and VLC in communication-based vehicle safety applications. 2014 IEEE 21st Symposium on Communications and Vehicular Technology in the Benelux (SCVT). Delft, Netherlands: IEEE.

Campolo, C., Vinel, A., Molinaro, A., & Koucheryavy, Y. (2011). Modeling Broadcasting in IEEE 802.11p/WAVE Vehicular Networks. IEEE Communications Letters, 199 - 201.

Chetoui, Y., & Bouabdallah, N. (2007). Adjustment mechanism for the IEEE 802.11 contention window: An efficient bandwidth sharing scheme. ELSEVIER Computer Communications, 2686-2695.

CHOE, C., AHN, J., CHOI, J., PARK, D., KIM, M., & AHN, S. (2020). A Robust Channel Access Using Cooperative Reinforcement Learning for Congested Vehicular Networks. IEEE Access.

Choe, C., Choi, J., Ahn, J., Park, D., & Ahn, S. (2020). Multiple Channel Access using Deep Reinforcement Learning for Congested Vehicular Networks. 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring). Antwerp, Belgium: IEEE.

Chollet, F. &. (2015). Keras: Deep Learning for humans. Retrieved from Github: https://github.com/fchollet/keras

Chuang, M.-C., & Chen, M. C. (2013). DEEP: Density-Aware Emergency Message Extension Protocol for VANETs. IEEE Transactions on Wireless Communications, 4983 - 4993.

Complete List - Atari. (2022). Retrieved from https://www.gymlibrary.dev/environments/atari/complete_list/

Dang, D. N., Hong, C. S., Lee, S., & Huh, E.-N. (2014). An Efficient and Reliable MAC in VANETs. IEEE Communications Letters , 616 - 619.

Diuk, C., Cohen, A., & Littman, M. (2008). An object-oriented representation for efficient reinforcement learning. ICML '08: Proceedings of the 25th international conference on Machine learning (pp. 240–247). Helsinki, Finland: Association for Computing Machinery.

Du, J., Wang, S., & Zhang, B. (2020). Vehicle density and signal to noise ratio based broadcast backoff algorithm for VANETs. Elsevier.

Eichler, S. (2007). Performance Evaluation of the IEEE 802.11p WAVE Communication Standard. 2007 IEEE 66th Vehicular Technology Conference. Baltimore, MD, USA: IEEE.

ETSI. (2009, 11). Intelligent Transport Systems (ITS);European profile standard for the physical and medium access. France: European Telecommunications Standards Institute.

ETSI. (2019). Intelligent Transport Systems (ITS); ITS-G5 Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band. Sophia Antipolis Cedex - FRANCE: ETSI.

Gawłowicz, P., & Zubow, A. (2018). ns3-gym: Extending OpenAI Gym for Networking Research. Networking and Internet Architecture (cs.NI).

Gawłowicz, P., & Zubow, A. (2019). ns-3 meets OpenAI Gym: The Playground for Machine Learning in Networking Research. MSWIM

'19: Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (pp. 113- 120). New York: Association for Computing Machinery.

Hafeez, K. A., Zhao, L., Ma, B., & Mark, J. W. (2013). Performance Analysis and Enhancement of the DSRC for VANET's Safety Applications. IEEE Transactions on Vehicular Technology, 3069-3083.

Harigovindan, V. P., Babu, A., & Jacob, L. (2012). Ensuring fair access in IEEE 802.11p-based vehicle-to-infrastructure networks. EURASIP Journal on Wireless Communications and Networking.

Hassan, M. I., Vu, H. L., & Sakurai, T. (2011). Performance Analysis of the IEEE 802.11 MAC Protocol for DSRC Safety Application. IEEE Transactions on Vehicular Technology, 3882 - 3896.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 1735–1780.

IEEE 802.11 Working Group. (2005). 802.11e-2005 - IEEE Standard for Information technology--Local and metropolitan area networks--Specific requirements--Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Amendment 8: Medium Access Control (MAC) Qual. Retrieved from ieeexplore: https://ieeexplore.ieee.org/document/1541572

IEEE 802.11p. (2010, July 15). 802.11p-2010 - IEEE Standard for Information technology-- Local and metropolitan area networks--Specific requirements-- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular En.

IEEE Standard for Information technology-- Local and metropolitan area networks-- Specific requirements-- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. (2010, July 15). IEEE.

