Adam Rizvi Thahir

A Master's Thesis

AGU 2022

GRAPH THEORY BASED TRAFFIC LIGHT MANAGEMENT

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Adam Rizvi Thahir
June 2022

GRAPH THEORY BASED TRAFFIC LIGHT MANAGEMENT

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Adam Rizvi Thahir
June 2022

SCIENTIFIC ETHICS COMPLIANCE

I hereby declare that all information in this document has been obtained in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Name-Surname: Adam Rizvi Thahir
Signature :

REGULATORY COMPLIANCE

M.Sc. thesis titled Graph Theory Based Traffic Light Mangement has been prepared in accordance with the Thesis Writing Guidelines of the Abdullah Gül University, Graduate School of Engineering & Science.

Prepared By	Co-Advisor	Advisor
Adam Rizvi Thahir	Asst. Prof. Dr. Mustafa Coşkun	Prof. Dr. Vehbi Çağrı Güngör

Head of the Electrical and Computer Engineering Program

Assoc. Prof. Dr. Kutay İçöz

ACCEPTANCE AND APPROVAL

M.Sc. thesis titled Graph Theory Based Traffic Light Management and prepared by Adam Rizvi Thahir has been accepted by the jury in the Electrical and Computer Engineering Graduate Program at Abdullah Gül University, Graduate School of Engineering & Science.

30 /05/ 2022

(Thesis Defense Exam Date)

JURY:

Advisor : Prof. Dr. Vehbi Çağrı Güngör

Member : Asst. Prof. Burcu Güngör

Member : Asst. Prof. Fehim Köylü

APPROVAL:

The acceptance of this M.Sc. thesis has been approved by the decision of the Abdullah Gül University, Graduate School of Engineering & Science, Executive Board dated ….. /….. / …….. and numbered ……..…… .

…….. /…….. / ……..

(Date)

Graduate School Dean
Prof. Dr. İrfan Alan

ABSTRACT

GRAPH THEORY BASED TRAFFIC LIGHT MANAGEMENT

Adam Rizvi Thahir
MSc. in Electrical and Computer Engineering
Advisor: Prof. Dr. Vehbi Çağrı Güngör
Co-advisor: Asst. Prof. Dr. Mustafa Coşkun

June 2022

Traffic congestion and delays caused in traffic light intersections can adversely affect countries in terms of money, time, and air pollution. With the advancement of computational power as well as artificial intelligent algorithms, researchers seek novel and optimized solutions to the traffic congestion problem. Most modern traffic light systems use manually designed traffic phase plans at intersections, and although this has proven to be relatively sufficient for today's traffic management systems, implementing a smarter traffic phase selection system is deemed to be more effective. Traditional approaches rely heavily on traffic history (static information), whereas Reinforcement Learning (RL) algorithms, which offer an "adoptable"/dynamic traffic management system, are gaining increased research interest. Despite the usefulness of these RL based deep learning techniques, they inherently suffer from training time to apply them in real-world traffic management systems. This study aims to alleviate the training time problem of deep learning-based techniques, The research brings forth a novel graph-based approach that is able to use known occupancies of roads to predict which other roads in a given network would become congested in the future. Based on the predictions obtained, we are able to dynamically set traffic light times in all intersections within a connected network, starting from roads with known occupancies, and moving along connected roads that are anticipated to be congested. Predications are done using edge-based semi-supervised graph algorithms. Conducted simulations show that our approach can yield comparable average wait time to that of deep-learning based approach in minutes, compared to the much longer training time required by the deep-learning models.

Keywords: Deep Learning, Reinforcement Learning, Traffic Flow, Congestion

ÖZET

GEAFİK TEORİSİ TABANLI TRAFİK IÇIĞI YÖNTEMİ

Adam Rizvi Thahir
Elektrik ve Bilgisayar Mühendisliği Anabilim Dalı Yüksek Lisans
Tez Yöneticisi: Prof. Dr. Vehbi Çağrı Güngör
Eş-Danışman: Dr. Mustafa Coşkun

Haziran 2022

Trafik ışıklı kavşaklarda meydana gelen trafik sıkışıklığı ve gecikmeler ülkeleri para, zaman ve hava kirliliği açısından olumsuz etkileyebilmektedir. Yapay zeka algoritmalarının yanı sıra hesaplama gücünün ilerlemesiyle birlikte, araştırmacılar trafik sıkışıklığı sorununa yeni ve optimize edilmiş bir çözümler aramaktadırlar. Çoğu modern kavşaklarda, manuel olarak tasarlanmış trafik faz planı kullanılmaktadır. Bunun günümüz trafik yönetim sistemleri için nispeten yeterli olduğu kanıtlanmış olsa da, akıllı bir trafik faz planı uygulanmasının daha etkili olduğu düşünülmektedir. Geleneksel yaklaşımlar büyük ölçüde geçmiş trafik verisine (statik bilgi) dayanırken, dinamik/adaptif bir trafik yönetim sistemi sunan Pekiştirmeli Öğrenme (RL) algoritmaları giderek daha fazla araştırmacıların ilgisini kazanmaktadır. Bu RL tabanlı derin öğrenme teknikleri kullanışlı olmasına rağmen eğitim sürelerinden dolayı gerçek hayattaki trafik yönetim sistemlerine uygulanması zordur. Bu çalışma, derin öğrenme tabanlı yöntemlerin eğitim süresi problemini çözmeyi amaçlamaktadır. Araştırma, belirli bir ağdaki diğer yolların gelecekte hangi durumda tıkanacağını tahmin etmek için bilinen yol doluluk durumlarından yararlanmayı sağlayan, yeni bir grafik tabanlı yaklaşım getirmektedir. Elde edilen tahminlere dayanarak, trafik sıkışıklığı bilinen bir yoldan başlayarak bir sonraki tıkanması beklenen bağlantılı yolları içeren ağdaki tüm kavşakların trafik ışık sürelerini dinamik olarak ayarlanabilmekteyiz. Tahminlemeler, kenar tabanlı yarı denetimli grafik algoritmaları kullanılarak yapılmaktadır. Yürütülen simülasyonlar, yaklaşımımızın derin öğrenme modellerinin gerektirdiği çok daha uzun eğitim süresiyle karşılaştırıldığında, birkaç dakika içinde derin öğrenme tabanlı yaklaşımla karşılaştırılabilir ortalama bekleme süresi sağlayabileceğini göstermektedir.

Anahtar kelimeler: Derin Öğrenme, Pekiştirmeli Öğrenme, Trafik Akışı, Tıkanıklık

Acknowledgements

First and foremost, I would like to express my most sincere thanks to my academic advisor Prof. Dr. Vehbi Çağrı Güngör for guiding and supporting me throughout my time as a master's student at Abdullah Gul University. It has been a privilege to collaborate with him and learn from his many years of experience.

I would also like to thank my academic co-advisor, Dr. Mustafa Çoşkun for all his profound knowledge and insight regarding Graph Theory, Semi-Supervised learning, and other topics vital to the research conducted.

Furthermore, I would also like to thank other members of the Computer Engineering department who have all paved my way leading up to this point. Dr. Zafer Aydın for initially getting me involved and interested in Machine Learning, Data Science, and other relevant topics. Dr. Gülay Yalçın for piquing my interest in theoretical computing topics, Dr. Burcu Bakır-Güngör for being a supporting pillar in other courses and academic writing, and finally, Dr. M. Şükrü Kuran who first recognized the potential within me and was a driving force behind my motivation to begin and continue along this path of my academic life.

My mentor, Süheyl Töken who has enlightened me with his many years of industrial and technical knowledge and provided me the opportunity to collaborate with him to develop real world applications relating to traffic lights, an opportunity from which I was able to gain firsthand experience on a topic relating to my research.

I would also like to thank a fellow master's student and colleague, Sultan Kübra Kılıç, whom I have worked with closely with over the past few years. They have been a great support to my research, having real life knowledge on traffic light management, and great practical knowledge on a variety of related topics.

My friends, I have made over the years in Turkey; Muhammad Fuad Farooqi, for helping me always keep my academic goals in focus. Süleyman Çiçek and Hussam Amoody, for all their support throughout the many years we have known each other.

TABLE OF CONTENTS

1. **INTRODUCTION** **1**

1.1 TRAFFIC LIGHT INTERSECTIONS 2

1.2 VEHICLE OCCUPANCIES AND QUEUES 3

1.3 TRAFFIC FLOW 3

2. **SIMULATION TOOL** **4**

2.1 NETEDIT 4

2.2 VEHICLE ROUTING 5

2.3 SUMO 6

2.4 TRACI 6

3. **TRAFFIC MANAGEMENT SYSTEMS** **7**

3.1 DATA COLLECTION 7

3.2 DATA PROCESSING/CLEANING 7

3.3 TRAFFIC PHASE CALCULATION/OPTIMIZATION 8

3.3.1 Conventional Traffic Control Systems. *8*

3.3.2 Reinforcement Learning for traffic control systems. *8*

3.3.3 Traffic Flow Prediction Based Traffic Control Systems *10*

4. **METHODOLOGY** **12**

4.1 PROBLEM DEFINITION 12

4.2 STATE-OF-THE-ART APPROACHES 12

4.2.1 Occupancy based algorithm *12*

4.2.2 Scoring Algorithm *13*

4.2.3 Reinforcement Learning Algorithms *14*

4.2.3.1 Proximal Policy Optimization (PPO) 14

4.2.3.2 Advantage Actor Critic (A2C) 14

4.3 PROPOSED: GRAPH-SEMI SUPERVISED LEARNING BASED APPROACH 15

4.3.1 Graph based Semi-Supervised Learning for vertices *16*

4.3.2 Graph based Semi-Supervised Learning for edge flows *17*

5. **RESULTS** **19**

5.1 PREDICTION PERFORMANCE RESULTS 21
5.2 SIMULATION PARAMETERS 22
5.2.1 Number of steps *22*
5.2.2 Number of simulations *22*
5.2.3 Traffic Phases *22*
5.3 PERFORMANCE EVALUATION 22
6. **CONCLUSIONS AND FUTURE PROSPECTS 29**
6.1 SOCIETAL IMPACTS AND CONTRIBUTION TO GLOBAL SUSTAINABILITY 29
6.2 CONCLUSIONS 30
6.3 FUTURE PROSPECTS 31
7. **APPENDIX 36**
Appendix A: Python Code used to run a SUMO simulation using TraCI 36

LIST OF FIGURES

Figure 1.1 Four-way traffic light intersection 2
Figure 2.1 Network design for a simulation environment with 20 traffic light intersections. 5
Figure 3.1 Visualization on how a generic Reinforcement Learning algorithm would work on a traffic light environment 9
Figure 4.1 Network design for simulation environment with 17 traffic light intersections 16
Figure 5.1 Flowchart representation breaking down the steps on how the proposed 20
Figure 5.2 Graph based SSL for synthetic flows from the simulation on 5, 17 and 20 intersection models. 21
Figure 5.3 Average waiting time for a single vehicle in the network 23
Figure 5.4 Average total waiting time for all vehicles in the network 24
Figure 5.5 Average waiting time for all vehicles in an intersection 25
Figure 5.6 Average maximum maximum waiting time for a single vehicle 26
Figure 5.7 Average wait time per vehicle per simulation step – 17 Intersections 27
Figure 5.8 Average wait time per simulation step - 20 intersections 27
Figure 5.9 Average running times for each simulation 28

LIST OF TABLES

Table 3.1 Comparison table between the proposed approach against other state-of-the-art approaches 11

LIST OF ABBREVIATIONS

A2C	Advantage Actor Critic
AI	Artificial Intelligence
NP	Non-deterministic Polynomial-time
PPO	Proximal Policy Optimization
RL	Reinforcement Learning
RRQR	Rank-revealing QR
SUMO	Simulation of Urban MObility
TL	Traffic Light
TMS	Traffic Management Systems
TraCI	Traffic Control Interface

To my parents,

M. Rizvi Thahir and Zeenathul Fathima

Chapter 1

Introduction

With the advancement of modern cities, growing population, and the increase in the number of vehicles owned by people within a city, traffic congestion has become a growing problem over the years. Along with traffic congestion comes time consumption, air pollution, and economic burdens. As per a study conducted by Inrix in 2019, an average driver in the United States has lost 99 hours in traffic congestion, with an estimated cost of $1,337 [1]. Naturally, in larger cities, the associated time consumption and cost are much more severe, such as in Boston, MA, where the number of hours lost was recorded to be up to 149, and the estimated cost per person was $2,205 [1]. Thus, utilizing an "adaptable" traffic management system has been one of the central research interests.

