INTELLIGENT DYNAMIC ROUTE OPTIMIZATION AND ROAD PRE-EMPTION SYSTEM FOR ON-ROAD EMERGENCY SERVICES

A THESIS SUBMITTED TO AUCKLAND UNIVERSITY OF TECHNOLOGY

IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Supervisor

Assoc.Prof. Roopak Sinha

Prof. Edmund Lai

Dr. Prakash Ranjitkar

30-01-2021

By

Subash Humagain

School of Engineering, Computer and Mathematical Sciences

Abstract

Increased response time of emergency vehicles (EVs) can cause an irreparable loss of life and property. Reducing emergency services' response times requires identifying the most effective optimization and pre-emption techniques and factors for reducing EVs travel times. Research in this domain has adopted one or both of optimization and pre-emption techniques for routing EVs. None of the existing studies provides any comparative evidence that a particular optimization technique and pre-emption system is better suited to reducing the EV's travelling time.

An appropriate solution to improve existing techniques requires dynamic optimization and efficient and precise pre-emption to cause minimal disruption to other vehicles. The success of such dynamic optimization and pre-emption systems depends on the availability of real-time dynamic traffic data. This means that sensors deployed at various road network infrastructures must communicate in real-time and support real-time decision-making. In general, traffic infrastructure requires a deeper integration with software systems to ensure high availability of accurate real-time data. This thesis's main focus is to develop precise pre-emption and dynamic optimization techniques that can aid in reducing the response time of EVs. The main contributions of this thesis are:

- a conceptual model to analogically map traffic scheduling with scheduling algorithms in real-time systems.

- an adaptive fair scheduling algorithm (FSA) that ascertains higher travel time reliability developed by analogically mapping traffic domain with real-time systems.
- an emergency vehicle pre-emption (EVP) technique with different levels of priorities that ascertains a certain level of performance assurance to different criticality levels in emergency services.
- a decentralized self coordinating traffic system to prioritize emergency vehicle movement through an isolated traffic intersection using Virtual traffic lights plus for emergency vehicles (VTL+EV) algorithm.
- an ensemble-based model that learns from multiple pre-engineered features related to topology, directions, the shape of the trajectory, path, speed limits, map distance traversed, real-time traffic conditions and precisely estimate the travel time.

We conducted exhaustive experimentation, and empirically proved that the FSA is more reliable and fairer in scheduling in terms of travel reliability compared to existing state-of-art algorithms in terms of buffer index and Jain's fairness index. Our experimental results confirm that the EVP algorithm can significantly reduce the average waiting time of regular traffic and ensure all EVs meet their target response time. Comprehensive experiments and results showed that VTL+EV has the evident advantage of reduced waiting time for regular traffic and emergency vehicles in both congested and non-congested traffic conditions. We also concluded that for wisely extracted manual features, ensemble-based gradient boost regression approach could outperform existing state-of-art baseline models that employ deep neural networks in travel time estimation.

Contents

List of Tables

List of Figures

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.

_____

Signature of candidate

Publications

This thesis includes following six articles, for which I am the main contributor (from problem specification, research design, methodology development and implementation to the analysis of results and writing):

Publications	**Author Percentage contribution**
Humagain, S., Sinha, R., Lai, E., & Ranjitkar, P. (2020). A systematic review of route optimisation and pre-emption methods for emergency vehicles. Transport reviews, 40(1), 35-53.	Humagain: 85% Sinha: 10% Ranjitkar & Lai: <5%
SubashHumagain, S., & Sinha, R. Adaptive Traffic Signal Control Using Travel Time Reliability for Vehicular Ad hoc Networks. Transportation Research Part c: Emerging Technologies (received feedback twice under review)	Humagain: 90% Sinha: 10%
Humagain, S., & Sinha, R. (2020, September). Routing Emergency Vehicles in Arterial Road Networks using Real-time Mixed Criticality Systems. In 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC) (pp. 1-6). IEEE.	Humagain: 90% Sinha: 10%
Humagain, S., & Sinha, R. (2020, October). Dynamic Prioritization of Emergency Vehicles For Self-Organizing Traffic using VTL+ EV. In IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society (pp. 789-794). IEEE.	Humagain: 90% Sinha: 10%
Humagain, S., Gowdra, N., & Sinha, R. . Travel Time Estimation with Decision-Tree-Based Ensemble Networks and Enhanced Feature Extraction. IEEE transactions on intelligent transportation systems. (submitted for publication, under review)	Humagain: 85% Gowdra: 10% Sinha: 5%
Humagain, S., & Sinha, R. (2019, July). Routing Autonomous Emergency Vehicles in Smart Cities Using Real Time Systems Analogy: A Conceptual Model. In 2019 IEEE 17th International Conference on Industrial Informatics (INDIN) (Vol. 1, pp. 1097-1102). IEEE.	Humagain: 90% Sinha: 10%

We, the undersigned, hereby agree to the percentage of contribution to the papers listed above.

Assoc.Prof. Roopak Sinha

Prof.Edmund Lai

Dr.Prakash Ranjitkar

Dr.Nidhi Gowdra

Acknowledgements

I want to express my special appreciation to my primary supervisor Assoc.Prof. Roopak Sinha for his support, time, ideas and guidance throughout my PhD. I want to thank my secondary supervisors Prof. Edmund Lai and Dr. Prakash Ranjitkar, for their support. Special thanks to my family for their support and patience.
I would also like to thank all the colleagues from software engineering and research lab (SERL), all the group members from embedded software (EMSOFT) group. Special thanks to my friend and brother Nidhi Gowdra to listen to my irrelevant questions and be there when needed.
My colleagues in AUT, fellow PhD candidates and students.

Subash Humagain

Chapter 1

Introduction

1.1 Background

An emergency vehicle (EV) response time is the time interval from receiving an emergency call to the arrival of an EV to the emergency location. The response time depends primarily on time it takes for an EV to travel to an emergency location. The travel time of an EV depends on several static parameters such as distance and number of intersections on the route (signalized/unsignalised) and dynamic parameters like flow, average speed, and the number of stops. The presence of numerous such parameters makes reducing EV travel times complex and challenging (Fitch, 2005a).

Increased response time of EVs can cause an irreparable loss of life and property. In medical emergencies such as cardiac arrest, every one-minute delay in response time causes mortality rate to increase by 1% and imposes additional USD 1542 in hospital costs, leading to 7 billion dollars increase in healthcare expenditure per year only in USA (RapidSOS, 2015). Other emergencies like building fires typically grow by 20% per minute, causing an average USD 4000 of additional damages (RapidSOS, 2015). These factors have led to extensive research on reducing EV travel time (Fitch, 2005b).

It is intuitively understood that EVs must receive priority over other vehicles when

travelling from the source to the response scene. EVs get priority using special appearance, sirens and strobe lights, a dedicated green light on approaching traffic signal, special lanes, etc. They travel to service an emergency in an optimized route (Eltayeb, Almubarak & Attia, 2013). To measure the used technique's effectiveness, governments impose target response times for Emergency Management Services (EMS). For instance, 90% of critical emergency calls must be responded to within 9 minutes in the USA (Pons & Markovchick, 2002a), while in the UK 75% of such cases must be responded to within 8 minutes (*Ambulance Quality Indicators Data 2019-20*, 2019). For Australia and New Zealand, the target is to respond to 50% of emergency calls within 8 and 10 minutes, respectively (*Annual report 2019*, 2019; *NSW state emergency service annual report 2015*, 2015). Due to barriers present in modern complex traffic networks like congested road network, synchronized traffic lights operation, continuous construction over lanes, and increased pedestrian population over cities, it has become an increasingly difficult challenge for EMS to meet contractual timings.

Road congestion is a growing problem. Presently we have about 1.2 billion vehicles globally, and the number is expected to touch 2 billion marks by 2035 (Voelcker, 2020). Every new vehicle is adding congestion on road networks. Globally, there has been an increase of 23% in congestion levels from 2008 to 2016 with an increase of 9.6% from 2015 to 2016 which means we are spending 11 minutes 18 sec more on travelling than 2015 daily (Traffic, 2020). It also has an adverse effect on the economy. Congestion cost UK $30 billion, Germany $48 billion, USA $124 billion and for a mid-sized city like Auckland 2 billion USD per year resulted from time and fuel wasted in traffic (INRIX, 2017).

Several studies have explored this problem of reducing response time of EVs, proposing several solutions, which can be broadly classified into route optimization and pre-emption (Zhu, Chen & Bing, 2014). Optimization is the process of attaining the highest achievable performance under a given set of constraints by maximizing desired

factors and minimizing undesired ones. Route optimization treats traffic networks as graphs. Different cost functions like distances over available travel paths, travel times along sections of the network, and fuel efficiency are assigned as weights to the graph's edges. Optimization algorithms then maximize or minimize the cost function (Winter, 2002). According to (Eksioglu, Vural & Reisman, 2009), route optimisation is mainly distance (path)-dependent or travel (time)-dependent. Route pre-emption, sometimes referred to as "traffic signal prioritization" or "transit signal priority", changes or alters traffic control to grant priority to special vehicles like EVs. The most commonly used tactic is manipulating traffic signals in the route of an EV, halting lower-priority traffic and providing right-of-way to the EV (Gedawy, Dias & Harras, 2008; Paniati & Amoni, 2006). Current pre-emption techniques carry out three major categories of tasks: defining when to activate pre-emption, determining the position of an EV using sensing techniques to trigger pre-emption, and using communication techniques between an EV and the infrastructure/other vehicles to execute pre-emption.

1.2 Research Gaps

Our systematic literature review on route optimization and pre-emption methods for emergency vehicles (Humagain, Sinha, Lai & Ranjitkar, 2020) has identified that the research in this domain has adopted the use of one or both of optimization and pre-emption for routing EVs. None of the studies provides any comparative evidence that a particular optimization technique and pre-emption system is better suited to reducing the EV's travelling time. We can also conclude that although much research has been conducted for reducing the response time of EVs, there has not been a considerable decrease in response time. This indicates a potential "dead-end" in the way research has approached reducing EV travel times and signals a need to explore newer methods to approach this problem. Detailed analysis of these current research gaps can help solve

the dead-end that research has produced in reducing EV travel times. Future research should focus on the following directions listed below:

- **Real-Time dynamic traffic Data:** Research should focus on integrating real-time on road traffic data to calculate more dynamic, reliable and accurate routes to EVs (Elmandili, Toulni & Nsiri, 2013; Fleischman, Lundquist, Jui, Newgard & Warden, 2013; W. Huang, Yang & Ma, 2011; Kai, Yao-ting & Yue-peng, 2014; Musolino, Polimeni, Rindone & Vitetta, 2013a).

- **Time as a critical parameter:** Finding the shortest path is not enough to improve emergency response system in a complex road network as minimum travel time is a major parameter to consider (Bachelder, 2011; Choosumrong, Raghavan & Bozon, 2012; Mali, Rao & Mantha, 2012).

- **Advanced algorithms:** Basic graph theory method and mathematical programming method cannot meet the calculation requirement of real-time traffic (Brady & Park, 2016; Chakraborty, Tiwari & Sinha, 2015; Elalouf, 2012; Sun, Yue & Yao, 2014).

- **Use of VANETs:** With the advancement of the wireless communication technologies like Cooperative Vehicle-Infrastructure System (CVIS), there is an opportunity to provide appropriate traffic signal pre-emption for an emergency vehicle based on real-time emergency vehicle data, traffic volume data, and traffic signal timings (Agarwal & Paruchuri, 2016; Anand & Flora, 2014; Djahel, Smith, Wang & Murphy, 2015; Jayaraj & Hemanath, 2015; Shekar et al., 2012; Y. Wang, Wu, Yang & Huang, 2013).

- **Concerns with multiple EVs:** Future studies can include considerations of more severe scenarios, such as disasters where a large number of EVs are required (C.-Y. Chen, Chen & Chen, 2013; Moroi & Takami, 2015).

- **Safety of EV travel:** It is a challenge to ensure the safe passage of an emergency vehicle (EV) or multiple EVs and at the same time to maintain safe and smooth traffic flow in the road network (Qin & Khan, 2012; Yoo, Kim & Park, 2010).

- **Intelligent pre-emption :** Limited research has been done on the use of intelligent pre-emption control, which has the ability to use real-time traffic information to minimize emergency vehicle delays. Simultaneously, reducing the adverse impacts of emergency vehicles on normal traffic, so that they can cause the least disturbance to network traffic flow is a challenge (Kang et al., 2014; Kamalanathsharma & Hancock, 2012; Nellore & Hancke, 2016).

A critical analysis of optimization and pre-emption suggests a difference between actual travel time and theoretically calculated travel time. This difference arises as dynamic parameters like increased congestion, halt on a road, pedestrian flow, queued vehicles, real and adaptive speed are not being addressed within the theoretical models. Similarly, pre-emption is also not effective. The timing of activation in implemented systems is not precise, and pre-emption techniques often do not consider the effect of pre-emption over other vehicles (Humagain et al., 2020).

An appropriate solution to improve existing techniques will require dynamic optimization and efficient and precise pre-emption to cause minimal disruption to other vehicles. The success of such combined and dynamic optimization and pre-emption systems depends on the availability of real-time dynamic traffic data. This means that sensors deployed at various road network infrastructures must communicate in real-time and support real-time decision-making. In general, traffic infrastructure requires a deeper integration with software systems to ensure high availability of accurate real-time data.

1.3 Research Focus

This section elaborates our research focus guided from wisely designed research questions. Our research's aim is to develop precise pre-emption and dynamic optimization technique that can aid in reducing the response time of EVs. To achieve the research aim, we define the following objectives:

- Conduct a systematic literature review on optimization and pre-emption techniques employed in EV routing.
- Design real-time adaptive traffic control system to improve performance of current signalised traffic intersections.
- Develop emergency vehicle pre-emption algorithm to prioritize EVs.
- Develop infrastructure free traffic lights that assigns priority to EVs.
- Implement a machine learning model to predict travel time accurately.

We follow thesis by publication format to answers the following research questions in order to achieve our objectives.

RQ1 ***What optimization and pre-emption techniques are employed in EV routing?*** Route optimization and pre-emption are powerful techniques used to reduce EV travel time. It is crucial to investigate the existing solutions and identify their advantages and limitations. The thesis conducts a systematic literature review of route optimization and pre-emption methods for emergency vehicles (Chapter: Literature Review, manuscript 1). It provides a detailed classification of existing techniques, presented along with critical analysis and discussion. It identifies multiple research gaps that researchers can consider to improve existing route optimization and pre-emption systems.

RQ2 ***How to design a real-time adaptive traffic control system to improve the performance of signalised traffic intersections?***

Efficient routing of vehicles is highly dependent on the performance of signalized traffic intersection, which can be measured using several performance parameters: throughput (total vehicles passing through an intersection within the time of observation), approach delay (delay a vehicle experience before passing an intersection), and speed of vehicles in each approach (Balke, Charara & Parker, 2005). Adaptive traffic control (ATC) systems manage traffic signals' timing following actual traffic demand changes (Maslekar, Boussedjra, Mouzna & Labiod, 2011). Vehicular ad hoc networks (VANETs) can provide more abundant data to aid ATC systems. We initially design a conceptual model to map real-time system task scheduling with traffic system scheduling which is visualized in manuscript 6. Further, We explore the possibility of implementing an adaptive real-time traffic scheduling algorithm in VANET environment that can improve traffic intersection performance by optimizing travel time index (TTI) which is defined as the probability of vehicle reaching its destination within a given time (Chapter: Adaptive Traffic Signal Control Using Travel Time Reliability for Vehicular Ad hoc Networks, manuscript 2).

RQ3 ***How to assign different level of priorities to emergency vehicles in arterial road network to meet their respective target response time?***

Assigning absolute priority to an individual EV serving any emergency level increases the overall waiting times for normal traffic. This problem can be solved if multiple EVs serving within a particular time are assigned with different priority levels, and Emergency Vehicle Pre-emption (EVP) is performed accordingly. We explore the opportunity to implement the EVP algorithm in a connected environment that can reduce the undesirable waiting time to normal traffic, but

still ascertain EVs meet target response time (Chapter: Routing Emergency Vehicles in Arterial Road Networks using Real-time Mixed-Criticality Systems, manuscript 3).

RQ4 ***How to implement infrastructure-free traffic lights for Cooperative vehicular technology to further enhance the limitation posed by current physical traffic signals studied in RQ3?***

Current traffic control infrastructure still works with the principle of optimized cycle time. With the advancement of wireless communication technology, more and more vehicles are being equipped with communication devices. This brings an opportunity for implementation of an infrastructure-free self-organizing traffic control system that eliminates unnecessary waiting at dead periods and makes the entire system adaptive. We investigate a futuristic system for autonomous vehicles that optimizes the timing in traffic phases and minimizes human-related loss like higher headway times and inconsistent inter-vehicle spacing when following each other. The system can expedite EVs movement through intersections and impose minimal waiting time for ordinary vehicles (Chapter: Dynamic Prioritization of Emergency Vehicles For Self-Organizing Traffic using VTL+EV, manuscript 4).

RQ5 ***How to implement a model that learns to estimate travel time accurately from an origin to the destination?***

Precise estimation of vehicle travel time is crucial in route planning and navigation (Y. Li, Deng, Demiryurek, Shahabi & Ravada, 2015), congestion detecting (Y.-y. Chen, Lv, Li & Wang, 2016), ride sharing (Asghari, Deng, Shahabi, Demiryurek & Li, 2016) and logistics (N. J. Yuan, Zheng, Zhang & Xie, 2012). In the case of EVs, travel time becomes the primary factor in assigning the priority level so that any case of emergency is responded within the allowable time frame. Estimation of travel time is a complex problem and is dependent

on multiple parameters. Mathematically created models based on the expertise and understanding of the problem can be error-prone. We explore the possibility of a machine learning model that can accurately estimate the travel time for a queried path that includes the origin and destination location (Chapter: Travel Time Estimation with Decision-Tree-Based Ensemble Networks and Enhanced Feature Extraction, manuscript 5).

1.4 Methodology

The thesis deals with research questions listed in section 1.3 by developing five methods:

- adaptive traffic control system
- assigning different levels of priority to EVs for adaptive traffic control system
- self-organizing traffic through virtual traffic lights
- Ensemble-based learning algorithm for travel time estimation
- analogical mapping of traffic scheduling with real-time system analogy

1.4.1 Traffic scheduling using real-time system analogy

The concept of resource allocation and meeting the timing constraint make EV routing analogous to task scheduling in the real-time system (RTS). Also, emergencies with several criticality levels with different service times make the EV routing problem very close to scheduling in a mixed-criticality real-time system (MCRTS). MCRTSs have tasks with two or more criticality levels, for example, non-critical, safety-critical and mission-critical. In MCRTS timing parameters like worst-case execution time (WCET) for processes rely mainly on criticality levels. For detailed conceptual model, refer

to chapter: Routing Autonomous Emergency Vehicles in Smart Cities Using Real-Time Systems Analogy: A Conceptual Model, manuscript 6. The concept has been implemented in manuscript 2, 3 and 4 that are presented in chapters Adaptive Traffic Signal Control Using Trave Time Reliability for Vehicular Ad hoc Networks, Routing Emergency Vehicles in Arterial Road Networks using Real-time Mixed-Criticality Systems and Dynamic Prioritization of Emergency Vehicles For Self-Organizing Traffic using VTL+EV respectively.

1.4.2 Adaptive traffic control system

The methodology of designing adaptive traffic control system tries to minimize travel time by optimizing objective expressed in terms of Buffer Index (BI), a typical travel time reliability measure, as the primary performance parameter in a connected environment. A detailed illustration is provided in chapter: Adaptive Traffic Signal Control Using Travel Time Reliability for Vehicular Ad hoc Networks, manuscript 2.

1.4.3 Assigning different levels of priority to EVs for adaptive traffic control system

We introduce mixed-criticality real-time system scheduling concept where a different level of emergencies is mapped with different criticality levels and assign certain success assurance level to respective criticality. We implemented the EVP algorithm for an arterial traffic network and leveraged valuable information transmitted via VANET to make critical decisions. Detailed implementation with empirical results can be visualized in chapter: Routing Emergency Vehicles in Arterial Road Networks using Real-time Mixed-Criticality Systems, manuscript 3.

1.4.4 Self-organizing traffic through virtual traffic lights

We design and implement self coordinating traffic system in a simulation environment to measure EV prioritization effectiveness. The model highlights on virtual traffic lights where vehicles communicate with themselves to pass through intersections. They organize within themselves manoeuvring different actions to prioritize any EVs within the vicinity of the intersection. Detailed implementation is visualized in chapter: Dynamic Prioritization of Emergency Vehicles For Self-Organizing Traffic using VTL+EV, manuscript 4.

1.4.5 Ensemble-based learning algorithm for travel time estimation

We design the ensemble-based machine learning model to estimate travel time for user queried origin to destination. We employ different feature engineering techniques to extract valuable features from raw GPS trajectory data and implement gradient boost regression tree to learn to estimate travel time. Detailed process can be referred in chapter: Travel Time Estimation with Decision-Tree-Based Ensemble Networks and Enhanced Feature Extraction, manuscript 5.

1.5 Contributions

Each manuscript included in this thesis have multiple contributions and can be outlined from the manuscripts. The overall contribution of each manuscript towards the research questions outlined in section 1.3 is illustrated below:

RQ1: What optimization and pre-emption techniques are employed in EV routing?

Our manuscript, "*A systematic review of route optimization and pre-emption methods*

for emergency vehicles" observes the limitations of existing routing systems and lack of real-world applications of the proposed pre-emption systems, leading to several interesting and important knowledge and implementation gaps that require further investigation. These gaps include optimizations using real-time dynamic traffic data, considering time to travel as a critical parameter within dynamic route planning algorithms, considering advanced algorithms, assessing and minimizing the effects of EV routing on other traffic, and addressing safety concerns in traffic networks containing multiple EVs at the same time (Humagain et al., 2020).

RQ2: How to design a real-time adaptive traffic control system to improve the performance of signalized traffic intersections?

We design and implement Fair Scheduling Algorithm (FSA), an adaptive traffic control algorithm to minimize average stretch in manuscript 2 named "*Adaptive Traffic Signal Control Using Travel Time Reliability for Vehicular Ad hoc Networks*". In this paper, we propose an adaptive Fair Scheduling Algorithm for VANETs that ascertains higher travel time reliability by minimizing TTI for an isolated intersection. We consider Buffer Index (BI), a typical travel time reliability measure, as the primary performance parameter to minimize. We achieve this through a novel approach of analogically mapping traffic control problems into real-time systems precisely mapping TTI with stretch (the factor by which a job is slowed down comparing with time it takes to process on a free system). We first prove that stretch produced by FSA is less than or equal to twice the stretch produced by an optimal offline algorithm implying FSA is *2-competitive*. Then we empirically prove that FSA online algorithm is more reliable and fairer in scheduling in terms of travel time compared to existing state-of-art approaches.

RQ3: How to assign different level of priorities to emergency vehicles in the arterial road network to meet their respective target response time?

We develop an emergency vehicle pre-emption technique with different levels of priorities. We leverage VANET to transmit critical information like current position,

speed, time EV takes to pass the intersection, and the route the EV follows in deciding when to trigger the EVP. Important decision parameters like speed, position, and vehicle route are accessed using VANET, which is impossible from the traditionally used inductive loop. It provides a way to arbitrate multiple EVs in an arterial road network. Performance parameters like waiting times and throughput achieved from such implementations are realistic and are helpful for emergency management services (EMS) to evaluate their actual performance. It also helps traffic engineers and transport planners anticipate the widespread effect of EVP over general traffic flows rather than limiting such explorations to a single intersection. Moreover, adding multiple EVs into the EVP model can solve unsolved problems faced by EMS in prioritizing EVs when two or more of them have conflicts passing an intersection. Manuscript 3, titled "Routing Emergency Vehicles in Arterial Road Networks using Real-time Mixed-Criticality Systems" enumerates several other significant contributions this approach can have over traffic planners and researchers.

RQ4: How to implement infrastructure-free traffic lights for Cooperative vehicular technology to enhance further the limitation posed by current physical traffic signals studied in RQ3?

Manuscript 4 titled "Dynamic Prioritization of Emergency Vehicles For Self-Organizing Traffic using VTL+EV" focus on the implementation of self-organizing traffic system. We implemented a decentralized self coordinating traffic system to prioritize emergency vehicle movement through an isolated traffic intersection. The proposed Virtual traffic lights plus for emergency vehicles (VTL+EV) algorithm for intersection control eliminates the loss generated from dead periods in a traffic light cycle time and human-related factors like increased headway time and inconsistent inter-vehicle spacing.

RQ5: How to implement a model that learns to estimate travel time accurately from an origin to the destination?

Manuscript 5 titled "*Travel Time Estimation with Decision-Tree-Based Ensemble Networks and Enhanced Feature Extraction*" uses gradient boosting that learns from multiple pre-engineered features related to topology, directions, the shape of the trajectory, path, speed limits, map distance traversed, real-time traffic conditions and compare it with existing deep neural network-based learning process using raw GPS trajectories. IT employs ensemble-based gradient boost regression tree method in origin-to-destination (O2D) travel time estimation using sole origin and destination GPS trace, making it pure O2D approach. For structured data, our approach learns better than existing deep neural approaches.

Chapter 2

Literature Review (Manuscript 1)

We conducted a systematic literature review (SLR) of route optimization and pre-emption methods for emergency vehicles. The primary research question in this study was "What optimization and pre-emption techniques from academic literature and industry can be used for effective EV routing?".

Different keywords related to route optimization, transit signal priority and road pre-emption for EVs were used for searching existing works. We experimented with several search strings, and the following is the trial search string used in this survey: (((“route optimization”) OR ((“Preemption” OR “pre-emption”) OR (“priority”)) AND “Emergency Vehicles”)).

The number of results found was recorded, and screening was undertaken on sections of search results checking relevance at the title, abstract and full-text levels. To make the literature search as inclusive as possible, we used relevant bibliographic databases like ITS Network and Transportation Research Board, web-based search engines like Google Scholar, Scopus, IEEE, Science Direct, ACM Digital Library and ISI Web of Science, as well as bibliographies of related reviews and directed calls for evidence using professional social networks like research gate and LinkedIn (Kitchenham, 2004).

A detailed classification of existing techniques is presented along with critical

analysis and discussion. The study observes the limitations of existing routing systems and lack of real-world applications of the proposed pre-emption systems, leading to several interesting and important knowledge and implementation gaps that require further investigation. These gaps include optimizations using real-time dynamic traffic data, considering time to travel as a critical parameter within dynamic route planning algorithms, considering advanced algorithms, assessing and minimizing the effects of EV routing on other traffic, and addressing safety concerns in traffic networks containing multiple EVs at the same time. Manuscript 1 is attached with this section.

A Systematic Review of Route Optimization and Pre-emption Methods for Emergency Vehicles

Subash Humagain[a*], *Roopak Sinha*[a], *Edmund Lai*[a] *and Prakash Ranjitkar*[b]

[a] *Information Technolog and Software Engineering, Auckland University of Technology Auckland, New Zealand*

[b] *Department of Civil and Environmental Engineering, University of Auckland, Auckland, New Zealand*

[*] *Contact details of the corresponding author*

AUT Tower, Level 7, 2-14 Wakefield Street, Auckland, New Zealand

Tel: +64-9-921 9999 Ext.6376; Email: subash.humagain@aut.ac.nz

2.1 Introduction

The response time of an emergency vehicle (EV) is the time interval from receiving an emergency call to the arrival of an EV to the emergency location. The response time depends primarily on the time it takes for an EV to travel to the location of an emergency. This travel time of an EV depends on several static parameters such as distance and number of intersections on the route (signalized/unsignalized), as well as dynamic parameters like flow, average speed, and number of stops. The presence of numerous such parameters makes reducing EV travel times complex and challenging.

Increased response time of EVs can have an irreparable loss of life and property. In medical emergencies such as cardiac arrest, every one-minute delay in response time causes mortality rate to increase by 1% and imposes additional USD 1542 in hospital costs, leading to 7 billion dollars increase in healthcare expenditure per year only in USA (RapidSOS, 2015). Other emergencies like building fires typically grow by 20% per minute causing an average USD 4000 of additional damages (RapidSOS, 2015). These factors have led to extensive research on reducing EV travel time (Fitch, 2005b).

Several studies have explored this problem, proposing several solutions, which

can be broadly classified into route optimization and pre-emption (Zhu et al., 2014). Optimization is the process of attaining the highest achievable performance under a given set of constraints by maximizing desired factors and minimizing undesired ones. Route optimization tries to choose the best route to achieve the minimum EV travel times. Route pre-emption, sometimes referred to as "traffic signal prioritization" or "transit signal priority", changes or alters traffic control in order to grant priority to special vehicles like EVs. The most commonly used tactic is manipulating traffic signals in the route of an EV, halting lower-priority traffic and providing right-of-way to the EV (Gedawy et al., 2008; Paniati & Amoni, 2006).

Reducing the response times of emergency services requires identifying the most effective optimization and pre-emption techniques, and factors for reducing EV travel times. This article presents a Systematic Literature Review (SLR) (Kitchenham, 2004) of the existing techniques proposed for route optimization and pre-emption for EVs as explained in Sections 2.3 and 2.4. We have reviewed 72 research articles out of which 85% are from 2006 - 2018, ensuring the relevance of this study to current research. For each study, we extract sufficient contextual and methodological details for individual analysis and comparison with others. This study can significantly help researchers pursuing research to improve EV travel times to understand the current state-of-the-art, and industry and government stakeholders looking to adopt better techniques for routing EVs. After describing the SLR method in Section 2.2, we survey route optimization and pre-emption techniques in Sections 2.3 and 2.4. Section 2.5 reviews methods employing both optimization and pre-emption. Section 2.6 provides knowledge gaps and Section 2.7 provides concluding remarks.

2.2 Systematic Literature Review Method

A systematic literature review (SLR) allows answering a clearly formulated question by appraising related research (Kitchenham, 2004). The primary research question in this study was “What optimization and pre-emption techniques from academic literature and industry can be used for effective EV routing?”

Different keywords related to route optimization, transit signal priority and road pre-emption for EVs were used for searching existing works. We experimented with several search strings and the following is the trial search string used in this survey: *((("route optimization") OR (("Preemption" OR "pre-emption”) OR ("priority")) AND "Emergency Vehicles"))*.

The number of results found was recorded, and screening was undertaken on sections of search results checking relevance at the title, abstract and full-text levels. To make the literature search as inclusive as possible, we used relevant bibliographic databases like ITS Network and Transportation Research Board, web-based search engines like Google Scholar, Scopus, IEEE, Science Direct, ACM Digital Library and ISI Web of Science, as well as bibliographies of related reviews and directed calls for evidence using professional social networks like research gate & LinkedIn (Kitchenham, 2004).

We excluded studies concentrating on optimization models in emergency logistics and optimization for multiple service stations during emergencies as this paper is primarily concentrated on reducing the response time of EVs focusing on travel time. We also excluded short papers (less than 4 pages), articles published in non-recognised journals, and those not written in English.

Our initial search retrieved 1069 studies, which were pruned to exclude papers meeting any of the exclusion criteria or those that were duplicated. In the end, 72 papers were chosen for a detailed review. Out of these, 25 articles propose route optimization, 34 report pre-emption based techniques, and the remaining 13 propose

systems employing both. Sections 3, 4 and 5 explore optimization, pre-emption and both techniques that aid in reducing the response time of EVs identified from the literature search.

2.3 Route Optimization Methods

A majority of Emergency Management Systems use dedicated software to locate, dispatch and route EVs. EVs contain installed software like Sygic and Infoware to guide the driver towards the emergency location using customized traffic information. Such software employ route optimization to ensure that EVs reach their destination in time (Togneri & Deriaz, 2013).

Route optimization treats traffic networks as graphs. Different cost functions like distances over available travel paths, travel times along sections of the network, and fuel efficiency are assigned as weights to the edges of the graph. Optimization algorithms then maximize or minimize the cost function (Winter, 2002). According to Eksioglu et al. (2009) route optimization is mainly distance (path)-dependent or travel time-dependent. Knowing the position of an EV is critical in suggesting optimized routes. A majority of studies employ global positioning system (GPS) technology. We use the vehicle routing problem (VRP) taxonomy defined by (Eksioglu et al., 2009) for computing transportation cost and to categorize the reviewed studies into path-based optimization and time-based optimization, and then critically analyze these studies.

2.3.1 Path-based Optimization

Intuitively, taking the shortest path between an EV's source and the intended destination can minimize travel time. In computer science, Dijkstra's shortest path algorithm is well-known for such optimization and several of the surveyed works extend this algorithm in some way. Kula, Tozanli and Tarakcio (2012) developed a stochastic shortest

path strategy by adding path-specific information such as hourly traffic behaviour and pedestrian involvement to a K-shortest path model which produce multiple paths with different travel times. Nordin, Zaharudin, Maasar and Nordin (2012) developed a deterministic shortest path calculator using an informed search algorithm called A* algorithm to find the shortest path among multiple points. Kai et al. (2014) and Winn (2014) used the geographical information system (GIS) network to determine the starting position of EVs and found the shortest path to the destination by calculating shortest distance from a starting node to every following node afterwards using Dijkstra's algorithm. Brady & Park (2016) also described a similar shortest path algorithm for routing EVs with additional road features like lane count, intersection control devices, and construction works.

Panahi and Delavar (2009) use Dijkstra's algorithm but their approach can intelligently update the proposed path during driving by integrating data from GIS and real-time traffic conditions. Shekar et al. (2012) and Elmandili et al. (2013) described Vehicular Ad Hoc Network (VANET) based navigation, which also used Dijkstra's algorithm for routing. Real-time traffic congestion was determined by GPS evaluating a larger cluster of slow moving nodes and suggesting a new route if needed. Nicoara and Haidu (2014), Winn (2014) and Sun et al. (2014) used GIS-based networks to find the shortest route access for EVs using real-time traffic data. Similarly, Fleischman et al. (2013) used GIS to estimate transport times for ambulances creating a linear regression model that increases the accuracy of these road network estimates using patient characteristics, use of lights and sirens, daylight, and rush-hour intervals.

The speed of calculation of the shortest path in Dijkstra's algorithm is dependent on the number of nodes available. Bu and Fang (2010) used improved Dijkstra's algorithm for solving the shortest path problem by focusing dynamically in a smaller area including the accident location and current EV location. As the EV approaches the accident location, the number of nodes involved in the calculation of the shortest path

gradually decreases, subsequently reducing the computation time.

2.3.2 Time-based Optimization

For EV routing, the actual time taken to reach the emergency location is more important than distance, cost, fuel consumption, etc. Cooke and Halsey (1966) were the first to provide theoretical insights on shortest path algorithms that can have variable travel times between vertices depending upon physical parameters. Hadas and Ceder (1996) implemented the shortest path algorithm to produce multiple time-based paths for EVs.

Zhu et al. (2014) developed an optimization model, which was based on both the shortest rescue time and the lowest rescue cost using a simulated annealing algorithm to find the global optimum of the objective functions to minimize rescue time and cost. X. Wang and Liu (2011) proposed an Internet of Things application using RFID tags in ambulances and wireless sensor nodes on the roads to collect real-time traffic data and forecasted path to provide the fastest route. Choosumrong et al. (2012) used the routing algorithm to calculate minimum travel time from the accident point to the nearest hospital using parameters like availability of beds and the patient's state. Apart from the above, Elalouf (2012) used exact pseudo-polynomial algorithm to find the optimal time-dependent route using real-time data for uncertain traffic conditions. The algorithm uses dynamic programming to simplify a complicated problem into simple sub-problems by breaking them. Finally, they improved the solution using an e-approximation algorithm by limiting results within allowable lower and higher bound of cost function.

Derekenaris et al. (2001) developed a solution to deploy an ambulance requiring the least time to reach the site of an incident. Like most other approaches, they used GPS to locate available ambulances and then employed Dijkstra's algorithm to calculate the shortest travel time between available ambulances and incident site. Vlad, Morel, Morel

and Vlad (2008) described a model that finds the fastest path for an EV to reach the site of an emergency by controlling traffic signals. It used a learning routing algorithm which reaches a decision with the help of a neural network that calculates expected time of arrival of every feasible route that EV may follow. Real-time traffic data collected from GPS equipment installed on EVs was used to train the neural network.

2.3.3 Other Optimization Methods

Some studies propose alternative approaches for routing EVs. Bura and Boryczka (2010) presented an ant-colony based vehicle navigation system that supports dynamic traffic conditions like traffic load and temporarily closed roads. Musolino et al. (2013a) dynamically designed routes taking into account travel time variations within a day for the same network. Barrachina et al. (2014) compared emergency services arrival time between density-based and non-density-based road networks. Similarly, Y.-z. Chen, Shen, Chen and Yang (2014) determined the optimized route using the Dijkstra's algorithm for different traffic conditions like morning peak, evening peak, and daytime. Afdhal and Elizar (2015) used Vehicle to Infrastructure (V2I) based communication that assisted in reducing EV travel times by increasing the average speed achieved from avoiding congested areas. Road side units sent congestion information to network simulator that instructs EVs to follow the route.

2.3.4 Discussion

Optimization techniques provide either the shortest or the fastest path for an EV to travel. Some of these studies are theoretical, some use simulation to validate an underlying theory, whilst a few are practically implemented. Table 2.1 provides a comparison of optimization techniques used in reducing travelling time of EVs. Column 1 lists the

Table 2.1: Analysis of EV route optimization method

References	Models	Type	Other Parameters / Techniques	TRL
Hadas & Ceder, 1996; Kula et al., 2012	K-Shortest Path	S	Accident place	TRL5
Nordin et al., 2012	A* Algorithm	D	Uses C # programing	TRL6
Kai et al., 2014; Winn, 2014	Dijkstra's Algorithm	D	Arc GIS network	TRL6
Brady & Park, 2016	Dijkstra's Algorithm (using GIS for position)	D	lane count, intersection control devices and median count	TRL5
Panahi & Delavar, 2009	Dijkstra's Algorithm (using GIS)	S	Dynamic, real-time traffic	TRL7
Elmandili et al., 2013; Smitha et al., 2012	Dijkstra's Algorithm	S	VANAT for real-time traffic, GPS for calculating position and congestion	TRL5
Bu & Fang, 2010	Improved Dijkstra's	S	Dynamic in terms of searching	TRL5
Nicoara & Haidu, 2014; Sun et al., 2014	Path Selection	D	Shortest route, real-time traffic data	TRL6
Fleischman et al., 2010	Linear regression	D	Transport time	TRL7
Zhu et al., 2014	Annealing Algorithms	D	Cost, Road resistance, distance	TRL4
Y. Wang et al., 2013	Internet of Things	S	RFID & Road side units collect real-time data	TRL6
Choosumrong et al., 2012	PgRouting Algorithm	D	Beds available, patient status	TRL5
Elalouf, 2012	Exact Pseudo-polynomial Algorithm	S	Dynamic/real-time data	TRL5
Derekenaris et al., 2001	Dijkstra's Algorithm	D	Locating ambulance location/ calculating travel time	TRL6
Vlad et al., 2008	Learning routing algorithm	D	Determine real-time traffic volume	TRL5
Bura & Boryczka, 2010	Ant colony static and dynamic optimization	S	Comparison of 3 version of Ant-based navigation	TRL6
Musolino et al., 2013	Taking 3 different level of congestion	S	Different route model	TRL6
Barrachina et al., 2014	Evolution Strategy	D	Comparison of density and non-density based network	TRL6
Chen et al., 2014	Dijkstra's	D	Optimal Route taking morning and evening peak	TRL5
Azmi & Mustafa, 2016	Network layer model	D	V2I based communication for avoiding congestion	TRL7

Note: *D represents deterministic and S represents stochastic.

studies and column 2 states the types of optimization model they have used. In terms of algorithms, Dijkstra's algorithm is the most popular choice but different algorithms like ant colony, linear regression, annealing, pseudo-polynomial etc. have also been implemented. Column 3 describes if a technique is deterministic (D) such that it provides the same output for the same inputs every time, or if the technique is stochastic (S) and involves some level of randomness or unpredictability. Optimization techniques from this aspect seem almost equally divided into deterministic and stochastic category. Column 4 provides details of other parameters and techniques used to achieve specific optimization. Studies listed in this section suggest the use of multiple optimization techniques with different level of implementation maturity. In conditions where studies suggest diverse approaches and solutions, it is difficult to reach a conclusion in finding the best optimization technique. Another problem with these studies is that they usually report simulation results obtained for a certain geographical location. Each location has distinct traffic parameters. These parameters play a vital role in the performance of past studies. Hence, a direct comparison in performance is not possible. So, in addition to the qualitative comparison based on columns 2-4, we have also further compared these works using the Technology Readiness Level (TRL) in column 5 that describes the progression of technologies (Mankins, 1995). It classifies technology maturation into nine different levels. Lower levels like TRL1 relate to new but untested technologies where only basic principles have been observed. Each subsequent level indicates a more mature technology, with TRL9 indicating a commercially produced technology.

The annealing algorithm by Zhu et al. (2014) for finding shortest rescue time and rescue cost for EVs was implemented using MATLAB. No traffic domain simulation and verification for these technologies have been performed. As these techniques have undergone only low-fidelity simulation testing, they are categorized as TRL4 (validated in a laboratory environment).

Most studies used well-known mathematical algorithmic implementations. More

comprehensive verification and validation were performed on these using simulation models, resulting in a slightly higher level of credibility. When such validation is done in relevant simulation environment, they belong to TRL5. Hence, works like K-shortest path by (Kula et al., 2012), pseudo-polynomial algorithm by Elalouf (2012), and Dijkstra's algorithm by Brady and Park (2016) etc. are classified as TRL5.

Works that involved prototype development, analysis using real-world test data and comprehensive system validation are categorized as TRL6 because they use well-known calibrated traffic simulation models. For example, path selection model by Nicoara and Haidu (2014), an optimization model using internet of things by Y. Wang et al. (2013), Dijkstra's algorithm by Derekenaris et al. (2001) etc. are included in this category.

When developed system prototypes are demonstrated in real environment, they belong to TRL7. Very few works involve system prototypes tested in the real world. These handful of works were deployed in real cities and were validated through verifiable results showing that optimization enhanced with carefully chosen recent techniques could significantly reduce travel time. Hence, works like linear regression by Fleischman et al. (2013), Dijkstra's algorithm by (Elmandili et al., 2013) etc. are classified under TRL7.

2.4 Pre-emption Techniques

Pre-emption involves allowing EVs to take priority at intersections. The most traditional and commonly used pre-emption system is the humble siren accompanied with flashing lights which are manually operated. Manual prioritizations such as police officers controlling intersections and use of sirens, sound, and flashing lights have drawbacks in determining the presence and direction of EV (Eltayeb et al., 2013). However, technology has evolved to provide automated and more effective pre-emption to allow EVs to travel even faster. Current pre-emption techniques carry out three major categories of tasks: defining when to activate pre-emption, determining the position of an EV

using sensing techniques to trigger pre-emption, and using communication techniques between an EV and the infrastructure/other vehicles to execute pre-emption. We have categorized pre-emption works based on these tasks.

