Traffic Congestion Reduction in SUMO using Reinforcement Learning Method

by

Radia Rahman Mouly
16101136
Puja Roy Rini
18201213
Ahsan Habib Ethic
16301200
Mahdi Islam Ayon
16301168

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

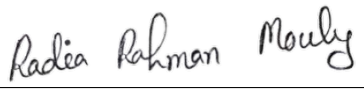
Department of Computer Science and Engineering
BRAC University
January 2021

Declaration

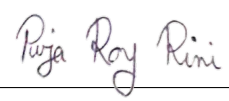
It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

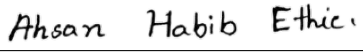
Student's Full Name & Signature:



Radia Rahman Mouly
16101136



Puja Roy Rini
18201213



Ahsan Habib Ethic
16301200

Mahdi Islam Ayon
16301168

Approval

The thesis titled "Traffic congestion reduction in SUMO using Reinforcement Learning Method" submitted by

1. Radia Rahman Mouly (16101136)
2. Puja Roy Rini (18201213)
3. Ahsan Habib Ethic (16301200)
4. Mahdi Islam Ayon (16301168)

Of Fall, 2020 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on December 8, 2020.

Examining Committee:

Supervisor:
(Member)

DR.Md. Khalilur Rahman
Associate Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

DR.Md.Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

DR.Mahbubul Alam Majumdar
Professor
Department of Computer Science and Engineering
BRAC University

Ethics Statement (Optional)

We, hereby declare that this thesis is based on the findings of our research. All other materials used have been properly noted in the document. This work, neither in whole nor in part, has never been submitted by anyone to any other university or institution for the award of any degree.

Abstract

The exemplary traffic controlling system is getting helpless because of urbanization and a consistently expanding populace. Living in a cutting-edge time of science and innovation, an advanced arrangement is a beggar description. Reinforcement learning appears to be the advanced promising answer for this endless issue. Thus, proposing a fitting and dynamic methodology to meet the excessive necessity is a significant part of the traffic control system. Our main objective is to using different algorithms in an environment to get the best possible result in order to reducing traffic congestion. Our algorithm ensured the best possible result by comparing different parameters in a SUMO(Simulation of Urban MObility) generated data-set. Firstly, we obtained a result by performing a normal simulation and then performed Q-Learning, Greedy Approach, SARSA, and Bias Q-Learning algorithms. We compared the results from the performed algorithms afterwards. The research is expected to improve productivity in bustling cities by effectively reducing traffic congestion.

Keywords: Traffic congestion, Reinforcement learning, Q-learning, Greedy approach, SARSA, Bias-Q-learning, MDP, SUMO.

Dedication

We would like to dedicate this research work to our parents. Without their utmost support, we could not have achieved or came so far. Also, we want to dedicate our work to Dr.Md. Khalilur Rahman sir who constantly pushed us all the way.

Acknowledgement

To begin with, all praise to the Great Allah for whom our thesis have been completed without any major interference. Further, we would like to thank our honorable supervisor Dr. Khalilur Rahman sir for his support, feedback, and guidance. He encouraged us to conduct the research, give guidance to us, and always were present to offer any help we could ask for. Moreover, we also thank our respected BRAC University for providing us with the necessary facilities and resources during our research period. Finally, special thanks to our parents; without their constant support and encouragement it might not have been possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

RL=Reinforcement Learning

MDP=Markov Decision Process

QL=Q-Learning

BQL=Bias Q-Learning

$TraCI$=Traffic Control Interface

Chapter 1

Introduction

1.1 Introduction

Traffic congestion is intolerant for present urban ars. Besides having a lot of drawbacks, it cost USD 11.4 billion only in Bangladesh [1].Exceptional Urban improvement has simultaneously brought about fast expansion in street transportation. The classic traffic controlling system is becoming vulnerable due to an ever-increasing population and urbanization. Therefore the idea of transportation has become a significant model in keeping up and furthermore building up the shrewd urban communities. The primary goal is to lessen the chance of over jamming of vehicles and furthermore see a smooth section of car without making disturbance to other passersby. As a result, proposing an appropriate and dynamic strategy to meet the remaining requirement is an important aspect of the traffic control system. The specific characteristics of the environment of the traffic control system handle the continuous changes of states and respond quickly by taking real-time traffic information and collecting data from different sensors. With the help of Reinforcement Learning algorithm, it is possible to achieve the target. The study will help to deal with erroneous conditions such as under and over both saturation of speed and density measurement which will be tested in a virtual environment developed in unity game engine.

1.2 Background and Motivation

Traffic management in general traffic monitoring and controlling which proposes support to reduce congestion is still impersonated as hot topics for research of different disciplines. Traffic congestion is a serious factor of concern in various parts around the world, especially here in Bangladesh. According to World Bank statistics, in the last 10 years, the average traffic speed decreased by 3 times and in the next decade it will fall less than 50% [1].It declines the productivity of modern civilization. In addition, the loss of GDP 7% in Bangladesh is due to traffic congestion [2]. Traffic jam can cause serious mental and physical problems which has leaded us to the death of more than 63 people each day, this year [1]. At the same time, the amount of wasted fuel increased from 1.9 billion liters to 11 billion liters. The total congestion costs were 121 billion dollars in 2011. Hence, it becomes

beggar description to solve the problem. So far, many solutions have been propose but none of them seems to be effective before the ever increasing population of the world. Recent years, study shows that the only light of hope is Machine learning and IOT. This paper focuses on minimizing traffic jams with the help of modern research field Reinforcement Learning. The paper also aims at experimenting those probable solution in simulations or virtual environments.

1.3 Research Objective

The main target of this research is to find a modern, simple but effective way to reduce traffic congestion at the same time finding the best algorithm to solve such problems.

*In addition, the objective of the research is to distinguish between efficiency of traditional traffic management and proposed systems in a virtual environment.

* Furthermore, the research intends to find a better traffic management system with the help of RL as the traffic network and controlling system is an ever developing sector and RL looks for each and every possible step to maximize the reward [3].

* After all these our objective is to observe how agents communicate and react under different scenarios [4], such as Peak hour vs. Non-peak hour, single lane road vs. multiple lane road and so on.

1.4 Problem Statement

Presently, traffic congestion problems in Bangladesh are increasing at an alarming rate. The traffic problem has become a very dangerous arena and has already inherent offending spread in the cities of Bangladesh. Undeliberate urbanization and vehicle policies have taken this problem to the point near to intolerance for different classes and occupation of people in this part of the world. One of the key reasons behind this problem is the stretched out of traditional traffic management system where constant service time of traffic signal for any point of time is nullified on point of its efficiency to handle the rise in the number of vehicles, complexities and variability's for arrival rate of vehicles in a specific traffic signal for a different day and time. And this traditional traffic system management cannot handle this uprising issue of traffic. Such kind of control system can work well when there is a limited amount of traffic in the network, but these techniques are not efficient enough for a heavy traffic volume. So only expanding road network can be a straightforward solution to resolve it but that is not enough financial friendly as well as space friendly. As per classic systems, most of them are static and do not interact with the environment or the system network. The traffic network is utmost controlled by microcontroller or linear programming which really cannot cope up with increasing numbers of private vehicles. In addition, the network system is a complex structure that needs sophisticated and modern technology like AI. In advanced techniques as the paper proposes, traffic control systems are divided into groups or actions

like trip time, jam area, vehicle density. In the system, we also try to adapt this technique one step advanced with taking the next probabilities of traffic volume on our account. We are detecting the prediction of traffic arrivals at signal points and adjust the signal timings to optimize an objective function. Variable message signs

can be installed, which communicate with speed limits to car drivers. These cars' speed limits can be adjusted depending on the situation of traffic conditions and such speed limits can create a positive influence on traffic flow [5]. The system needs to decide automatically how the problems should be solved and respond a dynamic and frequently changing environment and should be adaptive in the sense. Indicated characteristics motivate the introduction of an intelligent software system in this domain, also known as intelligent agents [6]. We also look forward to implementing a multi-agent system for network traffic signal control.

As Markov Decision Process suggests [7], formulating a traffic flow optimization problem and we find that policies dictating how speed limits should be assigned to highway sections to reduce traffic congestion and our objective is to observe how agents communicate and react under different scenarios, such as Peak hour vs. Non-peak hour, single lane road vs. multiple lane road and so on by applying Q-learning [8]. Traffic predictions can be included in our method. RL can be used to approximate policies defining the states and our objective is to simulate the scenario in the virtual environment so that our agent can work fine in the real world. Focusing in particular simple yet studied environment: a map appointed by traffic lights, we want to regulate through an agent perceiving the current condition of traffic environment and take the advantage of the experience carried out in a tool for Simulation of Urban MObility (SUMO) which is providing a artificial but realistic environment in which the research of the outcome of potential management actions can be carried out. With the help of simulations we show that our methods are able to reduce travel time on congestion periods under high traffic demanding small road networks and policies sufficiently robust to deal with inaccurate speed and density measurements.