Many modern cities today operate their traffic systems using predefined fixed times which are often based on manually hand-crafted traffic rules by people observing traffic in real time [2]–[4]. However, predefined rule-based traffic management systems are vulnerable to changes that are inevitable due to expanding cities and increase in the number of vehicles on the road, as such, these predefined systems are often deemed to be limited [5]. Inspired by the fascinating developments on Artificial Intelligence (AI) algorithms, more recent research attempts based on AI approaches aim at solving traffic light configurations for intersections. Earlier studies, such as those conducted in [6]–[8] propose that reinforcement learning algorithms provide a more efficient traffic management model. This study proposes the use of graph based semi-supervised learning for edge label prediction, which aims to alleviate the shortcomings of RL based systems while also providing an efficient traffic management model.

In order to assess the performance of our proposed approach against state-of-the-art deep approaches as well as heuristic-based approaches and fixed time approaches, we conduct experiments on a microscopic traffic simulation tool: Simulation of Urban MObility (SUMO) [9]. This tool is further discussed in Section 2.

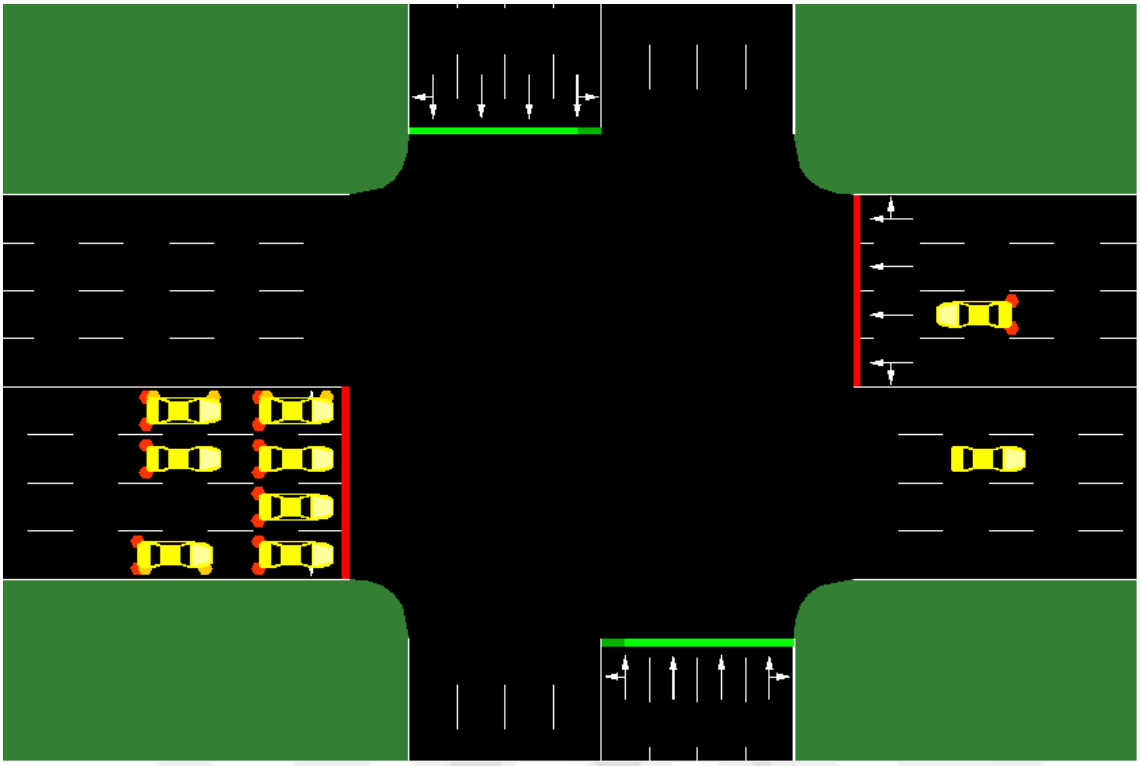


Figure 1.1 Four-way traffic light intersection

1.1 Traffic Light Intersections

Traffic light intersections are interest points within a road network in which multiple roads (edges) intersect with each other and are controlled by a traffic light system. These systems allow vehicles to pass at specific lanes at a given time, causing vehicles at other intersections to stay at an idle stage until they are allowed their turn.

A standard four-way intersection is depicted in Figure 1.1; As per the displayed network model, at the given timestamp. From this image, it is observed that the north and south roadways have been given the green signal, allowing vehicles originating from either one of these roads to pass through, whereas, vehicles originating from the East and West roadways are required to wait their turn.

For the scope of this project, we model multiple four-way intersections as seen above, additionally, we also model several three-way intersections in an attempt to create a robust traffic network with minimal bias. Model types and network designs are further discussed in Chapters 2 and 5.

1.2 Vehicle Occupancies and Queues

Vehicle occupancies are defined as the number of vehicles on each edge. A given edge may have a different number of lanes within the edge, the number of lanes within an edge would affect the total queue size of the edge, but not the vehicle occupancy.

Queue sizes are defined as the length of vehicles waiting in a given edge. The number of lanes within an edge would greatly affect the queue sizes of that edge.

The size of the queue within an edge would cause delays for the vehicles on that edge, due to latencies caused by the initial movement of a vehicle from a stopping state.

As per the image shown in Figure 1.1, the maximum vehicle occupancy observed in the network is 7, whereas the maximum queue size observed in the network is 2. Despite both values obtained from the same edge, the values are expected to be different due to the edge having multiple lanes. In the case of an edge having exactly one lane, the vehicle occupancy and queue size would be the same.

1.3 Traffic Flow

Traffic flow is defined as the continuous flow movement of vehicles within a network layout. Traffic flow heavily depends on a given network layout as well as vehicle route demands.

Traffic flow from one intersection to another intersection would vary, depending on the movement of each vehicle within the flow at a given time. A smooth traffic flow is described as the continuous movement of traffic between intersections. These flows may be disrupted by driver behavior, pedestrian movement, and other outside factors as well. Additionally, following the idle stopped state of vehicles at a red light, delays caused by vehicles ahead may also negatively affect traffic flow.

Chapter 2

Simulation Tool

Simulation environments were designed using a series of tools provided by the microscopic and continuous multi-modal traffic simulation package; SUMO [10].

SUMO is an open-source traffic simulation suite that allows us to design, model and simulate traffic networks [9]. During a simulation, SUMO provides necessary tools to interact with the running simulation, as well as extract required data from within the network, such as currently running traffic phase information and vehicle occupancies for a given road.

For the scope of this study, a variety of tools provided by SUMO were used. The image shown in Figure 1.1 was taken from a close-up view of a running simulation on a modeled intersection.

2.1 NetEdit

NetEdit is a tool provided within the SUMO suite [11]. This tool is used to create and model various traffic networks. A traffic network containing 20 different intersections constructed using NetEdit is represented in Figure 2.1.

Networks are designed by placing nodes (intersections) on a blank canvas, and then connecting links between two nodes (roads). Properties for a road can be defined using the ‘Inspect’ panel on the left side of the interface.

We then generate a net file output which would be used with other tools provided by SUMO suite.

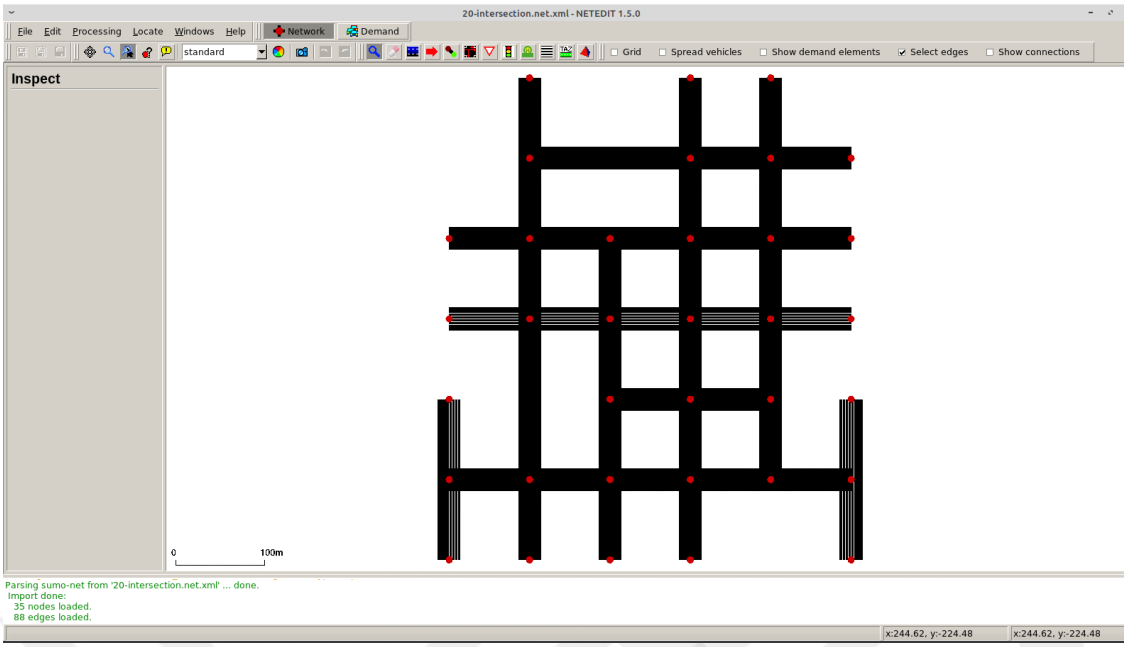


Figure 2.1 Network design for a simulation environment with 20 traffic light intersections.

Having a network layout allows us to proceed with constructing vehicle routes for the simulation.

2.2 Vehicle Routing

In order to construct a reliable network, randomly generated vehicle routes are vital. Our study uses the scripts RandomTrips [12] and DuaRouter [13] to generate these routes. RandomTrips is used to generate random trips vehicles within the network can take, this defines valid vehicle movement between nodes within the network.

Trips generated by the script are done at random by choosing random source and destination edges within the network layout file. A random seed can be provided as an input parameter upon running the script. Traffic volume and arrival rates can also be adjusted as per our needs, for the scope of this project we leave these settings to default, for all simulations in order to obtain unbiased results. The trips generated by are stored as a separate file which is then used as an input for DuaRouter.

Using the file generated by RandomTrips, DuaRouter creates vehicle demand for the simulation. The vehicle demand is created for the total simulation time provided; trips obtained from the RandomTrips output are validated such that each vehicle route approved by DuaRouter is a valid path in which a vehicle can move from source to destination.

Having the output from DuaRouter, and DuaRouter, we are able to start our simulations using the SUMO tool, which has the option to run with and without a GUI.