2.4.1 Activation

Defining when to activate pre-emption is needed for the efficient execution of pre-emption. It usually depends on either pre-emption is carried out for a particular traffic signal or for an entire route. Jones, Judge, Beck and Keegan (1999) , Jordan and Cetin (2015), Kodire, Bhaskaran and Vishwas (2016), Barthwal and Menghani (2017), and A. Goel, Ray and Chandra (2012) decompose road network into zones. The presence of an EV in a particular zone was identified via installed GPS and then pre-emption was done making related traffic signal go green. Similarly Y.-S. Huang, Shiue and Luo (2015), Kotani, Yamazaki and Jinno (2011) and Y. Wang et al. (2013) also used GPS to locate EV but pre-emption was activated when the EV was approaching a traffic signal.

A few studies like Hegde, Sali and Indira (2013), Iyyappan and Nandagopal (2013), Nellore and Hancke (2016), Noori, Fu and Shiravi (2016), Unibaso, Del Ser, Gil-Lopez and Molinete (2010), Van Gulik and Vlacic (2002), Weng, Huang, Su and Yu (2011), and Xie et al. (2017) set a fixed distance between a traffic signal and an EV to trigger pre-emption. The distance was measured using technologies like GPS, Infrared, VANET and RFID.

Another approach of activating pre-emption is to create a green wave for the entire route of EV. Initially, direction, lane, and route of EV responding to emergencies are defined and all the traffic lights located in that route are allowed to go green. Agarwal and Paruchuri (2016), Bachelder (2011), Chowdhury (2016), (Idris, Sivalingam, Tamil, Razak & Noor, 2013), Kamalanathsharma and Hancock (2012), Kang et al.

(2014), Moroi and Takami (2015), and Yoo et al. (2010) generate a green wave for the entire lane on which EV is travelling.

Unlike above mentioned approaches, Bhavani, Vishwasri and Chandrakala (2016) and Jayaraj and Hemanath (2015) focused on implemented pre-emption techniques for a single intersection. They used RFID to count the vehicle queue approaching an intersection to estimate the time to activate pre-emption. Similarly, Qin and Khan (2012) developed control strategies for a traffic light when an EV is passing through an intersection.

2.4.2 Sensing

Once pre-emption approach is fixed, it is fundamental to identify the location of EV. Multiple sensors, both vehicle-mounted or spread over the traffic network, support in identifying the exact position of EV. An analysis of studies reveals the use of GPS as technology of choice in determining position of EV to activate pre-emption. Barthwal and Menghani (2017), Idris et al. (2013), Iyyappan and Nandagopal (2013), Jones et al. (1999), Jordan and Cetin (2015), Kodire et al. (2016), Unibaso et al. (2010), and Y. Wang et al. (2013) all use GPS for sensing EV locations. In addition to employing GPS, Hegde et al. (2013) and Van Gulik and Vlacic (2002) used RFID tags for the determination of emergency case.

Vehicular ad hoc networks (VANETs) are the connected environments which support vehicle-to-X (V2X: vehicle, road, human, infrastructure, internet) communication through multiple communication protocols. In a connected environment, it is easier to locate the position of an EV as they are sharing location information among themselves too often. Agarwal and Paruchuri (2016), Jayaraj and Hemanath (2015), Kamalanathsharma and Hancock (2012) and Noori et al. (2016) use VANETs to determine the position of EV which considers EVs and roadside units as nodes of a

vehicular network. Nellore and Hancke (2016) used a visual sensing technique using cameras to determine the position of EV in VANET.

Some studies like Bhavani et al. (2016) and A. Goel et al. (2012) employed ZigBee technology to ascertain the progression direction of EV. As they created a green wave for the entire corridor knowing direction was sufficient to activate pre-emption.

2.4.3 Communication Approach

Pre-emption may be actuated through vehicle-mounted devices or remotely via emergency control centers (Kamalanathsharma & Hancock, 2012). In remote actuation, emergency dispatchers determine factors like the level of emergency and monitor the EVs location to activate pre-emption over the upcoming intersection. Vehicle mounted pre-emption devices are usually integrated with the vehicle's warning lights and can be switched on and off when needed. Current technologies use acoustic sensors, localized radio sensors, GPS or line-of-sight sensors to activate pre-emption. An additional light near the traffic lights known as a confirmation beacon notifies other traffic that the traffic light is under the influence of pre-emption and warns other drivers about the EV's approach (Y.-S. Huang et al., 2015). In some countries, a flashing confirmation beacon indicates to a vehicle that an EV is approaching from an opposing direction (front or side) while solid light indicates that the EV is behind the vehicle (Y.-S. Huang et al., 2015).

OPTICOM, EMTRAC, and Transmax are EV pre-emption products that are currently used in different cities of USA, UK, Canada and Australia (Global Traffic Technologies, 2016). All these commercial systems use infrared and GPS technology. Once approaching to traffic signals, EVs automatically send requests for pre-emption. The traffic signals wirelessly receive and authenticate the requests to provide green

lights to the EVs (Viriyasitavat & Tonguz, 2012). Japan uses the FAST vehicle pre-emption system (Kotani et al., 2011). It consists of a device mounted on EVs and overhead infrared beacons installed along roads. When EVs travel past these beacons, the traffic control system activates pre-emption at upcoming intersection. When EVs are located near to upcoming traffic lights the green time is extended whereas red time is reduced when EVs are far.

VANETs treat moving vehicles and roadside units as nodes of a mobile network, within a range of 300 meters (Pighin & Fierens, 2015). This technology has been prominently used for communication between different entities involved in route pre-emption. Agarwal and Paruchuri (2016), Jayaraj and Hemanath (2015), Jordan and Cetin (2015) , Moroi and Takami (2015), Nellore and Hancke (2016), Noori et al. (2016), Pighin and Fierens (2015), and Unibaso et al. (2010) all implemented VANET as communication tool to execute pre-emption effectively. Y. Wang et al. (2013) and Xie et al. (2017) used dedicated short-range communication broadcast to activate pre-emption.

Vehicle mounted transceivers send and receive signals to communicate with entities responsible to execute pre-emption. Jones et al. (1999) and Van Gulik and Vlacic (2002) used radio antenna to communicate with traffic control system. Hegde et al. (2013), Kodire et al. (2016) and A. Goel et al. (2012) used ZigBee transceivers to communicate and accomplish pre-emption. Iyyappan and Nandagopal (2013), Kotani et al. (2011) and Weng et al. (2011) all used some form of transceivers to communicate with traffic signals to actuate pre-emption.

In the case where emergency control center activates pre-emption EVs send messages using GSM cellular technology to activate pre-emption. Bachelder (2011), Barthwal and Menghani (2017), and Bhavani et al. (2016) used GSM technology for sending messages to control center and request pre-emption.

Apart from above mentioned studies, few other algorithms can help EVs to get priority at intersections. Asaduzzaman and Vidyasankar (2017) proposed an algorithm

to control traffic signal that can adjust time space of traffic phases to assist high priority vehicles. Qin and Khan (2012) developed control strategies for traffic signals to expedite the movement of EVs and avoid accidents. Viriyasitavat and Tonguz (2012)developed virtual traffic lights that can prioritize the movement of EVs.

2.4.4 Discussion

Table 2.2 presents a comparison of pre-emption techniques. Column 2 describes types of pre-emption as active (A) or passive (P). In active pre-emption, a pre-emption signal is adjusted as EV approaches an intersection. An active system can be a combination of real or fixed-time control strategies, and scheduled or headway-based strategies (Chada & Newland, 2002). For each work, it provides details on the pre-emption technique, how pre-emption is initiated, and how pre-emption works. Column 3 describes each work on basis of the control strategy (C.S.) used, like fixed time (FT), real-time (RT), schedule-based (SB) and headway-based (HB). Column 4 describes the concept and equipment used for sensing the presence of EVs. Column 5 “Initiating Pre-emption” notes process of initiating pre-emption and column 6 “Methods” lists how pre-emption is implemented.

Most pre-emption techniques are real-time control models that rely on constantly updated information regarding route and traffic network to make decisions. A real-time control model is flexible to changing conditions. Some studies also apply a signal control plan for a fixed time based on the conditions of a particular area like its congestion and vehicular flow. Fixed time control does not receive constantly updated road information and the best control scheme is applied regardless of actual traffic conditions.

A few studies use pre-emption control based on the schedule of an EV’s arrival. In such cases, the proper location of the EV is not known and most of the time this requires less communication equipment making this a more cost-effective (Bachelder,

2011; Idris et al., 2013; Iyyappan & Nandagopal, 2013; Kamalanathsharma & Hancock, 2012; Kang et al., 2014). In a headway-based control scheme used by one of the studies we reviewed, pre-emption is activated so that EVs can lead other vehicles heading in the same direction as it is effective in reducing waiting times (Pighin & Fierens, 2015). In such techniques, sometimes there is a possibility of EVs colliding with other vehicles.

Table 2.2: Analysis of EV pre-emption technique

References	**P**	**C.S**	**Concept / Equipment**	**Initiating Pre-emption**	**Method**
Jones et al., 1999	A	FT	GPS for position	EV in defined area	Green Light
Idris et al., 2013	A	SB	GPS for Position	All networks within route	Green Light
Kodire et al., 2017	A	RT	GPS for position	Sends signal using ZigBee	Green Light
Hegde et al., 2013; Van Gulik & Vlacic, 2002; Yoo et al., 2010	A	RT	GPS for position and RFID congestion	Congestion level	Green Light
Unibaso et al., 2010	A	RT	GPS &VANET	CAM message	Green Light
Bycraft, 2000	A	FT	Finalizing route first	Communicative Sensors	Green Light
Kamalanathsharma & Hancock, 2012	A	SB	Real-time traffic	Offset value	Green Light
Kotani & Yamazaki, 2011; Weng et al., 2011	A	RT	Infrared	Crossing of IR	Green Light
Iyyappan & Nandagopal, 2013	A	SB	Sensors and GPS for accident location	All networks within route	Green Light
Siddiqa & Shakeel, 2014	A	FT	ZigBee &GPS	Must reach a defined area	Green Light
Bhavani et al., 2016	A	FT	GPS	Detects EV using RFID tag	Green Light
Wang et al., 2013	A	RT	VANET	Congestion level	Green Light
Jordan & Cetin, 2015; Moroi & Takami, 2015	P	-	VANET	Alerting another vehicle	lane allocation
Pighin & Fierens, 2016	A	HB	VANET	EV Arrival	Green Light

References	P	C.S	Concept / Equipment	Initiating Pre-emption	Method
Noori, 2013; Noori et al., 2016	A	RT	VANET	Queue length	Green Light
Jordan & Cetin, 2014	A	RT	VANET	Congestion level	Change lane
Jayaraj & Hemanth, 2015	A	FT	VANET	Lane reservation	Green Light
Agarwal & Paruchuri, 2016	P	-	VANET	Fixed lane	lane allocation
Barthwal & Menghani, 2017	A	FT	M2M communication	Traffic flow controlling	Informs other
Bachelder, 2011	A	SB	GSM	Inter network communication	Green Light
Huang et al., 2015	A	RT	Traffic Control	Time Petri nets	Green Light
Kang et al., 2014	A	SB	Traffic Signal Control	Green wave for EV	Green Light
Nellore & Hancke, 2016; Qin & Khan, 2012; Viriyasitavat & Tonguz, 2012; Xie et al., 2017	A	RT	Distance between EV and network	Sensing, distance and presence of EV	Green Light
Chowdhury, 2016; Wang et al., 2013	A	FT	IOT	Type of incident	Green Light
Asaduzzaman & Vidyasankar, 2018	A	FT	Algorithmic control for traffic signal	Request from EV	Green Light

Some other studies invoke pre-emption for an EV's entire route passively. Our literature search indicates that passive priority systems that use fixed-time control strategies are rarely used though they have the benefit of being lower in cost.

2.5 Techniques employing both optimization and pre-emption

Studies combining both optimization and pre-emption first find an optimal route calculated using distance or time as the critical parameter and then use pre-emption on the selected route. W. Huang et al. (2011) and Eltayeb et al. (2013) suggest the shortest path and clear the path in advance from other vehicles and pedestrians by identifying the position of the EV using GPS. Chakraborty et al. (2015) assign a green signal when an EV is present near an intersection, measuring the queue length of traffic from a network. Similarly, Kwon and Kim (2003) and Djahel et al. (2015) propose an optimized route based on congestion level, then assign priority operations such as change in traffic signal, change in speed limit, lane clearance, using the reverse lane, re-routing to another route etc. Mirchandani and Lucas (2004) use existing transponders in EVs to pre-empt signals towards a destination. Shirani, Hendessi, Montazeri and Zefreh (2008) use packet signal with velocity information sent by one vehicle to another in VANET for finding the shortest path and pre-empt the particular path.

Gedawy et al. (2008) take real-time updates of congestion and other delays in travel time to plan optimal paths using GPS and then proposed a traffic signal pre-emption whereas C.-Y. Chen et al. (2013) use lane reservation strategy after suggesting an optimized route suggested from historical data. Polineni, Ravi Kumar and Ravi Kumar (2015) and Salehinejad, Pouladi and Talebi (2011) suggest shortest path algorithms and activate pre-emption. They use GPS for locating vehicle and a control center to activate green lights. Anand and Flora (2014) use tilt and vibration sensors to detect accidents and the GPS system gives the location. The server sends stored shortest route to ambulances. The traffic signal is controlled to give way to the ambulances using zombie protocol.

2.5.1 Discussion

Use of both optimization and pre-emption is a more practical approach in reducing EV travel times. Table 2.3 compares works in this category. Column 2 uses TRL to compare and classify the optimization technique implemented by these studies. Column 3 provides details of other parameters to achieve these optimizations and column 4 characterize if optimization technique is deterministic (D) or stochastic (S). As an intuitive concept, EV routing is more inclined towards prioritization and most of the studies we have reviewed in this section focus more on pre-emption than optimization. Column 5 categorizes pre-emption as active (A) or passive (P). Column 6 describes each work in terms of control strategy as discussed in section 4.4. Similarly, Column 7 explains the concept and equipment used for sensing the presence of EV. Column 8 lists how pre-emption is initiated and column 9 lists how pre-emption is implemented.

An ideal approach for achieving better optimization and pre-emption could be combining the best of techniques, from each category, as discussed in Section 3 and Section 4. Implementing both techniques requires a lot of resources. In general thought, techniques using both optimization and pre-emption employ more mature models in terms of implementation than techniques discussed earlier.

W. Huang et al. (2011) and Djahel et al. (2015) have developed a mathematical optimization model. Here verification and validation were simulation based so these studies are categorized as TRL5. Optimization techniques used by Eltayeb et al. (2013), Gedawy et al. (2008) , and Mirchandani and Lucas (2004) are summarized under TRL6 as they are able to adopt a few mature models developed by other researchers to implement the system. These models are simulation models with relevant environment verification. Some other works like Anand and Flora (2014), Chakraborty et al. (2015), C.-Y. Chen et al. (2013), Kwon and Kim (2003), Moraali (2011), Polineni et al. (2015), Salehinejad et al. (2011), and Shirani et al. (2008) develop or borrow optimization models that are

used in real traffic conditions, so we have grouped them under TRL7.

W. Huang et al. (2011) use pre-emption for the entire route and this is classified as a passive technique, as it allows traffic signals to go green for a fixed time. All remaining studies actively adjust the pre-emption signal once the EV approaches specific intersections. Most studies use real-time updated traffic information to decide the duration of the pre-emption signal. In contrast, Anand and Flora (2014) activate pre-emption signal as EV arrives at the traffic network but operates the signal for a fixed duration. Only C.-Y. Chen et al. (2013) employ active lane reservation for prioritizing the EVs.

Table 2.3: Analysis of technique employing both optimization and pre-emption

References	Optimization			Pre-emption				
	TRL	Other Parameters/ Technique	Type	P	C.S	Concept/ Equipment	Initiating Pre-emption	Method
(Eltayeb et al., 2013)	TRL6	GPS	D	A	RT	GPS GSM	Distance from Network	Green light
(Gedawy, 2010)	TRL6	GPS	S	A	RT	Heuristic speed	Expected Travel time	Green Light
(Huang et al., 2011)	TRL5	Historic Data	S	P	FT	Preset Route control	Entire Path	Green Light
(Chakraborty et al., 2015)	TRL7	Real traffic	S	A	RT	Queue length	Distance from Network	Green Light
(Kwon & Kim, 2003)	TRL7	Dijkstra's	D	A	RT	GPS	Location of EV	Green light
(Djahel et al., 2015)	TRL5	Historic Data	D	A	RT	choosing response Plan	Emergency, Congestion Level	Light change, lane clearence, use reserve lane
(Mirchandani & Lucas, 2010)	TRL6	Map	D	A	RT	Adaptive signal control	Real-time traffic flow	Green Light
(Shirani et al., 2008)	TRL7	GPS	D	-	RT	VANET	Speed	-
(Chen et al., 2013)	TRL7	GPS	D	A	_	VANET,GPS	Road Condition	lane reservation
(Moraali, 2011; olineni et al., 2015)	TRL7	A*algorithm	D	A	RT	GPS, GSM	Distance Threshold	Green Light
(Salehinejad et al., 2011)	TRL7	Ant Colony	S	A	RT	GPS, Fuzzy value	Pheromone level	Green light
(Anand & Flora, 2014)	TRL7	GPS	D	A	FT	GPS GS	Distance to network	Green Light

2.6 Gaps

This section describes the gaps in existing optimization and pre-emption techniques. We have categorized these gaps into implementation gaps and knowledge gaps. Implementation gaps focus on the practical difficulties in implementing advanced routing and pre-emption techniques and knowledge gaps include the analysis of research gaps present in these state-of-the-art technologies.

2.6.1 Implementation Gaps

As most of the studies are research based, the feasibility of implementing them in the real-world situation seems unclear. There are practical difficulties in the implementation of advanced routing and pre-emption techniques. A few of these gaps are discussed below:

1. **Adoption of academic research:** Researches develop efficient route optimization and pre-emption algorithms that can produce exceptional solutions but commercial EV routing software does not use these state-of-the-art technologies, commercial systems rather rely on simpler heuristics. This is because not all academic results can be engineered into effective systems. For industries, it is more efficient to develop simple optimization systems that fit a variety of problems like courier, logistic and trucking and give comparable results rather than to develop a complex solution for a specific EV routing problem. So, it is practical to assume that the implementation of advanced optimization and pre-emption will be gradual in nature (Pillac, 2012).

2. **Limited computation resources:** Most of the literature suggest on real-time optimization that demand computationally expensive resources and large computational time. In the case of EV routing, industry is not too much interested

in investing more on computing resources as it is a niche market with very few users (Winter, 2002). This eventually restrict number of real-time dynamic parameters to be considered during optimization.

3. **Handling multiple vehicle:** Current pre-emption systems are activated either by vehicle-mounted devices or traffic control system. There exists a problem in assigning priority when two vehicles request for pre-emption at the same time. As real-time dynamic optimization and pre-emption system will be completely automatic, there will be ambiguity in providing preference to EVs for pre-emption.

4. **Lack of real-time validation:**For the implementation of real-time dynamic optimization most of the required stochastic and real-time information will be available from different connected sensors and IOT networking which do not exist now. Though the relevance of dynamic real-time optimization for EVs has been documented well, there are always issues in comparing results from these approaches. Since these studies use artificial data created by researchers themselves, based on real world applications, most of the results are predictive (Pillac, 2012).

2.6.2 Knowledge Gaps

Our survey shows research in this domain has adopted the use of one or both of optimization and pre-emption for routing EVs. None of the studies provide any comparative evidence that a particular optimization technique and pre-emption system is better suited to solve the problem of reducing the travelling time of the EV. We can also conclude that although much research has been conducted for reducing the response time of EVs, there has not been a considerable decrease in response time (Moemi, Isong & Jonathan, 2017). This indicates a potential “dead-end” in the way research has approached the issue of reducing EV travel times and signals a need to explore newer

methods to approach this problem. Out of the 72 papers we reviewed, 35 have provided future research directions. Detailed analysis of these current research gaps can guide us towards solving the dead-end that research has produced in reducing EV travel times. Future research should focus on the following directions listed below:

1. **Real-Time dynamic traffic Data:** Research should focus on integrating real-time on-road traffic data to calculate more dynamic, reliable and accurate routes to EVs (Bhavani et al., 2016; Elmandili et al., 2013; Fleischman et al., 2013; W. Huang et al., 2011; Kai et al., 2014; Musolino et al., 2013a; Nicoara & Haidu, 2014; Winn, 2014).

2. **Time as a critical parameter:**Finding the shortest path is not enough to improve emergency response system in a complex road network as minimum travel time is a major parameter to consider (Barrachina et al., 2014; Choosumrong et al., 2012; Mali et al., 2012).

3. **Advanced algorithms:** Basic graph theory method and mathematical programming method cannot meet the calculation requirement of real-time traffic (Brady & Park, 2016; Chakraborty et al., 2015; Elalouf, 2012; Sun et al., 2014).

4. **Use of VANET:** With the advancement of the wireless communication technologies like Cooperative Vehicle-Infrastructure System (CVIS), there is an opportunity to provide appropriate traffic signal pre-emption for emergency vehicle based on real-time emergency vehicle data, traffic volume data, and traffic signal timings (Agarwal & Paruchuri, 2016; Anand & Flora, 2014; Djahel et al., 2015; Jayaraj & Hemanath, 2015; Y. Wang et al., 2013).

5. **Concerns with multiple EVs:** Future studies can include considerations of more severe scenarios, such as disasters where a large number of EVs are required (C.-Y. Chen et al., 2013; Chowdhury, 2016; Moroi & Takami, 2015; Pighin & Fierens,

2015).

6. **Safety of EV travel:** It is a challenge to ensure safe passage of an emergency vehicle (EV) or multiple EVs and at the same time to maintain safe and smooth traffic flow in the road network (Qin & Khan, 2012; Yoo et al., 2010).

7. **Intelligent pre-emption:** Limited research has been done on the use of intelligent pre-emption control, which has the ability to use real-time traffic information to minimize emergency vehicle delays. At the same time, reducing the adverse impacts of emergency vehicles on normal traffic, so that they can cause the least disturbance to network traffic flow is a challenge (Djahel et al., 2015; Kamalanathsharma & Hancock, 2012; Kang et al., 2014; Nellore & Hancke, 2016; A. Goel et al., 2012; Unibaso et al., 2010; X. Wang & Liu, 2011; Y. Wang et al., 2013).

A critical analysis of optimization and pre-emption suggests that there is difference between actual travel time and theoretically calculated travel time. This difference arises as dynamic parameters like increased congestion, halt on a road, pedestrian flow, queued vehicles, real and adaptive speed are not being addressed within the theoretical models. Similarly, pre-emption is also not effective, as oftentimes the timing of activation in implemented systems is not precise and pre-emption techniques often do not consider the effect of pre-emption over other vehicles.

An appropriate solution to improve existing techniques will require dynamic optimization and efficient and precise pre-emption so as to cause minimal disruption to other vehicles. The success of such combined and dynamic optimization and pre-emption systems depends on the availability of real-time dynamic traffic data. This means that sensors deployed at various infrastructures of road network must communicate in real-time and support real-time decision-making. In general, traffic infrastructure requires a deeper integration with software systems to ensure high availability of accurate

real-time data.

2.7 Concluding Remarks

This review has described and compared techniques used in reducing response time of EVs. The optimization and pre-emption can provide a solution for reducing response time however they need many further improvements. It has been suggested in this paper that researchers on emergency management services must focus on making optimization more dynamic by using real-time dynamic traffic data and taking time as a critical optimization parameter. They also need to work on making pre-emption intelligent and use advanced technologies like VANET. Such pre-emptive solutions need to ensure it creates the minimal effect on other traffic. Further research should bring most advanced optimization and pre-emption together. This, in turn, will solve the challenging job of reducing response time.

Chapter 3

Adaptive Traffic Signal Control Using Travel Time Reliability for Vehicular Ad hoc Networks (Manuscript 2)

The paper "Adaptive Traffic Signal Control Using Travel Time Reliability for Vehicular Ad hoc Networks." proposes adaptive traffic control (ATC) algorithm for isolated traffic intersections implemented in Vehicular Ad Hoc Networks (VANETs). Current ATC algorithms optimize the objective derived in terms of throughput, waiting time, queue length, overall travel time, fuel consumption and gas emission. Though some of these objectives indirectly try to optimize average travel time, travel time reliability from the user's perspective is not explicitly considered. The probability of vehicle reaching its destination from the point of origin within a given time is termed as travel time index (TTI) measured in terms of reliability. Travel time reliability indicates the variability in delays due to stochastic demand in traffic, stochastic traffic at intersections, mid-link disturbances like stopping of vehicles, pedestrian crossing, asynchronous traffic signal control with traffic demand and weather conditions.

We propose an adaptive Fair Scheduling Algorithm (FSA) for VANETs that ascertains higher travel time reliability by minimizing TTI for an isolated intersection. We consider Buffer Index (BI), a typical travel time reliability measure, as the primary performance parameter to minimize. We achieve this through a novel approach of analogically mapping traffic control problems into real-time systems precisely mapping TTI with stretch (the factor by which a job is slowed down comparing with time it takes to process on a free system). We first prove that stretch produced by FSA is less than or equal to twice the stretch produced by an optimal offline algorithm implying FSA is *2-competitive*. Then we empirically prove that FSA online algorithm is more reliable and fairer in scheduling in terms travel time compared to existing state-of-art approaches. We have herewith attached manuscript 2 below.

Adaptive Traffic Signal Control Using Travel Time Reliability for Vehicular Ad hoc Networks

Subash Humagain and Roopak Sinha, Senior Member, IEEE Department of Information Technolog and Software Engineering, Auckland University of Technology Auckland, New Zealand

AUT Tower, Level 7, 2-14 Wakefield Street, Auckland, New Zealand

Tel: +64-9-921 9999 Ext.6376;

corresponding author: subash.humagain@aut.ac.nz

3.1 Introduction

Adaptive Traffic Control (ATC) systems manage traffic signals' timing following changes in actual traffic demand (Maslekar et al., 2011). ATC has been widely researched and commercially deployed (Mannion, Duggan & Howley, 2016). Most current ATC systems like SCATS, SCOOT, OPAC, UTOPIA, RHODES, Balance, In-Sync, and MOTION implement inductive loop detectors as road sensors to monitor the traffic (Zhao & Tian, 2012). A loop detector can only identify the presence of vehicles with varying precision. Current ATC systems use this information to allocate the proper amount of green time to phases of a traffic light (Younes & Boukerche, 2018).

Vehicular ad hoc networks (VANETs) can provide abundant data to aid ATC systems. VANETs are the integration of wireless networks to vehicles and enable communication between mobile vehicles and roadside units. Technologies used for VANET-based vehicle-to-X (V2X: vehicle, infrastructure, road, human, internet) communications include the IEEE 802.11 standard for Dedicated Short-range Communication (Grilo & Nunes, 2002). VANETs are being increasingly studied by traffic engineers and researchers for improving traffic efficiency (ur Rehman, Khan, Zia & Zheng, 2013;

Gradinescu, Gorgorin, Diaconescu, Cristea & Iftode, 2007). VANET-based data like location and speed of individual vehicles, evidently richer than inductive loop-based data, can enable more precise ATC and traffic flow predictions (Maslekar et al., 2011).

The primary objective of ATC systems is to improve the efficiency of a traffic-light controlled intersection, which can be measured using several performance parameters: throughput, waiting time, queue length, overall travel time, fuel consumption and gas emission (Balke et al., 2005). Optimization of ATC system minimizes objective function derived in terms of above-listed performance metrics. Since users (drivers) are an integral part of a traffic control system, metrics that measure traffic system's performance from the user's perspective are of prime importance (Zheng, van Zuylen, Liu & Le Vine, 2016).

The probability of vehicle reaching its destination from the point of origin within a given time is termed as travel time index (TTI) measured in terms of reliability. TTI (also termed as travel time reliability) indicates the variability in delays due to stochastic demand in traffic, stochastic traffic at intersections, mid-link disturbances like stopping of vehicles, pedestrian crossing, asynchronous traffic signal control with traffic demand and weather conditions (Clark & Watling, 2005). TTI is defined as the ratio of actual travel time measured during congestion to the required time during free flow state (Zheng et al., 2016). Depending on the objective, TTI can be measured for a segment of road, a path comprising multiple segments and the entire network within origin to the destination.

TTI is an essential but mostly unexplored performance parameter for ATC-controlled intersections. ATC system implemented in VANETs make traffic control system equivalent to a real-time system. TTI in traffic domain is analogous to stretch in real-time systems. *Stretch* is defined as the factor by which a job is slowed down as compared with the time it takes to process on a free system (Harchol-Balter, Bansal & Schroeder, 2000). Stretch measures the quality of service offered to a job for its demand for

resources. In a system with fluctuating job sizes, stretch also emulates the psychological belief of users that they are ready for longer response times for larger requests. Therefore stretch measures the *fairness* of the service the job experiences within the system (Muthukrishnan, Rajaraman, Shaheen & Gehrke, 1999). The measure of stretch (the average stretch of the system) has been extensively used for experimental study of the performance of different applications like operating systems, databases, and parallel systems (Muthukrishnan et al., 1999).

In this paper, we propose an ATC algorithm that optimizes TTI for an isolated intersection achieved by analogically mapping TTI with stretch. We leverage the availability of real-time data like vehicle speed, position, and time it takes to pass the intersection from connected VANET environment. This data can be used to monitor and optimize TTI. Our algorithm, called Fair Scheduling Algorithm (FSA), identifies and schedules platoons (vehicles grouped together) at each approach of an intersection to minimize the objective function of TTI for the vehicles. Instead of minimizing overall TTI for a segment of road, a path comprising multiple segments and the entire network, FSA is the first algorithm in traffic engineering, which minimizes the average TTI of vehicles passing through an intersection. This motivation is based on the fact that previous studies suggest that TTI optimization not only optimizes average TTI but can simultaneously optimize response times (queue delays) and throughput up to certain constant factors (Zheng et al., 2016).

The FSA algorithm is developed through a novel approach of mapping traffic control problems into real-time systems problems analogically. We translate the scheduling problem in traffic domains into the problem of scheduling conflicting jobs in real-time systems. Once the nature of the real-time scheduling problem is characterized, an appropriate scheduling algorithm is identified and then adapted for use in traffic control. The conflict among competing vehicles from different approaches that cannot be scheduled simultaneously is visualized by reducing a traffic intersection (in any

possible configurations) into a conflict graph (Irani & Leung, 2003). This graph is used by the FSA algorithm to generate an optimal schedule for vehicles in traffic intersections. In traffic networks, vehicles can arrive at any time, and very little is known about future vehicle arrivals. Hence, we adopt a dynamic FSA algorithm to make decisions on the fly, based on the concept of online scheduling (Albers, 2003). Consequently, the proposed online FSA algorithm for ATC schedules platoons on conflicting approaches. FSA is shown to be 2-competitive compared to an optimal offline algorithm. This means that FSA is highly efficient when compared to similar online algorithms that minimize average TTI and the worst-case average TTI produced by FSA is bounded by twice the stretch produced by an optimal offline FSA algorithm with perfect knowledge of future vehicle arrivals.

FSA also includes a novel algorithm as one of its components for dynamically identifying and scheduling platoons. FSA minimizes the TTI for each *job*. Considering each vehicle as a job can incur massive context-switching and in the worst-case scenario, the ATC may act just like a stop sign. Therefore we propose an algorithm to group approaching vehicles into variable-sized platoons using VANETs. This is analogous to jobs with different processing times in real-time systems. The number of vehicles in a platoon is determined from the spatial distance between two vehicles, the direction they are travelling, the distance of vehicles from stop line and their status of being at rest or motion. Variable-sized platoons reflect the differences in resources (green-time) required by each platoon. Platooning allows us to monitor the fairness offered to vehicles in terms of waiting times by the FSA algorithm. Variable-sized platoons are realistic abstractions of traffic. For instance, two trucks can form a longer platoon than six cars. Since the length of different vehicles is measured in terms of passenger car unit, the overall processing time of platoons are in-line with the principle of headway time in traffic engineering (Roess, Prassas & McShane, 2004).

Evaluation of FSA shows promising results. We performed extensive experiments

using the simulation engine "Simulation of Urban Mobility" (SUMO) (Krajzewicz, Erdmann, Behrisch & Bieker, 2012) and INET/OMNET++ (Varga & Hornig, 2008). We empirically prove that FSA online algorithm is more reliable and fairer in scheduling in terms travel time with Buffer Index (BI) as optimization parameter compared to existing state-of-art approaches. Additionally, FSA achieves equivalent performance with delay minimizing and throughput maximizing baseline algorithms. We performed the fairness test of FSA using Jain's fairness index and compared it with existing state-of-the-art ATC algorithms.

The remainder of this paper is structured as follows: Section 3.2 describes and analyzes the previous VANET enabled ATC algorithms that optimizes multiple objectives and Section 3.3 describes in detail of the FSA algorithm. We elaborate on the use of VANET for FSA scheduling in Section 3.4 and describe the implementation of FSA in Section 3.5. We conducted experiments and compared the performance of FSA with other state-of-art algorithms in Section 3.6 and the final Section 3.7 concludes the paper.

3.2 Related Work

Adaptive traffic control (ATC) algorithms for isolated traffic intersections to optimize traffic performance in terms of queue delay and throughput (Zhao & Tian, 2012) using inductive loop data have been widely studied. The information extracted from loop detectors can only feature the presence or absence of a vehicle (Feng, Head, Khoshmagham & Zamanipour, 2015). Newer data sources are needed to improve ATC for next-generation traffic networks drastically.

With the availability of information like speed and position through VANETs, ATC algorithms can become more precise and real-time (Pandit, Ghosal, Zhang & Chuah, 2013). Several studies have used VANETs to report dynamic traffic parameters like traffic density and vehicle speed to the closest traffic signal (Maslekar et al., 2011;

Gradinescu et al., 2007; Priemer & Friedrich, 2009; Chang & Park, 2013). Roadside units (RSUs) installed at every intersection collect real-time data from travelling vehicles and optimize traffic flow by changing different traffic phases dynamically (Nafi & Khan, 2012).

VANET-based ATC can incorporate newer technologies, like agent-based reasoning and machine learning. Agent-based solution models vehicles and intersection management as agents, which dynamically exchange information to effect optimal signal timing (Kari, Wu & Barth, 2014). An alternative method (Chou, Deng, Li & Kuo, 2012) collects information like fuel consumption, pollutant emission and passenger loading information using different sensors. Sensor data and messages are transmitted via VANET to an RSU, implemented as a traffic signal control agent. The ATC algorithm computes the expected arrival time of each vehicle and allocates green time based on such computations. An online machine learning algorithm estimates travel time and apply adaptive traffic control in a V2I environment (Cai, Wang & Geers, 2013). The traffic controller learns progressively from its performance and the remaining travelling time of a vehicle when it approaches an intersection, using approximate dynamic programming. Likewise, dynamic programming based ATC is proposed in (Feng et al., 2015). This algorithm predicts signal phase sequences and values and then uses forward recursion to calculate optimal phase duration and backward recursion to calculate optimal signal policy.

VANETs provide an environment where vehicles can use real-time information to make decisions to arbitrate their movements in conflicting approaches even without using traffic lights. Vehicles in the same lane can be divided into different groups using the information received from a VANET. These groups use wireless communication to schedule themselves in an intersection without using traffic lights (Cheng, Wu, Cao & Li, 2016).

The use of real-time scheduling algorithms in ATC is promising, mainly when

vehicles are grouped into platoons. The oldest arrival first (OAF) was the first real-time online scheduling algorithm used for VANET-enabled ATC (Pandit et al., 2013). This study implemented VANET for obtaining speed and position information of vehicles approaching an intersection. The objective of a real-time algorithm is to schedule *tasks*, which are traditionally slices of software programs, onto available hardware resources. In ATC, vehicles can be seen as tasks, which need to use an available resource, namely the intersection. However, if each vehicle near the intersection is treated as a task, a scheduling algorithm might require an excessive amount of switching between approaches, reducing the performance of the intersection. Hence, most approaches combine vehicles into platoons. In (Pandit et al., 2013), platoons are formed so that they all have the same processing time, and the algorithm always schedules the oldest platoon first, which helps achieve the objective of minimizing the average delay experienced by vehicles at the intersection. Intelligent traffic light controlling (ITLC) implements a VANET based ATC for individual and multiple intersections (Younes & Boukerche, 2015). This approach also follows a platooning approach, and approaches with higher densities of platoons were scheduled first. This algorithm was optimal in terms of throughput maximization and also outperformed the oldest arrival first approach (Pandit et al., 2013). A delay-based, throughput optimal ATC was introduced in (J. Wu, Ghosal, Zhang & Chuah, 2017). It uses back pressure control to prevent platoons from experiencing an excessive delay because of their smaller queue length and tries to achieve fairness.

Almost all ATC systems try to optimize the system's metrics like waiting time, throughput, queue length, delay, fuel consumption and gas emission. But it is also equally important to address ATC system's response to the user. TTI is the ratio of actual travel time measured during the congestion to the required time during free flow state (Zheng et al., 2016). TTI indicates the variability in delays due to stochastic demand in traffic, stochastic traffic at intersections, mid-link disturbances like stopping

of the vehicle, pedestrian crossing, asynchronous traffic signal control with traffic demand and weather conditions (Clark & Watling, 2005). Though TTI measured in terms of reliability is not new in traffic domain, very few studies have incorporated TTI in optimizing traffic control systems. An adaptive traffic control system implemented using SCATS that considered TTI in terms of reliability measures was introduced in (S. K. Wu, 2009). A robust model to produced optimized signal timing to minimize reliability index (TTI) considering variations in demand over a single day and between multiple days was implemented by (Yin, 2008) and was further applied it over arterial road network by (L. Zhang, Yin & Lou, 2010). Recently, Zheng et.al. proposed a framework to optimize signal control strategies for reliability and expected values of travel time using genetic algorithm for urban arterial road networks (Zheng et al., 2016). Similarly, a traffic network signal optimization model using heuristic particle swarm optimization to optimize travel time reliability was proposed by (Z. Ma, Huang, Li & Guo, 2020).

Most of the studies try to achieve an optimized cycle timing for an isolated intersection or an arterial road network considering TTI as the optimization metrics. None of the studies has implemented ATC algorithm to minimize TTI in the connected environment for isolated traffic intersection. In this paper, we propose an ATC algorithm that optimizes TTI for an isolated intersection achieved by analogically mapping TTI with stretch. We leverage the availability of real-time data like vehicle speed, position, and time it takes to pass the intersection from connected VANET environment. This data can be used to monitor and optimize TTI. Our algorithm, called Fair Scheduling Algorithm (FSA), identifies and schedules platoons (vehicles grouped together) at each approach of an intersection to minimize the objective function of TTI for the vehicles. Instead of minimizing overall TTI for a segment of road, a path comprising multiple segments and the entire network, FSA is the first algorithm in traffic engineering, which minimizes the average TTI of vehicles passing through an intersection.

Existing VANET-based ATC has limitations. Most of the earlier studies that implemented real-time scheduling algorithms in the traffic domain used the same sized platoons to ascertain that all the tasks have the same processing time. In cases where platoons have very few vehicles within it (especially in intersections where they have an uneven distribution of traffic), the algorithms need to assign entire green time for the duration of the platoon. This prevents algorithms from being completely adaptive and green time calculated using such algorithms is not the optimized green time in the longer run. We have addressed this limitation by using variable-sized platoons. Platoon size determined from the exact number of vehicles present at that particular lane reflects the real-world traffic scenario and makes the algorithm more adaptive.

In addition, FSA is a novel algorithm to be used in traffic domain. Almost all existing ATC systems focus on optimizing either queue delay or throughput. Delay minimizing algorithms prioritize lanes experiencing maximum delays and throughput maximizing algorithms schedule lanes with higher traffic density. In cases where there is an uneven distribution of traffic densities, lanes with a minimal queue or very less density have to wait unnecessarily. This implies that existing scheduling algorithms used in the traffic domain are not fair to all the tasks. From the users' perspective, the system is less reliable as their expected and actual travel time vary largely. Furthermore, implementation of VANET can improve ATC system performance up to a certain level of traffic penetration. ATC system algorithms have opportunities to improve the performance only if the traffic flow is not in synchronization with the control algorithm in case of medium or lightly loaded intersections. Designing any advanced traffic control algorithm cannot further minimize waiting time or maximize throughput of highly loaded intersections. In such a scenario, recently, researchers have started considering an alternative measure of performance called *TTI* analogous to *stretch* in operating systems, parallel systems, web servers, and database systems (Muthukrishnan et al., 1999). So, we have implemented FSA, an algorithm that minimizes average TTI

experienced by each vehicle, which is fairer in scheduling than existing state-of-art ATC algorithms but still maintains the identical overall performance of minimizing queue delay or maximizing throughput.

3.3 Online Task Scheduling Implementation for Traffic Light

3.3.1 Traffic Intersection

Intersections arbitrate conflicts between vehicular movements. The most commonly encountered *four-legged intersection* with eight different movements is depicted in Fig. 3.1. Other types of intersections are rarer and often ignored when designing intersection control algorithms (Irani & Leung, 2003). However, our approach is general enough to be used for other intersections. Traffic lights installed at the intersection change their phases to control vehicular movements. The total time for completing all phases is called *cycle time*. The distance or time between two vehicles on the same approach is measured as *headway*. From Fig. 3.1, vehicles on approach 3 cannot be

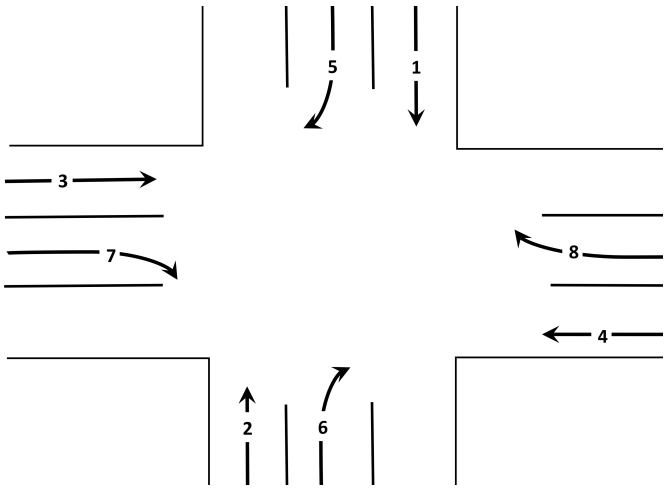

Figure 3.1: Four legged intersections with a movement for left hand drive

allowed to pass the intersection when any of the conflicting approaches 1, 2, 5, 6, or 8 are allowed. However, approach 3 can be enabled at the same time as non-conflicting approaches 4 and 7. We can enumerate all conflicting and non-conflicting approaches for each approach i as set C_i and NC_i.

3.3.2 Conflict Graph

A conflict graph sometimes referred to as precedence graph, is widely used in obtaining correct results for concurrent operations in real-time systems, and can also be used in traffic intersections. A traffic control algorithm operating on an intersection must allow vehicles to pass without conflicts. This problem is analogically equivalent to a job scheduling problem in a real-time system. Conflict graphs are extensively used in job scheduling where simultaneously occurring jobs compete within a system to utilize limited resources. For each approach i, we can construct a conflict graph $G = (V, E)$ with vertices V and edges $E \subseteq (V \times V)$ for traffic intersection in Fig. 3.1. The vertices are represented by the set of approaches $(1, \ldots, 8)$ and the edges are represented as

$$E = \bigcup_i \{(i, j) | j \in C_i\}$$

Intuitively, each approach is a node in the conflict graph and has an edge connecting it with every conflicting approach. Approaches connected directly within the conflict graph cannot be enabled at the same time. Fig. 3.2 shows the conflict graph for the intersection shown in Fig. 3.1.