IEEE802.11 WG - Wireless LAN Working Group. (2007). IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks -

Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specificatio. Retrieved from Standards.ieee.org: https://standards.ieee.org/ieee/802.11/3605/

Jang, H.-C., & Feng, W.-C. (2010). Network Status Detection-Based Dynamic Adaptation of Contention Window in IEEE 802.11p. 2010 IEEE 71st Vehicular Technology Conference. Taipei, Taiwan: IEEE.

Jiang, H., Guo, H., & Chen, L. (2008). Reliable and Efficient Alarm Message Routing in VANET. 2008 The 28th International Conference on Distributed Computing Systems Workshops. Beijing, China: IEEE.

Karabulut, M., Shah, A., & Ilhan, H. (2018). The Performance of the IEEE 802.11 DCF for Different Contention Window in VANETs. 2018 41st International Conference on Telecommunications and Signal Processing (TSP). Athens, Greece: IEEE.

Khan, A., Bilal, S., & Othman, M. (2012). A performance comparison of open source network simulators for wireless networks. 2012 IEEE International Conference on Control System, Computing and Engineering. Penang, Malaysia: IEEE.

Kim, J. H., & Lee, J. K. (1999). Performance of Carrier Sense Multiple Access with Collision Avoidance Protocols in Wireless LANs. Wireless Personal Communications, 161–183.

Kloiber, B., Härri, J., Strang, T., & Sand, S. (2014). Bigger is better — Combining contention window adaptation with geo-based backoff generation in DSRC networks. 2014 International Conference on Connected Vehicles and Expo (ICCVE). Vienna, Austria: IEEE.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 84–90.

Li, T., Tang, T., & Chang, C. (2009). A New Backoff Algorithm for IEEE 802.11 Distributed Coordination Function. 2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery. Tianjin, China: IEEE.

Li, W., Song, W., Lu, Q., & Yue, C. (2020). Reliable congestion control mechanism for safety applications in urban VANETs. Ad Hoc Networks.

Limouchi, E., Mahgoub, I., & Alwakeel, A. (2016). Fuzzy Logic-Based Broadcast in Vehicular Ad Hoc Networks. 2016 IEEE 84th Vehicular Technology Conference (VTC-Fall). Montreal, QC, Canada: IEEE.

López-Martín, C., Chavoya, A., & Meda-Campaña, M. (2013). Use of a Feedforward Neural Network for Predicting the Development Duration of Software Projects. 2013 12th International Conference on Machine Learning and Applications. Miami, FL, USA: IEEE.

Mehta, S., Sulatan, N., & Kwak, K. (2010). Network and System Simulation Tools for Next Generation Networks: a Case Study. In A. Mohamed, Modelling, Simulation and Identification. IntechOpen.

Miao, L., Djouani, K., Van Wyk, B. J., & Yskandar, H. (2013). Performance evaluation of IEEE 802.11p MAC protocol in VANETs safety applications. 2013 IEEE Wireless Communications and Networking Conference (WCNC). Shanghai, China: IEEE.

Miao, Z., Cai, X., Luo, Q., & Dong, W. (2016). A FLRBF scheme for optimization of forwarding broadcast packets in vehicular ad hoc networks. 2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC). Valencia, Spain: IEEE.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A., Veness, J., Bellemare, M., . . . Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature volume, 529–533.

Moloisane, N. R., Malekian, R., & Bogatinoska, D. C. (2017). Wireless machine-to-machine communication for intelligent transportation systems: Internet of vehicles and vehicle to grid. 2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija, Croatia: IEEE.

Monahan, G. (1982). State of the Art-A Survey of Partially Observable Markov Decision Processes: Theory, Models, and Algorithms. Management Science, 1–16.

Monahan, G. E. (1982). A Survey of Partially Observable Markov Decision Processes: Theory, Models, and Algorithms. Management Science, 1-16.

Negnevitsky, M. (2005). Artificial Intelligence: A Guide to Intelligence Systems. Nww York, NY,USA: Addison Weasly.

Nguyen, V., Chuan, P., Thant Zin, O., Nguyen H, T., Eui-Nam, H., & Choong Seon, H. (2018). MAC protocols with dynamic interval schemes for VANETs. Vehicular Communications, 40-62.

Nguyen, V., Kim, O. T., Pham, C., Oo, T. Z., Tran, N. H., & Hong, C. S. (2018). A Survey on Adaptive Multi-Channel MAC Protocols in VANETs Using Markov Models. IEEE Access, 16493 - 16514.