2.3 SUMO

SUMO is the tool within the suite that is responsible for the simulation. This tool allows us to simulate a defined scenario provided by a variety of configuration files, including a network file, often generated by NetEdit, and a vehicle demand file often generated by DuaRouter.

The tool comes in two forms, SUMO, and SUMO-GUI. As the name implies, SUMO-GUI also provides a Graphical User Interface allowing us to view our simulation as the simulation progresses. Using the GUI is a useful step to provide insight on created network models and simulation scenarios, however running a series of simulations continuously may seem slow with the GUI; For this reason, we use SUMO without a GUI to conduct our simulations once simulation scenarios are validated.

In this study, we use the Python programming language to start and control our simulations. Sample code to start a given scenario is provided in the appendix section.

2.4 TraCI

TraCI – short for "**Tra**ffic **C**ontrol **I**nterface," is another tool provided by the SUMO suite. This tool gives us access to monitor, extract data, and manipulate simulations running on SUMO. TraCI is able to access SUMO via a client/server architecture, where SUMO acts as the server listening in for incoming connections, and TraCI is the client.

TraCI allows us to observe changes within a network and interact with the simulation accordingly. Some features include monitoring vehicle occupancy within edges in the network and traffic light phase for any intersection at the current simulation time.

Using TraCI, we are also able to extract information for each vehicle currently inside the simulation. This information is used to model traffic flow from one intersection onto another intersection.

Chapter 3

Traffic Management Systems

The work executed by a traffic management system can be broadly categorized into three distinct categories.

i. Data collection and communication
ii. Data processing/cleaning
iii. Traffic Phase calculation/optimization

3.1 Data Collection

The very first step of the system involves obtaining information from the network. Most common data collection methods used in Traffic Light systems, involves traffic phase information, number of edges, number of lanes within each lane, vehicles within each lane and total number of vehicles within a network. However, the exact information obtained would vary depending on the system's requirements.

The data collection process itself may also vary based on a different set of needs. Common data collection techniques include inductive loops [14], [15] and image processing [16], [17].

In order to process the data, the collected data must be passed on to other sections of the system. Some common approaches are to utilize different levels of network-level protocols to achieve this need. These approaches have been further studied in [18]–[22].

3.2 Data processing/cleaning

In order the ensure that the collected data can be used by the system in a streamlined manner, it is important for all the incoming data to be further processed and/or cleaned before the system stores the data. This step is often entwined with the communication process of the system in order to regulate incoming data. The definite process of data cleaning heavily relies on how the data is to be used by the TMS. Some studies discussing data communication also propose viable forms of data processing.

3.3 Traffic Phase Calculation/Optimization

The final step of the TMS aims at solving traffic congestion problem by taking the number of vehicles and queue sized into account whilst calculating new phase times for each phase within the intersection.

Modern approaches to calculating these phase times can be classified into the following types.

i. Conventional Traffic Control Systems
ii. Reinforcement Learning for Traffic Control Systems
iii. Traffic Flow predication-based traffic control systems.

3.3.1 Conventional Traffic Control Systems.

Conventional traffic control systems are often used in earlier traffic system models. In such systems, traffic flows are observed manually, and traffic phase times are then set according to rules and patterns from these observations. These rules often include a change in priority based on time of day, construction interference, and other factors that would affect traffic flow. Such systems have been studied in [2], [3], [23].

Another approach is conducted by using real-time vehicle data for a given duration, and an optimal phase time is calculated for each of the phases. These systems have been further studied in [4], [24]. The calculated optimal times would then be set to last within the intersection until they would be updated again at some point in the future. Despite offering some merit, over time, the calculated optimal times may become redundant due to changes in vehicle behavior, and the system not having sufficient insight into any future data.

3.3.2 Reinforcement Learning for traffic control systems.

Traffic congestion is a dynamic, time-reliant problem. With the advancement of AI algorithms, modern reinforcement learning, and deep learning approaches have been employed to dynamically adjust traffic lights by learning traffic behaviors.

The general flow for the usage of an RL model in the scope of traffic lights can be classified into three sections

i. Environment: Composed of traffic light phases and traffic conditions, including traffic congestion and traffic outward flow.

ii. State: A feature representation of the environment, often a grid representation of the phase times within the intersection.

iii. Agent: A model which is able to make informed changes onto the environment based on the information obtained from the State.

Following the decision making of the agent, the environment would return a reward value back onto to the state which would be used as an additional input parameter to the agent before it makes its next decision. This reward value would inform the agent how the environment reacted to its previous decision, allowing a better-informed following decision to be taken. This flow is depicted in Figure 3.1

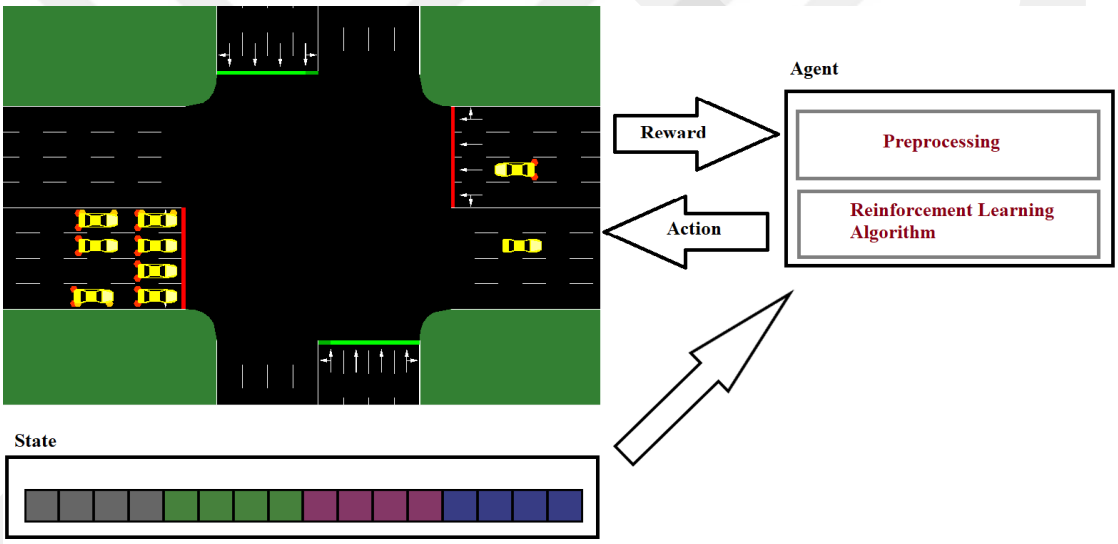


Figure 3.1 Visualization on how a generic Reinforcement Learning algorithm would work on a traffic light environment

Other approaches to model traffic networks into a RL model may use different definitions for their Environment, State and Agent sets of the model. The studies conducted in [6], [8], [25]–[27] model their network having the number of vehicles queued as their state space, similarly studies [28], [29] use traffic flow obtained by placing sensors at the edge of the traffic light. Certain studies also introduce different action spaces, such as having all possible signal phases as their action space [28], [30] or only the green signal phase within the action space [26], [29]. Reward functions for [28], [30] is set based on the change in delay for the vehicles in the network, whereas [26], [29] define their reward function based on the change in the number of queued vehicles.

Moreover, other studies such as [5] have used traffic flow and traffic delays as their reward function, implementing Deep Q-learning. Finally, a series of other RL algorithms such as Asynchronous method for deep reinforcement learning and Proximal Policy Optimization algorithms were also conducted in [31], [32] by Stable Baselines.

Comparing the conventional methods against the reinforcement learning and deep learning approaches, it is observed that reinforcement and deep learning approaches are a more effective solutions to traffic congestion as seen in [5], [33], [34].

However, despite the positive results, the time taken for the training for the neural network parameters is computationally expensive and often costs more as the number of intersections increases within the network.

3.3.3 Traffic Flow Prediction Based Traffic Control Systems

Traffic Flow Prediction Based Traffic Control Systems is the proposed system in this study. It is an alternative and efficient algorithm by approaching the TMS as a flow prediction problem. This approach renders predicting the outward flow of vehicles from a given intersection, onto connected intersections, and enables the TMS system to identify vehicle congestion that is likely to occur at the following intersections. More specifically, in a multi-intersection setting, our approach is able to provide information on traffic flow behavior, by pin-pointing intersections that would become congested within a short period of time, based on semi-supervised active learning on edges [35].

The main idea of our proposed approach is different from other deep learning and traditional methods as the flow of traffic can be predicted beforehand, allowing intersections with predicted congestions to set traffic light phase times according to the level of predicted congestion.

Furthermore, this approach treats intersections within a network as a node. This graph-based approach is able to efficiently process large amounts of intersections, allowing seamless implementation and adaptability to scale at higher levels for systems designed for larger cities.

A comparison table of different state-of-the-art approaches is depicted in the below table.

Table 3.1 Comparison table between the proposed approach against other state-of-the-art approaches

	Simulation	Predict Flows	Real-Time data	Historical data	Multi-intersection
INTELLILIGHT [34]	YES	NO	YES	YES	NO
Francois Dion et al [36]	NO	NO	NO	YES	NO
Alan J Miller [2]	NO	NO	NO	YES	NO
Brian L, Smith [37]	NO	YES	NO	YES	NO
FRAP[38]	YES	NO	YES	NO	YES
CoLight [33]	YES	NO	YES	NO	YES
OUR APPROACH	YES	YES	YES	YES	YES

Chapter 4

Methodology

4.1 Problem Definition

In a traffic management system (TMS), our aim is to find an optimum traffic light configuration in a multi-intersection network, such that traffic congestion can be prevented as much as possible.

We define the TMS environment as ε, which consists of multiple intersections. Each intersection within ε is set to have an "intelligent" traffic light agent TL_i. Each intersection agent, TL_i has three default phase settings.

i. Red: Vehicles **cannot** pass through the intersection.
ii. Green: Vehicles **can** pass through the intersection.
iii. Yellow: Phase changes from one phase to another.

Default Yellow time is set to be 7s.

4.2 State-of-the-art approaches

4.2.1 Occupancy based algorithm

The occupancy-based algorithm is a heuristic algorithm that tries to solve the TMS problem by relying on the basic idea of using predefined historical traffic data in order to determine the occupancies of the roads. More specifically, in this algorithm the occupancies of all roads with incoming traffic onto the intersection are used to prioritize traffic phases, i.e., if a road obtains a higher occupancy, then the green phase time given for that road is incremented up to a calculated threshold. The prioritization formula is defined as follows:

$$D = T_{max} * \left(N * \left(\frac{W_i}{W_T} \right) \right) \quad \textbf{(4.1)}$$

where D is the duration, which is also set to a certain upper limit to prevent blocking on other roads, T_{max} is the total time, N is the number of incoming roads, W_i is the occupancy value for the incoming road *i*, which is heuristically set to a certain value by observing historical traffic data, and W_T is the sum of all the road occupancies coming into the intersection.

Despite the effectiveness and simpleness of the occupancy algorithm, this algorithm sets its parameters based on historical data which is prone to change, i.e., some roads may have had higher occupancy values in the past, however, their occupancy values may decrease in the future. As a result, this algorithm introduces a certain bias towards the roads with historically higher occupancy rates.

4.2.2 Scoring Algorithm

The scoring algorithm is another heuristic algorithm that is proposed by Sébastien Faye et al [39]. The basic premise behind the scoring algorithm is to score each incoming road in an intersection and assign the priorities based on the assigned scores. Formally, the score of each incoming road is defined as:

$$LS = \left(\frac{N^{(s,d)}}{\sum_{\{a,b\} \in D} N^{(a,b)}}\right) + \beta \cdot \left(\frac{T_F^{(s,d)}}{\sigma_{\{a,b\} \epsilon D}\, T_F^{(a,b)}}\right) \quad \textbf{(4.2)}$$

where α and β are user-defined weights that are used to optimize average vehicle time and starvation, we use default values as 1 for both, (s, d) is a possible movement from source direction s to destination direction d, both of which are within a set of all possible directions D. $N^{(s,d)}$ is the weighted sum of the number of vehicles present on the coming lanes that compose movement, and $T_F^{(s,d)}$ is the time in seconds since the incoming road had a green phase. In the case that no vehicles are present on any of the roads, LS would be set to 0.

