Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC)

By
Samah Mohamed El-Shafie Mohamed El-Tantawy

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy

Graduate Department of Civil Engineering
University of Toronto

Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC)

Samah El-Tantawy

Doctor of Philosophy

Department of Civil Engineering

University of Toronto

2012

ABSTRACT

The population is steadily increasing worldwide resulting in intractable traffic congestion in dense urban areas. Adaptive Traffic Signal Control (ATSC) has shown strong potential to effectively alleviate urban traffic congestion by adjusting the signal timing plans in real-time in response to traffic fluctuations to achieve the desired objectives (e.g., minimizing delay). Efficient and robust ATSC can be designed using a multi-agent reinforcement learning (MARL) approach in which each controller (agent) is responsible for the control of traffic lights around a single traffic junction. Applying MARL approaches to ATSC problem is associated with a few challenges as agents typically react to changes in the environment at the individual level but the overall behaviour of all agents may not be optimal. This dissertation presents the development and evaluation of a novel system of Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC). The MARLIN-ATSC control system is developed to provide a self-learning ATSC using a synergetic combination of reinforcement learning approaches and game theory. MARLIN-ATSC operates in two modes: (1) independent mode, i.e. each intersection controller operates independently of other agents; and (2) integrated mode, where each controller coordinates the signal control actions with the neighbouring intersections. The system was tested on three networks (i.e., small, medium, large-scale) to ensure seamless transferability of the system design and results. The large-scale application was conducted on a computerized testbed network of 60 intersections in the lower downtown core of the City of Toronto for the morning rush hour handling 25,000 trips. The results show unprecedented reduction in the average intersection delay ranging from 27% in mode 1 to 39% in mode 2 at the network level; and travel time savings of 15% in mode 1 and 26% in mode 2, along the busiest routes in downtown Toronto. The thesis shows how mathematical modelling of the traffic control problem as a stochastic control problem, combined with the utilisation of artificial intelligence techniques such as reinforcement learning in a game-theory setup, can provide highly useful and economically inexpensive solutions to real-life problems such as urban traffic congestion.

EXTENDED ABSTRACT

The population is steadily increasing worldwide. Consequently the demand for mobility is increasing, resulting in severe traffic congestion in urban areas. Traffic congestion leads to undesirable impacts on mobility, accessibility and socio-economic activities, as well as the environment. Traffic congestion costs the Greater Toronto Area (GTA) $6B / year. Infrastructure expansions have been primarily used to alleviate congestion until relatively recently. However, very high constraints on financial resources and physical space have accentuated the need for alternative mitigations to traffic congestion. Adaptive Traffic Signal Control (ATSC) has the potential to effectively alleviate urban traffic congestion by adjusting the signal timing plans in real time in response to traffic fluctuations to achieve the desired objectives (e.g. minimising delay). However, existing ATSC systems (e.g. SCOOT in Canada) have limited closed loop optimal control and coordination abilities. In addition they are costly to implement and operate due to their centralised mode of operation and complexity of use. A typical SCOOT installation usually costs between $40,000 and $80,000 per intersection; which limits their applicability, in addition to the specialised skills and cost required to maintain and operate the system.

Reinforcement Learning (RL) has shown promising potential for self-learning ATSC due to its ability to perpetually learn and improve the service (traffic conditions) over time. In RL, a traffic signal represents a control agent that interacts with the traffic environment in a closed-loop system to achieve the optimal mapping between the environment's traffic state and the corresponding optimal control action, offering an optimal control policy. The agent iteratively receives feedback reward for the actions taken and adjusts the policy until it converges to the optimal control policy. Once the optimal policy is learned, the mapping of the observed system states to the optimal control actions is very fast. For a network with multiple signalised intersections, efficient and robust controllers can be designed using a multi-agent reinforcement learning (MARL) approach.

MARL is an extension of RL to multiple agents in a stochastic game (SG) (i.e. multiple players in a stochastic environment). However, applying MARL approaches to an adaptive traffic control problem is associated with a few challenges. Agents typically react to changes in the environment at the individual level, but the overall behaviour of all agents (e.g. in a network of agents/intersections) may not be optimal. Each agent is faced with a moving-target learning problem, as changes to the environment not only depend on its actions but also on the actions of the other agents. The agent's optimal policy changes as the other agents' policies change over time, which requires a coordination mechanism among agents. Moreover, for medium-to-large transportation networks the number of system states and actions grows exponentially as the number of agents/intersections grows. Although MARL approaches are investigated in the literature, agents still take locally optimal decisions (actions) independently, without coordination with their neighbours. As a result agents may select individual actions that are locally optimal, but that collectively produce inefficient suboptimal solutions.

This research presents the development and testing of a novel system of Multi-Agent Reinforcement Learning for an Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC). The MARLIN-ATSC control system is developed to provide self-learning ATSC using a synergetic combination of RL approaches and Game Theory (GT) concepts (from Artificial Intelligence) that enables traffic lights to time themselves in a manner that minimises motorist delay and other externalities. MARLIN attains the challenging compromise of achieving coordination-based decentralised adaptive traffic control without suffering from the curse of dimensionality. In this system, each agent plays a game with its adjacent intersections in its neighbourhood. MARLIN works in two possible modes: (1) independent mode, i.e. each intersection controller has a smart agent working independently of other agents; and (2) integrated mode, where each controller coordinates the signal control actions with the neighbouring intersections. MARLIN-ATSC offers the following features: 1) decentralised design and operation – typically costing between $10,000 and $30,000 per intersection, 2) scalable – to accommodate any network size, 3) robust – with no single point of failure, 4) model-free – does not require a model of the traffic system that is challenging to obtain, 5) self-learning – reduces human intervention in the operation phase after deployment (the most costly component of operating existing ATSCs), and 6) coordinated – by implementing mode 2 (integrated mode), which coordinates the operation of the intersections in two-dimensional road networks (e.g. grid networks), and introduces a new feature that is unprecedented in ATSC state-of-the-art and practice.

After the development phase was complete, an in-depth analysis and quantitative evaluation of the potential benefits of MARLIN-ATSC is conducted on a simulation testbed of downtown

Toronto. The MARLIN system was tested on three networks of varying sizes; small, medium, and large, to ensure seamless transferability of the system design and results. MARLIN was found to reduce intersection delays with up to 48% relative to optimal pre-timed and traffic responsive actuated control. The large-scale application of MARLIN-ATSC was conducted on a simulation testbed of 59 intersections in the lower downtown core of Toronto during the morning rush hour, resulting in 25,000 trips. MARLIN-ATSC was compared against the pre-timed and actuated timing plans provided by the City of Toronto. The results show an unprecedented reduction in the total traffic delay at intersections, ranging from 27% in mode 1 (independent mode) to 39% in mode 2 (integrated mode) at the network level; and travel time savings of 15% in mode 1 and 26% in mode 2, along the busiest routes of downtown Toronto. In addition, CO2 emissions and fuel consumption savings are estimated to be around 30%. Moreover, the simulation-based nature of training MARLIN enables the recommendation of the best candidate intersections and routes for MARLIN-ATSC treatment and also helps prioritise the roll-out of adaptive control in the field.

The thesis shows how mathematical modelling of the traffic control problem as a stochastic control problem, combined with the utilisation of artificial intelligence techniques such as reinforcement learning in a game-theory setup, can provide highly useful and economically inexpensive solutions to real-life problems such as urban traffic congestion.

ACKNOWLEDGEMENT

First of all, I sincerely thank Allah, my God, the Most Gracious, Most Merciful, for enabling me to finish my PhD thesis and for the so many blessings and loving people I have in my life. His grace is abundant throughout my life.

After the five years of my doctorate journey, it is finally time to finish my thesis and have the chance to acknowledge and thank the people who have helped me in many ways.

First and foremost I would like to express my deep gratitude to my thesis advisor Prof. Baher Abdulhai. Prof. Abdulhai, you have been a real model of how an advisor should be. This is not only because of your sincere efforts throughout my PhD work, but also because of your longsighted vision, continuous encouragement and support. I feel I am lucky that I have you as my advisor. Your continuous support and your high expectations of me had always driven me to challenge myself to do better and better throughout my 5 year journey. I cannot forget your appreciation and support when I had my first baby during my second year of my PhD, and how you guided me to smoothly progress in my PhD. Thank you for your eagerness to allow me to attend various conferences and present my work.

I would like to thank the committee for their feedback as received in the progress reports, and from the informal discussions we had. Thank you for spending your valuable time on this thesis.

I cannot begin to express my heartfelt thanks to my beloved parents for all their love and continuous encouragement. Although away in Egypt they have been always in touch on an almost daily basis. They were eager to listen to my news and sometimes to my complaints when struggling in my research, without their phone calls and follow-up I would have lost my momentum to finish up my PhD. Also I am deeply indebted to my dearest sisters, Hanan and Amal, and my lovely brothers, Ahmed and Hossam, for their love and constant motivation throughout my life, as well as my Uncle Taha for his continuous care and support. I owe them everything and wish I could show them just how much I love and appreciate them. I would like also to express my deep gratitude and respect to my in-laws for their love and support.

Special thanks go to my husband Hossam for his support, love and continuous help. I do not know how I would have progressed in my PhD without you. You were always like the sunshine to me, giving me feeling of hope and peace. It is so wonderful to have an understandable and knowledgeable person like you beside me, in the past while finishing my MSc, in the present while finishing my PhD, and in the future. You always cared about my problems as your own, guided me to solve them, encouraged me to achieve more and more. I cannot count how many times you took care of our lovely daughter and other home duties to maintain a calm, fruitful environment for me to work. Thank you to your endless help and lovely spirit.

Nour, my lovely daughter, you are the most precious thing in my life; you are the most beautiful girl I have seen in the world. You give me a new purpose of life. How many times you receive me after a long day of work with a warm hug, a cute face, and a lovely smile that always ensured all the duties and difficulties of work were left at the door. My lovely daughter, I hope you understand and appreciate why Mom could not spend as much time as she would have loved with you when you were three years old.

Thanks are also extend to my colleagues and friends in the Transportation Group; most notably Mohamed Wahba, Glareh Amirjamshidi, Aya Aboudina, Toka Moustafa, Sarah Salem, Mohamed Elshenawy, and Bryce Sharman. Mohamed Wahba, your work has been a model for me; I admired your deep knowledge of different topics other than transportation and how you harnessed this knowledge to transportation engineering. Glareh Amirjamshidi, thank you for your support and help in emission modelling and making the CMEM plug-ins available to me. Aya Aboudina, I have learnt many things from you during my PhD. I have learnt a great deal from your organised manner of presenting your ideas in our group meetings. You have been always a close and lovely friend during the past 6 years. Toka Moustafa, thank you for your support in the GIS work that I have done in my thesis. Second, and more importantly, thank you for your continuous encouragement and tireless attempts to make everybody around you happy. Sarah Salem I really enjoy your smiles and lovely spirit in our office and have learned a great deal from your positive attitude when dealing with any problem. Kasra Rezaee, I enjoyed every fruitful discussion we had and learned a lot from your critical thinking. Mohamed Elshenawy, I really appreciate your help, which I received whenever I asked for it. Bryce Sharman, I enjoyed working with you in organising extra-curricular activities in our group and learned a lot from your leadership capabilities and time management.

Words fail me in expressing my gratitude to my dear friends Sarah El-Mallah, Noha Wahid, Huda Abdelhamid, Dalia Hussein, and Aya Huzzayin who have always been there with great support throughout my PhD.

Special thanks go to Asmus Georgi, the ITS Lab Manager for his hard work in setting up the environment in the lab for everyone to work efficiently.

I would like to extend my gratitude to the staff of the City of Toronto for the data they provided to me, with special thanks to Rajnath Bissessar, Linda Francis, Yasmin Fong, Jim Thomas Smith, Peter Lasagna, and Simon Foo.

In this thesis, portions of four chapters have been reproduced (with modifications) from my previously published material. These chapters are:

Chapter 2: Literature Review

- El-Tantawy, S. and B. Abdulhai (2010). Towards multi-agent reinforcement learning for integrated network of optimal traffic controllers. *Transportation Letters: The International Journal of Transportation Research*.

Chapters 4, 5 and 6: MARLIN-ATSC Framework, Experimental Results 1: Isolated Intersection, and Experimental Results 2: Prototype Network

- El-Tantawy, S. and B. Abdulhai (2010). Temporal difference learning-based adaptive traffic signal control. *Proceedings of the 12th World Conference on Transport Research (WCTR), Lisbon, Portugal*.
- El-Tantawy, S. and B. Abdulhai (2010). An agent-based learning towards decentralized and coordinated traffic signal control. *Proceedings of the 13th International IEEE Annual Conference on Intelligent Transportation Systems (ITSC), Madeira, Portugal*.
- El-Tantawy, S. and B. Abdulhai (2011). Comprehensive analysis of reinforcement learning methods and parameters for adaptive traffic signal control. *Proceedings of Transportation Research Board Conference, Washington D.C.*
- El-Tantawy, S. and B. Abdulhai (2012). Neighbourhood coordination-based multi-agent reinforcement learning for coordinated adaptive traffic signal control. *Proceedings of Transportation Research Board Conference, Washington D.C.*
- El-Tantawy, S. and B. Abdulhai (2012). Multi-agent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC). *Proceedings of the 15th International IEEE Annual Conference on Intelligent Transportation Systems (ITSC), Alaska, USA.*

- The signal control system introduced in this thesis (MARLIN-ATSC) is patent-pending. The patent application was filed on 20th December 2011:
  - El-Tantawy, S. and B. Abdulhai. *Multi-Agent Reinforcement Learning for Networked Adaptive Traffic Signal Control*. US Patent Application No. 61/576,637

- Since 2nd April 2012, the University of Toronto is working jointly with an industrial partner (CIMA+) and the City of Burlington, Ontario, on a pilot study to evaluate the performance of MARLIN-ATSC in one of the busiest areas of the city. Pending the results of this on-going evaluation, the City of Burlington may consider deploying the MARLIN-ATSC system in the field.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

1-GC: First Generation of Control
2-GC: Second Generation of Control
3-GC: Third Generation of Control
4-GC: Fourth Generation of Control
AMSS: Arterial Master Signal System
ARF: Admissible Region Flag
ASF: Agent Synchronisation Flag
ATSC: Adaptive Traffic Signal Control
BC: Base Case
BR: Best Responses
DP: Dynamic Programming
ESF: Environment Synchronisation Flag
FPS: Fixed Phasing Sequence
GT: Game Theory
ITS: Intelligent Transportation Systems
LT: Left-Turn
MARL: Multi-Agent Reinforcement Learning
MARL-I: Multi-Agent Reinforcement Learning for Independent Controllers
MARLIN: Multi-Agent Reinforcement Learning for Integrated Network
MARLIN-ATSC: MARLIN of Adaptive Traffic Signal Controllers
MARLIN-DC: MARLIN of Direct-Coordinated Controllers
MARLIN-IC: MARLIN of Indirect-Coordinated Controllers
MARLIN-IC-A: MARLIN-IC Along Arterial Neighbours Only
MARL-PI: Multi-Agent Reinforcement Learning for Partially Independent Controllers
MARL-TI: Multi-Agent Reinforcement Learning for Totally Independent Controllers
MTSS: Main Traffic Signal System
NE: Nash Equilibrium
NEMA: National Electrical Manufacturers Association
OPAC: Optimised Policies for Adaptive Control
PRODYN: Programmation Dynamique
RHODES: Real-Time, Hierarchical, Optimised, Distributed, and Effective System
RL: Reinforcement Learning
SCATS: Sydney Coordinated Adaptive Traffic System
SCOOT: Split, Cycle, and Offset Optimisation Technique
SG: Stochastic Game
SR: Shared Repository
Std: Standard Deviation
TD: Temporal Difference
UTOPIA/SPOT: Urban Traffic Optimisation by Integrated Automation/Signal Progression Optimisation Technology
VPS: Variable Phasing Sequence

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 PROBLEM STATEMENT

The population is steadily increasing worldwide; consequently the demand for mobility is increasing, especially in times of good economy. When the growth in social and economic activities outpaces the growth of transportation infrastructure, congestion is inevitable. Severe congestion and long commutes plague many large urban areas around the word, and the Greater Toronto Area (GTA) is no exception. Congestion wastes time, hampers social and economic activity and harms the environment, all of which deteriorate the quality of our lives. Traffic congestion, as a major player in the economic cycle that has a direct impact on the national GDP, is costing the GTA $6B a year according to 2008 statistics (Metrolinx, 2008). Infrastructure improvements have been primarily used to alleviate traffic congestion until relatively recently. However, tight constraints on financial resources and physical space, as well as environmental considerations, have accentuated the need for alternative options to mitigate traffic congestion.

Therefore, the emphasis has shifted towards improving the existing infrastructure by optimising the utilisation of the available capacity. Intelligent Transportation Systems (ITS) achieve efficient operation of the transportation system – using telecommunication, information technology, and advanced control techniques – without expanding the existing infrastructure or building more roads. In urban areas, advancements in ITS and traffic signal control have the potential to substantially alleviate traffic congestion and long queues at intersections.

Pre-timed and actuated traffic signal control systems are the most commonly used control systems. Pre-timed signal control implements optimised but fixed timing plans; therefore it is not designed to adapt to rapid fluctuations in traffic flow. Although simple and not requiring skilled staff, the old practices of pre-timing traffic signals is laborious, time-consuming and tedious. Moreover, signal timing plans are known to age with time, i.e. many traffic lights operate with timing plans that were designed months or even years ago. Actuated signal control, on the other hand, reacts to changes in the demand patterns by implementing a window of green time (minimum green to maximum green) as opposed to the fixed green time in pre-timed signal control. Although proven to perform better than pre-timed signal control in most cases, actuated signal control does not offer any real-time optimisation of right-of-way allocation to properly adapt to traffic fluctuations. Therefore, actuated signal control is not adaptive to traffic fluctuations and might result in very long queues in grid-like networks (Zhang *et al.*, 2005).

Adaptive Traffic Signal Control (ATSC) has the potential to efficiently alleviate traffic congestion by adjusting the signal timing parameters in response to traffic fluctuations to achieve a certain objective (e.g. to minimise delay); therefore it has a great potential to outperform both pre-timed and actuated controls (McShane *et al.*, 1998). Several adaptive signal control methods have been implemented worldwide, including: SCOOT (Hunt *et al.*, 1981), SCATS (Sims and Dobinson, 1979), PRODYN (Farges *et al.*, 1983), OPAC (Gartner, 1983), and RHODES (Head *et al.*, 1992).

Although existing ATSC systems offer a wide range of performance improvements over pre-timed and actuated signal control, they still suffer from combinations of the following limitations:

- Employing a centralised control approach that limits the scalability and robustness of the overall system, especially in cases of communication failure between the intersections and the traffic management centre;
- Expensive to install, pay for communication lines, and operate; a typical SCOOT installation, the most commonly used ATSC system worldwide, usually costs between $40,000 and $80,000 per intersection (NCHRP, 2010);
- What is known as the “curse of dimensionality” when handling several intersections simultaneously, in which case the complexity of the system increases exponentially with the number of intersections;
- Dependence on an accurate traffic modelling framework that captures the dynamic and stochastic traffic assignment in response to signal control optimisation changes;
- The need for highly-skilled labour which is often hard to find, train and retain at municipalities or even large cities with ample resources. This problem is typical with advanced systems and knowledge-intensive applications. There is a need for considerable expertise to ensure the successful operation and implementation of an ATSC system. For instance, proper knowledge of and training on SCOOT is difficult and costly due to the complexity, steep learning curve and the time and expertise required for configuring and

managing the system. This shift in signal control labour and staff requirements from basic to highly-skilled is often a challenge that municipalities typically overlook and realise later, and often causes the discouragement of small/medium agencies/municipalities to adopt or expand ATSC systems.

- Limited applicability to either isolated intersections or a group of intersections along arterials. Although it is important to efficiently operate traffic signals along arterials where the major demand is (e.g. progression), it is also important to consider the network-wide effect of such operation. In a signalised urban network setting, considering a network-wide objective has the potential to improve the overall network performance, mobility, and reduce emissions.
- Dependence on frequently malfunctioning magnetic loop detectors for the detection and estimation of traffic arrivals. Loop detector failure (e.g. a 10% failure rate in Toronto) is a known issue. Upkeep of loop detectors is known for being costly, traffic disruptive and difficult to maintain during the winter season. Therefore, when a loop detector fails, the agency/city is often unable to fix it for months and hence the related ATSC (e.g. SCOOT) based intersections revert to a pre-timed control model until the detector is fixed.

1.2 OVERVIEW OF THE PROPOSED METHODOLOGY

Due to the stochastic nature of traffic flows there is a pressing need for a control strategy that is adaptive to the stochastic fluctuations in traffic flows and that does not require a pre-specified model of the traffic environment. Reinforcement Learning (RL) has shown great potential for self-learning traffic signal control in the stochastic traffic environment (Abdulhai and Kattan, 2003; Abdulhai *et al.*, 2003; Bazzan, 2009; Chen and Cheng, 2010; El-Tantawy and Abdulhai, 2010). RL methods are capable of solving many real-world problems and require little (or no) prior knowledge about the environment. RL can deal with stochastic changes in the environment, which is of the essence in traffic signal control problems. In RL an agent (e.g. signalised intersection) interacts with its environment (e.g. traffic network) in a closed-loop system in which the agent acts as the controller of the process (Sutton and Barto, 1998). The agent iteratively observes the *state* (e.g. queue lengths) of the environment, takes an *action* (e.g. switch phasing) accordingly, receives a feedback *reward* (e.g. delay savings) for the actions taken, and adjusts its policy until it converges to the optimal mapping from *states* to optimal *actions* that maximises the cumulative *reward* (e.g. maximises throughput or minimises delay).

Accumulating the maximum reward not only requires the traffic signal control agent to *exploit* the best-experienced actions, but also to *explore* new actions to potentially discover better actions in the future.

The interaction between the agent and the environment can be viewed as two processes performed repeatedly; a learning process and decision making process. In the learning process the agent adjusts the policy by updating the long-term value associated with each state-action pair using the immediate reward. In the decision making process, exploration and exploitation are balanced by implementing action selection algorithms such as ε-greedy and softmax (Gosavi, 2003).

Applying RL to a transportation network of multiple signalised intersections is faced with some challenges. Agents typically react to changes in the environment at an individual level, but the overall behaviour of all agents may not be optimal. Each agent is faced with a moving-target learning problem in which the agent's optimal policy changes as the other agents' policies change over time. Because agents are learning concurrently; the reward an agent receives depends not only on its own actions but also on the actions taken by the neighbouring agents (Bazzan, 2009). Game Theory (GT) provides the tools to model the multi-agent systems as a multiplayer game and provides the rational strategy for each player in a game. For a network of multiple signalised intersections, efficient and robust controllers can be designed using a Multi-Agent Reinforcement Learning (MARL) approach. MARL is an extension of RL used to model multiple agents in a stochastic game (SG) (i.e. multiple players in a stochastic environment). The decentralised traffic control problem is an excellent testbed for MARL due to the inherited dynamics and stochastic nature of the traffic system (Bazzan, 2009; El-Tantawy and Abdulhai, 2010). Unlike single agent control – that has been extensively studied over the past two decades – the MARL problem is still maturing in traffic control problems.

Although RL offers a promising self-learning method to the closed-loop traffic control problem, it faces numerous challenges. A common challenge in any RL system is the design of the following elements/parameters: exploration method, learning method, action definition, state representation, and reward function identification. In the literature, different RL algorithms, and specifically Temporal Difference (TD) algorithms, have been investigated separately without providing quantitative justification for the RL design parameters in solving the traffic signal control problem.

In addition to the typical RL challenges stated above, MARL faces many other challenges. First is exponential growth in the state-action space with the increase in the number of agents. Second, and more importantly, the majority of MARL-based ATSC in the literature assume that agents learn independently, in which case each agent acts individually in its local environment without explicit coordination[1] with other agents in the environment. Although this simplifies the problem, it limits their usefulness in the case of a network of agents. For example, in oversaturated traffic conditions queues could easily propagate from a downstream intersection (agent) and spill back to the upstream intersections (agents) in a network-wide cascading fashion; such cases require network-wide multi-agent coordination as discussed earlier.

1.3 Research Objectives and System Features

The objective of this research is to develop a novel Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC). To address the limitations of the existing state-of-the-practice (e.g. SCOOT) and state-of-the-art (e.g. MARL-based ATSC systems), MARLIN-ATSC is designed to offer the following features and characteristics (Figure 1-1):

1. Decentralised: enables controlling a large urban traffic network through a number of overlapping sets of control agents. Decentralisation ensures system robustness and eliminates the issue of a system-wide single point of failure in the centralised system. In addition it is less costly than centralised systems (usually costing between $10,000 and $30,000 per intersection (NCHRP, 2010));
2. Sensitive: senses the stochastic traffic patterns and therefore does not require a separate prediction module to estimate traffic arrivals;
3. Model-Free: improves the control policy over time without the need for an explicit model of the environment (e.g. state transition probabilities);

[1] *It is important to not confuse coordination concerned with the creation of a green wave along a certain corridor by adjusting the offset timing (hereafter termed progression) with the mechanism between agents (signalised intersections) to coordinate their policies such that a certain objective is achieved for the entire traffic network (hereafter called coordination). In this thesis, coordination refers to the latter.*

4. Self-Learning: alleviates the complexity of operating the system, reduces skilled staff requirements and enables traffic operators to focus on high-level monitoring and supervisory roles instead of the low-level management of traffic systems operations;
5. Efficient: handles oversaturated traffic conditions by developing an explicit coordination mechanism among agents at both the learning and decision making levels. This coordination allows, for instance, the automatic discovery of best traffic metering strategies to hold traffic at an upstream intersection to protect an oversaturated downstream intersection;
6. Coordinated: is capable of coordinating the operation of traffic signals in two-dimensional urban networks (e.g. grid networks);
7. Scalable: models any number of agents (e.g. intersections) without suffering from the exponential growth dimension problem with the increase in the number of signalised intersections in the network. Coordination propagates network-wide through overlapping sets of agents in a cascading manner;
8. Generic: is designed to enable the testing of different levels of coordination, learning methods, state representations, phasing sequence, reward definitions, action selection strategies, and any control task (e.g. signalised intersections, variable message signs, speed control, ramp metering, etc., as well as non-transportation applications). In addition, its simulation environment interface models:
    a. different intersection layouts
    b. different phasing sequences and numbers of phases
    c. different sensing mechanisms;
9. Capable of quantifying detailed performance measures at the local intersection level, the corridor level, and the network-wide level. This helps municipalities identify best candidate intersections for ATSC treatment and prioritise the roll-out of ATSC in the field.

The features above are the "system design" objectives of this research. MARLIN-ATSC can work in a traffic network in two possible modes: (1) independent mode, i.e. each intersection controller has a smart agent working independently of other agents; and (2) integrated mode, where each controller coordinates the signal control actions with neighbouring intersections.

Although developing the above MARLIN-ATSC features is essential for the "*system design*" objective, the implementation and testing of MARLIN-ATSC forms an equally important "*system analysis*" objective. MARLIN-ATSC is tested using the following experimental setup on a set of realistic networks using microscopic simulation models:

1. Apply MARLIN-ATSC to an isolated intersection and comprehensively analyse the effects of the following methods/parameters:
    a. Learning Method (Q-Learning, SARSA, Eligibility Traces).
    b. Exploration Method (Exploration vs Exploitation).
    c. Action Definition.
    d. State Representation.
    e. Reward Function;
2. Apply MARLIN-ATSC to a proof-of-concept prototype network of a few intersections and analyse the effect of different coordination levels:
    a. MARLIN-ATSC Independent Mode.
    b. MARLIN-ATSC Integrated Mode;
3. Apply MARLIN-ATSC to a large-scale network and analyse the effect of the following variables:
    a. Demand Variability.
    b. Drivers routing response based on their familiarity or unfamiliarity of the network;
4. Evaluate the performance of MARLIN-ATSC against the following benchmarks:
    a. Fixed-time Signal Control (with offset coordination in cases of corridors).
    b. Actuated Signal Control.

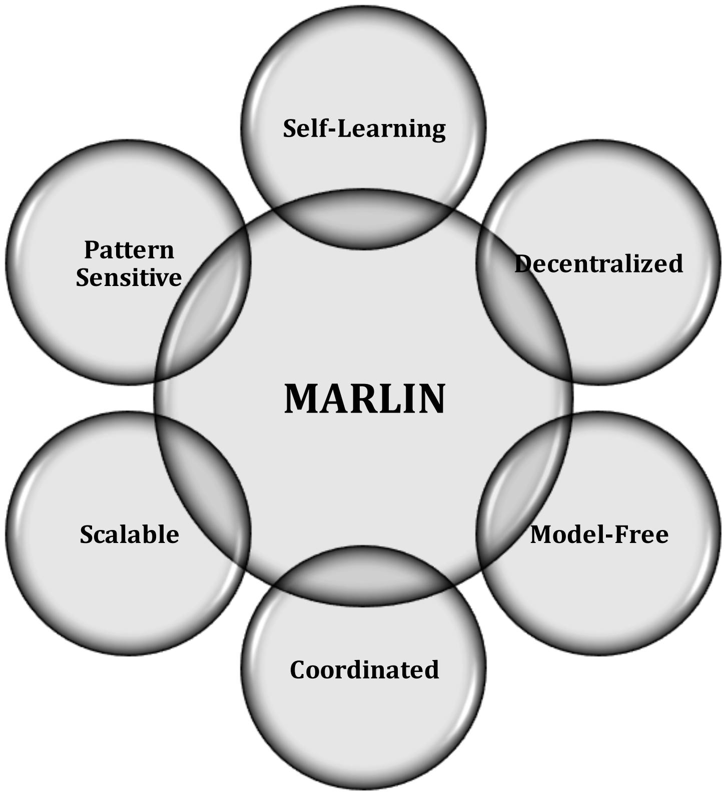


Figure 1-1 MARLIN-ATSC Design Features

1.4 THESIS ROADMAP

Based on the above defined objectives of MARLIN-ATSC, the primary research sections of this thesis have been organised into eight chapters. Figure 1-2 illustrates the thesis organisation and the interrelations among its chapters.

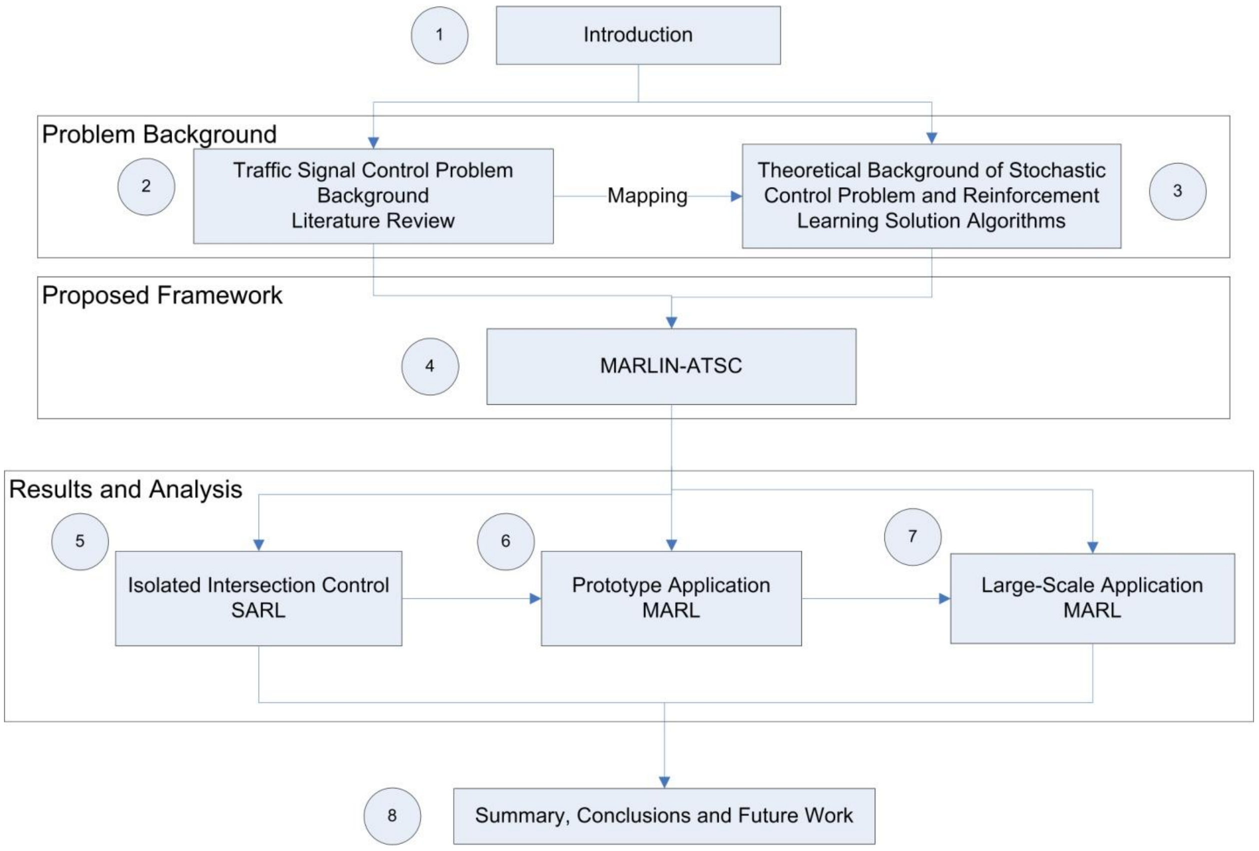

1
Introduction
Problem Background
2
Traffic Signal Control Problem Background Literature Review
Mapping
Theoretical Background of Stochastic Control Problem and Reinforcement Learning Solution Algorithms
3
Proposed Framework
4
MARLIN-ATSC
Results and Analysis
5
Isolated Intersection Control SARL
6
Prototype Application MARL
7
Large-Scale Application MARL
8
Summary, Conclusions and Future Work

Figure 1-2 Thesis Roadmap

Chapter 2 summarises the literature review of the state-of-the-practice and the state-of-the-art approaches in traffic signal control. It also highlights the history and trends in adaptive signal control research and development. Chapter 2 summarises the major challenges and gaps in existing state-of-the-practice literature. State-of-the-art studies considering RL and MARL in traffic control applications are also reviewed and their limitations are highlighted.

Chapter 3 reviews the theoretical foundation of the stochastic control problem as it represents the basis of the ATSC problem. The chapter reviews the problem from single-agent and multi-agent perspectives. Chapter 3 provides a brief description of the theoretical framework, solution, optimal methods, and RL/MARL methods applicable to the stochastic control problem. The challenges and limitations of MARL approaches are discussed in this chapter.

Chapter 4 details the development of the MARLIN-ATSC platform while highlighting how it addresses the limitations/gaps in the literature. It starts by providing a mathematical formulation for the MARLIN control algorithm which represents the theoretical contribution of the thesis. Then it provides a high-level description of the framework components.

Chapters 5, 6, and 7 demonstrate the applicability and feasibility of MARLIN in a set of simulation-based experiments. The experimental setup and performance details of MARLIN-ATSC-Independent-Mode are presented through a prototype implementation on an isolated intersection in Chapter 5. Chapter 5 provides guidelines and recommendations for the best RL-related design parameters in the ATSC problem (e.g. action definition, state representation, etc.). Chapter 6 presents the applicability of the two modes of operation of MARLIN-ATSC (independent and integrated) in a prototype network of 5 intersections. This prototype is important as it highlights the main principles of the MARLIN-ATSC; it does not, however, represent a large-scale implementation of the model. The application of MARLIN-ATSC to a large-scale case study of 59 intersections in downtown Toronto is presented in Chapter 7. The large-scale application demonstrates the essence of MARLIN-ATSC and builds on the lessons learned from the prototype implementations discussed in Chapters 5 and 6.

Chapter 8 summarises the major findings and the main contributions of this thesis. Chapter 8 concludes with proposing areas of research to enhance/extend the capabilities of MARLIN-ATSC.

2 LITERATURE REVIEW

A signalised intersection is designed to allow conflicting traffic movements to proceed efficiently and safely by separating the individual movements in time rather than in space. The basic unit of a traffic signal control system is the *signal group*, a set of lights that corresponds to a certain movement in the intersection. A *phase* is a set of signal groups that corresponds to a certain set of movements that have the right of way during the phase's *green-time*. These movements are chosen to proceed concurrently without conflicting with other traffic streams. A *signal cycle* is completed when each phase has been served once, hence the name *cycle time* (also known as *cycle length*). Signal control algorithms are typically designed to optimise the traffic flow using the *phase scheme,* the *split,* and the *offset* settings. The *phase scheme* determines the signal groups that have the right of way and their corresponding order in the cycle. A *split* represents a distribution of the *cycle time* to the individual phases, hence represents the green-time allocated to each phase. An *offset* refers to the time shift – with respect to a reference point in time – of the start of a new cycle. Offsets have been efficiently used to achieve progression along a certain corridor by creating a "green wave" for the vehicles travelling in one direction along this corridor, or minimising the number of stops and waiting times at intersections along the corridor.

It is important to not confuse the coordination achieved by adjusting the offset timing to create a green wave along a certain corridor (hereafter termed *progression*) with the mechanism between signalised intersections to coordinate their timings and splits such that a certain objective is achieved for the entire traffic network (hereafter called *coordination*). In this thesis, *coordination* refers to the latter.

The above-defined elements of a typical signalised intersection are maintained and operated by a traffic controller in the field. Traffic control systems differ in how they allocate the green time for each phase. As shown in Figure 2-1, in this chapter a comprehensive review of traffic signal control systems is presented and discussed. Traffic controllers can be classified into three main categories: Fixed-time Controllers, Actuated Controllers, and Adaptive Controllers. Adaptive signal controllers can be categorised into four generations (Gartner *et al.*, 1996): first generation of control (1-GC), second generation of control (2-GC), third generation of control (3-GC), and fourth generation of control (4-GC). In 1-GC, the optimisation of the signal timings is conducted off-line while it is conducted on-line in the other three generations. 1-GC and 2-GC are implemented using centralised control systems while the more recent two generations implement decentralised control systems. Due to the drawbacks associated with 3-GC systems (discussed in section 2.3.3), which limit their applicability, the 4-GC systems – although not yet implemented in the field – have the potential of maintaining the advantages of 3-GC systems while overcoming their limitations. Based on reviewing the literature on 4-GC systems using RL/MARL-based approaches, the studies are classified into three categories according to the level of interactions between the signalised intersections in a traffic network: Totally Independent MARL, Partially Independent MARL, and Coordinated MARL.

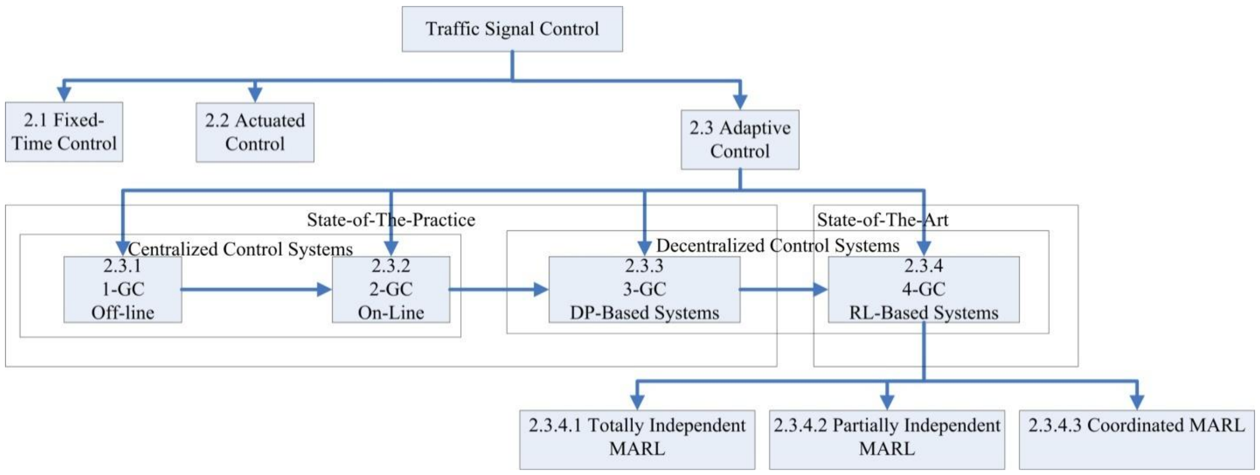


Figure 2-1 Chapter 2 Roadmap

2.1 FIXED-TIME CONTROL (PRE-TIMED CONTROL)

A fixed signal timing plan is developed off-line based on historical traffic data. The cycle length and splits are optimised off-line only once based on the historical traffic counts, and then deployed. Thereafter the duration and order of all green phases remain fixed and do not adapt to fluctuations in traffic demand. Fixed-time controllers do not require loop detectors at the intersection approaches to operate; hence its implementation cost is much lower than actuated and adaptive controllers. Fixed-time control is designed based on average historical counts which do not capture the dynamics and randomness associated with real-time traffic flows in the field. Therefore pre-timed control is best suited for locations where traffic exhibits predictably uniform arrival patterns over a long period of time. In addition, pre-timed control systems are susceptible to unpredicted events, accidents, and other disturbances that may disrupt traffic flows. The following sections explain two methods for obtaining signal timings for fixed-time signal control: isolated intersection-based control and progression-based control.

2.1.1 Isolated Intersection Fixed-Control: Webster Method

In fixed-time control, the signal's cycle time is distributed over the signal phases based on historical volumes. Between phase switches, start-up and clearance times are needed for drivers to clear the intersection; typically considered as lost time as they are not used for traffic flow. Therefore in one hour of traffic flow, shorter cycle times typically result in a higher percentage of lost times. Intersections with shorter signal cycles therefore operate at lower overall capacity levels. On the other hand, the longer the cycle times the longer the red phases and hence the longer the queues and the longer waiting times at the intersection. In addition, some approaches will drain their traffic faster than others and the remaining green times are not utilised, causing more time waste. Finding the balance between cycle time and intersection delay is not insurmountable. For known traffic flow rates, a formula can be derived to find an optimal cycle time and green splits of all phases such that the average delay for all vehicles is minimised. There are many such basic methods in the literature including, by way of an example, the Webster method (Webster, 1958).

Due to the complexity associated with calculating the delay, both theoretically and from direct observations in the field – due to the uncontrollable variations in the field; Webster (1958) introduced a computer simulation approach – in the British Road Research Laboratory – to derive the well-known Webster's delay formula for fixed-time signals (Webster, 1958). Details of the Webster method calculations are presented in Appendix 1.

The Webster method can also be used for designing actuated control, because the actuated control schemes have maximum green times equal to the fixed-time control. When traffic flows are consistently high, actuated control operates as fixed-time control. The Webster method is used in this research as a benchmark method to compare the efficiency of different control methods.

2.1.2 Progression-Based Fixed-Control

2.1.2.1 Bandwidth-Based Methods

Unlike the Webster method of finding the cycle length and phase splits, the bandwidth optimisation method uses traffic volumes, signal spacing, and a desired travel speed to determine the optimal progression band that can be achieved along a corridor. The bandwidth optimisation technique therefore attempts to provide the maximum progression band possible; therefore they generally result in longer cycle lengths. The following are two examples of bandwidth-based methods:

- MAXBAND developed by Little (1981) and extended to MULTIBAND by Gartner *et al.* (1990, 1991)
- PASSER (Progression Analysis and Signal System Evaluation Routine) developed by the Texas Transportation Institute (1998; Venglar *et al.*, 2000).

2.1.2.2 Disutility-Based Methods

Disutility-based methods are the second approach of the progression-based fixed control methods. They use a model to minimise the disutility of a specific measure such as intersection delay or the number of stops. These models are generally designed to first find a common cycle length that minimises the overall delay in the system, and then calculate the offset required for progression. Therefore disutility methods generally result in shorter cycle lengths than those optimised by bandwidth techniques. The following are two examples of disutility-based methods:

- TRANSYT (TRAffic Network StudY Tool (Hale, 2006)) developed by Robertson (1969)
- SYNCHRO® (Trafficware, 2012).

2.2 ACTUATED CONTROL

Unlike pre-timed signal control, actuated traffic signal control systems are responsive to traffic flow fluctuations. Actuated traffic control requires actuated traffic controllers and vehicle detectors placed on the approaches of the intersection. The actuated timing plan responds to traffic demand fluctuations by placing a *call* at the presence or absence of vehicles approaching or leaving the intersection, respectively. Once a call is received, the controller decides whether to extend or terminate the green phase in response to the actuation source. Although the actuated signal control settings are responsive to the presence of actuation (or *calls*), they are less sensitive to the traffic demand (i.e. number of vehicles) calling for the actuation (McShane *et al.*, 1998). Actuated control can be operated in one of two forms according to the designed phase detection: Semi-Actuated, and Fully Actuated.

2.2.1 Semi-Actuated Control

In semi-actuated control, detectors are only placed on minor streets (callable phases). The main street movements are coded as non-callable phase(s); hence activated for the entire split time every cycle, regardless of the traffic detection. The controllers are designed such that a *permissive period* is defined during which the controller unit allows the callable phases to be served upon a detection request. The callable phases are typically timed to maintain the minimum green time after which the green time could only be extended upon demand and to a maximum limit (refer to the mode of termination in section 2.2.3). In some cases, callable phases (minor streets) do not require all their allocated green time within a cycle, resulting in *unused* green time. In such cases the unused time is automatically re-assigned to the non-callable phases (main streets). Semi-actuated traffic control is, therefore, best suited for locations where local minor streets intersect with arterials and/or collectors.

2.2.2 Fully-Actuated Control

In fully-actuated control, detectors are typically placed on all traffic approaches, and phases could be skipped if no vehicular demand is detected. Similar to semi-actuated control, callable phases run their splits with a green interval that varies between minimum and maximum values depending on the traffic demand. After reaching the minimum green time for a phase, the actuation logic starts and eventually the phase terminates according to a specific criterion (refer to the mode of termination in section 2.2.3). Fully-actuated control is therefore appropriate for intersections exhibiting less predictable and high traffic volumes on all approaches.