Nguyen, V., Kim, O. T., Pham, C., Oo, T., Tran, N., Hong, C. S., & Huh, E.-N. (2018). A Survey on Adaptive Multi-Channel MAC Protocols in VANETs Using Markov Models. IEEE Access, 16493 - 16514.

Olariu, S., & Weigle, M. (2009). Vehicular Networks: From Theory to practice . New York: Chapman and Hall/CRC.

Otterlo, M., & Wiering , M. (2012). Reinforcement Learning and Markov Decision Processes. In M. v. Wiering, Adaptation, Learning, and Optimization (pp. 3-42). Berlin, Heidelberg: Springer.

Panichpapiboon, S., & Pattara-atikom, W. (2012). A Review of Information Dissemination Protocols for Vehicular Ad Hoc Networks. IEEE Communications Surveys & Tutorials , 784 -798.

Pekşen, Y., & Acarman, T. (2012). Relay of multi-hop safety message based on beaconing in VANET. 2012 IEEE International Conference on Vehicular Electronics and Safety (ICVES 2012). Istanbul, Turkey: IEEE.

Pfeifer, G., & Law, J. (2022, November 1). GCC 4.4 Release Series Changes, New Features, and Fixes. Retrieved from gcc.gnu: https://gcc.gnu.org/gcc-4.4/changes.html

Piotr, O., & Piotr, Z. (2014). Review of Simulators for Wireless Mesh Networks. Journal of Telecommunications and Information Technology, 82-89.

Pressas, A., Ali, F., Tian, D., & Nekovee, M. (2017). Contention-based learning MAC protocol for broadcast vehicle-to-vehicle communication. 2017 IEEE Vehicular Networking Conference (VNC). Torino: IEEE

Pressas, A., Sheng, Z., Ali, F., & Tian, D. (2019). A Q-Learning Approach With Collective Contention Estimation for Bandwidth-Efficient and Fair Access Control in IEEE 802.11p Vehicular Networks. IEEE Transactions on Vehicular Technology, 9136 - 9150.

Rawat , D., Popescu, D., Yan, G., & Olariu, S. (2011). Enhancing VANET Performance by Joint Adaptation of Transmission Power and Contention Window Size. IEEE Transactions on Parallel and Distributed Systems, 1528 - 1535.

Rehman, O. M., Bourdoucen, H., & Ould-Khaoua, M. (2015). Forward link quality estimation in VANETs for sender-oriented alert messages broadcast. Journal of Network and Computer Application, 23-41.

Reinders, R., Eenennaam, M., Karagiannis, G., & Heijenk, G. (2011). Contention window analysis for beaconing in VANETs. 2011 7th International Wireless Communications and Mobile Computing Conference. Istanbul, Turkey: IEEE.

Riedmiller, M., Gabel, T., Hafner, R., & Lange, S. (2009). Reinforcement learning for robot soccer. Springer Nature, 55–73.

Riley, G., & Henderson, T. (2010). The ns-3 Network Simulator. In K. Wehrle, J. Gross, & M. Güneş, Modeling and Tools for Network Simulation (pp. 15 - 34). Berlin, Heidelberg: Springer.

Rosenblatt, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. Cornell Aeronautical Laboratory, Psychological Review, 386 - 408.

Royer, E., & Toh, C.-K. (1999). A review of current routing protocols for ad hoc mobile wireless networks. IEEE Personal Communications, 46 - 55.

SAE J2735 DSRC Message Set. (2009, November). Retrieved from United states Department of Transportation : https://www.arc-it.net/html/standards/standard17.html

Saponara, S., & Gagliardi, A. (2018). Modeling and Analysis of 802.11p Physical Layer for V2X Connected Transport Systems Considering Harsh Operating Conditions and HW Device Performance. 2018 International Conference of Electrical and Electronic Technologies for Automotive. Milan, Italy: IEEE.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. (2014). Deterministic policy gradient algorithms. ICML'14: Proceedings of the 31st International Conference on International Conference on Machine Learning, I-387–I-395.

Slavik, M., Mahgoub, I., & Alwakeel, M. (2014). Analysis and evaluation of distance-to-mean broadcast method for VANET. Journal of King Saud University - Computer and Information Sciences, 153-160.

Slavik, M., Mahgoub, I., & Sibai, F. (2011). The distance-to-mean broadcast method for vehicular wireless communication systems. 2011 International Conference on Innovations in Information Technology. Abu Dhabi, United Arab Emirates: IEEE.

Spaan, M. (2012). Partially observable markov decision processes. Partially observable markov decision processes, 387- 414.