1.5 Our Contributions

We did a lot of contributions here :

* To reduce the traffic congestion we implemented 4 different algorithms such as Q-learning, SARSA, Greedy Approach, Bias Q-learning.
* We compare all the algorithms to find out the best possible one in a specific environment.
* We constructed the graphs to observe the slight changes in waiting time, waiting cars for different algorithms.
* Making the Q-learning biased so that it can perform better results for reducing the waiting time and steps of the traffic system.

1.6 Thesis Orientation

We divide the whole paper in different sections review different papers and algorithms for this research. In chapter 2, we reviewed the preceding work papers based on these methods and other similar attempts with their drawbacks based on which we came up with our approach. In chapter 3 the methodologies are described with the description of model, frame and softwares. Moreover, in chapter 3 result analysis of all the algorithms are shown. Then, in chapter 4 performance of all the algorithms are compared with each other. The conclusion and future ideas, scoop and limitations are described in chapter 5.

Chapter 2

2.1 Literature review

Traffic congestion has become a global problem nowadays. For the rapidly increasing amount of motor vehicles on the roads, heavy traffic transpires in the roadways. To lessen the traffic, many variants of researches have already taken place around the world. In the field of Computer science, advanced Machine Learning methodologies such as Reinforcement Learning have already been conducted in the traffic management system lately. Q-learning algorithm, on the other hand, is analogous to dynamic programming. It has been used to monitor the real-time situation found in the surrounding which responds quite fast. Here we have focused on the previous researches done concerning the Reinforcement Learning algorithm, Deep Q-learning algorithm, and Markov Decision Process to associate with our research.

Authors LA and Bhatnagar have tried a Reinforcement Learning algorithm with function approximation for traffic signal control. Their algorithm consolidates state-activity features and is effectively implementable in high-dimensional settings. Their research doesn't require exact data on line lengths and passed times at every path, rather works with the previously mentioned depicted data. The quantity of data that their calculation requires is straight to the number of flagged paths, in this manner prompting a few sets of magnitude reduction in the computational multifaceted nature [9]. The authors further collected some statistical traffic data and analyzed it later on. They mainly worked with Q-Learning with full-state representation and Q-learning with function approximation. It also shows how efficiently the Q-learning algorithm can manage the traffic signals not only under saturation but also over saturation.

In [10] the authors proposed a multi-objective control algorithm dependent on reinforcement learning for urban traffic signal control, named multi-RL. A multi-agent structure has been utilized to depict the trafficking framework. A vehicular specially appointed system has also been utilized for the information trade among specialists. A reinforcement learning algorithm has been applied to foresee the general estimation of the advancement target given vehicles' states. The arrangement which limits the combined estimation of the streamlining objective is considered the ideal one. To make the technique versatile to different traffic conditions, the authors have presented a multi-objective control scheme in which the streamlining objective is chosen adaptively to ongoing traffic states. The advancement targeted incorporates

the vehicle stops, the normal pausing time, and the most extreme line length of the following crossing point. On the other hand, they accommodated control to the buses and the crisis vehicles through their model. The recreation results demonstrated that their calculation could perform more efficiently than customary traffic light control techniques.

In [11] the authors have focused on the improvement of traffic light controllers utilizing a multi-operator, model-based reinforcement learning, or estimated real-time dynamic programming approach. Their techniques have been used to enhance singular traffic lights locally, yet the advancement considers traffic blockage at neighboring traffic lights, to such an extent that there is certain participation between them. They tried to minimize the total travel time of all the vehicles. Furthermore, they have used two states to represent the reward domain. One was a traffic light-based state representation, which speaks to all conceivable traffic arrangements around a traffic light intersection. Another one was a car-based state representation, which speaks to the world state from the point of view of individual cars.

Authors Findler and Stapp on the contrary portray a system of streets associated with traffic light-based expert systems [12]. The expert systems can impart to allow for synchronization. Execution of the system relies upon the standards that are utilized. For each traffic light controller, the arrangement of rules can be enhanced by breaking down how frequently each standard flame, and the achievement it has. The framework could even learn new guidelines. Findler and Stapp indicated that their framework could improve execution, yet they needed to make some disentangling presumptions to maintain a strategic distance from a lot of calculations.

In [13] the authors have concentrated on how to choose the traffic signs' term dependent on the gathered information from various sensors and vehicular systems. They proposed a deep reinforcement learning model to control the traffic light. In the model, they have evaluated the unpredictable traffic situation as states by gathering information also, partitioning the entire convergence into little frameworks. The time changes of a traffic light were the activities, which were demonstrated as a high-measurement Markov choice procedure. The reward was the combined holding up time distinction between two cycles. To tackle the model, a convolutional neural system was utilized to map the states to rewards. The proposed model was created of a few parts to improve the exhibition, for example, dueling system, target organize, twofold Q-learning system, organized experience replay. Finally, they assessed their model via simulation in the Simulation of Urban Mobility (SUMO) in a vehicular system, and the reenactment results showed the effectiveness of their model in controlling rush hour gridlock lights.

In [14] the focus point of the authors was on the best way to improve transfer speed productivity through versatile directing when the limit is saved on all Virtual Paths. The creators initially looked at two Virtual Path limit reservation methodologies. They at that point structure and assess computationally achievable Markov Decision Process-based routing algorithm and show that the system blocking likelihood can be fundamentally diminished by MDP steering.

In the paper [15] authors Arel, Liu, Urbanik, and kohl demonstrated that a difficult utilization of artificial intelligence frameworks includes the scheduling of traffic lights in multi-intersection point vehicular networks. The paper presented a novel utilization of a multi-specialist framework and Reinforcement Learning (RL) structure to acquire a productive traffic light control strategy. The last is pointed toward limiting the average delay, clog, and the probability of convergence cross-blocking. A five-crossing point traffic network has been concentrated in which every convergence is represented by a self-sufficient wise agent. There were two kinds of agents, a central agent, and an outbound agent. The outbound agents planned traffic lights by following the longest-Queue First (LQF) algorithm, which has been demonstrated to ensure security and reasonableness, and team up with the central agent by giving it nearby traffic insights. The central agent learns a worth capacity driven by its nearby and neighbors' traffic conditions. The novel system proposed here used the Q-Learning algorithm with a feedforward neural network for value work estimates. Trial results exhibit the upsides of multi-agent RL-based authority over LQF administered detached single-convergence control, along these lines making ready for productive circulated traffic light control in complex settings.

Authors Vidali, Crciani, Vizzari, and Bandini said in their paper [16] that Traffic observing and control, just as traffic recreation, are yet critical and open difficulties despite the noteworthy researches that have been completed, particularly on artificial intelligence ways to deal with these issues. Their paper presented a Reinforcement Learning (RL) way to deal with traffic lights control, combined with a microscopic operator-based test system (Simulation of Urban MObility- SUMO) giving a manufactured, however, practical condition in which the investigation of the result of potential guideline activities can be done. The paper presented the methodology, inside the momentum research scene, at that point the experimental test setting and accomplished outcomes are depicted.

In [17] the authors said, the ongoing advances in joining deep neural network models with reinforcement learning strategies have appeared promising expected outcomes in tackling complex control issues with high dimensional state and activity spaces. Motivated by these victories, in the paper, they assembled two sorts of reinforcement learning algorithms: deep policy-gradient and value function-based agents which can foresee the most ideal traffic signal for a traffic crossing point. At each time step, these versatile traffic light control operators get a preview of the present status of a graphical traffic test system and produce control signals. The policy-based gradient maps its perception legitimately to the control signal, anyway the value-function-based agent first estimates values for all legitimate control signals. The operator at that point chooses the ideal control activity with the most elevated value. Their strategies showed promising results about a traffic network recreated in the SUMO traffic test system, without suffering from unsteadiness issues during the preparation cycle.

Furthermore, authors Reddy and Mehta [18] stated that traffic congestion at intersections or on streets might be seen because of numerous reasons like moderate driving, expanded vehicle lines, and so on. In some crises when vehicles are halted for a longer period traffic jams may happen. Adaptive traffic light system is a traf-

fic management technique used to control the traffic by encouraging the signs to momentarily conform to the current traffic request. Adaptive traffic light capacities by using both equipment and programming coordination. Q-learning needs the previously planned exact type of the earth for choosing an activity. As a substitute, they could receive a dynamic correspondence system to discover the cooperation between state, activity, and awards of that specific condition. The current traffic signal works dependent on the predetermined traffic flow information to separate brief timeframe expectations which assist with assessing the result on the sign controlling framework. On the off chance that a solitary model is utilized, at that point at every single time adaptive traffic signal control agents need to gather photos of the current state of the traffic and create control signals. They have executed event, replay, and ideal systems to improve the consistency of the calculation. The primary point in planning the calculation is to control stuffed traffic and for this, we have fused the dynamic network with the direct sign course of action.