2.2.3 Termination Mode of Operation

As discussed above, in actuated traffic control the callable and non-callable phases are extended as long as vehicles are detected on the approaches. These extensions have to terminate according to a certain termination criterion. The following is a list of termination criteria typically employed in actuated traffic control (as illustrated in Figure 2-2):

- Gapping Out Termination: a phase "gaps out" if a pre-determined threshold time, known as gap time, is reached. The gap time should be long enough to enable an approaching vehicle to travel from the detector location to the intersection. Once a phase gaps out, the controller will terminate the green time for the current phase and switch to the next – in sequence – demanded movement;
- Maxing Out Termination: a phase "maxes out" if a pre-determined maximum green time is reached. A maximum green recall is defined as a call placed on the phase. Once the recall starts, it prevents the phase from termination prior to expiration of that maximum green time;
- Forced Green Termination: in cases where progression is maintained along corridors, the controller extends the green to a certain level such that the progression is maintained; beyond which the green time is forced to terminate.

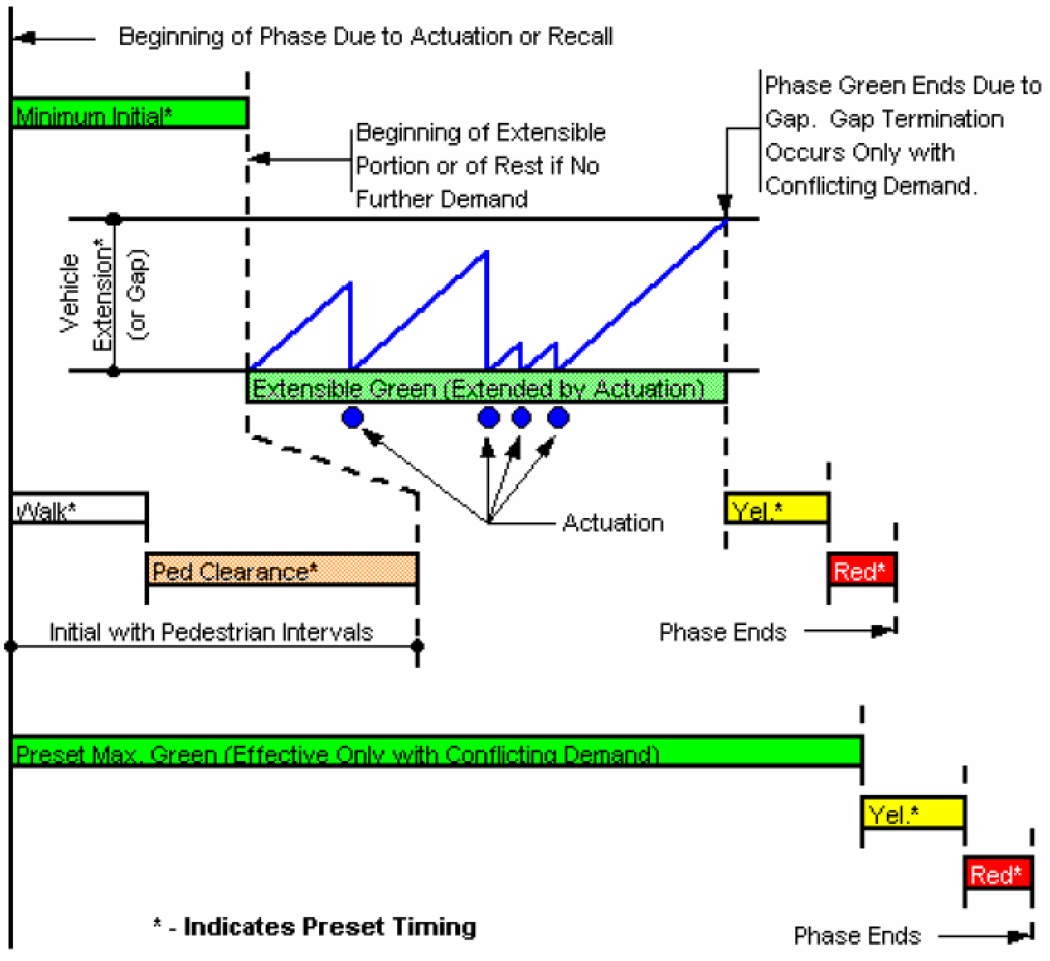

Beginning of Phase Due to Actuation or Recall
Minimum Initial*
Beginning of Extensible Portion or of Rest if No Further Demand
Phase Green Ends Due to Gap. Gap Termination Occurs Only with Conflicting Demand.
Vehicle Extension* (or Gap)
Extensible Green (Extended by Actuation)
Walk*
Actuation
Yel.*
Ped Clearance*
Red*
Initial with Pedestrian Intervals
Phase Ends
Preset Max. Green (Effective Only with Conflicting Demand)
Yel.*
Red*
* - Indicates Preset Timing
Phase Ends

Figure 2-2 Actuated Phase Timing Diagram (source: Trafficware, 2012)

2.3 Adaptive Control

In adaptive signal control, signal timing actions and/or plans are continuously updated according to traffic fluctuations and the number of vehicles approaching the intersection. Adaptive signal control parameters (e.g. cycle length, phase split, etc.) are determined to achieve a certain objective, and then the traffic controller automatically implements these parameters in response to the real-time traffic flows, typically measured by loop detectors.

Traffic flow is highly dependent on factors such as time, day, season, weather, and unpredictable events such as accidents, special events, work zones, etc. Such multi-dimensional variations in traffic flow patterns form a challenging task for fixed-time and actuated traffic control systems to efficiently operate signalised intersections. ATSC, on the other hand, has the potential to “adapt” to traffic fluctuations and can therefore alleviate traffic congestion in comparison to the more commonly used pre-timed and actuated control systems.

ATCS can be categorised into four levels or generations according to their level of adaptation and intelligence as illustrated in Figure 2-3. Table 2-1 summarises the main features of each generation. The categorisation presented in this section is commonly accepted in research as well as practice literature (Gartner *et al.*, 1996). First Generation Control (1-GC) uses offline calculation of signal timings, in which case the controller selects a signal timing plan from a pre-specified set of plans that are pre-optimised based on historical counts. Compared to 1-GC, the Second and Third control generations (2-GC, 3-GC) are distinguished by the optimisation technique for the timing plans and the hierarchy of control employed. Although still in the research stage and not yet implemented, Fourth Generation Control (4-GC) is seen as the next generation of ATSC as it maintains the same advantages of 3-GC while addressing the issues faced in the development of 3-GC (Gartner *et al.*, 1996). The 4-GC systems employ self-learning techniques that are based on direct experience with the traffic environment.

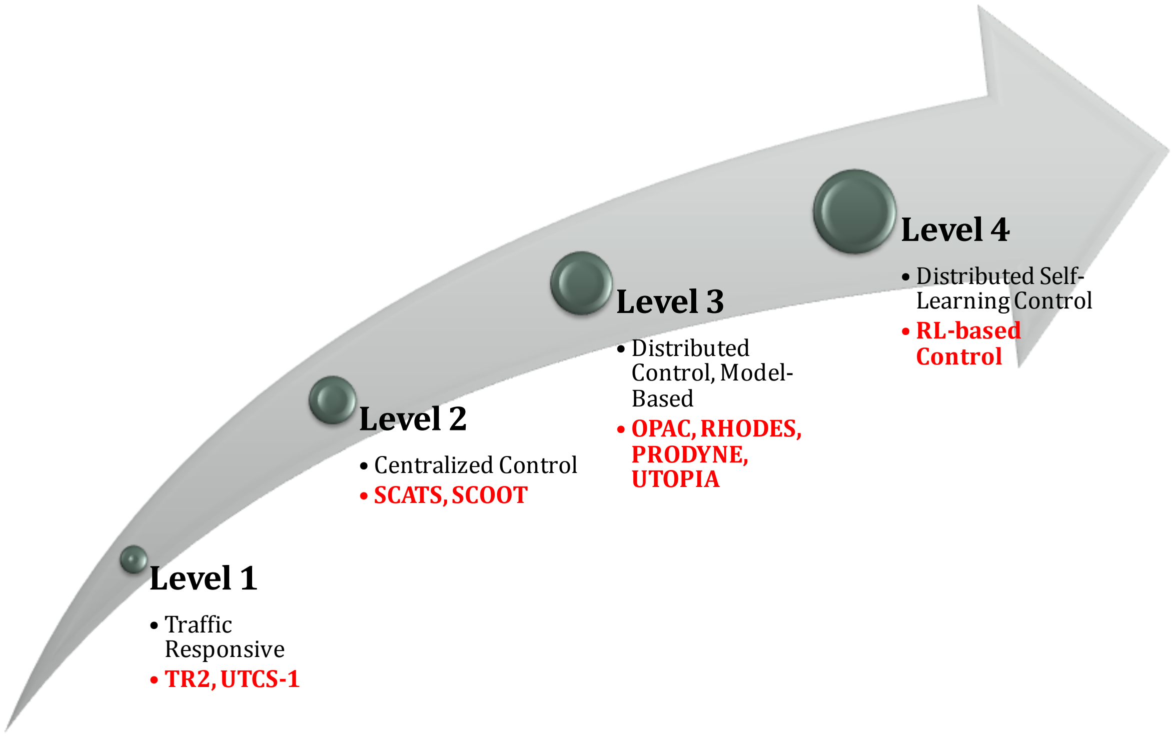


Figure 2-3 ATSC Evolution and Future Trend

Table 2-1 Summary of Adaptive Signal Control Generations and Features

Generation / Feature	First Generation (1-GC)	Second Generation (2-GC)	Third Generation (3-GC)	Fourth Generation (4-GC)
Optimisation	Off-line	On-line	On-line	On-line
Traffic Prediction	No	Yes	Yes	No
Predetermined Traffic Model	Yes	Yes	Yes	No
Hierarchies of Control	Centralised	Centralised	Decentralised	Decentralised
Cycle Length	Fixed	Fixed within variable groups of intersections	Variable	Variable

More than 20 different ATCSs have been developed during the last 30 years. However only about a dozen of them have been applied in the real world and have more than one field implementation. In this section, the focus will only be on the major systems that are implemented around the world under the four control generations (NCHRP, 2010).

2.3.1 First Generation Control (1-GC)

In 1-GC, a library of pre-stored signal control plans is implemented. These plans are developed off-line on the basis of historical traffic data. A few ad-hoc plans are typically designed based on the time of day (e.g. morning peak, off-peak, afternoon peak, evening period, midnight period) and the day of week (weekday vs weekend, Monday vs Friday, etc.). These plans are then selected and implemented directly by the operator. 1-GC traffic-adaptive systems are often referred to as traffic-responsive signal control. Due to its simple and straightforward operation, 1-GC systems dominated in practice for a few decades since the emergence of the first computerised traffic signal system in 1963. Examples of 1-GC include:

- TR2 (Traffic Responsive Control Mode 2) developed in Canada (Armstrong *et al.*, 1974)
- UTCS-1 (Urban Traffic Control System-First Generation) developed in the USA. (MacGowan and Fullerton, 1979).

2.3.1.1 1-GC Summary and Limitations

1-GC may plausibly operate under recurrent and defined peak traffic periods if random variations around the mean flows are not pronounced. In addition, incidents and unpredicted events are inevitable in transportation networks which cause traffic patterns to change substantially beyond the anticipated typical traffic flows and therefore cannot be efficiently handled by any pre-calculated and pre-scheduled timing plans. Furthermore, any pre-timed plans age with time as traffic flows change. Therefore, as traffic patterns change with time, the signal timing plans must be continually updated.

The limitations of 1-GC systems are summarised as follows:

1. Signal control plans are calculated to suit average traffic conditions per time of day and are typically updated at intervals that range from 2 to 10 years. These pre-scheduled plans can result in a 3% increase in intersection delay for each year lag if not frequently updated (Bell and Bretherton, 1986);
2. Signal control plans require extensive data collection effort every few years;
3. Transportation infrastructure and traffic networks do change frequently due to operational or capacity improvements, and the off-line pre-scheduled plans should be updated accordingly to reflect any change;

4. Traffic routing in transportation networks evolves dynamically with any change in signal timing plans; however offline plans typically neglect this fact by assuming fixed turning proportions at intersections (Roberston, 1979);
5. Offline signal control plans are incapable of handling: 1) stochastic variations in traffic patterns from day to day, and 2) non-recurrent traffic congestion.

2.3.2 Second Generation Control (2-GC)

2-GC ATSC systems employ on-line optimisation methods to dynamically adjust the signal timings (offsets, cycle time and splits) by utilising on-line surveillance information systems. In 2-GC, traffic progression along corridors is achieved in a centralised fashion. Examples of 2-GC include:

- SCATS (Sims and Dobinson, 1979)
- SCOOT (Hunt *et al.*, 1981).

2.3.2.1 SCATS

SCATS (Sydney Coordinated Adaptive Traffic System) (Sims and Dobinson, 1979) was developed in the early 1970s by the Roads and Traffic Authority of New South Wales, Australia. SCATS utilises a hierarchical system of the following components: 1) central computer, 2) regional computers, and 3) local controllers. The central computer monitors the system performance. In SCATS, the system is divided into a number of comparatively small subsystems, each of which is responsible for controlling a number of intersections (ranging from 1 to 10 intersections) including only one critical intersection – in each subsystem – that requires accurate phase splits. Each subsystem makes independent decisions about its timing parameters based on the requirements of the critical intersection and the flow travelling in each direction in the subsystem. Subsystems can be linked together to achieve traffic progression along a certain corridor for a number of signals by operating a common cycle length (the longest cycle time among the subsystems). In the SCATS hierarchy the second level is the regional computers which perform adaptive control strategies for each subsystem to respond to traffic fluctuations and congestion dynamics. At the lowest level, local controllers act in the traffic-actuated mode for individual intersections within each subsystem.

SCATS operates on two levels of control: strategic level and tactical level. The strategic control level determines the optimal signal timings for the subsystems based on average current traffic conditions. At the strategic control level, the basic traffic measure used by SCATS is the degree of saturation on each approach. This measure is used to determine the cycle length, splits, and the direction and value of the offset. The tactical control level on the other hand is concerned with the control of individual intersections (local controllers) while satisfying any constraints imposed by the strategic control level.

SCATS achieves a necessary synergy by combining strategic control, which determines the split, cycle time and offsets in response to gradual changes in traffic demand patterns, and tactical control, which handles the rapid but small changes in traffic demand cycle by cycle. This synergy enables SCATS to correct/overcome any errors/deviations in the prediction algorithm.

2.3.2.2 SCOOT

SCOOT (Split, Cycle, and Offset Optimisation Technique) was initiated by the British Transport and Road Research Laboratory (TRRL) in the 1970s, with the first commercial system being installed in 1980 (Hunt *et al.*, 1981). SCOOT is a centralised system that can be viewed as the on-line version of TRANSYT. SCOOT automates the off-line TRANSYT traffic signal optimisation model (Robertson, 1969) by utilising on-line surveillance information to incrementally adjust the signal timings. A SCOOT system divides the traffic network into “regions”, each consisting of a number of “nodes” (set of signalised junctions and pedestrian crossings with a common cycle length to allow for progression). SCOOT predicts the traffic arrival pattern based on the flow information collected at detectors upstream of the intersections. A SCOOT system requires the placement of upstream detectors (as far as possible, i.e. just downstream of the adjacent intersections). When vehicles pass the upstream detector, SCOOT converts this information into “link profile units” (lpu), a hybrid measure of link flow and occupancy. This unit is used over time to calculate “cyclic flow profiles” for each link approaching the intersection. This arrival profile is compared to the departure profile, and the difference between the two profiles represents the queued (and therefore delayed) vehicles at the intersection. SCOOT uses a combination of queued vehicles, the time to clear the queue, and the impact of the offset and split adjustment to estimate the traffic flows for each cycle. SCOOT then performs an optimisation algorithm at three levels: Split, Cycle and Offset.

2.3.2.3 2-GC Summary and Limitations

SCATS has been implemented in several countries around the world including: Australia, New Zealand, USA, China, Singapore, Philippines, and Ireland. SCOOT has been also widely used in several countries across the globe (e.g. UK, USA, Canada, China, South Africa, Cyprus, Pakistan, United Arab Emirates, Chile, and Spain). “Before” and “after” studies have been conducted to evaluate the benefits of SCATS and SCOOT in terms of reductions in intersection average delay, number of stopped vehicles, and travel times. SCOOT deployment reported a 17% reduction in intersection delay in the City of Toronto when compared to fixed timing control. SCATS reported travel time savings over fixed-timing control ranging from 7–23% depending on the time of day.

It is clear that 2-GC systems (SCATS and SCOOT) overcome some of the 1-GC systems’ drawbacks. SCATS uses a combination of strategic and tactical control to respond to the random fluctuation in traffic demand, while SCOOT uses a flexible plan that is expanded to suit the variations in demand which allows the transition to be less disruptive and less prone to overreacting than the transition between distinct plans. However 2-GC control systems still suffer from the following limitations:

1. SCOOT systems function effectively when traffic conditions are below saturation, but its performance declines in the case of severe congestion. SCOOT systems have been known to have a limited traffic-responsive behaviour during rapidly changing conditions (Papageorgiou, 2003);
2. SCOOT systems model the control problem at a “macro” level, in which the signal control parameters (e.g. cycle time, splits, and offsets) are optimised based on macroscopic traffic flow models. Although a simple approach to guide the optimisation process, it does not capture traffic flow models (e.g. car-following, gap and lane changes, etc.);
3. 2-GC systems are centralised systems which suffer from the following limitations (NCHRP, 2010):
    - They primarily depend on reliable communication networks because real-time commands are sent from a central computer to the local controllers/intersections. Therefore any failure in the communication network causes the local controller to switch to an off-line mode and reverts to its backup plan, thus operating as an isolated

intersection. In addition, the central computer is a risky single point of failure, i.e. failure of the central computer or the communication trunk near the central computer can bring down the entire signal control system;

- They are often not scalable to expand the size of the network. Traditional centralised systems are often designed around a maximum network size; therefore the addition of a few controllers requires extensive central computer and software upgrades;
- They are expensive, as the communication network's cost is typically at least two-thirds the overall system budget;
- They are relatively complex to operate with many parameters to be adjusted by a human operator. Maintaining highly-skilled and well-trained operators is expensive.

2.3.3 Third Generation Control (3-GC)

3-GC traffic adaptive systems are generally similar to the 2-GC systems in terms of adapting to traffic fluctuations. However they differ in the following important features:

- Operate in a decentralised control system which offers the following advantages over centralised control systems (Shoham *et al.*, 2003; NCHRP, 2010):
    - Decentralised systems are computationally less demanding as they only require and maintain the relevant information from the surrounding intersections/controllers.
    - Robustness is guaranteed in decentralised control system because if one or more controller fails, the remaining controllers can take over some of their tasks. Decentralised systems do not depend on real-time control commands from the central computer over the communication network; therefore they can still function even during communication disruption times.
    - Decentralised systems are scalable and easy to expand by inserting new controllers into the system.
    - Decentralised systems are often inexpensive to establish and operate as there is no essential need for a reliable and direct communication network between a central computer and the local controllers in the field. As a result, inexpensive communication alternatives, such as wireless communication networks, form a viable option that considerably reduces the system cost;

- Most 3-GC systems use dynamic programming (DP) techniques to model and solve the traffic signal control problem. DP is the theoretically ideal optimisation technique for decentralised closed loop optimal control, and it captures the stochastic nature and dynamics of the traffic system;
- 3-GC systems continuously update the signal plans parameters to respond to real-time field measurements, which therefore abandon the conventional notions of cycle length, phase splits, and offsets. Such flexibility in the signal timing setup enables the controller to produce "acyclic" signal settings (i.e. the cycle length is variable). In addition, 3-GC are flexible enough to relax the phase sequence constraint which leads to full adaptability to traffic flow fluctuations by possibly skipping unnecessary/undemanded phases.

Examples of 3-GC systems include:

- OPAC (Gartner, 1983)
- PRODYN (Farges *et al.*, 1983)
- UTOPIA/SPOT (Mauro and Di Taranto, 1989a)
- RHODES (Head *et al.*, 1992).

PRODYN and UTOPIA were developed and successfully tested through the DRIVE European research and development program (Dedicated Road Infrastructure for Vehicle Safety in Europe). On the other hand, OPAC and RHODES were successfully tested and implemented through a massive research project for Real-Time Traffic-Adaptive Signal Control Systems (RT-TRACS) that was sponsored by the Adaptive Control Software (ACS) program at FHWA in the USA.

2.3.3.1 OPAC

Gartner (1983) was the first to recognise the need to migrate from parametric models, which optimise parameters such as cycle time, splits, and offsets, to non-parametric models in which the decision to switch between phases is based on actual arrival data at the intersection. Such flexibility in the signal timing setup enables the controller to produce "acyclic" signal settings (i.e. the cycle length is variable), and hence it is more appealing for real-time signal control implementation. In OPAC (Optimised Policies for Adaptive Control), Gartner (1983) formulated the problem as a discrete-time optimal control problem for a single intersection, a formulation that was not practically solvable using DP-based methods. In an effort to adopt a practical solution method for this formulation, Gartner suggested the use of a restricted search heuristic (optimal sequential constraint search, OSCS) that enumerates a few alternative feasible solutions for a two phase intersection.

OPAC maximises the intersection throughput by considering the saturation flow and queue formation on each link. OPAC first determines the next phase to activate in cases where no critical link (a link with a queue spilling back to the upstream intersection) is identified. OPAC uses loop detector measurements to predict traffic arrival rates, which are then fed into the algorithm to determine the need for revisiting the neighbouring intersection timings in light of the intersection throughput and queue formation at neighbouring intersections.

More recently, OPAC has been extended to accommodate arterial networks (Gartner *et al.*, 1995). OPAC uses two levels of control in a decentralised fashion, a local level and a network level. At the local control level OPAC determines the next phase at the intersection. At the network control level OPAC provides *progression*. OPAC identifies several critical intersections using the measured traffic flows from all intersections within the controlled area, and then calculates a “virtual common cycle” length once every few minutes. A virtual fixed cycle is a cycle that is determined on-line and is fixed between intersections to enable progression. The length of the virtual cycle varies according to the needs of either the critical intersection or the majority of intersections. Therefore OPAC provides local progression by considering flows into and out of an intersection in selecting its offset and splits.

OPAC went through several developmental stages that ranged from OPAC-I to OPAC-VFC (Gartner, 1983; Gartner *et al.*, 1995, 1999, 2001). OPAC-I used DP to optimise intersection performance. OPAC-I could not be implemented in real-time because of the extensive time required to compute the optimal settings. OPAC-II used OSCS to calculate the total delay for all possible phase switching options. The optimal solution of OPAC-II was the phase switching that minimises the total intersection delay. Although OPAC-II was theoretically faster than OPAC-I, it predicts arrival traffic flows throughout the planning horizon; which lessens the speed advantage. OPAC-III overcame the limitation of OPAC-II by employing a rolling horizon approach on a simple two-phase intersection; and later OPAC-III was extended to an eight-phase intersection, which allowed phase skipping. OPAC-VFC extended the capabilities of OPAC-III to include an algorithm to achieve progression along corridors.

Although OPAC attempts to achieve theoretical optimum signal timing plans, it does not guarantee global optimality due to the approximation done to DP using the restricted search heuristic OSCS (Sen and Head, 1997). In addition, the application scale of OPAC is limited due to the tremendous computational effort involved in the OSCS search.

2.3.3.2 PRODYN

PRODYN (Programmation Dynamique) (Farges *et al.*, 1983) is a real-time traffic control system developed by the Centre d'Etudes et de Recherches de Toulouse (CERT), France.

In PRODYN, the switch-over control decides whether it is beneficial to switch from one phase to another using a set of discrete-time, non-linear state equations. Each intersection is optimised using a forward DP method, while progression is achieved in a decentralised fashion. A specific forward DP method was developed to reduce the computational time and memory requirements; thus enabling PRODYN to operate in real-time.

PRODYN relies extensively on detectors to predict the traffic arrival flows and queues at intersections. PRODYN goes through three main steps; first it simulates a specific intersection for each time step as soon as the intersection controller completes its optimisation over the time horizon; second, it sends the simulation output to each downstream intersection controller; and finally, it uses the output message from upstream controllers at the next time step to predict arrivals. The initial state, the queue length, and the predicted arrivals are used by PRODYN to optimise the intersection performance by minimising the sum of delays over the control horizon period.

PRODYN went through two stages of development: two-level hierarchical control (PRODYN-H) and then decentralised control (PRODYN-D) (Barriere *et al.*, 1986; Head *et al.*, 1992; Henry and Farges, 1989; Farges *et al.*, 1983, 1990). The former offers the best intersection performance; however its applicability is restricted due to the complex computations involved and hence the maximum network size (limited to about 10 intersections). The latter, on the other hand, alleviates the limitations of PRODYN-H. PRODYN-D includes two approaches: PRODYN-D1 with no exchange and PRODYN-D2 with an exchange of information among the intersections.

2.3.3.3 UTOPIA/SPOT

UTOPIA/SPOT (Urban Traffic Optimisation by Integrated Automation/Signal Progression Optimisation Technology) (Mauro and Di Taranto, 1989a, 1989b) is a traffic signal control strategy developed by Mizar Automazione in Turin, Italy. Similar to the local control approach of OPAC and PRODYN, SPOT gathers additional information on vehicle arrivals through communication between single controllers. At an area level, the UTOPIA-module divides the network into subareas (with a common cycle length) and then calculates the optimal control strategies for each subarea.

SPOT is a fully distributed traffic-adaptive signal control system that operates by performing a minimisation of local factors such as delays, stops, excess capacities of links, stops by public or special vehicles, and pedestrian waiting times. With each SPOT update, all SPOT intersections exchange traffic state information every few seconds to achieve progression at the subarea level.

UTOPIA can be implemented without a central computer for small systems of typically six intersections or less. However, for large-scale networks, UTOPIA requires a central PC-based control system to be added to the system. At an area level, UTOPIA provides a mechanism to handle critical situations in the form of two actions that a signal controller may request from adjacent signal controllers: 1) requesting a downstream signal to increase throughput (to cope with congestion and avoid queue spillback) or 2) requesting an upstream controller to decrease demand.

2.3.3.4 RHODES

RHODES (Real-Time, Hierarchical, Optimised, Distributed, and Effective System) (Head *et al.*, 1992; Mirchandani and Head, 2001; Sen and Head, 1997) is a hierarchical control system for intersection and network levels of control that uses predictive optimisation. RHODES's hierarchy is formed of three levels: network-loading level, network flow control level, and intersection control level. At each of the three levels there are two components: an estimation/prediction component and a control component. At the network-loading level the traffic network is loaded with vehicles and the stochastic changes in the current traffic pattern from real-time data are estimated. RHODES then uses this information to proactively predict future platoon sizes at the next level. A platoon prediction logic model is used in the middle level to select the signal timing that optimises the overall flow of vehicles in the network. Network optimisation is also established at this level and its results are used as constraints for the decision made in the next level. At the intersection level, RHODES is responsible for making the final second-by-second decisions regarding traffic signal operation by using two logical sublevels: Link Flow Prediction Logic, and Controlled Optimisation of Phases (COP). The link flow prediction logic uses data from detectors from each upstream intersection to estimate vehicle arrivals at the intersection being optimised. The COP logic uses the information from the network flow problem, in addition to the results from the link prediction logic, to determine whether the current phase should be extended or terminated.

The stochastic traffic equilibrium component at the top of the hierarchy and the model-based traffic predictions at each hierarchical level makes RHODES a proactive system that can easily capture traffic fluctuations and congestion dynamics. On the other hand, the intersection dispatching component at the bottom of the hierarchy is designed to make RHODES reactive, second-by-second, to random fluctuations in traffic and is typically implemented as a distributed control system. However RHODES suffers from the limitations of the 3-GC systems as discussed below.

2.3.3.5 3- GC Summary and Limitations

On-site experiments on PRODYN on a seven intersections network in ZELT (Zone Experimentale et laboratoire de Trafic de Toulouse) revealed a 12% saving in travel time when compared to the Webster fixed timing plans (Henry, 1989).

In Turin, Italy, UTOPIA has been compared against a traffic responsive control strategy and the benefits were estimated as a 35% increase in traffic speed in peak times (Wood, 1993).

OPAC and RHODES were tested in several North American cities/states including: Virginia, Reston, New Jersey, Washington DC, Seattle, Arizona, and Chicago. Field tests and simulation results using CORSIM simulation concluded that OPAC outperforms existing traffic pre-timed or actuated control systems.

In New Jersey, experiments indicated that OPAC significantly reduces stopped delay by 39.7% on major street approaches (southbound Route 18), without affecting the performance of the minor street approach (eastbound Commercial Avenue) (Andrews *et al.*, 1997). At a specific diamond interchange (in Tempe, Arizona) the improvements in the delay using RHODES compared to semi-actuated control were estimated as being up to 48% (Mirchandani and Lucas, 2001).

In summary, 3-GC systems suffer from the following common limitations:

- The unfeasibility of simultaneously handling/optimising several intersections due to the following limitation in the optimisation methods:
    - DP methods require a state transition probability model for the traffic environment which is difficult to obtain
    - the number of states that could represent wide traffic conditions is typically massive. Therefore, DP algorithms are computationally intractable (Sutton and Barto, 1998; Gosavi, 2003) which hinders a straightforward network-wide application of the optimisation method;
- The absence of an accurate traffic modelling framework due to the dynamic and stochastic interaction process of signal control policies/strategies and traffic flow/routing. Most traffic signal control optimisation methods fix the expected routing choices of vehicles and then optimise the signal timings for the resulting traffic pattern. Accurate traffic models should consider drivers' behaviour and variable route choices due to changing traffic conditions; this is the essence of dynamic traffic assignment-based methods (Gartner *et al.*, 1992). Extensive research efforts have been conducted to develop sophisticated models (Friedrich *et al.*, 2003); however the problem persists because: 1) models are not perfect, and equally importantly 2) the performance of controlling strategies depend on such inaccurate models;
- The need for numerous and reliable magnetic loop-based detectors. These detectors are typically placed at specific locations to report traffic data that are indicative of the temporal and spatial variations in traffic flows. These data are often interrupted due to either malfunction or a total failure of the inductive loops, a common limitation from which both 2-GC and 3-GC systems suffer (Boillot *et al.*, 2006);
- The control logic for an area (or arterial) of multiple intersections in 3-GC systems (also in 1-GC and 2-GC) is only meant to achieve *progression* along a certain corridor (e.g. a major artery). In traffic networks without well-defined traffic flow patterns (e.g. inbound in the morning and outbound in the afternoon peak periods), this approach may not be

effective. Most large cities, where business centres are no longer exclusively located downtown, have several locations that serve as attractors of traffic so that no clear patterns exist. In addition, in congested cities "minor" streets have become as important (and critical) as major arterials due to the saturated nature of congested downtown areas. Non-recurrent congestion (e.g. accidents, floods, snow, etc.) may also greatly affect traffic patterns in transportation networks. Therefore, the simple progression approach sought along arterials cannot be effectively applied to a two-dimensional grid-like complex network of intersections while capturing dynamic traffic patterns.

It is therefore clear from the above limitations that there is a need for a fourth generation of traffic signal control that is flexible, robust, scalable, model-free, and computationally efficient in controlling a two-dimensional network of multiple controllers.

2.3.4 Fourth Generation Control (4-GC)-Next Generation

The fourth generation of ATSC maintains the same advantages of 3-GC while addressing the issues that have faced the development of 3-GC (Gartner *et al.*, 1996). The 4-GC systems are principally built on self-learning capabilities that are based on experience under real-time conditions and reasonable computational requirements to be implemented in real-time. 4-GC of traffic systems is still under continuous research and development.

RL is an artificial intelligence technique that has shown promising potential for "self-learning" ATSC (Abdulhai and Kattan, 2003). RL not only achieves as much as DP but also requires less computation and does not need a perfect model of the environment. In addition, RL-based control methods are capable of learning from direct interaction with the environment, and can consequently capture the stochastic variations in traffic flow without the need for model-based traffic prediction.

The basic concept of RL is concerned with an agent (e.g. signalised intersection) interacting with its environment (e.g. traffic network) in a closed-loop system in which the agent acts as the controller of the process (Sutton and Barto, 1998). The agent iteratively observes the *state* of the environment, takes an *action* accordingly, receives a feedback *reward* for the actions taken and adjusts the *policy (control law)* until it converges to the optimal mapping from states to optimal actions (optimal *policy or control law*) that maximises the cumulative reward. Accumulating the maximum reward not only requires the traffic signal control agent to exploit the best-experienced actions, but to also explore new actions to possibly discover better actions in the future. The interaction between the agent and the environment can be viewed as two processes performed repeatedly: a learning process and decision making process. In the learning process, the agent adjusts the policy by updating the value associated with each state-action pair using, in-part, the immediate reward value. In the decision making process, the agent chooses its action by balancing exploration and exploitation using action selection algorithms (e.g. ε -greedy and softmax) (Gosavi, 2003).

MARL is an extension of RL to multiple agents (signalised intersections). The decentralised traffic signal control problem is an excellent testbed for MARL due to the inherent dynamics and stochastic nature of the traffic system (Bazzan, 2009; El-Tantawy and Abdulhai, 2010). The simplest way to extend RL to the MARL is to consider the local state and local action for each agent assuming a stationary environment, and that the agent's policy is the prime factor affecting the environment. However MARL in the traffic environment is associated with some challenging issues because the traffic environment is <u>non-stationary</u> since it includes multiple agents learning <u>concurrently</u>; i.e. the effect of any agent's action on the environment depends on the actions taken by the other agents. <u>Each agent is, therefore, faced with a moving-target learning problem because the best policy changes as the other agents' policies change which accentuates the need for coordination among agents</u>. Coordination can be achieved by considering the joint-state and joint-action for the other agents in the learning process. Moreover, given that all agents are acting simultaneously, the agents' choices of actions <u>must be mutually consistent</u> to achieve their common goal of optimising the signal control problem. Therefore, the agents not only require a <u>coordination mechanism</u> in the learning process, but also in the decision making process to make the optimal decision from the possible joint actions. <u>Agent coordination in this context is not to be confused with conventional traffic signal coordination that maximises green bands, offsets, etc</u>.

Since the focus of this research is on self-learning ATSC (4-GC), in this section we review traffic signal control studies that used RL approaches. As shown in Figure 2-4, the studies are classified according to the coordination approach as follows:

- ***Totally Independent Multi-Agent Reinforcement Learning Control***

In this type of control, agents (at each intersection) act totally independently while performing their tasks by considering: local states, local rewards, and local actions.

- ***Partially Independent Multi-Agent Reinforcement Learning Control***

This type of control extends the first type by following one of two approaches. One approach to take the presence of other agents into account is to extend the state space of the agent to include the state of other agents, while the agent is maximising its local reward. Another approach is to consider the same reward (global reward) or to exchange the reward with other agents. However none of the above is meant to achieve explicit coordination between agents' actions.

- ***Coordinated Multi-Agent Reinforcement Learning Control***

This type of control extends the previous types of control to include an explicit coordination mechanism that coordinates the actions of the individual strategies in real time. The agent is "aware" of the effect of other agents' actions in the network while all agents act simultaneously. In this case the coordination mechanism enables the agents in a group to coherently choose their actions from the optimal joint policy.

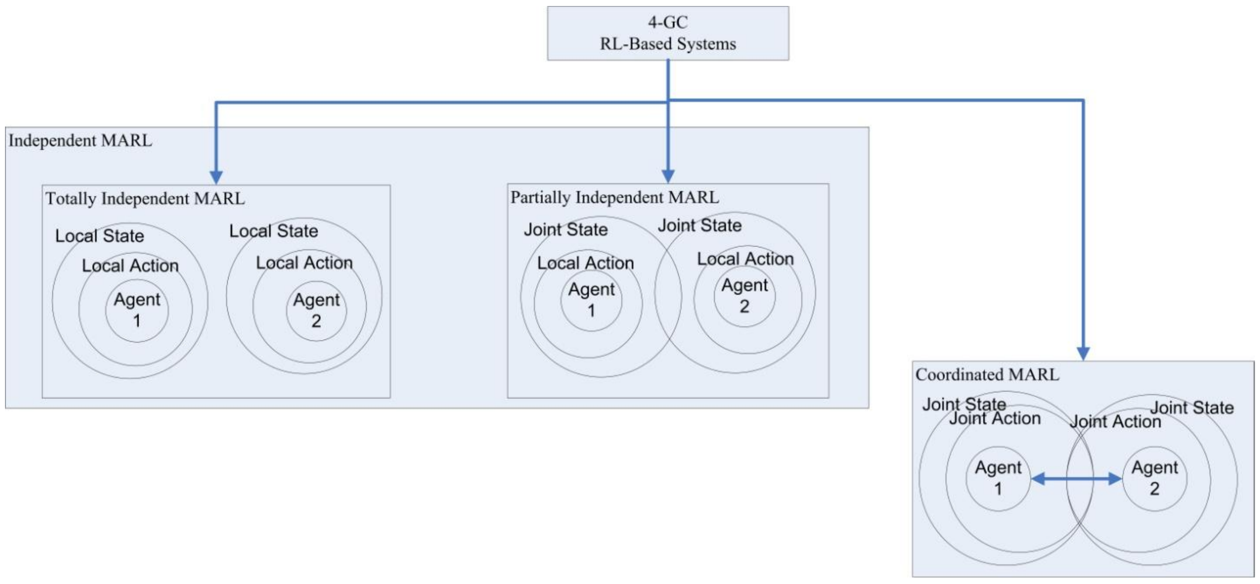


Figure 2-4 Categories of RL-based ATSC Systems

2.3.4.1 Totally Independent Multi-Agent Reinforcement Learning

Thorpe (1997) applied the SARSA RL algorithm to a simulated traffic light control problem using eligibility traces and greedy exploration methods. The objective of the RL agent was to minimise the time required to release a fixed traffic volume into a roadway network. The performance of SARSA was analysed while considering three traffic state representations: 1) vehicle counts, 2) relative distance of vehicles from the intersections, and 3) vehicle counts with signal light duration. In addition four performance measures were calculated: the total number of simulation steps required for all vehicles to reach their destinations, the average vehicle travel time, the total number of stops made by all vehicles, and the average vehicle wait time. The controller was trained on a single intersection after which it was replicated on other intersections and tested on a 4x4 grid network. This means that each controller only locally optimises its intersection. The results showed that the SARSA RL algorithm outperforms the fixed timing plans by reducing the average vehicle waiting time by 29%. The tests conducted with the above state representations showed that the choice of state representation is critical to the success of the SARSA strategy.

Wiering (2000) utilised model-based RL (with state transition models and state transition probabilities) to control traffic-light agents to minimise the waiting time of vehicles in a small grid network. The agents learn a value function that estimates the expected waiting times of vehicles given different settings of traffic lights. In that study, agents correspond to the traffic signals but the learning task is designed such that the state representation is a function of the waiting time for individual vehicles (i.e. vehicle-based state representation), aggregated over all vehicles around the intersection. The network is discretized into a number of lanes and each lane is discretized into possible places for cars, called cells. As a result, the number of states grows with the number of lanes and the number of vehicles occupying each cell in a lane. The author investigated the use of multiple representations of RL in a grid-like network using a simplified discrete event simulator. Three different model-based RL systems were designed: TC-1with no communication sought between traffic lights, TC-2 with communication between cars to estimate the state-transition function of the first car while taking into account the number of cars standing at the next light, and TC-3 which uses the information from TC-2 to compute transition probabilities for all cars. The experimental results showed that RL systems outperform the non-adaptive systems by 22% in waiting time, with the TC-3 model being the best. Although the results achieved by TC-3 are promising, it requires state representation that is dependent on the number of lanes, number of cells and consequently the number of vehicles travelling through the network. Therefore this approach faces the following challenges: 1) the number of states grows intractably with the network size and traffic volume which makes it impractical to implement in medium or large-scale traffic networks; and even for relatively small networks, the number of states will increase exponentially with the increase in the number of vehicles resulting in slow convergence speed, 2) to deploy such a system a loop detector has to places in each cell in each lane to determine whether or not the cell is occupied, which is impractical, 3) the use of a model-based RL approach adds unnecessary complexity compared to using model-free approaches like Q-learning, and 4) the system does not facilitate any explicit coordination among agents. Bakker *et al.* (2005) extended TC-1 by considering the queue propagation from one lane to another in the state representation, but explicit coordination was not considered.

Abdulhai *et al.* (2003) applied a model-free Q-learning technique to a simple two-phase isolated traffic signal in a two-dimensional road network. According to the state information that includes the queue lengths on the four approaches, the agent chooses to either remain in the current phase or to change it with the goal of minimising the average number of waiting vehicles in all approaches. Three different traffic profiles (uniform traffic flows, constant-ratio traffic flows, and variable traffic flow) are tested to evaluate the performance of the Q-learning agent under varying traffic conditions. Q-learning for the isolated traffic-light controller outperformed the pre-timed control scheme for the variable traffic flow case by around 44%. In the uniform and constant profile cases, Q-learning either slightly outperformed or was equal to the pre-timed control, which was as expected.

Camponogara and Kraus Jr. (2003) formulated the traffic signal control problem as a distributed SG in which agents employ a distributed Q-learning algorithm. A small network of two intersections with limited capacity roads is modelled. To account for traffic dynamics, traffic conditions were assumed to vary by applying the following policies: 1) uniformly random policy (assigns the same probability to all actions available to an agent), 2) Q-learning implemented by agent-1 (agent-1 applies the Q-learning algorithm to control traffic signals at its intersection while agent-2 follows the uniformly random policy), and 3) Q-learning implemented by agent-2. Results showed that policy 2 outperformed policy 1 with an 18% reduction in average waiting time. When testing policy 3 (i.e. both agents run Q-learning) a 43% reduction in waiting time was achieved compared to policy 1. Although formulating the problem as a distributed SG, the study did not consider the convergence of the agents to an equilibrium joint policy. They used the typical Q-learning algorithm (called "distributed Q-learning" in their paper) to reach a set of distributed control policies in which each agent independently computes its own optimal action given its local state.

De Oliveira *et al.* (2006) proposed a RL method called Reinforcement Learning with Context Detection (RL-CD) to control traffic lights at isolated junctions as an appealing approach for dealing with non-stationarity and handling stochastic traffic patterns. In this study the following assumptions were made: 1) environmental changes are restricted to a small number of contexts (traffic patterns) which are stationary with distinct dynamics, 2) environmental context changes are independent of the agent's actions, and 3) context changes are relatively infrequent. Once a new context is detected, an optimal policy using model-based RL methods is assigned to map the traffic conditions to the signal plans. Experiments were designed using the ITSUMO tool that implements the Nagel-Schreckenberg microsimulation model (a cellular automaton-model). This mechanism was tested on a network of nine traffic signals, and the empirical results showed that RL-CD is more efficient than the classical Q-learning techniques and that the real traffic states must be discretized in a finer-grained representation. However, none of the learning methods was able to cope with the oversaturated network cases. Also, the environmental context changes considered in this study are independent of the agent's actions; consequently the approach remains a single-agent based learning method where each agent selects its action independently and disregards the fact that the environment is non-stationary (depends not only on the agent's actions but also on the policies implemented by other agents).

2.3.4.2 Partially Independent Multi-Agent Reinforcement Learning

Richter *et al.* (2007) used the Natural Actor Critic (NAC) algorithm, in which four algorithms are used: policy gradient, natural gradient, TD, and least-square TD. In their simplified simulation, five scenarios were tested and each junction (intersection) on the grid had four phases. Agents used local rewards but with additional observations from neighbouring intersections (e.g. traffic flow). The following assumptions are considered to simplify the problem: all vehicles move at uniform speed, and interactions between cars are ignored within one road segment. NAC outperformed SAT (adaptive controller inspired by SCATS) with a 20% reduction in average network travel time in a 10×10 junction grid simulation network.

Salkham *et al.* (2008) utilised Collaborative Reinforcement Learning (CRL) to provide adaptive and efficient urban traffic control. In their study, each signalised junction utilised a CRL-based traffic agent that follows an adaptive phase cycle, namely Adaptive Round Robin (ARR), and observed the local traffic patterns from local vehicle location data. Q-learning was used and a common advertisement strategy was utilised to allow for a given ARR-CRL agent to exchange rewards with its neighbours. The authors applied CRL-based ARR (ARR-CRL) and RL-based ARR (ARR-RL) controllers to a relatively large-scale urban traffic control optimisation scheme (in Dublin's inner-city centre) of 64 signalised junctions. The state-action space was rather (overly) simple and very time-coarse. Each agent decided the phase splits every two cycles, and the state of each phase was represented by either busy/not busy, which lacked the capturing of the rapid dynamics of congestion. The average waiting time savings of ARR-CRL and ARR-RL were 57% and 45% compared to SAT control, respectively. Although the average reduction in waiting time per vehicle achieved by both ARR-RL and ARR-CRL scenarios is significant, coordination between the agents' actions is still largely missing.

Arel *et al.* (2010) considered a five-intersection traffic network as a testbed in which each agent represented an intersection. Two types of agent, a central agent and an outbound agent, were employed. The outbound agents scheduled traffic signals by following the longest-queue first (LQF) algorithm and collaborated with the central agent by exchanging their local states. The central agent learned a value function driven by its local and neighbours' traffic conditions. Given the large state space that was spanned under these assumptions, their proposed methodology utilised the Q-learning algorithm with the function approximation method to store the value function. Experimental results clearly demonstrated a 23% saving in average delay per vehicle when using the multi-agent RL-based control over LQF isolated intersection control.

Medina and Benekohal (2012) used a combination of Q-learning and approximate dynamic programming (ADP) algorithm approaches to control the traffic signals in real-time, based on acyclic approach. At each intersection, the learning agent not only considers its local state but also the congestion levels that are communicated between neighbouring intersections. A VISSIM simulation model was used to test the efficiency of the Q-learning and ADP agents against TRANSYT7F. Q-learning and ADP showed a 13% lower average delay and 10–12% increased average system throughput compared to TRANSYT7F. Similar to the above approaches, this implementation of Q-learning and ADP did not consider any explicit mechanism for coordination among agents' actions and states, and the only communication allowed between intersections was limited to a single bit of information representing potential downstream blockages.

2.3.4.3 Coordinated Multi-Agent Reinforcement Learning

Kuyer *et al.* (2008) and Bakker *et al.* (2010) extended the RL approach proposed by Wiering (2000) to include an explicit direct coordination between neighbouring traffic lights using coordination graphs. Coordination graphs were used to describe the dependencies between the directly connected agents. The max-plus algorithm is used to estimate the optimal joint action by sending locally optimised messages among connected agents. They compared their control system with the TC-1 (Traffic Controller 1) developed by Wiering (2000) and TC-SBC (Traffic Controller with State Bit for Congestion) extension of Bakker *et al.* (2005). The experiments were designed to test the following hypothesis: under highly-saturated conditions, coordination is beneficial when the amount of local traffic is small. Local traffic is defined by the number of vehicles that are interacting with just one learning agent (i.e. crossing a single intersection and then exiting the network). These experiments were tested on three hypothetical grid-networks of different sizes, ranging from 3 to 17 nodes.