Sung, Y., & Lee, M. (2012). Light-Weight Reliable Broadcast Message Delivery for Vehicular Ad-Hoc Networks. 2012 IEEE 75th Vehicular Technology Conference (VTC Spring). Yokohama, Japan: IEEE.

Sutton, R., & Barto, A. (2018). Reinforcement Learning: An Introduction. London, England: The MIT Press.

Tang, X., Geng, Z., Chen, W., & Moharrer, M. (2017). Data Dissemination Based on Fuzzy Logic and Network Coding in Vehicular Networks. Wireless Communications and Mobile Computing.

Wang, S., Yang, B., Gu, F., & Shi, X. (2016). A novel reliable broadcast protocol for VANET's safety applications. 2016 6th International Conference on Electronics Information and Emergency Communication (ICEIEC). Beijing, China: IEEE.

Wang, X., & Giannakis, G. (2005). CSMA/CCA: a modified CSMA/CA protocol mitigating the fairness problem for IEEE 802.11 DCF. 2005 1st International Conference on Multimedia Services Access Networks, 2005. MSAN '05. Orlando, FL, USA: IEEE.

Wang, Y., Ahmed, A., Krishnamachari, B., & Psounis, K. (2008). IEEE 802.11p performance evaluation and protocol enhancement. 2008 IEEE International Conference on Vehicular Electronics and Safety. Columbus, OH, USA: IEEE.

Watkins , C. J. (1989). Learning from Delayed Rewards. PhD Thesis. Retrieved from https://www.cs.rhul.ac.uk/~chrisw/new_thesis.pdf

Watkins, C., & Dayan, P. (1992). Q-learning. Springer .

Wisitpongphan, N., Tonguz, O. K., Parikh, J., Mudalige, P., & Bai, F. (2007). Broadcast storm mitigation techniques in vehicular ad hoc networks. IEEE Wireless Communications, 84 - 94.

Wu , Q., Nie, S., Fan , P., Liu , H., Qiang , F., & Li , Z. (2018). A Swarming Approach to Optimize the One-Hop Delay in Smart Driving Inter-Platoon Communications. Sensors.

Wu, C., Chen, X., Ji, Y., Ohzahata, S., & Kato, T. (2015). Efficient Broadcasting in VANETs Using Dynamic Backbone and Network Coding. IEEE Transactions on Wireless Communications, 6057 - 6071.

Wu, C., Ohzahata, S., Ji, Y., & Kato, T. (2014). A MAC protocol for delay-sensitive VANET applications with self-learning contention scheme. 2014 IEEE 11th Consumer Communications and Networking Conference (CCNC). Las Vegas, NV, USA: IEEE.

Wu, Q., Nie, S., Fan, P., Liu, H., Qiang, F., & Li, Z. (2018). A Swarming Approach to Optimize the One-Hop Delay in Smart Driving Inter-Platoon Communications. Sensors (Basel).

Xiaolan, T., Zhi, G., Wenlong, C., & Mojtaba, M. (2017). Data Dissemination Based on Fuzzy Logic and Network Coding in Vehicular Networks. Wireless Communications & Mobile Computing.

Yu, J., Antonio, A. d., & Mora, E. V. (2022). Deep Learning (CNN, RNN) Applications for Smart Homes: A Systematic Review. Computers.

ZeroMQ. (2022). Get started. Retrieved from zeromq.org: https://zeromq.org/get-started/

Zhu, W., Gao, D., & Foh, C. H. (2015). An Efficient Prediction-Based Data Forwarding Strategy in Vehicular Ad Hoc Network. International Journal of Distributed Sensor Networks.

Publications and Participations

Multi-hop broadcast protocol based on smart fuzzy inference for vehicular Ad-Hoc network.

3rd Smart Cities Symposium (SCS 2020)

Date of Conference: 21-23 September 2020

Authors: Omar Bani Fayyad*, Prof. Mohab Mangoud

الملخص

في هذه الأطروحة، تم تطوير بروتوكلات خاصة بشبكة السيارات المتحركة لمعالجة مشاكل البث واسع الانتشار من اجل تحسين سعة الموجة ونسبة تسليم الحزمة.