In [19] the authors said, relentless clogs that are shifting in quality and span in the thick traffic networks are the most unmistakable obstacle towards reasonable portability. Those sorts of clogs can't be satisfactorily settled by the customary Adaptive Traffic Signal Control (ATSC). The presentation of Reinforcement Learning (RL) in ATSC handled those sorts of blockages by utilizing on-line learning, which depends on the experimentation approach. Besides, RL is inclined to the dimensionality revile identified with the state–activity space size dependent on which a non-linear quality capacity is inferred. The Deep Reinforcement Learning (DRL) structure utilizes Deep Neural Networks (DNN) to process crude traffic information to rough the quality capacity of RL. The paper gave a complete examination of the latest DRL approaches utilized for the ATSC calculation plan. Extraordinary accentuation is set to outline the traffic state portrayal and multi-agent DRL structures applied for the enormous traffic networks. Best practices are accommodated by picking the sufficient DRL model, hyper boundaries tuning, and model engineering plan. Lastly, their paper gave a conversation about the significance of the open traffic information idea for the broad use of DRL in reality ATSC.

The capacity to apply real-time, adaptive control of transportation measures is the center of numerous shrewd transportation frameworks decision support instruments [20]. Reinforcement learning, an artificial intelligence methodology going through advancement in the machine learning network, offers key points of interest in such manner. The capacity of a control specialist to learn connections between control activities and their impact on the climate while seeking after an objective is an unmistakable improvement over pre-specified models of the climate. Pre-specified models are an essential of regular control techniques and their exactness restricts the exhibition of control specialists. This paper contains a prologue to Q-learning, a basic yet ground-breaking reinforcement learning calculation, and presents a contextual analysis including application to traffic light control. Empowering aftereffects of the application to a detached traffic light, especially under factor traffic conditions, are introduced. A more extensive exploration exertion is plot, including augmentation to direct and organized sign frameworks and mix with dynamic course direction. The examination objective includes ideal control of vigorously clogged traffic across a two-dimensional street organization—a testing task for ordinary traffic light control

systems.

Traffic congestion causes significant issues, for example, delays, increased fuel utilization and extra pollution [21]. In the paper the authors proposed a technique to improve traffic flow, based on reinforcement learning. They showed that a traffic stream improvement issue can be planned as a Markov Decision Process.They used Q-learning to learn strategies directing the greatest driving velocity that is permitted on a highway, such that traffic congestion is reduced. A significant distinction between their work and existing methodologies was that they took traffic expectations into account. An arrangement of recreation tests showed that the subsequent approaches essentially diminish traffic congestion under high traffic demand, and that incorporation of traffic forecasts improves the nature of the subsequent policies. Additionally, the arrangements are adequately robust to bargain inside precise speed and density estimations.

2.2 Preliminaries

In this part a elaborate description of the models, algorithms and road architecture will take place.

2.2.1 Road Architecture and design

A simple definition of road is: A thoroughfare, route, or way on land between two places that has been paved or otherwise improved to allow travel by foot or by some form of conveyance (including a motor vehicles, cart, bicycle etc.). For the betterment of traveling or reaching somewhere within a short amount of time, road architectures are being changed day by day. There are a lot of road architectures or design such as (euverus, 2017)

Table 2.1: Table, floating element

Four lane roads	Diamond Interchange
Six lane roads	Diverging Diamond Interchange (DDI)
Eight lane roads	Single Point Urban Interchange (SPUI)
Standard Roundabout	Two level Roundabout
Large Roundabout	Three level Roundabout
Custom Roundabout with slip lanes	Continuous flow Intersection

Here, in this paper describes about the four lane and six lane road architecture with traffic light and without traffic lights and dedicated left turn lane.

In 4 lane roads fig1 shows the road architecture is designed in a way so that from each side the vehicles can go in two lanes at the same time but there is always a chance of traffic congestion. For that there are some methods by which the traffic

Figure 2.1: a. lane without traffic lighted left turn b. lane with traffic lighted left turn c. lane with dedicated left turn d left turn

scenario changes a lot. In fig1 it is observed that as there is no traffic lights or signals the condition is much hazy and more traffic is common here but if we implement a traffic signal lights (fig2) then the traffic condition becomes better than fig1. It is the result of traffic signals that only by implementing signal lights traffic conditions is being more stable. However, in fig3 a dedicated left turn is shown in 4 lane roads. A dedicated left turn road with traffic signals shows how vehicles can turn left using the left turn lane and it is impacting to reduce the traffic in a very efficient way. Moreover, in six lane roads it is much more efficient to use these traffic lights and dedicated left turn roads. As a result traffic jams reduce much more efficiently.

2.3 Background Study

2.3.1 Algorithm Description

(a) Reinforcement Learning

The purpose of any modern machine learning technique is to make the agent more diversified and interactive with the environment. An established sector to achieve such characteristics is RL. [22]. The technique concentrates on training agents to mix with the environment by maximizing the reword. This is much similar to training a pet. If the pet performs a correct task as command, then it receives reward. More precisely the agent learns how to choose an action and change the state in such a way that it secures maximum reward, even in the long run. In this research, the agent aims to minimize the traffic congestion, so it takes an action A_t, (in this case controlling the time of the traffic lights) by observing the environment (in our case the intersections) and based on the action receives reward R_t. So, time after time the agent reaches to a human level traffic management capabilities sometimes even better. A brief image is shown in Figure 2.2

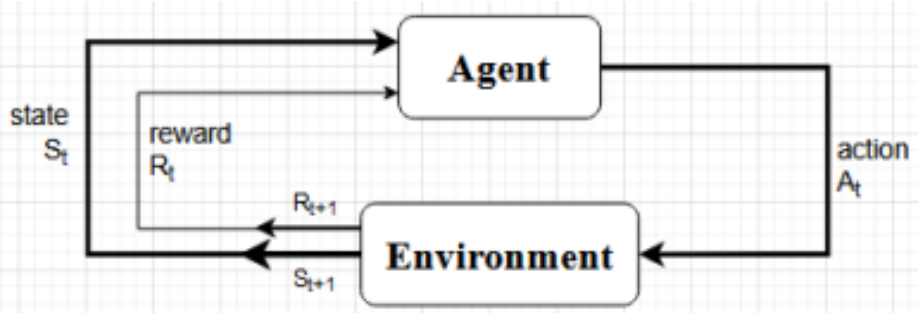


Figure 2.2: Simplified model of RL

(b)Markov Decision Process

In order to shape a RL algorithm few characteristics must be defined and formally it is called MDP (Markov Decision Process). MDP is the roadmap of the algorithm talking about the action based on any state aiming to maximize reward. So there are three parts to talk in MDP. They are,
*__State__: Arbitrarily state is the change of the environment due to any action taken by the agent. For example, in this research, the environment had an state St , the agent takes an action at and that action affects the environment and it changes to another state called S_{t+1}
*__Action__:In the context of this research, actions are basically controlling the time of specific traffic lights. The traffic light has its own combination. It will show Green for a time, then Red for a time period and in between these two lights it shows Yellow light as a transition. The action is to control the time for Red and Green light. With different time combinations the agent can affect the state which essentially affects the traffic congestion. In our case the agent has seven actions to choose. The actions are [8, 16, 24, 32, 48, 52, 64]. The traffic light will be activated for any of these times chosen by the agent as action. Point to be noted, these are not conventional times like we know in a minute or second unit. Rather these are time units for the experiment.
*__Reward__:It is a number or set of numbers to justify how effective the action is. So it is a performance indicator as well. For instance, the agent takes an action A_t on time t and due to that action state S_t changed. Now comparing the previous and current state agent receives the award R_t. In next action A_{t+1} the state changes to S_{t+1}. If the state makes the congestion better again the agent receives the award R_{t+1}. The agent may receive less or even negative reward R_{t+2} for action A_{t+2}. The less traffic congestion in any intersection, the more reward for the agent. The reward is totally dependent on a function, known as reward function. For this experiment, if the waiting time of the car in the intersection is in between 500 to 600 time unit then the agent receives -3, for 300 to 500 time unit the reward is +3. In a 150 to 300 time unit the reward is +5. For less than 150 time unit the agent receives +7. For other cases the allocated reward is -9.

(c)Q-Learning

This paper aims to suggest multiple algorithms. First let us discuss Q-Learning which is simple but powerful enough to operate a complex environment like this one we are working on. It is a RL based technique which is used for learning the optimal policy in Markov Decision Process.[23] The goal of Q-Learning algorithm is to find an optimal policy that will maximise the total reward. It uses q-values for each action-state pair. The algorithm uses a q-table for this purpose. For each episode the q-table is updated using q-function which will be briefly described in Chapter-3. In the q-table columns are the action and rows are the state. A simple q-table is shown in Figure 2.3. For each state an action is selected based on the policy. It uses exploration and exploitation to learn the unknown environment or model.[23] Hence it is called a model free algorithm.