The results demonstrated a strong correlation between the amount of local traffic and the value of coordinated learning. The max-plus method consistently outperformed the non-coordinated methods (TC-1 and TC-SBC) under highly-saturated conditions when the amount of local traffic is limited. Even when there is high amount of local traffic, the max-plus method achieved the same performance as TC-1 and TC-SBC; however it learned more slowly.

To the best of my knowledge, the above approach was the first RL-based approach that considered explicit coordination between agents. However, the algorithm used in this study, the max-plus algorithm, is computationally demanding as it requires negotiations between the agents to coordinate their actions through direct communication, without incorporating any social conventions or roles that can help reduce the computation time. Due to the fact that the system performs under time constraints, this forces the agents to report their current best action at any time even if the action found so far may be sub-optimal. This system also faces the same state representation challenges as the approach proposed by Wiering (2000) (refer to section 2.3.4.1)

2.3.4.4 4-GC Summary and Limitations

To summarise the studies reviewed in the previous section, Table 2-2 is presented to compare and contrast these studies. The rows represent the method or approach used to solve the problem, while the columns represent the comparison criteria. The criteria include: TD learning method, definition of each RL element (state, action (phasing sequence), reward), exploration method, and frequency of action selection), size of the problem (represented by the number of traffic elements involved), and control coordination level (described below).

It is clear that the efforts of the researchers are important accomplishments in RL-based traffic control. However some gaps still exist. The following discussion summarises the major challenges and gaps in the existing literature:

- Different RL algorithms, and more specifically TD algorithms, have been investigated separately in the literature without discussing the appropriateness of each TD method in solving the traffic signal control problem;
- Each study considered a specific RL design (state, action, and reward definitions) without providing quantitative or qualitative justification for the selected parameters;
- In most of these studies the algorithms have been applied to very simplistic scenarios with strong assumptions related to traffic behaviour and simplified simulation environments (Abdulhai *et al.*, 2003; Arel *et al.*, 2010; Camponogara and Kraus Jr., 2003; De Oliveira *et al.*, 2006; Richter *et al.*, 2007), and/or assuming hypothetical traffic flows (Abdulhai *et al.*, 2003; Arel *et al.*, 2010; Camponogara and Kraus Jr., 2003; De Oliveira *et al.*, 2006; Richter *et al.*, 2007; Shoufeng *et al.*, 2008; Thorpe, 1997; Wiering, 2000) which do not mimic the reality in traffic networks;
- As illustrated in Table 2-2, some of the previous studies utilised RL in traffic signal control by using totally independent learning agents. These studies were primarily conducted to either control an isolated traffic element or multiple controller, but with no action coordination between the traffic controllers (Abdulhai *et al.*, 2003; Bakker *et al.*, 2005; Camponogara and Kraus Jr., 2003; De Oliveira *et al.*, 2006; Thorpe, 1997; Wiering, 2000);
- Some studies accounted for other learning agents in the network by considering state information exchange or reward exchange between neighbours (Arel *et al.*, 2010; Medina and Benekohal, 2012; Richter *et al.*, 2007; Salkham *et al.*, 2008). It is noteworthy that in

these approaches the behaviour of individual agents is still not explicitly coordinated since each agent decides its optimal policy independently. Therefore, the agents may select individual actions that are locally optimal but collectively produce inefficient suboptimal solutions for the network (Busoniu *et al.*, 2008);

- The performance of these approaches is not guaranteed in oversaturated conditions where the effect of upstream and downstream intersections becomes crucial;
- To the best of my knowledge, only one approach considered explicit coordination between agents (Bakker *et al.*, 2010; Kuyer *et al.*, 2008); however this approach is fairly complex and relied primarily on the occupancy detection in each section of the road, which makes field deployment impractical;
- The absence of an efficient explicit coordination mechanism for a relatively large-scale two-dimensional network of controllers.

Table 2-2 Summary of 4-GC Systems

MARL	Criterion / Study	TD Learning Method	RL Elements			Exploration Method	Action Frequency	Problem Size	Coordination
			State	Phasing Sequence	Reward/ Penalty				
Totally Independent MARL	Thorpe (1997)	SARSA and SARSA(λ)	Vehicle counts in the links leading to intersection	Fixed	(Penalty) Time required to release a fixed volume of traffic through a road network	ε-greedy	1 sec	16 hypothetical	No
	Wiering (2000)	Model-based Q-learning	Number and location of vehicles in the links leading to intersection	Variable	(Penalty) Total delay incurred between successive decision points	ε-greedy	1 sec	6 hypothetical	No
	Abdulhai et al. (2003)	Q-learning	Queue lengths in the links leading to intersection	Fixed	(Penalty) Total delay incurred between successive decision points	softmax	1 sec	1 hypothetical	No

	Camponogara and Kraus Jr. (2003)	Q-learning	Number and location of vehicles on the links leading to intersection	Fixed	(Penalty) Number of vehicles waiting at intersection	ε-greedy	Cycle length	2 hypothetical	No
	De Oliveira *et al.* (2006)	Modified Q-learning	Vehicle counts in the links leading to intersection	Fixed	(Penalty) Squared sum of the incoming links' queues	ε-greedy	2 cycle length	9 hypothetical	No
Partially Independent MARL	Richter *et al.* (2007)	Modified Q-learning	Current cycle length, current phase durations, and detectors status	Variable	(Reward) Number of cars that entered intersection over the last time step	softmax	Cycle length	100 hypothetical	Consider neighbours' states
	Salkham *et al.* (2008)	Modified Q-learning	Vehicle counts in the links leading to intersection	Fixed	(Reward) difference Between the number of vehicles that cleared the junction during the last time step and the number of vehicles that are still waiting	softmax	3 cycle length	64 realistic	Consider neighbours' reward
	Arel *et al.* (2010)	Q-learning	Relative delay at each lane leading to intersection	Variable	(Reward) Savings in the delay in the last time step	ε -greedy	20 sec	5 hypothetical	Consider neighbours' states

	Medina and Benekohal (2012)	Q-learning and Approximate DP	Queues at all intersection's directions and indicators of potential downstream blockages	Variable	(Reward) a Combination of the number of vehicles being served by the green light (with a positive sign), the number of vehicles waiting to be served in the remaining approaches (with a negative sign), and a series of penalties	softmax	2 sec	20 realistic	Consider neighbours' states
Coordinated MARL	Isa *et al.* (2006) Kuyer *et al.* (2008) Bakker *et al.* (2010)	Modified Q-learning	Vehicle-based representation (dimensionality increases with no. of cars)	Variable	(Penalty) Increase in queue length	ε-greedy	1 sec	15 hypothetical	Explicit coordination

3 Methodology Background: Single Agent and Multi-Agent Stochastic Control Problem and Reinforcement Learning Solution

In order to solve the ATSC problem in general, and to particularly facilitate the understanding of the proposed approach in this research, the problem first has to be properly classified and formulated. Due to the stochastic nature of the traffic environment, ATSC can be classified as a stochastic control problem. This chapter provides the relevant theoretical foundation for this type of control problem from single agent and multi-agent perspectives. It also provides a review of RL as a promising solution algorithm for the stochastic control problem. Figure 3-1 presents the roadmap of this chapter.

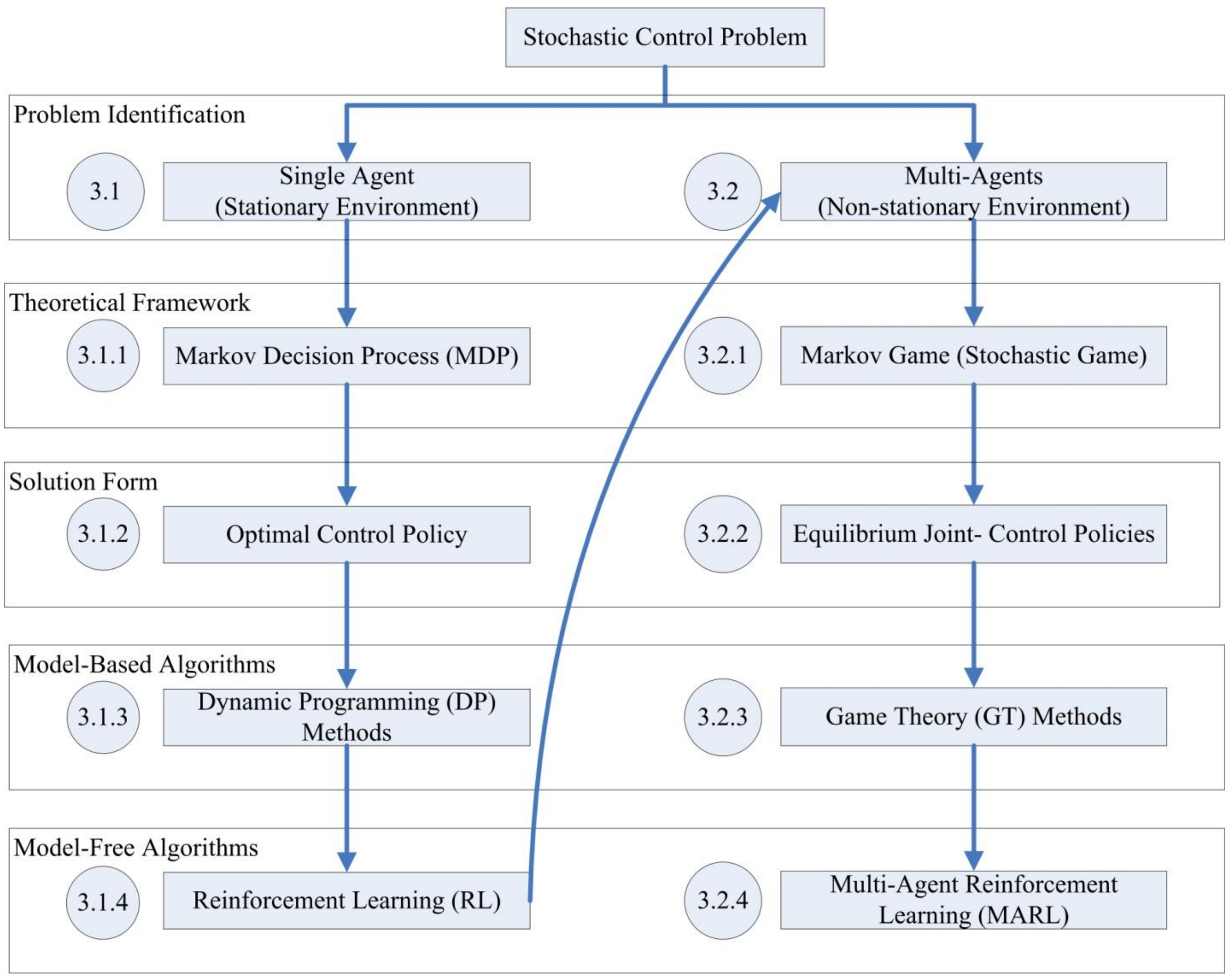

Stochastic Control Problem
Problem Identification
3.1
Single Agent (Stationary Environment)
3.2
Multi-Agents (Non-stationary Environment)
Theoretical Framework
3.1.1
Markov Decision Process (MDP)
3.2.1
Markov Game (Stochastic Game)
Solution Form
3.1.2
Optimal Control Policy
3.2.2
Equilibrium Joint- Control Policies
Model-Based Algorithms
3.1.3
Dynamic Programming (DP) Methods
3.2.3
Game Theory (GT) Methods
Model-Free Algorithms
3.1.4
Reinforcement Learning (RL)
3.2.4
Multi-Agent Reinforcement Learning (MARL)

Figure 3-1 Chapter 3 Roadmap

3.1 Stochastic Single Agent Control Problem

In this section the relevant background on single-agent RL is briefly introduced.

3.1.1 Theoretical Framework: Markov Decision Process

The framework for the single-agent based stochastic control problem is a Markov Decision Process (MDP). MDP is a subset of the Markov process, which is also a subset of stochastic processes (Figure 3-2).

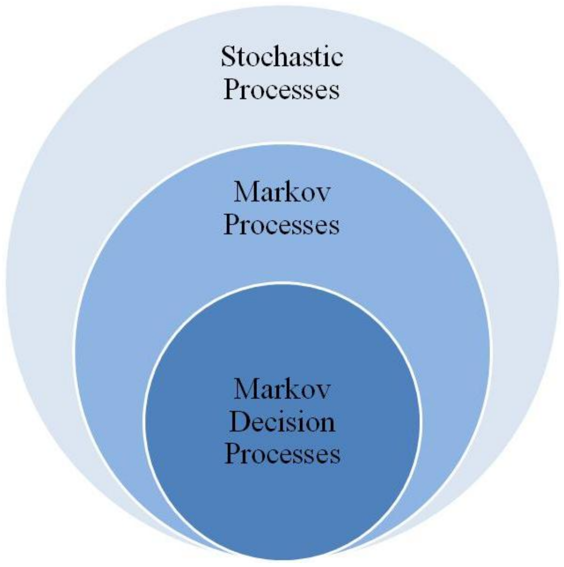


Figure 3-2 Single Agent-Based Stochastic Control Framework

3.1.1.1 Stochastic Processes

A stochastic process (Parzen, 1967) is a collection of time-ordered defined random variables. At different points in time, the defined random variable is expected to take different values. The variable can take any possible value that belongs to its “state space” (or sample space). A stochastic process can be written as $S^k, k = \{0, 1, 2, \dots\}$ where S is the random variable observed; this is referred to as a “discrete-time” stochastic process, a collection of random variable values at discrete points in time. The value of the random variable at a given time k represents the “state” of the system at that time.

Example 3.1: Stochastic Queuing Process at Signalised Intersections
Consider an observer counting the number of cars stopping at an intersection stop lines per minute; the data collected by the observer over a period of time represents a stochastic process. In this example, the

random variable is the number of cars. The observed number of cars can take different values at different times. The state space of the random variable includes all positive integer values, and zeros if any.

3.1.1.2 Markov Process

When the relationship among the random variable values in a stochastic process satisfies the Markov property, the stochastic process is then called a "Markov process" (Ethier and Kurtz, 1986).

The Markov property is satisfied if the conditional probability distribution of the future states of the process (i.e. random variable values at future time instances), given the current state and all past states, only depends on the current state and not on any past state. In other words, the current state of the system is necessary and sufficient to predict the future state of the process without the need for knowing the past state(s) of the system. To illustrate, at any time k, if the following information is available about the system state at time k (S^k), at time $k-1$ (S^{k-1}), at time $k-2$ (S^{k-2}),…, and at time $k=0$ ($S^0 = s^0$). When the only information required to define the future state of the system at time $k+1$ (S^{k+1}) is the current state of the system (S^k), the stochastic process is called a Markov process, or said to have the Markovian property. In mathematical terms, the Markovian property is stated as in Equation 3-1

$$P(S^{k+1} = s^{k+1} | S^k = s^k, S^{k-1} = s^{k-1}, S^{k-2} = s^{k-2}, \dots, S^0 = s^0) = P(S^{k+1} = s^{k+1} | S^k = s^k) \tag{3-1}$$

Example 3.2: Markov Queuing Process at Signalised Intersections

At a signalised intersection, a stochastic process can represent the change in the number of cars waiting in a queue over time (Figure 3-3). This number changes when a new car joins the queue or when a car leaves the queue. At any given time, the system state is described by the number of cars in the queue. The prediction of the number of cars in a queue at the next time instance will only *depend* on the present system state (i.e. the number of current cars in the queue) and is *independent* of the past temporal trajectory of the queue-length.

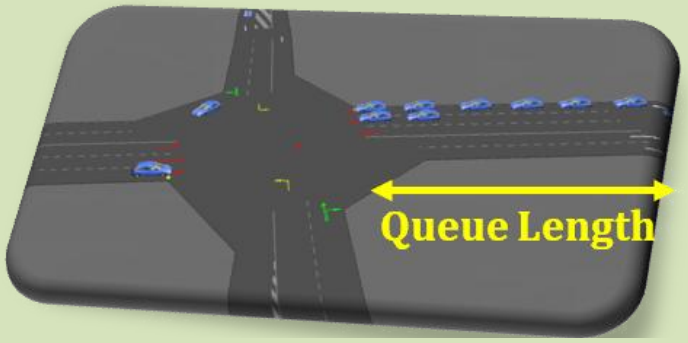


Figure 3-3 Waiting Cars at Signalised Intersections

3.1.1.3 Markov Decision Processes (MDP)

In Markov processes, the outcome of the stochastic process is not controlled but rather observed, hence it is totally random. Markov processes with outcomes that are partly random and partly under the control of a decision maker are called MDPs (Puterman, 1994). MDPs provide a mathematical framework for modelling the decision making process in the Markov processes.

In a MDP a decision-maker (agent) partly controls the transition probabilities from one system state to another through deliberate action choices at discrete points in time. At each state there are several actions from which the decision maker must choose. In MDPs there is a reward associated with the decision making process. Rewards can be earned for each state visited, or for each state-action pair implemented. The decision-maker in a MDP typically has a goal of optimising some function of the cumulative rewards. In mathematical terms, the MDP is stated in Definition 3.1:

Definition 3.1: Markov Decision Process
Markov Decision Process is defined by a quadruple $< S, A, R, T >$, where • S is a set of states • A is a set of actions • R is a reward function $S \times A \to \mathbb{R}$ • T is a transition function $S \times A \times S \to [0, 1]$

The transition function, $T(s^k, a^k, s^{k+1}) \in [0, 1]$, denotes the probability of making a transition from state s^k to state s^{k+1} using action a^k. In other words, it defines the probability of the system "state" taking a possible value from the "state space", given a pre-defined present state and action. The reward function, $r(s^k, a^k)$ defines the reward received when selecting an action a^k at the given state s^k.

3.1.2 Solution Form: Optimal Control Policy

MDPs is a stochastic optimisation problem where the goal is to find the optimal policy (mapping between states and actions) $\pi: S \to A$, which determines the agent's optimal actions at each state so as to maximise the discounted future reward, with discount factor γ, where $\gamma \in [0,1]$.

Example 3.3: Signalised Intersection Control Markov Decision Process

Consider a two-phase signalised intersection of a major street with a minor street. The queue length at the minor street at a specific point in time t is not known *a priori*; the queue length at the minor street is therefore *random*. If we collect observations about the queue length at the minor street at various points in time, this is considered a stochastic process.

In this stochastic process, the random variable represents the queue length on the minor street, S. Queue length level is considered either "low" (s_1) if the number of cars in the queue is, for example, less than or equal to 5 and "high" (s_2) if the queue length is more than 5. The state space (or sample space) for the random variable has two elements: $S = \{s_1, s_2\}$. At each state two actions $A = \{a_1, a_2\}$ are possible, where "a_1" means extending the green time of the major street phase for the next time interval (e.g. 30 sec) and "a_2" means switching to the minor street phase for the corresponding minimum green time (e.g. 15 sec) and then turning the green to the major street again for the corresponding minimum green time (e.g. 15 sec). The reward function is defined as the negative summation of squared queue lengths, which aims to minimise and balance the queue lengths across the approaches. The objective is to find the control policy that maximises the expected cumulative reward.

At $k = 0$, let $s^0 = s_1$; this means that queue length level was low at the beginning. After observing the process for time T, the set of transition probabilities are calculated to describe the observed transition of the queue length over time. T^1 and T^2 are the transition probability matrices (TPMs) with dimensions $\mathrm{S} \times \mathrm{S}$ associated with action a_1 and a_2, respectively:

$$T^1 = \begin{bmatrix} & s_1 & s_2 \\ s_1 & 0.5 & 0.5 \\ s_2 & 0.1 & 0.9 \end{bmatrix} \qquad T^2 = \begin{bmatrix} & s_1 & s_2 \\ s_1 & 0.9 & 0.1 \\ s_2 & 0.7 & 0.3 \end{bmatrix}$$

It is conventional to represent TPMs by a directed graph, where the edges are labelled by the probabilities of going from one state to the other states (Figure 3-4).

Similarly, R^1 and R^2 are the reward matrices associated with action a_1 and a_2, respectively:

$$R^1 = \begin{bmatrix} & s_1 & s_2 \\ s_1 & -1 & -16 \\ s_2 & -10 & -50 \end{bmatrix} \qquad R^2 = \begin{bmatrix} & s_1 & s_2 \\ s_1 & -10 & -50 \\ s_2 & -16 & -1 \end{bmatrix}$$

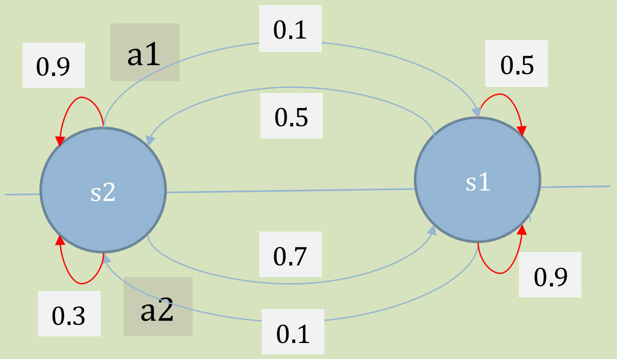


Figure 3-4 Transition Probability Matrices Directed Graph

3.1.3 Dynamic Programming

As mentioned previously, the goal of MDPs is to find the optimal control policy that maximises the time-cumulative reward. The focus here will be on finding the optimal policies for the infinite-horizon discounted tasks (i.e. continual process-control task), which is the case of the traffic signal control task. This is in contrast to episodic tasks (e.g. playing a win or lose game that terminates after a number of steps). The expected cumulative discounted reward at state s following policy π is defined as the value function of the state $V^{\pi}(s)$. For finite MDP (MDP with finite set of states (S) and actions (A)), $V^{\pi}(s)$ is defined as in Equation 3-2;

$$V^{\pi}(s) = V^{\pi}(s^k = s) = E_{\pi}\{\textstyle\sum_{k=0}^{\infty} \gamma^{k-1} r^k \,|s^k = s\} \qquad (3\text{-}2)$$

The discount rate, γ where $0 \leq \gamma \leq 1$, determines the present value of future rewards: a reward received k time steps in the future is only worth γ^{k-1} times of what it would worth if it was received immediately. To illustrate, in the traffic signal control if decisions are taken every 10 sec, γ =0.8 means that discounted future rewards are considered up to 30 time-steps (5 min) after which the effect of the future reward becomes negligible (γ^{k-1}<0.001).

$V^{\pi}(s)$ can also be expressed in terms of state transition probabilities. Equation 3-2 can then be rewritten in a recursive fashion as in Equation 3-3:

$$V^{\pi}(s) = \textstyle\sum_{s^{k+1} \in S} T(s^k, a^k, s^{k+1}) \, [r(s^k, a^k, s^{k+1}) + \gamma V^{\pi}(s^{k+1})] \qquad (3\text{-}3)$$

The last equation is called the Bellman equation for discounted reward; it expresses the relationship between the value of a state and the value of its successor state.

The optimal policy, denoted π^*, is the policy that has the optimal state-value function, V^*, denoted and defined as in Equation 3-4:

$$V^*(s) = \max_{a \in A} \sum_{s^{k+1} \in S} T\left(s, a, s^{k+1}\right) \left[r\left(s, a, s^{k+1}\right) + \gamma V^*\left(s^{k+1}\right)\right] \qquad (3\text{-}4)$$

While Equation 3-4 is defined as the Bellman optimality equation for state-value functions, Equation 3-5 represents the corresponding optimal policies.

$$\pi^*(s) \in arg \max_{a \in A} \sum_{s^{k+1} \in S} T\left(s, a, s^{k+1}\right) \left[r\left(s, a, s^{k+1}\right) + \gamma V^*\left(s^{k+1}\right)\right] \qquad (3\text{-}5)$$

DP is widely considered the only feasible way of solving general stochastic optimal control problem (Bertsekas, 1976, 2007). Value iteration is one DP technique that can be used to find the optimal policy. The value iteration method is an iterative method that estimates the optimal state-value function by simply turning the Bellman optimality equation into an update rule of the value function until it converges with the optimal values V^*. The value iteration algorithm is presented as Algorithm 3.1 (Bertsekas, 1976, 2007).

Algorithm 3.1: DP Value Iteration

```
Initialise $V^0(s)\ \forall s \in S$
Repeat
    For all $s \in S$ do
        For all $a \in A$ do
            $V^k(s) = \max_a \sum_{s^{k+1} \in S} T\left(s, a, s^{k+1}\right) \left[r\left(s, a, s^{k+1}\right) + \gamma V^{k-1}\left(s^{k+1}\right)\right]$
        End for
    End for
Until $\left|V^k(s) - V^{k-1}(s)\right| < \delta \quad \forall s \in S$
```

Example 3.4: Signalised Intersection Control Using Dynamic Programming

This example shows the solution of Example 3.3 using the DP value iteration algorithm; assuming $V^0(s_1) = V^0(s_2) = 0$ and the objective is to converge to the optimal control policy to accuracy $\delta = 0.1$

	Let $$V^0(s_1) = V^0(s_2) = 0, \qquad \delta = 0.1$$ $V^k(s_i) = \max_a [\sum_{j=1}^{2} T\,(s_i, a, s_j) r(s_i, a, s_j) + 0.8 \sum_{j=1}^{2} T\,(s_i, a, s_j)\, V^{k-1}(s_i)]$
Iteration 1	$V^1(s_1) = \max_a [\,(-1 * 0.5 - 16 * 0.5), (-10 * 0.9 - 50 * 0.1)] = \max[-8.5, -14\,]$ $$V^1(s_1) = -8.5$$ Similarly $$V^1(s_2) = -11.5$$
Iteration 2	Similarly $$V^2(s_1) = \max_a \big[[(-1 * 0.5 - 16 * 0.5) + 0.8 * (-8.5 * 0.5 - 11.5 * 0.5)], [(-10 * 0.9 - 50 * 0.1) + 0.8 * (-8.5 * 0.9 - 11.5 * 0.1)]\big] = -16.5$$ $$V^2(s_2) = -19.02$$
……	
Iteration 19	Similarly $$V^{19}(s_1) = -47.22$$ $$V^{19}(s_2) = -49.81$$
Iteration 20	Similarly $$V^{20}(s_1) = -47.32$$ $$V^{20}(s_2) = -49.89$$
	The optimal policy ($\varepsilon = 0.1$) $$\pi(s_1) = \arg\max_a [(-8.5 + 0.8\,[0.5 * -47.22 - 0.5 * 49.81\,], (-11.5 + 0.8\,[0.9 * \ -47.22 - \ 0.1 * \ 49.81] = \arg\max_a [(-47.32), (-51.99)]$$ $$\pi(s_1) = a_1$$ Similarly $$\pi(s_2) = a_2$$ The optimal control policy is $$\pi^*(s_1) = a_1$$ $$\pi^*(s_2) = a_2$$

In order to balance the queue lengths in this intersection, the following control policy has to be followed;

- If the queue length on the minor street is "low" (s_1), the optimal action is to extend the green time of the major street phase for the next time interval (e.g. 30 sec) (a_1)

- If the queue length on the minor street is “high” (s_2), the optimal action is to switch to the minor street phase for the corresponding minimum green time (e.g. 15 sec) and then turn the green again to the major street again for the corresponding minimum green time (e.g. 15 sec) (a_2).

As discussed in Chapter 2 (section 2.3.3.5) solving a real-time optimal stochastic control problem using DP is extremely challenging due to two main limitations associated with DP (Sutton and Barto, 1998):

- Curse of modelling: DP assumes the existence of a perfect model for the environment represented by the matrices of transition probabilities and rewards, which is not the case for non-trivial traffic networks;
- Curse of dimensionality: DP may not be practical for large-scale problems because it involves computations over the entire states of the MDP, in each iteration, and also it requires a large amount of memory to store large matrices (transition probabilities and rewards).

To illustrate, assume that the number of states is |S| and the number of actions is |A|, it is shown from Example 3.4 that the value iteration method required 20 iterations to converge, while in each iteration the values of all states are computed (|S|=2 computations). To calculate the value of each state, |A| (=2) values would be compared to find the maximum (|A|=2 comparisons). Therefore, it is clear that if the state set is very large, then even a single iteration can be prohibitively expensive computationally.

Policy iteration is another DP technique that can be used to find the optimal policy (Sutton and Barto, 1998) (refer to Appendix 2).

3.1.4 Reinforcement Learning

RL methods attempt to achieve as much as the DP, but with less computation and without assuming a perfect pre-defined model (the transition probabilities and rewards) of the MDP (the environment) (Sutton and Barto, 1998; Gosavi, 2003). The basic concept of RL is that the decision maker (the agent) must interact with the environment to learn these probabilities. Hence the values of all states and actions in RL are not solved systematically as in DP; rather the states and actions are experienced irregularly. The next section discusses the transition from DP to RL as a simulation-based optimisation approach.

3.1.4.1 From Dynamic Programming to Reinforcement Learning: Simulation-Based Optimisation

Similar to state-value function, state-action pair value $Q(s,a)$, also called *Q-Value* is defined as the expected cumulative discounted reward at state s^k taking action a^k as defined in Equation 3-6:

$$Q(s^k,a^k)=\sum_{s^{t+1}\in S} T(s^k,a^k,s^{k+1})\,[r(s^k,a^k,s^{k+1})+\gamma V^{\pi^*}(s^{k+1})] \qquad \textit{(3-6)}$$

From Bellman optimality (Equation 3-4) it is shown that:

$$V^{\pi^*}(s^{k+1})=max_{a^{k+1}\in A}\, Q(s^{k+1},a^{k+1}) \qquad \textit{(3-7)}$$

Substituting Equation 3-7 in Equation 3-6 results in the following equation

$$Q(s^k,a^k)=\sum_{s^{t+1}\in S} T(s^k,a^k,s^{k+1})\,[r(s^k,a^k,s^{k+1})+\gamma\, max_{a^{k+1}\in A}\, Q(s^{k+1},a^{k+1})\,] \qquad \textit{(3-8)}$$

Equation 3-8 represents the expected value of a random variable that is $[r(s^k,a^k,s^{k+1})+\gamma\,\max_{a\in A} Q(s^{k+1},a)\,]$.

Given that the random variable is a function of the immediate reward $r(s^k,a^k,s^{k+1})$ and the next state s^{k+1}, samples can be obtained from the interaction with the environment (e.g. within a traffic simulator in this case, or in the field after initial deployment).

The Robbins-Monro algorithm is an algorithm that estimates the mean of a random variable, X, from its samples, y (Gosavi, 2003).

$$X^k=(1-\alpha^k)X^{k-1}+\alpha^k y^{k-1} \qquad \textit{(3-9)}$$

where $\alpha^k=1/k$ is the step-size (or learning rate) at time k.

Assuming that $X=Q(s,a)=E(y)$ where $y=[r(s^k,a^k,s^{k+1})+\gamma\,\max_{a^{k+1}\in A} Q(s^{k+1},a^{k+1})\,]$,

substituting in Equation 3-8,

$$Q^k(s^k,a^k)=(1-\alpha^k)Q^{k-1}(s^k,a^k)+\alpha^k[r(s^k,a^k,s^{k+1})+\gamma\, max_{a^{k+1}\in A}\, Q(s^{k+1},a^{k+1})\,] \qquad \textit{(3-10)}$$

$$Q^k(s^k,a^k)=Q^{k-1}(s^k,a^k)+\alpha^k[r(s^k,a^k,s^{k+1})+\gamma\, max_{a^{k+1}\in A}\, Q(s^{k+1},a^{k+1})\;-Q^{k-1}(s^k,a^k)\,] \textit{(3-11)}$$

Equation 3-11 is the updating rule for one of the well-known RL approaches called Q-learning, which will be explained in detail in section 3.1.4.2.1.2.

3.1.4.2 *Reinforcement Learning Methods: Temporal Difference Algorithms*

Numerous RL algorithms have been investigated in the RL literature (Sutton and Barto, 1998; Gosavi, 2003); the most relevant set of algorithms to this study are the on-line value iteration with discounted reward algorithms which are called TD learning algorithms. TD algorithms are the general class of algorithms that include Q-learning (Equation 3-11). Similar to the DP value iteration algorithm, TD methods estimate the state-action value functions;

$$Q^k(s^k, a^k) = Q^{k-1}(s^k, a^k) + \alpha^k\left[r^k + \gamma Q^{k-1}(s^{k+1}, a^{k+1}) - Q^{k-1}(s^k, a^k)\right] \tag{3-12}$$

At time k, the value of state s^k, $V^k(s^k)$, is updated using the observed reward r^k and the estimate $Q^{k-1}(s^{k+1}, a^{k+1})$. Therefore TD methods can learn directly from experience without a dynamic model of the environment. In addition, TD methods update the estimates of the state values immediately after visiting the state. The TD notion emerged from the fact that the value function is estimated based on the *difference* between *temporally* successive estimations, which is called the TD error, denoted TD^k.

$$TD^k(s) = \left[\,(r^k + \gamma Q^{k-1}(s^{k+1}, a^{k+1})) - Q^{k-1}(s^k, a^k)\right] \tag{3-13}$$

Equation 3-13 represents the simplest form of TD known as TD(0). Two types of TD algorithm are presented in the literature, namely TD(0) and Eligibility Traces TD(λ) (Figure 3-5).

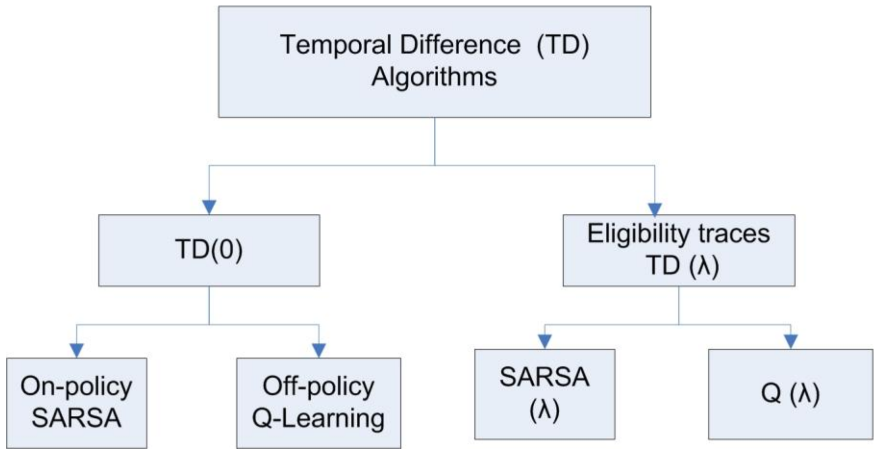

Temporal Difference (TD) Algorithms
TD(0)
Eligibility traces TD (λ)
On-policy SARSA
Off-policy Q-Learning
SARSA (λ)
Q (λ)

Figure 3-5 Temporal Difference Algorithms

3.1.4.2.1 *TD(0)*

TD(0), also known as 1-step TD, resembles the on-line form of DP value iteration, in which the agent looks ahead one step in time and updates the value functions based on the immediate reward r^k.

3.1.4.2.1.1 SARSA: On-Policy Algorithm

In the SARSA algorithm the estimates of the state-action pair's values are conducted strictly based on experience, which is represented by the quintuple ($s^k,\ a^k,\ r^k,\ s^{k+1},\ a^{k+1}$). The state-action value functions are updated after executing actions that are selected by the current policy, i.e. a policy that does not always select the greedy action (the action that has the highest value) (Algorithm 3.2). Therefore, the SARSA algorithm is considered an on-policy TD method since it is learning and improving the same policy that is followed in selecting the actions.

Algorithm 3.2: SARSA

```
Initialise $Q^0(s,a)$ , $s^0$
Choose $a^0$ at $s^0$ using policy derived from Q-values
Repeat for each time step
        Take action $a^k$, observe, $r^k$, $s^{k+1}$
        Choose $a^{k+1}$ at $s^{k+1}$ using policy derived from Q-values, with some exploration
          $Q^k(s^k,a^k) = Q^{k-1}(s^k,a^k) + \alpha^k[\,r^k + \gamma Q^{k-1}(s^{k+1},a^{k+1}) - Q^{k-1}(s^k,a^k)]$
        $s^k = s^{k+1}$   ;   $a^k = a^{k+1}$
```

3.1.4.2.1.2 Q-Learning: Off-Policy Algorithm

Q-learning updates the estimated value functions (Q-values) using greedy actions (Watkins and Dayan, 1992), independent of the current policy being followed, which involves exploratory actions. Q-learning is therefore considered an off-policy method as the agent attempts to improve a policy while following another one (Algorithm 3.3).

Algorithm 3.3: Q-Learning

```
Initialise $Q^0(s,a)$ , $s^0$
Choose $a^0$ at $s^0$ using policy derived from Q-values
Repeat for each time step
        Take action $a^k$, observe, $r^k$, $s^{k+1}$
        $Q^k(s^k,a^k) = Q^{k-1}(s^k,a^k) + \alpha^k\left[r^k + \gamma \max_{a^{k+1}\in A} Q(s^{k+1},a^{k+1}) - Q^{k-1}(s^k,a^k)\right]$
        Choose $a^{k+1}$ at $s^{k+1}$ using policy derived from Q-values, with some exploration
        $s^k = s^{k+1}$   ;   $a^k = a^{k+1}$
```

3.1.4.2.2 Eligibility Traces (TD(λ))

In eligibility traces algorithms, the agent looks ahead to the end of the learning horizon (T) and updates the value functions based on all future rewards. The TD error is then defined as follows (Dayan and Sejnowski, 1994):

$$TD^k(s) = \alpha^k \left((1-\lambda) \left(\sum_{n=1}^{T-t-1} \lambda^{n-1} R_k^{(n)} \right) + \lambda^{T-t-1} R_k \right) - V^k(s) \qquad (3\text{-}14)$$

$$R_k^{(n)} = \sum_{l=0}^{n-1} \gamma^l \, r^{k+l+1} + \gamma^n V^k\left(s^{k+n}\right) \qquad (3\text{-}15)$$

where $R_k^{(n)}$, the n-step return is defined as the discounted reward of n future steps, each is weighted proportionally to λ^{n-1}, where $0 \le \lambda \le 1$ and $1-\lambda$ is a normalisation factor to ensure that the weights sum to 1. This is known as a forward view (theoretical view) of the TD(λ) algorithm. If λ= 0, $TD^k(s)$ reduces to Equation 3-13, and hence the algorithm becomes the 1-step TD(0) which explains the notion of TD(0).

The forward view TD(λ) is not readily implementable (known as a causal) because its values are updated based on future time steps. The backward view on the other hand provides a plausible implementation of the theoretical view in a causal mechanism; which can be achieved using eligibility traces algorithms. An eligibility trace is an additional variable for each state that keeps track (i.e. *traces*) of how often and how recently a state is visited. In other words, the eligibility trace reflects how *eligible* a certain state is for update whenever a reward is received. At each time step the eligibility trace for each state is updated as follows:

$$e^k(s) = \begin{cases} \gamma\lambda e^{k-1}(s) & if\ s \neq s^k \\ \gamma\lambda e^{k-1}(s) + 1 & if\ s = s^k \end{cases} \qquad (3\text{-}16)$$

where λ is referred to as the trace-decay parameter. The update is then performed on the value estimates of all the states as follows:

$$TD^k(s) = \alpha^k [\, r^k + \gamma V^{k-1}(s') - V^{k-1}(s)] e^k(s) \qquad (3\text{-}17)$$

The implementation of Equations 3-16 and 3-17 is equivalent to Equation 3-14 (Sutton and Barto, 1998).

Eligibility traces therefore credit states for rewards obtained later on. In other words, eligibility traces allocates more credit for states visited recently than to states visited longer ago, which mimics the biological brain strategies for deciding how recently received incentives should be used in a combination with current incentives to take actions (Jang *et al.*, 1997).

The combination of TD(0) methods (e.g. SARSA and Q-learning) with eligibility traces results in SARSA(λ) and Q(λ) algorithms (Sutton and Barto, 1998).

3.1.4.2.2.1 SARSA(λ)

Combining the typical SARSA algorithm (Algorithm 3.2) and the eligibility traces concept results in SARSA(λ) as defined in Algorithm 3.4.

Algorithm 3.4: SARSA(λ)

```
Initialise $Q^0(s,a)$ , $s^0$, $e^0(s,a) = 0$ for all s, a
Choose $a^0$ at $s^0$ using policy derived from Q-values
Repeat for each time step
    Take action $a^k$, observe, $r^k$, $s^{k+1}$
    Choose $a^{k+1}$ at $s^{k+1}$ using policy derived from Q-values, with some exploration
        $\delta = \left[ r^k + \gamma Q^{k-1}(s^{k+1}, a^{k+1}) - Q^{k-1}(s^k, a^k) \right]$
        $e^k(s^k, a^k) = e^{k-1}(s^k, a^k) + 1$
        For all s, a:
            $Q^k(s,a) = Q^{k-1}(s,a) + \alpha\delta e^k(s^k, a^k)$
            $e^k(s,a) = \gamma\lambda e^{k-1}(s,a)$
    $s^k = s^{k+1}$ ;  $a^k = a^{k+1}$
```

3.1.4.2.2.2 Watkins' Q(λ)

Since Q-learning learns the greedy policy while following a policy involving exploratory actions, a subsequent experience can only be maintained as long as the greedy policy is being followed. Therefore, unlike SARSA(λ), Watkins' Q(λ) only looks ahead as far as the next exploratory actions without the need to look all the way to the end of the learning time (Algorithm 3.5).

Algorithm 3.5: Watkins' Q(λ)

```
Initialise $Q^0(s,a)$ , $s^0$
Choose $a^0$ at $s^0$ using policy derived from Q-values
Repeat for each time step
    Take action $a^k$, observe, $r^k$, $s^{k+1}$
```

```
$\delta = \left[ r^k + \gamma \max_{a^{k+1} \in A} Q(s^{k+1}, a^{k+1}) - Q^{k-1}(s^k, a^k) \right]$
$e^k(s^k, a^k) = e^{k-1}(s^k, a^k) + 1$
Choose $a^{k+1}$ at $s^{k+1}$ using policy derived from Q-values, with some exploration
    For all s, a:
        $Q^k(s, a) = Q^{k-1}(s, a) + \alpha \delta e^k(s^k, a^k)$
    If $a^{k+1} = arg \max_{a^{k+1} \in A} Q(s^{k+1}, a^{k+1})$, then $e^k(s^k, a^k) = \gamma \lambda e^{k-1}(s^k, a^k)$
    Else $e^k(s^k, a^k) = 0$
$s^k = s^{k+1}$ ; $a^k = a^{k+1}$
```

3.1.4.3 Exploration Strategy

One of the key aspects of RL is the trade-off between exploitation and exploration. To accumulate the maximum reward, the agent must reinforce and exploit the best-experienced actions. However, the agent has to explore new actions to discover better actions for the future. Exploration therefore enables the agent to visit a larger number of state-action pairs, which is the condition of convergence to optimal policy (i.e. visit each state-action pair an infinite number of times) (Watkins and Dayan, 1992). To balance exploration and exploitation in RL algorithms, algorithms such as ε-greedy and softmax are typically used (Sutton and Barto, 1998).

3.1.4.4 Learning Rate

The learning rate (step-size), α, controls how fast the estimates of value functions are modified. A recommended approach is to start with high learning rate at first to allow for fast modifications then use lower rates as time progresses (Gosavi, 2003). Reducing α with learning steps at an appropriate rate may achieve convergence even with a nondeterministic setting. The step-size rule suggested by Gosavi (2003) has been adopted in this research as follows:

$$\boldsymbol{\alpha^k = \frac{1}{v^k(s,a)}} \qquad (3\text{-}18)$$

where α^k is the learning rate at time k and $v^k(s, a)$ is the number of visits to a particular state-action pair (s,a).

3.1.4.5 Discount Rate

The discount rate γ sets the present value of the future rewards, i.e. a reward received k time steps in the future is only worth γ^{k-1} times what it would be worth if it was received immediately. Thus, by increasing the value of γ (i.e. approaching 1) the agent becomes more farsighted as it accounts for future rewards more strongly (Sutton and Barto, 1998).

Example 3.5: Signalised Intersection Control Using Q-Learning

This example shows the solution of Example 3 using the Q-learning algorithm. For illustration we assume a discount factor γ of 0.8. The learning rate α is defined as: B/v(s,a), where B is a constant =0.2 and v is the number of state-action pair visits. The exploration rate ϵ is defined as: C/t, where C is a constant = 0.9 and t is the number of time periods (when exploring, select action a_1). Q-values are represented in a form of S × A table (Q-table).

Table 3-1 shows the first 15 iterations using Q-learning. After 15 iterations, the control policy is represented by the following Q-table:

	a1	a2
s1	-4.51	-10.03
s2	-6.96	-3.08

Hence, the optimal control policy is similar to the one obtained from the DP method in Example 3.5:

$$\pi(s_1) = a_1,\ \pi(s_2) = a_2$$

Table 3-1 Example 3.5 Signalised Intersection Control Using Q-Learning

Time Period	State S_i	Probability Matrix				Reward Martix				Current Sate	Exploration Rate	ε-Greedy Action Seclection	Visits		Simulate Action & Observe Feedback		Learning Rate	Q-Values		
		a1		a2		R1		R2					a1	a2	Trans. state	reward	α	a1	a2	exploration
		S1	S2	S1	S2	S1	S2	S1	S2											
												0.787912			0.970993					
1	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50	1	0.9000	1	1	0	2	-16	0.200	-3.200	0.000	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1				0	0				0.000	0.000	1
												0.366825			0.903794					
2	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50		0.4500		1	0				-3.200	0.000	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1	2		2	0	1	2	-1	0.200	0.000	-0.200	2
												0.758477			0.056008					
3	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50		0.3000		1	0				-3.200	0.000	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1	2		1	1	1	1	-10	0.200	-2.000	-0.200	1
												0.548411			0.974868					
4	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50	1	0.2250	2	1	1	2	-50	0.200	-3.200	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1				1	1				-2.000	-0.200	
												0.037785			0.96202					
5	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50		0.1800		1	1				-3.200	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1	2		2	1	2	2	-1	0.100	-2.000	-0.296	2
												0.447718			0.636804					
6	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50		0.1500		1	1				-3.200	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1	2		2	1	3	1	-16	0.067	-2.000	-1.514	
												0.175936			0.0324					
7	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50	1	0.1286	1	2	1	1	-1	0.100	-3.236	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1				1	3				-2.000	-1.514	
												0.824864			0.191332					
8	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50	1	0.1125	1	3	1	1	-1	0.067	-3.260	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1				1	3				-2.000	-1.514	
												0.907186			0.981234					
9	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50	1	0.1000	1	4	1	2	-16	0.050	-3.957	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1				1	3				-2.000	-1.514	
												0.405184			0.069487					
10	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50		0.0900		4	1				-3.957	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1	2		2	1	4	1	-16	0.050	-2.000	-2.396	
												0.851621			0.653526					
11	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50	1	0.0818	1	5	1	2	-16	0.040	-4.503	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1				1	4				-2.000	-2.396	
												0.315411			0.997841					
12	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50		0.0750		5	1				-4.503	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1	2		1	2	4	2	-50	0.100	-6.960	-2.396	
												0.354803			0.258412					
13	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50		0.0692		5	1				-4.503	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1	2		2	2	5	1	-16	0.040	-6.960	-3.084	
												0.405184			0.352564					
14	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50	1	0.0643	1	6	1	1	-1	0.033	-4.506	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1				2	5				-6.960	-3.084	
												0.235134			0.256864					
15	s1	0.5	0.5	0.9	0.1	-1	-16	-10	-50	1	0.0600	1	7	1	1	-1	0.029	-4.509	-10.032	
	s2	0.1	0.9	0.7	0.3	-10	-50	-16	-1				2	5				-6.960	-3.084	

3.2 Stochastic Multi-Agent Control Problem

Employing adaptive signal control strategies at the local level (i.e. isolated intersection) might limit their potential benefits. Traffic systems are dynamic and hard to predict, not only due to the stochastic nature of the traffic patterns but also because of the complex effect of the actions performed by many other agents. Therefore optimally controlling the operation of multiple intersections simultaneously can be synergetic and beneficial for both individual intersections and the overall system of intersections. This section reviews the stochastic control problem in the multi-agent settings, which is more challenging since all decision makers are operating concurrently.