Finally, once the scores for all edges are obtained, the system would calculate priority duration using a variant of the equation seen in Eq 4.1. Replacing occupancy with the calculated local score redefines the formula as:

$$D = T_{max} * \left(N * \left(\frac{S_i}{S_T}\right)\right) \quad \textbf{(4.3)}$$

where D is the duration, T_{max} is the total time, N is the number of incoming roads. S_i and S_T represent the local score for the incoming road *i* and the sum of all scored, respectively.

4.2.3 Reinforcement Learning Algorithms

Using the A2C and PPO algorithms from the Stable Baselines [40] library RL environments were designed as per the environment definition provided in sub-section 3.3.2 Reinforcement Learning for traffic control systems.

4.2.3.1 Proximal Policy Optimization (PPO) [40]

PPO is a policy gradient method that is able to simplify the Trust Region Policy Optimization (TRPO) algorithm by using a clipped surrogate objective. The formula used is defined as:

$$J^{TRPO}(\theta) = \mathbb{E}\left[r(\theta)\hat{\mathrm{A}}_{\theta old}(s,a)\right] \quad \textbf{(4.4)}$$

$$J^{CLIP'}(\theta) = \mathbb{E}\left[J^{CLIP}(\theta) - c_1(V_\theta(s) - V_{target})^2 + c_2 H(s,\pi_\theta(.))\right] \quad \textbf{(4.5)}$$

As per the equation defined in Eq 4.4, maximizing TRPO has the possibility of causing instability due to frequent updates to its parameters. As such, PPO simplifies this approach by using a clipped surrogate objective, forcing $\mathrm{r}(\theta)$ to fit into a smaller interval $[1-\epsilon, 1+\epsilon]$ using the function clip $(\mathrm{r}(\theta), 1-\epsilon, 1+\epsilon)$, as seen in the equation defined in Eq 4.5.

4.2.3.2 Advantage Actor Critic (A2C) [40]

The A2C approach deals with two distinguishable networks: The 'actor' and the 'critic'. The actor network is trained to update the policy according to the value function learned by the critic network. The critic network estimates the value function.

The function A2C uses to calculate the advantage for taking a specific action can be defined as follows:

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s) \quad \textbf{(4.6)}$$

The reward is then calculated based on the difference of waiting time for the vehicles in the network. This function is defined as follows:

$$R = \begin{cases} -1 * \left(\sum T + \sum V\right) & N == 0 \\ -1 * \left(\frac{(\sum T + \sum V)}{N}\right) & N >= 1 \end{cases} \quad \textbf{(4.7)}$$

Where R is the calculated reward, T is an array of waiting times for the different vehicles, V is the number of new vehicles compared to the previous step iteration, N is the number of vehicles that moved onto a different road.

4.3 Proposed: Graph-Semi Supervised Learning Based Approach

The idea behind this approach is to solve the TMS problem by representing the TMS problem as a graph problem and predicting the edge flow on this graph. More specifically, in our graph-based modeling of the TMS problem, each intersection within the network is represented as a node in a graph, and the roads connecting two intersections are treated as edges connecting the nodes within the graph. If we consider the 17-intersection network design shown in Figure 4.1, a total number of 29 nodes are observed in the network. The additional 12 nodes are endpoints of the traffic network from which vehicle demand enters and exits the network. Thus, our graph-based representation for this network consists of a graph with a total of 29 nodes, and edges connecting nodes would relate to the true road connections visible on the network; Any connection between two distinct nodes within the network is recorded as a valid road connection.

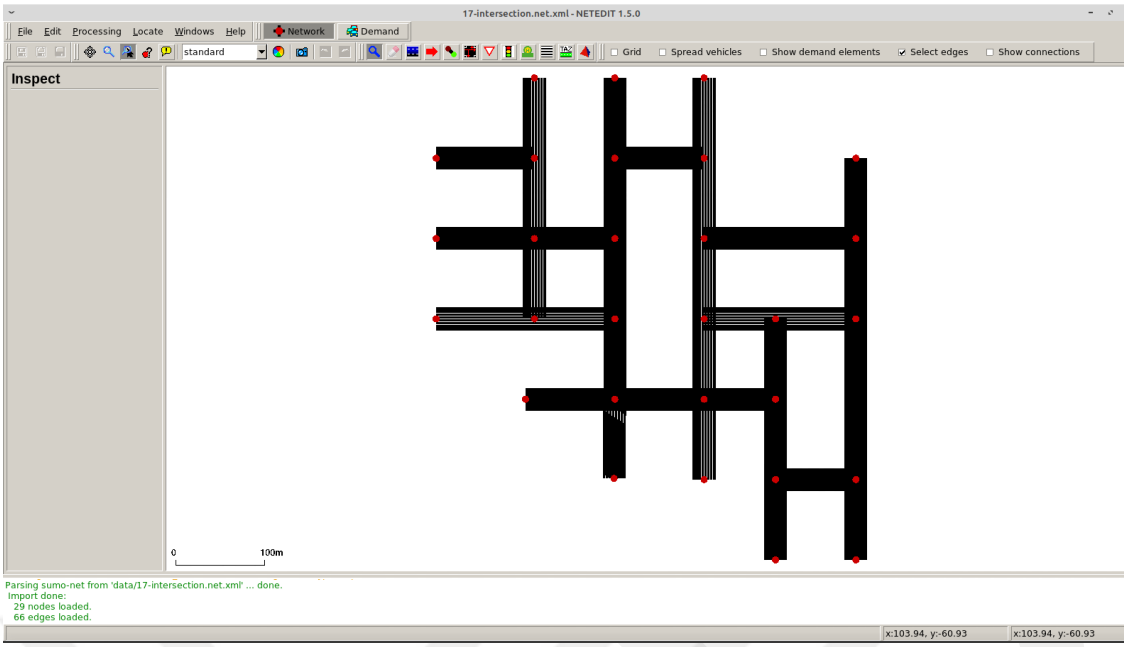


Figure 4.1 Network design for simulation environment with 17 traffic light intersections

Based on this graph model, we use the methodology proposed in [35], edge-based semi-supervised learning. As a result, we are able to predict the outward flow from one intersection on to another. The predictions are then normalized and read from the perspective of the connecting intersections; this is now read as inward traffic flow into the intersection. This now allows us to optimize phases within the intersection based on the predicted total inward vehicle flow coming from other intersections.

Mathematically, we have a set of vertices (traffic intersections and exit-points) V, a set of edges (roads) that connect the vertices ε, and a labeled set of edge flows; known traffic flows obtained from the network, ε^{L}. The goal of the algorithm is to predict the unlabeled edge flows ε^{U}. Here it is important to note that our aim is to conduct edge-based semi-supervised learning, rather than the well-known node-based semi-supervised learning method.

4.3.1 Graph based Semi-Supervised Learning for vertices

In this approach, a graph is constructed with nodes and edges, where nodes are specified by labeled V^{L} and unlabeled samples V^{U}. Edges are to be based on the similarities among the samples V. The goal of this algorithm would be to assign the labels for the unlabeled samples V^{U} based on the known existing data, such that the assigned

labels vary smoothly across neighboring nodes. The notion of smoothness is defined by the following log function:

$$\|B^T y\|^2 = \sum_{(i,j) \in \varepsilon} (y_i - y_j)^2 \quad \textbf{(4.8)}$$

where y represents the vector containing vertex labels, and $B \in R^{nxm}$ represents the incidence matrix of the network. The loss function can be written as $\|B^T y\|^2 = y^2 Ly$ in terms of the graph Laplacian $L = BB^T$.

Based on this, the labels for the unknown nodes can be found by minimizing the quadratic form $y^T Ly$ with respect to y while keeping the set of labeled vertices fixed.

4.3.2 Graph based Semi-Supervised Learning for edge flows

Using this approach, we obtain a graph based on traffic intersections as a set of vertices V, and have the edges connecting two or more vertices as labeled or unlabeled sets ε^L and ε^U.

Following the graph model, we represent the edge flows within the networks as the vector f. If we are to account only for the netflow along and edge, we obtain $f_r > 0$ when the flow orientation of the edge along with its reference orientation and $f_r < 0$ in all other cases. The true edge flow in the network is denoted $\hat{f}$. The divergence of a vertex can be found by calculating the sum of outgoing flows minus the incoming flows at that vertex. This formula is defined as follows:

$$(Bf)_i = \sum_{\varepsilon_r \in \varepsilon : \varepsilon_r \equiv (i,j), i < j} f_r - \sum_{\varepsilon_r \in \varepsilon : \varepsilon_r \equiv (j,i), j < i} f_r \quad \textbf{(4.9)}$$

A loss function for edge flows is also defined in order to enforce a notion of flow conservation. Having the edge Laplacian matrix defined as $L_e = B^T B$, the loss function is defined as:

$$\|Bf\|^2 = f^2 B^T B f = f^T L_e f \quad \textbf{(4.10)}$$

Studies in [41], [42] implement Active supervised learning in order to obtain a set of labeled edges that were most helpful in determining overall edge flows within their graph model. A similar approach is used with regard to our traffic graph model, giving us

insight into more relevant intersections when it comes to decision making. Practically, this information could be used as an additional parameter, when planning the system, in order to reduce real-world deployment costs.

The approach used in this study is called Rank-revealing QR (RRQR), a heuristic method for the optimal column subset selection, an NP hard problem sometimes known as maximum submatrix volume [43]. RRQR proposes the idea that in order to select the set of labeled edges, ε^{L}, we need to choose m^{L} rows from V_0 that would maximize the smallest singular value of the resulting submatrix. The RRQR heuristic is defined as:

$$V_C^T \prod = Q[R_1 R_2] \tag{4.11}$$

where $\prod$ is a permutation matrix which keeps R_1 well-conditioned. Additionally, the first m^{L} columns of $\prod$is chosen by the resulting edge set ε^{L} and edge indicated by the column permutation within $\prod$.

Chapter 5

Results

Various simulations on different network sizes were conducted for each of the algorithms implemented. The results of these simulations where then evaluated against each other.

We evaluated our proposed method on synthetically generated network models created using SUMO NetEdit tool. Vehicle trip data and routes were generated using RandomTrips and DuaRouter tools, respectively. Each simulation run required a few parameters as defined in subsection 5.2 Simulation Parameters

A standard four-way intersection is shown in Figure 1.1. These intersections are joined with other similarly designed intersections along connecting roads, making up a traffic network with larger amounts of intersections. Figure 2.1 displays a sample of what the 20-intersection network looks like. Similarly, Figure 4.1 depicts a 17-intersection network model. Different network models were designed to minimize bias and provide us with more versatile results. Intersections within a network are different from other intersections, while many of them are four-way intersections as seen in Figure 1.1, the length of the edges connecting into and out of the intersection varies in length, this affects vehicle travel time and would affect overall congestion. Additionally, the networks were designed to include three-way intersection models as well.

Figure 5.1 depicts a general workflow of how the program runs the proposed system. The workflow process is the same for other simulations conducted with other algorithms, however in those algorithms, we only extract information that is required by the specific algorithms, information such as the adjacency matrix is only required for the proposed system and does not get extracted when running simulations for other algorithms.