3.3.3 Behavior of Jobs

Vehicles either stop, slow down or speedup depending on the state of a traffic signal when they approach an intersection as shown in Fig. 3.3. Vehicles in the speed region are unaffected by the traffic signal and continue with their original speed. In the

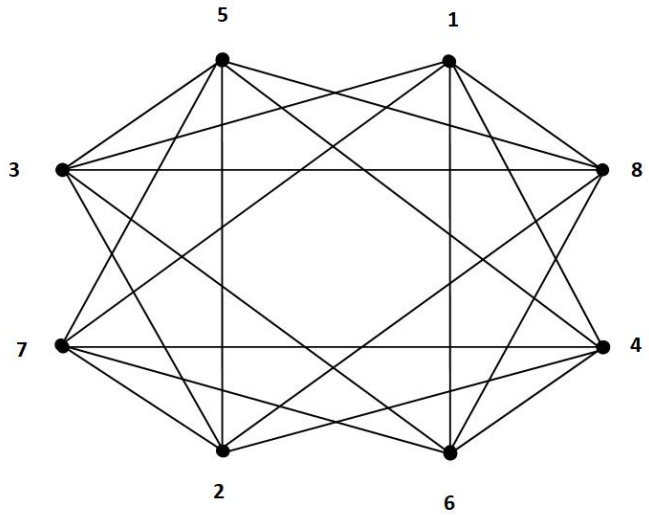

Figure 3.2: Conflict graph $G(V, E)$ of jobs in four-legged intersection

slowdown region, vehicles either decelerate to stop at the red light or accelerate to pass the intersection because of a green or orange light.

Different movement pattern of a vehicle crossing an intersection is captured by *headway*. The first headway, which is the time between the light going green and the first vehicle crossing the intersection, is usually longer than second and subsequent headways because it includes the acceleration time of the vehicle and the reaction time of the driver. Subsequent headways are generally shorter because reaction times of all drivers can overlap. After a certain number of vehicles the headway achieves a constant value and is termed as saturation headway h. The error e_i for few initial vehicles are added as start up time lost and compensated while designing green time. Therefore, green time required to clear N lined vehicles is given by,

$$T = \sum_{i=1}^{n} e_i + h * N \tag{3.1}$$

Grouping vehicles into platoons allow us to use real-time scheduling algorithms for traffic control. We group vehicles at each approach of an intersection into one or more variable sized *platoons* or *jobs*. Platooning aids to eliminate the inconsistencies

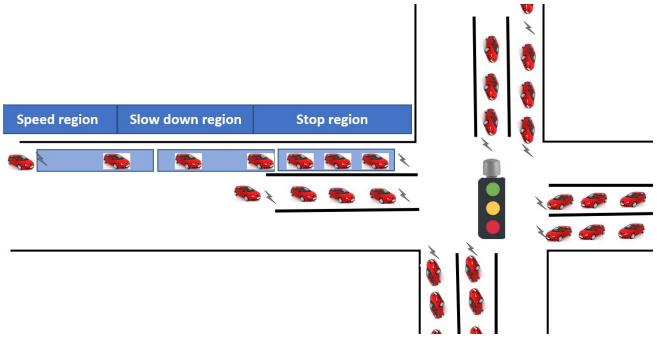

Figure 3.3: Traffic signal control architecture using VANET

created by human in maintaining a safe distance and nonzero reaction time (Bergenhem, Hedin & Skarin, 2012). The traffic control problem can then be posed as a problem of scheduling jobs with conflicts (Irani & Leung, 2003). Scheduling jobs with conflict for a given conflict graph to minimize average stretch is an NP-hard problem even if we consider that all jobs arrive at the same time (Karp, 1972).

The random pattern of vehicular arrival to an intersection and its unpredictable nature of future makes scheduling an online problem and needs to be dynamic. A scheduler can, therefore, only make scheduling decisions based only on the current set of jobs. This is an example of *online scheduling*. When a scheduler knows in advance the arrival time of all jobs, it can make calculated decisions based on past, current, and future information, which is classified as an *offline* scheduling strategy. An offline algorithm, having complete knowledge, generally produces better results than an online algorithm having partial knowledge. The performance of an online scheduling algorithm is therefore compared with an optimum offline algorithm in terms of its *competitive ratio*. An online algorithm is r-competitive if the total cost for its scheduling is r times the cost incurred by an optimal offline algorithm. For request sequence γ, let $C_A(\gamma)$ represents the cost incurred by an online algorithm A and $C^*_A(\gamma)$ represents the cost incurred by an optimal offline algorithm then A is called r-competitive if there is a constant c such that:

$$C_A(\gamma) \leq r \cdot C^*_A(\gamma) + c \tag{3.2}$$

3.3.4 Analogical mapping of TTI in traffic domain with stretch in real-time systems

The probability of vehicle reaching its destination from the point of origin within a given time is termed as travel time index (TTI) measured in terms of reliability. TTI is defined as the ratio of actual travel time measured during congestion to the required time during free-flow state, Mathematically,

$$TTI = \frac{TT_{Mean}}{TT_{Freeflow}} \tag{3.3}$$

Where TT_{Mean} is the mean actual travel time and $TT_{Freeflow}$ is the mean free-flow travel time. In this study we consider Buffer Index (BI), a typical travel time reliability measure, which represents the extra buffer time traveller must add to their average travel time before planning their trips which are incurred due to stochastic demand in traffic, stochastic traffic at intersections, mid-link disturbances like stopping of the vehicle, pedestrian crossing, asynchronous traffic signal control with traffic demand and weather conditions (Clark & Watling, 2005). We have used a 90^th percentile value of travel time for the representation of near worst-case travel time. Therefore,

$$BI = \frac{TT_{90\%} - TT_{Mean}}{TT_{Mean}} \tag{3.4}$$

Similarly, stretch is defined as the amount by which a job has been slowed down by the loaded system compared to a free system. Stretch is the ratio of response time and processing time of a job (Harchol-Balter et al., 2000). Response time is the difference between the completion and arrival time of a job, which reflects the amount by which a job has been slowed down expressed by the following equation:

$$s = \frac{c_j - a_j}{p_j} \tag{3.5}$$

Where c_j, a_j, and p_j are the completion time, the arrival time and processing time of job j respectively. From Equation. 3.4 and Equation. 3.5 we can deduce that buffer index in traffic domain is equivalent to stretch in real-time systems. Therefore, if we implement a real-time system scheduling algorithm to optimize stretch in traffic domain it eventually optimizes the BI. The objective function for average buffer time index of an approach i can be expressed as:

$$y = \frac{1}{n} \sum_{i=1}^{n} BI_i \tag{3.6}$$

Where, n is the total number of approaches in the traffic intersection and BI_i is the average buffer index of the vehicles travelling through that approach and y optimization objective function. For a signalised traffic intersection, effective green time $G_i \geq 0$; optimum cycle time $C_0 \geq 0$; and phase offset $\phi \geq 0$ Therefore, our optimization algorithm FSA should achieve a minimum value of the objective function satisfying above constraints for G_i, C_0, and ϕ.

$$min\{y\} = min\left(\frac{1}{n} \sum_{i=1}^{n} BI_i\right)$$

In this study, we have considered variable-sized platoons as jobs and implementing FSA as a real-time task scheduling algorithm, minimizing average stretch eventually achieves a minimum of the above objective function.

3.3.5 Competitive analysis of FSA online algorithm

The slowdown combines efficiency and fairness and is generated from the idea that larger jobs can tolerate a longer waiting time. Minimizing average stretch satisfies that the schedule so produced reduces overall slowdown and is fair to all the jobs in terms of waiting time. In the following sub-section, we present the Fair Scheduling Algorithm

(FSA) algorithm to minimize the average stretch.

FSA is an extension of an existing Shortest remaining time first (SRTF) algorithm for the traffic domain. This algorithm processes the job with the smallest amount of time remaining to execute or minimum value of stretch. Since the job being processed is the one with minimum remaining processing time and the processing time always reduces as execution continues, the jobs being executed always get a chance to complete unless a new job with a smaller amount of processing time arrives. FSA algorithm uses VANET to implement SRTF. We design an algorithm to divide approaching vehicles into variable-sized platoons using VANETs. In a real-time system, this is analogous to scheduling jobs with conflicts having different processing times. The traffic controller then processes conflict-free platoons using SRTF algorithm. The two-phase approach, where approaching traffic is divided into variable sized platoons using a platooning algorithm and scheduling them using SRTF algorithm to generate a conflict-free schedule, generate our FSA algorithm.

The input to FSA scheduling algorithm is set J with n number of jobs with processing time p_j and its equivalent conflict graph $G(V_i, E)$ where V_i represents the jobs and edge E represents pair of conflicting jobs that are not schedulable at the same instant. The schedule is nothing but an allocation of processor's time slots (effective green time in traffic domain) satisfying the following conditions:

- An individual job $j \in J$ can be assigned a time slot of p_j on the processor.
- Time slots allocated for two conflicting jobs can not overlap.

In case of a traffic intersection, all platoons (considered as jobs) generated from each lane demand effective green time. There is a conflict between a subset of jobs if their total demand for green time exceeds supply. Such problems of resource sharing can be modelled using a conflict graph (Even, Halldórsson, Kaplan & Ron, 2009).

Algorithm 1 FSA Algorithm

Input: vertices $V_i(1, 2, ..., 8)$, conflict graph $G(V_i, E)$
Output: schedule non-conflicting vertices with minimum stretch
1: Let S be the set of average stretch on all vertices of G;
2: **while** vertices V_i have jobs waiting **do**
3: Let S_j be the minimum value of stretch among S;
4: **for** each vertex without having edge with V_j, **do**
5: Schedule the job with minimum value of stretch S_j;

Let a_j = arrival time and p_j = processing time of j. The performance of an algorithm is measured by the total stretch achieved by the schedule. Ideally, in a relaxed system, jobs get processed immediately after they arrive, resulting in the value of stretch to 1. In a heavily loaded system, the value of stretch is greater than 1. The performance of the scheduling algorithm is measured relatively based on the stretch metric given below. The total stretch of a schedule for time unit of consideration is the sum, taken over all jobs j during that unit of time, of the stretch of j under the schedule, where the stretch of a job is the ratio of the response time of the job to the processing time of the job.

$$s_j = \sum_{j=1}^{n} \frac{c_j - a_j}{p_j}$$

Where, response time is the difference between completion time c_j and arrival time a_j.

Job j can be scheduled at any time after the job has arrived, i.e., $t \geq a_j$. Schedule assigned at time t by Algorithm. 1 stated earlier is only dependent on the jobs that have already arrived before t or exactly at t. Here the scheduler has to make decisions on the go. In this context, we are focused on assigning the processor's time slot in such a way that it minimizes the average stretch.

Adjacent approaches like 1 and 2 in traffic intersection shown in Fig. 3.1 can be merged into one node resulting $K_{2,2}$ conflict graph. We represent nodes of $K_{2,2}$ on the right side as r_a and r_b and on the left side as l_a and l_b. Every time FSA processes the

job from the side of the conflict graph with minimum stretch. Afterwards, the scheduler chooses the node from this side with remaining jobs and processes the job with minimum stretch. We prove that any job in the schedule produced by FSA experiences latency of at most two times the latency experienced by the jobs scheduled by offline optimal algorithm FSA* implying FSA algorithm is 2-competitive. To prove this, we require the following Lemma.

Lemma 1. *Let the upper bound for maximum latency created by the schedule of optimal offline algorithm FSA* be S. In such condition, for all times t, the online FSA will always maintain invariant listed below:*

1. *If FSA* offline optimal algorithm has an edge of weight w at any time instant t, then the number of jobs on the edge of optimal schedule produced by it is always at worst $w - S$ jobs.*

2. *If FSA* offline optimal algorithm has a node with weight w at time instant t, then the number of jobs on the node the optimal schedule produced by it is always at worst $w - S$ jobs.*

Proof. Let us Consider that FSA (online algorithm) has maintained both of the above invariants for time $t - 1$. We will prove that the same is maintained after time t. The conditions stated by the invariant are still maintained by all the jobs arriving at the end of time $t - 1$ and beginning of t. When both FSA and FSA* start scheduling the jobs from the set of independent nodes of $K_{2,2}$ conflict graph, we need to prove that these conditions are still maintained at the end of t. At the beginning of t, without-loss-of-generality, let us assume that node l_a contains the job with minimum stretch.

If at the beginning of t, FSA has any jobs on l_b, then it schedules individual jobs from each edge of conflict graph. So at the end of t condition 1 of the invariant is

preserved. We consider that for a unit time, the number of jobs processed by an optimal offline algorithm from each edge is only one.

If at the beginning of t, FSA has no jobs on l_b, then condition 1 of the invariant will only be maintained as long as condition 2 of the invariant will be maintained. But if FSA has a node on the right-hand side with at least $S + 1$ jobs, we cannot ascertain the condition 2 of the invariant. Consider node r_a is having at least $S + 1$ jobs. (Same holds true for r_b). At the beginning of t, let us assume node l_a has m jobs and node r_a has n jobs. This imply that, the optimal schedule on (r_a, r_b) has at least $m + n - S$ jobs out of which $n - S$ must be on r_a. We must ascertain that the optimal schedule will maintain at least $n - S$ jobs till the end of t to satisfy condition 2 of invariant. Let us address the following two cases:

Case I. At the beginning of t, Online FSA has a job on l_a with the latency of at least S. For this condition, if the optimal schedule also has the latency of at least S job on l_a, it has to schedule that job. This makes $n-S$ jobs still waiting to process on r_a when time unit t ends. If in case, the schedule generated by optimal offline algorithm does not have a latency of at least S job on l_a. We initially inspect that for time units of $S-1$, jobs arriving on l_a is no more than $m - 1$. This is due to the reason that the FSA online algorithm can have a maximum of $m - 1$ jobs that are recent after the latency of at least S job and the algorithm would not have scheduled any recent jobs within the latency of at least S job. Therefore, optimal schedule can only have at most $m-1$ jobs on l_a meaning it has at least $n-S+1$ jobs on r_a at the beginning of t. Thus resulting $n-S$ jobs remaining after t.

Case II. At the beginning of t, Online FSA does not have a job on l_a with the latency of at least S. Since l_a has the job with minimum stretch, the FSA online algorithm does not have a job on r_a with the latency of at least S. This implies n jobs have shown up on r_a during the last $S - 1$ units of time. At the beginning of t, the maximum number of

jobs that any algorithm (including optimal) can schedule is $S - 1$. Therefore, before time t, schedule generated by optimal algorithm has at least $n - S + 1$ number of jobs on r_a and $n - S$ jobs when t ends. ■

Theorem 1. *Online greedy FSA is 2-competitive.*

Proof. If FSA* (an optimal offline algorithm) can schedule jobs with any arrival sequence with the maximum latency of S, then worse case latency for the online algorithm cannot be more than $2S$. It provides an upper bound $(2S + 1)/(S + 1)$ for the algorithm on a competitive ratio with maximum response time as cost function.

If invariants are maintained at all times, we can never achieve an edge weight of $2S + 2$ or more. Because in such case, the optimal schedule will have edge weight of $S + 2$ which eventually imply some jobs might have a latency of at least $S + 1$, which is impossible. As algorithm can never have more than $2S + 1$ jobs on any edge, let us assume for node l_a there is the arrival of any job, say j, then there can never be more than $2S$ jobs on any of the incident edges to l_a. Let us assume m be the number of existing jobs l_a during the arrival of j and we know $0 \leq m \leq 2S$. Therefore either of right-hand side nodes can have a maximum of $2S - m$ jobs.

Once the left-hand side is chosen m number of time, j will be the job with minimum processing time left on l_a and this job will be scheduled when the left-hand side is chosen next time. So, by proving the right-hand side is not chosen $2S - m$ time before j is scheduled, we can state that the maximum amount of time j has to wait before it gets scheduled is at most $2S$. Once the right-hand side has been chosen $2S - m$ times and until now, if j is not picked yet, then it results in the left-hand side having a job with maximum processing time or stretch. ■

The above discussion concludes that a proper reduction of stretch, minimizing real-time online task scheduling can be implemented in traffic intersections that maintain 2-competitive performance bonds.

Optimality of FSA

In this subsection, we prove that the FSA algorithm presented previously achieved the optimal competitive ratio. We follow the method of the adversary to prove all the lower bonds. The adversary generates the sequence of arrival of jobs imitating the online algorithm's nature. The adversary can choose any of the nodes of conflict graph for the arrival of jobs at the start of each time unit. Once the sequence of the job has been finalized, the adversary determines the schedule of the job in an offline manner. Then we compare the cost incurred by the online algorithm to the cost of the offline algorithm decided by the adversary.

Lemma 2. *Let us consider $Alg.$ as an arbitrary algorithm. Let us consider an edge (l_x, r_y) at the end of any time t where the adversary does not have any jobs to process and $Alg.$ has i jobs. In such a case, the adversary can compel to generate the sequence where $Alg.$ has an edge weight of $i + 1$ and the adversary has scheduled all the jobs and left with no jobs left at any of the nodes and edges. Moreover, the adversary can never have any job with a latency greater than $i + 1$.*

Proof. The adversary forces the arrival of jobs on both nodes l_x and l_y until algorithm has $i + 1$ jobs on any one of the nodes. This happens at the start of the time $\bar{t} \leq t + i + 1$. Without loss of generality, let us consider node l_x has $i + 1$ jobs then, in that condition, the jobs scheduled by the adversary can achieve maximum latency of $i + 1$ resulting l_x to be empty after $\bar{t}$. This keeps the adversary following the strategy of emptying out its own graph while forcing algorithm to maintain $i + 1$ jobs at the edge. Let us consider a node r_z, which is not r_y on the right-hand side of the conflict graph. The adversary has jobs arriving at r_z for every time unit of $i + 1$. The adversary schedules the job with the highest processing time on r_y and with the lowest processing time, i.e., the newest job on r_z. Again after $i + 1$ unit of time, the graph of the adversary is empty. But the algorithm still has $i + 1$ jobs on (l_x, r_z) because a job was requested on r_z for every time

unit. ■

Theorem 2. *For a conflict graph of format $K_{2,2}$, the FSA algorithm has a maximum competitive ratio of 2.*

Proof. For any positive integer S, there exist an adversary that forces the algorithm for maintaining a job with a latency of at least $2S$; meanwhile, the latency created by adversary itself is at most L. This allows the lower bond of $(2S + 1)/(S + 1)$ for any deterministic algorithm to determine its competitive ratio. The theorem follows since S can be arbitrarily large.

When j increases from 0 and reach $S - 1$, the algorithm has to start every single state with j jobs on the edge. Let us now invoke Lemma. 2 in order to achieve $j + 1$ jobs on an edge. Once all the process ends, the graph of the adversary is empty, while the algorithm still has S jobs at an edge (l_x, r_y). For the next S units of time, the adversary will have a job arrive on l_x and l_y. Thus, scheduling the job with the lowest processing time first, the adversary never can have a latency of greater than S and the algorithm must incur latency of at least $2S$ to have $2S + 1$ jobs on edge (l_x, l_y). ■

3.4 Vanet Based Scheduling

In this section, we elaborate on how a platooning algorithm has been implemented to generate platoons of variable size and further implemented to adaptive traffic signal control in the VANET environment for minimizing the average stretch using FSA algorithm.

3.4.1 System Model

VANETs facilitate the free circulation of data among objects within a traffic intersection. Here, we explain the system model for VANET based adaptive traffic signal controller.

In this study, we considered that all the vehicles are equipped with the communication hardware for V2X communication and GPS to gather speed and position data. We have also assumed that the decision about the lane and route to follow is an independent driver's decision and is accessible to the platooning algorithm. We can implement this in SUMO by randomly assigning lane and route to the individual vehicle using an inbuilt function called `DUAROUTER`. RSU installed at the traffic controller can receive the information broadcasted by the vehicles. Communication is carried out using standards defined in the IEEE 802.11 protocol to support wireless access in the vehicular environment which uses overall bandwidth of 75MHz divided into seven 10MHz channels(composed of single Control Channel and six service channels) in the 5.9GHz spectrum band for Dedicated Short Range Communications (DSRC). To increase switching efficiency among seven DSRC channels, it uses the IEEE 1609.4 protocol, which is just an extension over Medium access control layer operation of IEEE 802.11. The architecture of this system is shown in Fig. 3.3. The information beacon sends data packets that consist of the speed, lane position, and destination lane of the vehicles. Speed and position data in the real-world are gathered from vehicle installed speedometer and GPS. During the implementation phase, we have access over these data from different parameters used in microscopic traffic simulator SUMO. These data are further encapsulated in a packet and broadcasted wireless.

The platooning and FSA algorithms use these data to make meaningful decisions. Once the data are circulated, we collect and process them. The platooning algorithm and FSA algorithm process these data and instruct the traffic controller to change the phases of the traffic signal. A detailed description of how these data are processed and implemented in the traffic controller to design an adaptive traffic control system is explained in Section. 3.5.

3.4.2 Platooning Algorithm

The performance of the FSA online algorithm enhances with the use of information transmitted over VANET. In the preceding section, we theoretically calculated the lower and upper bond of an online algorithm and proved that it is 2-competitive for minimizing average stretch where the algorithm had no further knowledge of input jobs. The performance bond of the FSA online algorithm holds with the condition that future information of jobs related to their arrival and processing times is unknown. Utilizing information related to speed, position, lane, etc. transmitted to the controller prior to the arrival of vehicles could improve 2-competitiveness of FSA online algorithm. But due to the presence of physical obstacles like buildings, trees, and attenuation from other vehicles, the radio range of DSRC communication is limited to a few hundred meters, which provides a short-sighted view of future jobs and eventually the performance of the FSA algorithm will come down to 2-competitive. In this paper, we instead tried a different approach to use information gathered using VANET. One major limitation that causes the performance of the algorithm to deplete is the variable length of Jobs, which is unknown to the scheduler. In this case, it is the variable length of the platoons which require different processing times. If the length of the platoons is known to the scheduler in advance before it makes any decision, the performance of the FSA online algorithm can be enhanced from the theoretical computed value of 2-competitive. This can be achieved by calculating the length of the platoons using the spatial headways of the vehicles joining the platoon, speed of the platoon, the position of the lead vehicle in the platoon, and the last vehicle that has joined the platoon. We have implemented this concept to calculate the reserved time for each platoon that is required to cross the intersection, and this information is encapsulated in the packet broadcasted to RSU using VANET.

Assurance of performance enhancement of FSA can only be achieved if the platooning algorithm follows some conditions. Platoon is the group of vehicles following a lead vehicle. Controlling a platoon is a lot easier than to control the individual vehicle as it is controlled from the lead vehicle. Platooning is implemented to mitigate the inconsistencies generated by human behaviour in maintaining a safe distance and nonzero reaction time to increase the overall capacity of the highway (Bergenhem et al., 2012). FSA processes the job with the minimum stretch and is considered that the job processed completes at the end of the scheduled slot. In case of a traffic intersection, we may have situations where some lane might have a fewer number of vehicles generating at an equal interval of time. This forces scheduler to process these smaller sized platoons preempting the larger sized platoons created from the congested lane which results in unnecessary starvation for larger platoons. This kind of situation restricts the scheduler in producing an optimal schedule. To solve this problem, we propose an optimized solution to calculate the platoon size.

- Allow vehicles to join the platoon being processed if they are within an allowable distance apart and increase execution time (green time) until this platoon is processed. This will minimize unnecessary context switching because of vehicle spacing.

- Limit the maximum number of vehicles joining a platoon to maximum 12 vehicles as platoon length of up to 35 meters does not reduce the performance (Fernandes & Nunes, 2012a) and allow the scheduler to preempt only after the execution of this platoon. This minimizes the starvation for the jobs with higher execution time.

The reserved time required for a platoon to cross the intersection (green time) is estimated in the following ways:

- When the platoon is stopped over the stop line, green time is the sum of start-up time and reserved time of the platoon for passing the intersection.
- For a moving platoon, green time is the sum of reserved time of the platoon and time required by this platoon to cover the distance between the stop line and the present position of the lead vehicle.

Random behaviour of vehicle arrival at the intersection generates variable-sized platoons and demand unequal green time. The platooning algorithm calculates the exact green time needed for each platoon in a controlled lane to pass the intersection. All vehicles generated are initially assigned as an individual platoon. Individual platoons in the same lane and having the same destination lane are eligible for merging thus they merge together to form a larger platoon. The very first vehicle of the merged platoon is assigned as the lead vehicle. All other vehicles follow the lead vehicle's speed and route. Whenever two platoons merge, both the merging platoons are disbanded, and a new platoon with a new length is created. Since the vehicles are randomly assigned to the lane, the size of platoons so created in each lane is different. We now calculate the reserved time for platoons at different lanes. For this calculated reserved time we further calculate the stretch of each platoon from all the approach and they are scheduled by FSA algorithm shown in Algorithm. 1. Once they cross the intersection platoons update to maintain themselves, that means if the original condition of merging as platoon satisfies, they continue to move as platoons else, they will disband. The platooning algorithm is shown in Algorithm. 2.

3.4.3 Adaptive Traffic Signal Control

In this section, we elaborate on the implementation of VANET on the adaptive traffic control system. We make use of the inbuilt feature of traffic light signal (TLS) logic in SUMO. Total cycle time should be chosen in such a way that it can serve the purpose of

Algorithm 2 Platooning Algorithm for FSA scheduling

```
1: for each controlled lane l do
2:     addVehicle Veh_l.i;
3:     create platoon Veh_l;
4:         if ( Veh_l.i CanMergeWith Veh_l ) then
5:             Veh_l.psize = Veh_l.psize + 1;
6:         else
7:             Veh_l.psize = Veh_l.psize;
8:     for each platoon Veh_l do
9:         Reserve time = calculateNewReservetime(Veh_l);
```

optimization of the objective function defined in Equation 3.6. Randomly chosen cycle time and green time slower the convergence towards optimal cycle time and green time to minimize the objective function. Too short cycle times result in frequent switching among the phases within a defined time resulting in a considerable amount of time lost due to multiple switching and will be eventually higher than effective green time. On the other hand, too long cycle time increases overall delays (waiting time) experienced by stopped vehicles. So it is important to determine a reasonable value of cycle time, which can solve both of these problems. in this study we start the implementation of FSA setting optimized cycle time and green time given by Webster's method, which is a well-established method in traffic engineering, determines optimum cycle time as a function of critical flow ratio (throughput) and lost times. Webster's equation is shown below.

$$C_o = (1.5L + 5)/1 - Y \tag{3.7}$$

Where C_0 is Webster's optimum cycle time in seconds, L is loss time determined as the sum of inter-green phases in seconds i.e. $L = (nT_sl + R)$ from n number of phases, T_sl as start-up lost time and R as time during which all signal goes red. Y is the total critical flow rate in vehicles measured in seconds and given as $Y = \sum_{n=1}^{n} Y_i$ where Y_i is the critical flow rate for i-th phase determined as $Y_i = V_i/S_i$ for saturation flow S of the particular lane and observed volume V. Thus calculated cycle time is distributed into

effective green time for different phases using the following equation.

$$G_i = \frac{y_i}{\sum_{n=1}^{n} Y_i}(C_o - L) \tag{3.8}$$

Once the optimal cycle time and individual phase's green time is calculated depending upon the critical traffic volumes of the individual lane, adaptive traffic light logic is initialized with variables like duration, the minimum duration (minDur), and maximum duration (maxDur). The ATC controller is set to the initial phase, and at this moment, an extension that is provided to the green phase is set to 0. We now calculate the reserved time required to platoons from each lane. We do this using function `calculateNewReservedTime()`. If the platoon is at the stop line we calculate length of the platoon using function `pv.getLength()` otherwise we calculate the position at which the platoon is located using `getLanePosition()` and add this value with the value retrieved from `pv.getLength()`. This gives us the length of the last vehicle that has joined the platoon. Now we divide this value with speed adhered by the platoon to calculate the reserved time required for this platoon to pass the intersection. Speed of the platoon is calculated using `getspeed()` function. All the variables like positions of vehicle, intervehicle headway, speed required to execute these functions are encapsulated in a packet and transmitted using VANET. The traffic controller now calculates the stretch of each approach and applies FSA algorithm to provide a green signal to the lane with the smallest value of stretch. The green time is the corresponding value of reserved time for platoon with minimum stretch. To remove multiple switching between phases which increases loss time, two platoons from the same lane with a gap less than threshold gap are provided with the extension in the current green phase. This extension EXT$\leq G_{OPT}$ where G_{OPT} is optimal green time set from Webster's method. EXT is calculated using the same process, as discussed earlier. Once EXT is equal to or greater than G_{OPT} the current phase goes RED and

controller process next lane.

3.5 Implementation

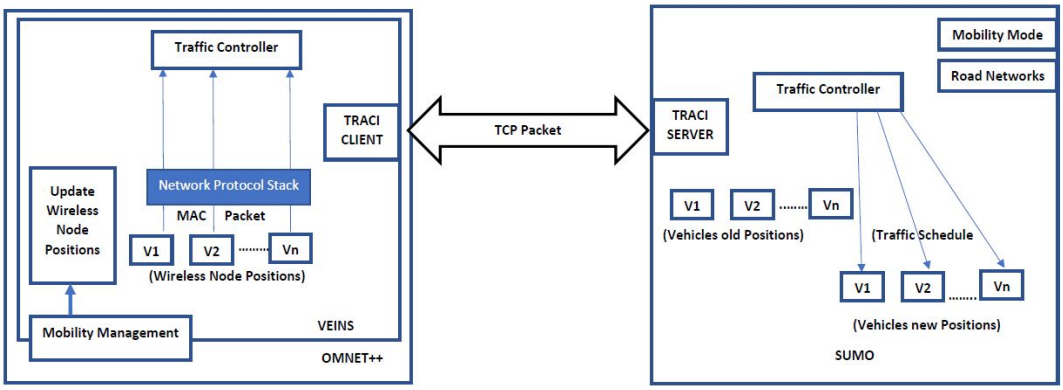

Figure 3.4: System diagram showing the connection between OMNET++ and SUMO

We have implemented FSA and other ATC algorithms using SUMO, OMNET++ and VEINS, as shown in Fig. 3.4. SUMO is a highly portable realistic open-source microscopic traffic simulation tool that is designed to handle large and small road networks. OMNET++ is a component-based, standard, and highly extensible framework used to design and build different network simulators. Specifically, we used VEINS, an OMNET++ extension for implementing IEEE 802.11p and IEEE 1609.4 VANET protocols. To access and control traffic network parameters in a SUMO simulation, we used TRACI, a client-server architecture that provides a programmatic interface to SUMO. SUMO/TRACI and OMNET++/VEINS are connected via a TCP/IP network protocol.

This integration of SUMO and OMNET++ achieves high fidelity between the traffic and network simulations in real-time. Each vehicle in SUMO is treated as a mobile node in OMNET++. The vehicular trajectory in SUMO is reflected as mobile node movement in OMNET++. A new node with a unique MAC ID is generated

in OMNET++ whenever a vehicle is created in SUMO, and it disappears when the corresponding vehicle reaches its destination. TRACI executes SUMO in discrete time steps, and the SUMO mobility manager module samples traffic information to send to OMNET++. OMNET++ mobility management then ensures that all mobile nodes are updated to their current positions.

3.5.1 Simulation Attributes

Multiple road network variables are used during the simulation. We have used the most common four-legged traffic intersection with four approaches and sixteen lanes. This results in eight different vehicular movements. Every approach has four uniquely identified lanes, such as `east_left_0, east_left_1, east_right_0, east_right_1,` etc. The saturation flow is set to 1800 pcu/hr/ln. Vehicular arrival is random and follows the Poisson distribution with an average vehicle flow rate λ. To assign a certain number of vehicles to a particular route, we assign probability values to the arrival rate.

We have divided our experiments into two basic categories. For *consistent* traffic distribution, the same value of λ is used for all four directions of the traffic intersection. Low (λ = 350 pcu/hr/ln), medium (λ = 750 pcu/hr/ln) and high (λ = 1800 pcu/hr/ln) values model off-peak, medium and peak hour distribution, respectively. For *inconsistent* distribution, different λ values are assigned to the North-South and East-West approaches such as (low, medium), (low, high), and (medium, high). We used different vehicle types to simulate real-world road networks.

Mobile nodes in VEINS contain information like Vehicle_ID, Lane_ID, current position, speed and time. IEEE 1609.4 short-range was modelled with default parameters: data rate of 6 Mbps with a carrier frequency of 5.9 GHz, five channels (1 CCH and 4 SCH), and beacon interval of 1s. The beacon length(B) was 400 Bytes, Service Packet Length (P) was 1000 Bytes, and transmission power was 100 mW.

3.6 Performance evaluation

3.6.1 Performance evaluation of FSA in terms of BI

The objective of FSA is to minimize average stretch experienced by the jobs being processed. Here, we calculate the average BI expressed in Equation. 3.4. which is analogous to stretch and compare it with OAF, ITLC and Webster's method for both consistent and inconsistent traffic distribution.

Firstly, we considered consistent traffic flow from all four directions towards the intersection, and we slowly change the traffic volumes over simulation time from low to medium, medium to high, and finally back to low (every 25 minutes). The effective green time for through traffic was considered as the 60s, 40s, and 30s for heavy, medium, and light traffic conditions, respectively. The right-turning traffic was allocated 50% of the effective green time. Since FSA is designed to optimize BI, it outperforms all the existing baseline algorithms. This signifies that the reliability of user towards FSA is higher than all other existing algorithms. Table. 3.1 depicts the calculated BI values for different traffic conditions. From the Table. 3.1 we can visualize that, user need to add additional 20% buffer time on their travel plan if the system is implemented with FSA whereas need to add 47%, 51% and 93% of extra time respectively for ITLC, OAF and Webster's method in case of consistent traffic flow of 350 pcu/ln/hr. Other results can be interpreted similarly.

Table 3.1: Performance of FSA in terms of Buffer Index

Algorithms	**Buffer Index**		
Consistent traffic	**350 pcu/ln/hr**	**750 pcu/ln/hr**	**1600 pcu/ln/hr**
FSA	1.2106	1.4159	2.4793
ITLC	1.4738	1.8207	2.5791
OAF	1.5127	1.8492	2.6853
Webster's	1.9369	2.1230	2.7529
Inconsistent traffic when N-S is 100 pcu/ln/hr and E-W varying	**upto 350 pcu/ln/hr**	**upto 750 pcu/ln/hr**	**upto 1600 pcu/ln/hr**
FSA	1.1358	1.1759	1.5923
ITLC	1.1924	1.3812	1.8625
OAF	1.2493	1.4216	1.9147
Webster's	1.4534	1.7686	2.2943
Inconsistent traffic when N-S is 750 pcu/ln/hr and E-W varying	**upto 350 pcu/ln/hr**	**upto 750 pcu/ln/hr**	**upto 1600 pcu/ln/hr**
FSA	1.2981	1.3629	1.9134
ITLC	1.6032	1.7416	2.2436
OAF	1.6981	1.8121	2.2912
Webster's	1.9753	2.0649	2.4782

We also calculated the standard deviation of the calculated BI for each vehicle to visualize the deviation of BI from the average. A lower value of standard deviation signifies that BI is grouped closer towards the average. From Table. 3.2 we deduce that for all traffic conditions FSA has a lesser value of standard deviation signifying that delay experienced by each vehicle is evenly distributed than other algorithms.

Table 3.2: Standard deviation of total Buffer Index

Algorithms	**Standard Deviation**		
Consistent traffic flow	**350 pcu/ln/hr**	**750 pcu/ln/hr**	**1600 pcu/ln/hr**
FSA	0.3676	0.4836	1.2290
ITLC	1.8345	2.3946	4.1791
OAF	2.1579	2.3842	4.5934
Webster's	2.9623	3.8929	7.9626
Inconsistent traffic when N-S is 100 pcu/ln/hr and E-W varying	**upto 350 pcu/ln/hr**	**upto 750 pcu/ln/hr**	**upto 1600 pcu/ln/hr**
FSA	0.4136	0.5297	1.3823
ITLC	1.9786	2.9142	5.6152
OAF	2.7685	3.1562	6.1287
Webster's	3.1249	4.1672	9.1563
Inconsistent traffic when N-S is 750 pcu/ln/hr and E-W varying	**upto 350 pcu/ln/hr**	**upto 750 pcu/ln/hr**	**upto 1600 pcu/ln/hr**
FSA	1.8126	2.3615	3.1481
ITLC	3.7645	4.9372	8.1176
OAF	4.1932	5.9126	9.3438
Webster's	7.1582	9.4378	12.1176

3.6.2 Performance Evaluation of FSA in terms of Waiting time and Throughput

The sole aim of designing FSA for a traffic intersection is to avoid the unfair nature of existing algorithms that minimizes delay (OAF) and maximizes throughput (ITLC), as discussed earlier. The results tabulated above justifies FSA is a more reliable and fair algorithm with BI as an optimization parameter. Contrary to this, it is equally important to visualize the performance of FSA with other objective functions. We first compare the total queue delay and throughput of FSA with OAF and ITLC. OAF

claims it is an optimal algorithm to minimize queue delay and ITLC claims maximum throughput. Both techniques claim superiority over all other ATC algorithms to date. To ascertain that FSA performs well as compared to OAF and ITLC, we crafted a simple Kolmogorov–Smirnov (KS) test. The Kolmogorov–Smirnov (KS) test quantifies the difference between the empirical distribution function of two samples, to check the basic difference between two samples of single-dimensional probability distribution (Massey Jr, 1951).

$$D_{n,m} = sup_x |F_{1,n}(x) - F_{2,m}(x)| \tag{3.9}$$

$F_{1,n}$ and $F_{2,m}$ are the empirical distribution functions of two samples and sup is the supremum function.

To establish if FSA is fair to all jobs, we require a fairness index calculated as a finite and continuous value, independent of population size and measurement metric. We chose Jain's fairness index (Jain, Chiu & Hawe, 1984), a popular fairness metric.

$$f(X) = \frac{[\sum_{n=1}^{n} x_i]^2}{n \sum_{n=1}^{n} {x_i}^2} \tag{3.10}$$

While measuring the fairness index with respect to waiting time, x is the normalized waiting time of i^{th} job and n is the number of jobs and $0 \leq f(X) \leq 1$. A higher value of f(X) implies fairer resource allocation.

Evaluation of FSA during consistent traffic conditions

The performance of FSA in comparison with OAF and ITLC is shown in Fig. 3.5. The performance parameter we compared was the average delay experienced by each vehicle in a 5 minutes interval. The labels low, medium, and high represent traffic volumes during different simulation times.

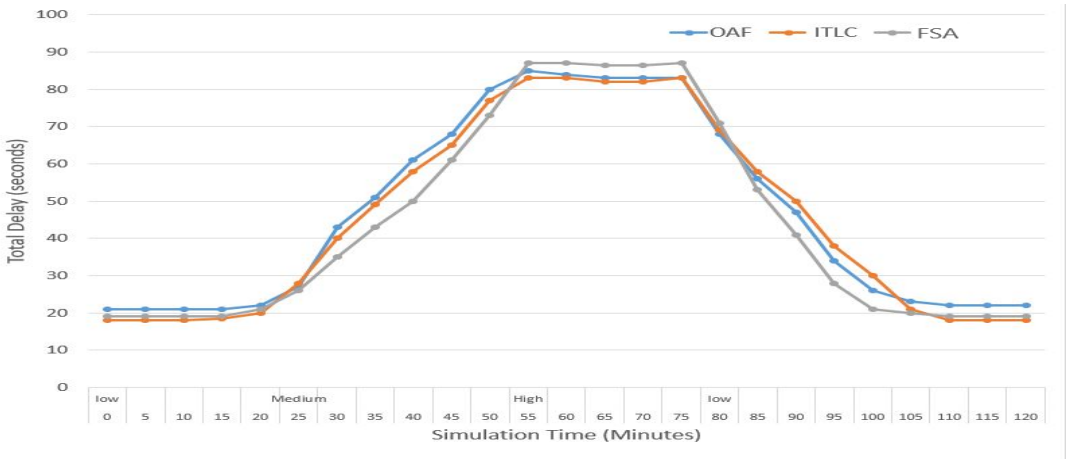

Figure 3.5: Performance of FSA, OAF, and ITLC in terms of total delay.

During initial and final low traffic volumes, the performance of FSA in terms of average delay experienced by each vehicle lies between OAF and ITLC. When traffic volume starts building to medium and then finally to high, FSA maintains the smallest delay. The rate of change of delay time with traffic volume for FSA is less when congestion is building and more while discharging the congestion. This signifies that FSA performs better in resisting and discharging congestion.

We also evaluated the performance of FSA in terms of throughput. Fig. 3.6 represents the total number of vehicles processed by each of these algorithms within a second.