3.2.1 Theoretical Framework: Markov Game

The general framework of any multiple decision makers' problem is GT. GT provides the tools to model the multi-agent systems as a multiplayer game and provide the rational strategy to each player in that game. GT includes different forms of games such as *extensive form*, *strategic/matrix form*, and *coalitional form* (Mendelson, 2004; Osborne, 2004). Stochastic control for multi-agent systems, such as traffic control systems, can be framed as generalised MDPs in the multi-agent case; that is SGs (Littman, 1994; Bazzan, 2009) which is a subset of the matrix game (Figure 3-6). In the next sections, the most relevant GT concepts, and in particular SGs, are discussed.

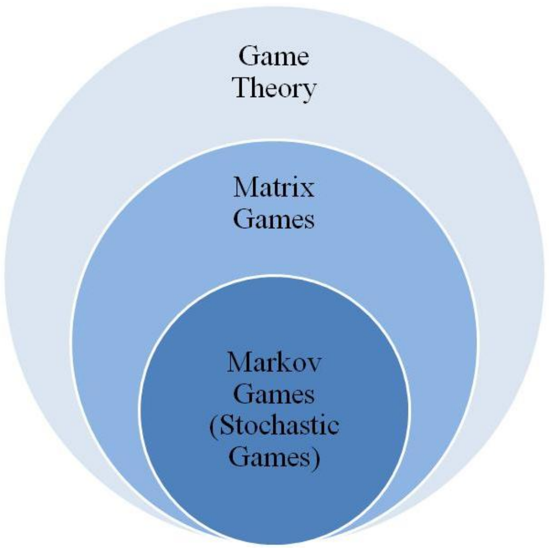


Figure 3-6 Multi-Agent Stochastic Control Framework

3.2.1.1 Game Theory (GT)

As noted by Littman (1994), "*no agent lives in a vacuum; the agent must be aware of other agents in the environment to achieve its target*". GT is considered a suitable method to model multi-agent systems as a multi-player game and to further provide the rational strategies of multiple players (Koller and Pfeffer, 1997).

Modern GT was created in 1944 (von Neumann and Morgenstern, 1944). One of the early games in GT are the zero-sum games, in which a gain for one player implies total loss for all other players. GT concepts evolved rapidly to cover a wide range of applications including both humans and non-humans, like computers (Osborne, 2004).

Many forms of games exist depending on the level of detail that describe the game, including: extensive form game, normal form game, coalition form game (Mendelson, 2004; Osborne, 2004). In this chapter the focus is on the normal form game since it is the most relevant form to the traffic signal control problem.

3.2.1.2 Normal Form Game (also known as the Matrix Game)

Definition 3.2: Matrix Game
The matrix game is a tuple $< N, A_{1 \dots N}, R_{1 \dots N} >$ where: • N is the number of players (agents) • A_i is the set of actions available to player i and $A = A_1 \times \dots \times A_N$ is the joint action space • R_i is player i 's reward function ($A \to \mathbb{R}$)

The players select actions from their available set with the goal to maximise their reward, which depends on all players' actions. In *matrix games*, the players decide their actions simultaneously or at least without knowing the actions of the others (Mendelson, 2004; Osborne, 2004).

It is called matrix game since the R_i functions can be written as n-dimensional matrices. It is worth noting that this definition is for a stage game, where the game is at certain state or the game has only one state, i.e. for a system of agents where the system progresses through a sequence of states, in each state the agents play a game to choose their actions. Also the agent's strategy could be pure (deterministic actions ($a_i \in A_i$)) or mixed in which each player has an independent distribution over their actions that assigns to each available action a likelihood of being selected. ($\mathrm{PD}(\mathrm{A_i}) = \{\sigma_\mathrm{i} = \mathrm{Pr_i}(\mathrm{a_i}) \ \ \forall\, \mathrm{a_i} \in \mathrm{A_i}\}$).

3.2.1.2.1 Classes of Matrix Games

As shown in Figure 3-7, matrix games can be categorised into two categories according to the reward structure of the players (Mendelson, 2004; Osborne, 2004):

3.2.1.2.1.1 Zero-sum Games

A game is defined as *zero-sum* or *purely competitive* if the sum of the rewards to the players is zero no matter what actions were taken by the players.

$$\sum_{i=1}^{N} r_i(a_1, \dots, a_N) = 0 \qquad (3\text{-}19)$$

3.2.1.2.1.2 Non-zero-sum game (general-sum game)

General-sum games are also categorised into two types:

- *Purely cooperative games* in which all agents have the same reward

$$r_i(a_1, \dots, a_N) = r_j(a_1, \dots, a_N) \ \forall\, i, j \in \{1, \dots, N\} \qquad (3\text{-}20)$$

- *Mixed games*, which are neither purely cooperative nor purely competitive

$$\sum_{i=1}^{N} r_i(a_1, \dots, a_N) \neq 0 \ and \ \exists\, i, j \in \{1, \dots, N\} \ where \ r_i(a_1, \dots, a_N) \neq r_j(a_1, \dots, a_N) \qquad (3\text{-}21)$$

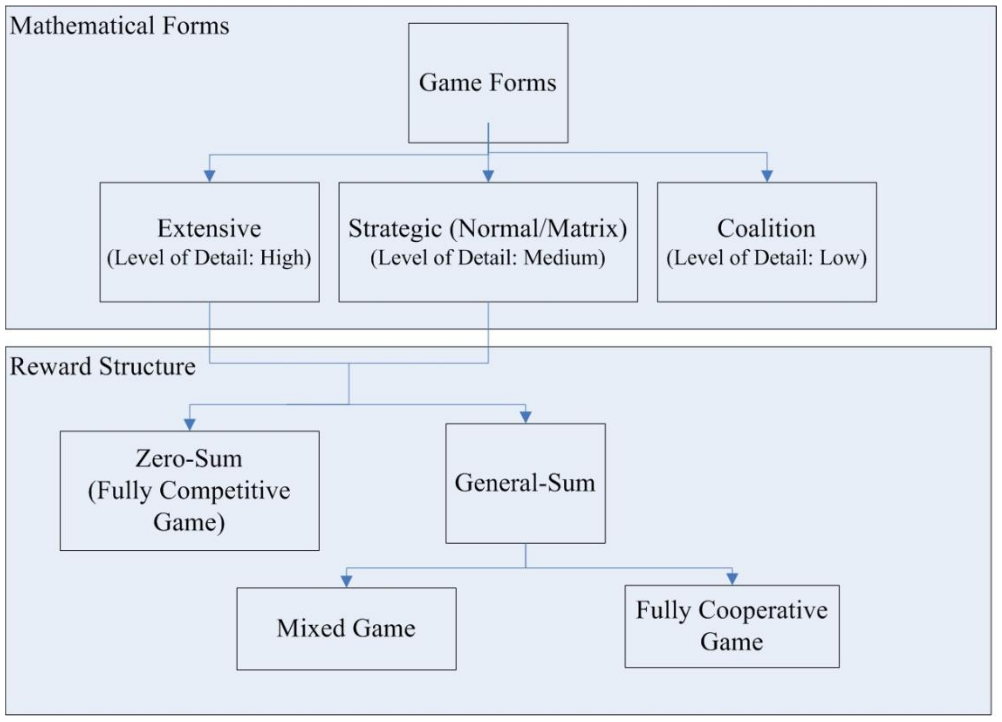

Mathematical Forms
Game Forms
Extensive (Level of Detail: High)
Strategic (Normal/Matrix) (Level of Detail: Medium)
Coalition (Level of Detail: Low)
Reward Structure
Zero-Sum (Fully Competitive Game)
General-Sum
Mixed Game
Fully Cooperative Game

Figure 3-7 Game Classes

3.2.1.3 Markov Game (Stochastic Game)

The Markov game (known as a stochastic game (SG)) is an extension of both the MDP and matrix game (Shapley, 1953). It is an extension to MDP with multiple agents and an extension to the matrix game with multiple environment states under uncertainty (stochastic environment). The game is played in a sequence of stages. At each stage, the game has a certain state in which the players select actions and each player receives a reward that depends on the current state and the chosen joint action. The game then moves to a new random state whose distribution depends on the previous state and the joint action chosen by the players. The procedure is repeated in the new stage and continues for a finite or infinite number of stages. The agents' objective is to find a joint policy in which each individual policy is a best response to the others, such as the Nash equilibrium (Basar and Olsder, 1999). Since SGs are extensions to the matrix game, the classification of games mentioned above is also applied to the SGs according to the reward structure of the agents. The SG can be seen as a mix of MDP and matrix game that is mathematically defined as in Definition 3.3 (Littman, 1994).

Definition 3.3: Stochastic Game (Markov Game)

A SG is a tuple $< N, S, A_{1\ldots N}, T, R_{1\ldots N} >$, where

- N is the number of agents
- S is the set of states
- A_i is the set of actions available to agent i and $A = A_1 \times \ldots \times A_N$ is the joint action space where $\vec{a} = \{a_1, \ldots, a_{N\}}\}$ represent the joint action vector
- T is a transition function $S \times A \times S \to [0, 1]$
- R_i is a reward function for the i^{th} agent ($A \to \mathbb{R}$).

Example 3.6: Stochastic Game of Two Signalised Intersections Control Problem

Two signalised intersections (N=2), each operating similarly to the logic mentioned in Example 3.1 above.

In this SG, the set of states, $S = \{s_1, s_2, s_3, s_4\}$, includes four states, which represents different combinations of traffic conditions at the two intersections. The first state (s_1) low queue length levels in both intersections $s_1 = \{\text{"low"}, \text{"low"}\}$. Similarly, $s_2 = \{\text{"low"}, \text{"high"}\}, s_3 = \{\text{"high"}, \text{"low"}\}$, $s_4 = \{\text{"high"}, \text{"high"}\}$. At each state two actions $A_i = \{a_{i1}, a_{i2}\}$ are possible for each intersection ($i = \{1, 2\}$), where "a_{i1}" extends the green time of the major street phase for the next time interval (e.g. 30

sec) and "a_{i2}" switches to the minor street phase for the corresponding minimum green time (e.g. 15 sec) and then activates the green again to the major street for the corresponding minimum green time (e.g. 15 sec). The reward function for each intersection is the negative summation of squared queue lengths. This example represents a general-sum game because the objectives of the agents are not identical and not competitive at the same time. Example reward matrices for intersection 1 and intersection 2 are represented in Table 1 and Table 2 below. This game can be viewed as four matrix games, one at each state.

Table 1

	a_{11}, a_{21}	a_{11}, a_{22}	a_{12}, a_{21}	a_{12}, a_{22}
s_1	-10	-20	-20	-10
s_2	-30	-20	-5	-30
s_3	-10	-5	-20	-30
s_4	-10	-20	-20	-10

Table 2

	a_{11}, a_{21}	a_{11}, a_{22}	a_{12}, a_{21}	a_{12}, a_{22}
s_1	-5	-10	-20	-10
s_2	-30	-5	-20	-30
s_3	-10	-20	-5	-30
s_4	-10	-20	-10	-5

3.2.2 Solution Form: Equilibrium

Unlike MDPs, it is difficult to define what it means to "solve" a matrix game. The solution concept for matrix games generally takes the form of equilibrium among the players, which implies that no player can improve on their current strategy. Different types of equilibrium assume different models of agreement among the players. The Nash equilibrium (Basar and Olsder, 1999) for example is a type of equilibrium applicable to a wide range of games. It is a well-known fact that finding Nash equilibria is an NP-Hard problem (Daskalakis *et al.*, 2006).

For a game, the *Best-Response* function for player i, $BR_i(a_{-i})$, is the set of all actions that are optimal (a_{-i}^{br}) given the other players' joint action, a_{-i}.

Definition 3.4: Best-Response Action

An action a_i^{br} is a best response to actions $\mathrm{a}_{-\mathrm{i}}$ of the other agents if

$r_i(a_i^{br}, a_{-i}) \geq r_i(a_i, a_{-i})$ $\forall a_i$ $\mathrm{r_i}(\mathrm{a_i^b}, \mathrm{a_{-i}}) \geq \mathrm{r_i}(\mathrm{a_i}, \mathrm{a_{-i}}) \forall \mathrm{a_i}$

The *Nash equilibrium (NE)* is the collection of best responses for all agents. Hence, no agent can do better by changing its action given that the other agents continue to follow the equilibrium actions.

Definition 3.5: Nash equilibrium

A joint action a^*, which regroups the actions for all agents, is a NE if

$\forall i, a_i^* \in BR_i(a_{-i}^*)$

where a_i^* is the action of the ith agent and a_{-i}^* are the actions of the other agents.

While zero-sum games converges to a unique NE, which can be thought of as the solution of a relatively simple linear programming problem (Nash and Sofer, 1996), finding equilibria in general-sum games requires more challenging quadratic programming solutions (Filar and Vrieze, 1997).

For illustration, the value of the equilibrium in two-player zero sum game ($\mathrm{r}_1 = -\mathrm{r}_2$) using mixed strategies can be computed by solving linear programming problem with the following objective function for which efficient algorithms are available (e.g. simplex being the most famous):

$$max_{\sigma_1 \in PD(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} r_1(a_1, a_2)\sigma_2(a_2) \tag{3-22}$$

It is proven that equilibria solutions exist for SGs just as they do for matrix games, for both zero-sum games (Shapley, 1953) and general-sum games (Filar and Vrieze, 1997).

Example 3.7: Nash Equilibrium Solution for a Two Signalised Intersections Control Problem

This example shows how to find the NE in the matrix game at state $s_1 = ("low", "low")$ in Example 3.6. Combining the reward matrices (Tables 1 and 2) of both agents at s_1, the resultant matrix game can be illustrated by the following matrix where the rows and the columns represent the actions of agent 1 and agent 2, respectively. The pair of values (r_1, r_2) at each cell represents the rewards associated with each joint-action for agent 1 (r_1) and agent 2 (r_2).

	a_{21} agent 2 action 1	a_{22} agent 2 action 2
a_{11} agent 1 action 1	-10, -5	-20, -10
a_{12} agent 1 action 2	-20, -10	-10, -30

Using definition 1, the best responses for agent 1 can be calculated as follows

$$BR_1 = \{arg \max_{a_{1j}} r_1(a_{1j}, a_{21}), arg\max_{a_{1j}} \mathrm{r}_1(a_{1j}, a_{22})\}$$

$$BR_1 = \{arg \max_{a_{1j}}(\{-10, -20\}), arg\max_{a_{1j}}(\{-20, -10\})\}$$

$$BR_1 = \{(a_{11}, a_{21}), (a_{12}, a_{22})\}$$

Similarly, the best responses for agent 2 are calculated below:

$$BR_2 = \{(a_{11}, a_{21}), (a_{12}, a_{21})\}$$

According to Definition 3.5, the NE is $NE = BR_1 \bigcap BR_2 = (a_{11}, a_{21})$

3.2.3 Game Theory Methods

Shapley's value iteration algorithm is the first GT method introduced for solving SG (Shapley, 1953). Shapley's value iteration algorithm learns a state-value function $\mathrm{V}(s)$ in which the goal is for V to converge to the optimal value function V^*, which is the expected discounted reward if the agents followed the game's NE (Algorithm 3.6).

Algorithm 3.6: Shapley's Algorithm

```
Initialise $V^0(s)$
Repeat
For each state, $s \in S$, compute the matrix
    $G_s^k(V^{k-1}) = \left[g^k_{\vec{a}\in A}: r(s, \vec{a}) + \gamma \sum_{s^{k+1}\in S} T(s, \vec{a}, s^{k+1}) V^{k-1}(s^{k+1})\right]$
For each state, $s \in S$, update $V^k = Value[G_s^k(V^{k-1})]$
Until $|V^k - V^{k-1}| < \delta$
```

As shown in Algorithm 3.6, the algorithm extends the TD technique to the matrix game. The value function is updated by the solving a matrix game $G_s(V)$ at each state. The dimensions of the matrix game $G_s(V)$ is $|A_1| \times \ldots \times |A_N|$ in which each entry represents the value of the joint action $\vec{a}$. The algorithm is conceptually very similar to the Value Iteration algorithm, except that the "max" operator is replaced by a "Value" operator. The operator "Value (G_s)" returns the value expected from playing the matrix game's NE, which is calculated either using linear programming (for zero-sum games) or quadratic programming (for general-sum games). Shapley's algorithm also shows that equilibria in SGs are solutions to Bellman-like equations. The algorithm's value function V converges to V^* which satisfies the following equation,

$$V^*(s) = Value[G_s(V^*(s))] \qquad \forall\, s \in S \tag{3-23}$$

Similar to DP, GT methods require a model of the environment since they make use of the transition, T and reward functions, R_i. Moreover, the goal of these algorithms is to compute the *equilibrium value* of the whole game (i.e. the expected discounted rewards for each of the agents) which is similar to the state-value function in DP, rather than finding equilibrium policies for each player.

3.2.4 Multi-Agent Reinforcement Learning (MARL)

MARL is an extension of RL to multiple agents in a SG. In MARL, it is assumed that the model of the environment (T and R) is not known *a priori*. The agents are required to act in the environment in order to converge to the optimal (equilibrium) policy. The second distinguishing characteristic is that these algorithms focus on the behaviour of each agent, and seek to find the equilibrium policy for that agent.

Although RL has been widely studied over the past two decades and the field of a single-agent RL is currently well-established with theoretical results and various practical applications, extending the single agent case to an environment where many agents exist is non-trivial as new challenges arise.

AI literature attempting to extend Bellman-style single-agent RL techniques (in particular Q-learning (Watkins and Dayan, 1992)) to the multi-agent setting have been well explored in the zero-sum repeated games, but less so in general-sum SGs. Therefore, decentralised decision making using MARL has become an active research topic in artificial intelligence (Weiss, 1999).

3.2.4.1 Independent MARL

The simplest way to extend RL to the multi-agent SG setting is just to add a subscript to the Q-learning update rule to reflect the agent's index (Equation 3-24). However, the learning agent, using this approach, pretends that the environment is stationary, i.e. the reward that the agent gets is assumed or perceived by the agent to be dependent only on its action and not affected by other agents (section 2.3.4):

$$Q_i^k\left(s^k, a_i^k\right) = Q_i^{k-1}\left(s^k, a_i^k\right) + \alpha^k\left[r_i^k\left(s^k, \vec{a}^k, s^{k+1}\right) + \gamma\, V_i^{k-1}(s^{k+1}) - Q_i^{k-1}\left(s^k, a_i^k\right)\right] \tag{3-24}$$

$$V_i^{k-1}(s^{k+1}) = \max_{a_i^{k+1} \in A_i} Q_i^{k-1}\left(s^{k+1}, a_i^{k+1}\right) \tag{3-25}$$

$$a_i^{k+1} = arg \max_{a_i^{k+1} \in A_i} Q_i^{k-1}\left(s^{k+1}, a_i^{k+1}\right) \tag{3-26}$$

3.2.4.2 Independent MARL Challenges

Although several authors have tested various forms of the above algorithm (e.g. Sen *et al.*, 1994), two main challenges make this approach implausible; one is learning related and the other is related to decision making.

3.2.4.2.1 Learning Challenge

The definition of the Q-values incorrectly assumes that they are independent of the actions selected by the other agents. In reality as well as in true MARL, the effect of the agents actions (i.e. the reward) on the environment is non-stationary, i.e. depends on the actions taken by the other agents; therefore considering joint-action in the learning process should not be ignored. The nonstationarity nature of the multi-agent learning problem exists due to the fact that all the agents are learning simultaneously. Therefore, each agent is faced with a moving-target learning problem in which the agent's optimal policy changes as the other agents' policies change (Busoniu *et al.*, 2008).

3.2.4.2.1.1 Overcoming the Learning Challenge

Overcoming the learning-related challenge can be achieved by defining the Q-values as a function of all agents' actions (joint-action):

$$\boldsymbol{Q_i^k\left(s^k, \vec{a}^k\right) = Q_i^{k-1}\left(s^k, \vec{a}^k\right) + \alpha^k [r_i^k\left(s^k, \vec{a}^k, s^{k+1}\right) + \gamma V_i^{k-1}\left(s^{k+1}\right) - Q_i^{k-1}\left(s^k, \vec{a}^k\right)]} \tag{3-27}$$

We are then left with the question of how to select the next action (the decision making-related challenge), given the more complex nature of the Q-values (Shoham *et al.*, 2003).

3.2.4.2.2 Decision Making Challenge

Although most MARL algorithms seek an equilibrium policy, achieving this goal if multiple equilibrium policies exist is challenging because agents are acting simultaneously. The agents are independent decision makers; each agent chooses its part from one of the equilibria, which might result in a non-equilibrium joint policy. In such cases, the agents require a <u>coordination</u> mechanism to take the optimal decision from the possible joint actions (i.e. agents have to coherently choose their actions so as to reach a unique equilibrium policy). Achieving this is not trivial, even for purely cooperative tasks (Busoniu *et al.*, 2008).

Example 3.8: Multiple Nash Equilibria Solutions for Two Signalised Intersections Control Problem

This example shows the possibility of the existence of multiple Nash equilibria in the matrix game at state s_2 in Example 3.6.

	a_{21}	a_{22}
a_{11}	-30, -30	-20, -5
a_{12}	-5, -20	-30, -30

Using Definition 3.4, the best responses for agent 1 and agent 2 are:

$$BR_1 = \{(a_{12}, a_{21}), (a_{11}, a_{22})\}$$

$$BR_2 = \{(a_{11}, a_{22}), (a_{12}, a_{21})\}$$

According to Definition 3.5, the Nash equilibria are:

$NE = BR_1 \cap BR_2 = \{(a_{11}, a_{22}), (a_{12}, a_{21})\}$

In this example, in the decision making process if each agent randomly chooses its action from this set of equilibria (given that the agents are acting simultaneously), the following scenario might happen:

If agent 1 randomly chooses the first equilibrium $(\mathrm{a}_{11}, \mathrm{a}_{22})$ and executes its part of this equilibrium (a_{11}) while agent 2 randomly choose the second equilibrium $(\mathrm{a}_{12}, \mathrm{a}_{21})$ and excutes its part of this equilibrium (a_{21}), the resulting joint-action is $(\mathrm{a}_{11}, \mathrm{a}_{21})$, which is the worst action for both agents.

3.2.4.2.2.1 Overcoming the Decision Making Challenge

As mentioned above, agents have to coordinate their actions to reach a unique equilibrium, which requires a coordination mechanism incorporated in MARL algorithms. As shown in Figure 3-8, MARL algorithms can be classified according to the type of coordination to Independent MARL, as discussed in section 3.2.4.1, and Coordinated MARL as follows (Busoniu *et al.*, 2008).

In coordination-based methods, each agent is not only considering joint-action while learning but also there is coordination mechanism in the decision making process. Two approaches are investigated in the literature under this category:

- Indirect-coordination methods: these methods drive the agent to select actions that will likely result in the best response to other agents' actions. The probability of obtaining best response values is evaluated using models of the other agents. These models are estimated from observed actions taken by the other agents in the past, without direct interaction/negotiation among the agents (Claus and Boutilier, 1998; Weinberg and Rosenschein, 2004);

- Direct coordination methods: in direct coordination, the agents – through direct interaction – are negotiating the action choice. Direct coordination mechanisms are inspired by social conventions, roles, and communication (Kok *et al.*, 2005; Yagan and Tham, 2007).

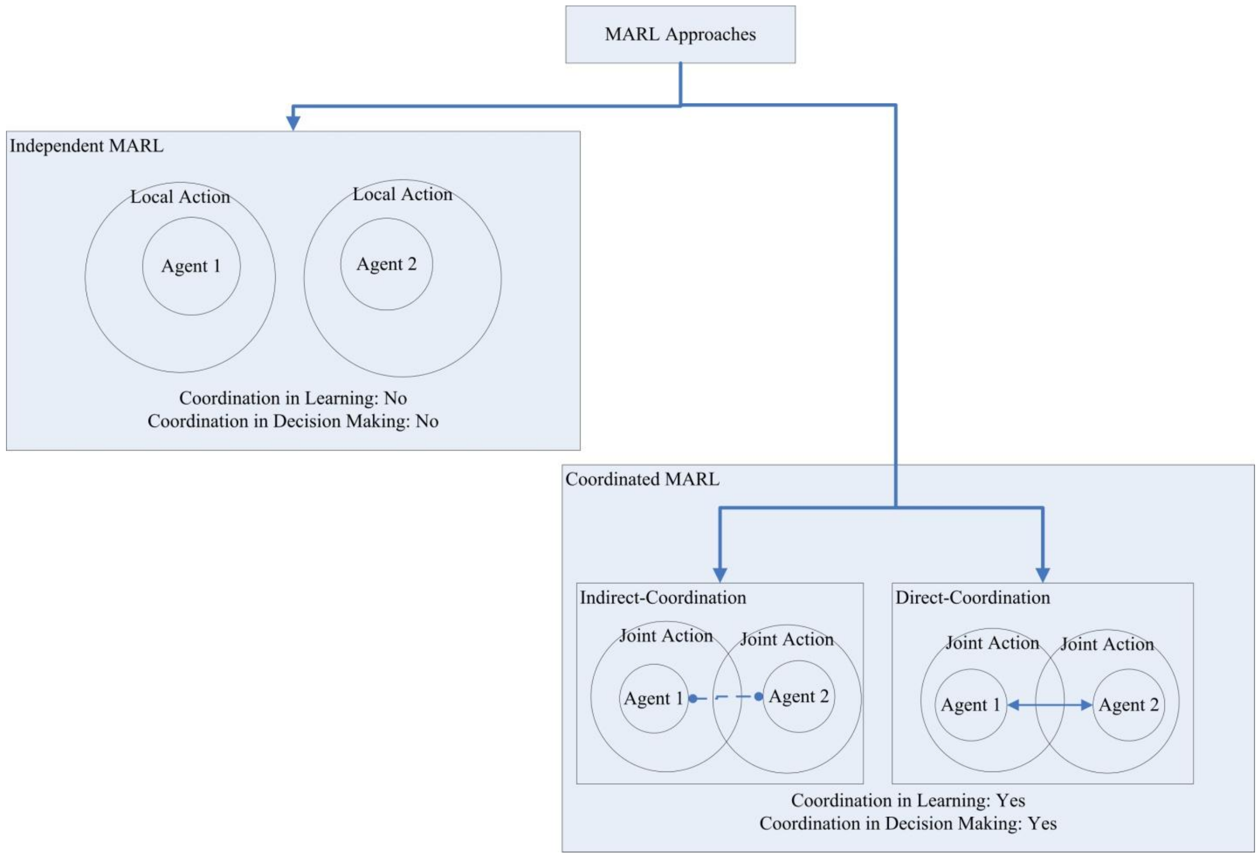


Figure 3-8 Categories of Multi-Agent Reinforcement Learning Approaches

3.2.4.3 Coordinated MARL Approaches

This section touches upon some of the MARL approaches according to the coordination techniques described above without any pretension of being comprehensive, given the huge number of approaches suggested in the literature.

3.2.4.3.1 MARL with Indirect- Coordination Mechanism

As discussed above, in the case of the existence of multiple equilibria, it is important to have a coordination mechanism that enforce the agents to play the same equilibrium joint-action.

One approach is to explicitly maintain a belief about the likelihood of the other agent's policy. In other words, each agent builds a model for the other agent ($M_i(s, a_{-i})$) that estimates the probability of selecting an action a_{-i} at each state s. The action is then selected based on the induced expectation of the Q values:

$$V_i(s^{k+1}) = max_{a_i} \sum_{a_{-i}\in A_{-i}} M_i(s^{k+1}, a_{-i})\, Q_i(s^{k+1}, (a_i, a_{-i})) \qquad (3\text{-}28)$$

$$a^{k+1} = arg\ max_{a_i} \sum_{a_{-i}\in A_{-i}} M_i(s^{k+1}, a_{-i})\, Q_i(s^{k+1}, (a_i, a_{-i})) \qquad (3\text{-}29)$$

Several MARL algorithms in GT follow the belief-based approach, such as *fictitious play* (Brown, 1951) and *rational learning* (Kalai and Lehrer, 1993) that is pursued by Claus and Boutilier (1998). However, these algorithm are designed for deterministic games (Wang and Sandholm, 2002).

For general-sum SGs, the Non-Stationary Converging Policies (NSCP) algorithm (Weinberg and Rosenschein, 2004) estimate models of the other agents' strategies or policies and computes a best response to the models and uses it to estimate state–action values (Algorithm 3.8). NSCP is the first algorithm that is proved to converge to the optimal policy against the non-stationary opponents in general-sum games (Weinberg and Rosenschein, 2004).

Algorithm 3.8: Non-Stationary Converging Policies algorithm (NSCP)

```
Initialise $Q_i^0(s,\vec{a})$ to arbitrary values, $M_i^0(s,a) = \frac{1}{|A|}$   $\forall\, s \in S, a \in A, and\ i \in N$
Repeat for each time step
For i=1 to N do
Observe the actions taken by other agents ($a_{-i}{}^k$), reward $r_i^k$, and next state $s^{k+1}$
Update the other agents' models ($M_{-i}{}^k$)
Update Q-values using the following rule
    $Q_i^k(s^k,\vec{a}^k) = Q_i^{k-1}(s^k,\vec{a}^k) + \alpha^k[r_i^k + \gamma \max_{a_i\in A_i} \sum_{a_1\in A_1} .. \sum_{a_j\in A_j, j\neq i} .. \sum_{a_N\in A_N} \prod_{j=1, j\neq i}^{N} M_j{}^k(s^{k+1},a_j).\ Q_i^{k-1}(s^{k+1},(a_i,a_{-i}{}^k)) - Q_i^{k-1}(s^k,\vec{a}^k)]$
Select the next action for state $s^{k+1}$ as the best-response action ($br_i{}^k$)
    $br_i{}^k = arg\max_{a_i\in A_i} \sum_{a_1\in A_1} \dots \sum_{a_j\in A_j, j\neq i} \dots \sum_{a_N\in A_N} \prod_{j=1, j\neq i}^{N} M_j{}^k(s^{k+1},a_j).\ Q_i^{k-1}(s^{k+1},(a_i,a_{-i}{}^k))$
End for
```

3.2.4.3.2 MARL with Direct- Coordination Mechanism

A general approach for solving the multiple equilibria problem is to make sure that the agents choose their actions in the same way. This clearly requires that random action choices are somehow coordinated or negotiated. The agents' action choices can be directly coordinated through mechanisms based on role assignments, social conventions, or/and communication (e.g. coordination graphs).

Using role assignments restricts the possible action choices available to the agent prior to action selection in a way that ensures that all agents will play the same equilibrium joint-action (Spaan *et al.*, 2002). Similar to role assignment, social conventions restrict the action choices of the agents but in a different way. Social conventions set *a priori* preferences to achieve unique joint-action selection. An example of a simple social convention is to set a unique ordering of agents and actions to be followed when the agents choose their actions, the ordering of which must be known to all agents (Boutilier, 1996).

On the other hand, coordination graphs (Kok *et al.*, 2005) is one of the communication-based direct coordination mechanisms in which the joint Q-value can be decomposed into local Q-values for each couple of agents that are connected. For instance, in an SG with three agents, the decomposition might be

$$Q(s,\vec{a}) = Q_1(s,[a_1,a_2]) + Q_2(s,[a_1,a_3]) + Q_3(s,[a_2,a_3]) \qquad \textit{(3-30)}$$

Max-plus (Kok and Vlassis, 2005) is an algorithm that estimates the optimal joint action by sending locally optimised messages among connected agents in coordination graphs.

Yagan and Tham (2007), proposed a direct-coordination mechanism (Locally Interacting Distributed Reinforcement Learning Policy Search: LID-RLPS) that uses communication in conjunction with social conventions and role assignment, and is proven to converge to the optimal policies. The logic of the LID-RLPS algorithm is that each agent is communicating information with neighbours to improve agent i's policy with respect to its neighbours' policies. Each agent is exchanging its policy and the gain obtained from changing its optimal policy to another policy given the policies of its neighbours. Role and social conventions are represented by allowing an agent to improve its policy if its gain message is larger than all the gain messages received from all its neighbours. Essentially, an agent changes its policy to the computed best local policy if it is the winner at the current time step. The pseudo-code of LID-RLPS is shown in Algorithm 3.9.

Algorithm 3.9: Locally Interacting Distributed Reinforcement Learning Policy Search (LID-RLPS)

```
Initiate random policies $a_i^{*0}$   $\forall\, i \in N$
Repeat for each time step
Observe state $s^k$
For each agent
Exchange policies $a_i^{*k}$ with neighbours
Choose and execute action $a_i^k$ using Q-values, with some exploration
                    $cValue = GetQvalue(s^k, (a_i^{*k}, a_{-i}^{*k}))$
                    $a{*}_i^k = getBestPolicy(s^k, (a_i^k, a_{-i}^{*k}))$
                    $mValue = GetQvalue(s^k, (a{*}_i^k, a_{-i}^{*k}))$
                    $gain_i = |cValue - mValue|$
Broadcast $gain_i$ to neighbours
                    $maxGain = \max_{k \in N} gain_k$
                    $winner = arg \max_{k \in N} gain_k$

If $maxGain > 0$
If $i = winner\ then$
Update policy $a_i^k = a{*}_i^k$
Broadcast $a{*}_i^k$ to neighbours
Else
Receive policy $a{*}_{winner}^k$ from winner
Update $a_{-i}^k$
End if
Else
End if
```

3.2.4.4 Limitations in Existing MARL Approaches

The decentralised traffic control problem is an excellent testbed for MARL due to the inherent dynamics and stochastic nature of the traffic system (Bazzan, 2009; El-Tantawy and Abdulhai, 2010). However, MARL methods face numerous challenges related to the exponential growth in the action space with the increase in the number of agents. In addition, problems such as traffic signal control offer a great challenge due to the additional complexity of the exponential growth in state-space with the increase in the number of agents, due to large number of possible states in a traffic network.

3.2.4.4.1 Action Space Curse of Dimensionality-General Limitation

In a SG (refer to section 3.2.1.3), assuming that the number of states is m for a game with two agents, each agent is required to maintain a Q-table for its own Q-values. For each agent i (i=1,2), a Q-table Q_i has its rows corresponding to $s \in S$, columns corresponding to joint-actions $(a_1, a_2) \in A_1 \times A_2$, and each entry as $Q_i(s, (a_1, a_2))$, i=1,2. The total number of entries agent i requires to learn for its Q-table is $|S| \times |A_1| \times |A_2|$ where $|S|, |A_i|$ are the sizes of the state space and action space of agent i. For N agents, each agent has to keep a set of tables with a size that is exponentially increasing with the number of agents: $|S| \times |A_1| \times \ldots \times |A_N|$. Therefore, a compact representation of action space is required.

3.2.4.4.2 State-Space Curse of Dimensionality- Problem Specific Limitation

In fact, the general SG framework does not assume a relation between the state space and the number of agents. However, this framework cannot cope with the problem dimensions of controlling a network of traffic signals. The main challenge is that the states in the traffic control problem are a function of the number of agents.

The curse of dimensionality arises because the state space grows exponentially with the number of agents. Even in SG-based MARL approaches that are proven to optimally converge to the joint policy (e.g. OAL (Claus and Boutilier, 1998), and NSCP (Weinberg and Rosenschein, 2004)), each agent has to keep a set of tables with a size that is exponential to the number of agents: $|S_1|\times\ldots\times|S_N|\times|A_1|\times\ldots\times|A_N|$ where S_i and A_i represent the state and action spaces for agent i, respectively. To illustrate the dimensionality of the problem, assume a simple discretization of states in traffic signal control, namely congested or not congested, i.e. $|S_i| = 2$ for $i = 1, \ldots, N$ and that traffic signals can only take two actions (e.g. extend the current green or switch to another phase); the size of the Q table is $2^N \times 2^N$ which is exponentially increasing with the number of agents.

In addition to the dimensionality issue, the theoretical definitions for the SG-based MARL methods require each agent (e.g. signalised intersection) to observe the conditions of all other agents (e.g. the entire network), which is impractical in the traffic signal control problem.

Acknowledging the limitations of existing SG-based MARL approaches, they cannot be readily employed in traffic control at the network level without major approximations and simplifications. The complexity of the problem could be addressed by organising agents in groups of limited size. However, the coupling between these groups remains a coordination issue. Thus flexible and computationally efficient approaches are instrumental in controlling a network of agents; plausibly by employing heuristics and approximate approaches based on modifying the existing MARL techniques (Bazzan, 2009).

In the next chapter, we develop and present a novel MARL-based approach to overcome these limitations and allow the applicability of the SG-based MARL algorithm on a network of connected agents in which the coordination mechanism is maintained without exacerbating the curse of dimensionality.

4 MARLIN-ATSC FRAMEWORK

In this chapter a novel MARLIN-ATSC framework is developed to advance the current state-of-the-art MARL-based ATSC, which is the core contribution of this research.

Chapter 2 presented a review of signal control with an emphasis on categorising the ATSC approaches that are based on MARL as shown in Figure 2-4 (i.e. highlighting the application side of the ATSC). Chapter 3, on the other hand, provides a review of the theoretical background of RL for a single agent and MARL for multiple agents with an emphasis on categorising MARL approaches as shown in Figure 3-8 (i.e. highlighting the theory side of ATSC). Figure 4-1 combines the categories summarised in Figure 2-4 and Figure 3-8, while mapping the contribution of the MARL-based ATSC studies to each category of the theoretical MARL approaches.

The findings from the previous two chapters resulted in identifying four major limitations related to MARL-based traffic signal control systems:

1. Lack of quantitative justification for the RL-based system design in solving the traffic signal control problem. Different RL algorithms, and more specifically TD algorithms, have been separately investigated without systematic investigation or quantitative justification for the RL-based system design in solving the traffic signal control problem;
2. Lack of efficient coordination between controllers in most of the MARL-based approaches for decentralised traffic signal control;
3. Lack of a resolution to the curse of the dimensionality issue in coordinated MARL approaches and hence a lack of modelling/optimisation approaches for large-scale stochastic-game MARL solution to the ATSC problem. This is due to the exponentially increasing joint space of states and actions with the increase in the number of agents;
4. Lack of a resolution to the fact that ATSC agents have a limited ability to observe the global state (e.g. the conditions of the entire network) – a limitation of existing coordinated MARL approaches. Observing traffic conditions throughout the entire network is impractical, costly and exacerbates the curse of dimensionality issue.

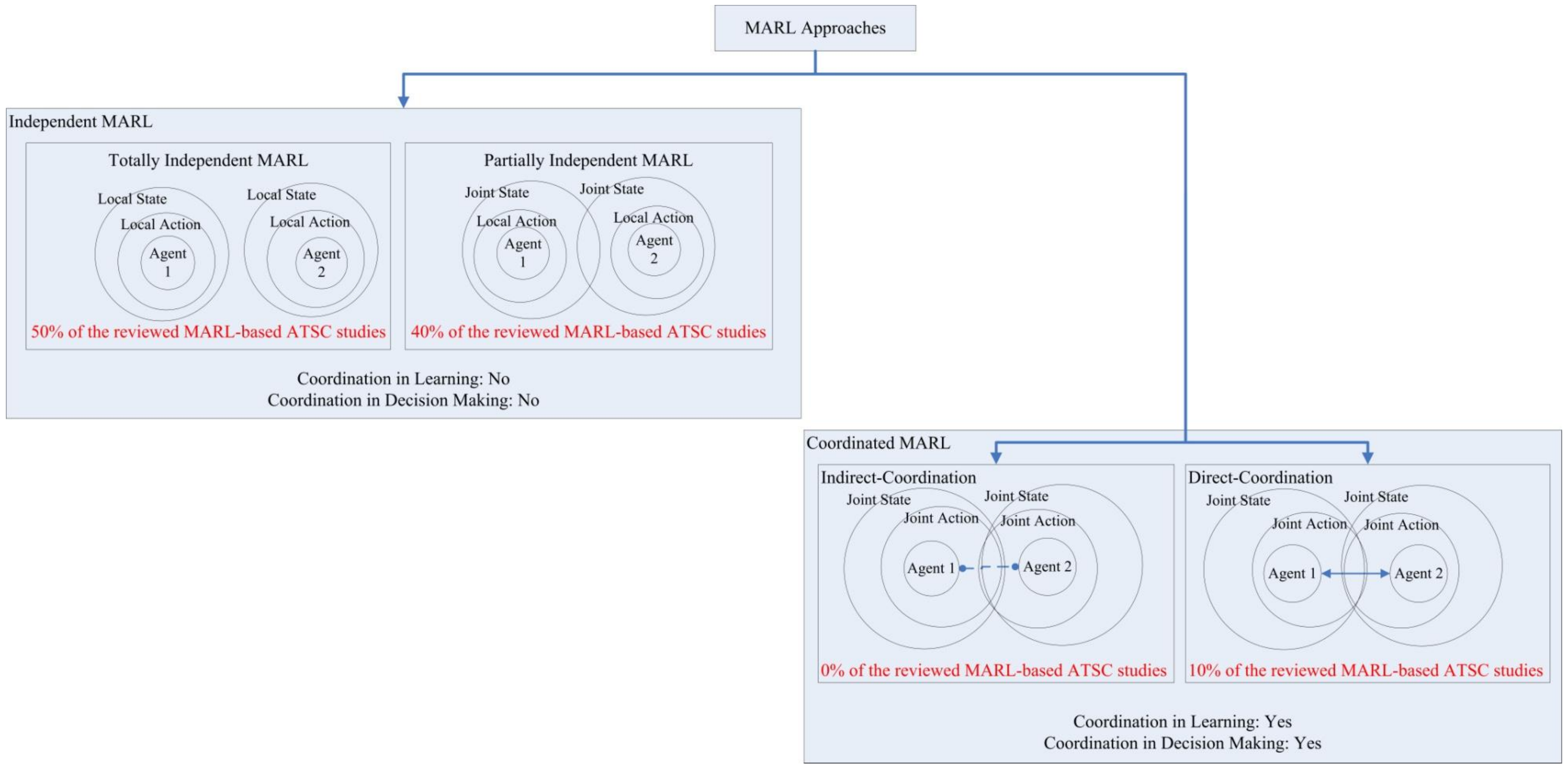


Figure 4-1 Mapping the Literature of MARL-based ATSC to the Theory of MARL

Therefore the need for a comprehensive investigation into RL design parameters, the need for coordination, and the curse of dimensionality form the major challenges for developing a decentralised traffic signal control system using SG-based MARL. In the case of a single intersection these challenges are not encountered. However, these challenges are more profound for controlling large traffic networks of many intersections where coordination among agents/intersections becomes essential to area-wide performance, and not only at the local level, and while the dimensionality of the problem increases intractably.

In this research a MARLIN-ATSC framework is developed to address the previous challenges by:

1. developing a new MARL algorithm that maintains a ***coordination*** mechanism between agents without compromising the dimensionality of the problem (hereafter referred to as the ***MARLIN*** Control Algorithm) which represents the theoretical contribution of this thesis. MARLIN synergises the principle of the locality of interaction (Nair *et al.*, 2005) and the modular Q-learning technique (Ono and Fukumoto, 1996) with the coordination-based MARL approaches presented in section 3.2.4.3. The generic development of MARLIN's control logic allows its applicability to any set of networked controllers in a stochastic environment (traffic and other). Two coordination methods are developed in MARLIN:
   a. Multi-Agent Reinforcement Learning for Integrated Network of *Indirectly* Coordinated controllers (***MARLIN-IC***). MARLIN-IC steers the action selection towards actions that represent the best response to the *expected* neighbours' actions, hence guiding the agent toward coordinated action selection. The best response is evaluated using models of the neighbours' behaviour that are *estimated* by the agent from observing their actions in the past.
   b. Multi-Agent Reinforcement Learning for Integrated Network of *Directly* Coordinated controllers (***MARLIN-DC***). MARLIN-DC uses a combination of communication and social conventions between the agent and its neighbours. Communication is used to negotiate the action choices among connected agents. A simple social convention is used to provide ordering between agents so they can select actions in turn and broadcast their selection to the remaining agents until the best joint action set is reached.

2. developing a generic software platform for investigating different coordination levels and different RL design parameters in the ATSC problem (***MARLIN-ATSC Platform***). In MARLIN-ATSC agents can implement one of the following two control modes:
   a. **Independent Mode:** In this mode each controller has an RL agent working independently of the other agents using Multi-Agent Reinforcement Learning for Independent controllers (***MARL-I***) explained in section 3.2.4.1. In order to investigate different independent mode options considered in the literature as discussed in sections 2.3.4.1 and 2.3.4.2, two MARL-I algorithms are developed in the MARLIN-ATSC platform:
      i. Multi-Agent Reinforcement Learning for Totally Independent controllers (***MARL-TI***). MARL-TI refers to the agents that implement the single-agent RL methods while solely considering its local state and action and can be applied for isolated intersections or in situations where the coordination between agents is not necessary (e.g. if intersections are far enough apart and hence have no effect on each other).