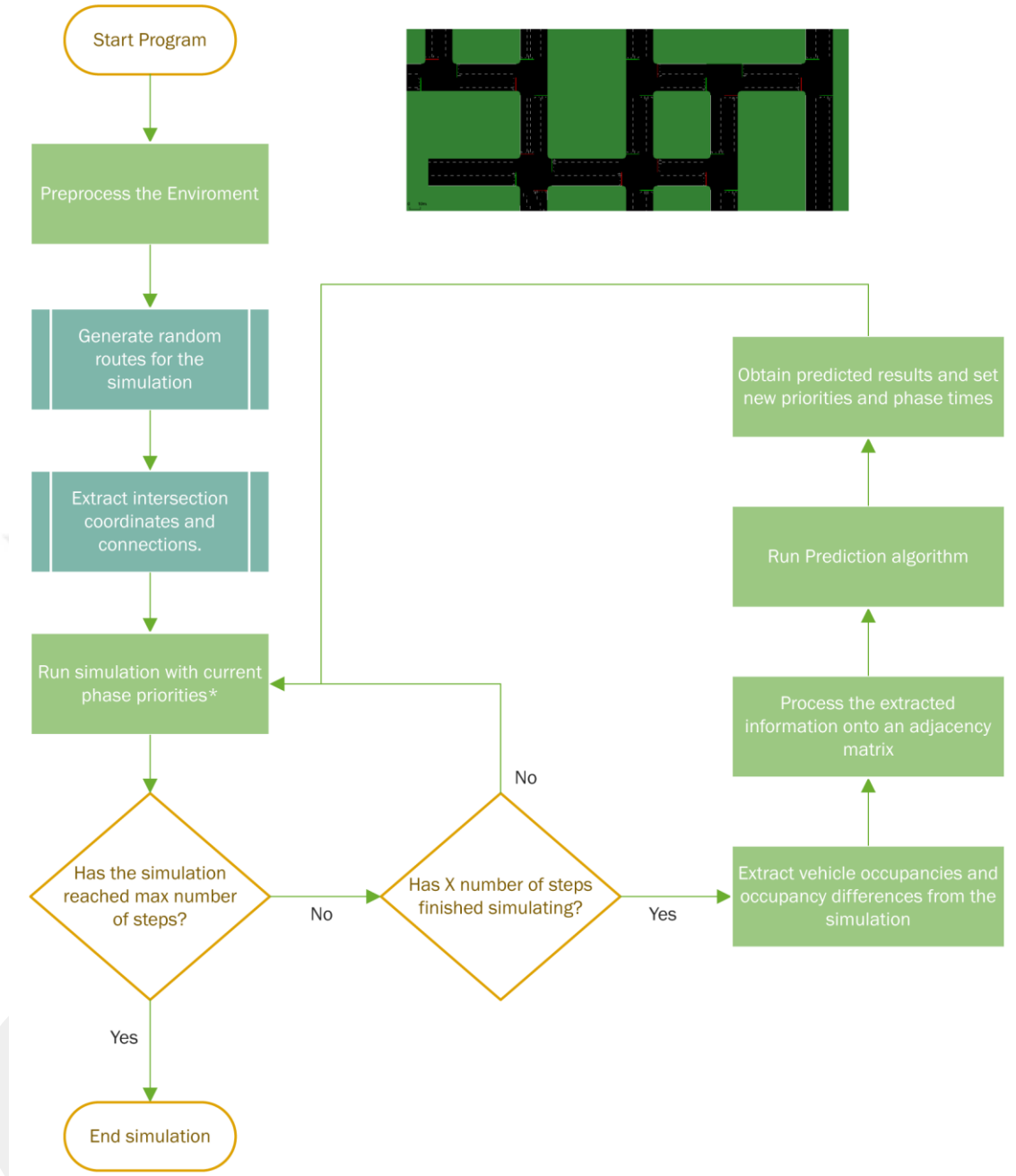
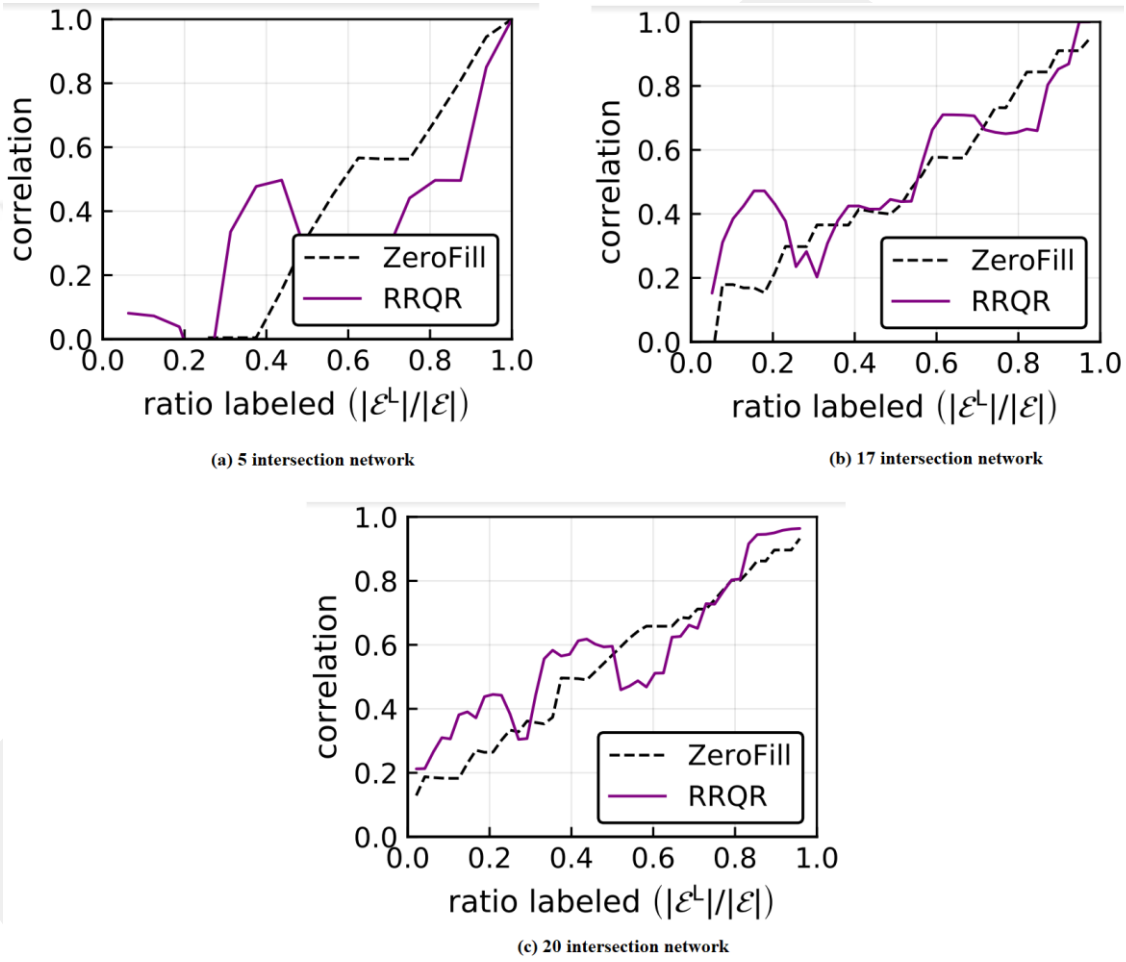


Figure 5.1 Flowchart representation breaking down the steps on how the proposed

5.1 Prediction Performance Results

Results obtained from our RRQR models were compared against ZeroFill baseline, which assigns 0 edge flows to all unlabeled edges, labeled edges were randomly selected. Performance was evaluated based on the Pearson correlation between the estimated flow vector f* and ground truth $\hat{f}$. Figure 5.2 displays the results for the 5, 17 and 20 intersection models, respectively.



(a) 5 intersection network

(b) 17 intersection network

(c) 20 intersection network

Figure 5.2 Graph based SSL for synthetic flows from the simulation on 5, 17 and 20 intersection models.

From Figure 5.2 it can be observed that our RRQR based model works better when there is a lower number of labeled edges. Additionally, it is also observed that the performance improves as the number of intersections within a network is increased.

5.2 Simulation Parameters

5.2.1 Number of steps

The number of steps determines how long a specific simulation would run. Ideally, the simulation would run after all the vehicles generated using DuaRouter have entered the simulation. However, the time at which vehicles enter and leave the simulation would vary based on vehicle routes and traffic control algorithms.

In the experiments conducted, we set the number of steps to 1600 for all simulations run.

5.2.2 Number of simulations

The number of simulations determines how many times we run each algorithm. We run the algorithm multiple times and then take the average results obtained to fairly evaluate the performance of the algorithms. As such, we repeat the simulations 10 times for each algorithm and report the mean and standard deviation of the algorithms' performance.

5.2.3 Traffic Phases

Each intersection in the network has predefined traffic phases. These phases are generated by the SUMO suite. As a default setting, we use the generated phase times. Each algorithm manipulates the traffic phase times according to the results obtained from the algorithm.

5.3 Performance Evaluation

Using three distinct network layouts with 5, 17, and 20 intersections respectively, and their own vehicle demands randomly generated for each simulation, SUMO along with TraCI were used to run the simulation and collect relevant data. Data collection and storage were done by having TraCI run the simulation for X number of steps, then recording all relevant data, and vehicle movement, and proceeding with the simulation. This was done in an event-loop manner as seen in Figure 5.1. The value X is defined as the minimum number of steps until any intersection within the network undergoes a phase

change. This logic ensures that all vehicle movements are correctly recorded synchronously.

Traffic Phase times were calculated as per each of the running algorithms. Performance evaluation was defined to be the vehicles' average wait time inside the network during the simulation. Figure 5.3 - Figure 5.9 presents various average results taken over several simulations conducted for each network layout. Results labeled as 'Flow Prediction' represent the results obtained from the proposed approach described in subsection 4.3 Proposed: Graph-Semi Supervised Learning Based Approach. The label 'Occupancy' represents the results obtained from the simulations using the algorithm described in subsection 4.2.1 Occupancy based algorithm, and the label 'Scoring' represents results obtained from simulations using the algorithm described in subsection 4.2.2 Scoring Algorithm. Certain plots also contain results with labels 'A2C' and 'PPO', these are results obtained from simulations using the algorithms described in subsections 4.2.3.2 Advantage Actor Critic (A2C) [40] and 4.2.3.1 Proximal Policy Optimization (PPO) [40] respectively.