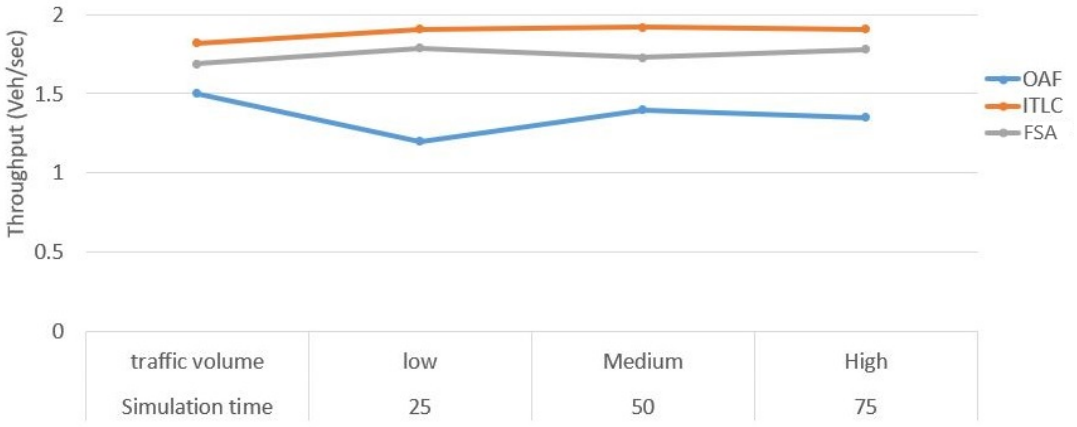

Figure 3.6: Performance of FSA, OAF, and ITLC in terms of throughput

Evaluation of FSA during inconsistent traffic conditions

This is where we wanted to see the performance of FSA. Uneven distribution of traffic flow generates variable size platoons, and the application of FSA fits right. In this experiment as shown in Fig. 3.7 and Fig. 3.8 we setup constant traffic flow from north-to-south at 100 pcu/hr/ln and 750 pcu/hr/ln and vary the traffic flow of east-to-west from low (350 pcu/hr/ln), medium (750 pcu/hr/ln) and high (1800 pcu/hr/ln). The average

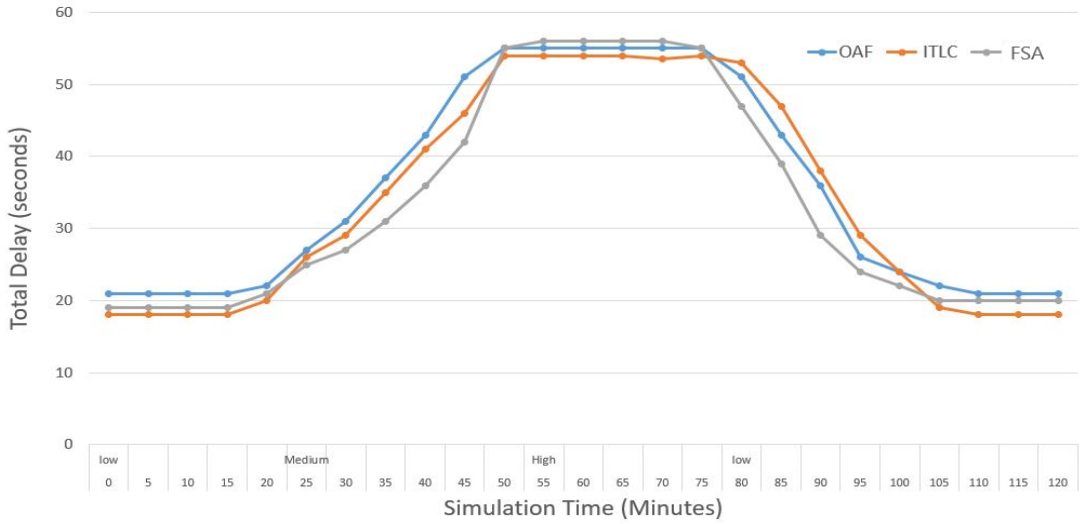

Figure 3.7: Performance of FSA compared to OAF and ITLC when traffic from north-to-south is 100 pcu/hr/ln and traffic from east-to-west is changing

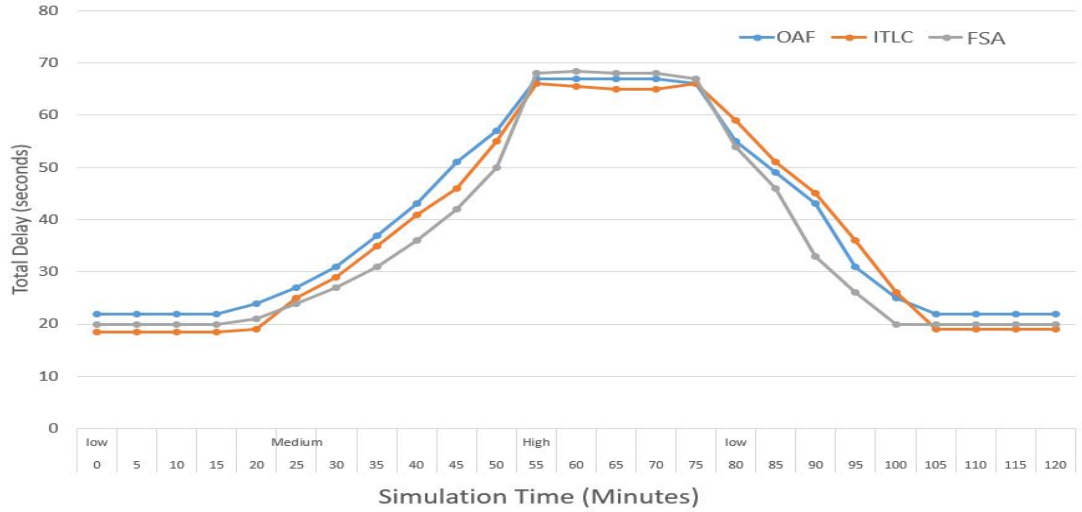

Figure 3.8: Performance of FSA compared to OAF and ITLC when traffic from north-to-south is 750 pcu/hr/ln and traffic from east-to-west is changing

delay per vehicle was the same comparison parameter. We can notice that the total delay of all the algorithms reduces and FSA builds up congestion slower than OAF and ITLC and recovers from the congestion faster than both of these algorithms. This is because FSA takes advantage of gaps between the vehicles (that may trigger phase change occurring additional delay) by converting it into variable-sized platoons from different approaches. This delay then gets distributed equally among all the vehicles within that platoon. Besides, FSA processes a platoon with certain processing time until its completion before it switches to another task. This property makes FSA more efficient in resisting and discharging congestion.

From Fig. 3.5, 3.7, and 3.8, we can observe the FSA outperforms both OAF and ITLC while resisting and flushing congestion but at high traffic volume (plateau in the figures) it looked FSA is slightly lagging. To examine this issue, we plotted a cumulative difference plot, as suggested in the KS test for plateaued values. The difference curve presented in Figure. 3.9 confirms that there is no statistically significant difference between total delay time among all three algorithms at high traffic penetration rates.

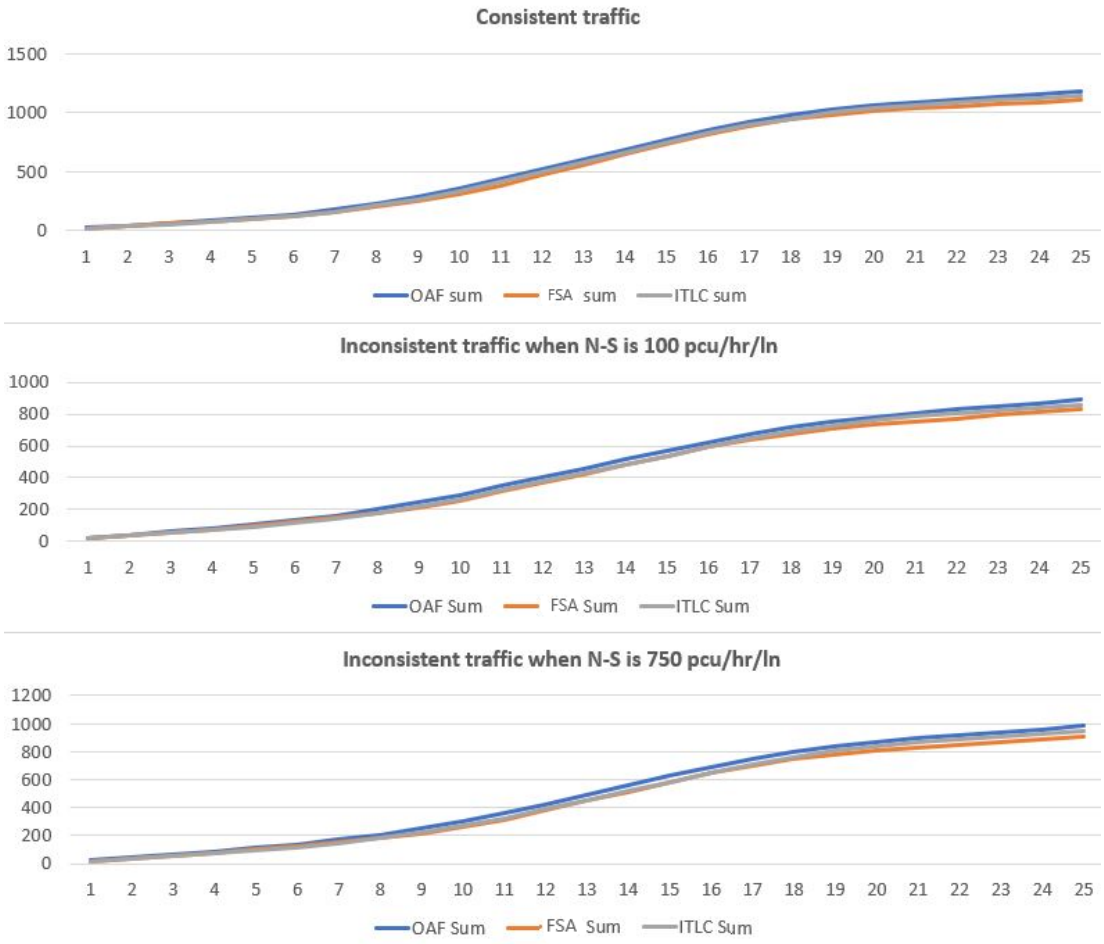

Figure 3.9: Visualization of the Kolmogorov–Smirnov (KS) test showing the difference in total delay

3.6.3 Evaluation of FSA in terms of Jain's fairness index

Table. 3.3 shows the calculated Jain's fairness index of FSA, OAF and ITLC, for both consistent and inconsistent traffic flows for the experiments discussed earlier. Though the overall fairness index for all the algorithms looks comparable for consistent and inconsistent traffic situations, FSA has a significantly higher value of fairness index for approaches with lower traffic volumes than OAF and ITLC. These results provide convincing evidence that tasks from approaches with lower traffic density are treated fairer by FSA than OAF and ITLC. So if we implement FSA at intersections with uneven traffic distribution, it eliminates vehicles from waiting unnecessarily.

Table 3.3: Jain's fairness index for different traffic flows

Different Algorithms	**Jain's Index**
Consistent traffic flow	
FSA f(X)	**0.9986**
OAF f(X)	0.9951
ITLC f(X)	0.9964
Inconsistent traffic when N-S is 100 pcu/ln/hr and E-W varying	
FSA f(X)	**0.6311**
OAF f(X)	0.5255
ITLC f(X)	0.5327
Inconsistent traffic when N-S is 750 pcu/ln/hr and E-W varying	
FSA f(X)	**0.9947**
OAF f(X)	0.9478
ITLC f(X)	0.9240

3.7 Conclusion

In this paper, we proposed an adaptive Fair Scheduling Algorithm (FSA) for VANETs that ascertains higher travel time reliability by minimizing the Travel Time Index (TTI) for an isolated intersection. We considered Buffer Index (BI), a typical travel time reliability measure, as the primary performance parameter to minimize. We achieved this through a novel approach of analogically mapping traffic control problems into real-time systems precisely mapping TTI with stretch (the factor by which a job is slowed down comparing with time it takes to process on a free system). We first proved that stretch produced by FSA is less than or equal to twice the stretch produced by an optimal offline algorithm implying FSA is *2-competitive*. Then we empirically

prove that FSA online algorithm is more reliable and fairer in scheduling in terms of travel time with Buffer Index (BI) as an optimization parameter compared to existing state-of-art approaches. Additionally, FSA achieves equivalent performance with delay minimizing and throughput maximizing baseline algorithms.

In future, we would like to implement FSA in a real-world trip data-sets so that other dynamic parameters that affect travel time reliability like time of day, week, driving behaviour and weather conditions are already considered. This allows us to visualize the sole effect of adaptive FSA on travel time reliability.

Chapter 4

Routing Emergency Vehicles in Arterial Road Networks using Real-time Mixed-Criticality Systems (Manuscript 3)

The manuscript titled "Routing Emergency Vehicles in Arterial Road Networks using Real-time Mixed-Criticality Systems" try to solve ever-existing two problems caused by the pre-emption of emergency vehicles (EVs). Although pre-emption techniques have been studied extensively, emergency management service providers always struggle to meet target response time. Second, a substantial cost of pre-emption that normal traffic has to handle in terms of waiting. We postulated a novel emergency vehicle pre-emption algorithm implemented in the Vehicular ad-hoc network to solve these issues. We introduced different criticality levels for different levels of emergencies and assigned a certain level of success assurance in terms of target travel time for these criticalities. Unlike other studies, instead of implementing the EVP algorithm in a single intersection, we implemented it in an arterial traffic network. We ran exhaustive simulations. The

results indicate that EVP algorithm can significantly reduce the average waiting time of normal traffic but still assures all EVs meet their target response time. The manuscript 3 is attached within this section.

Routing Emergency Vehicles in Arterial Road Networks using Real-time Mixed Criticality Systems*

Subash Humagain and Roopak Sinha, Senior Member, IEEE Department of Information Technology and Software Engineering, Auckland University of Technology Auckland, New Zealand

AUT Tower, Level 7, 2-14 Wakefield Street, Auckland, New Zealand

Tel: +64-9-921 9999 Ext.6376;

corresponding author: subash.humagain@aut.ac.nz

4.1 Introduction

Reducing the response time of Emergency Vehicles (EVs) has an enormous impact on saving life and property. According to a study conducted by RapidSOS in the USA, every single minute of delay in response time increases mortality by 1% and incurs 7 billion dollars of extra healthcare expenses (RapidSOS, 2015). To mitigate this issue, governments impose target response times for Emergency Management Services (EMS). For instance, 90% of critical emergency calls must be responded to within 9 minutes in the USA (Pons & Markovchick, 2002a), while in the UK 75% of such cases must be responded to within 8 minutes (*Ambulance Quality Indicators Data 2019-20*, 2019). For Australia and New Zealand, the target is to respond to 50% of emergency calls within 8 and 10 minutes, respectively (*Annual report 2019*, 2019; *NSW state emergency service annual report 2015*, 2015). Due to increasing pedestrian population and congested road networks it has become an increasingly difficult challenge for EMS to meet contractual timings.

Researchers and industry practitioners have examined the problem of reducing

response time for EVs extensively and have proposed or implemented different solutions. Current solutions offer either used route optimization, signal pre-emption or both techniques to reduce the response time of EVs (Humagain et al., 2020). Route optimization selects the fastest route for EVs within the available circumstances. Traffic pre-emption modifies traffic flow to prioritize selected vehicles like EVs. The most widely used pre-emption technique is altering traffic signals to halt normal traffic flow and provide passage to EVs (Gedawy et al., 2008). Present pre-emption techniques use GPS, localized radio, acoustic or line of sight sensors to activate pre-emption. Emergency Vehicle Pre-emption (EVP) has aided in reducing the response time of EV and served in saving the life from fatalities. Kamalanathsharma and Hancock in their study have shown that EVP can achieve savings of up to 31% in travel times compared to the system without EVP (Kamalanathsharma & Hancock, 2012). Though the performance of EVP in reducing EV response time is compelling, its consequences over general traffic (unnecessary waiting time) cannot be overlooked.

A Vehicular Ad-hoc Network (VANET) can aid in optimizing the time of triggering and ceasing of EVP. VANET is the wireless ad-hoc network created by moving vehicles where vehicle communicate with other vehicles or infrastructure using dedicated short-range communication standards outlined by IEEE 802.11p (F. Li & Wang, 2007). In VANET roadside infrastructures like traffic controllers and vehicles can share valuable information like speed, position, lane, route, and time of arrival, which are fundamental in optimizing green time available for EVs. This, in turn, can reduce the negative effects EVP may have over general traffic. Several studies have implemented VANET in sharing the current position of EV to determine the right time to trigger and cease EVP (Agarwal & Paruchuri, 2016; Jayaraj & Hemanath, 2015; Kamalanathsharma & Hancock, 2012). Unibaso et al. used standard Cooperative Awareness Message (CAM) defined by the European intelligent transportation system in their traffic control algorithm to provide a green light to EVs(Unibaso et al., 2010). Similarly, Walz and

Behrisch also used CAM to provide a green wave to EV travelling through multiple intersections (Bieker-Walz & Behrisch, 2019). Jordan and Cetin implemented VANET for efficient passage of EVs in closely spaced traffic intersections by preempting traffic signals in a definite order so that existing traffic within these intersections can be discharged prior to the arrival of EV (Jordan & Cetin, 2015). In (Younes & Boukerche, 2018), a dynamic traffic scheduling algorithm was designed that can fine-tune the green time allocated for emergency vehicles so that there is a lesser impact on normal traffic.

Pre-emption techniques have been studied widely for EV routing, but EMS companies are always struggling to achieve the target response times. This is because almost all cutting edge research and implementations of pre-emption techniques focus on reducing the travel time of an individual EV in an optimized route providing that EV with the highest priority. But in real-life city traffic and in case of natural calamities, there can be multiple EVs servicing different levels of emergencies at the same time. The approach of prioritizing a single EV can reduce the travelling time of that particular EV, but does not assist the EMS in meeting the target response times for all EVs on the ground. Moroi and Takami have also pointed towards the limitations of prioritizing single EVs in case of emergencies where we may require to route multiple EVs at the same time (Moroi & Takami, 2015). In such cases, other EVs following a recently prioritized EV experience more congestion. Furthermore, providing absolute priority to an individual EV serving any level of emergency increases the overall waiting times for normal traffic. This problem can be solved if multiple EVs serving within a particular time are assigned with different levels of priority and EVP is performed accordingly.

In this study, we propose an EVP technique for multiple EVs with different levels of priority. We leverage the use of VANET for transmitting critical information like current position, speed, the time EV takes to pass the intersection, and the route the EV follows in deciding when to trigger the EVP. Important decision parameters like speed, position and route of vehicles are easier to access using VANET which is impossible

from traditionally used inductive loops (Maslekar et al., 2011). There can be conflict in prioritizing EVs when they share the same route or some common intersections within that route. This issue is solved by assigning different levels of priorities to EVs. Assigning a different level of priorities to EVs is a common practice in EMS industry, for example, St. John's in New Zealand classify serious life-threatening incidents as purple, red and, and less serious emergencies as orange. Without affecting the generality of our work, we use the New Zealand system for illustration of concepts presented in this paper.

The EVP technique proposed is novel as it provides a way to arbitrate multiple EVs in an arterial road network. An arterial road network is the backbone of any urban road that is usually used to transport traffic from small collector roads to expressways or motorways. Rather than implementing EVP in a single intersection, modelling and implementing EVP for an arterial road network allows for real-world visualization and more efficient EV routing. Performance parameters like waiting times and throughput achieved from such implementations are realistic and will be helpful for EMS to evaluate their actual performance. In addition, it also helps traffic engineers and transport planners to anticipate the widespread effect of EVP over general traffic flows rather than limiting such explorations to a single intersection. Moreover, adding multiple EVs into the EVP model can solve unsolved problems faced by EMS in prioritizing EVs when two or more of them have conflicts passing an intersection.

The proposed traffic control algorithm designed for EVP of multiple vehicles is constructed using a novel approach of mapping traffic control domain into Real-Time Mixed-Criticality Systems (RTMCS) task scheduling. In real-time systems, the accuracy of jobs being processed does not rely solely on correct functionality but also on the timely completion of tasks and computations. In RTMCS, jobs with multiple levels of criticality like mission-critical, non-critical and safety-critical are designated with different timing constraints. The system is considered a failure if it cannot meet these

timing values (Ernst & Di Natale, 2016). We map EV priority levels (purple, red and orange) with different levels of criticality in RTMCS and impose EVP system to ascertain a certain level of success assurance against a different level of criticality.

The arbitration of conflicts that arise from scheduling equal priority EVs through the same intersection is equivalent to the well-known problem of scheduling with conflicts in real-time systems and can be resolved using conflicts graphs. Whenever two or more EVs must pass via a single traffic intersection, they cannot be scheduled simultaneously if they are travelling in conflicting directions. This problem of resource sharing in real-time systems is dealt through modelling a conflict graph (Irani & Leung, 1996). We model and utilize conflict graphs to create an optimal schedule for EVs in traffic intersections. As EV arrival at an intersection is a random event, this paper proposes an *online* scheduling algorithm.

The implementation of the proposed EVP in an arterial traffic network for multiple EVs with different levels of priorities shows promising results. We conducted multiple experiments using the microscopic traffic simulator Simulation of Urban Mobility (SUMO) (Krajzewicz, Hertkorn, Rössel & Wagner, 2002). Simulation results show that the waiting times and overall travel times are significantly reduced when EVP is enabled, as compared to traditional static traffic light control. Moreover, the EVP algorithm seems to also reduce waiting times for non-EV traffic when compared to absolute pre-emption.

4.2 System Model

A four-legged isolated traffic intersection with adaptive traffic control system implemented in a VANET environment can be embedded into an arterial road network with multiple intersections. Let us consider a four-legged traffic intersection designed for left-hand driving as illustrated in the left side of Fig. 4.1. It consists of four road

segments from North, South, East, and West directions. Each road segment is divided into two approaches so that this intersection has a total of eight approaches. Every single approach has conflict in movement with five other approaches and has only two non-conflicting approaches. For example, traffic from approach 2 cannot be scheduled with traffic form approaches 3, 4, 5, 7 and 8, but it can still be scheduled with traffic from approaches 1 and 6.

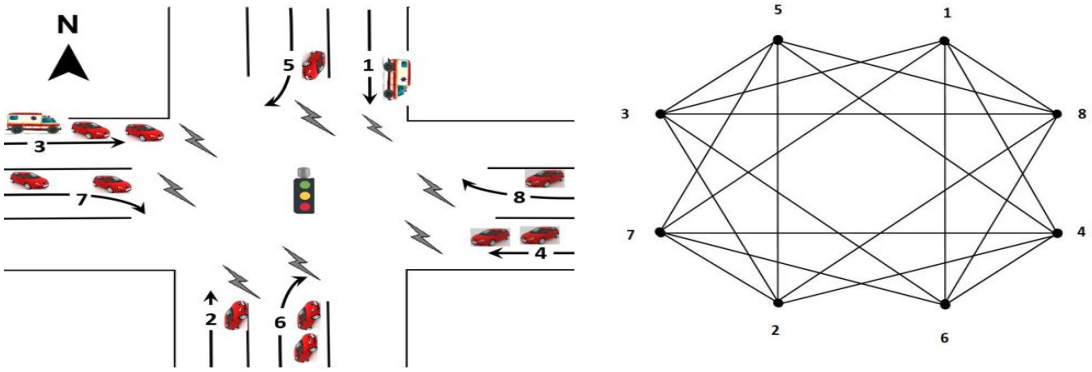

Figure 4.1: Four-way traffic intersection and its equivalent conflict graph

A conflict graph is used to model concurrency in simultaneous operations in real-time systems and can be used in traffic intersections. A traffic signal must always schedule non-conflicting approaches together. This can be analogically mapped to task scheduling in a real-time system which can be implemented through conflict graphs. A conflict graph with nodes N and edge E is represented as $G = (N, E)$ where each approach from traffic intersection is a node (in N) and an edge (in $E \subseteq N \times N$) is a connection between two conflicting approaches as depicted in the right side of Fig. 4.1. This approach can be generalized to any number or types of approaches. Our traffic scheduling algorithm schedules traffic by picking a set of non-conflicting approaches at every instance and allowing it appropriate green time.

VANET provides precious data to activate efficient EVP. VANET is a subset of Mobile Ad-hoc Network (MANET) designed especially for vehicular communication. The network topology of MANET changes rapidly as nodes are permitted to move in

any direction. Routing protocols for MANET are unable to provide optimum throughput for complex trails created by rapid change in relative positions of moving nodes due to the high speeds of vehicles in a predefined road. Therefore, VANET has become the technology of choice in such circumstances (Ranjan & Ahirwar, 2011). In this study, we assume that all EVs are equipped with Dedicated Short-Range Communication (DSRC) devices for the vehicle to infrastructure communication and GPS to collect speed and position information. Information about the route EVs follow from origin to the destination is initially set by the routing application installed within the vehicle. The communication between EVs and the road-side traffic controller is carried out in 5.9 GHz spectrum bandwidth for DSRC as standardized by IEEE 1609.4 protocol. The traffic controller receives beacons transmitted from EVs, which contain information like speed, position and route of an EV. Knowing the route before the activation of EVP allows the controller to decide which approach must be allowed to provide EVs right of way. Speed and position data allows determining exactly when to switch the current traffic signal phase to green or by what time the current green phase should be extended. The current speed of EV replicates dynamic traffic behaviour of that particular lane, and for example, a congested lane needs the controller to reserve a longer green time than a free lane.

A single intersection EVP approach is insufficient. Calculation of exact green time required for an EV to pass through an intersection is very crucial as allowing unnecessarily extended green time negatively affects the flow of general traffic waiting to pass through the intersection. In order to ensure that such issues are minimized, we calculate the exact time the traffic controller needs to reserve for the EV in real time so that it can pass the intersection within this time using information like current speed and position transmitted by the EV. Allowing a well-timed green signal for a particular intersection still cannot guarantee that the EV can meet the contractual time to reach a destination. So we have designed our system to work on an arterial road network

where an EV has a source and a destination. The fastest route is initially assigned to an EV. This route contains multiple intersections and a distance to cover. The contractual target time can only be ascertained if the EV maintains the maximum allowed speed throughout the journey. The main aim of the EV traffic control algorithm installed in each traffic controller is to allow the EV to maintain this speed, as much as possible. We take advantage of the information beacon transmitted by EVs to maximize this possibility. Additionally, to replicate the real-world situations, we introduce multiple EVs in an arterial network at the same time and study their routing.

Assigning a different level of criticality to EVs and implementing EVP accordingly is less disruptive to normal traffic. When EVs are serving emergencies, they travel with sirens and lights on. Current EVP systems provide a green signal to EVs when they identify an EV's presence through GPS, localized radio, acoustic or line of sight sensors. Once the presence of EVs is confirmed, they get absolute priority throughout the route. This is more disruptive to normal traffic, especially at intersections. In most cases, all EVs are not serving to the same levels of emergency. So instead of providing absolute priority at all intersections we can use a mixed model of EVP. In our experiment, we have assigned three levels of criticality to EVs. Our algorithm now schedules EVs with different levels of criticality and ascertains the times they reach the destination and if this time can be within a set response time. In doing so, the algorithm identifies the intersections it needs to preempt and leaves other intersection unaffected. This approach massively reduces the effect of EVP on normal traffic.

Scheduling EVs with different levels of criticality is analogous to RTMCS. In RTMCS the calculation of Worst-Case Execution Time (WCET) depends on the criticality of the task. For example, safety-critical tasks have lowest WCET than mission-critical tasks, and a non-critical task has the highest WCET (Burns & Davis, 2013). This aligns exactly with our case of multiple EVs with different levels of criticality. Standard response times of 8, 12 and 16 minutes for high, medium and low critical cases of

emergencies imply medium and low criticality cases are respectively assigned response times that are 1.5 and 2 times higher than high criticality cases. We consider these factors as assurance levels $L_i = \{l_1, l_2, l_3, .., l_n\}$ be a set of assurance levels, which is defined for each EV . In this study, we define target travel time at any instant i as Tt_i for an EV as follows.

$$Tt_i = (D_i/V_{max}) * L_i \tag{4.1}$$

Where D_i is the distance between the current EV location and the target location and V_{max} is the speed limit of the particular lane. We also define current travel time for any instant i as Tc_i and calculated as

$$Tc_i = (D_i/V_{EV}) \tag{4.2}$$

Where V_{EV} is the current speed of EV. Also, the relative difference in time ΔT is calculated as

$$\Delta T = Tc_i - Tt_i \tag{4.3}$$

From equation. 4.3 if the $\Delta T \leq 0$ then the EV can reach its destination within its target travel time without triggering EVP. When $\Delta T > 0$, it implies the route of EV is congested and EV cannot reach the destination within its target travel time and hence we may need to trigger EVP immediately.

The overall working of the system is depicted in Fig. 4.2. Consider the situation where we have three EVs, EV1, EV2 and EV3, serving at the same time with three different levels of criticality. This algorithm can perform equally well for more than three EVs and more than three levels of criticality. Initially, all the EVs are assigned with routes. Whenever an EV approaches an intersection, it sends information about its route (origin and destination), current position, criticality level and speed to the traffic

controller. The traffic controller first checks if there is any other EV approaching the same intersection. If there is more than one EV approaching the same intersection controller checks the level of criticality of EV and processes the request of EV with higher priority else it processes the current request. When multiple EVs approaching the same intersection have the same level of criticality, EV with a higher value of ΔT is prioritised. And in a case where two vehicles have the same level of criticality and equal value of ΔT the priority is given to EV approaching from the right. It then calculates the value of ΔT and determines the requirement of EVP. When pre-emption is required, the controller calculates the reserve green time required for each EV to pass the intersection from the position data. If the current state of the traffic light is green, it extends current green time with reserve green time else it alters the current phase and provides green phase to EV. The EVP algorithm is shown in Algorithm 3. Parameters like $EV_i.position$, $EV_i.speed$, $EV_i.criticality$, and $EV_i.destination$ can be set as vehicle parameters in SUMO and are accessible during the simulation.

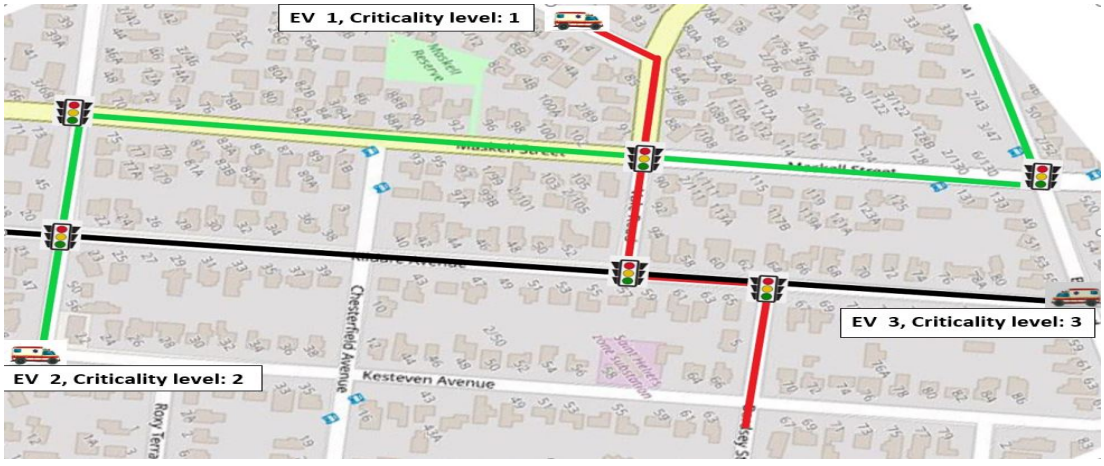

Figure 4.2: System Model with multiple intersections and EVs

Algorithm 3 Emergency Vehicle pre-emption

Input: Emergency vehicles $EV_i(1, 2, ..., n)$
Output: Alter traffic signal
1: **for** each EV_i **do**
2: find EV_{min} with minimum $EV_{min}.criticality$;
3: ΔT = calculate $\Delta T(EV_{min})$;
4: **if** ($\Delta T \leq 0$) **then**
5: Reserve time = calculateReservetime(EV_{min});
6: action = alterTrafficLight(Reserve time);
7: **else**
8: No action;

4.3 Implementation

This section illustrates the implementation of the proposed EVP algorithm using SUMO, OMNET++ and VEINS. SUMO is a widely-used and open-source realistic microscopic traffic simulator. OMNET++ is a simulation library designed to implement different network simulations. We use VEINS as a platform to couple SUMO and OMNET++ to simulate vehicular communication using IEEE 802.11p. Full-Duplex communication is established between SUMO and OMNET++ by VEINS using Transmission Control Protocol (TCP), which helps in examining how VANET implementations affect road networks (Arellano & Mahgoub, 2013).

Simulation parameters in SUMO can be altered using TRACI when the simulation is live. SUMO gathers prior information from its different components and runs the simulation continuously until it produces the result. The parameters of the simulation cannot be altered during the course of the simulation. To access the simulation parameters and alter its course, we use an extension tool developed in Python called TRACI. Different applications can communicate and control SUMO during simulation via client-server architecture using TRACI. All the communicating applications are set as client and SUMO acts as a server. They use TCP sockets to communicate with each other using TCP/IP protocol. Client applications send TCP packets to SUMO requesting to alter or change parameters like altering traffic light logic, changing vehicle route,

lane, destination etc. In response to the client's request, SUMO alters these parameters and replies back to the client application. TRACI also helps to establish communication between traffic simulator SUMO and the network simulator OMNET++.

SUMO and OMNET++ run concurrently. Each vehicle in SUMO is treated as a moving node in OMNET++. A mobile node gets created once a vehicle appears in the road network and disappears when the vehicle reaches its destination. Every new node created in OMNET++ has a unique MAC ID as a vehicle in SUMO has $vehicle\_ID$. We can use information transmitted over the network and make preemptive decisions and implement it by changing the traffic light signal phases.

4.3.1 Simulation parameters

Realistic traffic parameters are utilized for simulation-based validation of the proposed solution. We conduct our experiment over a section of Auckland city's arterial road network and calibrate the network with vehicular data. All intersections are initially modelled with static Traffic Light Control (TLC) logic. Traffic saturation is set to 1800 pcu/hr for every lane. Vehicles arrive randomly following the Poisson distribution. To achieve variable vehicular flows in a particular lane, we assign probability values for arrival rates. The car-following model is set to Krauss and parameter sigma is set to 0.5. For EVs vehicle class ($vClass$) is set to "emergency". We use New Zealand-based emergency codes purple, red and orange for assigning high, medium and low levels of criticality to EVs, respectively. EVP algorithm checks the presence of EVs within 200m range of the traffic intersection and refreshes all variables every 15 seconds. Once EVP is activated TRACI controls the simulation and alters traffic light phases to provide green waves for EVs as required.

The primary aim of our algorithm is to ascertain that EVs with different levels of criticality reach their destinations within desired times. EVs are created randomly with

randomly assigned criticality. We change the traffic volumes in the route of EV from low (450 pcu/hr/ln), medium (850 pcu/hr/ln) and high (1800 pcu/hr/ln) and for each setting, we perform simulations multiple times. In all cases, the EVs are generated randomly, assigned with random routes and criticality. According to an annual report published by St. John New Zealand, out of 546,000 emergency calls 63.6% cases were highly critical, 23.2% cases were medium and the remaining 13.3% of the cases were low critical (*Annual report 2019*, 2019). In our experiments as well, we have maintained the same proportions of the levels of criticality. We have considered 1% of total vehicles generated as EVs for a simulation time of 10,000 seconds and with longer duration of simulation time a lesser percentage of EVs can be considered. We measure the number of times EVs met the target travel time of 8, 12, and 20 minutes for purple, red and orange level of criticality respectively in terms of the success rate of EVP. We also measure the average waiting time of both non-EVs and EVs, queue length, and overall throughput in a system implemented with our algorithm and compare it with systems without pre-emption and absolute pre-emption (EV receives green phase once detected near intersection until it leaves).

4.4 Performance Evaluation

Our EVP implementation shows promising results when compared to traditional systems containing either no pre-emption or absolute pre-emption. As claimed in earlier sections, the impact of pre-emption is huge in normal vehicles. The claim made by other pre-emption technique that has been implemented in a single intersection cannot justify a real-world situation where multiple EVs move from source to the destination covering multiple intersections. The cost incurred by increased waiting times of normal vehicles when pre-emption is calculated in terms of CO2 emissions and the corresponding price of fuel used, which are high for the traditional settings and relatively lower in

the proposed EVP implementation. Our experiments study five important metrics: the percentage of EVs meeting their target travel time successfully, average waiting times for both EVs and non-EVs, average queue lengths at intersections and overall throughput of the network.

Fig. 4.3 shows the percentage of success EVs achieve with the proposed EVP algorithm in low, medium and high traffic densities. Whenever an EV is lagging behind to meet its target travel time, pre-emption is activated, which means that success percentage is expected to be 100%. However, after taking into account traffic uncertainties as modelled in SUMO, the overall success percentage is lower, especially in peak traffic. All simulations reported here ran for 36000 seconds, and in some cases, a longer duration of simulation gives almost perfect results. Our result depicts that at higher traffic densities few EVs struggle to meet target response time but still maintain promising 96% success. Some EVs with a high level of criticality, miss target response time, as their number is very high as compared to EVs with other criticality levels but still maintain exuberant 95% success.

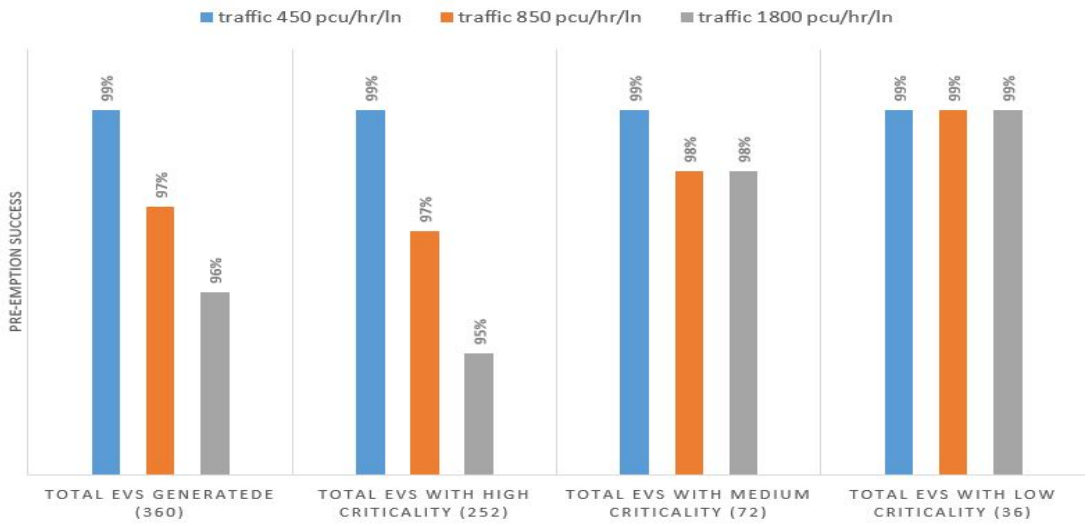

Figure 4.3: Success percentage of EVs meeting target travel time

Pre-emption affects normal traffic by increasing their average waiting times. Absolute pre-emption incurs a very high waiting time for non-EVs resulting in huge financial losses. A system without pre-emption becomes an obstacle to EVs. We implemented a practical approach where EVs are assigned with different level of criticality and imposed different target travel times. The EVP algorithm ascertains that EVs meet these target travel times. At different traffic penetration rates, the average waiting time of non-Evs is reduced drastically, almost comparable to the system without pre-emption. This ensures that the implementation of EVP algorithm can have huge perennial savings in terms of fuel cost and CO2 emission. The results are depicted in Fig. 4.4.

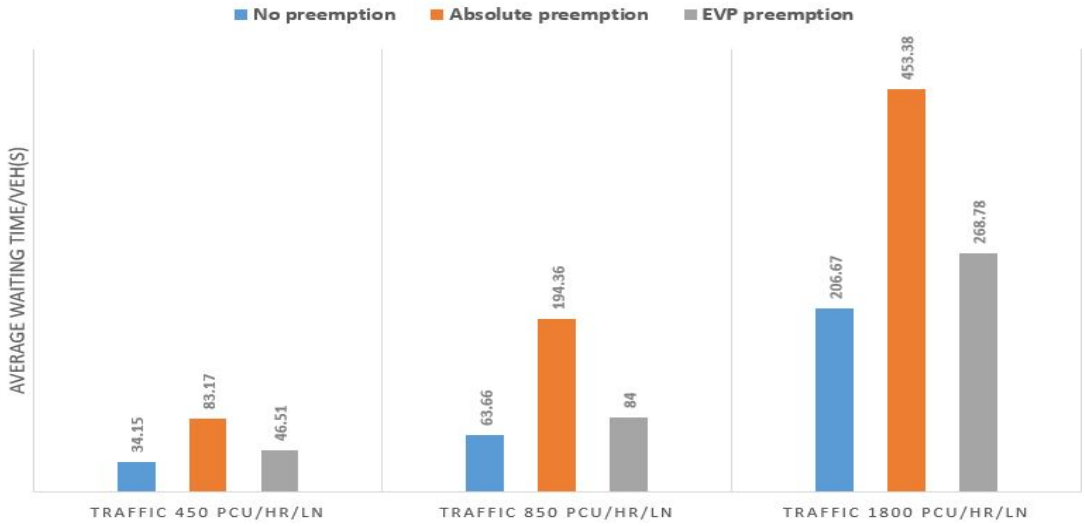

Figure 4.4: Average waiting time of non-EVs

Since we used a mixed model in making decisions to activate pre-emption, EVs routed using EVP algorithms experienced slightly increased waiting time as compared to a system with absolute pre-emption but performed exceptionally well than the system with no pre-emption as shown in Fig 4.5.

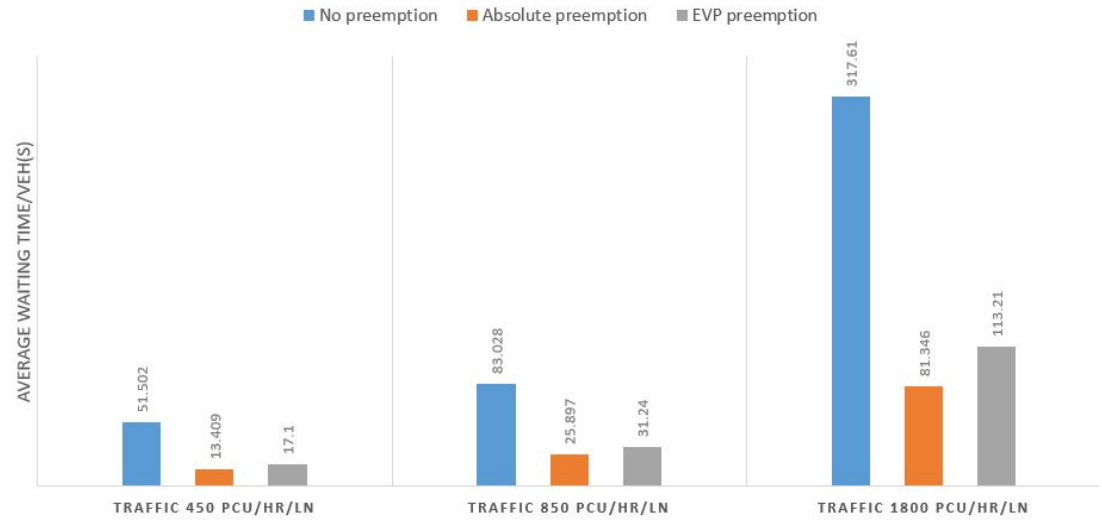

Figure 4.5: Average waiting time of EVs

Queue length for an intersection measures the efficiency of any traffic control algorithm. It is measured as the average number of vehicles waiting at an intersection. The EVP algorithm's impact on this metric is shown in Fig. 4.6. The experimental results show that average queue length is reduced up to 36% using EVP algorithm as compared to absolute pre-emption and still is comparable to the system with no pre-emption.

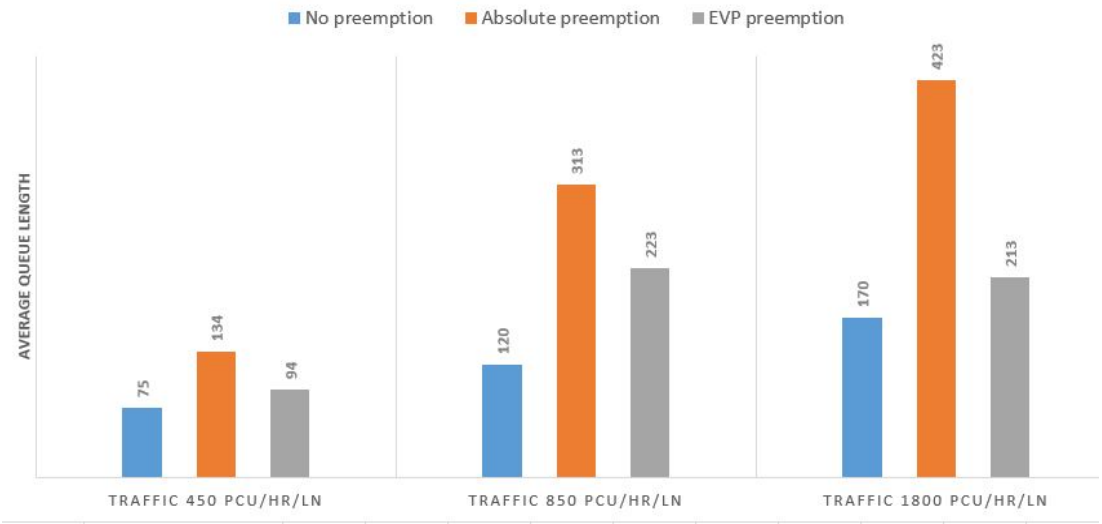

Figure 4.6: Queue length comparisons

The total traffic volume an intersection can flush through is measured in terms of throughput. This well-known productivity indicator is used to compare EVP, absolute pre-emption and no pre-emption in Fig. 4.7. The results illustrate that the throughput of EVP implemented system is high and is comparable to the system without pre-emption. The system with absolute pre-emption has the lowest throughput.