While MARL-TI is not new compared to the literature, developing MARL-TI in the MARLIN-ATSC platform is important to explore the best design for RL-specific parameters (e.g. exploration method, action definition, state representation, reward function) and to set a benchmark for comparing the performance of the new methods developed in this research.

      ii. Multi-Agent Reinforcement Learning for Partially Independent controllers (***MARL-PI***). The only difference between MARL-PI and MARL-TI is in the definition of the state space for each agent. MARL-PI extends MARL-TI by incorporating in the state space of an agent – in addition to its local state – in the states of the neighbouring agents.

In Chapter 5, the MARLIN-ATSC independent mode (MARL-TI) is tested on an isolated intersection to investigate different RL parameters. Furthermore, MARL-TI and MARL-PI are investigated in Chapter 6 on a prototype network of 5 intersections.

   b. **Integrated Mode:** In this mode, each controller coordinates the signal control actions with the neighbouring controllers as discussed in section 3.2.4.3 using either ***MARLIN-IC*** or ***MARLIN-DC***.

MARLIN-IC and MARLIN-DC are investigated in Chapter 6 on a prototype network of 5 intersections.

Figure 4-2 briefly summarises the limitations discussed in Chapters 2 and 3, and logically illustrates how the MARLIN-ATSC platform addresses these limitations in Chapter 4. While the first part of this chapter discusses the novel MARLIN control algorithm, the second part details the development of the MARLIN-ATSC platform as shown in the chapter roadmap in Figure 4-2.

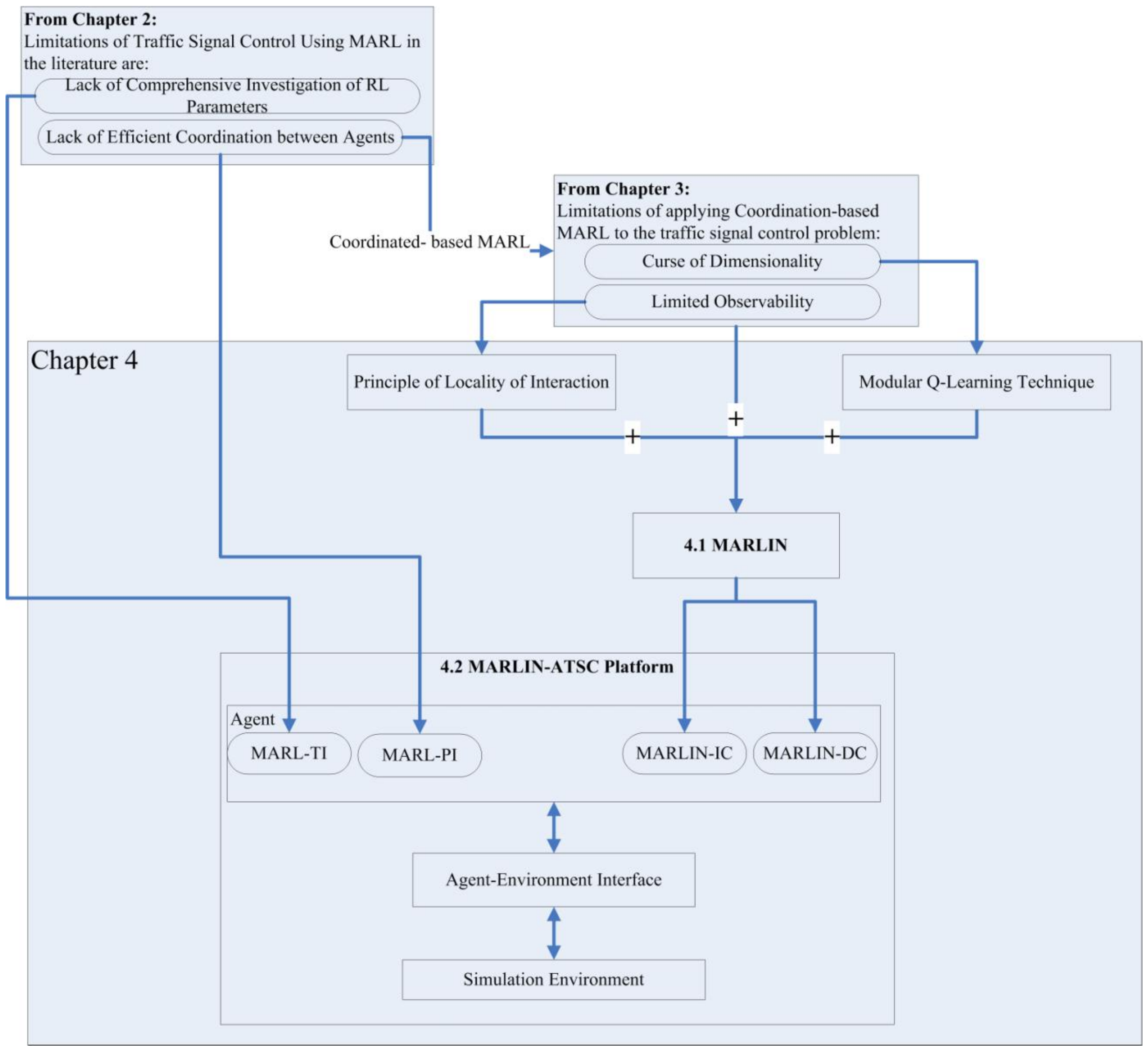

From Chapter 2:
Limitations of Traffic Signal Control Using MARL in the literature are:
Lack of Comprehensive Investigation of RL Parameters
Lack of Efficient Coordination between Agents
From Chapter 3:
Limitations of applying Coordination-based MARL to the traffic signal control problem:
Curse of Dimensionality
Limited Observability
Coordinated- based MARL
Chapter 4
Principle of Locality of Interaction
Modular Q-Learning Technique
+
+
+
4.1 MARLIN
4.2 MARLIN-ATSC Platform
Agent
MARL-TI
MARL-PI
MARLIN-IC
MARLIN-DC
Agent-Environment Interface
Simulation Environment

Figure 4-2 Roadmap for Chapter 4 and Linkage to Chapters 2 and 3

4.1 Multi-Agent Reinforcement Learning for Integrated Network (MARLIN) Control Algorithm

Most MARL approaches for N-Players general-sum SGs are developed to learn the optimal joint policy using the joint-state and the joint-action for the N players. These approaches are proven infeasible for large number of players (agents) due to the fact that the state and action spaces grow exponentially (curse of dimensionality) and the learning speed decreases dramatically with the number of agents (Bazzan, 2009).These methods are therefore intractable in the real-world scenario of a network of large number of signalised intersections. On the other hand, the theoretical definitions for these methods require each agent (e.g. signalised intersection) to observe the conditions of the other agents (e.g. the entire network), which is impractical. Thus, as suggested by Bazzan (2009), flexible and computationally efficient approaches are becoming instrumental in controlling a network of agents by employing heuristics and approximate approaches based on modifying the existing MARL techniques.

In typical traffic networks, signalised intersections (agents) are physically connected through the roadway network, which represents a set of neighbouring agents. Although in traffic networks the agent is practically incapable of observing the conditions of the entire network, it is possible to observe the conditions of the neighbouring agents. Each neighbour in turn observes its neighbours, spreading the coordination across the network in a cascading fashion

Therefore, MARLIN presents a new control system that maintains an *explicit coordination mechanism* for a large-scale network of connected agents by:

1. Exploiting the *Principle of* the *Locality of Interaction* (section 4.1.1) among agents to address the limited ability of an agent to observe an entire network of agents.
2. Utilising the *Modular Q-Learning Technique* (section 4.1.2) to address the curse of dimensionality problem.

4.1.1 Principle of Locality of Interaction

Although the existing SG-based MARL approaches capture the real-world uncertainty in multi-agent domains and include coordination mechanisms to allow the agents to coherently choose their optimal actions, they fail to exploit the locality of interaction among agents to approximate the joint optimal policy. The principle of locality of interaction (Nair *et al.*, 2005) (Definition 4.1) endeavours to estimate a *local* neighbourhood *utility* that maps the effect of an agent to the global value function while only considering the *interaction* with its neighbours. Hence, it is sufficient to consider the neighbours' policies to find the best policy for the agent.

Definition 4.1: Principle of the Locality of Interaction

For agent i, given that a set NB_i contains the agents that are linked (adjacent) to agent i, the property of the locality of interaction is satisfied if the expected reward for the joint policy π of the agents NB_i (local neighbourhood value function) is equal to the local neighbourhood value function for policy π', given that there is no change in the policies of agent *i* and its neighbours (NB_i).

$V^{\pi}[NB_i] = V^{\pi'}[NB_i]$ if $\pi_i = \pi_i'$ and $\pi_{NB_i} = \pi_{NB_i}'$

Where $V^{\pi}[NB_i] = \sum_{j \in i \cup NB_i} V_{\pi_j}(s) \quad \forall\, s \in S$

4.1.2 Modular Q-Learning Technique

Ono and Fukumoto (1996) proposed a Modular Q-learning technique to keep the size of the state-space to a manageable size. Modular Q-learning partitions the state-space to partial state spaces that consist of two agents. As a consequence, the size of the partial state space is always $|S|^2$ regardless of the number of agents, and therefore results in a reasonable state space. The Principle of Modular Q-learning (Ono and Fukumoto, 1996) was primarily proposed to solve the curse of dimensionality problem. This method however does not attempt to solve the coordination problem.

Definition 4.2: Modular Q-Learning Technique

For N agents, N – 1 partial state spaces exist with two agents each, and one learning module is assigned to each of them. Each learning module performs Q-learning and passes the Q value to the action selection module in which agent i chooses the next action using the following equation:

$$a_i^{k+1} = arg \max_{a \in A_i} \sum_{j=1, j \neq i}^{N-1} Q_{i,j}^{k}\,(s_{i,j}^{k}, a)$$

where $s_{i,j}^{k}$ denotes the partial state that consists of agent i and another agent j.

4.1.3 MARLIN Conceptual Design

MARLIN synergises the effect of the following:

1. The coordination-based MARL approaches proposed by Weinberg and Rosenschein (2004) and Yagan and Tham (2007);

2. The locality of interaction principle; and
3. The modular Q-learning technique.

In this system, each agent plays a game with all its adjacent intersections in its neighbourhood. The agent has a number of learning modules; each corresponds to one game. The state-space and the action-space are distributed such that the agent learns the joint policy with one of the neighbours at a time following the principle of modular Q-learning. The agent learns the policy based on implementing one of the two learning approaches: MARLIN-IC and MARLIN-DC. In MARLIN-IC, each agent estimates models for the neighbours' policies and uses the estimated models to update its policy. Similar to Q-learning, MARLIN-IC updates the estimated value functions (Q-values) using best-response actions, independent of the current policy being followed, which involves exploratory actions. MARLIN-IC is therefore considered a value-iteration algorithm with an off-policy approach as the agent attempts to improve a policy while following another one. On the other hand, in MARLIN-DC the agent starts with a random policy and exchanges that policy with the neighbours. The agent uses this information to evaluate and improve the policy. Therefore, MARLIN-DC is considered a policy-iteration algorithm with an on-policy approach, in which case the agent's policy is updated based on the current policy.

Finally, in the decision making process of the agent, given the joint policies with neighbouring agents (the output from the learning process), the agent decides the action based on implementing one of the two coordination mechanisms: MARLIN-IC and MARLIN-DC. The conceptual design of MARLIN is illustrated in Example 4.1. In the next section, the mathematical framework, the learning approach, and the decision making technique are presented.

Example 4.1: MARLIN Illustrative Example
In multi-intersection traffic control problems (multi-agents), each intersection is inevitably affected by its neighbours who are also affected by their neighbours in a network-wide cascading fashion. Hence, a network of agents is formed based on each agent's interactions with a small number of neighbours. In a typical transportation network, the waiting time of vehicles at an intersection depends on the traffic conditions and the action taken at that intersection and the actions taken at the neighbouring intersections. Three cases are shown in Figure 4-3 for an illustrative grid network: a) intermediate intersection I5, b) edge intersection I6, and c) corner intersection I9.

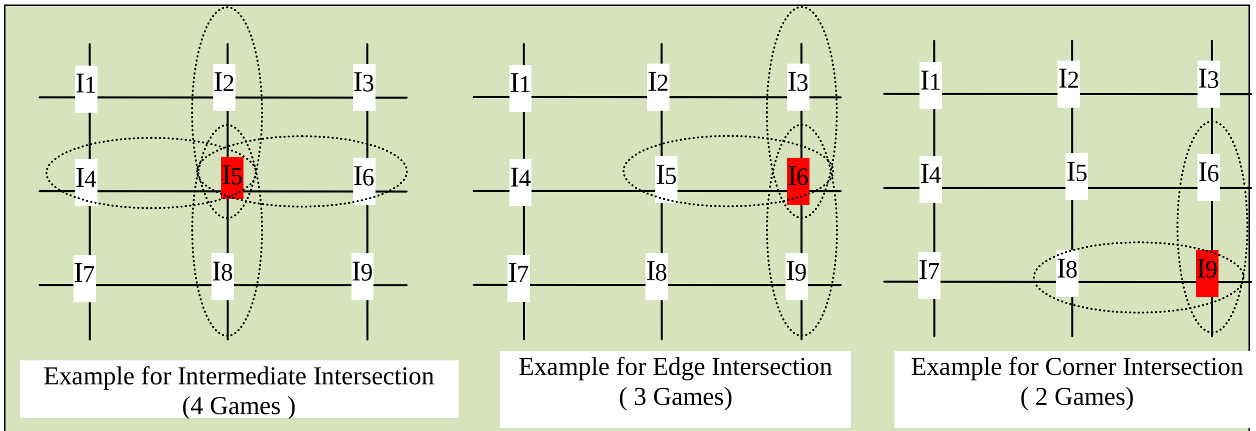

I1
I2
I3
I4
I5
I6
I7
I8
I9
Example for Intermediate Intersection (4 Games)
Example for Edge Intersection (3 Games)
Example for Corner Intersection (2 Games)

Figure 4-3 Illustrative Examples of Intersection's Neighbourhood in Grid Network Using MARLIN

For example, in Figure 4-3 (a) the agent I5 has four neighbours, hence agent I5 is a common player in 4 games, one with each neighbour (I2, I4, I6, and I8) and hence, there are four learning modules associated with agent I5 as shown in Figure 4-4.

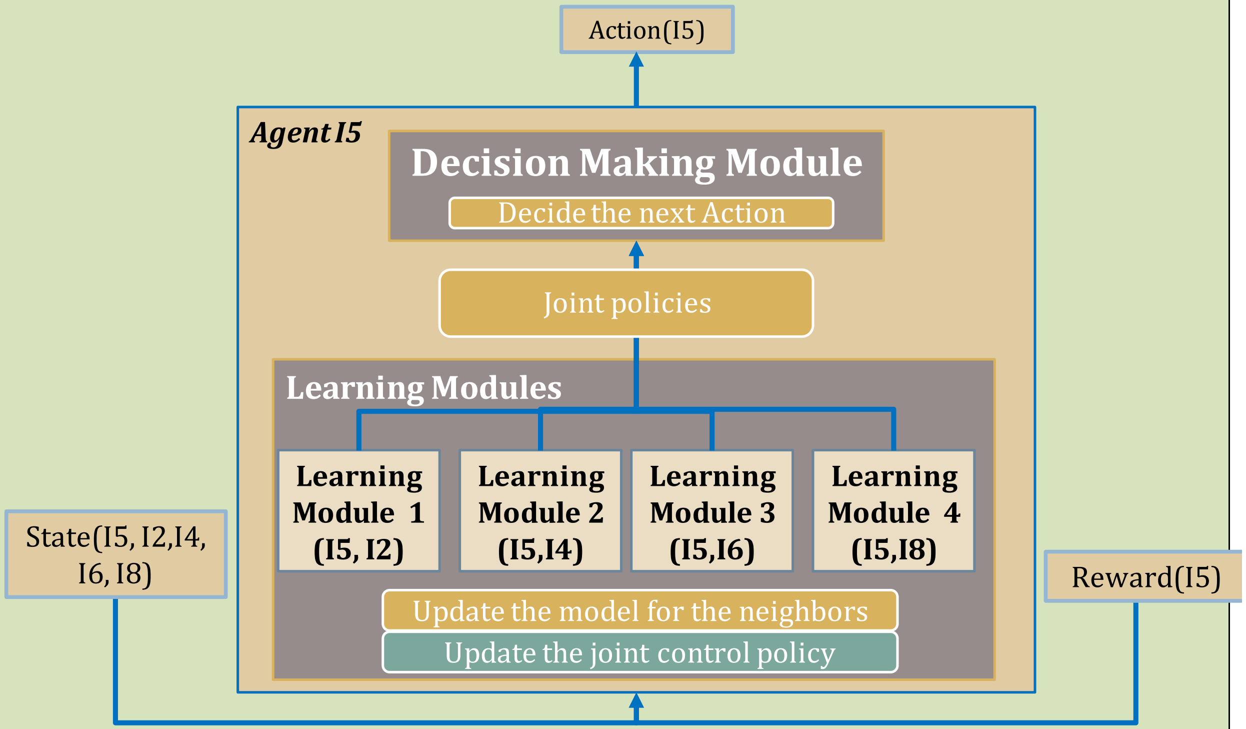

Action(I5)
Agent I5
Decision Making Module
Decide the next Action
Joint policies
Learning Modules
Learning Module 1 (I5, I2)
Learning Module 2 (I5,I4)
Learning Module 3 (I5,I6)
Learning Module 4 (I5,I8)
Update the model for the neighbors
Update the joint control policy
State(I5, I2,I4, I6, I8)
Reward(I5)

Figure 4-4 Illustrative Example of Agent Structure in MARLIN

4.1.4 MARLIN Framework

The MARLIN framework extends the definition of Markov games (Definition 2.3) to consider the case of a network of connected agents. Formally the framework is explained in Definition 4.3.

Definition 4.3: MARLIN Framework
MARLIN Framework is defined by the tuple: $(N, NB_1, \ldots, NB_N, S_1, \ldots, S_N, JS_1, \ldots, JS_N, A_1, \ldots, A_N, JA_1, \ldots, JA_N, R_1, \ldots, R_N,)$ N is the number of agents $\mathrm{NB_i}$ is a set of neighbours surrounding agent i S_i is a set of discrete local states for agent i $JS_i = S_i \times S_{\mathrm{NB_i[1]}} \times \ldots \times S_{\mathrm{NB_i[\|NB_i\|]}}$ is the joint state space observed by agent i A_i is a set of discrete local actions for agent i $JA_i = A_i \times A_{\mathrm{NB_i[1]}} \times \ldots \times A_{\mathrm{NB_i[\|NB_i\|]}}$ is the joint action space observed by agent i R_i is the reward function for agent i ri: $JS_i \times JA_i \rightarrow \mathbb{R}$

4.1.4.1 MARLIN Learning Approach

The learning approaches for MARLIN-IC and MARLIN-DC are discussed in the following sections.

4.1.4.1.1 MARLIN-IC Learning Approach

The following are the steps for the learning approach designed in MARLIN-IC that is described in a pseudo code in Algorithm 4.1:

- If there are $|\mathrm{NB_i}|$ neighbours for agent *i* with the joint state space JS_i and joint action space JA_i, there are $|\mathrm{NB_i}|$ partial state and action spaces for agent *i*. Each partial state space and action space consists of agent *i* and one of the neighbours $\mathrm{NB_i[j], s.t.\ j \in NB_i}$ $(\mathrm{S_i, S_{NB_i[j]}, A_i, A_{NB_i[j]}})$;
- Each agent *i* builds a model that estimates the policy for each of its neighbours and is represented by a matrix $M_{i,NB_i[j]}$, $\mathrm{s.t.\ j \in NB_i}$ where the rows are the joint states $\mathrm{S_i \times S_{NB_i[j]}}$ and the columns are the neighbour's actions $\mathrm{A_{NB_i[j]}}$. Each cell $M_{i,NB_i[j]}([\mathrm{s_i, s_{NB_i[j]}}], \mathrm{a_{NB_i[j]}})$ represents the probability that agent $NB_i[j]$ takes action $\mathrm{a_{NB_i[j]}}$ at the joint state $[\mathrm{s_i, s_{NB_i[j]}}]$.

$$M_{i,NB_i[j]}([s_i, s_{NB_i[j]}], a_{NB_i[j]}) = \frac{v_{NB_i[j]}([s_i, s_{NB_i[j]}], a_{NB_i[j]})}{\sum_{a \in A_{NB_i[j]}} v_{NB_i[j]}([s_i, s_{NB_i[j]}], a])} \qquad (4\text{-}1)$$

where $v_{NB_i[j]}([s_i^k, s_{NB_i[j]}^k], a_{NB_i[j]}^k)$ is a function which counts the number of visits agent $NB_i[j]$ visited the state $[s_i^k, s_{NB_i[j]}^k]$ after taking action $a_{NB_i[j]}^k$;

- Each agent *i* learns the optimal joint policy for agents *i* and $\mathrm{NB_i[j]}\ \forall\ \mathrm{j} \in \{1, \dots, |\mathrm{NB_i}|\}$ by updating the Q-values that are represented by a matix of $|\mathrm{S_i} \times\ \mathrm{S_{NB_i[j]}}|$ rows and $|\mathrm{A_i} \times \mathrm{A_{NB_i[j]}}|$ columns where each cell $Q_{i,NB_i[j]}([s_i, s_{NB_i[j]}], [a_i, a_{NB_i[j]}])$ represents the Q-value for a state-action pair in the partial spaces corresponding to the pair of connected agents $(\mathrm{i}, \mathrm{NB_i[j]})$;
- Each agent *i* updates Q-values $Q_{i,NB_i[j]}([s_i, s_{NB_i[j]}], [a_i, a_{NB_i[j]}])$ using the value of the best-response action taken in the next state. The best-response value (br_i) is the maximum expected Q-value at the next state, which is calculated using models for other agents.

$$br_i = max_{a\in A_i}\left[\textstyle\sum_{a'\in A_{NB_i[j]}} Q_{i,NB_i[j]}([s_i, s_{NB_i[j]}], [a, a'])\ \cdot M_{i,NB_i[j]}\left([s_i, s_{NB_i[j]}], a'\right)\right] \qquad (4\text{-}2)$$

Algorithm 4.1: MARLIN-IC Learning

```
Initialisation at time $k = 0$:
For each agent $i$, $i\epsilon\{1,2,\dots,N\}$:
    For each neighbour $j\epsilon\{1,2,\dots\ |NB_i|\}$
     Initialise $\mathrm{s_i^0}, \mathrm{a_i^0}, \mathrm{a_{NB_i[j]}^0}$
     $M^{\,0}_{i,NB_i[j]}([s_i, s_{NB_i[j]}], a_{NB_i[j]})=1/|\,A_{NB_i[j]}|$,   $Q^k_{i,NB_i[j]}([s_i, s_{NB_i[j]}], [a_i, a_{NB_i[j]}])=0$
    End for
End for
For each time step $k$ , do:
   For each agent $i$, $i\epsilon\{1,2,\dots,N\}$, do:
         For each neighbour $NB_i[j]$, $j\epsilon\{1,2,\dots\ |NB_i|\}$ do:
             a. Observe $\mathrm{a^k_{NB_i[j]}}$, $\mathrm{s_i^{k+1}}$ $\mathrm{s^{k+1}_{NB_i[j]}}$, and $\mathrm{r_i^k}$
             b. $v^k_{NB_i[j]}([s_i^k, s^k_{NB_i[j]}], a^k_{NB_i[j]}) = v^{k-1}_{NB_i[j]}([s_i^k, s^k_{NB_i[j]}], a^k_{NB_i[j]}) + 1$
             c. Update $M_{i,NB_i[j]}$
                $M^k_{i,NB_i[j]}([s_i^k, s^k_{NB_i[j]}], a^k_{NB_i[j]}) = \frac{v^k_{NB_i[j]}([s_i^k, s^k_{NB_i[j]}], a^k_{NB_i[j]})}{\sum_{a\in A_{NB_i[j]}} v^k_{NB_i[j]}([s_i^k, s^k_{NB_i[j]}], a)}$
             d. Choose the maximum expected Q-value at state $[s_i^{k+1}, s^{k+1}_{NB_i[j]}]$
        $br_i^k = \max_{a\in A_i}\left[\sum_{a'\in A_{NB_i[j]}} Q^k_{i,NB_i[j]}([s_i^{k+1}, s^{k+1}_{NB_i[j]}], [a, a'] \cdot M^k_{i,NB_i[j]}([s_i^{k+1}, s^{k+1}_{NB_i[j]}], a')\right]$
             e. Update $\alpha^k$
```

```
            $v_i^k([s_i^k, s_{NB_i[j]}^k], a_{\mathrm{i}}^k) = v_i^{k-1}([s_i^k, s_{NB_i[j]}^k], a_{\mathrm{i}}^k) + 1$
            $\alpha^k = \frac{\alpha_o}{v_i^k\left(\left[s_i^k, s_{NB_i[j]}^k\right], a_{\mathrm{i}}^k\right)}$
        f.  Update $Q_{i,NB_i[j]}$
    $Q_{i,NB_i[j]}^k([s_i^k, s_{NB_i[j]}^k], [a_i^k, a_{NB_i[j]}^k]) = (1-\alpha^k)Q_{i,NB_i[j]}^{k-1}([s_i^k, s_{NB_i[j]}^k], [a_i^k, a_{NB_i[j]}^k]) + \alpha[r_i^k + \gamma\ br_i^k]$
        g.  Choose $a_i^{k+1}$
        End For
    End For
End For
```

4.1.4.1.2 *MARLIN-DC Learning Approach*

The following are the steps for the learning approach designed in MARLIN-DC that is described in a pseudo code in Algorithm 4.2:

- If there are $|\mathrm{NB_i}|$ neighbours for agent *i* with the joint state space JS_i and joint action space JA_i, there are $|\mathrm{NB_i}|$ partial state and action spaces for agent *i*. Each partial state space and action space consists of agent *i* and one of the neighbours $\mathrm{NB_i[j], s.t.\ j \in NB_i}$ $(\mathrm{S_i, S_{NB_i[j]}, A_i, A_{NB_i[j]}})$;
- Each agent *i* starts with a random local policy $(\mathrm{a_i^{*0}})$ and exchanges this policy with its neighbours NBi;
- The agent learns the optimal joint policy with the neighbour $\mathrm{NB_i[j]}\ \forall\ \mathrm{j} \in \{1, \dots, |\mathrm{NB_i}|\}$ by updating the Q-values that are represented by a matix of $|\mathrm{S_i} \times\ \mathrm{S_{NB_i[j]}}|$ rows and $|\mathrm{A_i} \times \mathrm{A_{NB_i[j]}}|$ columns where each cell $Q_{i,NB_i[j]}([s_i, s_{NB_i[j]}], [a_i, a_{NB_i[j]}])$ represents the Q-value for a state-action pair in the partial spaces corresponding to the pair of connected agents $(\mathrm{i, NB_i[j]})$;
- Each agent *i* updates Q-values $Q_{i,NB_i[j]}([s_i, s_{NB_i[j]}], [a_i, a_{NB_i[j]}])$ using the value of the action that should be taken in the next state following the current policy and given the policy of the neighbouring agents.

Algorithm 4.1: MARLIN-DC Learning

```
Initialisation at time $k = 0$:
For each agent $i$, $i \epsilon \{1,2,\dots,N\}$:
    For each neighbour $j \epsilon \{1,2,\dots\ |NB_i|\}$
        Initialise $\mathrm{s_i^0, a_i^0, a_{NB_i[j]}^0},\ \mathrm{a_i^{*0}}, \mathrm{a}_{NB_i[j]}^{*0}$
        $Q^{k}_{i,NB_i[j]}([s_i, s_{NB_i[j]}], [a_i, a_{NB_i[j]}])=0$
      End for
End for
For each time step $k$ , do:
   For each agent $i$,  $i \epsilon \{1, 2, \ldots, N\}$, do:
   Broadcast the current policy $a_i^{*k}$
          For each neighbour $NB_i[j]$, $j \epsilon \{1,2, \ldots |NB_i|\}$ do:
               h.  Receive $a^{*k}_{NB_i[j]}$
               i.  Observe $s_i^{k+1}$ $s^{k+1}_{NB_i[j]}$, and $r_i^k$
               j.  Update $\alpha^k$
                   $v_i^k([s_i^k, s^k_{NB_i[j]}], a_i^k) = v_i^{k-1}([s_i^k, s^k_{NB_i[j]}], a_i^k) + 1$
                   $\alpha^k = \frac{\alpha_o}{v_i^k([s_i^k, s^k_{NB_i[j]}], a_i^k)}$
               k.  Update $Q_{i,NB_i[j]}$
               $Q^k_{i,NB_i[j]}([s_i^k, s^k_{NB_i[j]}], [a_i^k, a^k_{NB_i[j]}])$
                   $= (1-\alpha^k) Q^{k-1}_{i,NB_i[j]}([s_i^k, s^k_{NB_i[j]}], [a_i^k, a^k_{NB_i[j]}])$
                   $+ \alpha \left[ r_i^k + \gamma \sum_{j \in \{1,2,\ldots |NB_i|\}} Q^k_{i,NB_i[j]}([s_i^{k+1}, s^{k+1}_{NB_i[j]}], [a_i^{*k}, a^{*k}_{NB_i[j]}] \right]$

               l.  Update  $a_i^{*k+1}$
                 End For
          End For
End For
```

4.1.4.2 *MARLIN Decision Making Approaches*

The decision making approach can be implemented through indirect coordination (IC) or direct coordination (DC) mechanisms. The decision making approaches for MARLIN-IC and MARLIN-DC are discussed in the following sections.

4.1.4.2.1 *MARLIN-IC Decision Making Approach*

The pseudo code for the decision making approach in MARLIN-IC is presented in Algorithm 4.2. In MARLIN-IC the agent decides its action without direct interaction with the neighbours. Instead, the agent uses the estimated models for the other agents and acts accordingly. Agent *i* chooses the next action using a simple heuristic decision procedure which biases the action selection toward actions that have the maximum expected Q-value over its neighbours NB_i. The likelihood of Q-values is evaluated using the models of the other agents estimated in the learning process, as illustrated in Algorithm 4.2.

Algorithm 4.2: MARLIN-IC Decision Making Approach

```
For each time step $k$ , do:
    For each agent $i$, $i \epsilon \{1, 2, \dots, N\}$, do:
        If agent i exploits, then
$a_i^{k+1} = \underset{a \in A_i}{argmax} \left[ \sum_{j \epsilon \{1,2,..,|NB_i|\}} \sum_{a' \in A_{NB_i[j]}} Q_{i,NB_i[j]}^k([s_i^{k+1}, s_{NB_i[j]}^{k+1}], [a, a']) \cdot M_{i,NB_i[j]}^k([s_i^{k+1}, s_{NB_i[j]}^{k+1}], a') \right]$
        Else (agent i explores)
            $a_i^{k+1}$=random action a∈ $A_i$.
            End For
End For
```

4.1.4.2.2 MARLIN-DC Decision Making Approach

In MARLIN-DC, the agent generates the next action by negotiating and directly interacting with its neighbours. Then the agent calculates its utility (U_c) with respect to its current policy and its neighbours' policies. The agent also calculates the utility of its best-response policy (U_{br}) given the policies of its neighbours. The difference between the two utilities ($U_{br} – U_c$) represents a *gain message*. The agent broadcasts its gain message to its neighbours and receives their gain messages. The agent then improves its policy if its gain message is higher than all the gain messages received from its neighbours (i.e. if the subject agent is the winner). If the agent is the winner in the current cycle of the algorithm, it changes its policy to the best policy and broadcasts it to the neighbours. This process may be repeated until all connected agents change their policies. The pseudo code for the MARLIN-DC is presented in Algorithm 4.3.

Algorithm 4.3: MARLIN-DC Decision Making Approach

```
For each time step $k$, do:
For each agent $i$, $i \epsilon \{1,2,\dots,N\}$:
    For each neighbour $j \epsilon \{1,2,\dots |NB_i|\}$
        $Gain\ (i) = 0, Gain\ (NB_i(\mathrm{j})) = 0, Done\ (i) = 0; Done(NB_i(\mathrm{j})) = 0$
    End For
End For
For each time step $k$, do:
   For each agent $i$, $i \epsilon \{1,2,\dots,N\}$, do:
      If agent i exploits, then
         Compute the gain message given the current policy $\mathrm{a}_{\mathrm{i}}^{*\mathrm{k}}$ and the received neighbours' policies $\mathrm{a}_{NB_i(\mathrm{j})}^{*\mathrm{k}}$
            $U_{br} = max_{a \in A_i} \sum_{j \epsilon \{1,2,..,|NB_i|\}} Q_{i,NB_i[j]}^k \left(\left[s_i^{k+1}, s_{NB_i[j]}^{k+1}\right], \left[a, a_{NB_i(j)}^{*k}\right]\right)$
            $U_c = \sum_{j \epsilon \{1,2,..,|NB_i|\}} Q_{i,NB_i[j]}^k \left(\left[s_i^{k+1}, s_{NB_i[j]}^{k+1}\right], \left[a_i^{*k}, a_{NB_i(j)}^{*k}\right]\right)$
            $Gain\ (i) = [U_{br} - U_c]$
         Broadcast Gain (i) and receive Gain $(NB_i(j))$, $\forall$j $\in \{1,2,\dots |NB_i|\}$
         Update agent i's policy and choose next action
         If (Gain (i) $\geq$ Gain $(NB_i(j))$ or Done (j) $= 1$ $\forall$ j $\in \{1,2,\dots |NB_i|\}$),
               $\mathrm{a}_{\mathrm{i}}^{\mathrm{k+1}} = \mathrm{a}_{\mathrm{i}}^{*\mathrm{k+1}} = \underset{a \in A_i}{argmax} \sum_{j \epsilon \{1,2,..,|NB_i|\}} Q_{i,NB_i[j]}^k \left(\left[s_i^{k+1}, s_{NB_i[j]}^{k+1}\right], \left[a, a_{NB_i(j)}^{*k}\right]\right)$
                                        Done (i) $= 1$
      Else (agent i explores)
               $\mathrm{a}_{\mathrm{i}}^{\mathrm{k+1}}$=random action a$\in$ $\mathrm{A}_{\mathrm{i}}$
         End If
   End For
End For
```

4.2 MARLIN-ATSC PLATFORM

The MARLIN-ATSC platform is illustrated in Figure 4-5. The platform consists of two main layers; the first layer is an input configuration layer that is responsible for configuring and providing the necessary input to the second layer. The second layer is a control layer that includes three interacting components (as shown in Figure 4-5):

- ***Agent***: implements the control algorithm;
- ***Simulation Environment***: models the traffic environment;

- ***Interface***: facilitates the interaction between the agent and the traffic simulation environment.

Each of the above layers is described below in detail.

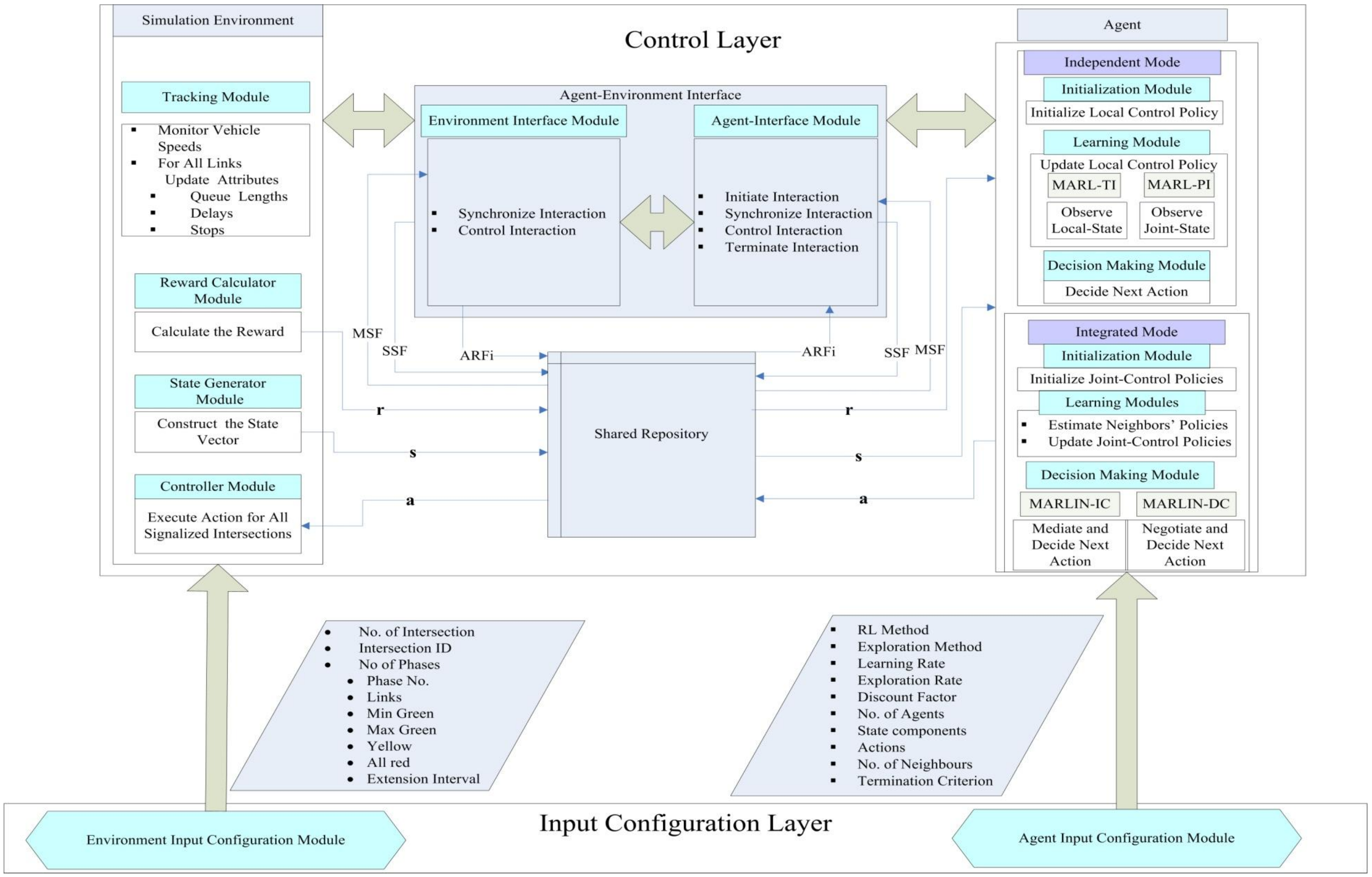


Figure 4-5 MARLIN -ATSC Platform

4.2.1 Configuration Layer

The configuration layer has two main roles: 1) configures the simulation-based learning environment (model) such that the simulated environment closely matches the real-world environment, 2) configures the RL-design parameters. Five main tasks are conducted in this layer:

1. Define the area of control and build the simulation model in Paramics;
2. Collect real traffic counts for each intersection in the area under control;
3. Design the phasing scheme based on the observed turning-counts;
4. Estimate an Origin-Destination (OD) matrix in Paramics using the Paramics Estimator. The following elements are integrated in one platform to perform the OD-Estimation (refer to Figure 4-6):
    a. Model of the Environment (traffic network representation, i.e. Paramics network)
    b. Input Data (turning movement counts at each intersection from the field)
    c. Optimisation Logic (minimise distance between modelled and observed counts, i.e. minimise GEH)
    d. Method of Traffic Assignment (dynamic traffic assignment)
    e. Convergence Criterion (number of iterations);
5. Select the RL-Design parameters (state definition, action definition, reward definition, learning rate, discount factor, exploration rate, etc.).

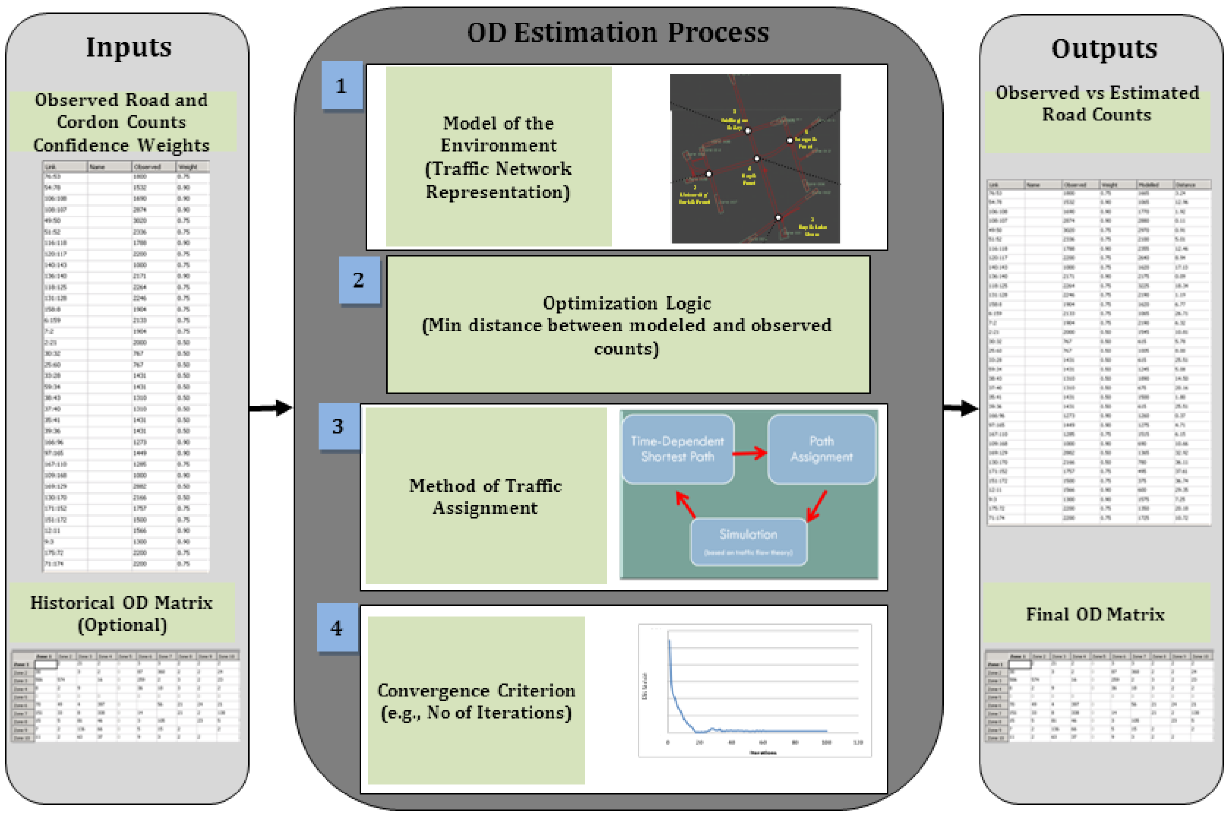

Inputs
Observed Road and Cordon Counts Confidence Weights
Historical OD Matrix (Optional)
OD Estimation Process
1
Model of the Environment (Traffic Network Representation)
2
Optimization Logic (Min distance between modeled and observed counts)
3
Method of Traffic Assignment
Time-Dependent Shortest Path
Path Assignment
Simulation
4
Convergence Criterion (e.g., No of Iterations)
Outputs
Observed vs Estimated Road Counts
Final OD Matrix

Figure 4-6 OD-Estimation Process

By performing the five steps above, the inputs for the second layer will be ready. The control layer components are designed to be generic such that all the problem-related parameters are exogenous inputs to the platform. Two groups of parameters represent the inputs to the control layer: Agent component's parameters and Simulation Environment component's parameters. The input parameters for the *Agent* component include general RL-related parameters and problem-specific-RL-related parameters, while the inputs for the Simulation Environment component include parameters related to the network and the characteristics of each intersection. The detailed input parameters for each component are listed below:

- ***General-RL-Related Parameters:***
  - Learning Method (1 for Q-learning, 2 for SARSA, 3 for Q(λ), 4 for SARSA(λ), 5 for MARLIN-IC, 6 for MARLIN-DC)
  - Exploration Method (1 for ε-greedy, 2 for softmax, 3 for ε-softmax)
  - Learning Rate Initial Value
  - Discount Rate

   - Exploration Rate Initial Value
   - Termination Criterion (number of iterations).
- ***Problem-Specific-RL-Related Parameters***
   - State Space (define the state components and the discretization intervals for each component)
   - Action Space (define the possible actions).
- ***ATSC-Simulation Environment -Related Parameters:***
   - Intersection Name and Node Number
   - Specification of Phases (define the number of associated lanes for each phase, link name and lane number for each associated lane)
   - Timings (define minimum green time, yellow time, all red for each phase according to the number of lanes in each phase).

4.2.2 Control Layer

As shown in Figure 4-5, the control layer has three main interacting components to perform the control task: 1) Agent, 2) Simulation Environment, and 3) Interface. Each component is described below in detail.

4.2.2.1 Agent

The agent is the learner and the decision maker that interacts with the environment by first receiving the system’s state and the reward and then selecting an action accordingly. A generic agent model is developed using Java Programming Language such that different levels of coordination, learning methods, state representations, phasing sequence, reward definition, and action selection strategies can be tested for any control task.

The agent in MARLIN-ATSC implements two modes of coordination as discussed in section 4.1. The agent structure for each mode of operation consists of three modules: Initialisation Module, Learning Module, and Decision Making Module as shown in Figure 4-5.

4.2.2.1.1 MARLIN-ATSC Independent Mode (MARL-TI and MARL-PI)

In this mode of coordination, agents implement independent learning and decision making processes.

4.2.2.1.1.1 Initialisation Module

This module is important to set the starting/reference agent's control policy. The inputs for this module are the RL-related parameters (e.g. learning method, exploration method, initial learning rate (α_o), discount factor, exploration rate, state space, action space) that are passed from the configuration layer as shown in Figure 4-5. In the initialisation module, the agent creates and initialises the variables corresponding to the input learning method as shown in Algorithms 3.2, 3.3, 3.4 and 3.5 (e.g. initialise Q-table). The initial values of the inputs to this module can optionally allow the agent to start with a previously learned policy. The initialisation module is also responsible for measuring the initial state and choosing the initial action.