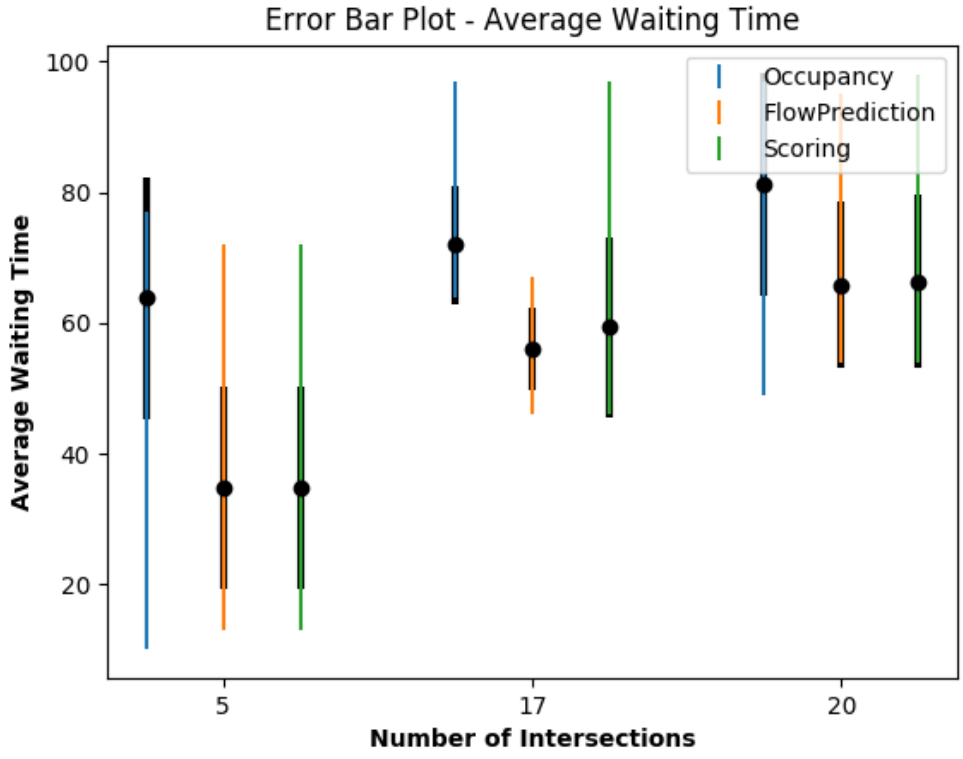


Figure 5.3 Average waiting time for a single vehicle in the network

Figure 5.3 represents the average wait time a single vehicle spends idly waiting on the simulation. It can be observed that the Flow Prediction algorithm outperforms the occupancy-based algorithm significantly for all network models. Additionally, compared to the Scoring Algorithm, it is observed that the performance difference of the Flow Prediction algorithm, compared to the other algorithms, is improved upon increasing the number of intersections.

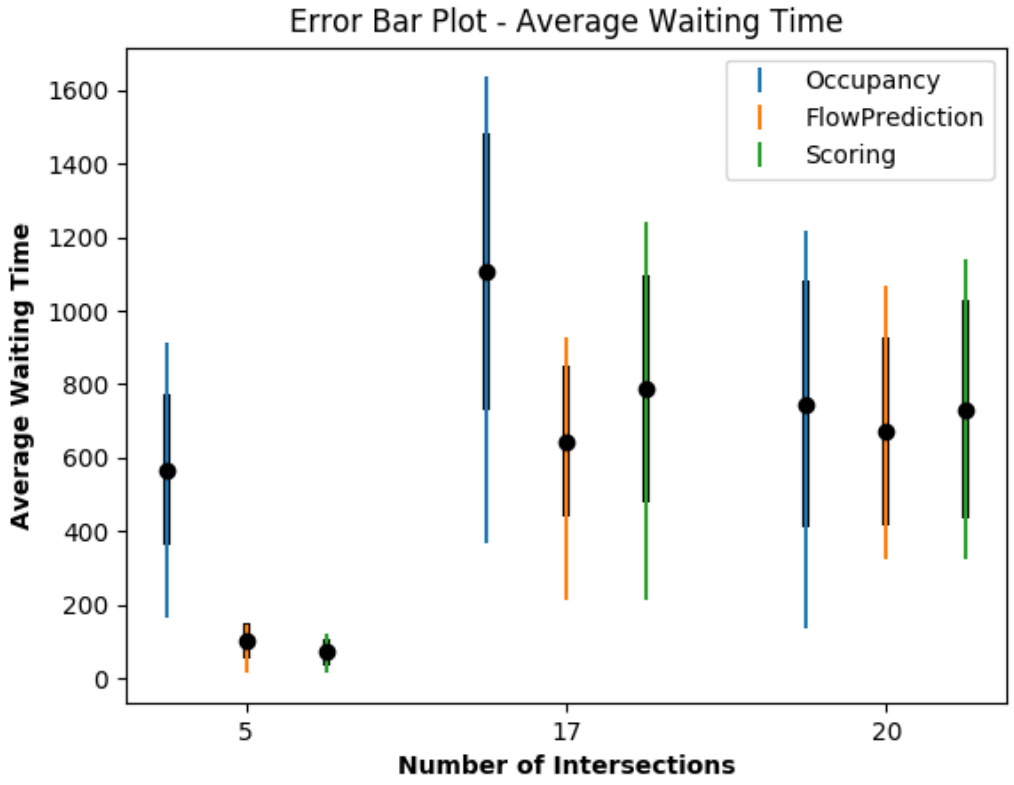


Figure 5.4 Average total waiting time for all vehicles in the network

Figure 5.4 represents the average waiting time for all the vehicles in the network. Observations here are similar to the observations from Figure 5.3. The performance of the flow prediction algorithm is significantly better than the other algorithms, and the performance also improves as the number of intersections increases.

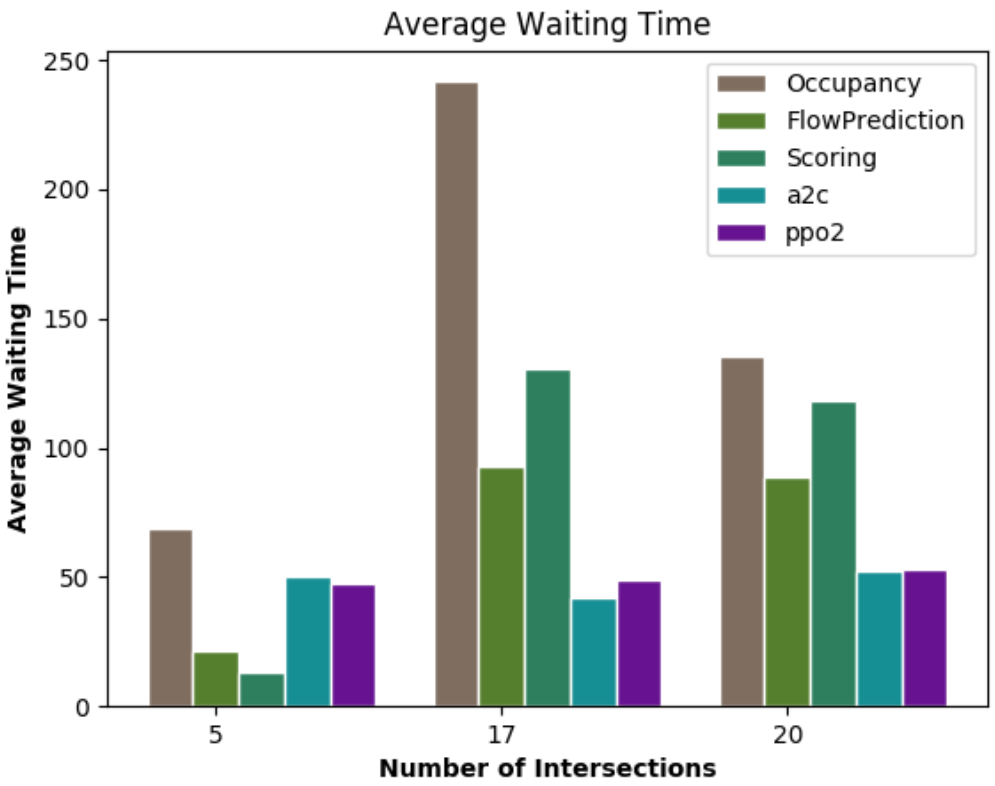


Figure 5.5 Average waiting time for all vehicles in an intersection

The number of simulations run for the A2C, and PPO algorithms were set to 3, due to their running times being significantly higher than the other algorithms. For this reason, standard bar plots were constructed for Figure 5.5 and Figure 5.6.

Figure 5.5 presents the average wait time for all vehicles in the network using a smaller simulation step value. Here it is observed that the flow prediction algorithm always outperforms occupancy-based algorithm, scoring algorithms for larger networks, and RL algorithms in only the smaller network.

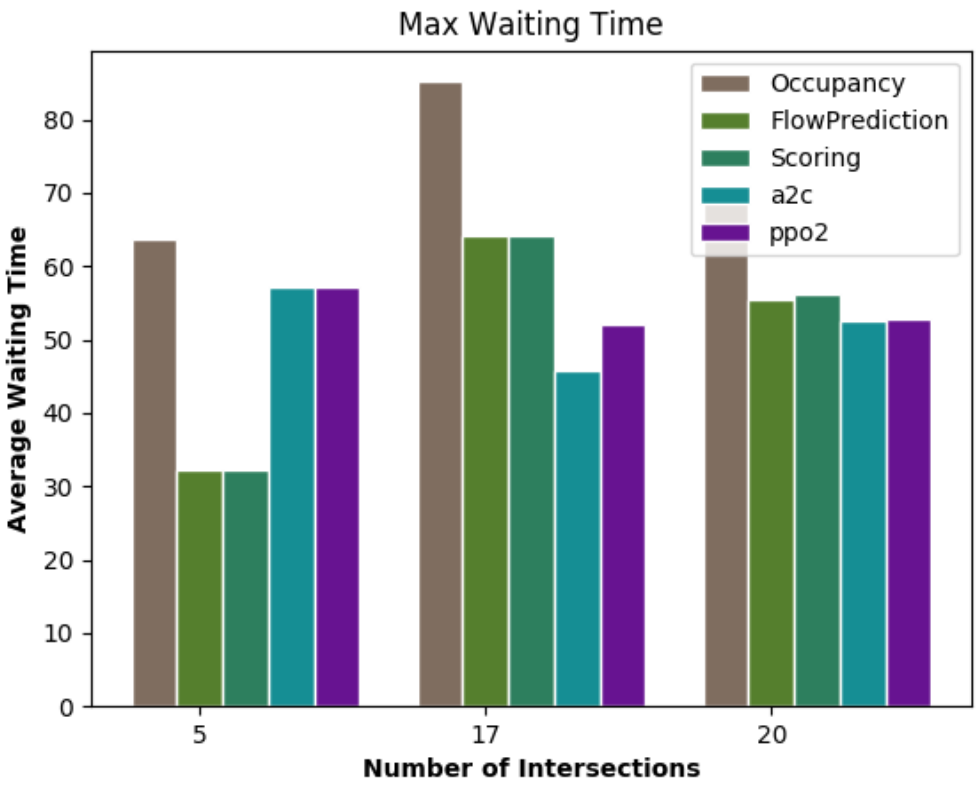


Figure 5.6 Average maximum maximum waiting time for a single vehicle

Figure 5.6 presents the maximum wait time for a single vehicle in the network. Similar to the observations in Figure 5.5; We observe the results in Figure 5.6 show that RL algorithms provide better results in larger networks.

Figure 5.7 and Figure 5.8 presents the average wait time per vehicle per simulation step for the Occupancy based algorithm, Flow Prediction based algorithm, and the Scoring algorithm for simulations run on the 17 intersection and 20 intersection network models respectively. These graphs visualize the decrease in wait time over time for each of the algorithms. It can also be observed how the Flow prediction algorithm outperforms the other algorithms at a much faster rate, and a relationship with the Figure 5.3 and Figure 5.4 can be formed.