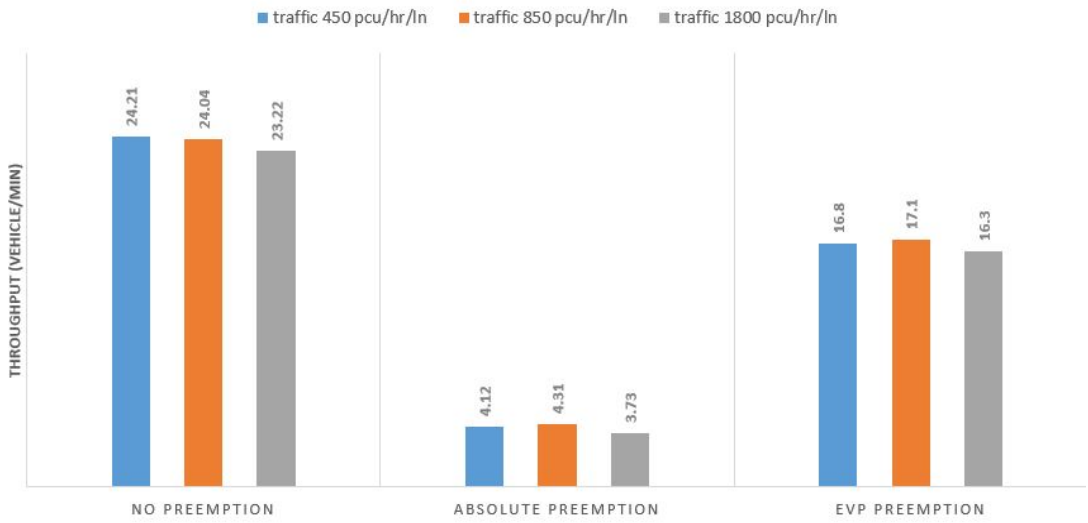

Figure 4.7: Throughput comparison

4.5 Conclusion and Future Work

In this study, we try to solve ever-existing two problems caused by the pre-emption of emergency vehicles (EVs). First, though pre-emption techniques have been studied extensively, emergency management service providers always struggle to meet target response time. Second, a substantial cost of pre-emption that normal traffic has to handle in terms of waiting. We postulated a novel emergency vehicle pre-emption (EVP) algorithm implemented in Vehicular ad-hoc network that aid to solve these issues. We introduced different levels of criticality for different levels of emergencies and assigned a certain level of success assurance in terms of target travel time for these criticality. Unlike other studies, instead of implementing EVP algorithm in a

single intersection, we implemented it in an arterial traffic network. We ran exhaustive simulations, and the results indicate that EVP algorithm can significantly reduce the average waiting time of normal traffic but still assures all EVs meet their target response time. In future, we aim to implement EVP algorithm in a larger city map with real-world calibrated traffic data-set. Also, we intend to see the VANET parameters like message delay range, beacon rate and throughput optimization.

Chapter 5

Dynamic Prioritization of Emergency Vehicles For Self-Organizing Traffic using VTL+EV (Manuscript 4)

Organizing Traffic using VTL+EV" implements a decentralized self coordinating traffic system to prioritize emergency vehicle movement through an isolated traffic intersection. The Virtual traffic lights plus for emergency vehicles (VTL+EV) algorithm for intersection control eliminates the loss generated from dead periods in a traffic light cycle time and human-related factors like increased headway time and inconsistent inter-vehicle spacing. We conducted comprehensive experiments and results showed that VTL+EV has the evident advantage of reduced waiting time for regular traffic as well as emergency vehicles. (EVs). The overall throughput of VTL+EV implemented traffic intersection are higher and experiences fewer queue lengths. The manuscript 4 is attached within this section.

Dynamic Prioritization of Emergency Vehicles For Self-Organizing Traffic using VTL+EV *

Subash Humagain and Roopak Sinha, Senior Member, IEEE Department of Information Technology and Software Engineering, Auckland University of Technology Auckland, New Zealand

AUT Tower, Level 7, 2-14 Wakefield Street, Auckland, New Zealand

Tel: +64-9-921 9999 Ext.6376;

corresponding author: subash.humagain@aut.ac.nz

5.1 Introduction

Cooperative vehicle technology has helped advanced applications to evolve in recent times. Advancements in wireless communication techniques that use dedicated short-range communications (DSRC) for vehicular ad-hoc networks (VANET) have already found application in the real-world. Multiple applications like electronic brake lights (allowing drivers or autonomous vehicles react to obstructions by enforcing braking), platooning (following a leader vehicle within inches utilizing real-time exchange of acceleration and speeding information), emergency response services (allowing emergency vehicles to respond on time) and add-on services (advertising for restaurants, petrol stations, etc., to the driver) are already being used in the automotive industry using the IEEE 802.11p communication protocol (Sommer & Dressler, 2014). These applications eventually increase road safety and elevate overall traffic management.

Traffic management and road safety are the principal problems traffic engineers and planners struggle to solve. According to the World Health Organization's Global status report in 2018, 2.34 million road users lost their lives in 2016. It also states that road injury has emerged as the eighth major cause of death for people of all ages and

the number 1 cause of death among children and adults aged between 5-29 years(Tran et al., 2018). Studies show that more than 90% of road accidents are due to human errors(Administration, 2008). Intersections are more prone to accidents because of lack of surveillance (44.1%), the wrong assumption of others' movement (8.4%), movement in obstructed view (7.8%), not following the priority rules (6.8%), distractions within the vehicle (5.7%) and misinterpretation of inter-vehicle gaps (5.5%) (Choi, 2010). Existing technology and measures to reduce incidents at intersections seem insufficient. In case of accidents and emergencies, the traffic control system must aid EVs to expedite movement. In current practice, it is achieved by prioritizing EVs movement, which is termed as Emergency Vehicle Preemption.

Contemporary preemption techniques are inefficient and infrastructure dependent. Different preemption systems like Tramsmax, OPTICOM, GERTRUDE, and FAST are currently being used in multiple cities of UK, USA, Australia, Canada and Japan(Technologies, 2016). These solutions provide empirical evidence of reducing the overall response time of EVs, but all of these solutions can be realized only with the installation of additional devices. The presence of EV is detected from different sensors like localized radio, line of sight or acoustic sensors and GPS installed within vehicles or near the intersections. The traffic is usually controlled from centralized traffic control centres using information sent by these sensors and priority is assigned to EV by altering the traffic signals. Unlike the centralized approach, some systems like EMTRAC (*EMTRAC Signal-Priority System: Transit Signal Priority (TSP) and Emergency Vehicle Preemption (EVP)*, n.d.) use traffic intersections to decide locally on preemption. Decentralized systems can be operated without any centralized backbone network connecting all the traffic intersections. However, this kind of system has two obvious disadvantages:

- Individual intersections have higher operation, maintenance and installation costs.

- No coordination among the intersections results in sub-optimal preemption.

Innovative state-of-art technologies like cooperative vehicular technologies and autonomous driving can mitigate limitations of current preemption techniques elevating efficiency and road safety. Implementations of VANET help in realizing infrastructure-free self coordinating traffic control. An ad-hoc wireless network generated form vehicular movement communicating with each other or infrastructure is termed as VANET. Vehicles and infrastructure use IEEE 802.11p communication protocol to communicate with each other using DSRC devices (F. Li & Wang, 2007). Vehicles can use information beacons with speed, position and direction data transmitted within VANET to coordinate their movement in a traffic intersection. Decisions regarding stopping, acceleration, and turning are adopted by vehicles if they are autonomous or informed to the drivers using inbuilt display devices installed within vehicles. Such self-coordinating traffic control system does not rely on costly infrastructures like traffic lights and any backbone network infrastructures and is termed as virtual Traffic Lights (VTL). Contemporary VTL systems propose a traffic intersection control system that improves the overall efficiency in terms of waiting time and throughput.

The VTL concept has been realized in real-world traffic conditions. The concept of self-organized traffic control termed as VTL was first introduced in (M. Ferreira, Fernandes, Conceição, Viriyasitavat & Tonguz, 2010) where a leader vehicle was elected which acts as a temporary traffic controller and generates traffic light signals using the information available via VANET and sends traffic light information to the drivers or in-vehicle display units. An extension to provide priority to EVs once detected near intersection using VTL was implemented in (Viriyasitavat & Tonguz, 2012). Through simulation results they claimed that the travel time of EVs was reduced significantly and impact over general traffic was marginal. A VTL system, software and apparatus required for coordinating traffic approaching a conflicting zone was designed and

patented. The system developed a dynamic traffic plan and sent to the in-vehicle display unit, which controls the vehicle to follow the plan (M. C. P. Ferreira, Tonguz, Fernandes, DaConceicao & Viriyasitavat, 2015). The system described in (M. C. P. Ferreira et al., 2015) was implemented for the real-world trials in the street of Pittsburgh and was proved that the developed system was capable of coordinating traffic at intersections and reduce the commute time (R. Zhang et al., 2018).

VTL is gaining popularity among researchers and innovators. Shi et al. introduced a concept where vehicle express their will of moving forward, and the leader provides the way according to the score of will that depends on the dynamic traffic condition of the intersection (Shi et al., 2015). Instead of using VANET, the conceptual model of implementing VTL using mobile communication and cloud server was introduced in (Münst et al., 2015). A distributed algorithm that uses both broadcast signals and unicast messages for assigning priority to vehicles that struggle to resolve movement conflicts while approaching an intersection were proposed in (Bazzi, Zanella, Masini & Pasolini, 2014) and (Bazzi, Zanella & Masini, 2016). A VTL framework that can work for both DSRC enabled modern vehicles and normal cyclist and pedestrians was proposed in (Martins et al., 2019). The traffic control information to these users like cyclist and pedestrian was transmitted to their smartphones using Bluetooth devices. In addition, (Olaverri-Monreal, Gomes, Silvéria & Ferreira, 2012) developed a graphical user interface that projects VTL sequences on the windscreen of the vehicle using head-up displays and (Avin, Borokhovich, Haddad & Lotker, 2012) identified the optimal area within the vehicle to place the VTL. Eventually, Sinha et al. identified two issues for the adoption of the VTL in industry, mainly functional safety analysis and migration from non-equipped vehicles to VTL. The solution for the first issue was proposed using a model-driven engineering approach. The solution to the second problem was proposed as implementing VTL+ that uses additional vehicle-to-infrastructure communication from existing infrastructure (Sinha, Roop & Ranjitkar, 2013).

Very less has been studied about prioritizing EVs using VTL. Most of current VTL system elects a leader within an intersection that process the speed and distance information available from multiple vehicles and sends the scheduling instructions to others. In case of accidents, VTLs should aid on prioritizing EVs movement by instructing other vehicles around the accident scene. So far to our knowledge, (Viriyasitavat & Tonguz, 2012) is the only study that has implemented a self-organized traffic control system that manages the priority of EVs. They implemented a conventional static traffic control system with traffic lights to a VTL decentralized to a single leader vehicle stopped at an intersection waiting for the green phase. They utilized the dead periods (unwanted green signal to complete the phase time even if there are no vehicle from that approach) to show a significant increase in traffic throughput. In their approach, an EV approaching an intersection broadcasts priority request message, the leader replies with acknowledgement message, halts regular signal operation and assigns priority by granting green phase to EV. Once EV passes the intersection, it sends a clear message, and regular traffic signal operation is resumed.

In this study, we propose a novel self organizing traffic control system. We consider the vehicles can manoeuvre autonomous driving. Unlike current VTL (replication of traditional traffic lights controlled by a lead vehicle at an intersection), the Virtual traffic light plus for EV (VTL+EV) algorithm we propose completely eliminates the everlasting traffic signalling concept of optimized cycle time and phase duration. Vehicles approaching an intersection calculate the exact speed and duration to pass the intersection and adapt accordingly. Vehicles from any direction should not wait until they get a green phase to pass. This eliminates unnecessary waiting at dead periods and makes the entire system adaptive. Since vehicles are autonomous, they can maintain a minimal distance when they follow other vehicles and the time allowed for a human driver to respond to the change in traffic phase (headway time) can be neglected. The closely following group of vehicles are divided into platoons and each platoon gets a

reserved time to pass the intersection. Since there is no phase change, any platoon from any direction can pass the intersection minimizing all the delays experienced in the traditional traffic control system. The speed maintained by each platoon is dependent on the platoon crossing the intersection. If there is any platoon of vehicles crossing the intersection, the next platoon to cross maintains such a speed so that it does not collide with the platoon crossing the intersection. This ascertains that the intersection is thoroughly utilized and does not allow unnecessary waiting for vehicles when it is free due to inappropriate phase cycles.

VTL+EV dynamically prioritizes EVs approaching an intersection. Whenever an EV approaches the intersection, it transmits information beacon that contains its current position, speed and ID (code to identify it as EV). This halts the current operation of the traffic controller and calculates reserve time for EV to cross the intersection from the current location and allows traffic from this direction only. Once EV crosses the intersection, it resumes its regular operation. In this approach of prioritizing EVs (self-organized traffic control) vehicles from any direction Zip while crossing the intersection. This minimizes all the delays that exist in contemporary systems. The effect of preemption on other vehicles is negligible.

The proposed system shows promising results. We considered well known four-legged traffic intersection to perform multiple experiments using microscopic traffic simulation engine called SUMO (Krajzewicz et al., 2002). We compared average waiting time of EVs and normal vehicles in VTL+EV system with self-organized Traffic control system called as VTL-PIC to manage the priority of EVs (Viriyasitavat & Tonguz, 2012) and enhanced traffic light scheduling algorithm (ETLSA) that assigns adaptive green phase to traffic according to the number of traffic flow (Younes & Boukerche, 2018). We also compared the overall throughput and queue-length of intersection implementing each of above-mentioned EV's preemption system. VTL+EV algorithm outperformed both VTL-PIC and ETLSA preemption systems. The average

waiting time was considerably less, throughput was significantly high, and average queue-length was minimal.

5.2 System Model

A four-legged intersection is the most commonly used model in traffic engineering for studying the performance as it can be scaled to simpler or more complex intersections. Whenever any vehicle (both autonomous and manual driven) approaches an intersection, it examines the traffic phases. If the traffic phase is green, it crosses the intersection else manoeuvres deceleration or stops altogether. In case of manual vehicles, the driver uses senses to identify the traffic phases whereas autonomous vehicles combine signals from multiple sensors like lidar, sonar, radar, odometer, GPS and inertial measurement units to identify appropriate driving actions(Taeihagh & Lim, 2019). In our approach, we consider that every intersection comprises of an Intersection Traffic Controller (ITC) that communicates with incoming platoons of autonomous vehicles sending instructions regarding stopping, acceleration and turning. The same system can be deployed for manual vehicles enabled with DSRC devices. However, it will result in less efficiency because of involved human factors like increased headway time and inconsistent platoon size.

In this study, we assume that all the vehicles are equipped with DSRC communication devices to communicate position, speed, and vehicle type data collected from inbuilt vehicle sensors like GPS. Every ITC has a defined range of area. A roadside unit (RSU) collects traffic information and supplies it to the ITC. The communication between vehicles and RSU is carried out in the 5.9GHz spectrum dedicated for DSRC using IEEE 802.11p. These data are used by platooning algorithm and traffic control algorithm to generate traffic instructions, which are transmitted back to vehicles to adhere to. Fig. 5.1 reflects the basic system operation. Whenever an EV shows up

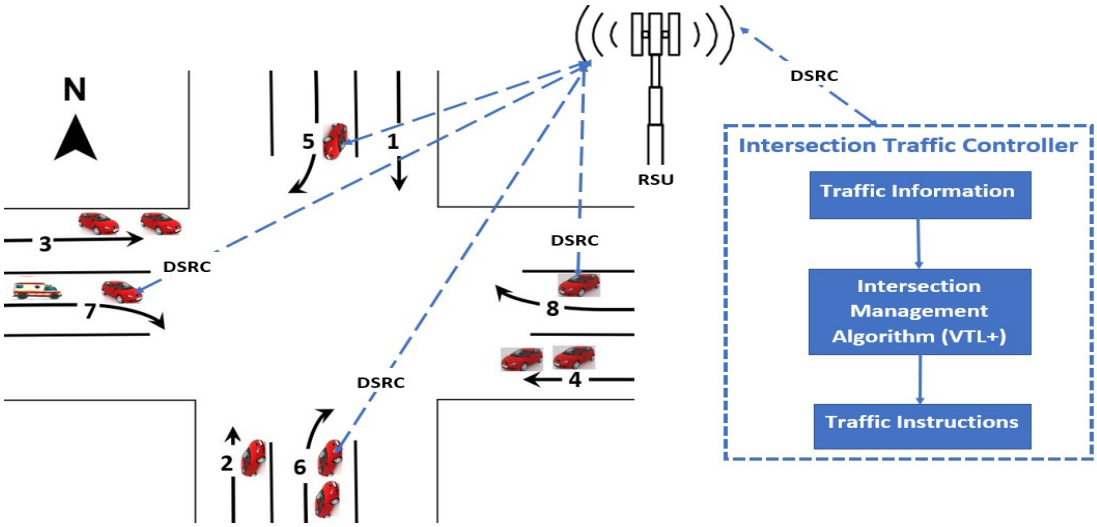

Figure 5.1: Intersection Traffic Control System

within the range of the ITC, it notifies the RSU. The RSU distinguishes EVs from other vehicles using its vehicle type parameter which is set to Emergency. Once detected, the traffic control algorithm calculates the reserved time required for this particular EV to cross the intersection. Updated speed, acceleration and stopping instructions are sent to the EV and all other vehicles. Once the EV crosses the intersection, it sends an acknowledgement message to the ITC to resume normal operation.

5.2.1 Platooning

Platooning increases road capacity by eliminating constant stop and go of individual vehicles. Modern traffic management systems have leveraged a lot from inter-vehicle communication technologies. The advent of autonomous vehicles is one of the promising examples to cite. Inter vehicle communication has become very reliable; that is why autonomous vehicles can follow each other within a distance of inches forming platoons. Autonomous vehicles abolish time loss due to human factors like slow reaction time, diverse driving behaviour and possibilities of attention lapses. Platooning, in addition, increases traffic efficiency (Fernandes & Nunes, 2012b). To leverage the

Algorithm 4 Platooning Algorithm

Input: startingVehicles, active, color, currentSpeed, disbandReason, eligibleForMerging, lane, lanePosition, controlledLanes, and targetSpeed
Output: Platoons

```
for each ControlledLanes l do
    addVehicle (vehicle);
    mergePlatoon(P_i):
        if (l.P_i.VehPathsConverge(l.P_i.getAllVehicles())
        && l.P_i.getLane() == l.P_i.getLane() ) then
            l.P_i.disband();
            l.P_i.disbandReason = Merged;
            for vehicle in l.P_i.getAllVehicles() do
                l.P_i.addVehicle(vehicle);
        else
            l.P_i.eligibleForMerging = False
    setTargetSpeed(speed);
    setGap(gap);
    updateSpeed(speed, inclLeadingVeh=True);
```

advantages mentioned above, instead of implementing self coordinating traffic for individual vehicles, we have divided vehicles approaching an intersection into platoons.

The algorithm to construct platoons is depicted in Algorithm. 4. The area within the range of ITC is divided into different controlled lanes. Controlled lanes are gradually added with vehicles one at each time step. The first vehicle, so added, converts into a single sized platoon. If two platoons in a controlled lane are within a certain distance and share the same route, they are eligible for merging. In that case, two existing platoons are disbanded, and a new platoon of size two is created. This condition is checked until the size of platoon increases to a maximum allowable size. Literature suggests that platoon length of a maximum of 35 meters does not diminish overall performance (Fernandes & Nunes, 2012b). Therefore, considering a single-vehicle size of 3 meters, we have defined platoon size to 12 vehicles. Every platoon has a leader. Other vehicles within the platoon follow leader's behaviour within the set gap in terms of speed and acceleration. Once a platoon gets outside the ITC zone, vehicles continue being within the same platoon if they still satisfy the above-mentioned condition else they disband themselves.

5.2.2 Proposed VTL+EV Algorithm

ITC sets instructions regarding speed required for each platoon to adhere while crossing an intersection using the VTL+EV algorithm. An area is defined for the intersection controller. We add platoons from all lanes and calculate the reserved time required for the platoon to cross the intersection. If this platoon is the first one to post the reservation, we calculate distance from the stop line to the platoon's current position and add it with the platoon's length which is done as shown in Fig. 5.2.

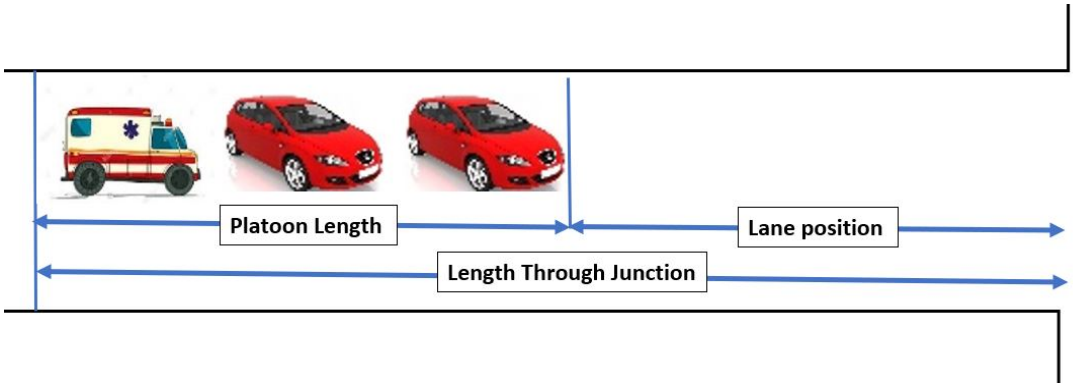

Figure 5.2: Calculation of reserved time

If this platoon contains EV we allow this platoon to pass else we process platoons in ascending order of its distance from the stop line. The VTL+EV algorithm is illustrated in Algorithm 5. Every platoon within the influence of VTL+EV algorithm is allocated with the time calculated from reserved time to pass the intersection.

5.3 Implementation

We compare the performance of VTL+EV algorithm with self-organized Traffic control system called as VTL-PIC and enhanced traffic light scheduling algorithm (ETLSA) using well known microscopic traffic simulator SUMO. The key performance parameter we compared was the average waiting time measured in seconds for both EVs and

Algorithm 5 VTL+EV Algorithm

```
1:  for this Controller c do
2:      addPlatoon (P_i);
3:      calculatePlatoonReservedTime (P_i);
4:      removeUncontrolledPlatoons ( );
5:      addAllControlledPlatoons ( );
6:      setZipOrderForController (P_i ):
7:          if ((c.P_i.getVehicletype()) == Emergency)        then process P_i;
8:          else distSort(elem):
9:              return elem.getLanePositionFromFront()
10:     createZipPlatoons ( );
11:     getlaneposition (P_i);
12:     setNewSpeed (P_i, reservedtime);
13:     removePlatoon (P_i);
14:     update ( );
```

general traffic. The efficiency of the intersection controller was measured in terms of throughput expressed in passenger car unit per lane per hour (pcu/ln/sec).

SUMO is an open-source continuous microscopic traffic simulator. SUMO gathers information on aspects like networks, routes, trips, and additional sensor devices from its components in advance and runs a simulation until completion. We cannot alter any parameters when the simulation is on. We used an additional tool implemented using Python called Traffic Control Interface (TRACI). TRACI gives access to the live road traffic simulation, allows to capture the required values of simulation and changes their behaviour while the simulation is live. TRACI and SUMO communicate during the simulation using virtual ports (TCP sockets) following the TCP/IP protocol. SUMO behaves as a server that provides services to the client TRACI whenever it sends any service request. As in client-server, architecture SUMO can handle multiple TRACI request at the same time. TRACI also helps to connect different network simulators like NS-3 and OMNET++ with SUMO.

Inter-vehicle communication is essential for the implementation of VTL+EV. A wireless network VANET created because of vehicle communicating with other vehicles and infrastructure is the backbone of the VTL+EV algorithm. This can be realized using an open-source framework Veins that connects OMNET++ and SUMO using TRACI.

Using TRACI inbuilt functions, we can directly access traffic network information like lanes, vehicle lane position, speed, acceleration, vehicle ID and vehicle type. We can use this information to execute both platooning and VTL+EV algorithms. In this study, we study if VTL+EV enhances the identified traffic parameters while assuming robust and dependable VANET communications. Since we are not interested in the analysis of wireless communication performance, we do not implement traffic simulation through Veins (an open source framework based on SUMO and OMNET++ for implementation of different models of inter-vehicle communication) and instead use TRACI to access this information. Fig. 5.3 visualizes the simulation of VTL+EV algorithm using SUMO and TRACI.

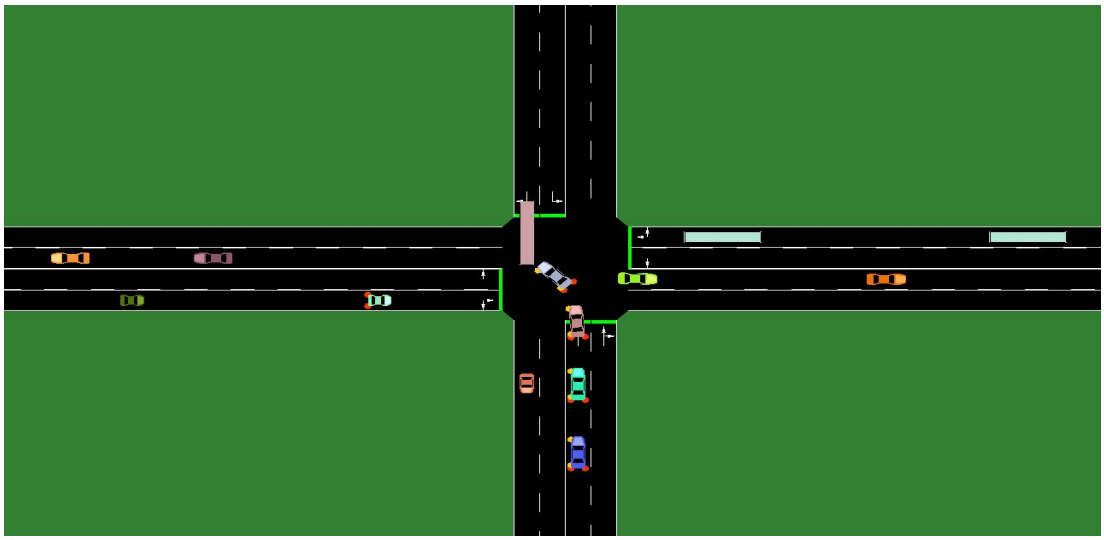

Figure 5.3: Implementation of VTL+EV algorithm in SUMO

We choose traffic parameters during our simulations to replicate real-world traffic scenarios. During the entire course of simulation we first kept on increasing the traffic volume from low (400 pcu/ln/hr), to medium (800 pcu/ln/hr), and then high (1600 pcu/ln/hr) and subsequently decreased it back to medium and then low. Vehicles are generated randomly using inbuilt python command `randomTrips.py` and the result follows Binomial distribution approximating to Poisson distribution for small

probabilities as expressed in Fig. 5.4. We have considered 1% of total vehicles generated as EVs.

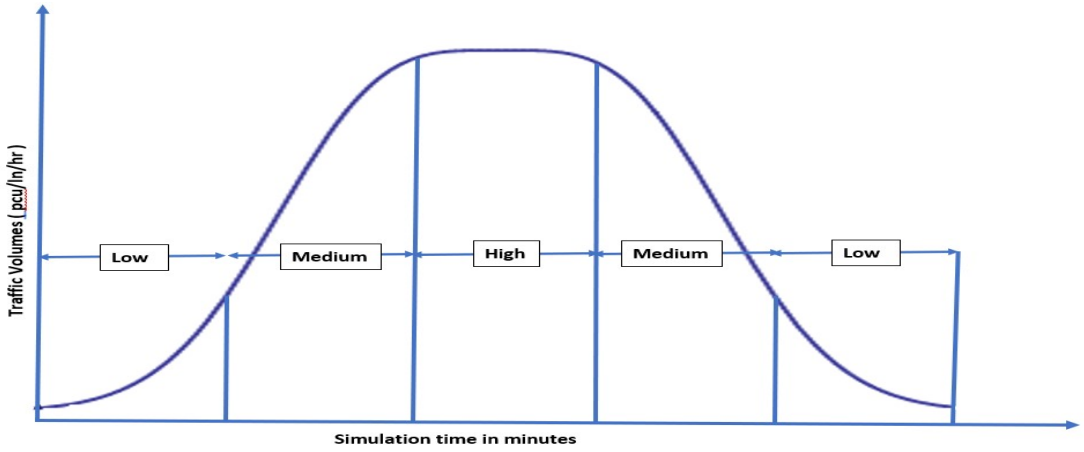

Figure 5.4: Traffic volume in Poisson distribution

5.4 Performance Evaluation

VTL being a substitute for physical traffic lights has the advantage of being more economical, but to compare its performance with a simplistic system would be implausible as there are already more advanced adaptive traffic control systems. Therefore to justify that the results produced by VTL+EV are promising we compare its performance with the following state-of-art technologies implemented earlier:

- VTL-PIC: Basic VTL implementation with the use of DSRC technology to detect the presence and absence of EVs at an intersection and assign priority to EVs(Viriyasitavat & Tonguz, 2012).
- ETLSA: An adaptive traffic scheduling algorithm that dynamically adjusts green phases of traffic signals based on real-time traffic distribution and allows EVs to pass smoothly by coordinating traffic signals using VANET(Younes & Boukerche, 2018).

We conduct multiple experiments with different traffic penetration rates to evaluate below-listed performance parameters for a four-legged traffic intersection:

- *Average Waiting Time* of all Non-EVs.
- *Average Waiting Time* of all EVs.
- *Average Queue length* of intersection under study.
- *Overall Throughput* of intersection under study.

Fig. 5.5 dissipates the overall waiting time of non-EVs for VTL-PIC, ETLSA, and VTL+EV algorithms. Since VTL-PIC is a virtual representation of a traditional preemption system, such a system incurs a very high waiting time for non-EVs. ETLSA, being the most recent adaptive traffic control system that prioritizes EVs, claimed to outperform existing VANET enabled traffic control system. The increased waiting time resulting from static preemption has been considerably reduced in ETLSA. However, VTL+EV outperforms both of these algorithms in terms of waiting time as VTL+EV is more adaptive and eliminates the loss that arises from multiple traffic phase change and human-related factors. For low and medium traffic volumes, waiting time of VTL+EV is almost zero as autonomous vehicles prefer maintaining the required speed to cross the intersection than to stop. We also compared the average waiting time of EVs in all of these systems. VTL-PIC still used the traditional approach of continuous cycle time. The average waiting time for EVs in this system is highest because the controller must complete the current traffic phase before changing it to green for providing EVs uninterrupted passage. VTL+EV still outperforms both VTL-PIC and ETLSA in terms of average waiting time for EVs, as illustrated in Fig. 5.6.

Queue lengths and throughput represent elementary performance parameter for quantifying the capacity of any traffic controller. Queue length is measured as the difference in the number of vehicles approaching and leaving an intersection. We

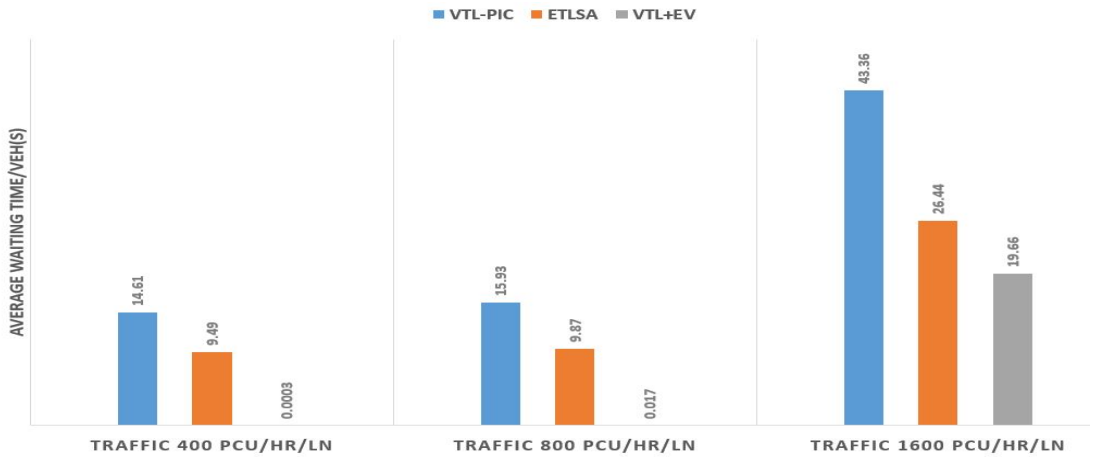

Figure 5.5: Average waiting time of normal traffic

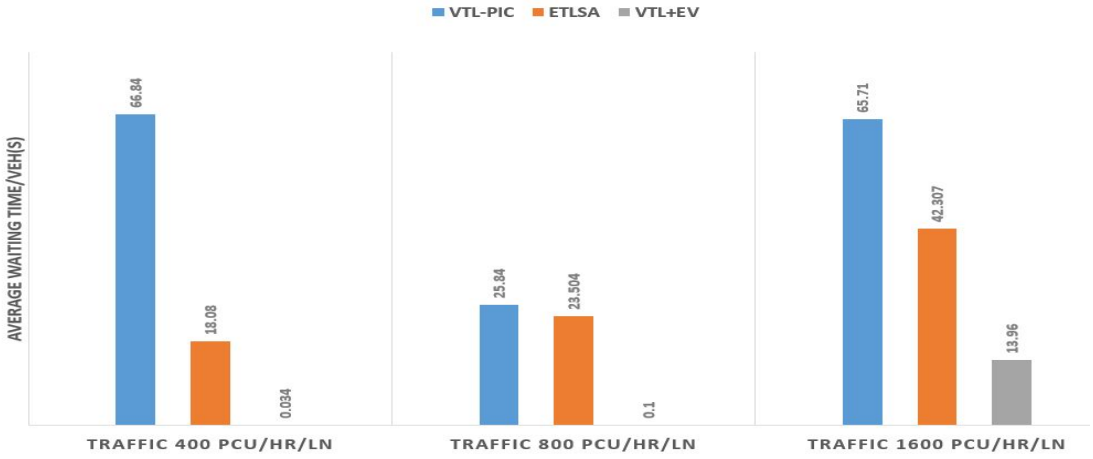

Figure 5.6: Average waiting time of EVs

compared the average queue length of VTL+EV with VTL-PIC and ETLSA. The simulation results show that the queue length for VTL+EV under all traffic penetration rates is considerably less than VTL-PIC and ETLSA as displayed in Fig. 5.7. Similarly, throughput computes the traffic volume that an intersection can process within a specific time. Our experimental results show that VTL+EV has the highest throughput compared to the remaining two algorithms under comparison, as pictured in Fig. 5.8.

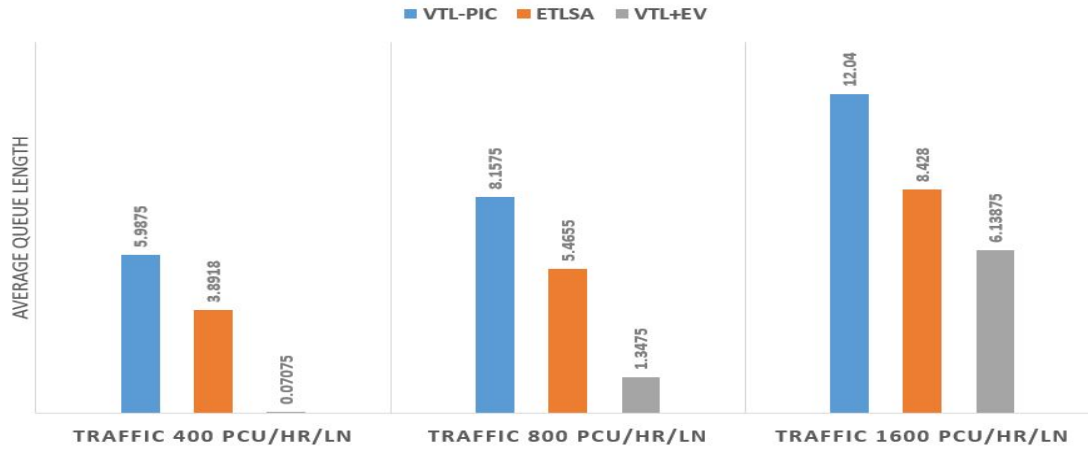

Figure 5.7: Average queue length

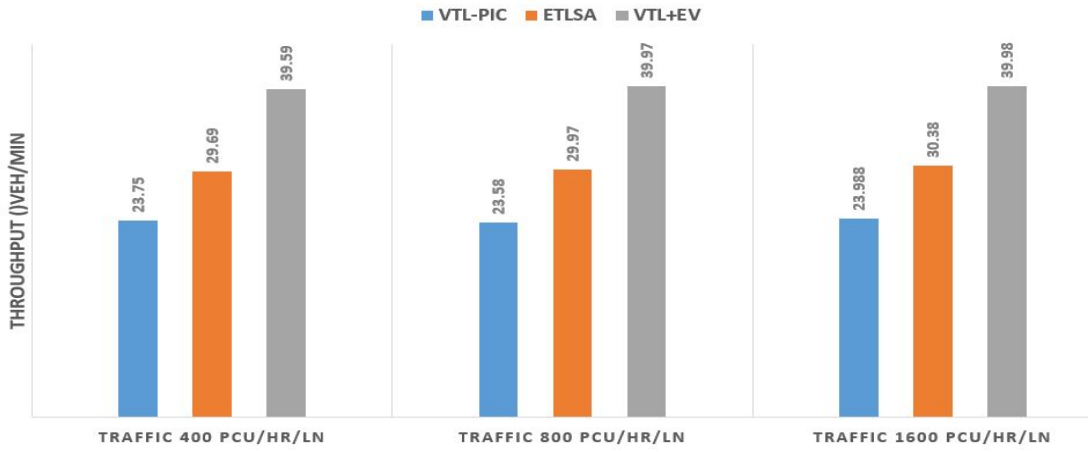

Figure 5.8: Average throughput

5.5 Conclusion and Future Work

We implemented a decentralized self coordinating traffic system to prioritize emergency vehicle movement through an isolated traffic intersection. The proposed Virtual traffic lights plus for emergency vehicles (VTL+EV) algorithm for intersection control eliminates the loss generated from dead periods in a traffic light cycle time and human-related factors like increased headway time and inconsistent inter-vehicle spacing. We conducted comprehensive experiments and results showed that VTL+EV has the evident advantage of reduced waiting time for regular traffic as well as emergency vehicles

(EVs). The overall throughput of VTL+EV implemented traffic intersection are higher and experiences fewer queue lengths. In future, we aim to realize VTL+EV algorithm by implementing it in larger calibrated city maps.

Chapter 6

Travel Time Estimation with Decision-Tree-Based Ensemble Networks and Enhanced Feature Extraction (Manuscript 5)

This paper "*"Travel Time Estimation with Decision-Tree-Based Ensemble Networks and Enhanced Feature Extraction"*" a novel solution for Origin-to-Destination (O2D) travel time estimation utilizing, enhanced feature engineering and ranked sub-features. Enabling decision-tree-based ensemble networks to learn from multiple pre-engineered features enhances the extraction of non-redundant sub-features. These features mainly relate to geographical topology, navigational bearing (directions), spatial patterns within the traversed path/trajectory, travel distance and dynamic constraints such as speed-limits and traffic conditions. All of the aforementioned sub-features are extracted from only Origin-to-Destination (O2D) GPS co-ordinate information without the need for intermediate GPS traces, making our network models pure-O2D requiring significantly less computational resources to train.

Deep neural network-based solutions using raw GPS trajectories perform sub-optimally with an additional drawback of perceived incomprehension, often referred to as a black-box. Decision-tree-based ensemble networks perform adequately but, suffer from gradient clipping, necessitating the need for extreme boosting. Our proposed solution employs gradient boosted decision-tree-based ensemble networks and 02D GPS co-ordinates, hence named, Gradient Boosted, Origin-to-Destination (GB02D). We rank and visualize individual features' contributions in the entire learning process, eliminating the perpetual issue of interpretation and visualization. Furthermore, our proposed GB02D outperforms existing state-of-the-art deep neural network baseline models. The manuscript 5 is attached within this section.

Travel Time Estimation with Decision-Tree-Based Ensemble Networks and Enhanced Feature Extraction

Subash Humagain, Nidhi Gowdra and Roopak Sinha, Senior Member, IEEE

Department of Information Technology and Software Engineering,

Auckland University of Technology Auckland, New Zealand

AUT Tower, Level 7, 2-14 Wakefield Street, Auckland, New Zealand

Tel: +64-9-921 9999 Ext.6376;

corresponding author: subash.humagain@aut.ac.nz

6.1 Introduction

Precise estimation of vehicle travel time is crucial in route planning and navigation (Y. Li et al., 2015), congestion detecting (Y.-y. Chen et al., 2016), ride sharing (Asghari et al., 2016) and logistics (N. J. Yuan et al., 2012). Accurate estimation of travel time for user queried paths is critical in providing superior services to cater for such services. Compared to two traditional techniques (regression & statistical approaches and historical averages & smoothing), machine learning approaches have shown outstanding performances (Hunter, Herring, Abbeel & Bayen, 2009). Undoubtedly, the problem has been extensively explored in the last decade (Y. Li et al., 2018; D. Wang, Zhang, Cao, Li & Zheng, 2018; Z. Wang, Fu & Ye, 2018). However, the accuracy of travel time estimation using machine learning approaches still poses a challenge. Whenever users query for such services, they only provide the origin and destination of the trip; and travel time estimation can be carried out either by *point-to-point* (P2P) or *origin-to-destination* (O2D) approaches.