→ action on column
↓ state on row

7.99185933623721	14.2733155816872	13.248677572517	15.5076024698892	10.0935380938949	17.0423238134869	13.6001741169933
12.0220244763764	14.9854245469914	14.4033985785156	11.3946237431826	16.5756959481049	10.883111174945	13.137764944244
12.222280125419	11.9461442343573	13.1303363646644	15.590472621375	10.7364065345617	12.3295366348835	18.3479748798876
11.7111316425724	17.2355364712717	13.5986238708587	15.6572083536337	10.2720423641436	10.9122042483438	17.0377669228321
13.9270644303487	16.843380511591	13.4105887387825	15.9455827015921	12.3806318700611	13.2010655353528	12.9137880180755
15.535421824465	11.823578115264	6.70597069830099	12.0123467265059	16.04807186234	15.9970359281842	14.2757232991006
16.7090751620708	13.5385637539639	11.7574534997829	17.0272153261125	15.6765975478379	10.0718367987668	12.8200588650658

Figure 2.3: Small part of a q-table

(d)SARSA

It is an advanced Reinforcement Learning algorithm slightly different from Q-learning. SARSA follows on policy which means the agent learns the value function or the q-function as per the current action, that is derived from the current used policy.[24] On the contrary, in QL the agent learns the value function as per the action. The action itself is derived from another policy. So, QL learns from the q-value calculated from different policies whereas SARSA learns from the q-value calculated from the same policy. Again the equations and evaluation is explained in Chapter-3 and Chapter-4.

(e)Greedy Approach

As the name suggests, it follows a greedy approach. Simply, it opens the road among four which has the most waiting time for cars. Though it is not a learning or model free algorithm like other algorithms discussed in this paper. But it has a potential to reduce traffic congestion which is discussed briefly in Chapter-4. However, this approach is followed in real life traffic controlling. So the results are much interesting to observe in simulation.

(f)Bias Q-Learning

Just like its predecessor QL, BQL has some similar characteristics. It also depends on the q-table made of action state pair or q-values. It also tries to find optimal policy to maximize total reward. Just like QL it explores initially and then explodes when needed. But the only difference is in q-function. By adding a little bias in the equation it dramatically effects on the results. The equation is discussed in Chapter-3 and results in Chapter-4.

2.3.2 Softwares

In our paper we have used **SUMO (Simulation of Urban MObility)** and **TraCI (Traffic Control Interface)** for simulating traffic systems. SUMO is an open source, profoundly convenient, microscopic and persistent traffic simulation bundle intended to deal with huge networks. It considers multi-purpose recreation incorporating people on foot and accompanies a huge arrangement of tools for situation creation. In our project, we did all the simulations of traffic control in SUMO.
TraCI is the present moment for "Traffic Control Interface". Offering admittance to a running street traffic simulation, it permits to recover estimations of recreated objects and to control their conduct "on-line". TraCI is basically a connection between the model and the simulation. The model sends the data to TraCI and then TraCI forwards the data for simulation.
Spyder is an incredible logical condition written in Python, for Python, and planned by and for researchers, designers and information experts. It offers an extraordinary blend of the advanced altering, analysis, troubleshooting, and profiling usefulness of a complete advancement device with the information investigation, intuitive ex-

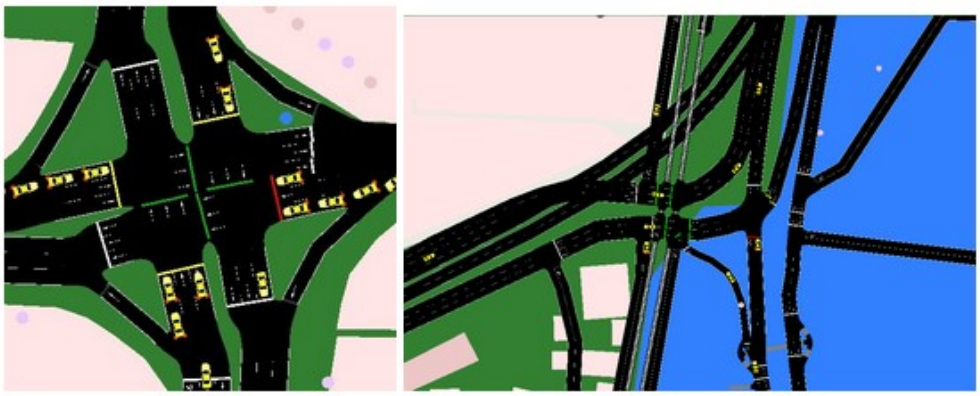


Figure 2.4: Simple view of the Environment in SUMO

ecution, deep assessment, and delightful representation abilities of a logical bundle. We used Spyder for data analysis and modeling of the project

2.3.3 Environment Analysis:

In this research the environment is designed in such a way that the cars are bound to follow some traffic rules in intersections. A simple intersection is shown in Figure 2.5. The vehicles can move through the points stated in Table 2.2:

Table 2.2: The points car can go

A to F
E to B
C to H
G to D

However, vehicles are prohibited to move through specific points that is shown in Table 2.3.

Table 2.3: The points car can not go

F to A , C to B	G to A , B to E
C to A , G to B	H to C , C to H
E to D , D to G	C to G , E to C
A to H , C to F	E to G , A to G
C to E , E to H	G to F , A to C
A to D	G to E

Environments can be changed anytime based on the situation or structure of the road. For this research we focused on a specific environment. Few photos are shown in Figure 2.4.

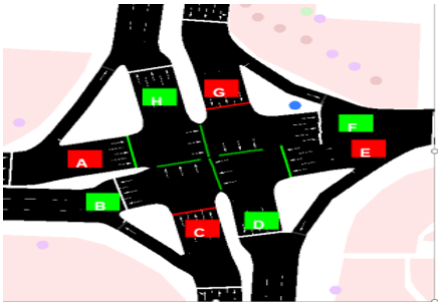


Figure 2.5: Intersection map

This environment is created based on a real-world road map located in Gulshan-1 area, Dhaka, Bangladesh. The vehicles can only go from the red color points to the green color points. If a car wants to go somewhere else then it has to take a U-turn. So, the environment has some own rules to maintain the simulations efficiently which is described in Table 2.2 and Table 2.3. The traffic control lights can choose any of this time unit (8,16,24,32,48,52,64) to initiate red or green light on a specific road marked as Red in Figure 2.5. The traffic control lights are located only near the red marked points and the changing of the lights are controlled by different algorithms which is described in chapter 2.1.2.

Chapter 3

Methodology

3.1 Workflow

3.1.1 Proposed Idea

The main goal of this paper is to perform different algorithms on an environment. So, focusing on the problem of managing traffic reduction Q-learning, SARSA, Greedy approach, Biased Q-learning algorithms are implemented.
1. Firstly, the simulation is performed using basic(default) settings of SUMO.
2. Secondly, using Q-learning where the agent has to learn by exploring the environment.
3. Thirdly, SARSA algorithm is performed on the same model where the agent learns from the Q value calculated from the same policy.
4. Fourthly, a greedy approach was performed on the same model but this algorithm is not a learning algorithm. Rather, it follows the shortest way to reduce traffic congestion.
5. Then, using Bias Q-learning the agent tries to find a different policy which potentially effects the traffic congestion.
6. Finally, the evaluation is performed based on the waiting time and waiting cars which is mentioned in chapter 4.
All the simulation was performed using SUMO. It is a powerful software to simulate real life situation in spite of being a free tool.

3.1.2 Frame Diagram

Now as we are familiar with the model and the environment, it is high time to have a closer look at the frame diagram of the experiment. As mentioned earlier, the paper aims to suggest multiple algorithms hence the experimental setup may vary. But a general diagram of the experimental setup is shown in figure 2.6.
1. First the agent takes data from the environment specifically traffic density and waiting time of vehicles.
2. Next the agent processes the data and chooses an action.

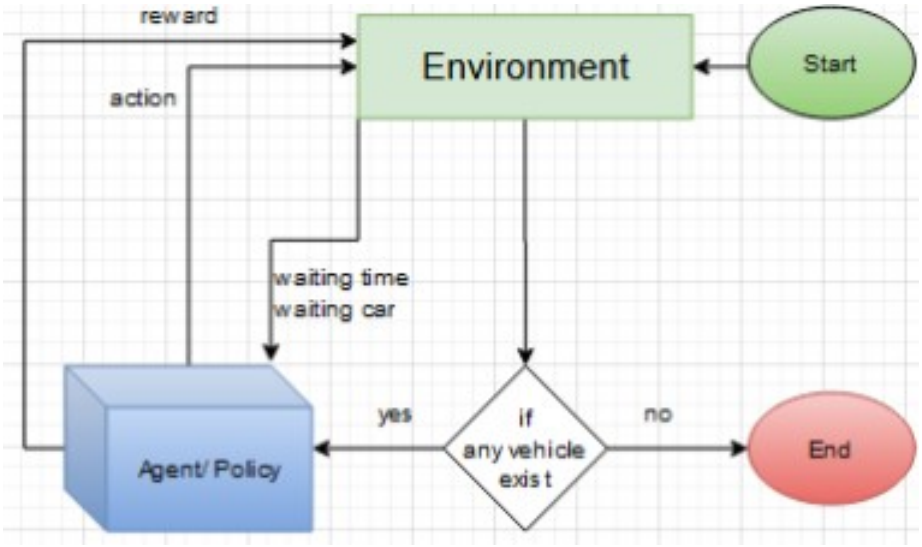


Figure 3.1: Experimental Setup

3. The action is then performed in the environment which also mean the state change.
4. Now, the agent gets a reward based on its action.
5. The whole process keeps running until there is no vehicle left in the environment.
6. When it reaches the end that means an episode is completed.