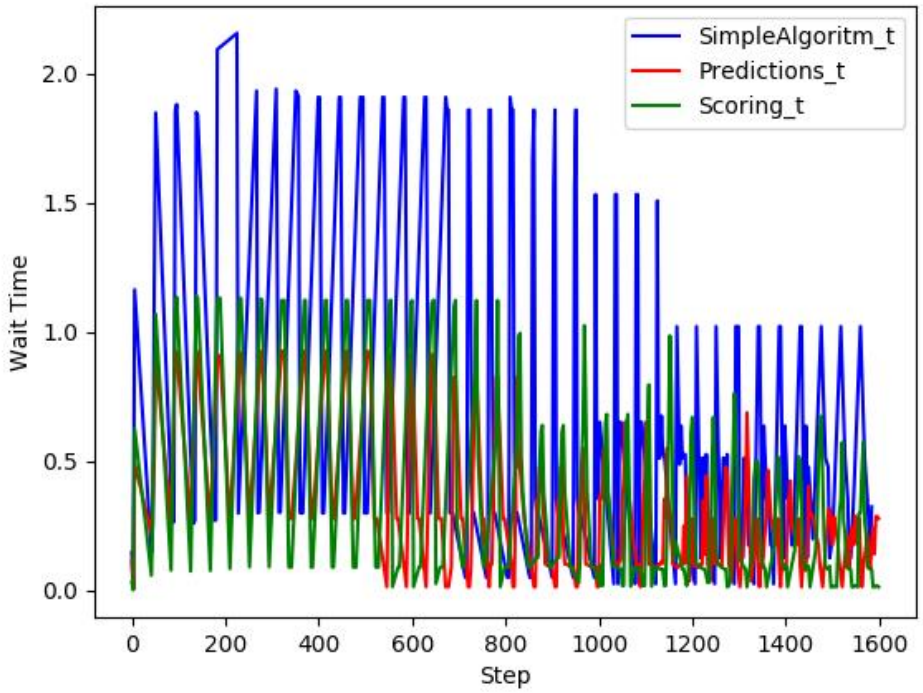


Figure 5.7 Average wait time per vehicle per simulation step – 17 Intersections

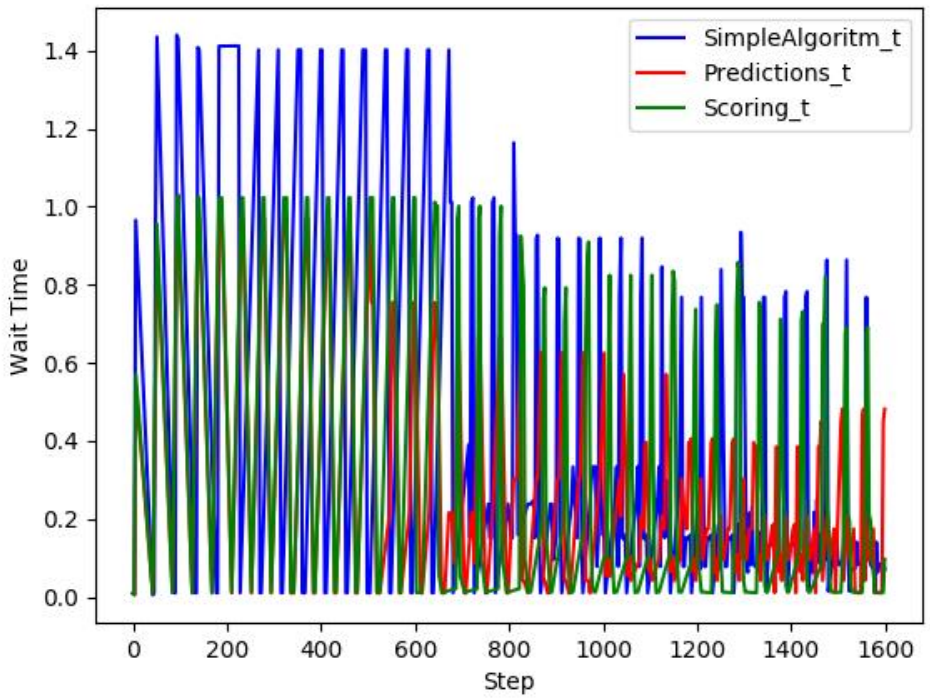


Figure 5.8 Average wait time per simulation step - 20 intersections

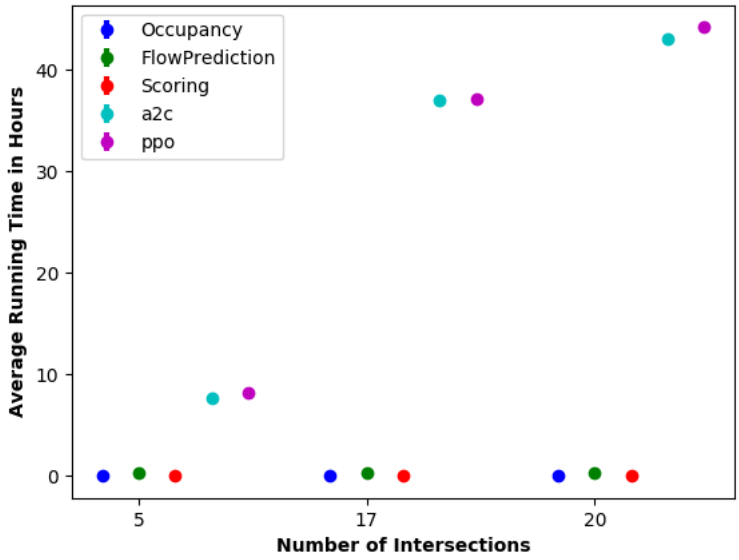


Figure 5.9 Average running times for each simulation

Finally, Figure 5.9 presents the average running time for each of the simulation. In all simulations conducted, the occupancy-based algorithm performs the fastest, followed by the flow prediction and scoring algorithms without too much of a time difference. However, when it comes to the RL based algorithms, the time taken for each of the simulation models is considerably high, additionally, this time also increases with the number of intersections being increased.

Chapter 6

Conclusions and Future Prospects

6.1 Societal Impacts and Contribution to Global Sustainability

As per the studies shown and descriptions provided throughout section 1, it is observed that traffic congestion and delay often impact our society in terms of environmental damage, as well as the monetary cost to individuals and governments in the form of time. Additionally, due to the evergrowing size of cities, population, and the need for vehicles increasing, congestions are bound to occur more frequently. Therefore it is imperative that congestion and delays are minimized.

The research conducted in this study aims to address these issues, by smoothening traffic flow between intersections. This is achieved by modeling multi-intersection network models into a graph-like model, and then predicting traffic flow between the nodes within the model, followed by making informed decisions on connecting intersection phase durations. Additionally, as proposed in Section 4.3.2, our model is also able to identify intersections that are more impactful in reducing congestion, allowing us to focus on them in greater detail. A significant advantage of focusing on the more impactful intersections is cost, as in a real-world scenario, we could dramatically reduce the cost of such a system by having the sensors placed in the more impactful junctions and have the semi-supervised learning features of the study work with unlabelled data on the less impactful junctions. The results displayed in Figure 5.2 demonstrate that this approach provides improving results based on an increasing number of intersections.

Having connecting intersections make informed decisions based on predicted traffic flow reduces the vehicle delay within an intersection significantly, as seen from the results obtained on the various simulations done in different network models. In return, the economical and environmental costs caused by being at an idle state within intersections

are negated. Further research could also contribute to better decision-making systems, which may provide more optimal calculations for predicted traffic flow, and have optimal traffic movement.

To summarize, the conducted study contributes mainly toward economic growth and sustainable cities. These goals are achieved by having improved traffic flow throughout the city. Costs incurred due to traffic-related delays would be significantly reduced, additionally, the proposition of focusing on the more important intersections allows equally effective and cost-efficient implementations of the proposed system.

6.2 Conclusions

This study proposes a novel adaptive traffic light management system that is able to predict traffic flow from one intersection onto another. The principal algorithm behind the proposed system is graph-based semi-supervised learning for edge flows, where each traffic light intersection and vehicle entry/exit points are treated as a vertex node, the roads connecting any two vertices are taken as connecting edges. Magnitudes of edge connections are then calculated using the proposed RRQR method. The obtained information is then used to select and optimize the predefined traffic phases.

Comparative performance evaluations on various traffic intersection configurations show that our approach can produce comparable average vehicle waiting time and drastically reduce the training/learning time of learning an adequate traffic light configurations for all intersections within a short period of time, whereas training deep learning based approaches can consume over a few hours.

Main conclusions are as follows:

i. The proposed approach is able to predict traffic outflow from a given junction. This outflow can be used on connected junctions to predict congestion and optimize phase durations.
ii. Using the proposed RRQR heuristic for the prediction allows us to optimize the average waiting times in a traffic environment as well as pay more attention to more impactful junctions.
iii. Larger network models lead to higher accuracy in predictions, as well as better performance of the model.

iv. To the best of our knowledge, the proposed approach is a novel concept. No other study uses edge-based semi-supervised learning to predict vehicle flows between traffic networks.

6.3 Future Prospects

Future efforts in this direction would include further optimization of the traffic decision-making process, based on predicted traffic flows; This may include additional concepts such as fixed or dynamic cycle-times and phase orders. Additionally, it would also be beneficial to improve the running time of the flow prediction-based algorithm, faster algorithm runtime would lead to seamless integration with real-world applications.

BIBLIOGRAPHY

[1] "Leading Transportation Analytics Solutions | INRIX." https://inrix.com/ (accessed Apr. 26, 2022).

[2] A. J. Miller, "Settings for Fixed-Cycle Traffic Signals," *Journal of the Operational Research Society*, vol. 14, no. 4, pp. 373–386, Dec. 1963, doi: 10.1057/JORS.1963.61.

[3] F. Webster, "Traffic signal settings," 1958, Accessed: Apr. 26, 2022. [Online]. Available: https://trid.trb.org/view/113579

[4] S. B. Cools, C. Gershenson, and B. D'Hooghe, "Self-organizing traffic lights: A realistic simulation," *Advanced Information and Knowledge Processing*, no. 9781447151128, pp. 45–55, 2013, doi: 10.1007/978-1-4471-5113-5_3.

[5] M. Coşkun, A. Baggag, S. C.-2018 I. International, and undefined 2018, "Deep reinforcement learning for traffic light optimization," *ieeexplore.ieee.org*, Accessed: Apr. 26, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8637414/?casa_token=xcQHEqR9R68AAAAA:q9tqSuM-WMtzl-CksB9VZbbTP2jv53_Rs2xKx10ffa_3dhKacWSL9sr9G6PFddWUxghMBTqaQd1L

[6] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis, "Multiagent reinforcement learning for Urban traffic control using coordination graphs," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5211 LNAI, no. PART 1, pp. 656–671, 2008, doi: 10.1007/978-3-540-87479-9_61.

[7] E. van der Pol, F. O. learning, inference and control of, and undefined 2016, "Coordinated deep reinforcement learners for traffic light control," *elisevanderpol.nl*, Accessed: Apr. 26, 2022. [Online]. Available: https://www.elisevanderpol.nl/papers/vanderpolNIPSMALIC2016.pdf

[8] M. W.-M. L. P. of the Seventeenth and undefined 2000, "Multi-agent reinforcement learning for traffic light control," *dcsc.tudelft.nl*, Accessed: Apr. 26, 2022. [Online]. Available: http://www.dcsc.tudelft.nl/~sc4081/2018/assign/pap/Reinforcement_Learning.pdf

[9] P. Lopez, M. Behrisch, … L. B.-W.-2018 21st, and undefined 2018, "Microscopic traffic simulation using sumo," *ieeexplore.ieee.org*, Accessed: Apr. 26, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8569938/?casa_token=kU8yoFC35CEAAAAA:c-oxQPWhT9apGfe9qLy-cb3ANkLXHmVcqGHebRNTyfL_54hnyhVj2yWCdqDCWHUtVasGjvr5XA_5

[10] "Eclipse SUMO - Simulation of Urban MObility." https://www.eclipse.org/sumo/ (accessed Apr. 27, 2022).

[11] "netedit - SUMO Documentation." https://sumo.dlr.de/docs/Netedit/index.html (accessed Apr. 27, 2022).

[12] "Trip - SUMO Documentation." https://sumo.dlr.de/docs/Tools/Trip.html (accessed Apr. 27, 2022).