P2P approaches (Hunter et al., 2009; Pan, Demiryurek & Shahabi, 2012; Rahmani, Jenelius & Koutsopoulos, 2013; Y. Wang, Zheng & Xue, 2014; Tang et al., 2016) divide

the entire route into multiple road segments, estimate the travel time of each segment and finally add estimated travel time of each segment to provide an overall estimation. Although *P2P* approaches can provide accurate travel time estimation of an individual road segment, learning to estimate travel time in this approach incurs a very high computational cost, which is highly undesirable in online prediction services (Y. Li et al., 2018). Also, collecting data of intermediate points within a trajectory demands very high technological, time and capital investment. This kind of approach cannot model multiple traffic factors that directly affect travel time like traffic lights, intersections, and driver preferences. Besides, if the overall route consists of many road segments, the local error of each road segment adds together, resulting in unrealistic travel time predictions (D. Wang et al., 2018). Researchers and industry practitioners have tried to solve the underlying issues of *P2P* approach by implementing *O2D* travel time estimation (Jindal, Chen, Nokleby & Ye, 2017; D. Wang et al., 2018; Y. Li et al., 2018; H. Wang, Tang, Kuo, Kifer & Li, 2019; Xu, Zhang, Chao & Xing, 2019) using capabilities of deep learning. This approach intends to provide accurate travel time predictions without any details of the actual route. Deep learning approaches have the potential of learning underlying features from data which are not apparent and can solve complex dynamic problems (Yao et al., 2018). This feature of deep learning enables *O2D* approach to capture complex spatio-temporal traffic conditions within an entire trip indirectly like traffic lights, intersections, driver preferences and direction change. The *O2D* approach using deep learning still have the following limitations:

- For longer paths, the statistical confidence of predicting the travel time is very less as there are very few trajectories passing the entire route. In most of the cases, no trajectories are passing the entire route (Y. Li et al., 2018; D. Wang et al., 2018).

- Minimal input features like origin, destination and time of departure carry minimal

information, making prediction difficult (Y. Li et al., 2018; H. Yuan, Li, Bao & Feng, 2020).

- It is tough to learn features like similarity of the road network and total distance covered just from available latitudes and longitudes (Y. Li et al., 2018; H. Yuan et al., 2020).
- There is very less control over what NNs learn and identify which variables are dominant in the learning process (T. Chen & Guestrin, 2016).
- Though most studies claim to use only *O2D* pair; detail exploration reveals that they use intermediate GPS points for extracting features in their estimation which is still computationally expensive.

A detailed literature survey provided in Section 3.2 suggests almost all existing studies directly make use of limited input features for developing learning models neglecting that multiple features can be learned using the map data from transportation road network which are valuable for accurate travel time prediction in *O2D* approach. Thus, to overcome the aforementioned limitations of input features in *O2D* travel time estimation, recent state-of-art technologies have taken advantage of immense spatio-temporal information embedded within map data underlying a road network (Y. Li et al., 2018; H. Yuan et al., 2020). Deep learning's beauty lies in its characteristic of learning features from huge data where manual feature engineering is impossible. It has achieved exceptional success over unstructured data like natural languages, videos and images, but its performance over structured data is comparatively less effective (Q. Zhang, Yang, Chen & Li, 2018). The main reason behind this is sparse feature selection characteristic of deep learning as these approaches assume continuous features, i.e. transition between possible values in a data-set is not abrupt, which rarely holds in case of structured data. Additionally, all of these approaches employ feature engineering before employing deep

learning (D. Wang et al., 2018; Y. Li et al., 2018; H. Wang et al., 2019). They still use intermediate GPS trajectory traces to find the overall route to employ map embedding, making it a pseudo *O2D* approach.

Recently, ensemble-based algorithms achieved overwhelming celebrity status from machine learning practitioners in solving classification and prediction problems. Ensemble-based algorithms' application in diverse fields with superior accuracy has added more values (Zhou, 2012). Winning of $1 Million rewards from Netflix competition using an ensemble-based algorithm to predict user rating for movies with highest accuracy (Koren, 2009) lured machine learning experts adopt this approach to deep learning. Among ensemble-based learning method in practice, gradient tree boosting (also known as gradient boosted regression tree (GBRT) or gradient boosting machine (GBM)) is the most popular technique implemented in many applications (Friedman, 2001).

This paper proposes a novel solution for *O2D* travel time estimation using gradient boosted origin-to-destination (GBO2D) regression tree. GBRT based scalable machine learning system for tree boosting called XGBoost(T. Chen & Guestrin, 2016) has achieved unrealistic success in recent days. According to the Kaggle blog 2015, a machine learning competition website, 17 out of 29 winning solutions used XGBoost to train the model (*Kaggle*, 2015). Similarly, XGBoost was used by every top 10 winning teams in KDD Cup 2016 (*KDD Cup*, 2015). We also employ ensemble-based gradient boost regression tree method in *O2D* travel time estimation using sole origin and destination GPS trace making it pure *O2D* approach. We extract multiple valuable features from raw GPS data before we train the model as the success of unsupervised machine learning problems can be improved exceptionally when re-framed as a supervised problem (LeCun, 2018). We utilize Open Source Routing Machine (OSRM) (Haklay & Weber, 2008) to extract features like route, total distance travelled and total time to cover this distance using the open street map for the underlying road network. The proposed solution learns better from extracted meaningful features compared to raw

GPS trajectory data. We employ Principle component analysis (PCA) (Abdi & Williams, 2010), an unsupervised learning model, to learn feature representation from large raw GPS trace data and reduce its dimensions. We also extract spatial features dividing the overall range of location into multiple clusters, identifying locations of importance like airports and classifying trips originated or ended in such locations. We also extracted temporal features like a week, days of the week and hourly distribution of trips. Unpredictable daily events like accidents were also acknowledged to extract additional features. Finally, after extracting multiple spatial and temporal features, we employ extreme gradient boosting (XGboost) (T. Chen & Guestrin, 2016) machine learning approach to estimate travel time which enhances the existing learning performances. The major contributions of this paper can be summarised as follows:

- We propose a novel solution for pure *O2D* travel time estimation.
- We utilize the information from the road network to extract additional features to overcome the limitation of information present in raw GPS trajectory data.
- We extract spatial features dividing the overall range of location into multiple clusters, identifying locations of importance like airports and classifying trips originated or ended in such locations which help the model learn the pattern of trips improving prediction.
- We extract multiple temporal features like a week, days of the week and hourly frequency of trips to ease model to learn the temporal pattern.
- We utilize unpredictable daily events like accidents in a different location to learn the difference in speed profile around events' location.
- We perform multiple experiments on two available real-world GPS trajectory data sets collected by taxis in New York and DIDI ride-sharing app in Chengdu

China. The experimental results show that our approach significantly improves the existing state-of-the-art approach of *O2D* travel time estimation using deep learning.

- We also rank and visualize the contribution of extracted features in learning to estimate *O2D* travel time which is not possible in deep NNs.

The remaining of the paper is structured as follows. Section 6.2 discusses related works in travel time estimation followed by a detailed model of gradient boost model. Section 6.4 explains data used in this study and all extracted features from available data with its visualizations. Section 6.5 analyzes the result of our model and compares with other state-of-art methods. Lastly, the conclusion is outlined in 6.6.

6.2 Related Work

Travel time estimation problems can be broadly classified into two categories: *P2P* and *O2D* approaches. *P2P* approaches require information of all the routes passing the queried trajectory and their associated sub-trajectories. *O2D* approaches can estimate travel time without the knowledge of actual route traversed. A *P2P* approach divides the entire route into multiple road segments, estimates the travel time of each segment separately and finally adds estimated travel time of each segment to provide an overall estimation of the entire route. The travel time of each segment is calculated using loop detectors (Tang et al., 2016) or from floating car data collected from GPS sensors (Hunter et al., 2009; Pan et al., 2012; Rahmani et al., 2013; Y. Wang et al., 2014). This kind of approach cannot model multiple traffic factors that directly affect travel time like traffic lights, intersections, and right or left turn. These factors need to be considered separately. In cases where multiple trajectories share the same sub-path, the estimated travel time of an individual sub-path can be calculated based on the effect of all the trajectories that

pass through this particular sub-path. This limitation of *P2P* approaches was improved by concatenating the estimation of individual sub-path with other sub-paths that sum-up to form entire trajectory which avoids the need of considering above mentioned complex traffic factors like traffic lights, intersections, and right or left turn (Rahmani et al., 2013). However, the number of trajectories passing the entire path becomes sparse for the longer routes, which eventually decreases the statistical confidence of the accurate prediction. Further, Wang et al. (Y. Wang et al., 2014), and Song et al. (Song, Sun, Zheng & Zheng, 2014) used historical trajectories data to mine frequent trajectories for a sub-path and implemented the optimal way of sub-path concatenations for sparse trajectories. Although *P2P* approaches can provide accurate travel time estimation of an individual road segment, they incur a very high computation cost, which is highly undesirable in online prediction services (Y. Li et al., 2018). Also, in real practice, the overall route consists of a large number of road segments, the local error of each road segment add together, resulting in unrealistic travel time predictions (D. Wang et al., 2018).

Recently, researchers and industry practitioners started working on *O2D* approaches for travel time estimation to solve the underlying limitations of *P2P* approach. This approach can estimate the travel time without any information of actual trajectory a vehicle has traversed. With GPS sensors being readily available viz user's mobile devices or vehicles inbuilt GPS devices, large trip data were publicly available for researchers. This availability of data lured researchers working on *O2D* approaches to incline towards learning different patterns from the available data and predicting travel time using Neural Networks (NNs) (H. Zhang, Wu, Sun & Zheng, 2018). Since deep NNs can learn features on their own (unless implemented supervised learning), deliver high-quality results, and eliminate data labelling requirements, travel time estimation performed using deep NNs has produced most accurate results (D. Wang et al., 2018; Y. Li et al., 2018; Xu et al., 2019).

Siripanpornchana et al. proposed Deep Belief Network (DBM) that automatically learn generic traffic features in an unsupervised way using Restricted Boltzmann Machines and predicted travel time (Siripanpornchana, Panichpapiboon & Chaovalit, 2016). Any GPS trajectory data contains both spatial and temporal information. More accurate prediction of travel time can be made if the spatial-temporal nature of trajectory data is acknowledged. Temporal property of travel data was incorporated by using Recurrent Neural Networks (RNN) implemented through Long Short-Term Memory (LSTM) networks for the first time in travel time predictions (Duan, Yisheng & Wang, 2016). DEEPTRAVEL, ST-NN and STDR made use of temporal labels of trajectory data and predicted travel time using inherit feature extraction property of deep neural networks (H. Zhang et al., 2018; Jindal et al., 2017; Xu et al., 2019). Based on usable feature extracted from a large set of trajectory data, authors in (Z. Wang et al., 2018) formulated travel time estimation as a spatial-temporal regression problem and deployed a wide deep recurrent network to predict the travel time accurately. Wang et al. introduced DeepTTE to make more accurate travel time predictions. Apart from using spatial-temporal attributes of GPS trajectory data, they also embedded additional factors like starting time, day of the week, weather conditions and corresponding driver in the learning process. The geo-based convolutional layer was implemented to transform raw GPS samples into feature maps, and LSTM was implemented to learn temporal dependencies obtained from the feature maps and embedding of external factors (D. Wang et al., 2018). Instead of learning from single source trajectory data, authors in (Lin, Wang, Xiao, Li & Bhowmick, 2019), incorporated hybrid trajectories data obtained from different types of vehicles to generalize and improve the travel time estimation process. Though the implementation of deep learning in *O2D* approach improved travel time estimation considerably, just using raw GPS trajectories in the prediction process still has limitations which are listed below:

- Minimal input features like origin, destination and time of departure carry very less information, making feature extraction and prediction difficult.
- It is tough to learn features like similarity of the road network and total distance covered just from available latitudes and longitudes.
- For longer paths, the statistical confidence of predicting the travel time is very less as there are very few trajectories passing the entire route. In most of the cases, no trajectories are passing the entire route. Prediction from sparse trajectory data is a real challenging problem.

As a recent advancement in *O2D* travel time estimation, some researchers used additional spatio-temporal information embedded within map data underlying road networks (Y. Li et al., 2018; Das et al., 2019; H. Yuan et al., 2020). This approach outperformed existing approaches with limited input features for developing learning models which neglected the possibility of learning multiple features from the map data for road networks. Embedding map information proved valuable for accurate travel time prediction in *O2D* approach. Li et al. proposed a multi-task representation learning (MURAT) that utilizes underlying road networks structures in learning process (Y. Li et al., 2018). DeepWalk (Perozzi, Al-Rfou & Skiena, 2014) was implemented to improve the learning process to extract enhanced representational features using an unsupervised graph embedding approach based on the underlying map and raw trip data. A multi-task learning framework was proposed by (Y. Li et al., 2018) incorporated spatial and temporal domain data. Representations of these corresponding domains were implemented using individual grids.

Deep learning's beauty lies in its characteristic of learning features by itself from very large data where manual feature engineering is impossible. It has achieved exceptional success over unstructured data like natural languages, videos and images, but its performance over structured data is comparatively less effective. The main reason

behind this is sparse feature selection characteristic of deep learning as these approaches assume continuous features, i.e. transition between possible values in a data-set is not abrupt, which rarely holds in case of structured data. Rather, in case of these data:

- The majority of correlations among the labels is contributed from a small number of features.
- The categorical features can overflow according to the data.

Fortunately, current ensemble-based algorithms can address the above-mentioned issues.

Recently, ensemble-based algorithms achieved overwhelming celebrity status among machine learning practitioners for solving classification and prediction problems. Their applications in diverse fields with superior accuracy has added more values (Zhou, 2012). Among ensemble-based learning methods in practice, gradient tree boosting (also known as gradient boosted regression tree (GBRT) or gradient boosting machine (GBM)) is the most popular technique implemented in many applications (Friedman, 2001). Unlike, other machine learning approaches that tend to find a single best-fitting model, tree-based gradient boosting algorithms strategically combine small tree model to optimize the prediction. Shallow trees within themselves are often considered as weak predictive models. But, when boosted to form a strong committee and tuned appropriately, they can outperform other learning algorithms. Boosting addresses the bias-variance trade-off. Initially, it starts with a weak model (i.e. a tree with fewer splits) and sequentially boosts a weak tree's performance by generating new trees that learn from an earlier tree's error. Sharing insight from both machine learning and statistics this model can achieve very strong and accurate prediction and also identify interactions of relevant variables (Elith, Leathwick & Hastie, 2008). Properties like handling different type of feature variables, the requirement of little data prepossessing, and ability to fit complex nonlinear problem make GBRT a superior candidate to be used in *O2D* travel time estimation than deep neural networks (Y. Zhang & Haghani,

2015).

Though GBRT has obvious features to be used in *O2D* travel time estimation, there are minimal studies that have used the ensemble-based gradient boost regression tree in travel time estimation. Zhang and Haghani implemented GBRT to predict travel time on a freeway using historical travel time data. The study was limited to a small section of the city with limited data-set. The same model was used to visualize the importance of different variables in predicting travel time in freeways (Cheng, Li & Chen, 2018). Similarly, GBRT was used to predict the incident clearance time and rank the influential factors that contribute to incident clearance time (X. Ma, Ding, Luan, Wang & Wang, 2017). Nevertheless, the possibility of implementing GBRT in *O2D* travel time estimation for the entire city with a large number of trajectory data (millions) was never explored.

In this paper, we introduce GBO2D, a novel approach of using ensemble-based GBRT method in *O2D* travel time estimation that learns from multiple pre-engineered features related to topology, directions, the shape of the trajectory, path, speed limits, map distance traversed, real-time traffic conditions and compare it with existing deep neural network-based learning process using raw GPS trajectories. We propose to employ a better and simplified technique to predict travel time, generating superior results than employing deep NNs. We do not consider intermediate GPS trace making our model pure O2D travel time model and less computationally expensive. We rank and visualize individual features' contributions in the entire learning process, which eliminates the perpetual issue of interpretation and visualization in deep learning and perceiving it as a black box.

6.3 Methodology

6.3.1 Gradient Boosting

Gradient Boosting Machines (GBMs) are currently highly acknowledged machine learning algorithms that work in the principle of building ensembles. GBMs construct ensembles of shallow trees in a sequence where new trees are constructed by learning and improving from earlier trees. Shallow trees within themselves are often considered as weak predictive models. But, when boosted to form a strong committee and tuned appropriately, they can outperform other learning algorithms. Boosting addresses the bias-variance trade-off. Initially, boosting starts with a weak model (i.e. a tree with fewer splits) and sequentially boosts weak trees' performance by generating new trees that learn from the error of earlier tree as shown in Figure. 6.1. Each new model generated in sequence slightly improves the earlier model's performance by focusing on rows of training data where the previous model had the largest errors.

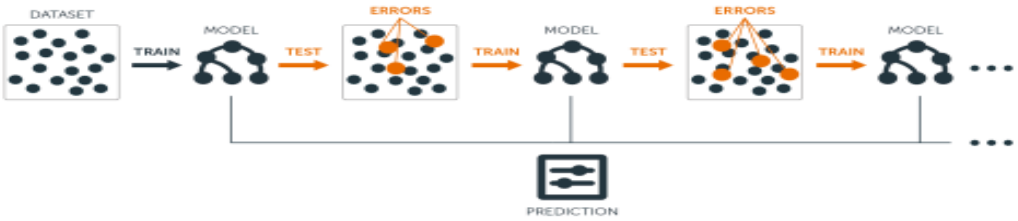

Figure 6.1: Sequential boosting approach

6.3.2 Learning with regularization

In this study we use scalable tree boosting system called XGBoost which is developed over the principle of optimized gradient boost tree (T. Chen & Guestrin, 2016) to predict the *O2D* travel time. It follows the same principle over the gradient boosting postulated by Friedman et al. (Friedman, 2001) with minor modifications for improving regularised

objective function. For a given data set D with n samples (usually number of rows) and m features (usually number of columns) $D = \{(X_i, y_i)\}$ $(|D| = n,\ X_i \in \mathbb{R}^m,\ y_i \in \mathbb{R})$, a tree ensemble model predicts the output by using K additive functions and is given by:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(X_i),\ \ f_k \in F \tag{6.1}$$

Where $F = \{f(x) = \omega_{q(x)}\}(q : \mathbb{R}^m \to T, \omega \in \mathbb{R}^{\mathrm{T}})$ represents the space of regression tree, q represents the structure of each tree with corresponding leaf index and T is the number of leaves in the respective tree. The function f_k represents an independent tree structure q with leaf weights ω. Regression tree model assigns continuous score for each leaf and ω_i represents the score of i^{th} leaf. Further, using the decision rules to classify leaves in the tree defined by q, we calculate the final prediction by summing up the score of corresponding leaves represented by ω. For making the prediction we learn from the set of functions used in the model by minimizing the following objective function:

$$Obj(\phi) = \sum_{i=1}^{n} l(y_i\ ,\ \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{6.2}$$

This is a regularized model and measures the complexity of the tree. Mathematically, it means that our objective function has two components: **training loss**, which measures how well the model fits training data and **regularization component**, which measures the complexity of trees. The **Loss Function** used here is a squared loss, and it is the sum of the squared difference between the predicted value and the actual value which is given as:

$$L(\Theta) = \sum_{i} l(\hat{y}_i\ ,\ y_i)^2$$

The objective function will optimize the trade-off between the training loss and the regularization loss. When we optimize the training loss, it encourages the predictive

models for higher accuracy on training data. However, if we optimize the regularization function, it creates a generalized simpler model for better prediction accuracy. The **regularization component** in equation 6.2 defines the complexity of the tree and is given as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2$$

Where γT represents the number of leaves and $\frac{1}{2}\lambda \sum_{j=1}^{T} \omega_j^2$ represents the L2 norm of leaf scores respectively. Other parameters λ and γ in the above equation are hyper-parameters.

6.3.3 Gradient Tree Boosting

Tree ensemble model represented by equation 6.2 uses functions as parameters so cannot be optimized using traditional method of optimization in Euclidean space. Therefore, in gradient tree boosting, the **training loss** is represented as additive model as we are in the boosting space where trees are sequentially added with each other to finally aggregate an ensemble model. Using the same principle we can see that the prediction function $\hat{y}_i$ from equation 6.2 can be written as follows:

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)$$

Where final model $\hat{y}_i^{(t)}$ is composed of previous model $\hat{y}_i^{(t-1)}$ and a new learning model $f_t(x_i)$. Rewriting our objective function in 6.2 using above terms as an additive model yields:

$$Obj^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega\left(f_t\right) \tag{6.3}$$

Applying Taylor second-order approximation for differentiable function expressed in

equation 6.3 we get:

$$Obj^{(t)} \simeq \sum_{i=1}^{n} \Big[l\left(y_i \, , \, \hat{y}_i^{(t-1)} \right) + g_i \, f_t(x_i) \, + \, \frac{1}{2} \, h_i \, f_t^2(x_i) \Big] + \Omega \left(f_t \right) \tag{6.4}$$

Where g_i and h_i are first and second order differentiation which represents gradient statistics over loss function respectively, and given as:

$$g_i = \delta_{\hat{y}^{(t-1)}} \, l(y_i, \, \hat{y}^{(t-1)})$$

$$h_i = \delta^2_{\hat{y}^{(t-1)}} \, l(y_i, \, \hat{y}^{(t-1)})$$

Removing the constant terms from equation 6.4 and replacing the values of g_i and h_i we deduce,

$$Obj^{\sim(t)} = \sum_{i=1}^{n} \Big[g_i \, f_t(x_i) \, + \, \frac{1}{2} \, h_i \, f_t^2(x_i) \Big] + \, \Omega \left(f_t \right) \tag{6.5}$$

Let us define $I_j = \{ i | q(x_i) = j \}$ as instance set of leaf j and substitute the value of $\Omega(f_t)$ in equation 6.5 we can simplify to:

$$Obj^{\sim(t)} = \sum_{j=1}^{T} \Big[G_j \, \omega_j + \frac{1}{2} \left(H_j + \lambda \right) \omega_j^2 \Big] + \gamma \, T \tag{6.6}$$

where

$$G_j = \sum_{i \epsilon I_j} g_i \; and \; H_j = \sum_{i \epsilon I_j} h_i$$

Eventually, g and h represent the values of each leaf in the trees and G and H represent entire tree structure for the above objective function. The above objective function is the sum of T quadratic functions. Now for each quadratic functions:

$$G_j \omega_j + \frac{1}{2} \left(H_j + \lambda \right) \omega_j^2$$

we can now be able to derive an optimal weight

$$\omega_j^* = -\frac{G_j}{H_j + \lambda}$$

Substituting the value of optimal ω_j^* in original objective function expressed in equation 6.6, we get a new minimal objective function which is the most simplified form of quadratic approximation of the original objective function :

$$minObj = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j + \lambda} + \gamma T \tag{6.7}$$

Above equation 6.7 is used as scoring function in XGBoost to measure the quality of tree structure q. This score is similar to impurity score in other decision tree except that it can serve for the wider range of objective function.

6.3.4 Ranking of Candidate Split points

Finding the candidate split points is one of the major step in approximate algorithm. To ascertain that candidates distribute evenly on the data, as a general rule, percentile of the features are used. Rank function of training data can be defined as:

$$r_k(\beta) = \frac{1}{\sum_{(x,h)\in D_k} h}\sum_{(x,h)\in D_k, x<\beta} h \tag{6.8}$$

where, $r_k : \mathbb{R} \to [0, +\infty)$ represents the instances with feature value k smaller than β. D_k is a set consisting k^{th} feature values and second order gradient statistics of each training instances and is represented as: $D_K = \{(x_{1k}, h_1), (x_{2k}, h_2), (x_{3k}, h_3), ..., (x_{nk}, h_n)\}$. The aim of above equation 6.8 is to find candidate splits points $\{sp_{k1}, sp_{k2}, sp_{k3}, ..., sp_{kl}\}$ which satisfies:

$$|r_k(sp_{k,j}) - r_k(sp_{k,j+1})| < \epsilon \tag{6.9}$$

where $sp_{k1} = minX_{ik}$, $sp_{kl} = maxX_{ik}$ and ϵ is approximation factor which intuitively represent that there are approximately $\frac{1}{\epsilon}$ candidate split points. In this calculation each data point is weighted by h_i. To prove this, let us rearrange the 6.5 as follows:

$$\sum_{i=1}^{n} \frac{1}{2} h_i (f_t(X_i) - \frac{g_i}{h_i})^2 + \Omega(f_t) + C \tag{6.10}$$

which exactly represents weighted squared loss with labels g_i/h_i and weights h_i. In a large data set it is very significant to find the candidate split that satisfies above criteria. An existing algorithm called quantile sketch (Greenwald & Khanna, 2001) can solve the problem of finding the candidate split only if every instance of training data has equal weights. For data set with different weights distributed weighted quantile algorithm is implemented in XGBoost. This algorithm proposes a data structure that can support merge and prune operations and each operation ascertains a certain level of accuracy.

6.4 Data Description and Feature Extraction

6.4.1 Data Description

In this study, we use the following data sets:

- Yellow and green taxi trip data available via the NYC Taxi and Limousine Commission (TLC) website which was collected by different technology providers authorized under the Taxicab and Livery Passenger Enhancement Programs (TPEP/LPEP) from January 1 to July 30, 2016 (*Kaggle*, 2015).

- Taxi trajectory data collected by DIDI ride sharing company for Chengdu and Xi'an city of China from january 1 to November 30, 2016 (Chuxing, 2019).

The trajectory data consists of GPS traces of pick-up and drop-off location recorded in terms of longitude and latitude. It also consist of timestamp when these GPS traces

were recorded and the total duration of the trip. We have around 2.1M trajectory data for New York and 7.07M for Chengdu. The data is split into training, validation and test data sets. Both the data-set were in different formats and had multiple columns that were not useful in our study. We converted both the data-set into similar format and used seven training data columns and six test columns. A sample data set used in this study is depicted in Table 6.1. We check for any missing values in the data and if there is any missing value , the entire row entries is removed.

Table 6.1: Processed data-set used in experiment

New York Data-set						
id	**pickup datetime**	**pickup longitude**	**pickup latitude**	**dropoff longitude**	**dropoff latitude**	**trip duration**
id2875421	2016-03-14 17:24:55	-73.982155	40.767937	-73.964630	40.765602	455
id2377934	2016-06-12 00:43:35	-73.980415	40.738564	-73.999481	40.731152	663
Chengdu Data-set						
eb9dd4095	2016-11-01 01:46:37	104.094640	30.703971	104.089270	30.650850	1710
387a742fa5a3	2016-11-01 07:33:05	104.076509	30.767430	104.063700	30.589510	2090

We make sure that the training and test data are aligned together to prevent model performance mismatch, model over-fitting and under-represent data samples. The alignment of test and train data is depicted in Figure. 6.2 and Figure. 6.3.

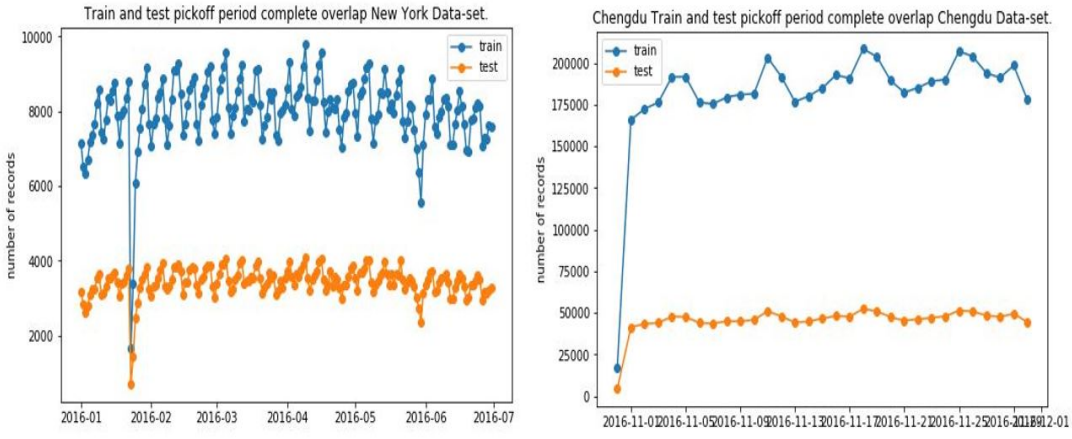

Figure 6.2: Training and test data alignment in pick-up date and time

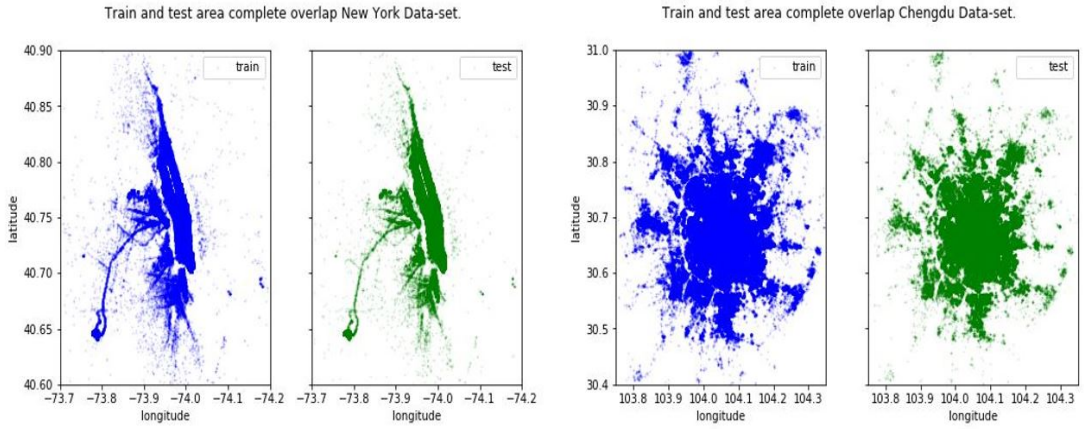

Figure 6.3: Training and test data alignment in GPS locations

6.4.2 Data processing and Feature Extraction

Training data set T can be represented as matrix $T = [A_{rc}]$. Each row in T represents a single observation of the data, and all columns that a row comprised of represent features to describe that observation. We can directly feed raw data to the training model, but learning from these features will be very difficult, resulting in poor prediction accuracy. To increase the learning algorithm's predictive power, engineered features that capture additional information that is not easily visible are extracted using feature extraction techniques. Feature extraction reduces complexity by simply representing each variable in feature space as a linear combination of original input variable (Khalid, Khalil & Nasreen, 2014). In this study, we have extracted multiple features from the raw trajectory data using well-established feature engineering techniques to improve O2D travel time estimation accuracy. Techniques that are implemented in this study are explained below.

Open Source Routing Machine Route Features

Open Source Routing Machine (OSRM) is a high-performance routine machine that usages the open street map network data to provide services like finding the fastest route

between provided coordinates, compute the duration for the fastest route, snapping given GPS points on the road network and finding the optimized path using greedy heuristic (Luxen & Vetter, 2011). Existing studies (D. Wang et al., 2018; H. Yuan et al., 2020; Y. Li et al., 2018) used multiple GPS traces in between the origin and destination points to calculate the actual trip, distance and time which is computationally expensive, we only provided the origin and destination GPS points to the OSRM to find the total distance and total travel time. It also returned total steps which consist of manoeuvres such as turning or merging. The total travel time and distance calculated by OSRM represents the road network distance and time. We created an additional feature by calculating the difference between total trip duration and travel time calculated from OSRM. This difference between real and OSRM calculated travel time results from different dynamic traffic parameters like traffic lights, intersections, driving preferences, weather conditions and congestion. Acknowledgement of this feature in the learning can represent above-listed parameters that directly affect overall travel time.

Logarithm Transformation

Log transformation is the most widely used mathematical transformations in feature extraction. Log transformation handles skewed data and approximates it to more normal by normalizing the magnitude difference. It helps the model to be more robust by decreasing the effect of outliers. We conducted log transformation on total trip duration to extract log duration as a new feature.

Outlier detection and removal

We conducted a manual as well as statistical (employing standard deviation) to identify the outliers. Trip durations too long (greater than 6600 seconds) and too short (less than 60 seconds) were removed to make data more evenly distributed. The distribution of training data after outlier detection and removal operation can be visualized in

Figure. 6.4.

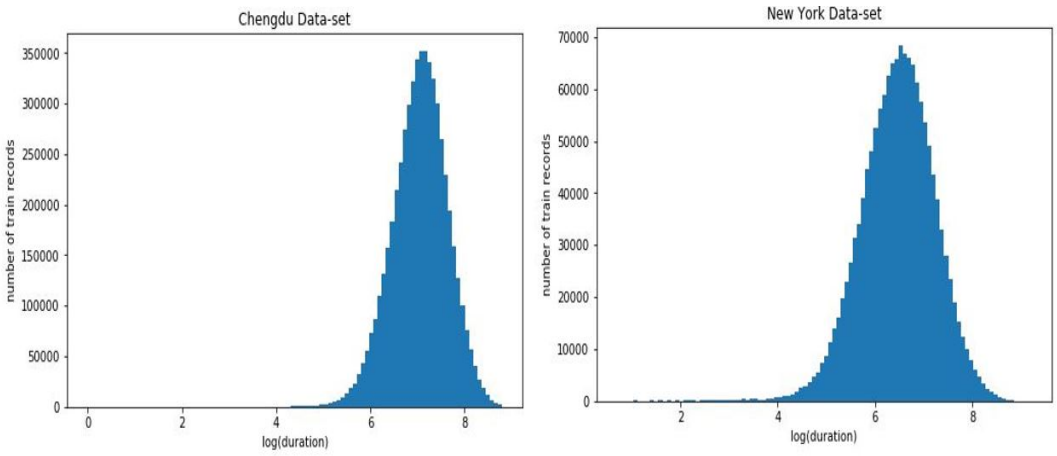

Figure 6.4: Visualization of training data in terms of log duration

Principal Component Analysis Transformation

In a feature space data usually lie near low dimensional manifold or subspace. Principal Component Analysis (PCA) is a dimensionality reduction technique that identifies pattern and correlation among multiple features by identifying the principal component. The lower dimension transformed feature still preserves most of the valuable information that multiple features before transformation had. It is the primarily non-parametric statistical technique and employs unsupervised learning to reduce dimensions of data. PCA rotates the axes so that maximum variability present in data can be visualized easily. PCA constructs axes for the principal component to represent the data and then rank each principal component for the amount of variance captured by each component (Abdi & Williams, 2010). We used PCA transformation to latitude and longitude coordinates for both pick-up and drop-off locations. The low dimension PCA transformed location coordinates still preserved the spatial information present in the original data and is illustrated in Figure. 6.5.

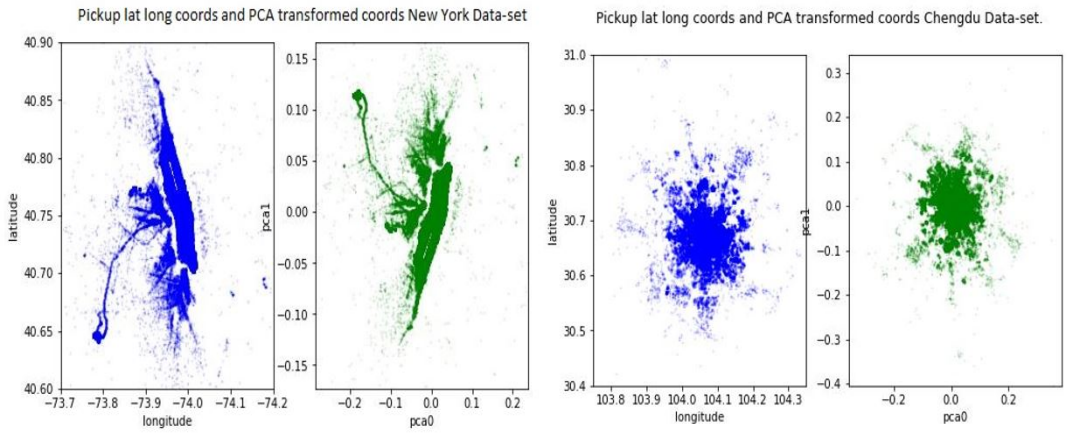

Figure 6.5: Pick-up latitude and longitude with PCA transformed coordinates

Distance features

The easiest information that can be mined from spatial geo-locations coordinates is distance between two GPS points. We calculated Haversine distance to calculates the great circle distance between two latitude and longitude coordinates which is mathematically given as:

$$2\,r\,arcsin\left(\sqrt{sin^2\left(\frac{lat_2-lat_1}{2}\right)+cos(lat_1)\,cos(lat_2)\,sin^2\left(\frac{long_2-long_1}{2}\right)}\right)$$

Where lat and $long$ represent latitude and longitude in radians, respectively, r is the average earth radius in KM, and other mathematical expressions have usual meanings.

Assuming taxi cannot move freely in Euclidean plane, we implemented taxicab geometry to calculate Manhattan distance between pick-up and drop-off locations. Manhattan distance between pick-up and drop-off location is the sum of absolute difference of their Cartesian coordinates. We also conducted PCA transformation on calculated Manhattan distance for pick-up and drop-off coordinates.

The direction between pick-up and drop-off location carries significant information for training model. Whenever we follow a great circle path, initial heading and the final heading changes and is dependent on the distance and latitude. In navigation, we use **bearing** (also referred to as forward azimuth) to calculate the direction between two

GPS coordinates. We have calculated bearing angle θ using the following mathematical formula:

$$\theta = atan2(x, y)$$

where $x = sin(long_2 - long_1) * cos(lat_2)$ and

$y = cos(lat_1)sin(lat_2) - sin(lat_1)cos(lat_2)cos(long_2 - long_1))$

Temporal feature extraction

Extraction of temporal features from the pickup date-time data enhances the learning. Travel time between two locations does not only depend on the distance but also a lot of other factors. Time-dependent vehicle routing problem has been widely studied. The travel time depends on the time of the day, day of the week and the particular week of the year. We extracted multiple temporal features to aid model learning. From the Haversine and Manhattan distance, we try to profile the average speed of each trip with respect to the temporal features. The rush-hour average speed is depicted in the Figure. 6.6.

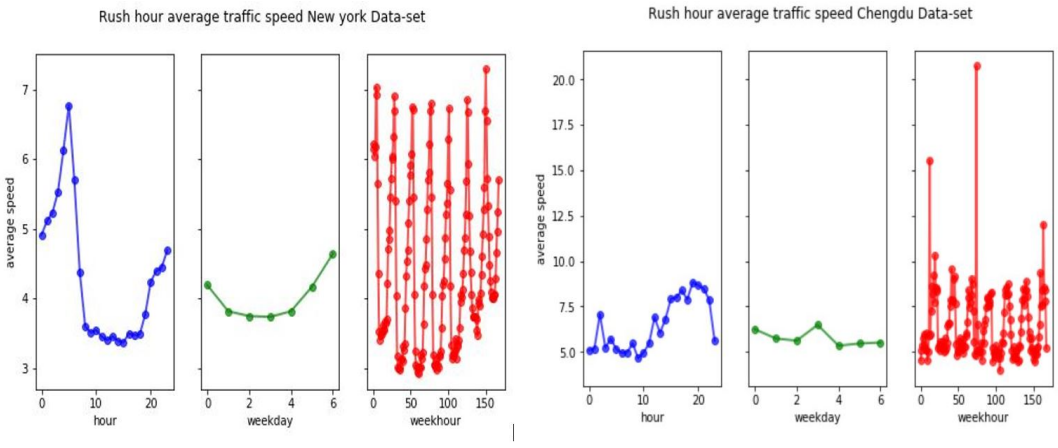

Figure 6.6: Average speed profile in m/s

Similarly to link the speed profile with the spatial data we divided the entire city into

different regions by binning the latitude and longitude values. We use scatter plot for diverging map to plot the speed profile for each region which is shown in the Figure. 6.7.

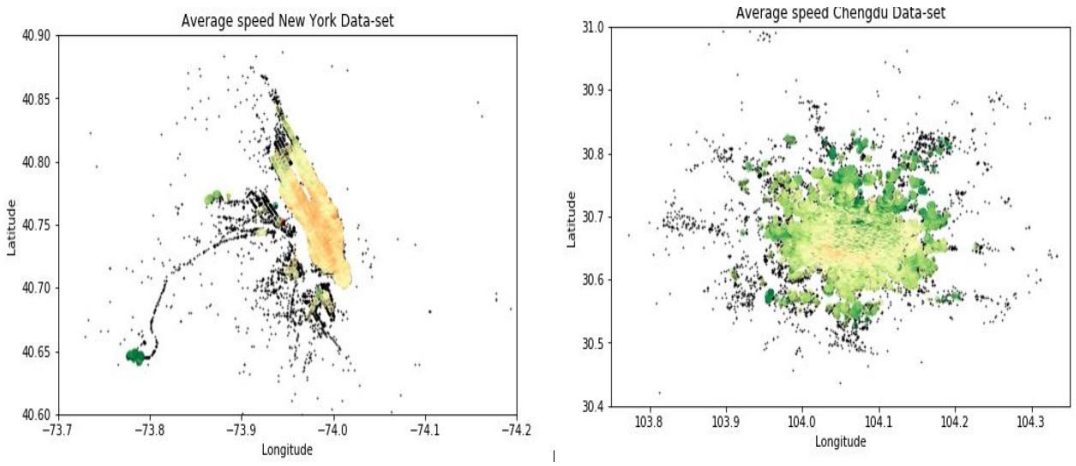

Figure 6.7: Average speed profile in m/s for locations clusters

Pick-up and drop-off clustering

We used k-means clustering method to divide pick-up and drop-off locations into clusters. We rank different clusters according to the number of trips originated and terminated from one cluster to the other. Clusters with larger numbers of trips get a darker color. For each cluster, we calculate the frequency of trips within one hour. Additionally, we also identified that there are two international airports for New York data-set within our GPS coordinate range: John F. Kennedy International airport and LaGuardia airport. For Chengdu data-set there is Chengdu Shuangliu International Airport. We extracted pick-up and drop-off coordinates that originated or were a destination to these locations as features. The overall visualization of New York and Chengdu clusters is pictured in Figure. 6.8. We also used daily accident data within our GPS range to profile the cluster's average speed where accidents have occurred.

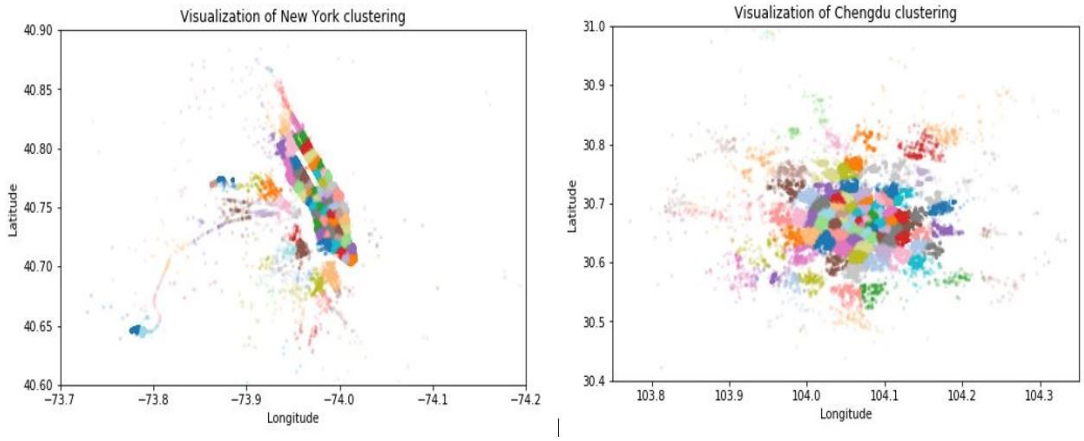

Figure 6.8: Average speed profile in m/s for locations clusters

6.5 Experiments and Results

6.5.1 Evaluation Metrics

In this study, for the evaluation of O2D travel time estimation of our model, we employ three widely used matrices: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and Root Mean Square Log Error (RMSLE). RMSLE is the default evaluation metric in XGBoost for linear regression. MAE and MAPE were additionally computed to compare the performance of our model with existing baseline models. As mentioned earlier in Section. 6.3, let model predicted value is y_i' for the actual value of y_i (which is the ground truth) then:

$$MAE(y_i', y_i) = \frac{1}{N}\sum_{i=1}^{N}|y_i' - y_i|$$

$$MAPE(y_i', y_i) = \frac{1}{N}\sum_{i=1}^{N}|\frac{y_i' - y_i}{y_i}|$$

$$RMSLE(y_i', y_i) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(log(y_i' + 1) - log(y_i + 1))^2}$$

6.5.2 Settings of regularization parameters

We used 60% of the data for training, 20% for validation, and the remaining 20% for the testing, which helped overcome the model over-fitting. XGBoost model was implemented using Python, and its performance was tested using different combinations of regularization parameters. Following regularization parameters were used while developing the model:

- eta : Defines the learning rate of the model, *eta* helps in shrinking the feature weights to prevent overfitting and makes the system more conservative. Range: [0, 1]; default value: 0.3; values implemented in the model [0.01, 0.02, 0.05, 0.1].