الشبكة اللاسلكية للسيارات المؤقتة هي حل واعد في نظام النقل الذكي والمدن الذكية لتعزيز سلامة الطريق وكفاءة حركة المرور وراحة السائق والركاب. تم إجراء العديد من الدراسات في هذا المجال لتوفير بروتوكولات بث موثوقة وفعالة للنطاق الترددي لشبكات السيارات اللاسلكية المؤقتة (VANET) القائمة على IEEE 802.11p. المشكلة الرئيسية التي سيتم التركيز عليها في هذه الرسالة هي عدم كفاءة بروتوكولات البث VANET وكيفية الوصول إلى قناة البث، مما يؤدي إلى كم هائل من البث اللاسلكي للمعلومات واصطدام الرسائل وبالتالي إهدار نطاق القناة وتقليل معدل توصيل المعلومات. تستخدم VANET بروتوكول بث تقليدي يستخدم تقنيات البث لنشر أنواع مختلفة من الرسائل بين المركبات داخل نطاق الاتصال. الغرض من ذلك هو إعلام جميع المركبات عن أي أحداث أو حوادث محتملة. يتطلب البروتوكول التقليدي للبث من كل مركبة إعادة بث نفس الرسالة التي تتلقاها للمركبات في محيطها، مما يؤدي إلى ارسال عدد كبير من نفس الرسائل. هذا، بدوره، يؤدي إلى استلام المركبات لنفس الرسالة مرات عديدة، مما يؤدي إلى إضاعة نطاق القناة. وعلاوة على ذلك، خلال الوصول إلى القناة، تواجه مركبات VANET صعوبة في الوصول إلى القناة بسبب المنافسة الزمنية بين المركبات في طبقة التحكم في الوصول إلى الوسائط (MAC)، والتي تتحمل مسؤولية تخصيص الموارد. تستخدم VANET حاليًا طريقة وصول إلى القناة الافتراضية المعروفة باسم Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). تعتمد هذه الطريقة على آلية Slotted Binary Exponential Backoff (BEB) للوصول إلى القناة بناءً على المنافسة، بهدف تقليل احتمالية التعارض بين عدة مركبات تتنافس على الوصول إلى القناة. للوصول إلى القناة، تختار المركبات قيمة عشوائية كوقت انتظار يسمى Contention Window (CW) للحصول على وصول إلى القناة. بعد كل محاولة ناجحة للإرسال، يتم مضاعفة وقت CW وإعادته إلى القيمة الدنيا لـ CW بعد نجاح الإرسال. ومع ذلك، هذه الآلية لزيادة وإعادة تعيين CW لا تتكيف مع التغييرات في ظروف الشبكة ويمكن أن تؤدي إلى تدهور أداء الشبكة.

تقترح هذه الأطروحة حلًا لمعالجة مشكلات البث التقليدي وتصادم الرسائل. يتم تقديم بروتوكول يعتمد على استدلال المنطق الضبابي لاختيار مركبة التي سوف تقوم بارسال الرساله ، للتأكد من استلام الرسائل الإذاعية بواسطة جميع المركبات ذات الصلة بأقل عدد ممكن من المراسلات. يأخذ استدلال المنطق الضبابي المقترح في الاعتبار المسافة والحركية ومؤشر قوة الإشارة المستقبلة (RSSI) عند اختيار المركبة التالية للإرسال.

وقد قدمت الأطروحة أيضًا بروتوكولات بث معززة بالذكاء الاصطناعي جديدة ومحسنة وخطط وصول للقناة لتحسين أداء خوارزمية نافذة الاشتباك النموذجية (BEB) وتحسين الأداء العام لشبكات VANET. تمت اقتراح ثلاثة خوارزميات Deep Reinforcement Learning (DRL) ، وهي Q-learning و Deep Q-network (DQN) و Actor-critic وتطبيقها.

تم إجراء محاكاة متنوعة باستخدام NS3-gym لتقييم أداء الخوارزميات المقترحة، والتي أظهرت تحسين أداء الشبكة مقارنةً بالبروتوكولات التقليدية. تم تقييم الخوارزميات المقترحة في مجموعة متنوعة من بيئاتVANET ، باستخدام الإنتاجية ومعدل تسليم الحزم (PDR) كمقاييس أداء.

جامعة البحرين

كلية الهندسة



قسم الهندسة الكهربائية والالكترونية

أنظمة البث التكيفية في شبكات المعلومات للمركبات المخصصة

أطروحة مقدمة كجزء من متطلبات الحصول على درجة

دكتوراه الفلسفة في الهندسة الكهربائية

إعداد

عمر يوسف بني فياض

20150041

إشراف

الدكتور مهاب منجود

أستاذ

جامعة البحرين

مملكة البحرين

مايو 2023