3.1.3 Simulation Setup

From the open street map, the Gulshan-1 and its surrounding area is imported in SUMO. For every algorithm the same model has been used to test. The algorithms have different characteristics and policies but the interaction with the simulation is quite same. In this regard, TraCI has been used. It is an API that connects the SUMO with the agents or algorithms. A simplified graphical representation of environment setup is shown using Graphical Representation and Pseudo Code.

1. Graphical Representation

As the graph shows in Figure 2.7, TraCI is the API that is taking and executing the information and commands. Firstly, TraCI commands to initialize SUMO and start the agent. The agent has the power to make SUMO do something like what to do when the congestion is dense or when to open which road. But all commands are executed via TraCI. Even TraCI fetches the information from SUMO. For example, how many cars are waiting at the intersection, or how long the cars are waiting and so on. In case of model free algorithms feedback is needed but in other cases no feedback is performed.

However, in normal simulation or in the simulation where no external algorithms are performed the setup changes slightly. In that case there is no agent. So all the communication and execution is directly done in between SUMO and TraCI. Most

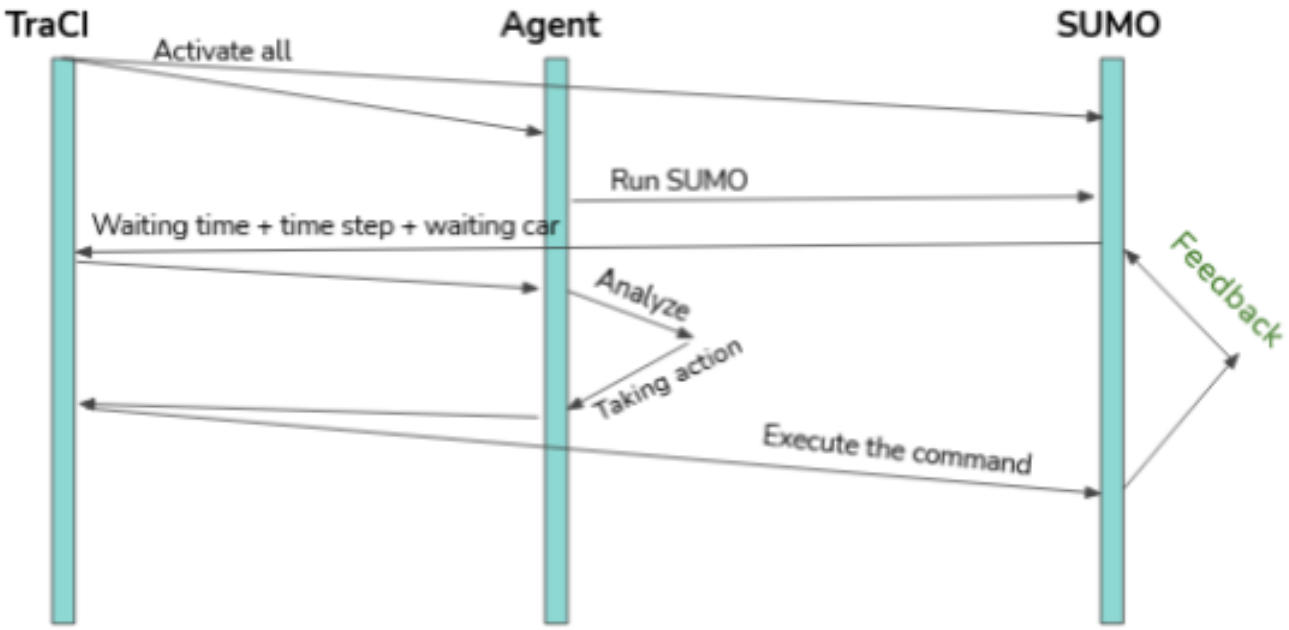


Figure 3.2: Graphical Representation of the back-end of simulation with algorithms

of the command is taken by SUMO itself using its default algorithm.

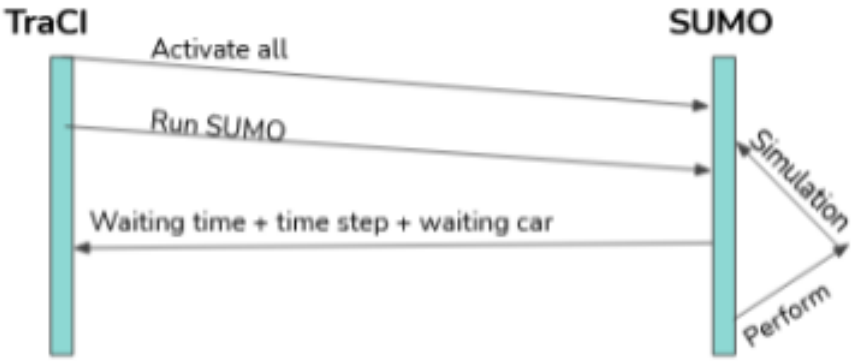


Figure 3.3: Back end of normal simulation

2. Pseudo code

The whole scenario discussed in Graphical Representation can be written in a program using the pseudo code. The simulation that is controlled by agents is stated in Table 2.4. Here it is clearly visible that action is taken by an agent which later changes the state. Waiting time and waiting car is stored for future evaluation.

On the contrary, in Table Table 2.5 a pseudo code of default simulation is stated. Simply, SUMO's default program does what it is programmed to do similar to real life scenarios. Again, Waiting time & waiting car is stored for future evaluation.

Table 3.1: Pseudo code of simulation controlled by agent

Simulation with Q-learning
1. Initialize TraCI
2. Initialize agent_function():
3. Initialize SUMO
4. while(at least one vehicle in environment == True):
5. Choose an action, a
6. Perform a step, s
7. Store waiting time and waiting car in a list
8. Get reward
9. Write waiting time, waiting car in csv file
10. End While
11. Close SUMO

Table 3.2: Pseudo code of simulation controlled by default program of SUMO

Simulation without agent
1. Initialize TraCI
2. Initialize SUMO
3. while(at least one vehicle in environment == True):
4. Perform step
5. Store waiting time and waiting car in a list
6. Get reward
7. Write waiting time, waiting car in csv file
8. End While
9. Close SUMO

Chapter 4

Algorithm Analysis

Here, in this chapter, we will mainly analyze the results of the algorithms. At the same time we will focus on relevant equations and performance of the algorithms. The discussion will follow this order: Q-learning, SARSA, Greedy Approach, Bias Q-learning.

4.1 Q-learning

In Q-learning the agent follows a q-table consisting of a state-action pair. The q-value always gets updated by following a function called q-function shown in equation - 3.1(a).

4.1.1 Equation

The updated Q value depends on previous Q value, learning rate alpha, discount rate gamma, maximum value from the state action pair. Here, the focus is the Greedy Policy in target q-value which is $r+ \ max \ a'\gamma \ Q(s', a')$. The algorithm always tries to find optimum policy while exploitation.

$$Q(s,a) = Q(s,a) + \alpha[r + maxa'\gamma Q(s', a') - (s,a)].....3.1(a)$$

4.1.2 Pseudo code

Table 4.1: Pseudo code of Q-learning based simulation

Simulation with Q-learning
1. Initialize TraCI
2. Initialize agent_function():
3. Initialize SUMO
4. while(at least one vehicle in environment == True):
5. If (exploration_rate_threshold ¿ exploration_rate):
6. Take an action from the q-table that has max q-value
7. Else:
8. Take a random action from action space
9. Perform a step, s
10. Store waiting time and waiting car in a list
11. Get reward
12. Update Q-table with Q-function
13. Write waiting time, waiting car in csv file
14. End While
15. Close SUMO

4.2 SARSA

4.2.1 Equations

Q-learning follows greedy policy, whereas SARSA follows the behaviour policy. The target Q-value in SARSA is $r + \gamma\ Q(s', a')$

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)].....3.2(a)$$

4.2.2 Pseudo Code

In Table 3.1(b) line number 6 where the agent follows Behaviour Policy which makes it different from Q-learning.