- $min\_child\_weight\,(MCW)$: Minimum sum of instance weight required for tree to continue partition. Further partitioning stops when tree partition results in leaf node with sum of instance less than min_child_weight. Range: [0, ∞]; default value: 1; values implemented in the model [3, 5, 6, 10, 20].

- $max\_depth\,(MD)$: Defines the depth of the tree. Higher value results in complex model and may result in overfitting. Range: [0, ∞]; default value: 6; values implemented in the model [6, 8, 10, 12, 15].

- $sub\_sample\,(SS)$: Defines sub-sample ratio of particular training instance. Range: (0, 1]; default value: 1; values implemented in the model [0.5, 0.6, 0.7, 0.8, 0.9].

- $colsample\_by^{*}\,(CS)$: Consist of multiple parameters to sub-sample columns which include $colsample\_bytree$, $colsample\_bylevel$ and $colsample\_bynode$. All parameters have range: [0, 1]; default value: 1; values implemented in the model [0.3, 0.4, 0.5], and

- $lambda\ (\lambda)$: Which is L2 regularization terms on weights, setting this value

greater than 0 results in an elastic regularization. Range: [0, ∞]; default value: 1; values implemented in the model [0.5, 1.0, 1.5, 2.0, 3.0]

6.5.3 Performance of GBO2D model

We train our model using different combinations of above regularization parameters. Initially, we start training the model with one set of parameters. We set early stopping point at 50 rounds if the system's performance improves we continue with these parameters else we stop and choose different combinations. The snapshot of RMSLE values for different regularization parameters are pictured in Fig. 6.9. Finally, We extracted the best score for MAE, MAPE, and RMSLE from all the combinations which are tabulated in Table. 6.2.

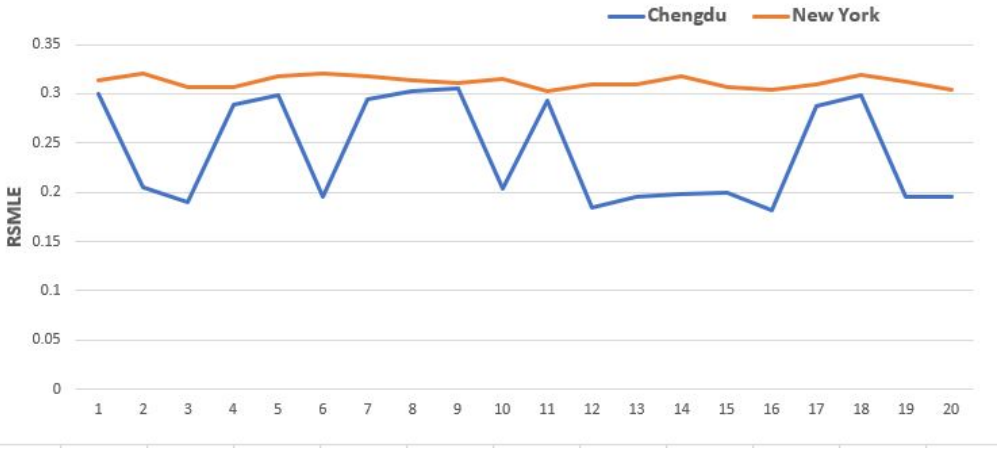

Figure 6.9: RMSLE with different regularization parameters

Table 6.2: Performance of GBO2D

Evaluation Metric	**Data-set**	
	New York	**Chengdu**
RMSLE	0.3042	0.1818
MAE	126.36	91.2628
MAPE	25.455	18.252

6.5.4 Ranking of features in GBO2D learning

We implemented different feature extraction techniques over both New York and Chengdu data sets and were able to extract 76 and 72 features. To explore the influence of different extracted features in GBO2D travel time estimation process, relative contributions of extracted features were calculated and extracted. From Table. 6.3 we can visualize the top 15 features that contribute to O2D travel time estimation. For New York data-set, the trip's direction contributes most in the prediction of O2D travel time estimation. Similarly, JFK_drop , pickup_dt, total_travel_time and total_distance are the respective top five contributors. Whereas, pickup_minute, pickup_latitude, pickup_longitude, pickup_pca0 and distance_manhattan are top five contributors respectively.

Table 6.3: Top Extracted Features with their corresponding ranks

Rank	Features New York data-set	Features Chengdu data-set
1	direction	pickup_minute
2	JFK_drop	pickup_latitude
3	pickup_dt	pickup_longitude
4	total_travel_time	pickup_pca0
5	total_distance	distance_manhattan
6	pickup_longitude	center_latitude
7	JFK_pick	center_longitude
8	distance-haversine	dropoff_longitude
9	dropoff_latitude	distance_haversine
10	dropoff_longitude	pca_manhattan
11	distance_manhattan	direction
12	pca_manhattan	pickup_pca1
13	LGA_pick	pickup_dt
14	pick_latitude	CSIA_drop
15	LGA_drop	dropoff_latitude

6.5.5 Comparison with baseline methods

We compare the GBO2D model with two of the most recent baseline methods for estimating O2D travel time implemented using deep NNs. They claim to outperform all the existing models to date:

- MUlti-task Representational Learning (MURAT): Extract representational features using unsupervised graph embedding approach based on underlying map data. Incorporated spatial and temporal embeddings using grids and used deep

NNs to predict O2D travel time (Y. Li et al., 2018)

- DeepOD: Moreover like MURAT, matches OD pair with respective trajectories in road network embeddings. Uses timestamp associated with the trajectories for time slots embeddings. Encodes the spatial and temporal features of trajectory and external features and finally combine using deep NNs to estimate O2D travel time estimation. Claims to outperform most of the existing baseline methods (H. Yuan et al., 2020).

Environment Settings: Since both MURAT and DeepOD models are implemented using deep NNs we follow the instructions mentioned in their respective documentation to replicate the experiments and use MAE and MAPE as evaluation metrics. Both MURAT and DeepOD use intermediate GPS trace in their experiment, whereas we have not used any intermediate GPS trace to estimate that travel time estimation is pure O2D. Therefore, we replicate MURAT and DeepOD first using intermediate GPS trace exactly as in their experiments and next by only using origin and destination GPS trace. The experiments were implemented with TensorFlow 2.3.0 and Python 3.7.3. and trained with two Nvidia TITAN RTX GPU. The XGBoost model was implemented with Python 3.7.3, pandas 0.24.2, XGBoost 1.21 using Jupyter notebook under similar settings. All the platform ran on windows OS. The performance of GBO2D compared to existing baseline models is tabulated in Table. 6.4 below.

Table 6.4: Performance Comparison of GBO2D

Dataset	**New York / Chengdu**	
Metrics	**MAE**	**MAPE**
MURAT (Original)	144.12 / 131.74	27.451 / 29.26
DeepOD (Original)	128.26 / 97.61	26.78 / 20.83
MURAT (Only OD GPS trace)	207.67 / 153.25	35.82 / 33.26
DeepOD (Only OD GPS trace)	137.26 / 118.38	29.412 / 21.18
GBO2D	126.36 / 91.2628	18.33 / 18.252

From the table above we can summarize:

- GBO2D achieves best performance for both data-sets in terms of MAE and MAPE as shown in Table. 6.4.
- Performance of MURAT and DeepOD is worse when we consider origin and destination GPS trace only. This is because deep NNs cannot learn valuable information only from the raw GPS data.
- Similarly, the performance of both MURAT and DeepOD improves considerably with the increase in the size of data-set (a basic feature of deep NN).
- GBO2D uses ensemble-based gradient boosted regression tree and can outperform deep NNs when valuable features are manually extracted from the provided raw structured data-sets. Domain knowledge is fundamental in such cases.
- Since in GBO2D, we can visualize each feature's contribution to travel time estimation. The performance can always be improved by replacing less contributing features with other features having significant contributions.

6.6 Conclusion and Future work

In this paper, we proposed a novel solution for Origin-to-destination (O2D) travel time estimation named GBO2D using gradient boost regression that learns from multiple pre-engineered features related to topology, directions, the shape of the trajectory, path, speed limits, map distance traversed, real-time traffic conditions and compare it with existing deep neural network-based learning process using raw GPS trajectories. We employed a better and simplified technique to predict travel time, generating superior results than deep neural networks. We did not consider intermediate GPS trace making our model pure O2D travel time model and less computationally expensive. We ranked and visualized individual features' contributions in the entire learning process which eliminated the perpetual issue of interpretation and visualization in deep learning and perceiving it as a black box. We concluded that for wisely extracted manual features, ensemble-based gradient boost regression approach could outperform existing state-of-art baseline models that employ deep neural networks.

In future, we would like to compare the performance of GBO2D with huge data-set (larger than 50 million). We would also like to explore the opportunities of implementing GBO2D with limited feature engineering but still maintaining similar prediction accuracy.

Chapter 7

Routing Autonomous Emergency Vehicles in Smart Cities Using Real-Time Systems Analogy: A Conceptual Model (Manuscript 6)

This paper "Routing Autonomous Emergency Vehicles in Smart Cities Using Real-Time Systems Analogy: A Conceptual Model" proposes a conceptual model of routing EVs in smart cities. Routing EVs and task scheduling in Mixed criticality real-time system (MCRTS) are considered analogous. We use design-by-analogy approach (Verhaegen, D'hondt, Vandevenne, Dewulf & Duflou, 2011) to convert traffic network parameters into MCRTS parameters using different task functions. This allows us to use sophisticated task scheduling algorithms developed for complex MCRTS like aircraft systems for EV routing. Also, we have designed this model to route autonomous vehicles (AVs) to serve emergencies. We explore the idea of using AVs in normal mode and emergency mode. Details of the paper can be viewed from the attached manuscript 6

Routing Autonomous Emergency Vehicles in Smart Cities Using Real Time Systems Analogy: A Conceptual Model

Subash Humagain and Roopak Sinha, Senior Member, IEEE Department of Information Technology and Software Engineering, Auckland University of Technology Auckland, New Zealand

AUT Tower, Level 7, 2-14 Wakefield Street, Auckland, New Zealand

Tel: +64-9-921 9999 Ext.6376;

corresponding author: subash.humagain@aut.ac.nz

7.1 Introduction

The evolution of present cities into smart cities of the future has provided assurance of easing the way we live. Smart city is mainly focused on urban environment which offers advanced and innovative services to inhabitants to improve the quality of life using information communication technology (ICT)(Piro, Cianci, Grieco, Boggia & Camarda, 2014).These advanced and innovative services will help us in solving several current problems easily like traffic congestion, digital security, mobility etc. that are hard to solve using existing technologies. Having impacts on different dimensions, road congestion is one of the major challenges urban planners, traffic authorities and communities are struggling to address. Among different impacts of road congestion, increased response time of emergency vehicles (EVs) like ambulance, fire, police etc. is most severe as it can have an irreparable loss in terms of life and property. According to(RAPIDSOS, 2015),one minute in response time increases mortality by 1% which leads to a 7 billion dollars increase in healthcare expenses yearly. To solve the underlying traffic management problem and overcome losses caused by increased congestion, we need advanced ICT-based solutions. Smart cities especially smart

transportation provide an ideal environment to implement such solutions.

It is intuitively understood that EVs must get preference over other vehicles when they are traveling to the response scene. EVs get priorities by using special color, sirens and strobe lights, dedicated green light on approaching traffic signals, special lanes etc. and they travel to service an emergency in an optimized route. To measure the effectiveness of optimization and pre-emption techniques, emergency management services companies are set with a target time to respond to different level of emergencies. For example, St. John's of New Zealand categorizes life-threatening alerts as purple and red, and less threatening events as orange. The contractual target of the purple and red incident is to respond to 50% of cases within 8 minutes and 95% within 20 minutes (St.John's, 2016). In the UK and Canada the target is 75% of purple and red cases within 8 minutes (England, 2015), 90% of similar cases within 8 minutes 59 seconds in the USA (Pons & Markovchick, 2002b), 50% of cases within 10 minutes in Australia (Service, 2016), and 92% of cases within 12 minutes in Hong Kong (Fitch, 2005a).

Over the years, there has been no significant decrease in EV response time (Gedawy, 2009) because contemporary traffic networks constitute multiple hurdles to the timely movement of EVs. For instance, synchronized operation of traffic lights, increased pedestrian population over cities, continuous construction over lanes and prominently congested road networks have regularly obstructed smooth movement of EVs. In addition, 90% of EVs accidents are caused by human errors. The safety and effectiveness of EVs' movement can be improved if we have access to dynamic road parameters like road congestion, pedestrian flow, travelling time, men at work, halt at road and queued vehicles in real-time and they can be processed to make intelligent driving decisions.

There has been a massive investment in smart cities both from public and private sectors. large ICT business leaders like IBM, Intel, Siemens, CISCO and SAP are putting huge effort in developing revolutionary concepts for smart cities. Not only these companies but also governments, philanthropic organizations, and academics are

advocating for smart cities. The global smart city market is expected to be valued at US$1.565 trillion in 2020(Glasmeier & Christopherson, 2015) and number of smart cities to be 88 by 2025 (Technology, 2014). As current technology seems insufficient and growth of smart cities is inevitable there is an immediate need to develop ICT driven EV route optimization and pre-emption technique to meet overwhelming interest and investment.

Smart cities are built on the idea of deep connectivity. Vehicles have access to vehicle-to-X (V2X: vehicle, road, human, infrastructure, internet) communication through several protocols (Yaqoob et al., 2017). This connectivity can help in optimizing EV routing. Connectivity gives access to information on dynamic road parameters in real-time. Connectivity also enhances information sharing among smart objects. Present EV routing systems have not blended in real-time traffic data to generate accurate, dynamic and reliable routes for EVs (Musolino, Polimeni, Rindone & Vitetta, 2013b). In the connected environment of a smart city, we can react to dynamic parameters in real-time so that EVs can respond to all levels of emergencies within a certain time. The concept of resource allocation and meeting the timing constraint make EV routing analogous to task scheduling in the real-time system (RTS). In addition, emergencies having several levels of criticality with different service times, which makes the EV routing problem very close to scheduling in a mixed-criticality real-time system (MCRTS). MCRTSs have tasks with two or more levels of criticality, for example, non-critical, safety-critical and mission-critical components. In MCRTS timing parameters like worst-case execution time (WCET) for processes rely mainly on criticality levels(Burns & Davis, 2013).

In this paper, we propose a conceptual model of routing EVs in smart cities. Routing EVs and task scheduling in MCRTS are considered analogous. We use design-by-analogy approach(Verhaegen et al., 2011) to convert traffic network parameters into MCRTS parameters using different task functions. This allows us to use sophisticated

task scheduling algorithms developed for complex MCRTS like in aircraft systems for EV routing. In addition, we have designed this model to route autonomous vehicles (AVs) to serve emergencies. This kind of AVs are termed as autonomous emergency vehicles (AEVs)(Newby, 2015). So the model is based on a novel idea of routing EVs using task scheduling in MCRTS for autonomous emergency vehicles (AEVs) that can meet the critical response time and drive through a complex road network in smart cities efficiently and safely.

The approach discussed in the preceding paragraph has multiple contributions for researchers and industry partners. This approach:

1. explores the idea of using AVs in normal mode and emergency mode. The use of AEVs increases traffic safety and connectivity, and the are described in Sec. 7.2.

2. provides an insight that routing of EVs/AEVs can be done using modern scheduling algorithms developed for MCRTS. For this, it presents an analogical mapping framework in Sec. 7.3 .

3. suggests using dynamic optimization method to find routes for AEVs in smart-cities leveraging access to real-time traffic data in Sec. 7.4.

4. focuses on multiple levels of emergencies having a different response time. Using MCRTS helps emergency management services to meet or exceed the minimal contractual standards of response time.

5. provides a detailed view of how users, AEVs and real-time traffic management systems communicate with each other.

7.2 Autonomous Emergency Vehicles (AEVs)

Autonomous vehicles can sense their surroundings and can move with no or very little human interference (Krasniqi & Hajrizi, 2016). A central computer within AV analyze and processes the information received from sensors like GPS, LIDAR, video cameras, radar, ultrasonic sensors and then controls steering, brakes, and accelerator in accordance with the formal and informal rules of the road. With the DSRC system it can communicate with its surrounding.

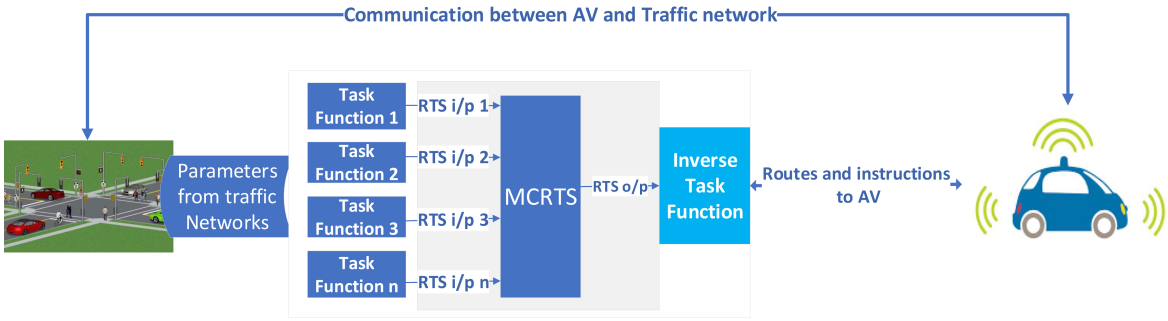

Figure 7.1: MCRTS design diagram.

AVs that are used to serve emergencies are termed as Autonomous emergency vehicles (AEVs). There can be two categories of AEVs. First normal AVs which can also serve for emergencies of lower criticality and second custom-designed AVs e.g. autonomous ambulances. These kinds of AEVs have facilities built within to serve a particular purpose like autonomous ambulance have paramedic facilities. The distinguished property of EVs is that they get priority when they move. AEVs can also get priority by requesting for lane reservation, continuous green light, change of speed limit etc. For this, they are equipped with different communication transmitters and receivers like DSRC, 5G networks etc.

Using of AEVs in place of the traditional emergency service vehicles have the following benefits:

- Use of AEVs will reduce response time and meet or exceed the minimal contractual standards.
- According to National Highway Traffic-Safety Administration only in the USA there is an average of 4500 accidents involving ambulances each year, 3160 accidents involving fire vehicles and 300 fatalities during police pursuit(NHTSA, April 2014). Use of AEVs improves safety on roads. Fewer crashes as they are without driver error (McAllister et al., 2017).
- Processing of traffic network data allows AEVs to avoid congestion which in turn contribute to less carbon emission due to fuel burning(Greenblatt & Shaheen, 2015).
- Provide better mobility to elderly, young and child(Alessandrini, Campagna, Delle Site, Filippi & Persia, 2015).
- AEVs are able to generate and request pre-emptive request like creating a green wave, lane reservation, informing other vehicles, changing speed limit, use of reverse lane with minimal or no disturbance to other traffic using V2X communication technology (Kokuti, Hussein, Marín-Plaza, de la Escalera & García, 2017).
- Locating, instructing and tracking is easier as they are always connected(Kokuti et al., 2017).
- Use of AEVs reduces massive expenses in infrastructures like traffic lights, lanes, instructions etc. as these things will be stored in the memory of the vehicle and can be utilized virtually (Malikopoulos, Cassandras & Zhang, 2018).

Due to higher level of connectivity among the infrastructures, physical infrastructures presently treated as barriers in solving traffic problems can be used like functions

which can return values whenever we require. In such condition, the major aim of emergency traffic management system is to align all infrastructures in such a way that emergency vehicles moving within these connected traffic network can respond to emergencies within a predefined time. This means in smart cities with AEVs routing of emergency vehicles present complex infrastructure problems converts into mere timing problems. This provides us with an opportunity of solving routing of emergency vehicles as MCRTS task scheduling problem because success or failure of MCRTS is completely dependent on solving a task within a stipulated time. The following section explains the relevance of AEV routing in smart cities using MCRTS analogy.

7.3 Mixed Criticality Real Time System Concept

In emergency response systems used presently, EVs are located at a certain location. Once there is any emergency call, the level of emergency is determined, and response time is set. The number of available EVs with their location is identified. From the present location of the EV to destination there may exist multiple routes. The optimization system must provide the fastest route to serve within a definite period considering different factors associated with the particular route that may create hindrances in the movement of EV. Next, the system schedules the EV to serve the emergency. If it becomes difficult to respond within stipulated period due to changing road parameters, the system must be able to provide an alternate route or activate pre-emption to provide priority to EVs so that they can respond to the emergency on or before the set time.

Producing a physical result within a certain time is the basic property of real-time systems (RTS). Inputs from sensors are taken at a periodic interval and real-time computer must send responses to actuators within chosen time. The ability of system to produce the results within this chosen time is completely dependent on the system's ability to process necessary computations within dedicated time. In case of concurrent

events the system must schedule the computations to completed within time. Every task in RTS bears a timing property within which it needs to be processed. While scheduling any task this timing property must be considered by RTS. Therefore, in RTS the accuracy does not only depend on logical results from computation but also on the time when these results are produced. System failure occurs when the system cannot meet this timing property. Therefore, it is indispensable to guarantee the timing property of the system. To guarantee timing behavior, the system must be predictable which means once a task is activated, we must be able to determine its completion time with full confidence (Ramamritham, Stankovic & Shiah, 1990). A real-world RTS is usually composed of multiple tasks with multiple criticality levels. If the system fails to meet the timing constraints, we must designate some level of assurance against failure depending upon the criticality of the task. This kind of RTS are termed as mixed-criticality real-time systems (MCRTS) (Ernst & Di Natale, 2016).

From the above discussion, we can conclude that there exists certain similarities between routing of EVs and task scheduling in mixed-criticality RTS. There is only one difference between these two approaches. In contemporary emergency response systems, EVs and surrounding cannot communicate with the environment except the use of light strobes and sirens but in smart cities, all the components relate to each other and they can communicate. Smart city provides a connection platform where all the homogeneous smart object communicate using prescribed communication standards. Utilization of interacting traffic resources to process the task of responding to different level of emergencies within precalculated time is like task scheduling in MCRTS. Modeling AEVs routing using MCRTS task scheduling meets the following goals:

- Meeting timing constraints of different emergency responses with different level of criticality.
- Emergency vehicles meet the target response time utilizing available resources

but ascertain that other vehicles also make the optimum use of the same resource.

- Though pre-emption is activated it causes the nominal effect to other vehicles.
- Reducing the communication cost between the components of the traffic network system.
- Considers all level of emergencies in terms of critical tasks.
- Scheduling of EVs in real-time system whose behavior is intelligent, dynamically adaptive, reflexive and reconfigurable.

Parameters of MCRTS are dependent on the level of criticality of each task. Estimates of worst-case execution time (WCET) for any task is also dependent on level of criticality. For example, the same task can have a lower WCET target if it is assigned as safety-critical task rather than mission-critical or non-critical tasks(Burns & Davis, 2013). This attribute of MCRTS align completely with our AEVs routing problem where we have different level of emergencies with corresponding response time. In the following subsection we have discussed analogical mapping between MCRTS and traffic network parameters.

7.3.1 MCRTS and Traffic Network Analogy

Generation of creative ideas for design and problem solving can be sometimes interpreted from similarity of products, shared functionality or shared relation between items of different domains. This kind of design methodology is termed as design-by-analogy(Verhaegen et al., 2011). AEV routing and task scheduling in MCRTS have certain similarity. A MCRTS is a system which provides a certain level of assurance against system failure for some critical tasks. This kind of system exactly matches with the design of the system where AEVs can be used to respond to different level of

emergencies. A certain time is allocated for AEVs to respond to a certain emergency case. If it can be responded within that time a task success is noted. In any other case system provides some flexibility to the timing constant so that more resources can be assigned to complete the task within WCET. Usually, a MCRTS system comprises of multiple inputs and outputs. Some of them have been listed in Table 7.1.

Table 7.1: Inputs and outputs in MCRTS

Inputs	**Outputs**
Number of periodic, aperiodic or sporadic tasks	Assigning task to processor
Number of Pre-emptive and non-pre-emptive tasks	Assign new deadline
Number of Fixed or dynamic priority tasks	Queue task
Number of Independent or dependent tasks	Alter priority
Number of processors	Assign pre-emption etc.
Number of reserved processor	
Release time, completion time	
deadlines, priority, precedence, constraints	

MCRTS generates outputs like assigning task to processor or assign new deadline considering input parameters like number of available processor or completion time of present task. There are algorithms to achieve this. With analogical mapping we convert real world traffic network parameters into equivalent MCRTS parameters. This can be achieved by using tasks functions as shown in Figure 7.1. We use compositional analogy for mapping of traffic network variables with variables of MCRTS. It first does mapping at level of structure, and that mapping at a level of structure transfers some information. That in turn allows to transfer information at the behavioural level. Once information at behavioural level is transferred, it climbs up this abstraction hierarchy, and transfers information at a functional level(A. K. Goel & Bhatta, 2004).

For example, real-world traffic network have inputs like EV location, destination, possible routes, congestion level of road network, previously selected route, halt on road, speed limit of the road, no of lanes, likely speed, travel time from previous user, time of day, slope on road, no of traffic nodes, roundabout, traffic lights, pedestrian flow, queued

vehicles, recovery time of traffic pre-emption, traffic phase timing etc. Task functions use analogical mapping to maps these traffic domain input parameters to MCRTS inputs. Now this allows us to use properties of MCRTS. The outputs of MCRTS is now passed through inverse task functions which finally convert MCRTS output into equivalent traffic parameters. For example assign processor can be equivalent to assign route.

In the following section, we have elaborated the concept of modelling AEVs routing using MCRTS task scheduling discussed in section 1, section 2 and section 3 using mathematical notations and different diagrams.

7.4 A Conceptual Model of the system

In this section, we are introducing a novel model of route optimization and pre-emption for different types of AVs. We assume that the service function $S(e, c, r, v, p)$ is the set of all the attributes required to route an AV in AEV mode from a source to destination. A simplified diagram to visualize the process of how AVs service is represented in Figure 7.2. Here, e stands for level of emergency, c stands for the level of criticality, r represents the number of available routes, v represents the number of available autonomous vehicles and p stands for the type of pre-emption to be activated. The process is defined below:

- Mode (E_i) represents the service mode of AVs. It can take two values E_o and E_1. E_o represents AVs are serving in normal mode and E_1 represents AVs are operating on AEV mode. These values are updated by the user who requests the AV service.

- Criticality (C_i) is level of criticality of emergency that AVs are going to serve. From normal practice in different countries, we can have total of four values of criticality C_o....., C_3. C_o represents no critical emergency case so AVs can operate

in normal mode. C_3 and C_2 are life-threatening alert that are symbolised as purple and red by emergency management service companies and C_1 are orange cases which are less life-threatening. These values are updated by the user by answering certain questions that appear in their application.

- Route (R_i) is the number of available routes from source to the destination of the service. It can take any values from R_o.....,R_n depending upon the number of traffic nodes available in that particular geographical location. The optimization function calculates the fastest path considering all the road parameters and supply the value to the system.

- Vehicle (V_i) represents a number of available AVs to serve. V_o.....,V_n are the possible types of AVs. AVs are normal autonomous car that can also serve in cases of less critical emergencies like user needs to visit hospital and doesn't require any paramedic support during travel. Other AVs can be autonomous ambulance, fire, police car etc. GPS system installed inside AVs and their service notification status give the value of (V_i).

- Pre-emption (P_i) represents the instruction to the AVs weather to activate pre-emption or not. It also instructs which type of pre-emption to activate like creating a green wave, lane reservation, informing other vehicles, changing the speed limit, use of reverse lane with minimal or no disturbance to other traffic etc. It can take values from P_o.....,P_n. P_o symbolizes no use of pre-emption and all other values are the type of pre-emption the system suggests to activate.

The suggestion of pre-emption is guided from the calculation of estimated arrival time (ETA). As we are suggesting MCRTS approach, we can compare this ETA with WCET for AVs to serve. Once all the values of service function $S(e, c, r, v, p)$ is calculated the user and AVs are advised with the service time. During the service, if

there is any change in the dynamic traffic parameters like pedestrian flow, congestion, traveling time, men at work, halt at road, queued vehicles etc. a new updated ETA are advised to user and AVs with proper pre-emption instruction.

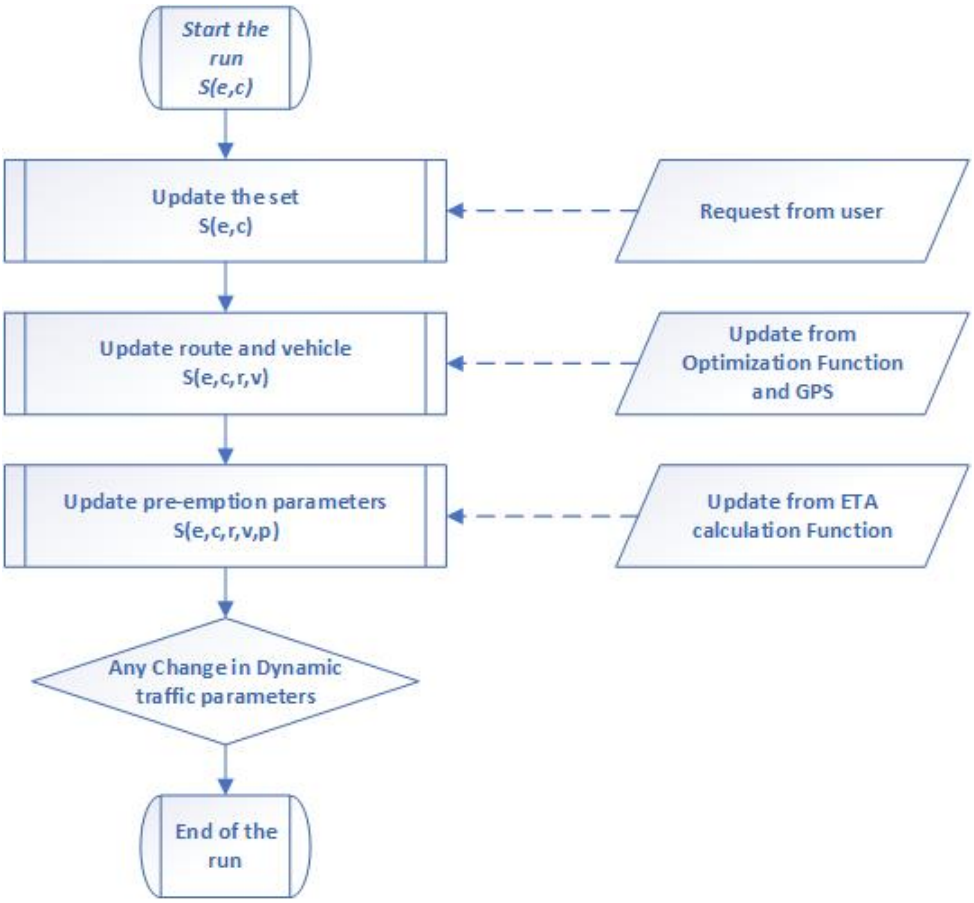

Figure 7.2: Execution of service for AEVs

The core of our conceptual model is visualization of AVs as AEVs. User using a simple mobile app can initiate this service and most of AVs can serve in different types of emergencies as AEVs. The system is dynamic and keeps on updating in real time with the help of data received from the array of sensors installed inside AVs and environment. The entire system has five components mobile application for users, AV sensory system, AV control system, dedicated short-range communication (DSRC) system and real-time traffic control system. All these systems and their interactions are shown in Figure 7.3.

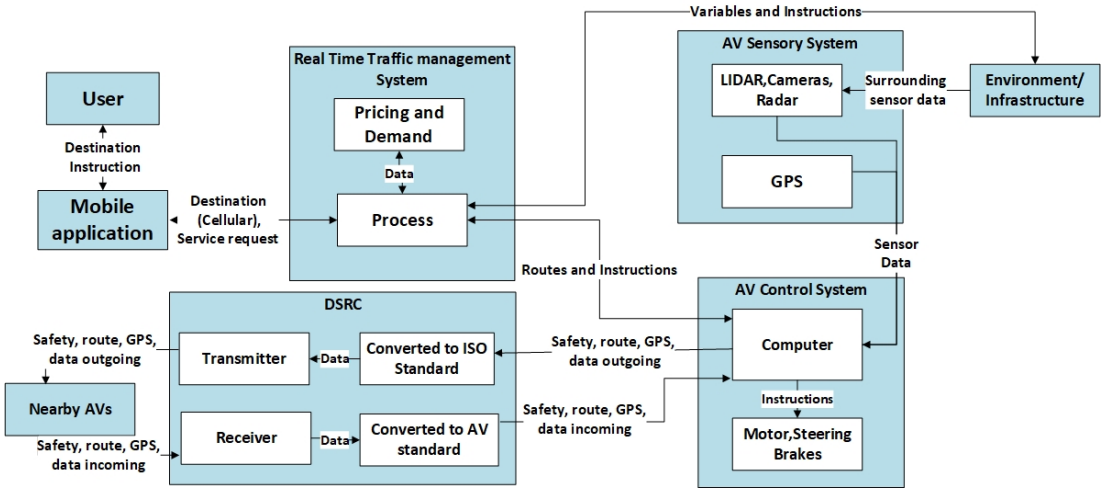

Figure 7.3: System interface diagram

The solution provided has five major components:

- *Mobile Application for user:* This component is focused on user side application where different users request AV's service. They can request for any kind of emergency service. Their request sets the AVs to operate in AEV mode. Their response to certain questions can also set the criticality level of the emergency, source, and destination of the service and also the type of EV to be scheduled to serve the emergency.

- *AV sensory system:* This comprises of different sensors that AV is installed with like LIDAR, radar, ultrasonic sensors, GPS, video camera etc. These all sensor help AV to visualize the environment.

- *AV control system:* AV's central computer processes all the sensor data and instructions received from real-time traffic management system, processes it and generates driving instructions for motor, steering, and braking.

- *Dedicated Short-Range Communication (DSRC) system:* This system permits AV to establish V2X communication using different transmitters and receivers.

- *Real-time traffic control system:* The core idea of our conceptual framework lies within real-time traffic control system. The system gets service mode information (E_i) and level of criticality (C_i) and type of vehicle (V_i) to deploy to serve from the user using mobile application. Once these parameters are known the system initiates the optimization function and returns the route with its associated ETA. This ETA is now set as WCET of the MCRTS. All the available resources are now assigned to execute the task of responding to the service (emergency or non-emergency) within WCET.

7.5 Conclusion

In this paper we introduce a conceptual model of routing autonomous emergency vehicles to respond to emergencies using a mixed-criticality real-time systems (MCRTS) approach in smart cities. To use the highly refined scheduling algorithms of complex MCRTS we have suggested the use of analogical mapping between traffic network parameters and MCRTS. The preliminary goal of this paper is to use autonomous vehicles as part of emergency response system in smart cities termed as autonomous emergency vehicles. Through an analogical mapping between MCRTS and the AEV routing problem, we propose a framework to route AEVs using dynamic traffic parameters. The proposed framework can cater to different levels of criticality for AEVs, and can be extended for controlling traffic networks in general.

Chapter 8

Conclusions and Future Work

8.1 Summary

Reducing emergency services' response times requires identifying the most effective optimization and pre-emption techniques and factors for reducing emergency vehicle (EV) travel times. Expedited movement of EVs is dependent on the following factors:

- Finding the fastest route from origin to destination for EVs.
- Identification of EVs when they approach an intersection.
- Determining whether prioritization is required or not.
- If pre-emption is required, determining right time and duration of pre-emption.

This thesis develops and implements novel effective pre-emption and dynamic route optimization techniques leveraging recent advancement in communication technology and the availability of huge trajectory data-sets that aid in advancing above listed dependent factors. We proposed different pre-emption techniques and travel time estimation models. We empirically proved that our models could better reduce EVs' response time than current state-of-art technologies.

8.2 Conclusions

This thesis implemented an incremental approach in designing effective pre-emption and dynamic route optimization. We divided the problem of route optimization and pre-emption system for EVs into three different sub-problems:

- Traffic Signal Optimization
- Traffic Signal pre-emption/ Priority
- Route Optimization

Each manuscript from 2-6 contributes towards solving one or more sub-problems identified above by answering research questions (RQs) associated with it. Problem identification from the existing knowledge gap was carried out by conducting a systematic literature review to answer **RQ1** and is outlined in manuscript 1. Figure. 8.1 illustrates a mapping among six manuscripts, five research questions and three sub-problems that we tried to solve in this thesis.

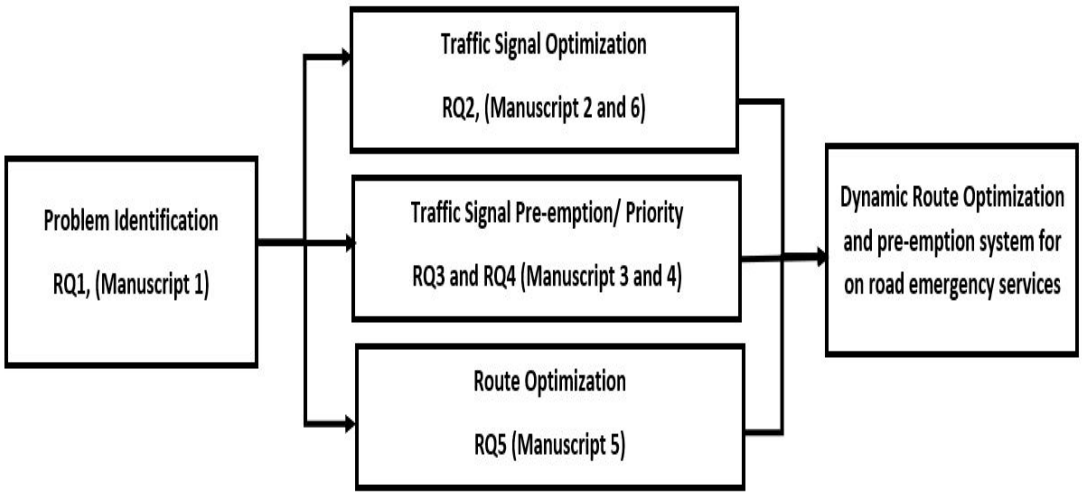

Figure 8.1: Illustration of mapping among manuscripts, RQs and sub-problems

Initially, we proposed an adaptive Fair Scheduling Algorithm (FSA) for VANETs that ascertains higher travel time reliability by minimizing travel time index (TTI) for

an isolated intersection in manuscript 2. We consider Buffer Index (BI), a typical travel time reliability measure, as the primary performance parameter to minimize. We achieve this through a novel approach of analogically mapping traffic control problems into real-time systems precisely mapping TTI with stretch (the factor by which a job is slowed down comparing with time it takes to process on a free system). This was initially conceptualized in manuscript 6 and further elaborated in manuscript 2. We first prove that stretch produced by FSA is less than or equal to twice the stretch produced by an optimal offline algorithm implying FSA is *2-competitive*. Then we empirically prove that FSA online traffic control signal algorithm is more reliable and fairer in scheduling in terms of travel time compared to existing state-of-art approaches. This solution answers **RQ2** and contributes towards solving traffic signal optimization sub-problem.

The thesis also presented emergency vehicle pre-emption (EVP) technique with different priorities in the arterial road network illustrated in manuscript 3. We leverage VANET to transmit critical information like current position, speed, the time EV takes to pass the intersection, and the route the EV follows in deciding when to trigger the EVP. We introduced different criticality levels for different levels of emergencies and assigned a certain level of success assurance in terms of target travel time for these criticalities. Unlike other studies, instead of implementing the EVP algorithm in a single intersection, we implemented it in an arterial traffic network. We ran exhaustive simulations. The results indicate that EVP algorithm can significantly reduce the average waiting time of normal traffic but still assures all EVs meet their target response time. This approach responds to **RQ3** and aims to solve traffic pre-emption sub-problem.

We extended the above idea to implement a decentralized self coordinating traffic system to prioritize emergency vehicle movement through an isolated traffic intersection as Virtual traffic lights plus emergency vehicles (VTL+EV) algorithm explained in manuscript 4. The algorithm eliminates the loss generated from dead periods in a traffic light cycle time and human-related factors like increased headway time and inconsistent

inter-vehicle spacing. We conducted comprehensive experiments and results showed that VTL+EV has the evident advantage of reduced waiting time for regular traffic as well as emergency vehicles (EVs). The approach of implementation of virtual traffic light addresses traffic pre-emption and traffic signal optimization sub-problems and respond to **RQ4**.

In parallel, we leverage the availability of huge GPS trajectory data-sets in proposing an origin-to-destination (O2D) travel time estimation model through manuscript 5. The ensemble-based model learns valuable information from manually engineered features extracted from raw GPS data and accurately estimates the O2D travel time. We concluded that for wisely extracted manual features, ensemble-based gradient boost regression approach can outperform existing state-of-art baseline models that employ deep neural networks. This approach of improving the accuracy of travel time estimation solves route optimization sub-problem by answering **RQ5**

Finally, we can conclude that all our research questions, **RQ1-RQ5**, were answered by six papers that we have developed.

8.3 Limitations

Traffic signal optimization realised through Fair Scheduling Algorithm in manuscript 2 focuses on achieving higher travel time reliability. The choice of optimization parameter largely depends on the nature of traffic each intersection experiences. Since we have limited our study to a single signalised intersection, the generalization of the algorithm to achieve the global optimum in terms of travel time reliability for all traffic intersections with varied nature of traffic distribution is harder. In such a case, each traffic intersection demand algorithm with appropriate objective function which is not a straightforward task as it demands different hardware resources, modelling variables and a different set of constraints. We assume the connectivity among the vehicles for sharing information

like current speed, position and time to pass the intersection. But, current adaptive traffic infrastructure is dependent on inductive loops. We can implement our models in existing systems that only use inductive loop detectors but need to compromise performance. We implemented the algorithm in a simulation environment with different levels of traffic flow. The results obtained from this model will vary slightly when implemented in real-world or a properly calibrated model.

Traffic signal pre-emption, achieved in manuscript 3 largely depends on the level of criticality assigned to EVs by emergency centre's operators. This study is limited to a sample arterial road network randomly chosen. The performance of such systems can vary when the system is implemented in a real-world traffic condition where achieving overall connectivity among the vehicles and roadside units are not feasible. In addition, virtual traffic lights proposed in manuscript 4 is implemented in an environment where all vehicles are connected with each other and can manoeuvre autonomous driving capability. The results obtained in such case are really promising but can deplete when implemented in a partially connected environment.