4.2.2.1.1.2 Learning Module

The MARLIN-ATSC Independent Mode implements an independent agent that learns at the local level. The inputs for the learning module are: 1) the environment's state corresponding to the subject agent ($s \in S$ for MARL-TI, and $s \in JS$ for MARL-PI, where S is a set of discrete local states for the agent and JS is the joint state space observed by the agent), and 2) the reward value resulting from the action taken at the last state. The learning module works to update the agent's control policy to be used by the decision making module to choose the next action. The learning module in independent mode is capable of implementing four TD learning algorithms to update the control policy, Q-learning, SARSA, Q(λ), and SARSA(λ) as illustrated in Figure 4-7 (refer to section 3.1.4.2 for more details on each algorithm).

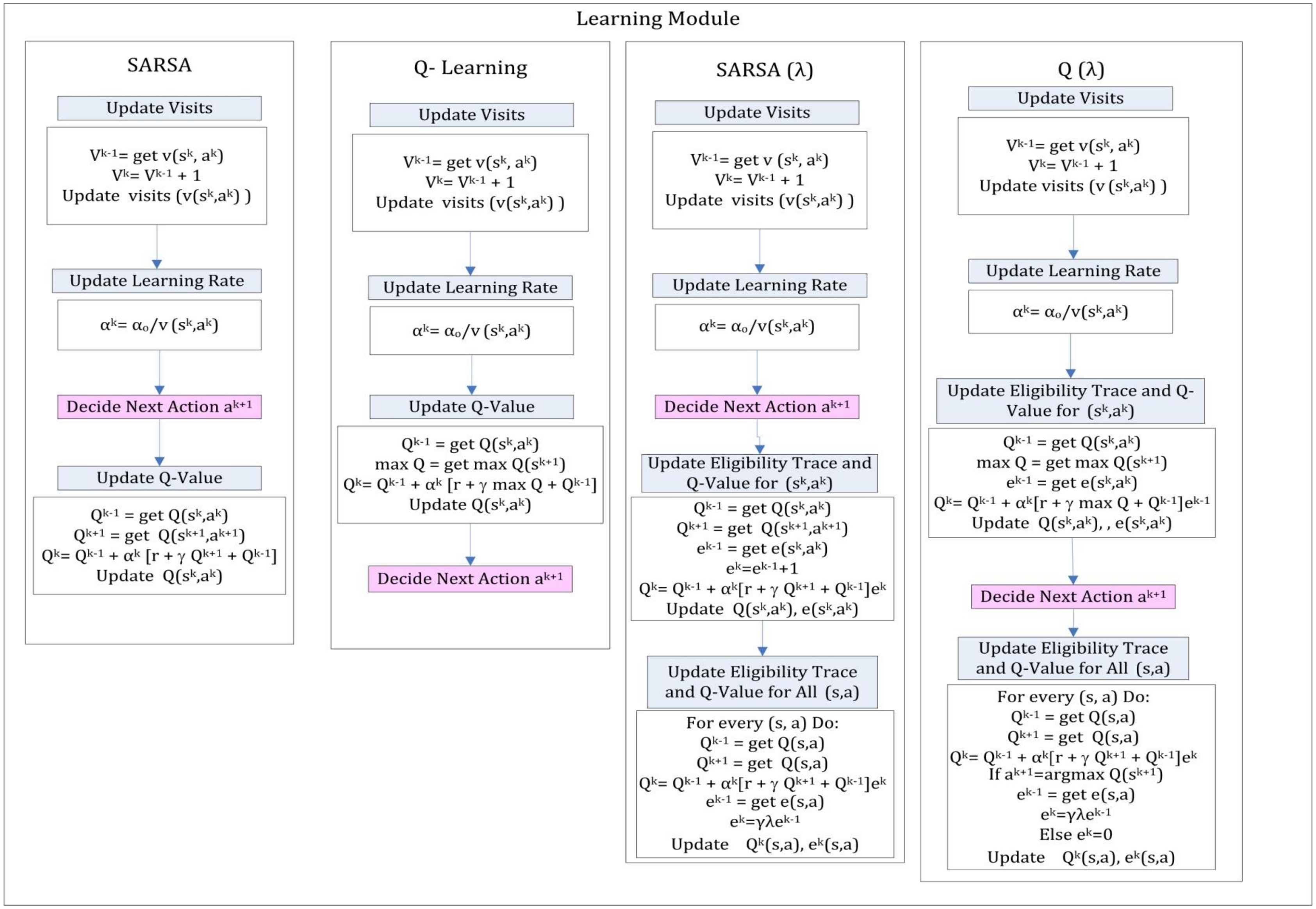

Learning Module
SARSA
Update Visits
V^{k-1}= get v(s^k, a^k)
V^k= V^{k-1} + 1
Update visits (v(s^k,a^k))
Update Learning Rate
$\alpha^k = \alpha_o / v(s^k,a^k)$
Decide Next Action a^{k+1}
Update Q-Value
Q^{k-1} = get Q(s^k,a^k)
Q^{k+1} = get Q(s^{k+1},a^{k+1})
$Q^k = Q^{k-1} + \alpha^k [r + \gamma Q^{k+1} + Q^{k-1}]$
Update Q(s^k,a^k)
Q- Learning
Update Visits
V^{k-1}= get v(s^k, a^k)
V^k= V^{k-1} + 1
Update visits (v(s^k,a^k))
Update Learning Rate
$\alpha^k = \alpha_o / v(s^k,a^k)$
Update Q-Value
Q^{k-1} = get Q(s^k,a^k)
max Q = get max Q(s^{k+1})
$Q^k = Q^{k-1} + \alpha^k [r + \gamma \max Q + Q^{k-1}]$
Update Q(s^k,a^k)
Decide Next Action a^{k+1}
SARSA (λ)
Update Visits
V^{k-1}= get v (s^k, a^k)
V^k= V^{k-1} + 1
Update visits (v(s^k,a^k))
Update Learning Rate
$\alpha^k = \alpha_o / v(s^k,a^k)$
Decide Next Action a^{k+1}
Update Eligibility Trace and Q-Value for (s^k,a^k)
Q^{k-1} = get Q(s^k,a^k)
Q^{k+1} = get Q(s^{k+1},a^{k+1})
e^{k-1} = get e(s^k,a^k)
$e^k = e^{k-1} + 1$
$Q^k = Q^{k-1} + \alpha^k [r + \gamma Q^{k+1} + Q^{k-1}] e^k$
Update Q(s^k,a^k), e(s^k,a^k)
Update Eligibility Trace and Q-Value for All (s,a)
For every (s, a) Do:
Q^{k-1} = get Q(s,a)
Q^{k+1} = get Q(s,a)
$Q^k = Q^{k-1} + \alpha^k [r + \gamma Q^{k+1} + Q^{k-1}] e^k$
e^{k-1} = get e(s,a)
$e^k = \gamma \lambda e^{k-1}$
Update Q^k(s,a), e^k(s,a)
Q (λ)
Update Visits
V^{k-1}= get v(s^k, a^k)
V^k= V^{k-1} + 1
Update visits (v (s^k,a^k))
Update Learning Rate
$\alpha^k = \alpha_o / v(s^k,a^k)$
Update Eligibility Trace and Q-Value for (s^k,a^k)
Q^{k-1} = get Q(s^k,a^k)
max Q = get max Q(s^{k+1})
e^{k-1} = get e(s^k,a^k)
$Q^k = Q^{k-1} + \alpha^k [r + \gamma \max Q + Q^{k-1}] e^{k-1}$
Update Q(s^k,a^k), , e(s^k,a^k)
Decide Next Action a^{k+1}
Update Eligibility Trace and Q-Value for All (s,a)
For every (s, a) Do:
Q^{k-1} = get Q(s,a)
Q^{k+1} = get Q(s,a)
$Q^k = Q^{k-1} + \alpha^k [r + \gamma Q^{k+1} + Q^{k-1}] e^k$
If a^{k+1}=argmax Q(s^{k+1})
e^{k-1} = get e(s,a)
$e^k = \gamma \lambda e^{k-1}$
Else e^k=0
Update Q^k(s,a), e^k(s,a)

Figure 4-7 MARLIN-ATSC Independent Mode Learning Module

4.2.2.1.1.3 Decision Making Module

The decision making module is responsible for selecting the next action at each learning step as shown in Figure 4-8. In the MARLIN-ATSC Independent Mode, the agent individually decides its action without collaboration with other agents. The inputs for this module are the environment's state, agent's current policy and the selection strategy, while the output is the agent's next action. The Decision Making Module in the independent mode (Figure 4-8) is capable of implementing three exploration (action selection) strategies; ε-greedy, softmax, and ε-softmax. A general description of each exploration method is presented below.

ε-greedy

In each iteration, the ε-greedy method is designed such that the agent selects the greedy action most of the time except for ε amount of time, it selects a random action uniformly (Sutton and Barto, 1998).

Softmax

One disadvantage of the ε-greedy selection method is that it treats all exploratory actions equally, irrespective of the estimated value function of each action; which means that it is as likely to select the worst action as it is to select the next-to-best action. In real-life applications this disadvantage could adversely affect the system, particularly when the worst action can be drastic.

To overcome this limitation, exploration can be concentrated on the most promising actions in terms of their estimated values. Thus the selection of the action probabilities can be represented by a proportional function to the estimated values, which is referred to as the softmax action selection strategy (Sutton and Barto, 1998). The most common softmax methods use Boltzman distribution and result in the following probability of action selection:

$$P_s(a) = \frac{e^{\frac{Q(s,a)}{\tau}}}{\sum_{b\in A} e^{\frac{Q(s,b)}{\tau}}} \tag{4-3}$$

The parameter τ is called the temperature. When τ is large each action will have approximately the same probability of being selected (i.e. more exploration). When τ is small, actions will be selected proportionally to their estimated payoff (i.e. more exploitation).

Mixed ε-Greedy and Softmax (ε-Softmax)

The choice rule suggested by Wahba (2008) follows a mixture of ε-greedy and softmax action-choice models, with a 1-ε probability of exploitation and ε probability of exploration:

when exploiting,

$$P_s(a) = \begin{cases} 1 & Q(s,a) = max_{a' \in A} Q(s,a') \\ 0 & otherwise \end{cases} \tag{4-4}$$

and when exploring

$$P_s(a) = \frac{e^{Q(s,a)/\tau}}{\sum_{b \in A} e^{Q(s,b)/\tau}} \tag{4-5}$$

In this method, the greedy action is still given the highest selection probability, and all other actions are weighted according to their estimated values.

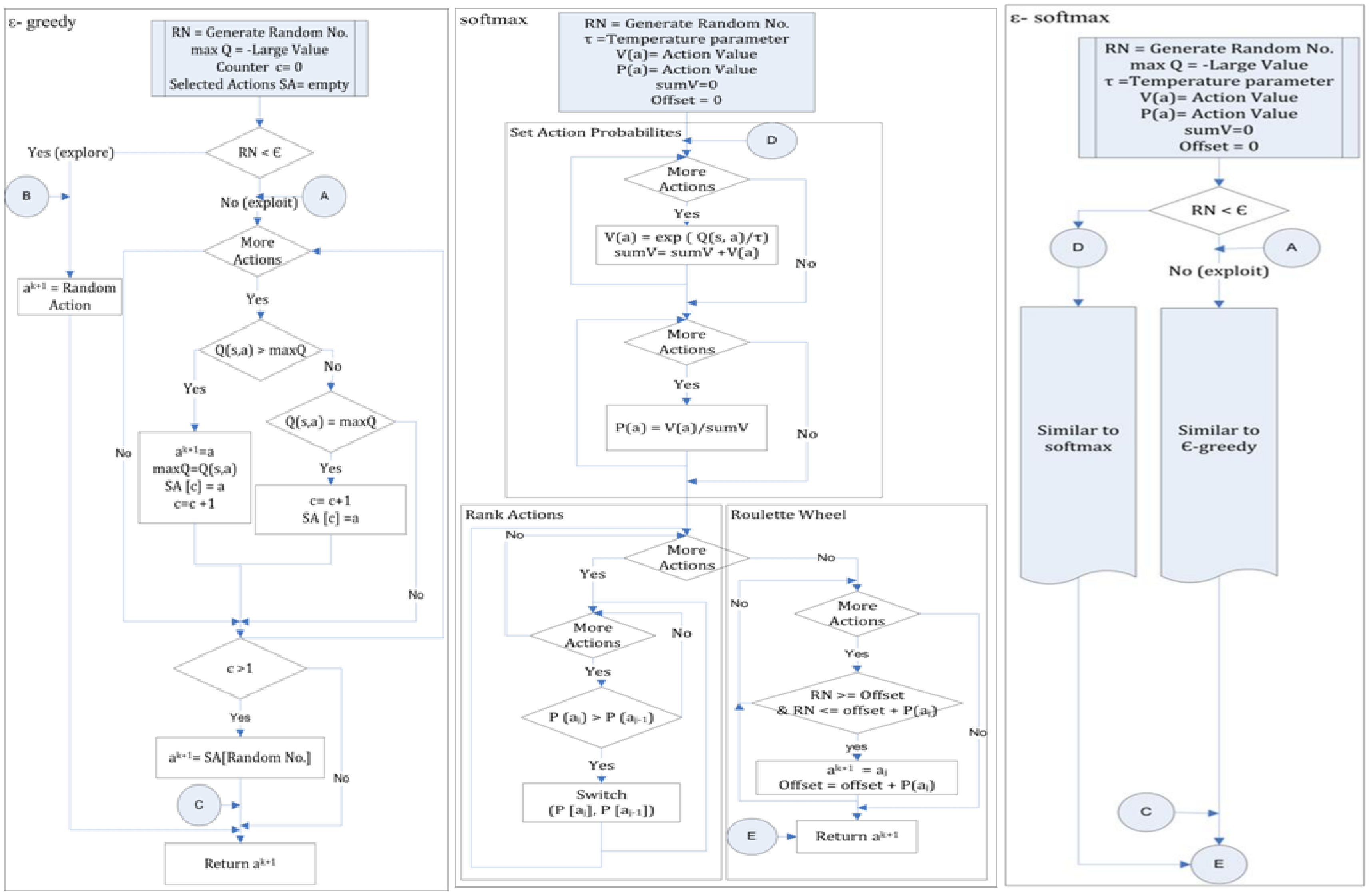


Figure 4-8 MARLIN-ATSC Independent Mode Decision Making Module

4.2.2.1.2 MARLIN-ATSC Integrated Mode (MARLIN-IC and MARLIN-DC)

4.2.2.1.2.1 Initialisation Module

Similar to the independent mode, the initialisation module in the integrated module is important to set the starting/reference point of the joint-control policies. The inputs for this module are the RL-related parameters (e.g. coordination method, initial learning rate (α_o), discount factor, exploration rate, state space, action space, set of neighbours) that are passed from the configuration layer, as shown in Figure 4-5. In the initialisation module, the agent creates and initialises the variables corresponding to the input learning method as shown in Algorithms 4.2 and 4.3 (e.g. initialise Q-tables). The initial values optionally allow the agent to start with previously learned joint-policies. The initialisation module is also responsible for measuring the initial state, and choosing the initial action.

4.2.2.1.2.2 Learning Modules

The MARLIN-ATSC Integrated Mode implements explicit coordination with the neighbours by implementing the MARLIN Learning Algorithm as shown in Figure 4-9 (refer to Algorithms 4.1 and 4.2). For agent i there are $|\mathrm{NB_i}|$ learning modules, as illustrated in Example 4.2. Each module corresponds to an agent i and one of the neighbours $\mathrm{NB_i}(\mathrm{j})$. The inputs for each learning module j ($\mathrm{j} \in \{1,2, \dots, |\mathrm{NB_i}|\}$) is the state corresponding to agent i and agent $\mathrm{NB_i}(\mathrm{j})$, the last action taken by $\mathrm{NB_i}(\mathrm{j})$, and the reward value resulting from the action taken in the last state. The main task of the learning module is to update the agent's current joint-policy (Q-values).

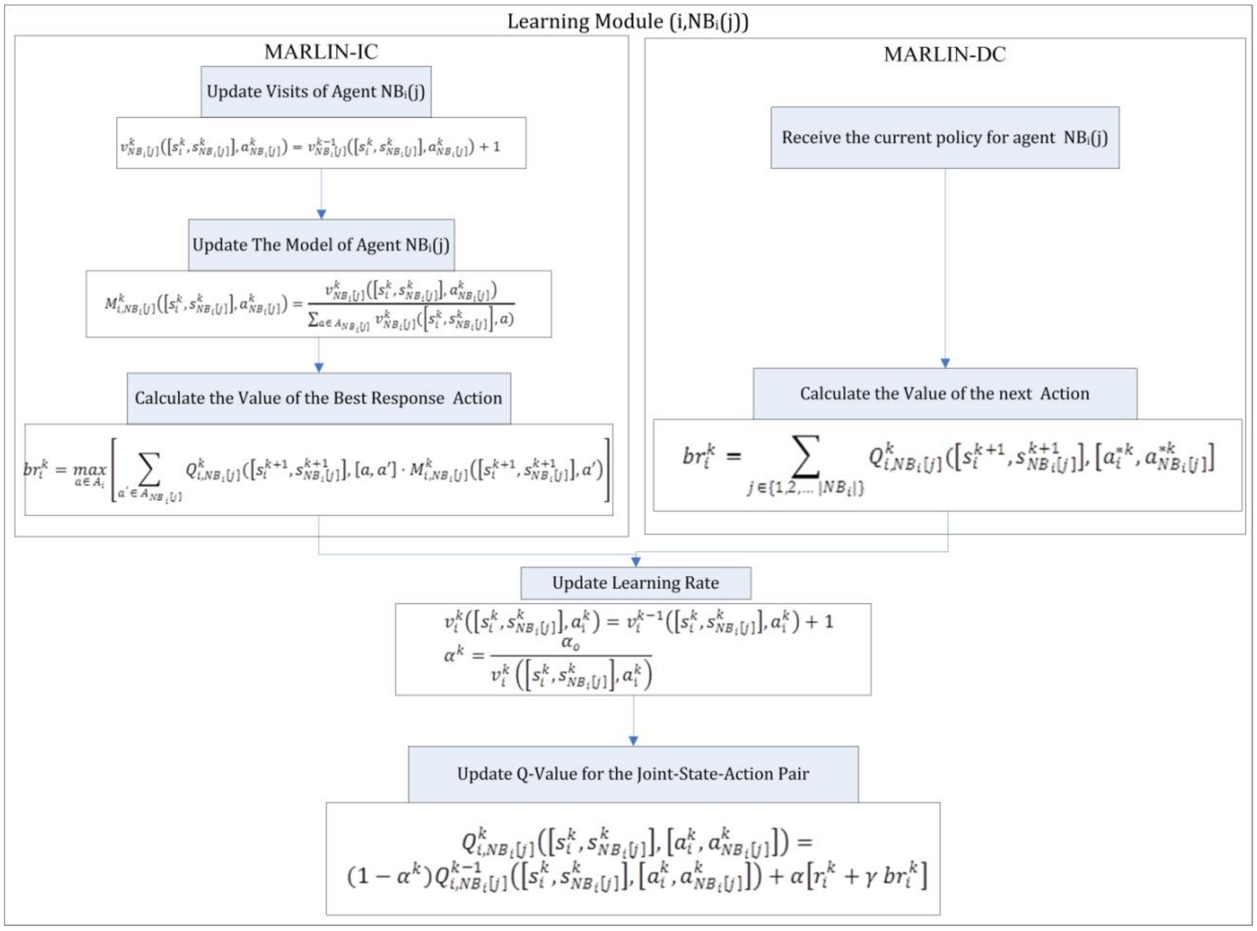

Learning Module (i,NBi(j))
MARLIN-IC
Update Visits of Agent NBi(j)
Update The Model of Agent NBi(j)
Calculate the Value of the Best Response Action
MARLIN-DC
Receive the current policy for agent NBi(j)
Calculate the Value of the next Action
Update Learning Rate
Update Q-Value for the Joint-State-Action Pair

Figure 4-9 MARLIN-ATSC Integrated Mode Learning Module of Agent i and Neighbour $\boldsymbol{NB_i}(\boldsymbol{j})$

4.2.2.1.2.3 Decision Making Module

The inputs for this module are the environment's state observed by agent i, the joint policies for agent i and each one of its neighbours, and the selection strategy. The output of this module is the agent's next action. The decision making module implements two action selection strategies as shown in Figure 4-10; MARLIN-IC (Algorithm 4.2), and MARLIN-DC (Algorithm 4.3).

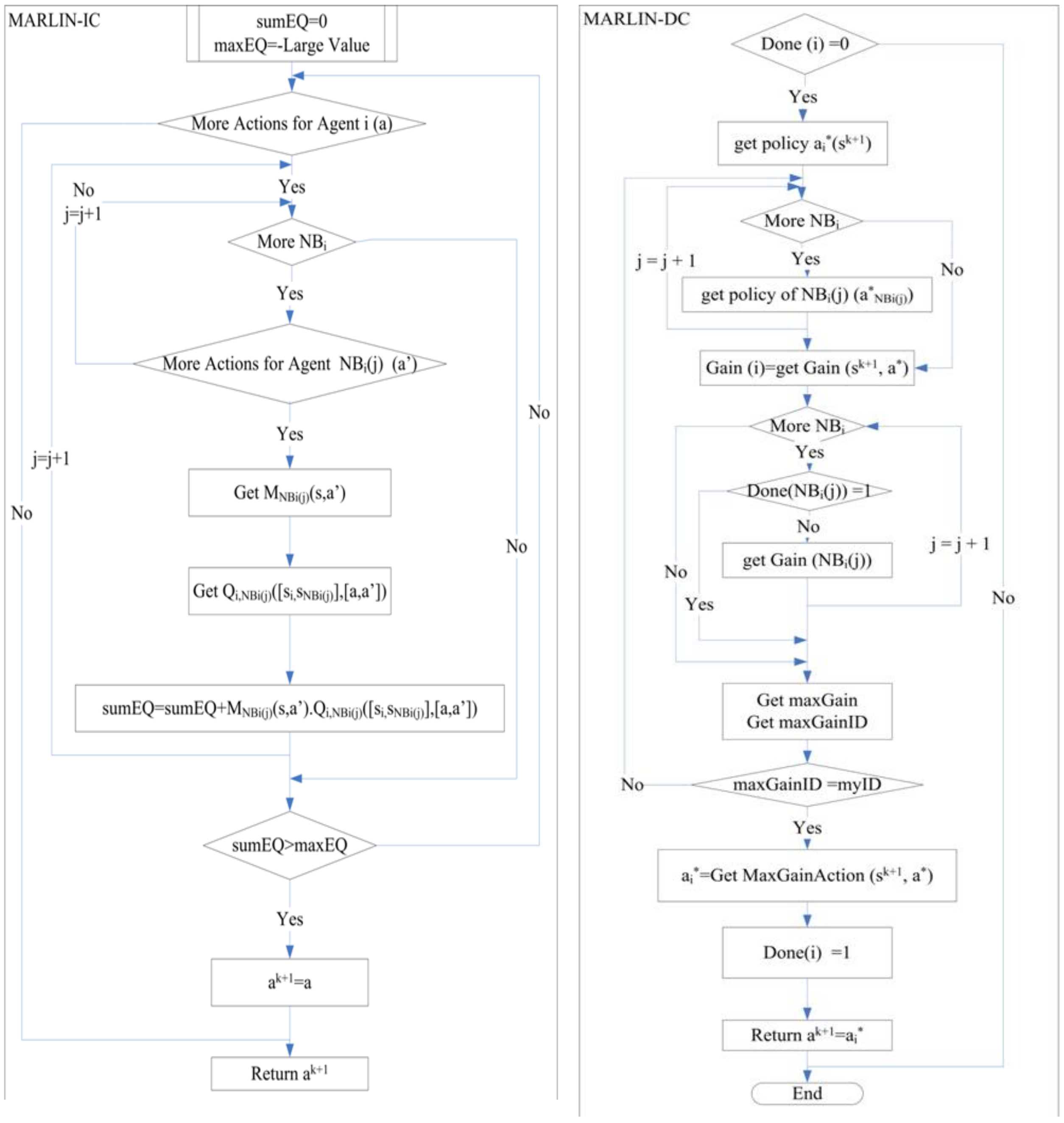
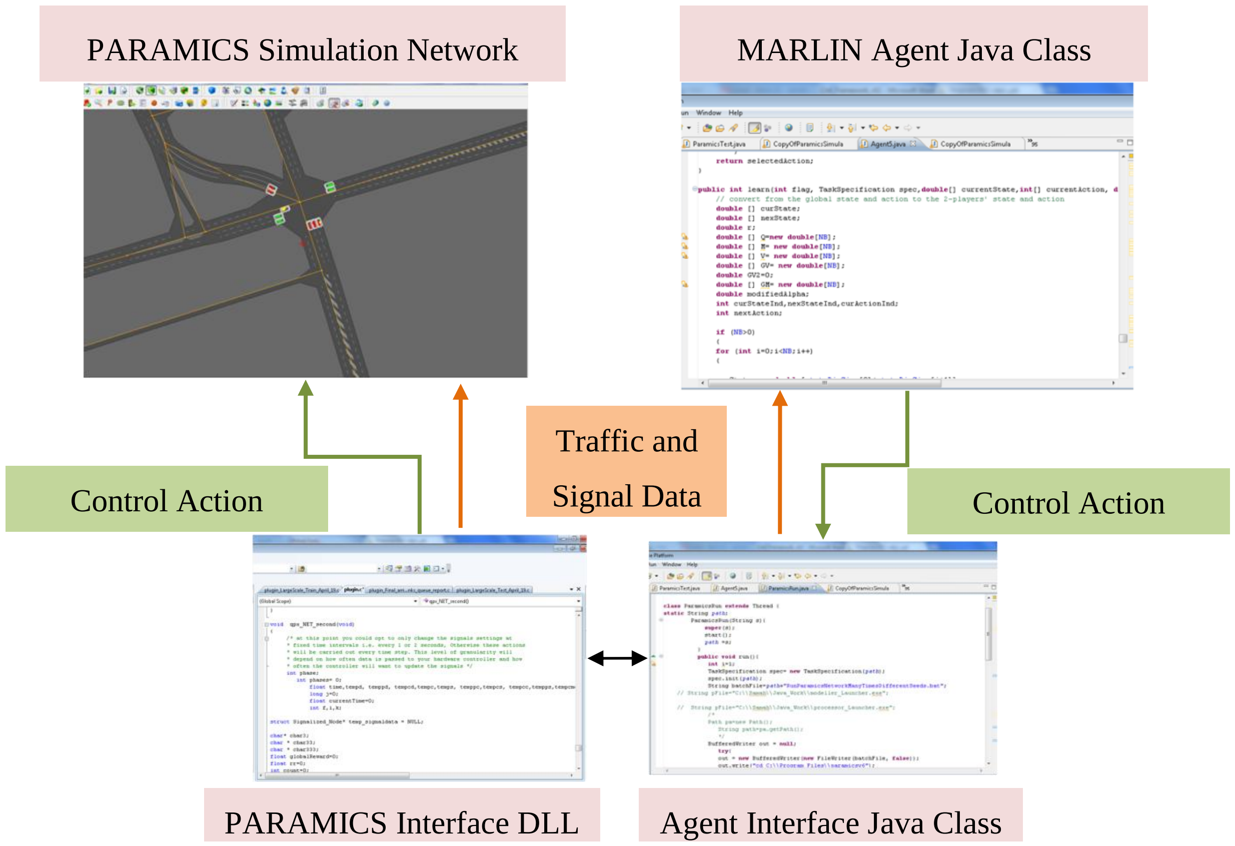

MARLIN-IC
sumEQ=0
maxEQ=-Large Value
More Actions for Agent i (a)
Yes
No
j=j+1
More NB$_i$
Yes
More Actions for Agent NB$_i$(j) (a')
Yes
j=j+1
No
Get M$_{NBi(j)}$(s,a')
Get Q$_{i,NBi(j)}$([s$_i$,s$_{NBi(j)}$],[a,a'])
No
No
sumEQ=sumEQ+M$_{NBi(j)}$(s,a').Q$_{i,NBi(j)}$([s$_i$,s$_{NBi(j)}$],[a,a'])
sumEQ>maxEQ
Yes
a^{k+1}=a
Return a^{k+1}
MARLIN-DC
Done (i) =0
Yes
get policy a$_i^*$(s^{k+1})
More NB$_i$
Yes
j = j + 1
No
get policy of NB$_i$(j) (a$^*_{NBi(j)}$)
Gain (i)=get Gain (s^{k+1}, a*)
More NB$_i$
Yes
Done(NB$_i$(j)) =1
No
get Gain (NB$_i$(j))
j = j + 1
No
Yes
No
Get maxGain
Get maxGainID
No
maxGainID =myID
Yes
a$_i^*$=Get MaxGainAction (s^{k+1}, a*)
Done(i) =1
Return a^{k+1}=a$_i^*$
End

Figure 4-10 MARLIN-ATSC Integrated Mode Decision Making Module

4.2.2.2 Agent-Environment Interface

The interface component has two modules; an Agent interface module developed using *Java* programming language and a simulation environment module developed using the API in Paramics, as shown in Figure 4-11. These modules manage the interactions between the agent and the simulation environment by exchanging the state, reward, and action.



```
        return selectedAction;
    }

public int learn(int flag, TaskSpecification spec,double[] currentState,int[] currentAction, d
        // convert from the global state and action to the 2-players' state and action
        double [] curState;
        double [] nexState;
        double r;
        double [] Q=new double[NB];
        double [] M= new double[NB];
        double [] V= new double[NB];
        double [] GV= new double[NB];
        double GV2=0;
        double [] GM= new double[NB];
        double modifiedAlpha;
        int curStateInd,nexStateInd,curActionInd;
        int nextAction;

        if (NB>0)
        {
        for (int i=0;i<NB;i++)
        {
```

```
 }

void  qpx_NET_second(void)
{
        /* at this point you could opt to only change the signals settings at
        * fixed time intervals i.e. every 1 or 2 seconds, Otherwise these actions
        * will be carried out every time step. This level of granularity will
        * depend on how often data is passed to your hardware controller and how
        * often the controller will want to update the signals */
        int phase;
            int phases= 0;
                float time,tempd, temppd, tempcd,tempc,temps, temppc,tempcs, tempcc,tempps,tempcm
                long j=0;
                float currentTime=0;
                int f,i,k;

struct Signalized_Node* temp_signaldata = NULL;

char* char3;
char * char33;
char * char333;
float globalReward=0;
float rr=0;
int count=0;
```

```
class ParamicsRun extends Thread {
static String path;
        ParamicsRun(String s){
            super(s);
            start();
            path =s;
        }
        public void run(){
            int i=1;
            TaskSpecification spec= new TaskSpecification(path);
            spec.init(path);
            String batchFile=path+"RunParamicsNetworkManyTimesDifferentSeeds.bat";
    // String pFile="C:\\Sameh\\Java_Work\\modeller_Launcher.exe";

    //  String pFile="C:\\Sameh\\Java_Work\\processor_Launcher.exe";
            /*
            Path pa=new Path();
                String path=pa.getPath();
                */
            BufferedWriter out = null;
                try{
                out = new BufferedWriter(new FileWriter(batchFile, false));
                out.write("cd C:\\Program Files\\paramicsv6");
```

Figure 4-11 Agent-Environment Interface in MARLIN-ATSC

Both modules facilitate and synchronise the interaction between the agent and the specific simulation environment. In the traffic control problem, the interaction between the agent and the environment is associated with the following challenges:

- ***Agent-Environment Synchronised Interaction***

In traffic control problems, the interaction between an agent and the environment is challenging because the simulation environment is continuously running. Hence, there is a need for synchronised interaction between the agent and the environment to ensure that the simulation environment is *held* while the agent is performing the learning and the decision making processes, and finally produces the action that should be executed by the simulation environment. At the same time, the agent should be *on hold* until the action is executed in the environment and the resultant state and reward are measured. In order to achieve these synchronised interactions, a shared memory (or shared repository (SR)) technique has been used in which two flags (one for each component) are assigned; Environment Synchronisation Flag (ESF) and Agent Synchronisation Flag (ASF) as shown in Figure 4-5.

- ***Interaction Frequency***

Another challenge associated with the interaction between the agent and the environment is the determination of the frequency of interaction. It is known that the higher the frequency of interaction the more adaptive is the system. This is true for systems that do not require the setting of a minimum or maximum constraint on the frequency of interaction. In the case of adaptive signal control the minimum green time constraint (safety constraint for pedestrians) must be satisfied, and then the frequency of the interaction comes into effect. The system is designed such that the interaction frequency is variable for each agent. The interaction occurs at each specified time interval (1 sec in this research) as long as the current green for a signalised intersection that is associated with an agent i exceeded the minimum green time. Otherwise, the interaction starts after the minimum green. The interaction frequency is achieved using a shared-memory technique in which a flag, associated with each signalised intersection (agent), is assigned to indicate whether or not the agent is allowed to perform interaction (Admissible Region Flag (ARF)).

4.2.2.2.1 Agent-Interface Module

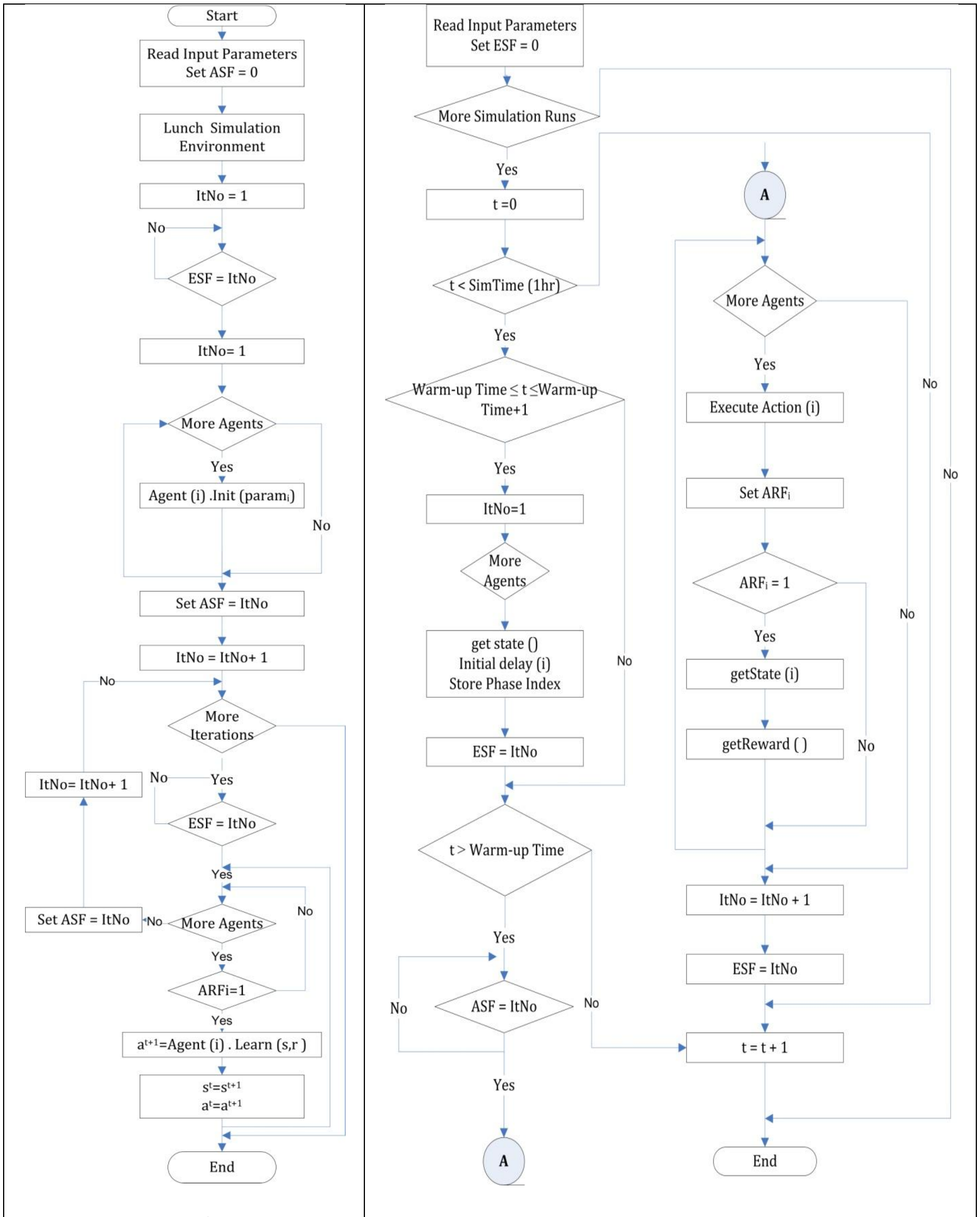
This module manages the interactions between an agent and the simulation environment from the agent's perspective, as shown in Figure 4-12-a. It performs the following functions in order:

- Initiate the interaction: after processing the input parameters, the interaction is initiated by launching the environment simulation and initiating the learning process;
- Synchronise the interaction: the interaction is synchronised at each learning step using flags such as ASF and ESF as discussed above;
- Control the interaction: the interaction is limited for agents that are ready to update their policies and decides the next action as aforementioned;
- Terminate the interaction: the interaction is terminated after satisfying the stopping criteria (e.g. maximum number of simulation runs).

4.2.2.2.2 Environment-Interface module

Similar to the Agent Interface Module, this module manages the interactions between an agent and the simulation environment but from the perspective of the environment, as shown in Figure 4-12-b. In this module the following functions are performed:

- Synchronise the interaction: the interaction is synchronised each simulation second using flags such as ASF and ESF;
- Control the interaction: the interaction is limited for intersections that are ready to change their current signal settings.



(a) Agent Interface Module (b) Environment Interface Module

Figure 4-12 Agent-Environment Interface Modules

4.2.2.3 Traffic Simulation Environment

Paramics, a microscopic traffic simulator, is used as the traffic environment (Quadstone Paramics, 2012). Four modules are developed using the Paramics API: 1) Signal Controller Module, 2) Vehicle Tracking Module, 3) State Construction Module, 4) and Reward Calculation Module. The following sections discuss the theoretical background of Paramics and each of the aforementioned modules.

4.2.2.3.1 Paramics

This section discusses the theoretical background of Paramics and some of its functionality that is essential in this research.

4.2.2.3.1.1 Vehicle Dynamics and Car-Following Models

Paramics models stochastic vehicle flow by simple speed regulations, car-following, and overtaking rules. Each vehicle in the simulation has a set of characteristics and can modify its speed at the resolution of a tenth of a second (i.e. time step). Paramics uses a car-following model, where the distance between two successive vehicles depends on: current position, vehicle length, vehicle speed, safe gap, and reaction time. Reaction time is modelled by attaching a short memory to a vehicle that not only carries its current speed and position, but also a record of its speed and position at some time in the past, where the time interval is the accepted reaction time.

4.2.2.3.1.2 Gap Acceptance and Lane-Changing Models

Paramics models three types of gap: merging, lane-changing, and crossing gaps. The merge type is used when a vehicle is approaching a junction and must merge with another stream of traffic, and the other stream has right of way. In this case a minimum merge gap should be satisfied for the vehicle to safely merge with the other stream. The lane-change type is used when a vehicle is being impeded on a multi-lane road or is approaching a junction, in this case a similar gap acceptance should exist for the vehicle to change lane. In cases where vehicles cross traffic streams (e.g. permitted left-turn), crossing gap acceptance is employed by deciding how large a gap is needed in an opposing stream for the current vehicle to proceed.

4.2.2.3.1.3 Method of Traffic Assignment

Paramics provides three methods of traffic assignment that can be employed at different levels: “all-or-nothing”, stochastic, and dynamic feedback assignment. All-or-nothing, as implied by its name, is a basic traffic assignment technique that assumes that all drivers choose the same route between an origin-destination pair, and that route costs (e.g. travel time) do not depend on traffic flow. On the other hand, stochastic assignment methods model the error in the perceived travel cost for drivers, this error varies randomly within predefined limits to model the heterogeneity among drivers' perception of travel cost to their destination. Dynamic feedback assignment updates route travel times to destinations at each intersection in the network, meaning that at each feedback interval (e.g. 5 min) route travel times from any intersection to all destinations are updated, mimicking real-time traveller information systems. Recognising that only drivers who are familiar with the network and traffic conditions, or subscribers to this real-time information, can access this information, a certain familiarity percentage needs to be defined and calibrated in Paramics. In this framework dynamic stochastic traffic assignment is employed.

4.2.2.3.1.4 Paramics Application Programming Interface (API)

Paramics API has its own basic functions and the user can implement the required functionality by providing the control logic, either as override functions or as overload control functions. It also has Callback functions that provide information about some of the attributes of the current vehicles and their environment, such as the link a vehicle is traversing, as well as providing simple debugging facilities.

The Simulation Environment component in this research is developed on a combination of these API functions. Four main modules are developed to build this component; Signal controller Module, Vehicle Tracking Module, State Construction Module, and Reward Calculation Module.

4.2.2.3.2 Signal Controller Module

This module is responsible for executing the action for each signalised intersection. Two phasing sequence schemes are developed: fixed phasing sequence and variable phasing sequence. The fixed phasing sequence allows a fixed order of phases; however the variable phasing sequence allows for phase jumping/skipping in which the phasing sequence is not predetermined.

It should be noted that for a fixed-time controller, all available phases should be activated at least once within a cycle. This is not the case however for actuated controllers where phases can be shortened or skipped, depending on the demand and where the minimum and maximum green times for each phase are considered. On the other hand, the variable phasing sequence method decides on actions to be taken at regular intervals of time (1 time unit for example) after satisfying the green time constraints. It should also be noted that both phasing sequence schemes (i.e. fixed and variable) are working under acyclic timings in which there is no constraint on the cycle length. Therefore, the proposed algorithm deviates from the common notion of signal timing plans (split, cycle and offset) and can potentially result in better overall network performance, minimising the average delay, congestion and the likelihood of intersection cross-blocking.

4.2.2.3.2.1 Action Definition 1: Fixed Phasing Sequence (FPS)

In fixed phasing sequence, the proposed order of phases in the phasing scheme is followed as shown in Figure 4-13 a. The action is a binary number that indicates to either extend (i.e. a=0) or terminate the current green phase and move to the next phase (i.e. a=1) (Equation 4-6).

$$a_i^k = j\ ,\ j \in \{0,1\} \tag{4-6}$$

4.2.2.3.2.2 Action Definition 2: Variable Phasing Sequence (VPS)

In the VPS approach (see Figure 4-13-b) the sequence constraints are relaxed. Although VPS can be more efficient than fixed phasing, jumping phases may confuse some drivers who are used to fixed phasing. It is up to the municipality in charge to weigh potential efficiency gains from variable phasing against possible driver confusion. The action in the VPS is the phase to be in effect in the next time step (Equation 4-7).

$$a_i^k = j\ ,\ j \in \{1,2,\dots,P_i\} \tag{4-7}$$

where P is the number of phases associated with agent *i*.

It is worth noting that if the action is the same as the current green phase, then the green time for that current phase will be extended by a specific time interval (t*). Otherwise the green light will be switched to phase *a* after accounting for the minimum green ($\mathrm{G_{min}}$), yellow (Y), and all red (R) times (Equation 4-8).

$$\Delta^k = \begin{cases} G_{min}^{a_i^k} + Y^{a_i^k} + R^{a_i^k} & if\ a^k \neq a^{k-1} \\ t^* & if\ a^k = a^{k-1} \end{cases} \tag{4-8}$$

The definitions of the minimum green, yellow, and all red times are as follows:

Minimum Green: the minimum time that should be served before switching to the next phase. Minimum green time is typically determined to satisfy the following pedestrian timing requirements:

- Walk Time Interval: the walk interval is typically set to 4 to 7 seconds (5 sec on average) under normal conditions. The interval is meant to allow pedestrians to have adequate time to leave the curb before the clearance interval is displayed;
- Flashing Don't Walk Interval: also known as the pedestrian clearance time, is determined to allow a pedestrian to clear the roadway width before the opposing vehicles receive a green indication, and is calculated as follows:

$$FDW = \frac{W}{WS} \tag{4-9}$$

where: FDW = *flashing don't walk (pedestrian clearance) time in sec;*

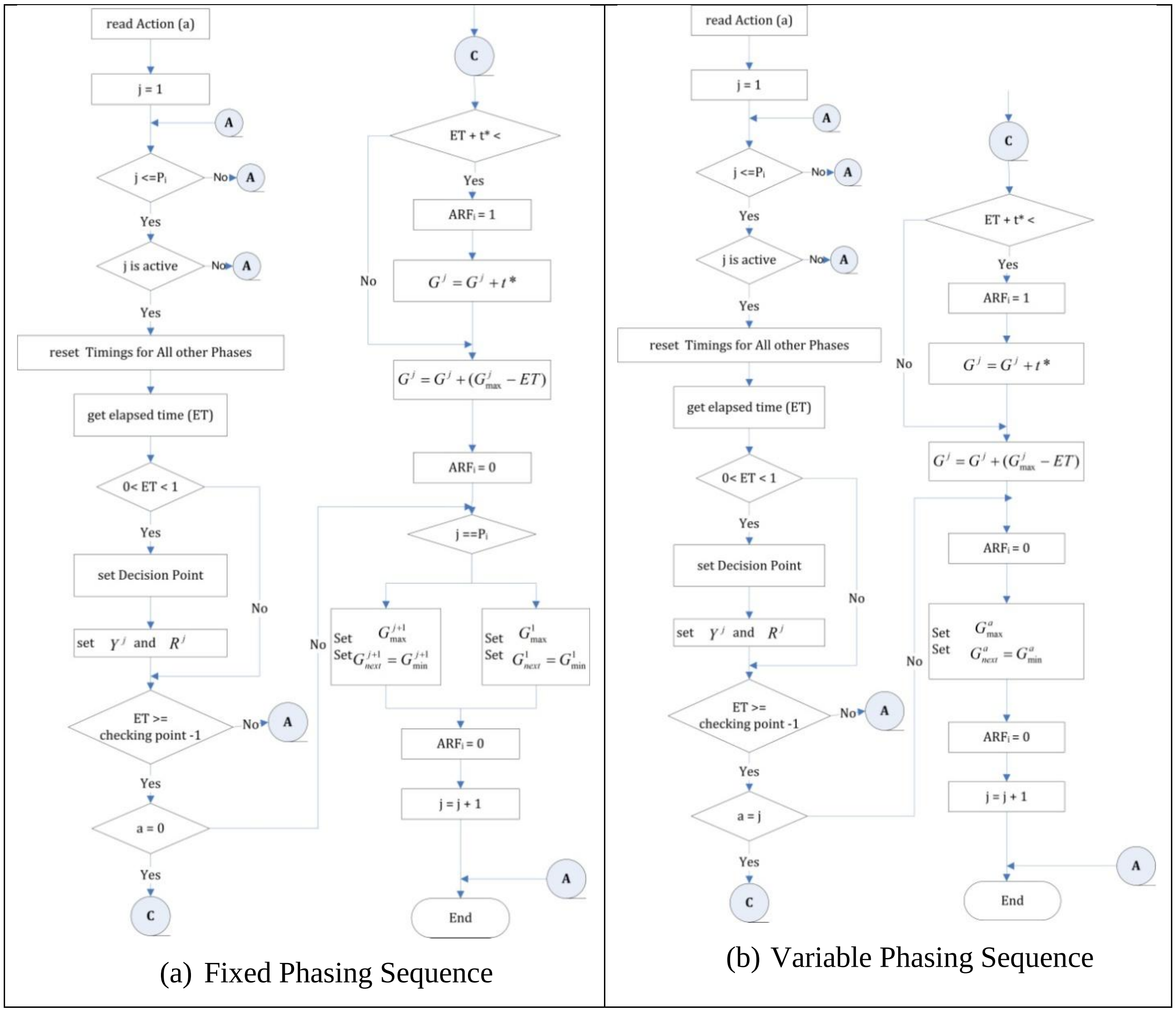
W= walking (crossing) distance in metres, as noted above; and

WS= average walking speed in m/sec (typically 1 m/sec)

Hence $\mathrm{G_{min}} = 5 + FDW$

Yellow: a predetermined constant (e.g. 3 sec.). The yellow time succeeds the green time.