Table 4.2: Pseudo code of SARSA based simulation

Simulation with SARSA
1. Initialize TraCI
2. Initialize agent_function():
3. Initialize SUMO
4. while(at least one vehicle in environment == True):
5. If (exploration_ rate_ threshold ¿ exploration_rate):
6. Take an action from the q-table following Behaviour Policy
7. Else:
8. Take a random action from action space
9. Perform a step, s
10. Store waiting time and waiting car in a list
11. Get reward
12. Update Q-table with Q-function
13. Write waiting time, waiting car in csv file
14. End While
15. Close SUMO

4.3 Greedy Approach

In Greedy Approach, unlike Q-learning or SARSA there is no training or learning. More precisely, the shortest path algorithm is followed.

4.3.1 Pseudo Code

Table 4.3: Pseudo code of Greedy Approach Algorithm based simulation

Simulation with Greedy Approach
1. Initialize TraCI 2. Initialize agent_function(): 3. Initialize SUMO 4. while(at least one vehicle in environment == True): 5. Get maximum waiting time of each road 6. Choose an action from the action space 7. Perform a step, s (Open the road having maximum waiting time of car) 8. Store waiting time & waiting car in a list 9. Write waiting time, waiting car in csv file 10. End while 11. Close SUMO

4.4 Bias Q-Learning

4.4.1 Equation

Just like Q-learning, here in this algorithm a greedy policy has been followed. But, a simple bias is added. After calculation new q-value it is added with previous q-value and then they are divided by two in order to find the average or Current and Previous q-value. The equation is shown in 3.4(a)

$$Q(s,a) = [(Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)] + Q(s,a)]/2 \text{.....} 3.4(a)$$

4.4.2 Pseudo Code

Table 4.4: Pseudo code of Bias Q-learning based simulation

Simulation with Bias Q-Learning
1. Initialize TraCI
2. Initialize agent _function():
3. Initialize SUMO
4. while(at least one vehicle in environment == True):
5. If (exploration _rate _threshold ¿ exploration _rate):
6. Take an action from the q-table that has max q-value
7. Else:
8. Take a random action from action space
9. Perform a step, s
10. Store waiting time & waiting car in a list
11. Get reward
12. Update Q-table with q-function (bias added)
9. Write waiting time, waiting car in csv file
10. End while
11. Close SUMO

Chapter 5

Performance Evaluation

5.1 Q-Learning and Normal Simulation

In this chapter performance analysis based on Q-learning and Normal simulation are described based on waiting time. Where, x axis is the time unit steps and y axis is the waiting time.

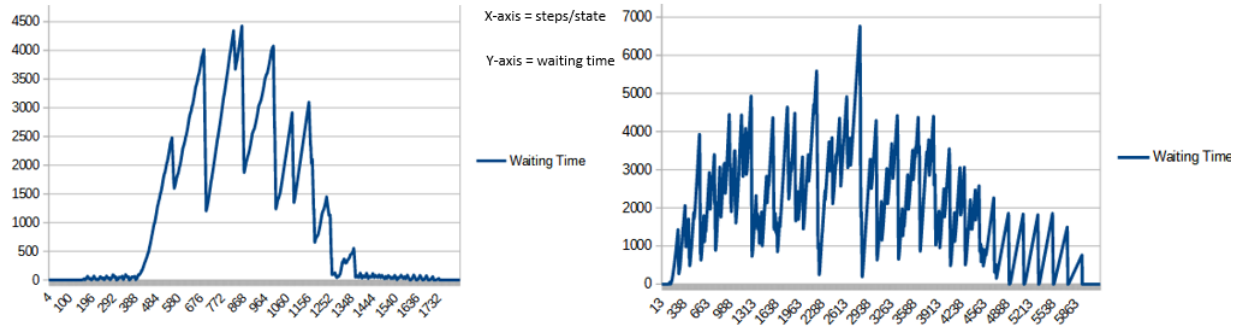


Figure 5.1: Q-learning Normal simulation waiting time comparison

After performing these two algorithms it shows that with Q-learning algorithm it completes an episode 3 times faster than the normal simulation. It means with Q-learning algorithm, decisions can be taken within 2000 units where in normal simulation it takes 6000 units to take a decision. Moreover, the graphs are made based on the outputs. Furthermore, the graphs have a clear indication that there is a huge difference in between waiting time. In, Q-learning the waiting time is only 4500 unit but Normal simulation has a waiting time of 7000 unit. So, the evaluation says Q-learning is 15% faster in terms of waiting time. When the simulation is performed it is clearly visible that vehicles need to wait less amount of time in the road queue in Q-learning than Normal simulation. Furthermore, between the two graphs shown in Figure 4.1 normal simulation waiting time is more randomly disheveled than the Q-learning one. Now, from the Figure 4.2 we can see the statistics of waiting car.

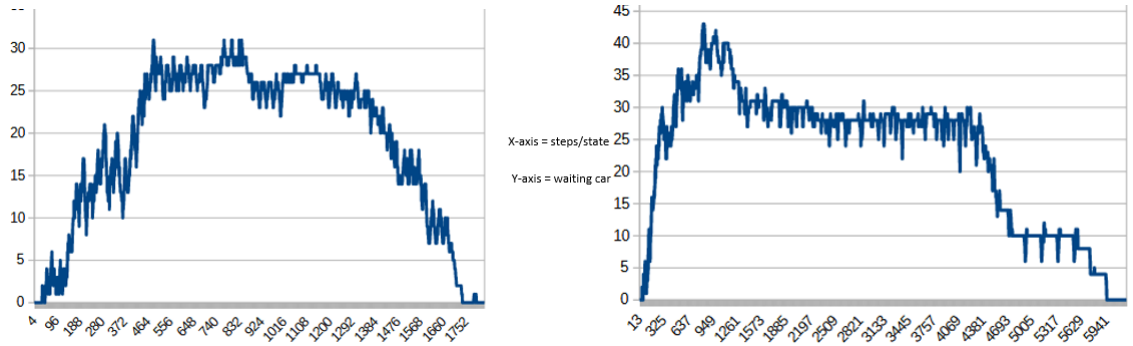


Figure 5.2: Q-learning Normal simulation waiting cars

Here, Y-axis is the number of waiting cars and X-axis is steps same as Figure 4.1. From the graphs it is visible that nearly 30-32 cars are waiting in the simulation with Q-learning on the other side this number is near 45 in Normal simulation. It proves the Q-learning is performing better than Normal simulation.

5.2 Q-Learning and Greedy approach

Previously, Q-learning and normal simulation are discussed in chapter 4.1. Now, there is a difference between Greedy and Q- Learning while performing the simulation. From the graph below The changes are visible. Previously, in the performance

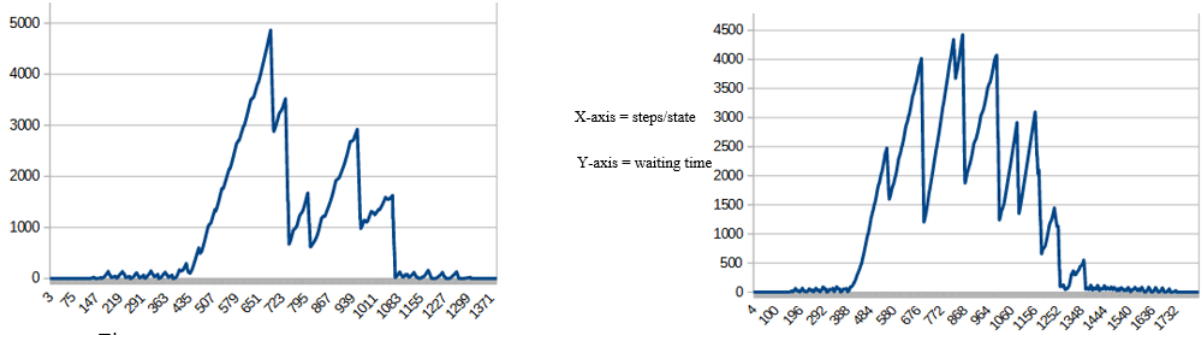


Figure 5.3: Greedy approach and Q-Learning waiting time comparison

evaluation between Q-learning and normal simulation we can see that Q-learning performs better but now Greedy approach is performing better then Q-learning in some cases. Here, X-axis and Y-axis are units and waiting time respectively. While simulating with the Greedy approach it is a faster process as it takes nearly 1500 units to complete a simulation, whereas Q-learning takes 2000 units. So, it is clearly visible that Greedy approach is (10-12) % faster in terms of decision making. Though decision-making process is faster, the waiting time for Greedy approach and Q learning seems different. It completes the episode (10-12) % slower than the Q-learning as shown in Y-axis of the figures. The greedy approach has been discussed in chapter 3. From the Figure 4.3 it is clear that, the waiting time of vehicle in Greedy Approach is 500-time unit substantial than Q-learning but it is a better algorithm than Q-learning in terms of making decisions. Furthermore, next graphs represent the difference between the car counts or more precisely waiting cars.