In O2D travel time estimation explained in manuscript 5, we have used New York taxi trip data-set (2.1 million trajectories) and Chengdu data-set collected by ride-sharing company DIDI (7.2 million trajectories). Though we have the availability of other larger data-sets, lack of computational resources limited our experiment to these two data-sets only. Even though we have extracted more than 72 features, there is still the possibility of extracting more features if data related to different traffic conditions are publicly available. Also, accuracy in calculating travel time difference from open street mapping engine largely vary with difference calculated from real-time mapping engines like google map. We were limited to use open street mapping engine as services from google map would be very expensive for the amount of data-set we have used.

8.4 Implications and Future Work

Dynamic route optimization and pre-emption system for on-road emergency services is a complex problem to solve. Choosing the best of both optimization and pre-emption cannot be the most efficient solution. In this thesis, we were able to design efficient traffic signal optimization that adepts with real-time traffic, effective pre-emption system that incurs less effect on normal traffic, and a reliable travel time estimation which can be used to find the fastest path with a higher level of assurance. Though manuscripts 2-6 make individual contributions towards traffic signal optimization, signal pre-emption and route optimization, a combination of all manuscripts as a whole contribute towards intelligent dynamic route optimization and road pre-emption system for on-road emergency services.

Several future research directions have been identified. The proposed FSA optimizes TTI as a performance parameter. FSA outperforms current state-of-art models in terms of travel time reliability. FSA is also better at resisting and flushing congestion for consistent and inconsistent traffic conditions. The algorithm can be further improved to perform better in all traffic conditions by introducing a multi-objective optimization algorithm that optimizes all three performance parameters (waiting time, throughput and reliability) of a traffic network. Also, we can develop a system that can learn to optimize multiple objective functions with different constraints to achieve a global optimum for an entire city.

In the case of EVP and VTL+EV algorithms, we consider that vehicles are connected. A further direction of research is to develop a model that can efficiently work for a mix of connected and non-connected vehicles. This can bring a futuristic model into a realistic one. Instead of picking a section of the road network, we can use map matching tools that automatically connect origin and destination of EVs with a section of the road including all the traffic lights which can be controlled for assigning priority.

Additionally, studying the computational efficiency of all proposed algorithms against baseline algorithms and the study of traffic recovery after the implementation of pre-emption explores new future research directions.

Another direction for future research is to incorporate multiple sensors data like floating car data, floating people data, data collected from loop detectors, automatic number plate recognition systems, automatic vehicle identification system and Bluetooth sensors for accurate calculation of travel time. Can these data improve the prediction accuracy of machine learning models? In future, the penetration of multiple connected sensors in the road network can be used to improve existing pre-emption and optimization models. It creates new opportunities to model the entire road network as a digital twin for visualizing every pulse of the road network in real-time. Also, we can use historical and current trajectory data to more precisely predict future travel times within a certain prediction horizon.

References

Abdi, H. & Williams, L. J. (2010). Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, *2*(4), 433–459.

Administration, N. H. T. S. (2008). National motor vehicle crash causation survey: Report to congress. *National Highway Traffic Safety Administration Technical Report DOT HS*, *811*, 059.

Afdhal, A. & Elizar, E. (2015). Enhanced route guidance and navigation for emergency vehicle using v2i-based cooperative communication. In *2015 international electronics symposium (ies)* (pp. 145–150).

Agarwal, A. & Paruchuri, P. (2016). V2v communication for analysis of lane level dynamics for better ev traversal. In *2016 ieee intelligent vehicles symposium (iv)* (pp. 368–375).

Albers, S. (2003). Online algorithms: a survey. *Mathematical Programming*, *97*(1-2), 3–26.

Alessandrini, A., Campagna, A., Delle Site, P., Filippi, F. & Persia, L. (2015). Automated vehicles and the rethinking of mobility and cities. *Transportation Research Procedia*, *5*, 145–160.

Ambulance quality indicators data 2019-20. (2019). NHS England. Retrieved from `https://www.england.nhs.uk/statistics/statistical-work-areas/ambulance-quality-indicators/ambulance-quality-indicators-data-2019-20`

Anand, J. & Flora, T. (2014). Emergency traffic management for ambulance using wireless communication. *IPASJ International Journal of Electronics & Communication (IIJEC)*, *2*(7), 43–52.

Annual report 2019. (2019). St.John. Retrieved from `https://www.stjohn.org.nz/News--Info/Our-Performance/Annual-Report`

Arellano, W. & Mahgoub, I. (2013). Trafficmodeler extensions: A case for rapid vanet simulation using, omnet++, sumo, and veins. In *2013 high capacity optical networks and emerging/enabling technologies* (pp. 109–115).

Asaduzzaman, M. & Vidyasankar, K. (2017). A priority algorithm to control the traffic signal for emergency vehicles. In *2017 ieee 86th vehicular technology conference (vtc-fall)* (pp. 1–7).

Asghari, M., Deng, D., Shahabi, C., Demiryurek, U. & Li, Y. (2016). Price-aware real-time ride-sharing at scale: an auction-based approach. In *Proceedings of the 24th acm sigspatial international conference on advances in geographic*

information systems (pp. 1–10).

Avin, C., Borokhovich, M., Haddad, Y. & Lotker, Z. (2012). Optimal virtual traffic light placement. In *Proceedings of the 8th international workshop on foundations of mobile computing* (pp. 1–10).

Bachelder, A. D. (2011, January 11). *Cellular-based preemption system.* Google Patents. (US Patent 7,868,783)

Balke, K. N., Charara, H. A. & Parker, R. (2005). *Development of a traffic signal performance measurement system (TSPMS)* (Tech. Rep.). Texas, USA: Texas Transportation Institute, Texas A & M University System College ....

Barrachina, J., Garrido, P., Fogue, M., Martinez, F. J., Cano, J.-C., Calafate, C. T. & Manzoni, P. (2014). Reducing emergency services arrival time by using vehicular communications and evolution strategies. *Expert Systems with Applications*, *41*(4), 1206–1217.

Barthwal, S. & Menghani, P. (2017). An advance system for emergency vehicles: Based on m2m communication. In *2017 11th international conference on intelligent systems and control (isco)* (pp. 374–378).

Bazzi, A., Zanella, A. & Masini, B. M. (2016). A distributed virtual traffic light algorithm exploiting short range v2v communications. *Ad Hoc Networks*, *49*, 42–57.

Bazzi, A., Zanella, A., Masini, B. M. & Pasolini, G. (2014). A distributed algorithm for virtual traffic lights with ieee 802.11 p. In *2014 european conference on networks and communications (eucnc)* (pp. 1–5).

Bergenhem, C., Hedin, E. & Skarin, D. (2012). Vehicle-to-vehicle communication for a platooning system. *Procedia-Social and Behavioral Sciences*, *48*, 1222–1233.

Bhavani, B., Vishwasri, Y. & Chandrakala, B. (2016). Design and implementation of intelligent smart traffic control system for emergency ambulance clearance and stolen vehicle detection. *International Journal of Research*, *5*(4), 265–270.

Bieker-Walz, L. & Behrisch, M. (2019). Modelling green waves for emergency vehicles using connected traffic data. *EPiC Series in Computing*(62), 1–11.

Brady, D. & Park, B. B. (2016). Improving emergency vehicle routing with additional road features. In *Vehits* (pp. 187–194).

Bu, F. & Fang, H. (2010). Shortest path algorithm within dynamic restricted searching area in city emergency rescue. In *2010 ieee international conference on emergency management and management sciences* (pp. 371–374).

Bura, W. & Boryczka, M. (2010). Ant colony system in ambulance navigation. *Journal of Medical Informatics & Technologies*, *15*.

Burns, A. & Davis, R. (2013). Mixed criticality systems-a review. *Department of Computer Science, University of York, Tech. Rep*, 1–69.

Cai, C., Wang, Y. & Geers, G. (2013). Vehicle-to-infrastructure communication-based adaptive traffic signal control. *IET Intelligent Transport Systems*, *7*(3), 351–360.

Chada, S. & Newland, R. (2002). *Effectiveness of bus signal priority* (Tech. Rep.). Washington, UAS: National Center for Transit Research (US).

Chakraborty, P., Tiwari, A. & Sinha, P. (2015). Adaptive and optimized emergency

vehicle dispatching algorithm for intelligent traffic management system. procedia comput sci 57: 1384–1393. *doi. org/10.1016/j. procs*, *454*.

Chang, H.-J. & Park, G.-T. (2013). A study on traffic signal control at signalized intersections in vehicular ad hoc networks. *Ad Hoc Networks*, *11*(7), 2115–2124.

Chen, C.-Y., Chen, P.-Y. & Chen, W.-T. (2013). A novel emergency vehicle dispatching system. In *2013 ieee 77th vehicular technology conference (vtc spring)* (pp. 1–5).

Chen, T. & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794).

Chen, Y.-y., Lv, Y., Li, Z. & Wang, F.-Y. (2016). Long short-term memory model for traffic congestion prediction with online open data. In *2016 ieee 19th international conference on intelligent transportation systems (itsc)* (pp. 132–137).

Chen, Y.-z., Shen, S.-f., Chen, T. & Yang, R. (2014). Path optimization study for vehicles evacuation based on dijkstra algorithm. *Procedia Engineering*, *71*, 159–165.

Cheng, J., Li, G. & Chen, X. (2018). Research on travel time prediction model of freeway based on gradient boosting decision tree. *IEEE Access*, *7*, 7466–7480.

Cheng, J., Wu, W., Cao, J. & Li, K. (2016). Fuzzy group-based intersection control via vehicular networks for smart transportations. *IEEE Transactions on Industrial Informatics*, *13*(2), 751–758.

Choi, E.-H. (2010). Crash factors in intersection-related crashes: An on-scene perspective. *U.D of Transportation*.

Choosumrong, S., Raghavan, V. & Bozon, N. (2012). Multi-criteria emergency route planning based on analytical hierarchy process and pgrouting. *Geoinformatics*, *23*(4), 159–167.

Chou, L.-D., Deng, B.-T., Li, D. C. & Kuo, K.-W. (2012). A passenger-based adaptive traffic signal control mechanism in Intelligent Transportation Systems. In *2012 12th international conference on its telecommunications* (pp. 408–411).

Chowdhury, A. (2016). Priority based and secured traffic management system for emergency vehicle using iot. In *2016 international conference on engineering & mis (icemis)* (pp. 1–6).

Chuxing, D. (2019). Gaia open dataset. *Internet Link: https://outreach. didichuxing. com/research/opendata/en/(Accessed on August 18, 2019)*.

Clark, S. & Watling, D. (2005). Modelling network travel time reliability under stochastic demand. *Transportation Research Part B: Methodological*, *39*(2), 119–140.

Cooke, K. L. & Halsey, E. (1966). The shortest route through a network with time-dependent internodal transit times. *Journal of mathematical analysis and applications*, *14*(3), 493–498.

Das, S., Kalava, R. N., Kumar, K. K., Kandregula, A., Suhaas, K., Bhattacharya, S. & Ganguly, N. (2019). Map enhanced route travel time prediction using deep neural networks. *arXiv preprint arXiv:1911.02623*.

Derekenaris, G., Garofalakis, J., Makris, C., Prentzas, J., Sioutas, S. & Tsakalidis, A. (2001). Integrating gis, gps and gsm technologies for the effective management

of ambulances. *Computers, Environment and Urban Systems*, *25*(3), 267–278.

Djahel, S., Smith, N., Wang, S. & Murphy, J. (2015). Reducing emergency services response time in smart cities: An advanced adaptive and fuzzy approach. In *2015 ieee first international smart cities conference (isc2)* (pp. 1–8).

Duan, Y., Yisheng, L. & Wang, F.-Y. (2016). Travel time prediction with lstm neural network. In *2016 ieee 19th international conference on intelligent transportation systems (itsc)* (pp. 1053–1058).

Eksioglu, B., Vural, A. V. & Reisman, A. (2009). The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, *57*(4), 1472–1483.

Elalouf, A. (2012). Efficient routing of emergency vehicles under uncertain urban traffic conditions. *Scientific Research Publishing*.

Elith, J., Leathwick, J. R. & Hastie, T. (2008). A working guide to boosted regression trees. *Journal of Animal Ecology*, *77*(4), 802–813.

Elmandili, H., Toulni, H. & Nsiri, B. (2013). Optimizing road traffic of emergency vehicles. In *2013 international conference on advanced logistics and transport* (pp. 59–62).

Eltayeb, A. S., Almubarak, H. O. & Attia, T. A. (2013). A gps based traffic light pre-emption control system for emergency vehicles. In *2013 international conference on computing, electrical and electronic engineering (icceee)* (pp. 724–729).

Emtrac signal-priority system: Transit signal priority (tsp) and emergency vehicle preemption (evp). (n.d.). Retrieved from `https://www.emtracsystems.com/`

England, N. (2015). Statistical note: Ambulance quality indicators (aqi) december 15.

Ernst, R. & Di Natale, M. (2016). Mixed criticality systems—a history of misconceptions? *IEEE Design & Test*, *33*(5), 65–74.

Even, G., Halldórsson, M. M., Kaplan, L. & Ron, D. (2009). Scheduling with conflicts: online and offline algorithms. *Journal of Scheduling*, *12*(2), 199–224.

Feng, Y., Head, K. L., Khoshmagham, S. & Zamanipour, M. (2015). A real-time adaptive signal control in a connected vehicle environment. *Transportation Research Part C: Emerging Technologies*, *55*, 460–473.

Fernandes, P. & Nunes, U. (2012a). Platooning with ivc-enabled autonomous vehicles: Strategies to mitigate communication delays, improve safety and traffic flow. *IEEE Transactions on Intelligent Transportation Systems*, *13*(1), 91–106.

Fernandes, P. & Nunes, U. (2012b). Platooning with ivc-enabled autonomous vehicles: Strategies to mitigate communication delays, improve safety and traffic flow. *IEEE Transactions on Intelligent Transportation Systems*, *13*(1), 91–106.

Ferreira, M., Fernandes, R., Conceição, H., Viriyasitavat, W. & Tonguz, O. K. (2010). Self-organized traffic control. In *Proceedings of the seventh acm international workshop on vehicular internetworking* (pp. 85–90).

Ferreira, M. C. P., Tonguz, O., Fernandes, R. J., DaConceicao, H. M. F. & Viriyasitavat, W. (2015, March 3). *Methods and systems for coordinating vehicular traffic using in-vehicle virtual traffic control signals enabled by vehicle-to-vehicle communications.* Google Patents. (US Patent 8,972,159)

Fitch, J. (2005a). Response times: myths, measurement & management. *Journal of Emergency Medical Services*.

Fitch, J. (2005b). Response times: myths, measurement & management. *JEMS: a journal of emergency medical services*, *30*(9), 47.

Fleischman, R. J., Lundquist, M., Jui, J., Newgard, C. D. & Warden, C. (2013). Predicting ambulance time of arrival to the emergency department using global positioning system and google maps. *Prehospital Emergency Care*, *17*(4), 458–465.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189–1232.

Gedawy, H. K. (2009). Dynamic path planning and traffic light coordination for emergency vehicle routing..

Gedawy, H. K., Dias, M. & Harras, K. (2008). Dynamic path planning and traffic light coordination for emergency vehicle routing. *Comp. Sci. Dept., Carnegie Mellon Univ., Pittsburgh, Pennsylvania, USA*.

Glasmeier, A. & Christopherson, S. (2015). *Thinking about smart cities.* Oxford University Press UK.

Goel, A., Ray, S. & Chandra, N. (2012). Intelligent traffic light system to prioritized emergency purpose vehicles based on wireless sensor network. *International Journal of Computer Applications*, *40*(12), 36–39.

Goel, A. K. & Bhatta, S. R. (2004). Use of design patterns in analogy-based design. *Advanced Engineering Informatics*, *18*(2), 85–94.

Gradinescu, V., Gorgorin, C., Diaconescu, R., Cristea, V. & Iftode, L. (2007). Adaptive traffic lights using car-to-car communication. In *2007 ieee 65th vehicular technology conference-vtc2007-spring* (pp. 21–25).

Greenblatt, J. B. & Shaheen, S. (2015). Automated vehicles, on-demand mobility, and environmental impacts. *Current sustainable/renewable energy reports*, *2*(3), 74–81.

Greenwald, M. & Khanna, S. (2001). Space-efficient online computation of quantile summaries. *ACM SIGMOD Record*, *30*(2), 58–66.

Grilo, A. & Nunes, M. (2002). Performance evaluation of IEEE 802.11 e. In *The 13th ieee international symposium on personal, indoor and mobile radio communications* (Vol. 1, pp. 511–517).

Hadas, Y. & Ceder, A. (1996). Shortest path of emergency vehicles under uncertain urban traffic conditions. *Transportation research record*, *1560*(1), 34–39.

Haklay, M. & Weber, P. (2008). Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, *7*(4), 12–18.

Harchol-Balter, M., Bansal, N. & Schroeder, B. (2000). *Implementation of SRPT scheduling in web servers* (Tech. Rep.). CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE.

Hegde, R., Sali, R. R. & Indira, M. (2013). Rfid and gps based automatic lane clearance system for ambulance. *Int. J. Adv. Elect. Electron. Eng*, *2*(3), 102–107.

Huang, W., Yang, X. & Ma, W. (2011). Signal priority control for emergency vehicle operation. In *2nd international conference on models* (pp. 22–24).

Huang, Y.-S., Shiue, J.-Y. & Luo, J. (2015). A traffic signal control policy for emergency vehicles preemption using timed petri nets. *IFAC-PapersOnLine*, *48*(3), 2183–2188.

Humagain, S., Sinha, R., Lai, E. & Ranjitkar, P. (2020). A systematic review of route optimisation and pre-emption methods for emergency vehicles. *Transport reviews*, *40*(1), 35–53.

Hunter, T., Herring, R., Abbeel, P. & Bayen, A. (2009). Path and travel time inference from gps probe vehicle data. *NIPS Analyzing Networks and Learning with Graphs*, *12*(1), 2.

Idris, M. Y. I., Sivalingam, M., Tamil, E. M., Razak, Z. & Noor, N. M. (2013). Emergency vehicle preemption system based on global positioning system (gps), a-star (a*) algorithm and fpga. *Computer systems science and engineering*, *28*(3), 147–156.

INRIX, W. (2017). *Inrix global traffic scorecard.*

Irani, S. & Leung, V. (1996). Scheduling with conflicts, and applications to traffic signal control. In *Soda* (Vol. 96, pp. 85–94).

Irani, S. & Leung, V. (2003). Scheduling with conflicts on bipartite and interval graphs. *Journal of Scheduling*, *6*(3), 287–307.

Iyyappan, M. S. & Nandagopal, M. V. (2013). Automatic accident detection and ambulance rescue with intelligent traffic light system. *International journal of advanced research in electrical, electronics and instrumentation engineering*, *2*(4), 1319.

Jain, R. K., Chiu, D.-M. W. & Hawe, W. R. (1984). A quantitative measure of fairness and discrimination. *Eastern Research Laboratory, Digital Equipment Corporation, Hudson, MA*.

Jayaraj, V. & Hemanath, c. (2015). Emergency vehicle signalling using vanets. In *2015 ieee 17th international conference on high performance computing and communications, 2015 ieee 7th international symposium on cyberspace safety and security, and 2015 ieee 12th international conference on embedded software and systems* (pp. 734–739).

Jindal, I., Chen, X., Nokleby, M. & Ye, J. (2017). A unified neural network approach for estimating travel time and distance for a taxi trip. *arXiv preprint arXiv:1710.04350*.

Jones, G. V., Judge, K., Beck, J. C. & Keegan, R. (1999, November 16). *Automatic determination of traffic signal preemption using gps, apparatus and method.* Google Patents. (US Patent 5,986,575)

Jordan, C. A. & Cetin, M. (2015). *Signal preemption strategy for emergency vehicles using vehicle to infrastructure communication* (Tech. Rep.). Washington, USA: TRB.

Kaggle. (2015). `https://www.kaggle.com/competitions`. (Accessed: 2020-12-30)

Kai, N., Yao-ting, Z. & Yue-peng, M. (2014). Shortest path analysis based on dijkstra's algorithm in emergency response system. *Indonesian Journal of Electrical Engineering and Computer Science*, *12*(5), 3476–3482.

Kamalanathsharma, R. K. & Hancock, K. L. (2012). Intelligent preemption control for emergency vehicles in urban corridors. In *Transportation research board 91st annual meeting.*

Kang, W., Xiong, G., Lv, Y., Dong, X., Zhu, F. & Kong, Q. (2014). Traffic signal coordination for emergency vehicles. In *17th international ieee conference on intelligent transportation systems (itsc)* (pp. 157–161).

Kari, D., Wu, G. & Barth, M. J. (2014). Development of an agent-based online adaptive signal control strategy using connected vehicle technology. In *17th international ieee conference on intelligent transportation systems (itsc)* (pp. 1802–1807).

Karp, R. M. (1972). Reducibility among combinatorial problems. In *Complexity of computer computations* (pp. 85–103). Springer.

Kdd cup. (2015). `https://www.kdd.org/kdd-cup/view/kdd-cup-2016`. (Accessed: 2020-12-30)

Khalid, S., Khalil, T. & Nasreen, S. (2014). A survey of feature selection and feature extraction techniques in machine learning. In *2014 science and information conference* (pp. 372–378).

Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, *33*(2004), 1–26.

Kodire, V., Bhaskaran, S. & Vishwas, H. (2016). Gps and zigbee based traffic signal preemption. In *2016 international conference on inventive computation technologies (icict)* (Vol. 2, pp. 1–5).

Kokuti, A., Hussein, A., Marín-Plaza, P., de la Escalera, A. & García, F. (2017). V2x communications architecture for off-road autonomous vehicles. In *2017 ieee international conference on vehicular electronics and safety (icves)* (pp. 69–74).

Koren, Y. (2009). The bellkor solution to the netflix grand prize. *Netflix prize documentation*, *81*(2009), 1–10.

Kotani, J., Yamazaki, K. & Jinno, M. (2011). Expanding fast emergency vehicle preemption system in tokyo. In *18th its world congress.*

Krajzewicz, D., Erdmann, J., Behrisch, M. & Bieker, L. (2012). Recent development and applications of sumo-simulation of urban mobility. *International journal on advances in systems and measurements*, *5*(3&4).

Krajzewicz, D., Hertkorn, G., Rössel, C. & Wagner, P. (2002). SUMO (Simulation of Urban MObility)-an open-source traffic simulation. In *Proceedings of the 4th middle east symposium on simulation and modelling (mesm20002)* (pp. 183–187).

Krasniqi, X. & Hajrizi, E. (2016). Use of iot technology to drive the automotive industry from connected to full autonomous vehicles. *IFAC-PapersOnLine*, *49*(29), 269–274.

Kula, U., Tozanli, O. & Tarakcio, S. (2012). Emergency vehicle routing in disaster response operations. In *Poms 23rd annual conference, chicago* (Vol. 28, p. 16).

Kwon, T. M. & Kim, S. (2003). Development of dynamic route clearance strategies for emergency vehicle operations, phase i. *Center for Transportation Studies*.

LeCun, Y. (2018). The power and limits of deep learning: In his iri medal address, yann lecun maps the development of machine learning techniques and suggests

what the future may hold. *Research-Technology Management*, *61*(6), 22–27.

Li, F. & Wang, Y. (2007). Routing in vehicular ad hoc networks: A survey. *IEEE Vehicular technology magazine*, *2*(2), 12–22.

Li, Y., Deng, D., Demiryurek, U., Shahabi, C. & Ravada, S. (2015). Towards fast and accurate solutions to vehicle routing in a large-scale and dynamic environment. In *International symposium on spatial and temporal databases* (pp. 119–136).

Li, Y., Fu, K., Wang, Z., Shahabi, C., Ye, J. & Liu, Y. (2018). Multi-task representation learning for travel time estimation. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining* (pp. 1695–1704).

Lin, X., Wang, Y., Xiao, X., Li, Z. & Bhowmick, S. S. (2019). Path travel time estimation using attribute-related hybrid trajectories network. In *Proceedings of the 28th acm international conference on information and knowledge management* (pp. 1973–1982).

Luxen, D. & Vetter, C. (2011). Real-time routing with openstreetmap data. In *Proceedings of the 19th acm sigspatial international conference on advances in geographic information systems* (pp. 513–516).

Ma, X., Ding, C., Luan, S., Wang, Y. & Wang, Y. (2017). Prioritizing influential factors for freeway incident clearance time prediction using the gradient boosting decision trees method. *IEEE Transactions on Intelligent Transportation Systems*, *18*(9), 2303–2310.

Ma, Z., Huang, D., Li, C. & Guo, J. (2020). Travel time reliability-based signal timing optimization for urban road traffic network control. *Mathematical Problems in Engineering*, *2020*.

Mali, V., Rao, M. & Mantha, S. (2012). Enhanced routing in disaster management based on gis. In *International conference on intuitive systems & solutions* (pp. 5–6).

Malikopoulos, A. A., Cassandras, C. G. & Zhang, Y. J. (2018). A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections. *Automatica*, *93*, 244–256.

Mankins, J. C. (1995). Technology readiness levels. *White Paper, April*, *6*(1995), 1995.

Mannion, P., Duggan, J. & Howley, E. (2016). An experimental review of reinforcement learning algorithms for adaptive traffic signal control. In *Autonomic road transport support systems* (pp. 47–66). Springer.

Martins, V., Rufino, J., Silva, L., Almeida, J., Miguel Fernandes Silva, B., Ferreira, J. & Fonseca, J. (2019). Towards personal virtual traffic lights. *Information*, *10*(1), 32.

Maslekar, N., Boussedjra, M., Mouzna, J. & Labiod, H. (2011). VANET based adaptive traffic signal control. In *2011 ieee 73rd vehicular technology conference (vtc spring)* (pp. 1–5).

Massey Jr, F. J. (1951). The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association*, *46*(253), 68–78.

McAllister, R., Gal, Y., Kendall, A., Van Der Wilk, M., Shah, A., Cipolla, R. & Weller, A. V. (2017). Concrete problems for autonomous vehicle safety: advantages of

bayesian deep learning..

Mirchandani, P. & Lucas, D. (2004). *Integrated transit priority and rail/emergency preemption in real-time traffic adaptive signal control. intell transp syst 8: 101-115.*

Moemi, T. J., Isong, B. & Jonathan, B. (2017). Hfinder: Anti-emergency medical service late response time system. In *2017 international conference on health informatics and medical systems* (p. 15-20).

Moroi, Y. & Takami, K. (2015). A method of securing priority-use routes for emergency vehicles using inter-vehicle and vehicle-road communication. In *2015 7th international conference on new technologies, mobility and security (ntms)* (pp. 1–5).

Münst, W., Dannheim, C., Mäder, M., Gay, N., Malnar, B., Al-Mamun, M. & Icking, C. (2015). Virtual traffic lights: Managing intersections in the cloud. In *2015 7th international workshop on reliable networks design and modeling (rndm)* (pp. 329–334).

Musolino, G., Polimeni, A., Rindone, C. & Vitetta, A. (2013a). Travel time forecasting and dynamic routes design for emergency vehicles. *Procedia-Social and Behavioral Sciences*, *87*, 193–202.

Musolino, G., Polimeni, A., Rindone, C. & Vitetta, A. (2013b). Travel time forecasting and dynamic routes design for emergency vehicles. *Elsevier,Procedia- Social and Behavioral Sciences*, *87*, 193-202.

Muthukrishnan, S., Rajaraman, R., Shaheen, A. & Gehrke, J. E. (1999). Online scheduling to minimize average stretch. In *40th annual symposium on foundations of computer science (cat. no. 99cb37039)* (pp. 433–443).

Nafi, N. S. & Khan, J. Y. (2012). A VANET based intelligent road traffic signalling system. In *Australasian telecommunication networks and applications conference (atnac) 2012* (pp. 1–6).

Nellore, K. & Hancke, G. P. (2016). Traffic management for emergency vehicle priority based on visual sensing. *Sensors*, *16*(11), 1892.

Newby, J. (2015). The future of autonomous emergency response. *The Ride Ahead*.

NHTSA. (April 2014). The national highway traffic safety administration and ground ambulance crashes.

Nicoara, P.-S. & Haidu, I. (2014). A gis based network analysis for the identification of shortest route access to emergency medical facilities. *Geographia Technica*, *9*(2/2014), 60-67.

Noori, H., Fu, L. & Shiravi, S. (2016). A connected vehicle based traffic signal control strategy for emergency vehicle preemption. In *Transportation research board 95th annual meeting.*

Nordin, N. A. M., Zaharudin, Z. A., Maasar, M. A. & Nordin, N. A. (2012). Finding shortest path of the ambulance routing: Interface of a^* algorithm using c# programming. In *2012 ieee symposium on humanities, science and engineering research* (p. 1569-1573).

Nsw state emergency service annual report 2015. (2015). NSW state emergency service. Retrieved from `https://www.ses.nsw.gov.au/media/1225/`

nswses_annual_report_2015_16_8mb.pdf

Olaverri-Monreal, C., Gomes, P., Silvéria, M. K. & Ferreira, M. (2012). In-vehicle virtual traffic lights: A graphical user interface. In *7th iberian conference on information systems and technologies (cisti 2012)* (pp. 1–6).

Pan, B., Demiryurek, U. & Shahabi, C. (2012). Utilizing real-world transportation data for accurate traffic prediction. In *2012 ieee 12th international conference on data mining* (pp. 595–604).

Panahi, S. & Delavar, M. (2009). Dynamic shortest path in ambulance routing based on gis. *International journal of Geoinformatics*, *5*(1).

Pandit, K., Ghosal, D., Zhang, H. M. & Chuah, C.-N. (2013). Adaptive traffic signal control with vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, *62*(4), 1459–1471.

Paniati, J. F. & Amoni, M. (2006). Traffic signal preemption for emergency vehicle: a cross-cutting study. *US Federal Highway Administration*.

Perozzi, B., Al-Rfou, R. & Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th acm sigkdd international conference on knowledge discovery and data mining* (pp. 701–710).

Pighin, M. & Fierens, P. I. (2015). Vanet for emergency vehicles: Preliminary results. In *2015 xvi workshop on information processing and control (rpic)* (pp. 1–4).

Pillac, V. (2012). *Dynamic vehicle routing: solution methods and computational tools* (Unpublished doctoral dissertation). Ecole des Mines de Nantes.

Piro, G., Cianci, I., Grieco, L. A., Boggia, G. & Camarda, P. (2014). Information centric services in smart cities. *Journal of Systems and Software*, *88*, 169–188.

Polineni, J. N., Ravi Kumar, K. & Ravi Kumar, P. (2015). Traffic free route system using gsm and gps for ambulance. *International Journal of Advanced Technology and Innovative Research*, *7*(18), 3593–3598.

Pons, P. T. & Markovchick, V. J. (2002a). Eight minutes or less: does the ambulance response time guideline impact trauma patient outcome? *The Journal of emergency medicine*, *23*(1), 43–48.

Pons, P. T. & Markovchick, V. J. (2002b). Eight minutes or less: Does the ambulance response time guideline impact trauma patient outcome? *Journal of Emergency Medicine*, *23*, 43-48.

Priemer, C. & Friedrich, B. (2009). A decentralized adaptive traffic signal control using V2I communication data. In *2009 12th international ieee conference on intelligent transportation systems* (pp. 1–6).

Qin, X. & Khan, A. M. (2012). Control strategies of traffic signal timing transition for emergency vehicle preemption. *Transportation research part C: emerging technologies*, *25*, 1–17.

Rahmani, M., Jenelius, E. & Koutsopoulos, H. N. (2013). Route travel time estimation using low-frequency floating car data. In *16th international ieee conference on intelligent transportation systems (itsc 2013)* (pp. 2292–2297).

Ramamritham, K., Stankovic, J. A. & Shiah, P.-F. (1990). Efficient scheduling algorithms for real-time multiprocessor systems. *IEEE Transactions on Parallel and Distributed systems*, *1*(2), 184–194.

Ranjan, P. & Ahirwar, K. K. (2011). Comparative study of vanet and manet routing protocols. In *Proc. of the international conference on advanced computing and communication technologies (acct 2011)* (pp. 517–523).

RAPIDSOS. (2015). Outcomes quantifying the impact of emergency response times.

RapidSOS. (2015). *Quantifying the impact of emergency response times.* RAPIDSOS. Retrieved from `www.RapidSOS.com`

Roess, R. P., Prassas, E. S. & McShane, W. R. (2004). *Traffic engineering*. Pearson/Prentice Hall.

Salehinejad, H., Pouladi, F. & Talebi, S. (2011). Intelligent navigation of emergency vehicles. In *2011 developments in e-systems engineering* (pp. 318–323).

Service, N. S. E. (2016). *Nsw state emergency service annual report 2015* (Tech. Rep.).

Shekar, B. S., Kumar, G., Rani, H. U., Divyashreee, C., George, G. & Murali, A. (2012). Gps based shortest path for ambulances using vanets. In *Proc. international conference on wireless networks (icwn 2012)* (Vol. 49).

Shi, J., Peng, C., Zhu, Q., Duan, P., Bao, Y. & Xie, M. (2015). There is a will, there is a way: A new mechanism for traffic control based on vtl and vanet. In *2015 ieee 16th international symposium on high assurance systems engineering* (pp. 240–246).

Shirani, R., Hendessi, F., Montazeri, M. A. & Zefreh, M. S. (2008). Absolute priority for a vehicle in vanet. In *Computer society of iran computer conference* (pp. 955–959).

Sinha, R., Roop, P. S. & Ranjitkar, P. (2013). Virtual Traffic Lights+ A Robust, Practical, and Functionally Safe Intelligent Transportation System. *Transportation Research Record*, *2381*(1), 73–80.

Siripanpornchana, C., Panichpapiboon, S. & Chaovalit, P. (2016). Travel-time prediction with deep learning. In *2016 ieee region 10 conference (tencon)* (pp. 1859–1862).

Sommer, C. & Dressler, F. (2014). *Vehicular networking*. Cambridge University Press. doi: 10.1017/CBO9781107110649

Song, R., Sun, W., Zheng, B. & Zheng, Y. (2014). Press: A novel framework of trajectory compression in road networks. *arXiv preprint arXiv:1402.1546*.

St.John's. (2016). Annual report 2016 health and wellbeing.

Sun, H. L., Yue, L. Y. & Yao, S. Y. (2014). Study on path selection of emergency rescue based on gis. In *Advanced materials research* (Vol. 864, pp. 2804–2807).

Taeihagh, A. & Lim, H. S. M. (2019). Governing autonomous vehicles: emerging responses for safety, liability, privacy, cybersecurity, and industry risks. *Transport reviews*, *39*(1), 103–128.

Tang, J., Zou, Y., Ash, J., Zhang, S., Liu, F. & Wang, Y. (2016). Travel time estimation using freeway point detector data based on evolving fuzzy neural inference system. *PloS one*, *11*(2), e0147263.

Technologies, G. T. (2016). Emergency vehicle preemption solutions. *Global Traffic Technologies Report*.

Technology, I. (2014). Smart cities to rise fourfold in number from 2013 to 2025. *Smart Cities: Business Models, Technologies and Existing Projects*.

Togneri, M. C. & Deriaz, M. (2013). On-board navigation system for smartphones. In

International conference on indoor positioning and indoor navigation (Vol. 28, p. 31).

Traffic, T. (2020). *Tomtom traffic index: Live congestion statistics.*

Tran, N., Breene, J., Khayesi, M., McInerney, R., Sukhai, A., Toroyan, T. & Ward, D. (2018). *Global status report on road safety 2018*. World Health Organization. Retrieved from `https://www.who.int/publications-detail/global-status-report-on-road-safety-2018`

Unibaso, G., Del Ser, J., Gil-Lopez, S. & Molinete, B. (2010). A novel cam-based traffic light preemption algorithm for efficient guidance of emergency vehicles. In *13th international ieee conference on intelligent transportation systems* (pp. 74–79).

ur Rehman, S., Khan, M. A., Zia, T. A. & Zheng, L. (2013). Vehicular ad-hoc networks (VANETs)-an overview and challenges. *Journal of Wireless Networking and Communications*, *3*(3), 29–38.

Van Gulik, J. & Vlacic, L. (2002). Intersection priority system [roads]. In *Proceedings. the ieee 5th international conference on intelligent transportation systems* (pp. 572–575).

Varga, A. & Hornig, R. (2008). An overview of the OMNeT++ simulation environment. In *Proceedings of the 1st international conference on simulation tools and techniques for communications, networks and systems & workshops* (p. 60).

Verhaegen, P.-A., D'hondt, J., Vandevenne, D., Dewulf, S. & Duflou, J. R. (2011). Identifying candidates for design-by-analogy. *Computers in Industry*, *62*(4), 446–459.

Viriyasitavat, W. & Tonguz, O. K. (2012). Priority management of emergency vehicles at intersections using self-organized traffic control. In *2012 ieee vehicular technology conference (vtc fall)* (pp. 1–4).

Vlad, R., Morel, C., Morel, J.-Y. & Vlad, S. (2008). A learning real-time routing system for emergency vehicles. In *2008 ieee international conference on automation, quality and testing, robotics* (Vol. 3, pp. 390–395).

Voelcker, J. (2020). 1.2 billion vehicles on world's roads now, 2 billion by 2035: Report. *Green car reports*, *7*(29).

Wang, D., Zhang, J., Cao, W., Li, J. & Zheng, Y. (2018). When will you arrive? estimating travel time based on deep neural networks. In *Thirty-second aaai conference on artificial intelligence.*

Wang, H., Tang, X., Kuo, Y.-H., Kifer, D. & Li, Z. (2019). A simple baseline for travel time estimation using large-scale trip data. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *10*(2), 1–22.

Wang, X. & Liu, J. (2011). Design and implementation for ambulance route search based on the internet of things. In *2011 third international conference on communications and mobile computing* (pp. 523–526).

Wang, Y., Wu, Z., Yang, X. & Huang, L. (2013). Design and implementation of an emergency vehicle signal preemption system based on cooperative vehicle-infrastructure technology. *Advances in Mechanical Engineering*, *5*, 834976.

Wang, Y., Zheng, Y. & Xue, Y. (2014). Travel time estimation of a path using sparse

trajectories. In *Proceedings of the 20th acm sigkdd international conference on knowledge discovery and data mining* (pp. 25–34).

Wang, Z., Fu, K. & Ye, J. (2018). Learning to estimate the travel time. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining* (pp. 858–866).

Weng, Y.-S., Huang, Y.-S., Su, S.-F. & Yu, C.-S. (2011). Modelling of emergency vehicle preemption systems using statecharts. In *2011 ieee international conference on systems, man, and cybernetics* (pp. 556–561).

Winn, M. T. (2014). *A road network shortest path analysis: Applying time-varying travel-time costs for emergency response vehicle routing, davis county, utah: a thesis presented to the department of humanities and social sciences in candidacy for the degree of master of science* (Unpublished doctoral dissertation). Northwest Missouri State University.

Winter, S. (2002). Modeling costs of turns in route planning. *GeoInformatica*, *6*(4), 345–361.

Wu, J., Ghosal, D., Zhang, M. & Chuah, C.-N. (2017). Delay-based traffic signal control for throughput optimality and fairness at an isolated intersection. *IEEE Transactions on Vehicular Technology*, *67*(2), 896–909.

Wu, S. K. (2009). *Adaptive traffic control effect on arterial travel time charateristics* (Unpublished doctoral dissertation). Georgia Institute of Technology.

Xie, H., Karunasekera, S., Kulik, L., Tanin, E., Zhang, R. & Ramamohanarao, K. (2017). A simulation study of emergency vehicle prioritization in intelligent transportation systems. In *2017 ieee 85th vehicular technology conference (vtc spring)* (pp. 1–5).

Xu, J., Zhang, Y., Chao, L. & Xing, C. (2019). Stdr: A deep learning method for travel time estimation. In *International conference on database systems for advanced applications* (pp. 156–172).

Yao, H., Wu, F., Ke, J., Tang, X., Jia, Y., Lu, S., … Li, Z. (2018). Deep multi-view spatial-temporal network for taxi demand prediction. In *Thirty-second aaai conference on artificial intelligence.*

Yaqoob, I., Hashem, I. A. T., Mehmood, Y., Gani, A., Mokhtar, S. & Guizani, S. (2017). Enabling communication technologies for smart cities. *IEEE Communications Magazine*, *55*(1), 112–120.

Yin, Y. (2008). Robust optimal traffic signal timing. *Transportation Research Part B: Methodological*, *42*(10), 911–924.

Yoo, J. B., Kim, J. & Park, C. Y. (2010). Road reservation for fast and safe emergency vehicle response using ubiquitous sensor network. In *2010 ieee international conference on sensor networks, ubiquitous, and trustworthy computing* (pp. 353–358).

Younes, M. B. & Boukerche, A. (2015). Intelligent traffic light controlling algorithms using vehicular networks. *IEEE transactions on vehicular technology*, *65*(8), 5887–5899.

Younes, M. B. & Boukerche, A. (2018). An efficient dynamic traffic light scheduling algorithm considering emergency vehicles for intelligent transportation systems.

Wireless Networks, *24*(7), 2451–2463.

Yuan, H., Li, G., Bao, Z. & Feng, L. (2020). Effective travel time estimation: When historical trajectories over road networks matter. In *Proceedings of the 2020 acm sigmod international conference on management of data* (pp. 2135–2149).

Yuan, N. J., Zheng, Y., Zhang, L. & Xie, X. (2012). T-finder: A recommender system for finding passengers and vacant taxis. *IEEE Transactions on knowledge and data engineering*, *25*(10), 2390–2403.

Zhang, H., Wu, H., Sun, W. & Zheng, B. (2018). Deeptravel: a neural network based travel time estimation model with auxiliary supervision. *arXiv preprint arXiv:1802.02147*.

Zhang, L., Yin, Y. & Lou, Y. (2010). Robust signal timing for arterials under day-to-day demand variations. *Transportation Research Record*, *2192*(1), 156–166.

Zhang, Q., Yang, L. T., Chen, Z. & Li, P. (2018). A survey on deep learning for big data. *Information Fusion*, *42*, 146–157.

Zhang, R., Schmutz, F., Gerard, K., Pomini, A., Basseto, L., Hassen, S. B., … Tonguz, O. (2018). Virtual traffic lights: System design and implementation. In *2018 ieee 88th vehicular technology conference (vtc-fall)* (pp. 1–5).

Zhang, Y. & Haghani, A. (2015). A gradient boosting method to improve travel time prediction. *Transportation Research Part C: Emerging Technologies*, *58*, 308–324.

Zhao, Y. & Tian, Z. (2012). An overview of the usage of adaptive signal control system in the United States of America. In *Applied mechanics and materials* (Vol. 178, pp. 2591–2598).

Zheng, F., van Zuylen, H. J., Liu, X. & Le Vine, S. (2016). Reliability-based traffic signal control for urban arterial roads. *IEEE Transactions on Intelligent Transportation Systems*, *18*(3), 643–655.

Zhou, Z.-H. (2012). *Ensemble methods: foundations and algorithms*. CRC press.

Zhu, C., Chen, X. & Bing, Z. (2014). Research on vehicles routing under emergencies. *JOURNAL OF INFORMATION &COMPUTATIONAL SCIENCE*, *11*(16), 5969–5976.