All Red: a predetermined constant (e.g. 2 sec.) during which all phases are shown red. The red time succeeds the yellow phase.



(a) Fixed Phasing Sequence (b) Variable Phasing Sequence

Figure 4-13 Signal Controller Module

4.2.2.3.3 Vehicle Tracking Module

This module is responsible for monitoring the vehicle speeds and continuously updating the attributes for each intersection accordingly. These attributes are: cumulative delay, queue length, and number of stops associated with each lane of approaching links to the signalised intersections. As shown in Figure 4-14, each vehicle ($v \in V_l^k$) entering the approaching links to a signalised intersection (all the roadway segments connecting this intersection to the upstream intersection) is continuously monitored and its speed is tracked. If the vehicle's speed is less than a certain threshold (Sp^{Thr}), which means that the vehicle is joining a queue, the queue length and the delay associated with the corresponding lane are updated. Once the vehicle's speed exceeds Sp^{Thr} , the queue length is updated (e.g. shortened by the number of vehicles leaving the queue).

If the vehicle came to a complete stop, the number of stops associated with the corresponding lane is updated. If the vehicle crossed the stop line and left the intersection, the value of the cumulative delay associated with the previous lane in which the vehicle was travelling is decreased by the value of the cumulative delay associated with this vehicle. It is worth noting that if the vehicle changed the lane from one lane that belongs to a certain phase to another lane that belongs to another phase, the queue length and the delay attributes are transferred to those of the new phase to which the vehicle just joined. In addition to reporting the queues, stops and delay for each lane/phase, two intersection-wide measures are continuously updated, the *throughput* (TH) and the *total cumulative delay (Cd),* once the vehicle leaves the intersection.

The above description can be formulated mathematically as shown below:

4.2.2.3.3.1 Queue Length

Vehicle (v) is considered in a queue if its speed is below a certain speed threshold (Sp^{Thr}), in this study $Sp^{Thr} = 7$kph (Equation 4-10);

$$q_l{}^k = \sum_{v \in V_l^k} q_v^k \tag{4-10}$$

$$where\ q_v^k = \begin{cases} 1\ if\ \ Sp_v^{k-1} > Sp^{Thr}\ and\ Sp_v^k \le Sp^{Thr} \\ -1\ if\ \ Sp_v^{k-1} \le Sp^{Thr}\ and\ Sp_v^k > Sp^{Thr} \\ 0\ if\ \ Sp_v^{k-1} \le Sp^{Thr}\ and\ Sp_v^k \le Sp^{Thr} \end{cases} \tag{4-11}$$

where V_l^k is the set of vehicles travelling on lane l at time k and $q_l{}^k$ is the number of queued vehicles in lane l at time k and Lj is the lane-group corresponding to phase j.

4.2.2.3.3.2 Cumulative Delay

The cumulative delay is the total time spent by a vehicle in a queue. The cumulative delay associated with lane l up to time step Cd_l^k , is defined as follows:

$$Cd_l{}^k = \sum_{v \in V_l^k} Cd_v^k \qquad \forall\ l \in L_i \tag{4-12}$$

$$Cd_v^k = \begin{cases} Cd_v^{k-1} + ts & if\ \ Sp_v^k \le Sp^{Thr} \\ Cd_v^{k-1} & if\ Sp_v^k > Sp^{Thr} \\ -Cd_v^{k-1} & if\ \ v \in V_l^{k-1} \cap\ v \notin V_l^k \end{cases} \tag{4-13}$$

where Cd_v^k is the total time spent by vehicle v in a queue up to time step k, ts is the time step, and Sp_v^k is vehicle’s speed at time k.

4.2.2.3.3.3 Number of Stops

The total number of stops in lane l at time *k*, ${St_l}^k$, is defined as follows:

$$ {St_l}^k = \sum_{v \in V_l^k} St_v^k \qquad \forall\, l \in L_i \tag{4-14} $$

$$ St_v^k = \begin{cases} 1 & if \;\; Sp_v^{k-1} > 0 \; and \;\; Sp_v^k \leq 0 \\ 0 & if \;\; Sp_v^k > 0 \end{cases} \tag{4-15} $$

where ${St_v}^k$ is a binary value associated with each vehicle, v, that takes the value 1 if the vehicle is stopped at time k, and 0 otherwise.

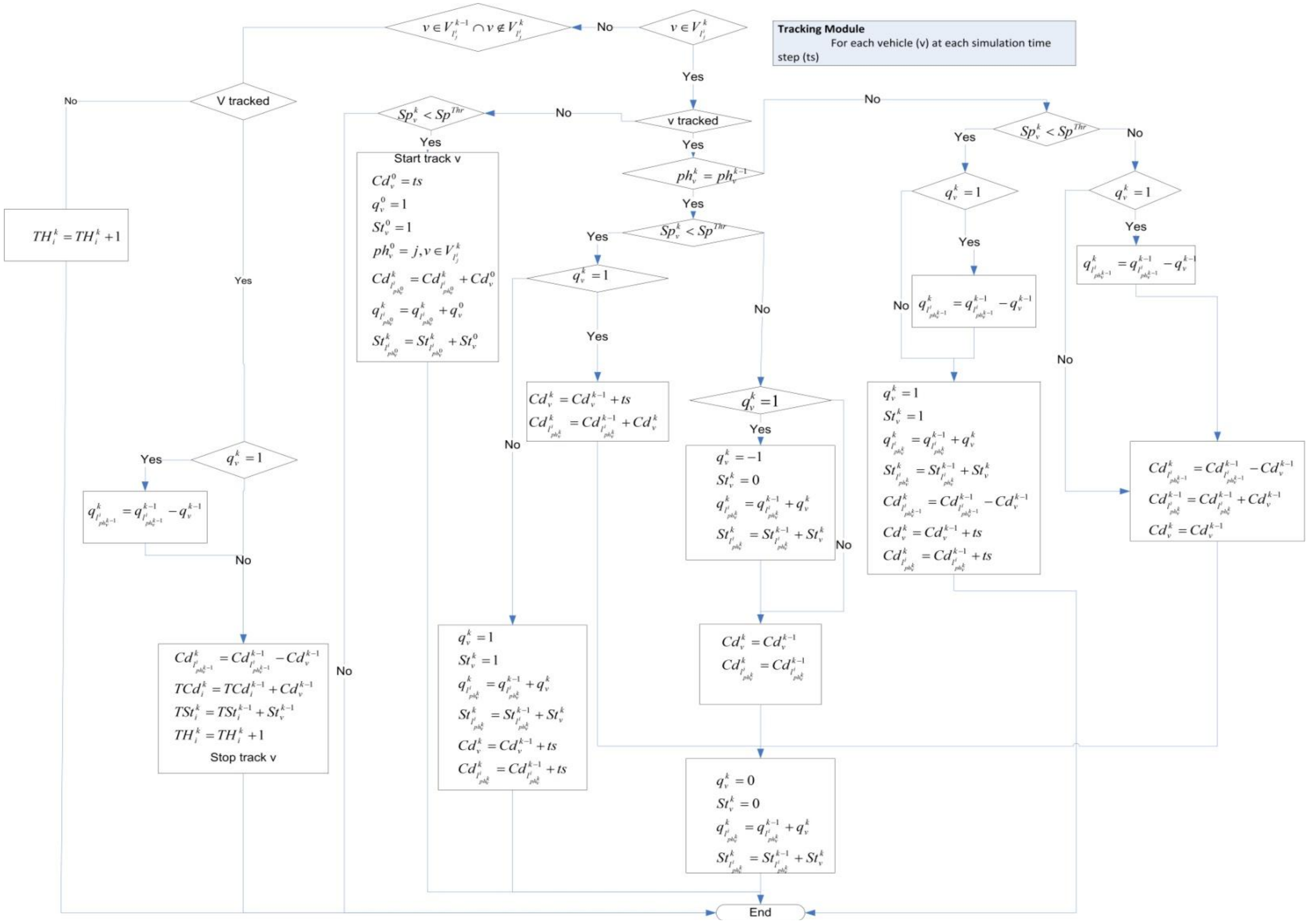

Tracking Module
For each vehicle (v) at each simulation time step (ts)
$v \in V^{k}_{l^i_j}$
$v \in V^{k-1}_{l^i_j} \cap v \notin V^{k}_{l^i_j}$
V tracked
v tracked
$Sp^k_v < Sp^{Thr}$
Start track v
$Cd^0_v = ts$
$q^0_v = 1$
$St^0_v = 1$
$ph^0_v = j, v \in V^k_{l^i_j}$
$Cd^k_{l^i_{ph^0_v}} = Cd^k_{l^i_{ph^0_v}} + Cd^0_v$
$q^k_{l^i_{ph^0_v}} = q^k_{l^i_{ph^0_v}} + q^0_v$
$St^k_{l^i_{ph^0_v}} = St^k_{l^i_{ph^0_v}} + St^0_v$
$TH^k_i = TH^k_i + 1$
$ph^k_v = ph^{k-1}_v$
$q^k_v = 1$
$q^k_v = -1$
$St^k_v = 0$
$St^k_v = 1$
$Cd^k_v = Cd^{k-1}_v + ts$
$Cd^k_{l^i_{ph^k_v}} = Cd^{k-1}_{l^i_{ph^k_v}} + Cd^k_v$
$q^k_{l^i_{ph^{k-1}_v}} = q^{k-1}_{l^i_{ph^{k-1}_v}} - q^{k-1}_v$
$q^k_{l^i_{ph^k_v}} = q^{k-1}_{l^i_{ph^k_v}} + q^k_v$
$St^k_{l^i_{ph^k_v}} = St^{k-1}_{l^i_{ph^k_v}} + St^k_v$
$Cd^k_{l^i_{ph^{k-1}_v}} = Cd^{k-1}_{l^i_{ph^{k-1}_v}} - Cd^{k-1}_v$
$Cd^{k-1}_{l^i_{ph^k_v}} = Cd^{k-1}_{l^i_{ph^k_v}} + Cd^{k-1}_v$
$Cd^k_v = Cd^{k-1}_v$
$Cd^k_{l^i_{ph^k_v}} = Cd^{k-1}_{l^i_{ph^k_v}}$
$Cd^k_{l^i_{ph^k_v}} = Cd^{k-1}_{l^i_{ph^k_v}} + ts$
$TCd^k_i = TCd^{k-1}_i + Cd^{k-1}_v$
$TSt^k_i = TSt^{k-1}_i + St^{k-1}_v$
$TH^k_i = TH^k_i + 1$
Stop track v
$q^k_v = 0$
Yes
No
End

Figure 4-14 Vehicle Tracking Module

4.2.2.3.4 State Module

This module is responsible for constructing the state. In order to fully represent the state of an intersection, the state should be represented by the current settings of signal timings and the current status of traffic approaching the intersection. In this research, the state of each intersection is represented by a vector of 2+P_i components, where P_i is the number of phases associated with intersection *i*. The first two components are related to signal timing: 1) index of the current green phase, and 2) elapsed time of the current phase.

Three RL models are developed for the remaining P_i components that represent the state of traffic approaching the intersection; each considers one of the state representations that are investigated separately in the literature as follows:

4.2.2.3.4.1 State Definition 1: Queue length

State definition 1 is represented by a vector of P_i components that are the maximum queue lengths associated with each phase (Equation 4-16). This is one of the most common state definitions in RL-based signal control literature (Abdulhai *et al.*, 2003; Richter *et al.*, 2007).

$$s_i{}^k = max_{l \in L_i} q_l{}^k \quad \forall\, i \in \{1,2,\dots,P_i\} \qquad \textit{(4-16)}$$

4.2.2.3.4.2 State Definition 2: Arrival of Vehicles to the Current Green Phase and Queue Length at Red Phase

In this definition, one of the state vector components is the maximum arrivals in the green phase and the other P-1 components are the maximum queue lengths for the red phases. A similar state definition is considered in the literature (Camponogara and Kraus Jr., 2003; De Oliveira *et al.*, 2006; Salkham *et al.*, 2008; Thorpe, 1997; Wiering *et al.*, 2003). This definition is represented by Equation 4-17:

$$s_i{}^k = \begin{cases} max_{l \in L_i} q_l{}^k & if\ i = red\,phase \\ max_{l \in L_i} Ar_l{}^k & if\ i = green\,phase \end{cases} \quad \forall\, i \in \{1,2,\dots,P_i\} \qquad \textit{(4-17)}$$

where $Ar_l{}^k$ is the number of vehicles arriving in lane l at time *k*.

4.2.2.3.4.3 State Definition 3: Cumulative Delay

The cumulative delay for phase i is the summation of the cumulative delay of all the vehicles that are currently travelling in lane-group Li (Equation 4-18). Vehicles leave the system once they clear the stop line of the intersection. A similar state definition is investigated by Arel *et al.*

(2010).

This state is also represented by P_i components, where each component is the cumulative delay of the corresponding phase.

$$s_i{}^k = \sum_{l \in L_i} Cd_l{}^k \qquad \forall\, i \in \{1,2,\ldots,P_i\} \tag{4-18}$$

It is noteworthy that there is always a trade-off between state representation accuracy by having fine discrete intervals for each state component and the memory and computation time required in the learning and decision making processes.

4.2.2.3.5 Reward Calculation Module

This module is responsible for calculating the reward. Three definitions are considered for the immediate reward as follows:

4.2.2.3.5.1 Reward Definition 1: Minimising the Delay

A typical reward function considers the negative value of the delay experienced by the vehicles between two successive decision points (Equation 4-19) (Abdulhai *et al.*, 2003; Lu *et al.*, 2008; Shoufeng *et al.*, 2008; Wiering, 2000), and the system works to maximise the cumulative future reward (in this case minimising the negative delay value). This typical definition however does not consider how long the vehicles were delayed before the last two decision points.

$$r^k = -\sum_{i \in N} \sum_{l \in L_i} \sum_{v \in V_l^k} b_v^k . \Delta^{k-1} \tag{4-19}$$

$$b_v^k = \begin{cases} 1 & if \;\; Sp_v^k \leq Sp^{Thr} \\ 0 & if \;\; Sp_v^k > Sp^{Thr} \end{cases} \tag{4-20}$$

4.2.2.3.5.2 Reward Definition 2: Maximising the Reduction in the Total Cumulative Delay

The immediate reward is defined as the reduction (saving) in the total cumulative delay, i.e. the difference between the total cumulative delays of two successive decision points (Arel *et al.*, 2010). The total cumulative delay at time *k* is the summation of the cumulative delay, up to time *k*, of all vehicles that are currently in the system.

If the reward has a positive value, this means that the delay is lower than that of the previous time step by this value after executing the selected action. On the other hand, the agent is subject to a penalty (negative value) if an increased cumulative delay is observed (Equation 4-21).

$$r^k = \begin{cases} \sum_{i \in N} \sum_{l \in L_i} \left(Cd_l^k - Cd_v^{k-1} \right) & If \quad a^k = a^{k-1} \ for\ VPS\ or\ a^k = 0\ for\ FPS \\ \frac{\sum_{i \in N} \sum_{l \in L_i} \left(Cd_l^k - Cd_v^{k-1} \right)}{G_{min}^{p^{k-1}}} & If \quad a^k \neq a^{k-1}\ for\ VPS\ or\ a^k = 1\ for\ FPS \end{cases} \tag{4-21}$$

where p^{k-1} indicates the active phase in the time step *k-1*.

While reward definition 1 considers the delay experienced by the vehicles between two successive decision points (Abdulhai *et al.*, 2003; Lu *et al.*, 2008) this definition however does not consider how long the vehicles were delayed before the last decision point. Reward definition 2 on the other hand provides a more accurate representation for the cumulative delay. Table 4-1 demonstrates the difference between the two reward definitions through a two-phase intersection example.

Table 4-1 Illustrative Comparison of Different Reward Definitions

Iteration k-1	Intersection phase 1		Intersection phase 2	
Queue length (no. of vehicles)	2		10	
Cumulative delay (sec)	60		0	
Action	Switch to phase 2 (for minimum green of 10 sec)		Extend phase 1 (by 1 sec time interval)	
Iteration K	Phase 1	Phase 2	Phase 1	Phase 2
Queue length (no of vehicles)	2	0	0	20
Delay experienced between the two iterations (sec)	2*10=20	0	0	20*1=20
Cumulative delay (sec)	60+20=80	0	0	0+20=20
Reward definition 1	-20		-20	
Reward definition 2	(60+0)-(80+0)=-20		(60+0)-(0+20)=**+40**	

It is clearly shown from the above example that the first reward definition does not differentiate between the two actions taken by the agent (switch to phase 2 or extend phase 1). However the second reward definition not only clearly differentiates between the two actions, it also has two opposite signs for the corresponding rewards. This is primarily because reward definition 2 reflects the change in the total cumulative delay while the first definition only considers the absolute value of the delay for the time interval between the successive decision points.

It is also expected that reward definition 2 would speed up the Q-values convergence since all the Q-values are initiated to zero values, and hence the actions that result in negative reward values will have the lowest probability to be chosen by the Q-learning agent compared to those with high positive values.

4.2.2.3.5.3 Reward Definition 3: Minimising and Balancing Queue Length

Reducing the cumulative delay does not guarantee preventing queue spillback, and may therefore result in jeopardising minor streets and upstream intersections. A reward can be defined as the reduction in the sum of the squared maximum queues (Equation 4-22). The objective is to minimise and equalise queue lengths across different phases (Camponogara and Kraus Jr., 2003; De Oliveira *et al.*, 2006):

$$r^k = \sum_{i \in N}\left(max_{l \in L_i}\, {q_l}^{k}\right)^2 - \sum_{i \in N}\left(max_{l \in L_i}\, {q_l}^{k-1}\right)^2 \qquad (4\text{-}22)$$

4.2.2.4 Field Implementation Consideration

The agent is designed to learn off-line through a simulation environment (such as the microsimulation model employed in the experiments) before field implementation. After convergence to the optimal policy, the agent can either be deployed in the field – by mapping the measured state of the system directly to optimal control actions using the learned policy – or it can continue learning in the field by starting from the learned policy. In both cases, no model of the traffic system is required in the field implementation, which makes the proposed approach more appealing for field deployment.

It is worth noting that as learning matures in the simulation environment and continues in the field, exploration is asymptotically decreased to stabilise the convergence process. In addition, when exploration is terminated the agent can still continue learning by improving its Q values in the field over time, i.e. the control policy continues to slightly change by updating the values of state-action pairs.

The general platform, including the learning algorithm, requires: 1) queue length (state), and 2) the cumulative delay (reward). In the near future, and with increased market penetration of connected vehicles that can wirelessly interact with traffic lights, these values can be readily measured. However, the current state-of-the-practice relies on two types of detection system to obtain this information: 1) intrusive detection systems, and 2) modern non-intrusive detection systems.

Intrusive detection systems (mainly inductive loop detectors), although costly and inaccurate in some cases, are the most widely used. Therefore, we assessed the possibility of acquiring queue length and delay from loop detector data. Loop detectors can plausibly produce queue length by detecting queue spillbacks beyond the detector located upstream of the traffic signal. Loop detector data could also be used to estimate delay from a deterministic queuing diagram based on cumulative arrivals at the upstream detector and cumulative departures from the stopline detector. The arrival and departure profiles can be then compared to calculate the total time spent in the system and then estimate the queue accumulation and the delay on the intersection approach. To assess the potential of using loop detector data to estimate average delay (the reward in this case), a simulation experiment is conducted to compare the results of the average delay of two methods: 1) tracking vehicle method (using Paramics API), and 2) loop detector method (using deterministic queuing graphs) (see Figure 4-15 for loop detector locations). The average delay calculated from the vehicle tracking method is found to be 75 sec/veh. The delay calculated from the loop detector method is found to be 73 sec/veh. Therefore the system states and rewards could be plausibly reproduced using existing and commonly-used inductive loop detectors (Buckholz, 2007).

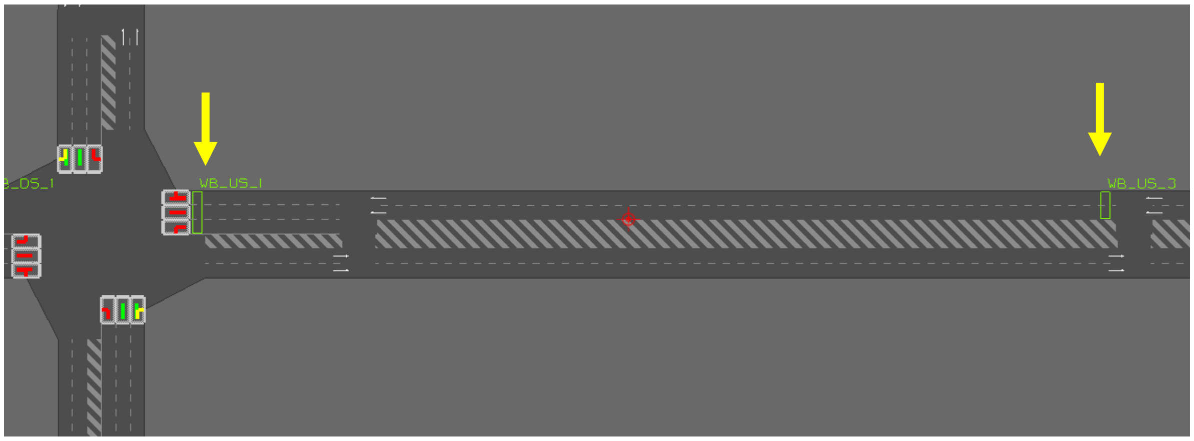


Figure 4-15 Loop Detector Location

On the other hand, non-intrusive detection systems such as video detection can track individual vehicles and directly estimate delay. These detection systems have gradually been used to replace inductive loops in many cities around the world, including the City of Toronto and the City of Burlington to name but a few. For example, a video image processor (VIP) from a CCTV camera can simply substitute inductive loops, because most control devices merely accept data available from loop detectors. Detection capabilities such that of VIPs include queue detection and vehicle tracking, which can be plausibly harnessed to benefit traffic signal control systems, similar to the platform developed by Tian and Abbas (2007).

In the next few decades, with the evolution of the Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communication, the cumulative delay at intersections should be an easy measure to obtain. In essence, MARLIN-ATSC is designed to consider current and future technologies to sense and measure the states of the system.

The next three chapters (Chapters 5, 6 and 7) demonstrate the applicability of MARLIN-ATSC using three testbed traffic networks (i.e. small-, medium- and large-scale) to ensure the seamless transferability of the system design and results. Chapter 5 investigates the effect of different RL-design parameters on an isolated intersection. Chapter 6 investigates the different modes of MARLIN-ATSC (independent and integrated modes) on a prototype network of 5 intersections. Chapter 7 builds on the lessons learned from Chapters 5 and 6 by applying MARLIN to a large-scale network of 59 intersections in a model of downtown Toronto. The structure of each of these chapters follows a standardised approach that is shown in Figure 4-16.

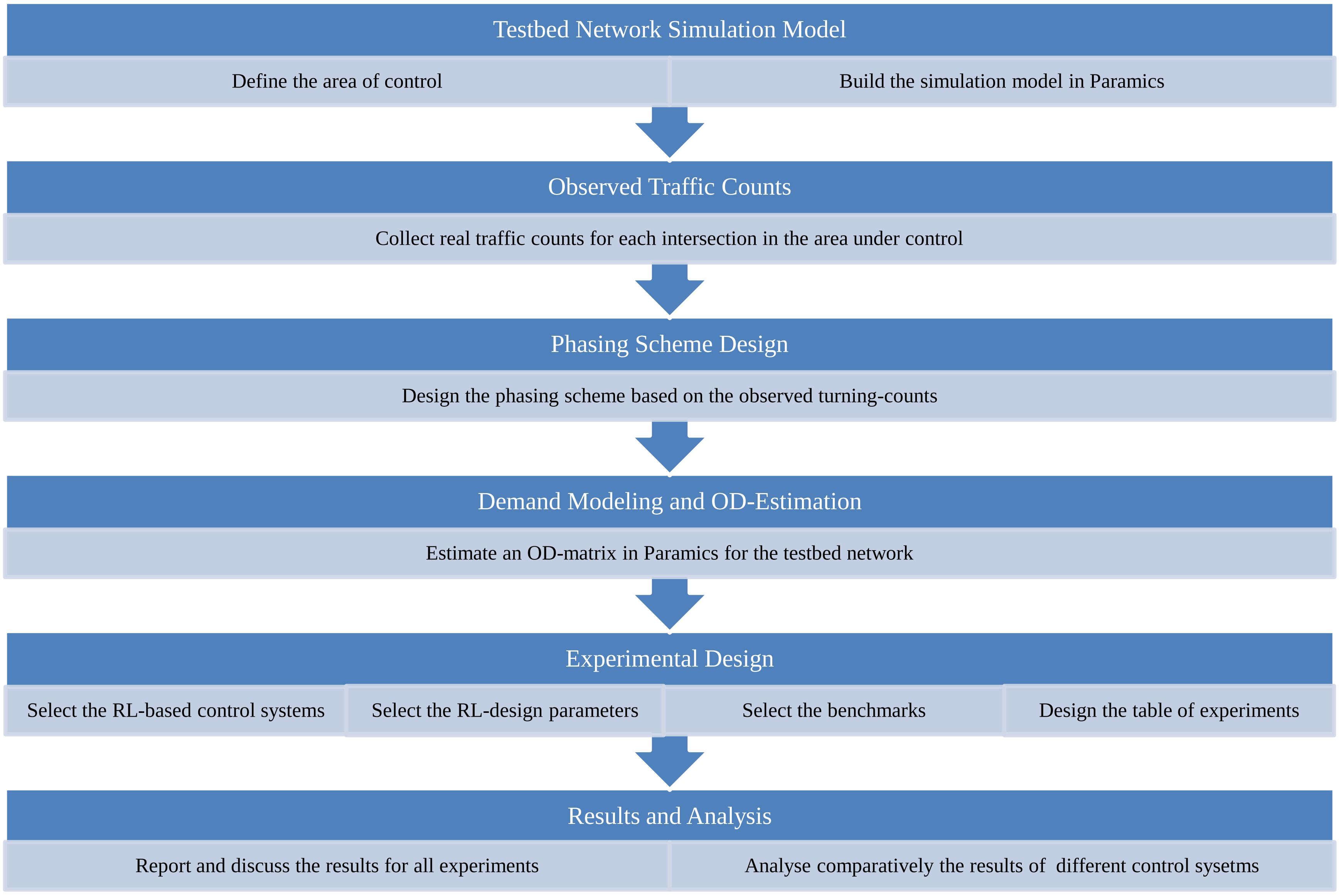

Testbed Network Simulation Model
Define the area of control
Build the simulation model in Paramics
Observed Traffic Counts
Collect real traffic counts for each intersection in the area under control
Phasing Scheme Design
Design the phasing scheme based on the observed turning-counts
Demand Modeling and OD-Estimation
Estimate an OD-matrix in Paramics for the testbed network
Experimental Design
Select the RL-based control systems
Select the RL-design parameters
Select the benchmarks
Design the table of experiments
Results and Analysis
Report and discuss the results for all experiments
Analyse comparatively the results of different control sysetms

Figure 4-16 Standardised Development Approach and Roadmap for Chapters 5, 6, and 7

5 EXPERIMENTAL RESULTS 1: ISOLATED INTERSECTION CONTROL

As discussed in Chapter 2, numerous RL-based traffic control methods were reviewed in the literature (Gosavi, 2003; Sutton and Barto, 1998), however these methods only investigated a certain TD method without quantitatively discussing the suitability of each TD method in solving the traffic signal control problem. In addition, each study considered a certain set of RL design parameters without providing quantitative justification for their selection. Moreover, most of these studies considered a simplified simulation environment (Abdulhai *et al.*, 2003; Arel *et al.*, 2010; Camponogara and Kraus Jr., 2003; De Oliveira *et al.*, 2006; Richter *et al.*, 2007) and/or assumed hypothetical traffic flows (Abdulhai *et al.*, 2003; Arel *et al.*, 2010; Camponogara and Kraus Jr., 2003; De Oliveira *et al.*, 2006; Richter *et al.*, 2007; Thorpe, 1997; Wiering, 2000). Simplifying assumptions about RL methods and parameter choices, as well as intersection details, are good for the proof of concept stages of applying RL to traffic signal control. However, to get to the level of realistic implementation and deployment of RL-based ATSC more thorough quantitative investigations of basic RL methods are required. Also such quantitative investigations are paramount before embarking on the development of more complex RL-based ATSC methods in this research. Therefore, although the ultimate goal of this research is to develop a coordinated traffic control system for large networks, this chapter represents the first step toward achieving this goal by examining the effect of various RL parameters on the performance of the MARLIN-ATSC on a fully-fledged isolated intersection with realistic traffic flow data. This strategy first solidifies the methodology then expands the scope of the problem later in Chapters 6 and 7 for medium and large-scale applications.

This chapter presents a comprehensive investigation of key parameters in any RL-based signal control problem for isolated intersections by offering analysis on the effect of the following design parameters on RL design: 1) exploration method, 2) TD learning method, 3) traffic signal phasing scheme, 4) traffic state representation, and 5) reward definition. The structure of this chapter follows the roadmap presented in Figure 4-16.

5.1 TESTBED SIMULATION MODEL

The experiments are tested using the Independent Mode version of MARLIN-ATSC (MARL-TI) (section 4.2.2.1.1) on a 4-leg simulated intersection located in the heart of the financial district in downtown Toronto (Front and Bay Streets) as shown in Figure 5-1.

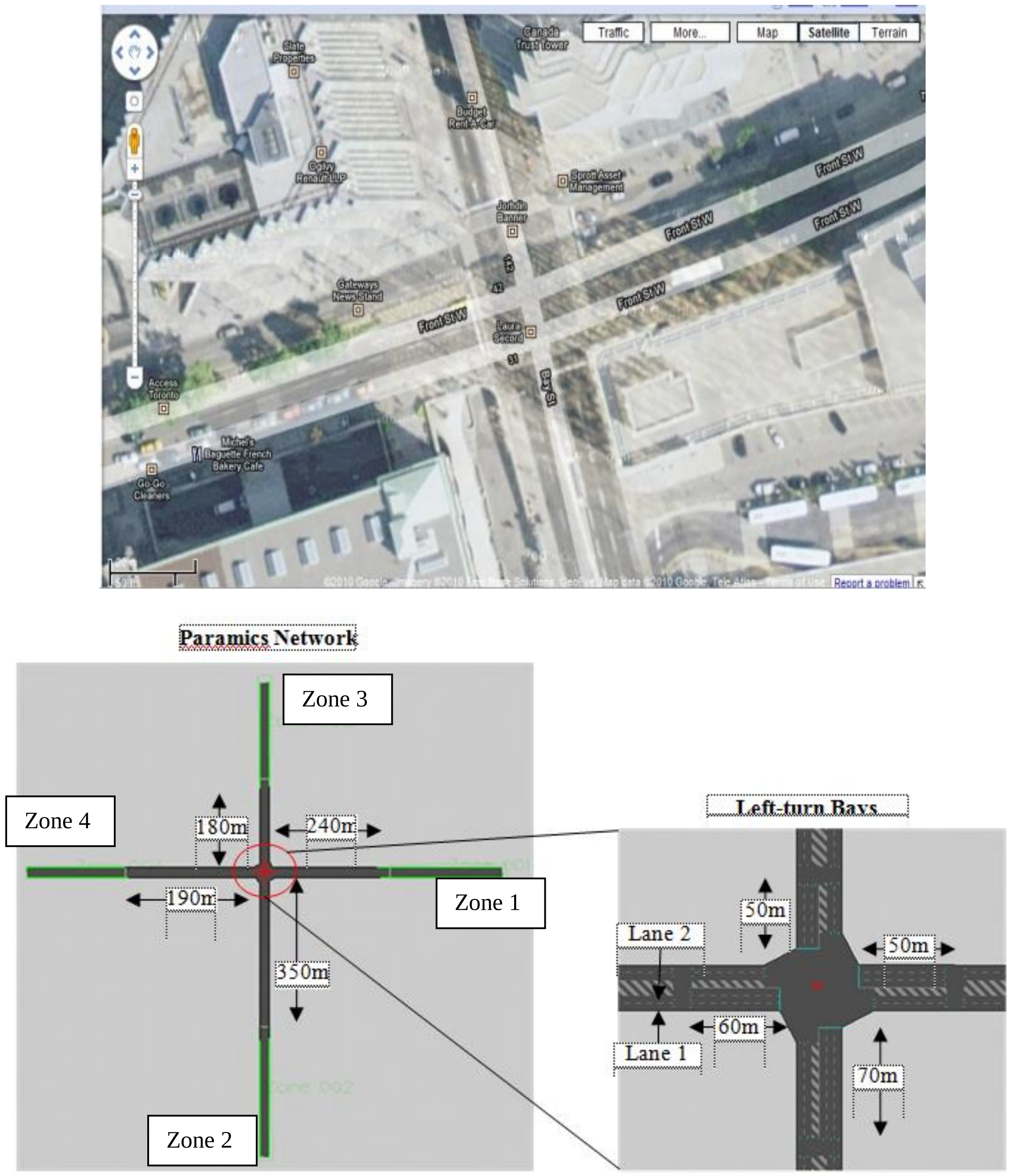


Figure 5-1 Testbed Intersection

5.2 Observed Traffic Counts

In cases where traffic volumes are low and/or do not exhibit considerable arrival pattern variations, fixed traffic signal control performance may be practically sufficient. When demand, congestion levels and/or flow fluctuations increase, advanced ATSC methods are often warranted. Although data were made available for this intersection for the years 2002, 2005, 2007 and 2009 for the AM and PM peak hours, it is found that the afternoon rush hour (PM Peak) for the year 2005 represents the highest total demand approaching this intersection, as shown in Table 5-1, and was hence used in the analysis.

Table 5-1 Traffic Flow Patterns During PM peak hour in 2005

Direction / Volume	Northbound (NB)	Eastbound (EB)	Southbound (SB)	Westbound (WB)
Through	721	1223	806	844
Left-Turn	71	134	188	278
Right-Turn	88	121	100	86

5.3 Phasing Scheme Design

The subject of phasing and selecting the appropriate phase plan is critical for the effective control of signalised intersections, as discussed in the Traffic Engineering book (McShane *et al.*, 1998). For isolated intersections, the standard 8-phase NEMA (National Electrical Manufacturers Association) controller (FHWA, 2005a) is recommended. In NEMA controllers the phasing plan consists of eight phases, but only two to six phases are implemented in any sequence (see Figure 5-2). However, the greater the number of phases the longer the lost time in any given cycle length. An important factor in designing the phasing plan is the left-turn (LT) movement, which could be designed as a permitted LT (i.e. opposing traffic is allowed), as a protected LT (i.e. opposing traffic is stopped), or a combination of the two (called compound phasing). According to the procedure for developing a reasonable phase plan recommended by McShane *et al.* (1998), it is not advisable to include any compound phasing in preliminary signal timing design. Also it is recommended considering a protected LT phase in the following cases:

- LT volume in excess of 200 vph;
- cross-product of the LT volume and opposing through volume per lane (in vphpl) in excess of 50,000 for one opposing through lane, 90,000 for two opposing through lanes, or 110,000 for three or more opposing through lanes.

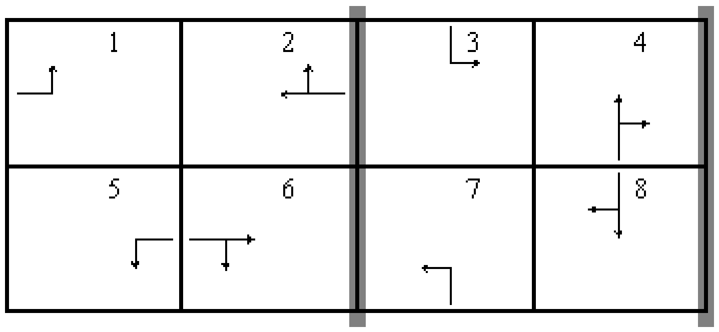


Figure 5-2 Standard NEMA Controller Phases (FHWA, 2005a)

Following the phasing scheme design guidelines discussed above and according to the observed traffic counts shown in Table 5-1, it was found that each approach (Eastbound (EB), Westbound (WB), Northbound (NB), and Southbound (SB)) requires separate through and LT phases, resulting in the phasing scheme design shown in Figure 5-3.

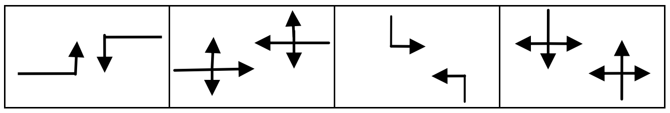


Figure 5-3 Phasing Scheme Design

5.4 Demand Modelling

The traffic flow volumes shown in Table 5-1 are then translated to a typical Origin-Destination (OD) matrix format. The OD matrix forms the input to Paramics as shown in Table 5-2. An advantage of Dynamic Traffic Assignment (DTA) models is the capability of modelling the variation in traffic flow within the period of analysis (PM peak hour in this case). Therefore in DTA models, traffic demand is sliced into discretized time intervals that form the traffic flow arrival pattern (aka *Profile*) at the intersection. To examine the variations in traffic arrivals to intersection approaches, it requires the availability of 10–15 minutes traffic count intervals at a few locations. The Toronto 24-hour counts with 10 min intervals along the intersections of University Avenue NB (from Front St. to College St.) in downtown Toronto – during a typical weekday – were made available. From observing the 24-hour traffic flow volume sheets it was found that the morning peak hour exhibits variations in traffic flow around the mean that range from ± 5% to ± 25% as shown in Figure 5-4.

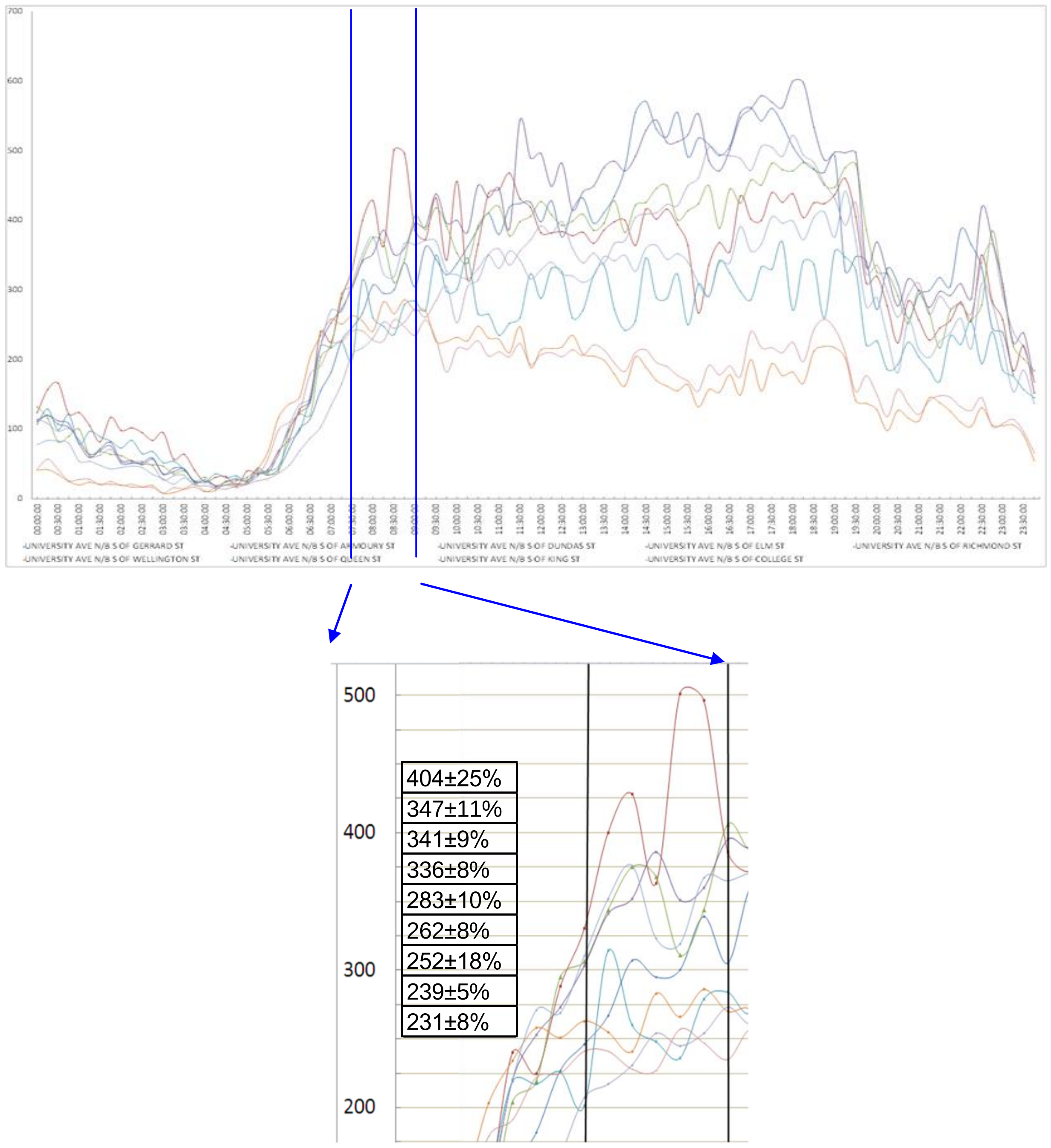


Figure 5-4 Demand Profiles for University Ave. NB Intersections

The simulation typically starts with a warm-up period to have realistic start-up traffic volumes in the traffic network prior to starting the analysis period. A ½ hour warm-up linearly increasing arrival pattern is used in this experiment, followed by the PM peak hour simulation. By examining the 24 traffic flow arrival pattern (and specifically the PM peak hour) at a few intersections within the downtown core of Toronto, it was found that traffic flow fluctuates across the hour with a random pattern around the mean. Therefore, in this experiment, as shown in Figure 5-5, two profiles were examined: 1) uniform arrival profile for all phases (benchmark), and 2) randomised variable arrival profiles for each phase while maintaining the same mean flow rate as in the uniform case. The uniform profile case mimics situations where the cumulative traffic flow is steadily increasing with time until the end of the hour. However, the variable profile case mimics situations where the cumulative traffic flow is increasing but with different rates across the hour which is more realistic in congested urban areas.

Table 5-2 OD-Matrix

Zone No.	**1**	**2**	**3**	**4**	Total
1	0	278	86	844	1208
2	88	0	721	71	880
3	188	806	0	100	1094
4	1223	121	134	0	1478
Total	1499	1205	941	1015	4660

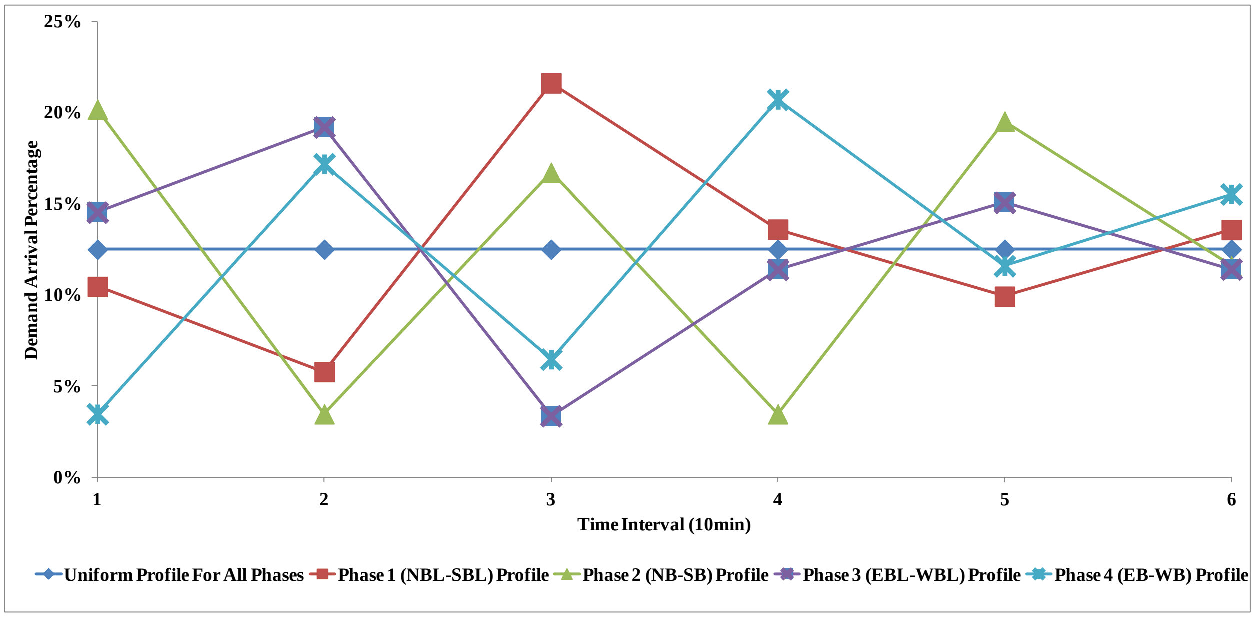


Figure 5-5 Demand Profiles

5.5 Experimental Design

As discussed previously, the main objective of this chapter is to examine and investigate the effect of various RL design parameters on the performance of the RL-based traffic control algorithms. Therefore the experiments are designed to include the following three main tasks:

1) Choice of the parameters for RL-based control systems

a. exploration method

b. RL method

c. state definition

d. action definition

e. reward definition;

2) Selection of benchmark control systems for comparison purposes;

3) Setup of the experiments pool.