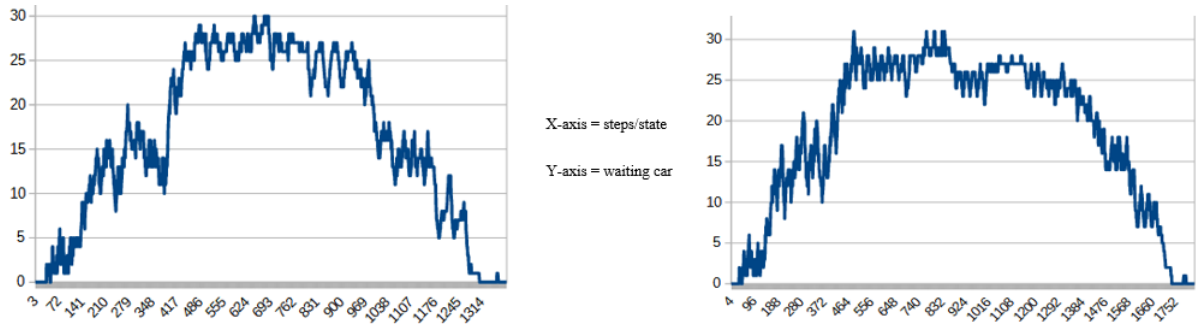


Figure 5.4: Greedy approach and Q-Learning waiting cars comparison

After analyzing the number of waiting cars in worst case scenario it is clear that for both simulations, the waiting cars are nearly 30. However, in case of Greedy approach 30 cars are waiting in a smaller number of steps. It means, in a certain part of the simulation 30 cars are waiting. Whereas, for Q-learning 30 cars are waiting in the simulation in different phases of the steps. Which also proves that Greedy approach is performing better here.

5.3 SARSA and Greedy Approach:

Now from the below graphs we can compare the SARSA and Greedy approach while performing.

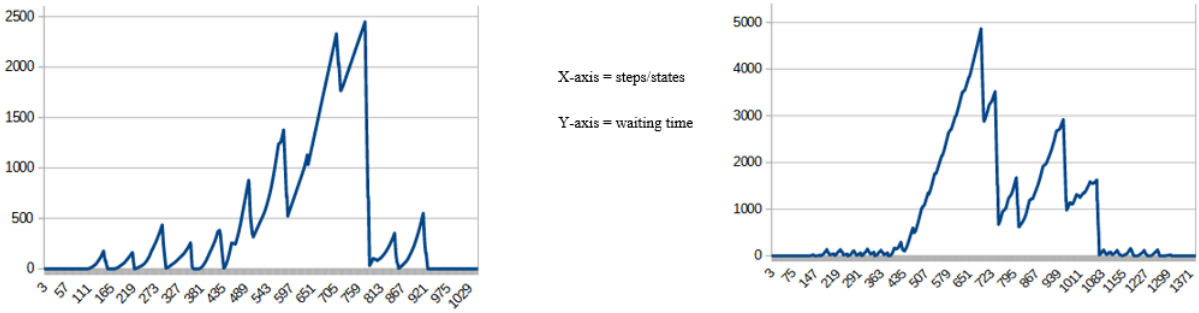


Figure 5.5: SARSA and Greedy approach waiting time comparison

Here, in both graphs X-axis and Y-axis are steps/states and waiting time respectively. In SARSA it is clear that while performing the simulation it takes only 1029 units to complete an episode. On the other side, Greedy approach takes 1371 units to complete a simulation episode. Here, the difference between SARSA and Greedy approach in terms of completing an episode is 340 units. Which is much faster than greedy approach algorithm. So, it shows SRASA can perform much better than Greedy approach. Moreover, the waiting time for SARSA is 2500 units but in Greedy approach waiting time is 5000 units. Which is 2 times faster than Greedy approach. Now, from the below graphs we can see the waiting car statistics of both simulation algorithm.

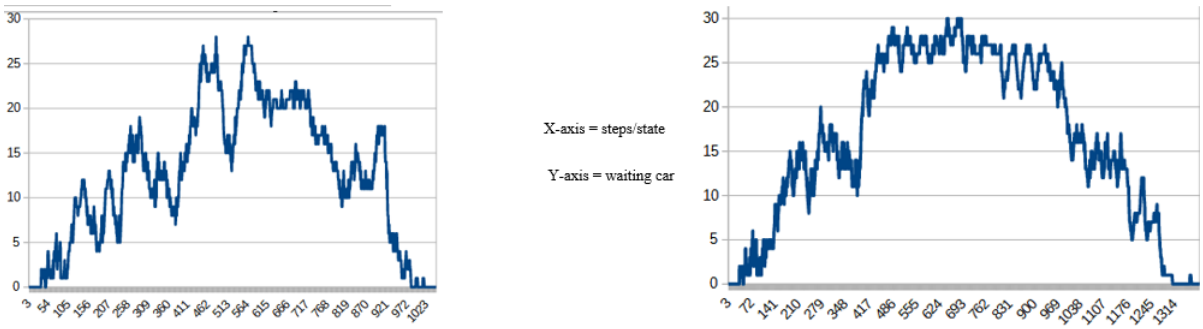


Figure 5.6: SARSA and Greedy approach waiting cars comparison

From the graphs we can see that, in SARSA near 25-28 cars are waiting in the line queue of roads but in Greedy approach the number is near 30. Moreover, in SARSA 25-28 cars need to wait only for less amount of time. On the other side in Greedy approach cars wait for a long period of time than SARSA. So, the performance evaluation for SARSA is much better than Greedy approach.

5.4 Bias Q-Learning and Q-Learning:

Q-Learning and Bias Q-Learning both have one thing in common. Both follows a greedy approach to find target value. However, the main difference arrives when calculating q value in Bias Q-Learning. Bias Q-Learning finds the average of current Q value and previous Q value. For this simple biasness q values cannot increase frequently in Bias Q-Learning. This essentially makes the agent to exploit more in the learning process. However, from the graph below waiting for both of the process is shown In the graphs X-axis and Y-axis are steps and waiting time respectively.

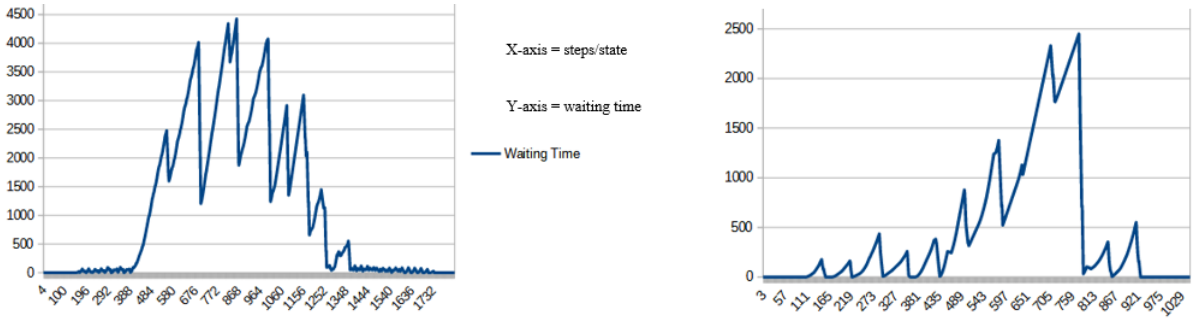


Figure 5.7: Q-Learning and Bias Q-Learning waiting time comparison

Bias- Q Learning has the same graph as SARSA but inside the .xml file the data is slightly different. That means, the values are not as same as SARSA in Bias-Q-Learning. However, in Q-Learning it is visible that the Q-Learning takes 1732 unit to complete an episode of the simulation. On the other side Bias-Q-Learning takes only 1029 unit to complete an episode. Which is much faster than Q-Learning. Furthermore, the waiting time for Q-Learning is 4500 unit and for Bias-Q-Learning waiting time is only 2500 unit. To add more, this waiting time is only for a certain number of steps. Now, from the below graphs we can see the waiting cars amount for both simulations.

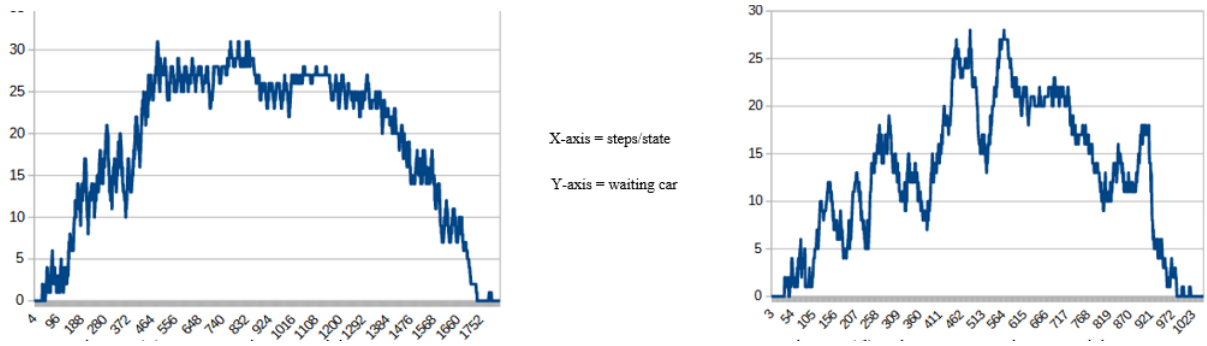


Figure 5.8: Q-Learning and Bias Q-Learning waiting cars comparison

In Q-Learning we can see that the waiting cars number is 30-32 but in Bias-Q-Learning waiting cars number is near 26-27. Which is less than the Q-Learning. So, it shows that Bias-Q-Learning is performing better than the Q-learning.