[13] "duarouter - SUMO Documentation." https://sumo.dlr.de/docs/duarouter.html (accessed Apr. 27, 2022).

[14] S. Ali, B. George, … L. V.-I. T. on, and undefined 2011, "A multiple inductive loop vehicle detection system for heterogeneous and lane-less traffic,"

ieeexplore.ieee.org, Accessed: Apr. 26, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6095626/?casa_token=Kffy7QWvTaMAAAAA:pbA6_m0YEKG3BiEUjB6xw8bEJaW0IU46dj2fh8qp1Ljxno5fRIAUWezeBLqK-w-S9hcE9vx6hl0k

[15] N. Lanke, S. K.-I. J. of C. Applications, and undefined 2013, "Smart traffic management system," *researchgate.net*, vol. 75, no. 7, pp. 975–8887, 2013, doi: 10.5120/13123-0473.

[16] W. Hooda, P. Yadav, … A. B.-P. of the S., and undefined 2016, "An Image Processing Approach to Intelligent Traffic Management System," *dl.acm.org*, vol. 04-05-March-2016, Mar. 2016, doi: 10.1145/2905055.2905091.

[17] Y. I.-P. of C. on Intelligent and undefined 1997, "An image processing system to measure vehicular queues and an adaptive traffic signal control by using the information of the queues," *ieeexplore.ieee.org*, Accessed: Apr. 26, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/660474/

[18] F. Barrero, J. Guevara, E. Vargas, … S. T.-C. S. &, and undefined 2014, "Networked transducers in intelligent transportation systems based on the IEEE 1451 standard," *Elsevier*, Accessed: Apr. 26, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548912000633?casa_token=TJXiHvDNavcAAAAA:gT3ETkfnZYtyhh92Piy0kI3TkYiyDaWemrMTGFf0xurrBD1-5racSjTWMJdjt1QCUrIXD_NZYHU

[19] D. Gregor, S. Toral, T. Ariza, F. Barrero, … R. G.-C. S. &, and undefined 2016, "A methodology for structured ontology construction applied to intelligent transportation systems," *Elsevier*, Accessed: Apr. 26, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548915001178?casa_token=TTnXKgYHTmcAAAAA:wj_cRVzYF8PvvVbrqhrQ0Re2FcrXzsz2BZfihe4ya-NFaUj9KFY6waNVNtrVoa1JFz91aaP-rWk

[20] H. Pan, S. Wang, K. Y.-C. S. & Interfaces, and undefined 2014, "An integrated data exchange platform for Intelligent Transportation Systems," *Elsevier*, Accessed: Apr. 26, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548913000949?casa_token=53hZpyTLXHkAAAAA:8MVMZCjlqQJPhNSgL7GHcRR8FVAS6NMXGmaP9hhZ-XBl_99JtXw97HHFxqy8r2res3hic-b7GBU

[21] I. Román, G. Madinabeitia, L. Jimenez, … G. M.-C. S. &, and undefined 2013, "Experiences applying RM-ODP principles and techniques to intelligent transportation system architectures," *Elsevier*, Accessed: Apr. 26, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092054891100136X?casa_token=Uk7FbPjczboAAAAA:U4l_t9kA4nqdJDbbsJ4uEVsp9OvqGQ3rK4G5RzeXgrZNcPifa0j3uRW12Kq08pPq2062WX5jRew

[22] S. Toral, F. Barrero, F. Cortés, D. G.-C. S. & Interfaces, and undefined 2013, "Analysis of embedded CORBA middleware performance on urban distributed transportation equipments," *Elsevier*, Accessed: Apr. 26, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0920548912000815?casa_token=Iuqw3KLHBlkAAAAA:pyphx31NHmvoMS16OkcBEmxomL7c7rfBqP_mNnOb0j2ni5fEPLYwpo_oyxWaGNaftZ15CmFgZUk

[23] F. Dion, H. Rakha, Y. K.-T. R. P. B, and undefined 2004, "Comparison of delay estimates at under-saturated and over-saturated pre-timed signalized intersections," *Elsevier*, Accessed: Apr. 26, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261503000031?casa_tok

en=j5ClsHrFQkMAAAAA:usevIMm0OIuonFFQ4rYJGIOXKCV_MKTkxxxOB2QWGjFMeA5Q5VBnMo-xp185VoXEtk_MFbazxu4
[24] I. Porche and S. Lafortune, "Adaptive look-ahead optimization of traffic signals," *ITS Journal*, vol. 4, no. 3, pp. 209–254, 1999, doi: 10.1080/10248079908903749.
[25] B. Abdulhai, R. Pringle, and G. J. Karakoulas, "Reinforcement Learning for True Adaptive Traffic Signal Control", doi: 10.1061/(ASCE)0733-947X(2003)129.
[26] N. Bolong, S. S. Yang, Y. Kwong Chin, A. Kiring, K. Tze, and K. Teo, "Q-learning based traffic optimization in management of signal timing plan," *researchgate.net*, 2011, doi: 10.5013/IJSSST.a.12.03.05.
[27] P. Mannion, J. Duggan, and E. Howley, "An Experimental Review of Reinforcement Learning Algorithms for Adaptive Traffic Signal Control," *Autonomic Road Transport Support Systems*, pp. 47–66, 2016, doi: 10.1007/978-3-319-25808-9_4.
[28] 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.
[29] P. Balaji, … X. G.-I. I. T., and undefined 2010, "Urban traffic signal control using reinforcement learning agents," *ieeexplore.ieee.org*, 2010, doi: 10.1049/iet-its.2009.0096.
[30] S. El-Tantawy, … B. A.-I. T. on, and undefined 2013, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): methodology and large-scale application on downtown," *ieeexplore.ieee.org*, Accessed: Apr. 26, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6502719/?casa_token=auJaJ-DP84AAAAAA:8itU4G6RzIPzVKxb4oMpdLTv1b3bkILUZABQLdstDqwr7dYRFPjYm2WgxYQ8juMmudn2FDYBOKOx
[31] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," *proceedings.mlr.press*, 2016, Accessed: Apr. 26, 2022. [Online]. Available: http://proceedings.mlr.press/v48/mniha16.html?ref=https://githubhelp.com
[32] J. Schulman, F. Wolski, P. Dhariwal, … A. R. preprint arXiv, and undefined 2017, "Proximal policy optimization algorithms," *arxiv.org*, Accessed: Apr. 26, 2022. [Online]. Available: https://arxiv.org/abs/1707.06347
[33] H. Wei *et al.*, "Colight: Learning network-level cooperation for traffic signal control," *dl.acm.org*, pp. 1913–1922, Nov. 2019, doi: 10.1145/3357384.3357902.
[34] 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.
[35] J. Jia, M. Schaub, S. Segarra, A. B.-P. of the 25th ACM, and undefined 2019, "Graph-based semi-supervised & active learning for edge flows," *dl.acm.org*, pp. 761–771, Jul. 2019, doi: 10.1145/3292500.3330872.
[36] F. Dion, H. Rakha, and Y.-S. Kang, "Comparison of delay estimates at under-saturated and over-saturated pre-timed signalized intersections," *Transportation Research Part B: Methodological*, vol. 38, no. 2, pp. 99–122, 2004.
[37] B. Smith, M. D.-P. of I. international, and undefined 1994, "Short-term traffic flow prediction models-a comparison of neural network and nonparametric regression approaches," *ieeexplore.ieee.org*, Accessed: Apr. 26, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/400094/?casa_token=z7eH56jjQAMAAAAA:WxVaVYZx1rXyN6nQdoz2WgADNXLdHjcq2WiCyzYDerFc03pd9lh9ZB1Ss3r1UBMHfcYiWLd54LN7

[38] G. Zheng *et al.*, “Learning phase competition for traffic signal control,” *dl.acm.org*, p. 10, Nov. 2019, doi: 10.1145/3357384.3357900.

[39] S. Faye, C. Chaudet, and I. Demeure, “A distributed algorithm for multiple intersections adaptive traffic lights control using a wireless sensor networks,” *CoNEXT UrbaNe 2012 - Proceedings of the ACM Conference on the 1st Workshop on Urban Networking*, pp. 13–18, 2012, doi: 10.1145/2413236.2413240.

[40] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-Baselines3: Reliable Reinforcement Learning Implementations,” *jmlr.org*, vol. 22, pp. 1–8, 2021, Accessed: Apr. 26, 2022. [Online]. Available: https://www.jmlr.org/papers/volume22/20-1364/20-1364.pdf

[41] A. Gadde, A. Anis, … A. O.-20th A. S. international conference, and undefined 2014, “Active semi-supervised learning using sampling theory for graph signals,” *dl.acm.org*, pp. 492–501, 2014, doi: 10.1145/2623330.2623760.

[42] A. Guillory, J. B.-I. P. Systems, and undefined 2009, “Label selection on graphs,” *proceedings.neurips.cc*, Accessed: Apr. 26, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2009/hash/90794e3b050f815354e3e29e977a88ab-Abstract.html

[43] A. Civril, M. M.-I.-T. C. Science, and undefined 2009, “On selecting a maximum volume sub-matrix of a matrix and related problems,” *Elsevier*, Accessed: Apr. 26, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304397509004101

APPENDIX

Appendix A: Python Code used to run a SUMO simulation using TraCI

```
# Import system packages to add SUMO files to python path
import os, sys

# Import traci package. Can be installed using `python -m pip install
traci`
import traci

# Defines some of the variables used when running on IDLE or as a script
file.
LOCAL_SUMO_FOLDER = 'sumo_tools'
SUMO_BINARY = None
N_STEPS = 1600

tools = None

if 'SUMO_HOME' in os.environ:
    tools = os.path.join(os.environ['SUMO_HOME'], 'tools')
    sys.path.append(tools)
elif os.path.isdir(LOCAL_SUMO_FOLDER):
    tools = os.path.join(os.getcwd(), f'{LOCAL_SUMO_FOLDER}/tools')
    sys.path.append(tools)
else:
    localPath = os.path.join(os.getcwd(), f'{LOCAL_SUMO_FOLDER}')
    print (f'[ERR] SUMO_HOME environment variable not defined.')
    print (f'If you are using local files, ensure the path is correct.
[{localPath}]')
    sys.exit(1)

# Ensure that valid tools folder was found. Required to define
SUMO_BINRAY and SUMO_COMMAND
assert not type(tools) == type (None), 'No tools folder found.'

SUMO_BINARY  = f'{tools}/sumo-gui'
SUMO_COMMAND = [SUMO_BINARY, '-c',
'sample_network_configuration.sumocfg']

if __name__ == "__main__":
    # Start the simulation
    traci.start (SUMO_COMMAND)

    # Run the simulation for N_STEPS number of steps
    localStep = 0
```

```
    while localStep < N_STEPS:
        traci.simulationStep()

    # Close the connection
    traci.close(False)

    print (f'Simulation Completed :)')
```

CURRICULUM VITAE

EXPERIENCE

2018	Summer Intern, Attune Consulting Colombo, SRI LANKA
2020 – Present	Computer Engineer, Kayseri Ulaşim Kayseri, TURKEY

EDUCATION

2017 – 2019	B.Sc., Computer Engineering, Abdullah Gul University Kayseri, TURKEY
2019 – Present	M.Sc., Electrical and Computer Engineering, Abdullah Gul University Kayseri, TURKEY

PUBLICATIONS

J1) Dundar, Munis, et al. "Clinical and molecular evaluation of MEFV gene variants in the Turkish population: a study by the National Genetics Consortium." Functional & Integrative Genomics (2022): 1-25.

J2) Adam R. Thahir, et al. "Intelligent Traffic Light Systems using Edge Flow Predictions" (under review).