Chapter 6

Conclusion

The proposed model is an outline of the major problem which is traffic jam in urban areas. The challenges are countless. The thesis work on few parameters of traffic jam but there are more to include in the model and simulation for better outcome. So far, this thesis focuses on implementing Reinforcement Learning, Q-Learning, multi-object RL, Markov Decision Processing and so on. All these are great models and algorithm for such complex problem but we want to go one step farther by introducing quantum computing in this project which will not only increase the accuracy of the training model or agent but also reduce the time and space complexity of computation. Last but not the least, we look forward to implement the project on real life, expecting, this will help to minimize the suffering of people in general.

6.1 Scope and Limitation

As like coins which have two sides, this research has few drawbacks itself with its advantages. Here is the most challenge to work with real data to do principal component analysis. Instead of this lacking this paper analysis with artificial data which is experienced from the SUMO environment through TarCI API. The most common challenge is the unavailability of/difficulties to access real data pays a drawback as an inability to realistic analysis to decision making. As a result, we are determined to work with artificial environment which is taken from open street map . But in our situation it's a little bit difficult to access this organizational existing data without any permission. Also on the other hand it draws attention to funding. Impact of lack of funding's do not purchase any domain to access the external existing data source. So, we are working on artificial data analysis through algorithms which simulate the environment and make decisions. Another challenge is about the confusion of environmental area based responding. In regard to artificial analysis we trained the model on various environments but in the situation of the real world sometimes we can face data inconsistent or poorly defined data through the model. But once we do real world data analysis, this problem management ability is also trained to the model.

6.2 Recommendation and Future Implementation

In future, possible analysis with more efficient nanometer scale by electronic computing, as predicted by Moore's Law [25] and are well performing to coverage to the quantum scale which takes the simulation of traffic signals in the next decade [26]. The quantum feature's success control can lead to quantum algorithms with incredible speed-ups in computation. An unstructured database can possible to search with a universal quantum computer in time which is quadratic in the size of database and that an integer can be factored in time polynomial in the number of its digit, shown by the famous result known as Grover's [27] and Shor's [28] algorithms. With this possibility of more corresponding between traffic lights includes more accuracy. On second note, for further research, Hierarchical methods might be applied for diving the clustered higher level environment into smaller spaces and making easier the optimization problems that use a divide-and-conquer strategy to deal with [29]. Drawing attention in the third point, in future we are planning to simulate the possibility of road accidents as sometimes traffic congestion also occurs as a result of road accidents which is also regulated by traffic signal controllers using existing simulation methods. Now our existing simulation method can regulate traffic signals though artificially of an environment so as this can adapt accidental spots also. Fourth, we are ready to work with real data through our simulation model which weighted our paper with accurate analysis and given the assurity of proper investigational results after dealing with all drawbacks professionally. In future, our ultimate goal is to implement this RL method in real traffic system and tested it with required sensor's capabilities (through loops in the road, cameras, and/or communication with cars) and establish it with the most lively output.

6.3 Q/A SESSION

Question-1: Why q-learning?

Answer: QL is basic but efficient enough to train an agent. Moreover, we choose QL over Deep Q-learning because a) DQL requires more computational power b) our model does not have multiple agents so using DQL would not make much difference.

Question-2: Did any other paper used q-learning before? **Answer:** Most of the papers found during research, used q-learning, as well as deep q-learning. However, the purpose to use Q-learning in this paper is to comparing with other algorithms.

Bibliography

[1] A. A. Haider, "Traffic jam: The ugly side of dhaka's development," *The Daily Star*, 2018.

[2] M. T. Barnamala, "Traffic jam is freezing strong economy and healthy environment: A case study of dhaka city," *IOSR Journal of Economics and Finance*, vol. 6, pp. 36–40, 2015.

[3] E. Van der Pol and F. A. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," *Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016)*, 2016.

[4] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2496–2505.

[5] C. Watkins, "Learning form delayed rewards," *Ph. D. thesis, King's College, University of Cambridge*, 1989.

[6] N. R. Jennings and M. Wooldridge, "Applications of intelligent agents," in *Agent technology*, Springer, 1998, pp. 3–28.

[7] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[8] C. J. C. H. Watkins, "Learning from delayed rewards," 1989.

[9] L. Prashanth and S. Bhatnagar, "Reinforcement learning with function approximation for traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 2, pp. 412–421, 2010.

[10] D. Houli, L. Zhiheng, and Z. Yi, "Multiobjective reinforcement learning for traffic signal control using vehicular ad hoc network," *EURASIP journal on advances in signal processing*, vol. 2010, no. 1, p. 724 035, 2010.

[11] M. Steingrover, R. Schouten, S. Peelen, E. Nijhuis, B. Bakker, *et al.*, "Reinforcement learning of traffic light controllers adapting to traffic congestion.," in *BNAIC*, Citeseer, 2005, pp. 216–223.

[12] N. V. Findler and J. Stapp, "Distributed approach to optimized control of street traffic signals," *Journal of Transportation Engineering*, vol. 118, no. 1, pp. 99–110, 1992.

[13] X. Liang, X. Du, G. Wang, and Z. Han, "A deep reinforcement learning network for traffic light cycle control," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1243–1253, 2019.

[14] R.-H. Hwang, J. F. Kurose, and D. Towsley, "Mdp routing in atm networks using virtual path concept," in *Proceedings of INFOCOM'94 Conference on Computer Communications*, IEEE, 1994, pp. 1509–1517.

[15] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 128–135, 2010.

[16] A. Vidali, L. Crociani, G. Vizzari, and S. Bandini, "A deep reinforcement learning approach to adaptive traffic lights management.," in *WOA*, 2019, pp. 42–50.

[17] S. S. Mousavi, M. Schukat, and E. Howley, "Traffic light control using deep policy-gradient and value-function-based reinforcement learning," *IET Intelligent Transport Systems*, vol. 11, no. 7, pp. 417–423, 2017.

[18] D. Mehta R. K. Reddy, *Smart traffic management system for smart cities using reinforcement learning algorithm.* Mar. 2019. [Online]. Available: https://www.ijrte.org/wp-content/uploads/papers/v7i6s/F02060376S19.pdf.

[19] M. Gregurić, M. Vujić, C. Alexopoulos, and M. Miletić, "Application of deep reinforcement learning in traffic signal control: An overview and impact of open traffic data," *Applied Sciences*, vol. 10, no. 11, p. 4011, 2020.

[20] B. Abdulhai, R. Pringle, and G. J. Karakoulas, "Reinforcement learning for true adaptive traffic signal control," *Journal of Transportation Engineering*, vol. 129, no. 3, pp. 278–285, 2003.

[21] E. Walraven, M. T. Spaan, and B. Bakker, "Traffic flow optimization: A reinforcement learning approach," *Engineering Applications of Artificial Intelligence*, vol. 52, pp. 203–212, 2016.

[22] N. Kheterpal, K. Parvate, C. Wu, A. Kreidieh, E. Vinitsky, and A. Bayen, "Flow: Deep reinforcement learning for control in sumo," *EPiC Series in Engineering*, vol. 2, pp. 134–151, 2018.

[23] N. S. Jadhao and A. S. Jadhao, "Traffic signal control using reinforcement learning," in *2014 Fourth International Conference on Communication Systems and Network Technologies*, IEEE, 2014, pp. 1130–1135.

[24] C. E. Arruda, P. F. Moraes, N. Agoulmine, and J. S. Martins, "Enhanced pub/sub communications for massive iot traffic with sarsa reinforcement learning," *arXiv preprint arXiv:2101.00687*, 2021.

[25] G. Moore, "Moore's law," *Electronics Magazine*, vol. 38, no. 8, p. 114, 1965.

[26] H. Hussain, M. B. Javaid, F. S. Khan, A. Dalal, and A. Khalique, "Optimal control of traffic signals using quantum annealing," *Quantum Information Processing*, vol. 19, no. 9, pp. 1–18, 2020.

[27] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219.

[28] S. Goldwasser, *35th Annual Symposium on Foundations of Computer Science: Proceedings*. IEEE Computer Society Press, 1994.

[29] H. Rabinowitz, *Smart traffic management system for smart cities using reinforcement learning algorithm.* Jul. 2018. [Online]. Available: https://qz.com/1323987/quantum-computing-could-put-a-stop-to-traffic-jams/.