5.5.1 RL-Design Parameters

The choice of the best design for RL parameters is an important initial task. The general RL architecture and design parameters are illustrated in Figure 5-6. As shown in the figure, in this experimental design the following dimensions of the problem are investigated:

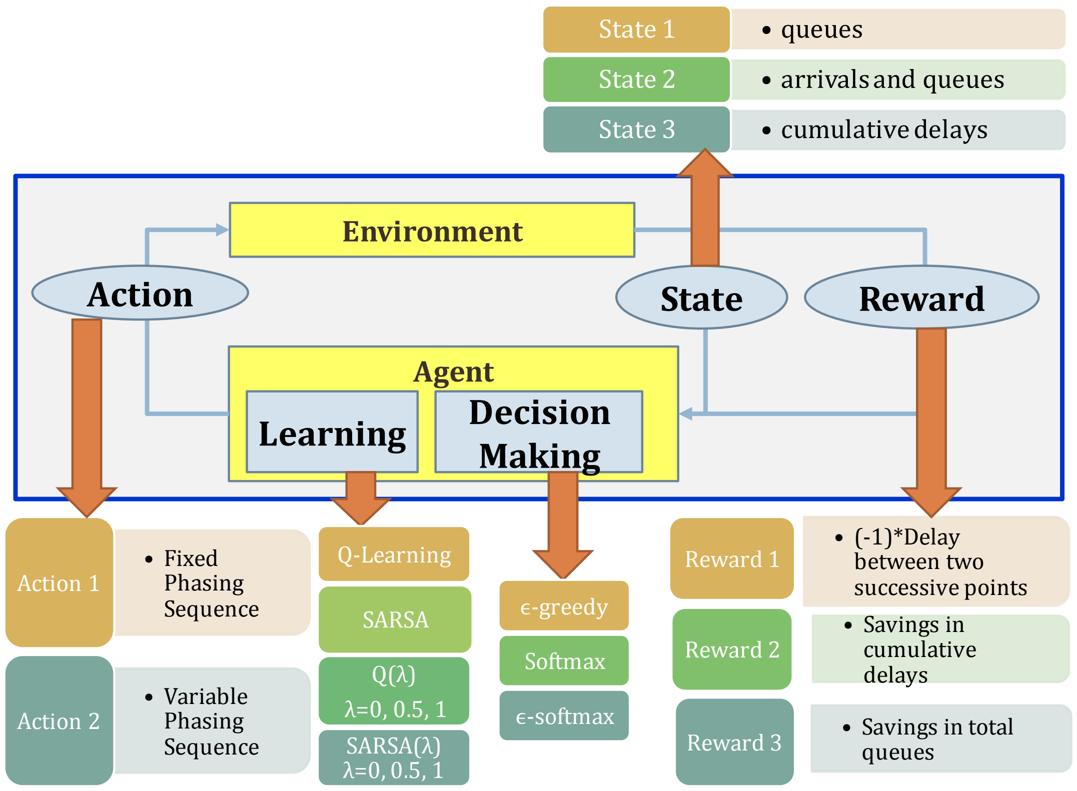


Figure 5-6 RL Architecture and Design Parameters

1) Exploration Method

The following exploration methods are considered (refer to section 3.1.4.3): ε-greedy, softmax, and ε-softmax.

In this set of experiments, two profiles of ε are tested: 1) constant value of 0.1, and 2) exponential function ($\mathrm{e}^{-\mathrm{En}}$). The exponential function ($\mathrm{e}^{-\mathrm{En}}$) is a simple ε-greedy exploration method with a gradually decreasing rate of exploration, so that at the beginning the agent mostly explores, as it has no prior knowledge to exploit, and the agent gradually increases the extent of exploitation towards the end of the learning process. Gradual shifting is required to ensure that the entire state space is covered during the learning process. One possibility for selecting an action is to pick the "best" action with probability $P = 1 - \mathrm{e}^{-\mathrm{En}}$, as suggested by Jacob (2005), where E is a constant and *n* is the iteration number (the "age" of the agent). Three values of E (0.5, 0.05 and 0.005) are tested as shown in Figure 5-7.

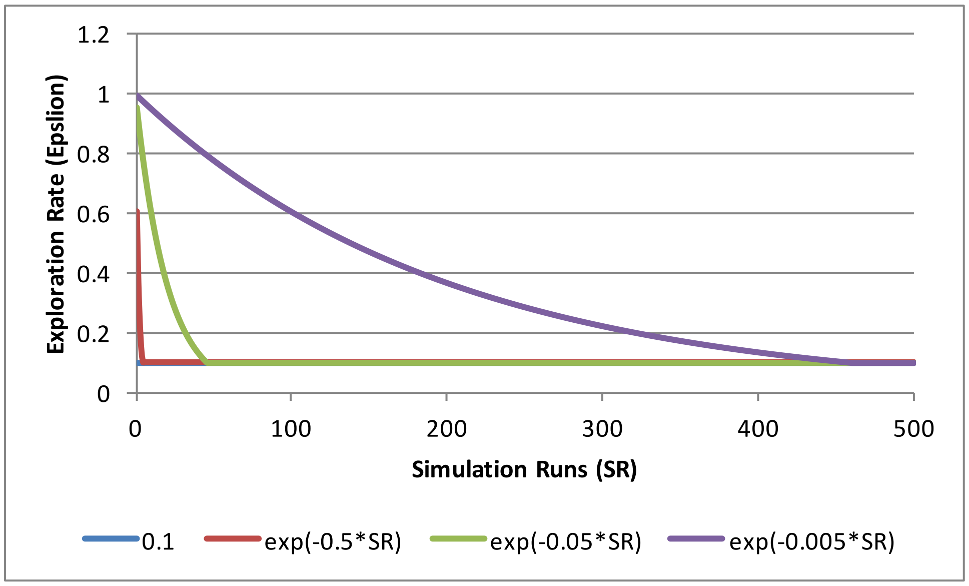


Figure 5-7 Different Exploration Rate Functions

2) RL Method

Both TD(0) and TD(λ) learning methods are investigated. Specifically Q-learning, SARSA, Q(λ), and SARSA(λ) algorithms are tested (refer to section 3.1.4.2 for details on the learning methods).

3) Action Definition

RL-ATSC is designed to account for both FPS and VPS schemes as discussed in section 4.2.2.3.2. Table 5-3 shows the action space corresponding to each action definition. In FPS, the action set is as simple as stay in the current phase (0), or switch to the *next* phase (1). However, in VPS the algorithm has the flexibility to either stay in the current phase (say phase 1) or switch to any other phase (2, 3 or 4).

Table 5-3 Action Set for Each Action Definition

Action Definition	Action Space
Fixed Phasing Sequence	{0,1}
Variable Phasing Sequence	{1,2,3,4}

4) State Representation

State representation takes a number from 1 to 3, which refers to one of the state representation definitions discussed in section 4.2.2.3.4 (state 1 for queue lengths, state 2 for queue lengths and arrivals, and state 3 for cumulative delays). Table 5-4 shows the intervals that define the state space for each state representation. For example, in state definition 1, the through movement phase has four discretized queue length intervals (in vehicles): 0, 1, 5, 15, and > 15.

Table 5-4 State Discretization Intervals for Each Phase

State Definition	State Intervals for Through Phases	State Intervals for Left-Turn Phases
State 1	{0, 1, 5, 15, >15}	{0, 1, 4, >4}
State 2	{0, 1, 5, 15, >15}	{0, 1, 4, >4}
State 3	{0, 0.05, 1, 5, 10, >10}	{0, 0.05, 1, 5, 10, >10}

5) Reward Definition

Three reward definitions are considered in the experiments: 1) savings in delay between two successive points, 2) savings in cumulative delay, and 3) savings in total queues (refer to section 4.2.2.3.5 for details on reward definitions).

5.5.2 Benchmarks:

After the experimental design of different RL-design parameters for RL-ATSC is conducted, the performance of the best designed RL-ATSC is compared to two benchmark systems; namely fixed-time control and fully-actuated control.

5.5.2.1 *Fixed-Time Control*

The fixed-time signal plan is optimised using the Webster method (Webster, 1958). Following the steps of the Webster method (section 2.1.1), the optimised effective green time for each phase is illustrated in Table 5-5 using the following parameters:

Saturation Flow S=1900 veh/greenhr/lane

Yellow Time = 3 sec

All Red Time = 2 sec

Table 5-5 Green Time Splits Using Webster Method

	NS		EW		Total
Parameter	Through	Left-Turn	Through	Left-Turn	
yellow	3	3	3	3	
all red	2	2	2	2	
lost time	2	2	2	2	
demand per lane	463	188	473.6	278	
	413.3	71	684.1	134	
Left turn Factors	1	1.05	1	1.05	
critical volume	463	197.4	684.1	291.9	1636.4
v/S	0.24368	0.10389	0.36005	0.15363	0.86126
C					120
Total Ge					112
Ge	32	14	47	20	112
Ga	27	9	42	15	92

5.5.2.2 *Actuated Control*

A NEMA fully-actuated signal control system is implemented in which two loop detectors are embedded in each approach; a stopline detector and upstream extension detector. The control logic for the NEMA controller is based on three timers to control the green phase as discussed below: minimum green, vehicle extension and maximum green.

- Minimum green: green period that the selected phase must serve, typically governed by the minimum green time for pedestrian crossings (FDW + and walk times) to ensure the safe passage of pedestrians;
- Extension time (Passage time): amount of green time extended by the controller during a green interval when a vehicle is detected. The passage time is applied to a vehicle travelling from the extension detector to the stopline during the green interval. The

passage time only comes into effect once the minimum green has elapsed. It is calculated based on a speed of 40 kph and the distance between the stopline detector and the extension detector with a typical range of 3–5 sec. For the testbed isolated intersection the passage time was found to range from 3–4 sec;

- Maximum Green: maximum green period for the selected phase. The maximum green time starts as soon as a conflicting vehicle is detected during which the selected phase is green.

The following highlights the control logic for the NEMA actuated controller:

- When a phase is green the minimum green timer starts;
- When the minimum green is met the controller checks for vehicle calls on conflicting phases
    - If there is a vehicle call on any conflicting phase the vehicle extension timer starts
    - If there is no vehicle call on any conflicting phase the phase remains green until a vehicle call on a conflicting phase is detected;
- During the vehicle extension timer
    - If a vehicle is detected before the vehicle extension timer expires the vehicle extension timer resets and the phase remains green.
    - If no vehicle is detected and the vehicle extension timer has expired the phase terminates. This is also known as Gap Out phase termination.
    - If vehicles continue to be detected before the vehicle extension timer expires the phase will run green until the maximum green timer is met. When this is met the phase terminates. This is also known as Max Out;
- The maximum green timer starts as soon as a vehicle call is detected on a conflicting phase at any point while the currently running phase is green.

5.5.3 Experimental Setup

Table 5-6 shows the set of experiments conducted to investigate the effect of different RL design parameters as discussed above. Experiment No. 4 forms the base case and the reference point when conducting the experimental design. The choice of Experiment No. 4 parameters is based on the most commonly-used RL parameters in the literature.

Table 5-6 Design of Experiments

Problem Dimension	Exp. No.	Control Method	State Rep.	Action	Reward	Exploration Method	Demand Profile
Exploration Method	1	Q(0)	1	2	2	1 (constant ε)	Uniform
	2	Q(0)	1	2	2	1 (variable ε)	Uniform
	3	Q(0)	1	2	2	2	Uniform
	4	Q(0)	1	2	2	3	Uniform
TD Learning Method	5	Q(0.5)	1	2	2	3	Uniform
	6	Q(1)	1	2	2	3	Uniform
	7	SARSA(0)	1	2	2	3	Uniform
	8	SARSA(0.5)	1	2	2	3	Uniform
	9	SARSA(1)	1	2	2	3	Uniform
Action Definition	10	Q(0)	1	2	2	3	Variable
	11	Q(0)	1	1	2	3	Uniform
	12	Q(0)	1	1	2	3	Variable
State Representation	13	Q(0)	1	2	2	3	Variable
	14	Q(0)	2	2	2	3	Variable
	15	Q(0)	3	2	2	3	Variable
Reward Function	16	Q(0)	2	2	1	3	Variable
	17	Q(0)	2	2	3	3	Variable
Control System	18	Fixed-Time	-	-	-	-	Uniform
	19	Fixed-Time	-	-	-	-	Variable
	20	Actuated	-	-	-	-	Uniform
	21	Actuated	-	-	-	-	Variable

5.6 Results and Analysis

To study the effect of a specific parameter in factorial design analysis, the rest of the parameters should remain fixed to enable the sensitivity of the system performance to the design parameter to be examined. In Table 5-6 the parameters being examined are marked. The following sections highlight the effect of each design parameter on the RL-based signal control performance.

5.6.1 Effect of Exploration Method (Exp. No. 1 to 4)

The experiments shown below are conducted using the Q-learning algorithm to investigate the effect of different action selection strategies on the performance of the adaptive signal control system.

5.6.1.1 ε-Greedy

While it is practically infeasible to continue the learning process indefinitely, a stopping criterion is specified to bring the RL to an end. In this implementation, the learning process is terminated after 100 one-hour simulation runs as it exhibits a reasonable convergence pattern with small improvement (less than 0.1%) in the convergence curve with an increase in the number of runs. Since the only variable in this set of experiments is the exploration method, we focus the impact on convergence using the most common traffic-related performance measures. The experiments are therefore compared while considering the following measures of effectiveness (MOEs):

1. Average Delay (sec/veh): the total delay for all vehicles divided by the throughput vehicles from the intersection. This value is measured after convergence;
2. Convergence Speed: number of simulation runs required to converge. This represents the number of days required for the agent to learn the optimal control policy if the agent is deployed in the field and starts the learning process from scratch;
3. % State-Action Pairs Visited: the number of state-action pairs visited (at least one visit) divided by the total number of state-action pairs;
4. % States Visited More Than 20% of the Time: the number of states visited at least 20% of the number of total visits divided by the total number of states.

As shown in Table 5-7, no considerable improvement is observed (less that 1%) in the average delay values for different ε-functions. However, the convergence speed improves substantially when using exponentially decreasing ε with lower values of E. As shown in Table 5-7, the exponentially decreasing ε function captures more state-action space in terms of the percentage of the state-action pairs visited (from the entire state-action space) during the learning time. For a more gradually decreasing exponential ε (the lower the value of E), it is found that more state-action space is covered. A possible explanation for the minor difference in the ultimate average delay values despite the difference in the percentage of visited state-action space, is that the percentage of states that occurred frequently (e.g. more than 20% of the time) is almost the same (~75–76%) using all ε-functions; irrespective of the total % of state-action pairs visited. Therefore, visiting the most commonly occurring state-action pairs is sufficient for the learning process to converge to the same average delay. Hence it can be concluded that it is sufficient to visit the most recurring portions of the state-action space to attain good average delay at a reasonable speed of convergence.

Table 5-7 Average Delay for Different ε - Profiles

ε - Profile	Average delay (sec/veh)	Convergence speed (simulation runs)	% State-action pairs visited	% States visited more than 20% of the time
Constant (0.1)	29.922	20	81%	75%
Exp(-0. 5*n)	29.915	10	86%	75%
Exp(-0.05* n)	29.645	48	94%	76%
Exp(-0.005* n)	29.636	490	99%	76%

5.6.1.2 ε-Greedy, Softmax, and ε-Softmax

Three action selections methods are investigated and compared; ε-greedy, softmax, and ε-softmax. Since the function $\varepsilon = e^{-En}$ with E=0.05 resulted in a good compromise between the percentage of state-action pairs visited, the average delay value, and the convergence speed, this function is utilised for ε and τ. The comparison criterion in this experiment is the speed of convergence since it is the only measure that is expected to vary by changing the action-selection method. The learning curve for ε -greedy, softmax, and ε -softmax are illustrated in Figure 5-8. As shown in Figure 5-8, although the softmax, ε-greedy, and ε-softmax algorithms converge to almost the same average delay value, as expected they differ in the speed of convergence. Although softmax explores the actions more intelligently which makes it converge faster than ε-greedy, it exploits actions less aggressively than ε-greedy. Also by using the softmax selection method, the exploration rate not only depends on the value of τ (like the ε-greedy method), but also depends on the magnitude of the Q-Values, so it is harder to find a parameter setting for τ as cited by Sutton and Barto (1998): "*Most people find it easier to set the ε parameter with confidence; setting τ requires knowledge of the likely action values and of powers of e*". On the other hand, ε-softmax synergises the effect of both methods and converges in a fewer number of simulation runs. Finally, ε-softmax offers a good starting point for learning; therefore the use of ε-softmax is recommended for RL-based ATSC learning.

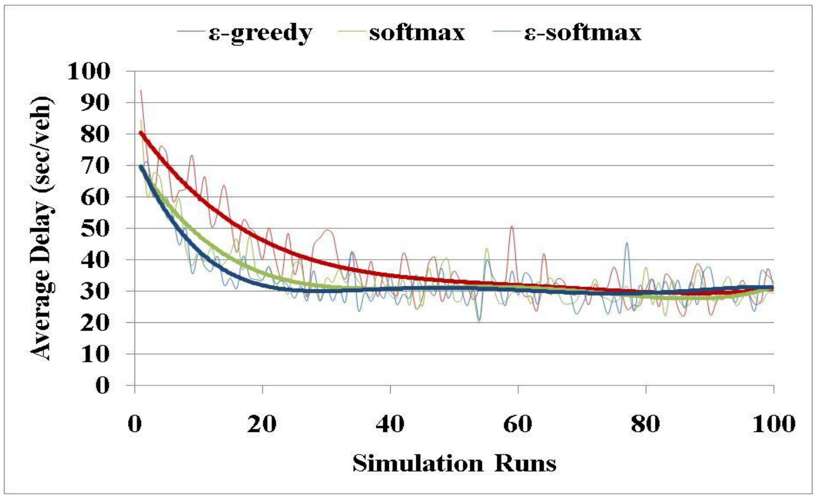


Figure 5-8 Average Delay for Different Exploration Methods

5.6.2 Effect of TD Methods on Convergence and Solution Quality (Exp. No. 4 to 9)

5.6.2.1 TD(0)

Q-learning (Q(0)) and SARSA (SARSA(0)) are tested with the ε-softmax exploration method where $\varepsilon = e^{-En}$, and E=0.05. As shown in Figure 5-9, both TD(0) algorithms converge to almost the same optimal average delay (i.e. same optimal policy). However, Q-learning converges slightly faster than SARSA. Early convergence of the Q-learning agent is attributed to the fact that the agent is trained using an off-policy method, in which case the agent learns actions that are not necessarily performed during the learning process. In other words, the agent is gaining useful experience even while exploring actions that may later turn out to be non-optimal. In SARSA, on the other hand, the agent takes actions using the currently learned policy and may select exploratory actions that are harmful to the agent's performance. Ultimately SARSA learns the same policy as Q(0) but in a longer time.

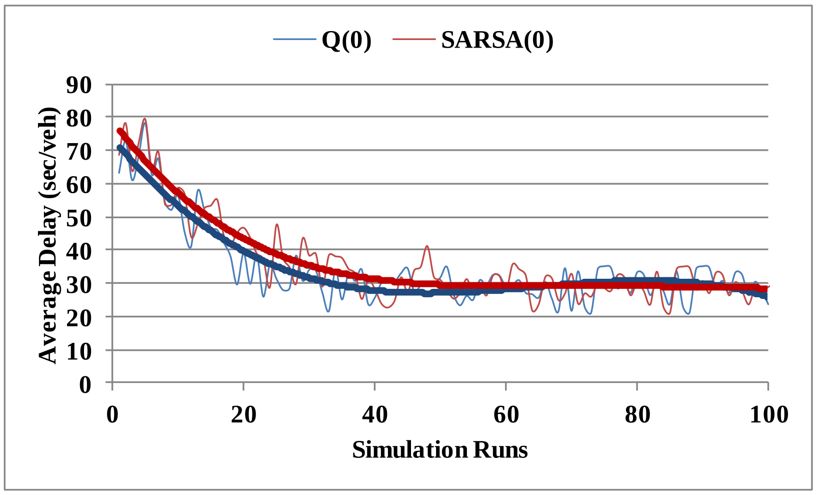
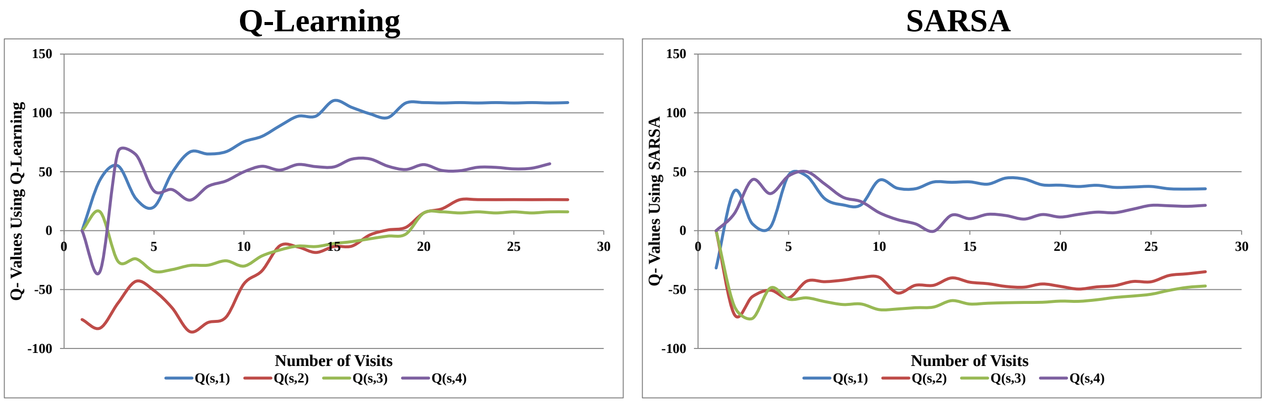


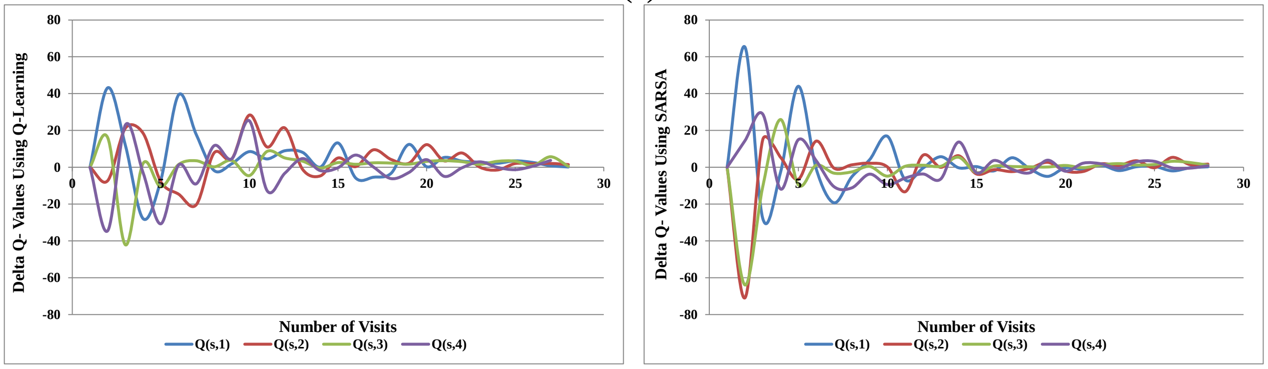
Figure 5-9 Average Delay using Q-Learning and SARSA

The convergence of the individual Q-values (not the overall delay) is also monitored for Q-learning and SARSA. For illustration purposes, the results of a sample experiment of the evolution of Q values over time are shown in Figure 5-10. In Figure 5-10, the convergence is shown for a state where queues in one direction (N–S, i.e. phase 1) were much higher than those of the other approaches. As expected, in both cases the values converged to the optimal policy, which is switching to (or extending) phase 1. However, as shown in Figure 5-10-a, Q-learning converges faster to this policy because the agent learned that action 1 is the optimal action after 5 visits (it has the maximum Q-value) while it took the SARSA agent 9 visits to reach the same optimal policy.

It is also worth noting, from Figure 5-10, that although both algorithms converge to the same policy the magnitude of the Q-values is largely different, and this is mainly because Q-learning uses the value of the greedy action at the next state while updating Q-values regardless of the actual action that will be taken.



(a)



(b)

Figure 5-10 Convergence of Q Values using Q-Learning and SARSA in a Sample State

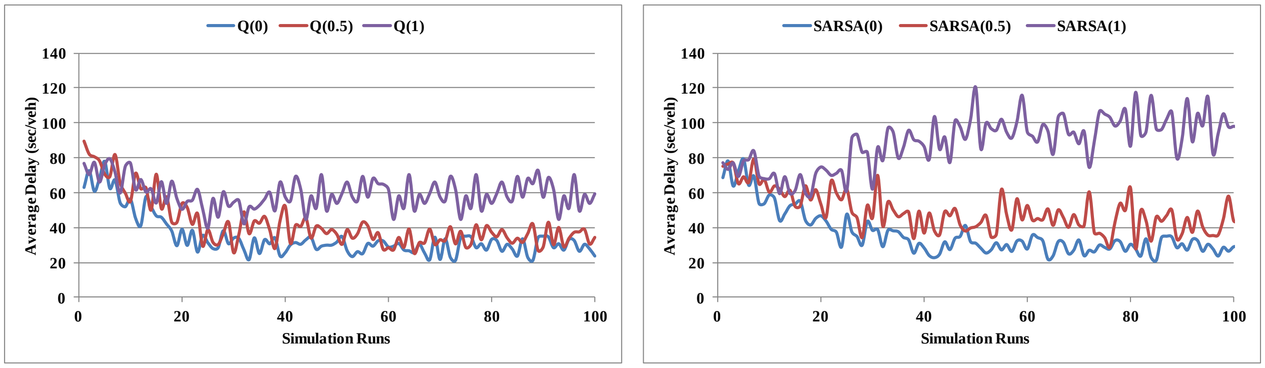
5.6.2.2 Eligibility Traces TD(λ)

As described in section 3.1.4.2.2, the eligibility traces algorithms updates the value functions based on all future rewards by emphasising states visited recently over states visited a longer time ago, which results in faster learning as shown in Figure 5-11. As shown in Figure 5-11, the higher the value of λ the faster the convergence; but unexpectedly also to a higher value of average delay. This possibly means that for high values of λ, the algorithm converges to a local minimum. One possible explanation for this finding could be the nature of the signal control problem; because the control task is a continuing task (i.e. not a finite episodic task) with a discounted reward in which looking ahead to future steps is less important compared to a finite episodic task with undiscounted reward.

To the best of my knowledge, no study in the literature investigated the effect of TD(λ) methods on ATSC. Most of the studies (de Queiroz *et al.*, 2006; Leng *et al.*, 2009) investigated TD(λ) for episodic tasks in which there is an initial state and the target is to reach a final state with an undiscounted reward.

Also, the substantial effect of TD(λ) mostly appears when rewards are delayed by many steps (Kaelbling *et al.*, 1996), which could not be adopted in this case since the reward definition is the difference in the cumulative delay between two successive decision points.

It is worth noting that Watkins' Q(λ) has a better performance than SARSA(λ), which confirms the previous conclusion that Watkins' Q(λ) does not look ahead all the way to the end of learning time; it only looks ahead as far as the next exploratory action (refer to section 3.1.4.2.2.2). In Watkins' Q(λ), the eligibility traces are reset after each exploration, therefore from time to time the agent stops looking many steps ahead in the future and starts from the beginning by looking only one step ahead, which leads to better performance compared to SARSA(λ).



(a) (b)

Figure 5-11 Average Delay using (a) Watkins' Q(λ) and (b) SARSA(λ)

For illustration purposes, the Q-values and the standard deviation (Std) across the actions are plotted for a selected sample state as shown in Figure 5-12. It is clear from Figure 5-12 that the larger the value of λ, the higher the Q-values and the lower the Std in Q-values across different actions. This means that looking ahead longer in the future confuses the agent's policy and hence the performance deteriorates.

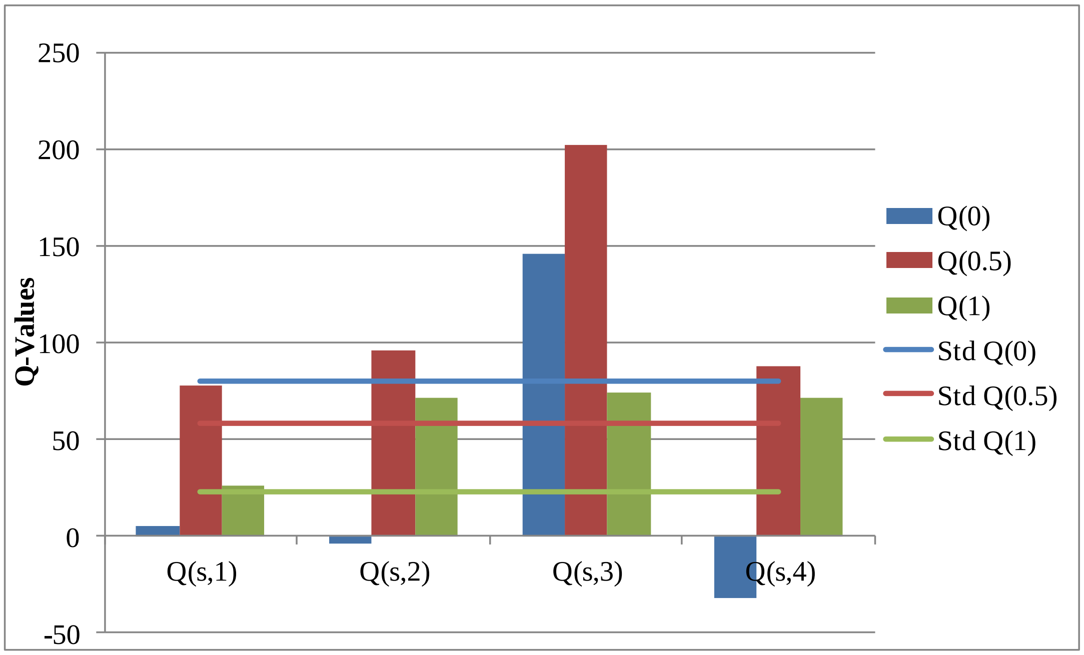


Figure 5-12 Q-Values for a Selected Sample State

5.6.3 Effect of Action Definition (Exp. No. 4, 10 to 12)

As discussed in section 4.2.2.3.2, two action definitions are considered: 1) FPS, and 2) VPS. The following performance measures are considered to judge the performance of each action definition:

- Average Delay: average delay experienced per vehicle (total cumulative delay averaged over all vehicles that crossed the intersection);
- Average Number of Stops: average number of stops per vehicle (total number of stops averaged over all vehicles that crossed the intersection);
- Average Queue Length: average of maximum queue lengths in all approaches;
- Std of Queue Length: Std of maximum queue lengths across different approaches.

FPS and VPS schemes are tested under both uniform and variable profiles. Figure 5-13 shows the above performance measures for each action definition and demand profile. Interestingly, under the uniform demand arrivals case, the FPS scheme presents a slightly better performance than VPS, in terms of the average delay and the number of stops. A possible interpretation for this result is that in the FPS case it is easier for the agent to find the optimal policy, because under uniform demand arrival rates there is less need to jump phases. Therefore activating the green phases in order may result in good performance. Also, given that the agent has only two actions to choose from, the agent can easily learn when to extend the green based on the observed traffic state. On the other hand, VPS performs better than FPS under the variable demand profile (saving 44% in average delay, 37% in average number of stops, 34% in average queue length, and 57% in Std of queue length). It is worth noting that these results are obtained under high demand level; it is anticipated that the agent will achieve similar performance using VPS in cases of a lower demand level and variable arrivals profile. In such highly stochastic arrival patterns under lower demand, the traffic state is more affected by the arrival rate than the queue spillback from the last interval (which is the case in high demand). It is noteworthy that the savings obtained using VPS are almost the same regardless of the arrival profiles, which indicates the robustness of the RL controller.

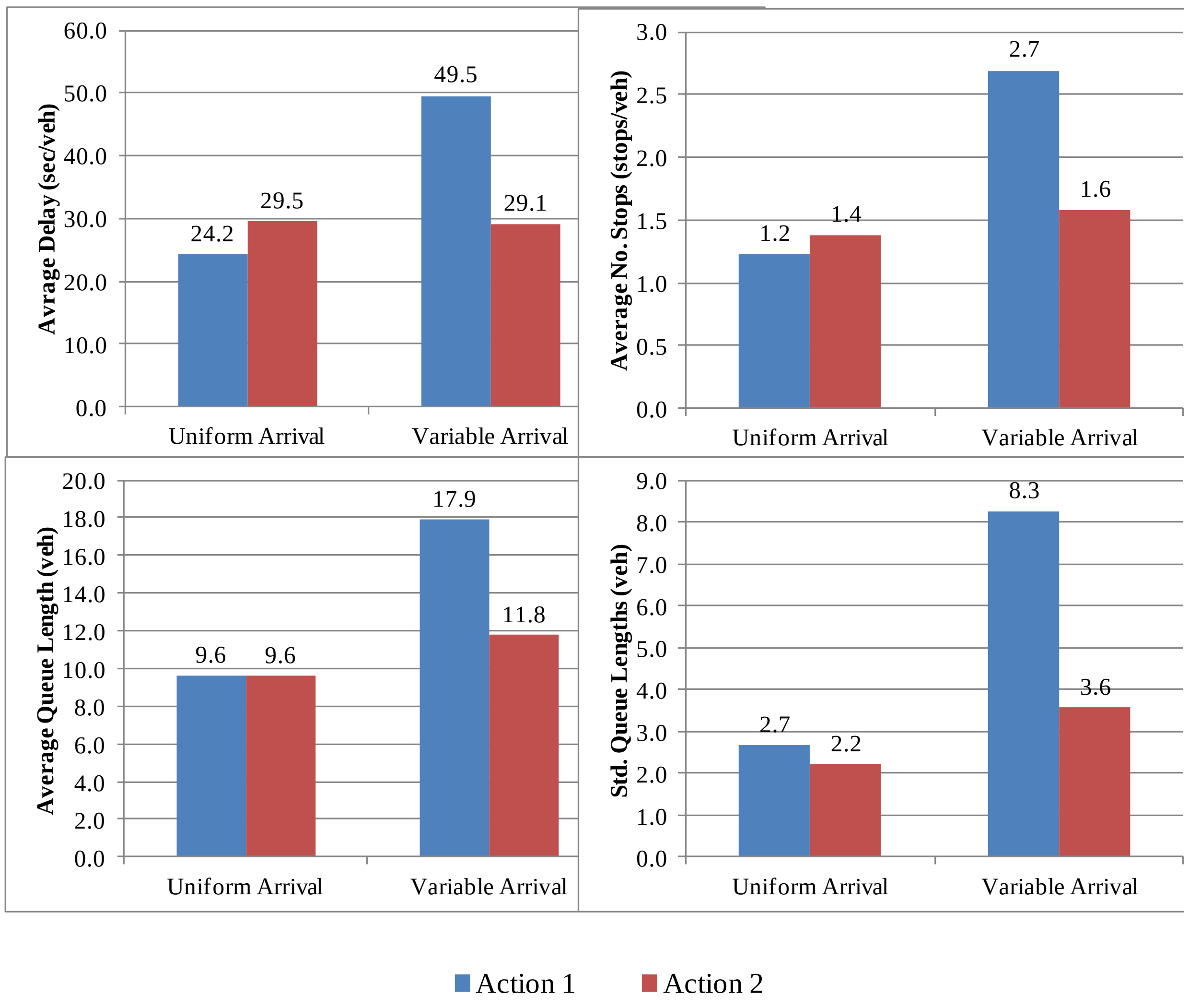


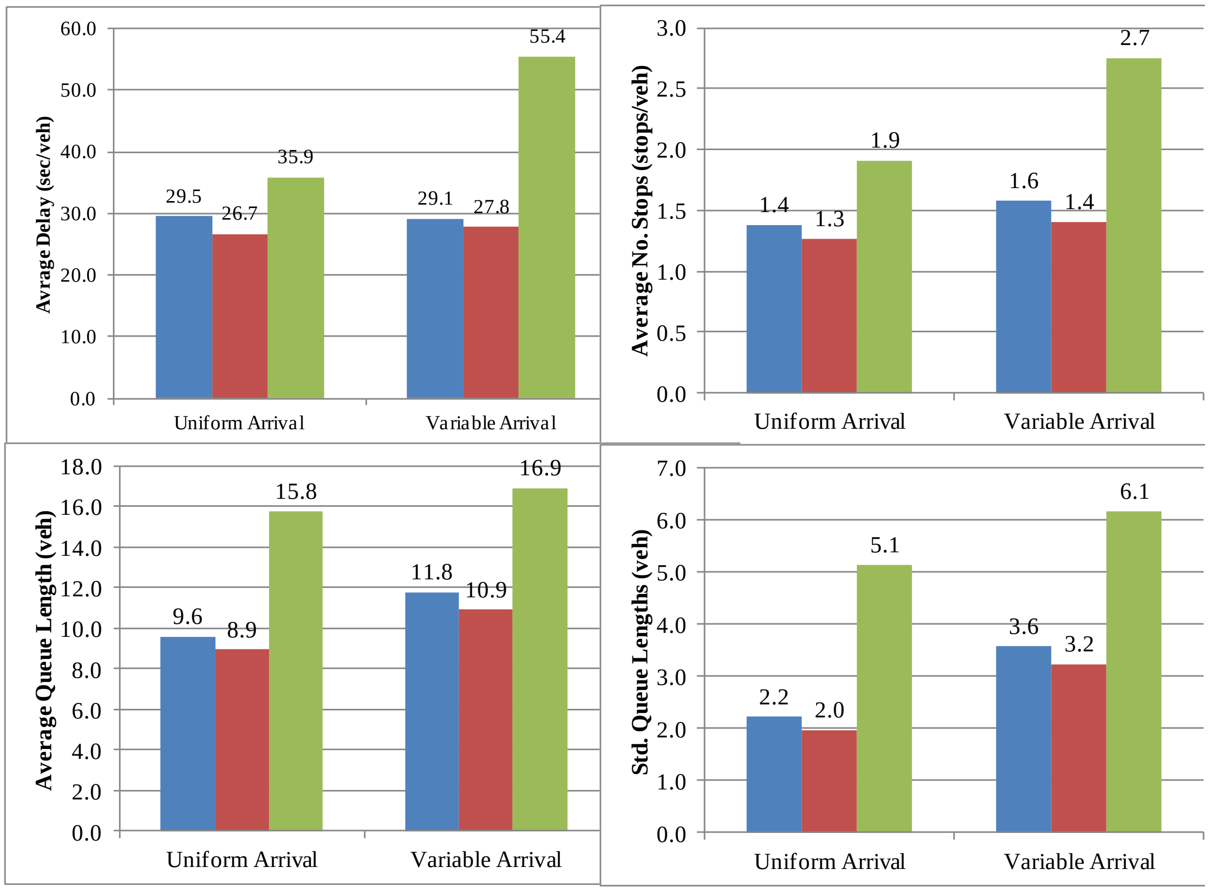
Figure 5-13 Performance Measures for Different Action Definitions under Uniform and Variable Profiles

5.6.4 Effect of State Representation (Exp. No. 13 to 15)

Three state representations are investigated under the uniform and variable demand profiles (state 1 for queue lengths, state 2 for queue lengths and arrivals, and state 3 for cumulative delays). As shown in Figure 5-14, no considerable difference is reported between the performance of state definitions 1 and 2 in terms of average delay, however state definition 2 results in a slightly better performance. By examining the traffic flow approaching the intersection, the EW direction carries a higher demand than the NS direction and both directions have the same minimum green values. State definition 2 drives the agent to extend green for the EW phases as long as the number of arrivals is high and, at the same time, the queue in the opposite direction is low; hence it endeavours to equalise and minimise the average queue lengths.

The motivation for investigating state definition 3 is the direct relation between the state definition (cumulative delays) and the reward (reduction in the total cumulative delay). However the experimental results (see Figure 5-14) show that state definition 3 performs worse than the other two definitions. This result could be attributed to the discretization bins for the state values. Determining the discretization bins for the queue length or the number of arrivals (i.e. for state definition 1 and 2) is more straightforward because its range is limited by the approaching link lengths, while discretizing the delay values can be more complex due to the wide range of delay values.

Therefore it is found that considering the queue lengths in the red phases and the arrivals to the green phase to be the best state representations, especially in case of unbalanced demand across directions.



🟦 State 1 🟥 State 2 🟩 State 3

Figure 5-14 Performance Measures for Different State Representations

5.6.5 Effect of Reward Definition (Exp. No. 14, 16, and 17)

Three reward definitions – 1) savings in delay between two successive points, 2) savings in cumulative delay, and 3) savings in total queues – are examined under variable demand profiles (see Figure 5-15). As shown in Figure 5-15, reward definition 1 performs worse than reward definition 2. This finding was expected because reward definition 1 does not consider the cumulative delay for all vehicles. The agent therefore finds it better to activate green for the phase with the highest queue length that would minimise the delay between the two successive decision points, regardless of the delay experienced by the queued vehicles before the last decision point. Hence the agent learned to activate green for the through movements (phases 1 and 3) more than the LT movements (phases 2 and 4), which maximises its objective. However, taking this action negatively affects the total intersection delay and consequently the average delay. Reward definition 1 therefore better suits intersections with two phases as conducted by Abdulhai *et al.* (2003) and Lu *et al.* (2008).

Considering the reward as the savings in the delay (reward definition 2) allows the agent to learn the optimal policy that results in the lowest total/average delay. Although the ultimate goal is to reduce the cumulative delay, the average numbers of stops and the average queue lengths also decreased.

Although reward definition 3 results in a similar queue length to definition 2, the average delay of the former is higher. This could be attributed to the unbalanced demand of the EW and NS directions, in which the signal will allocate more green to the EW phases in order to balance and minimise the queue length; hence increasing the delay of the vehicles in the NS direction. It is expected that consideration of reward definition 3 (i.e. equalising the queue lengths) better suits intersections with a balanced demand in all approaches, as conducted by De Oliveira *et al.* (2006), otherwise it will only minimise the average queue length while resulting in a high average delay in some cases.

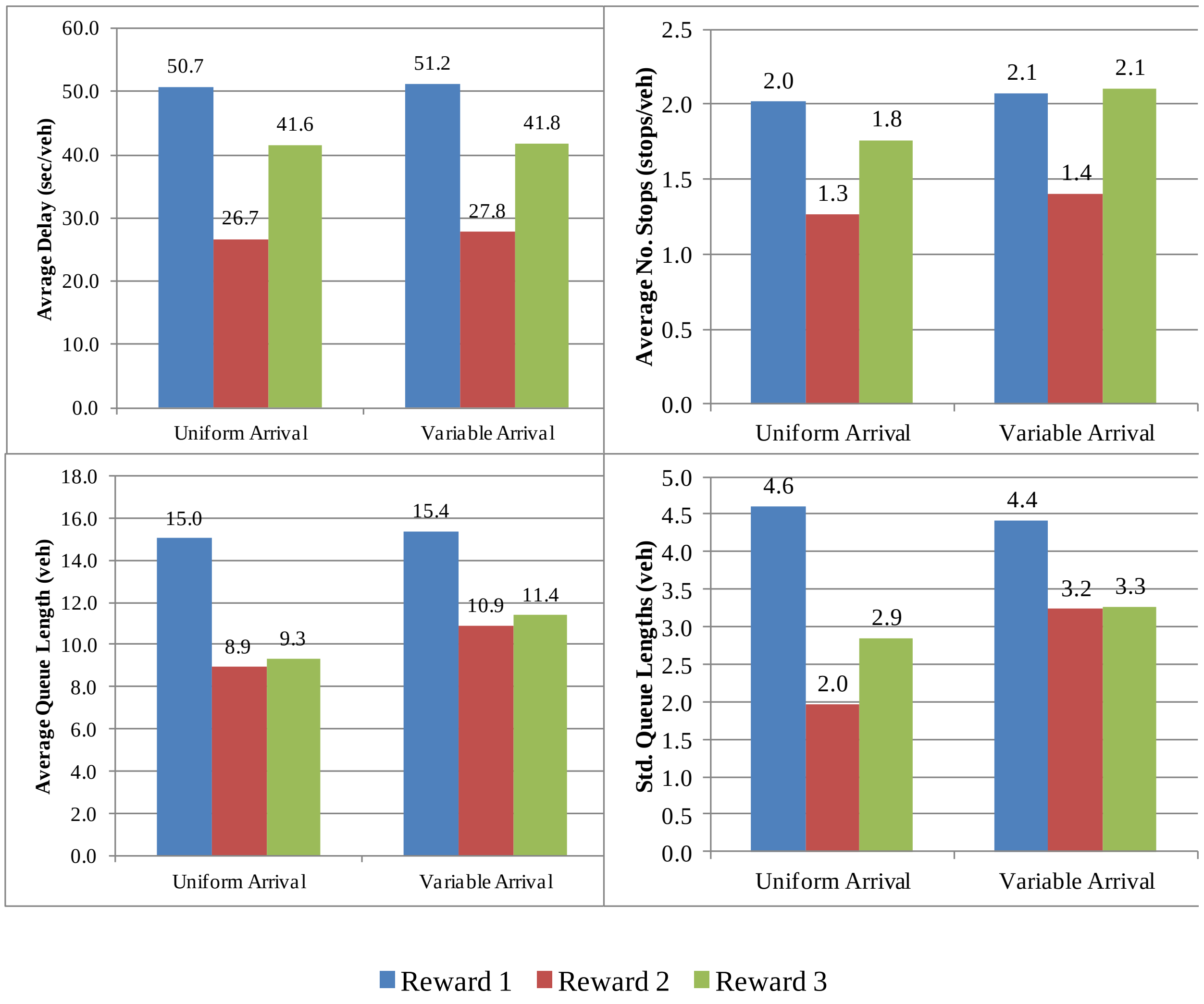


Figure 5-15 Performance Measures for Different Reward Functions

5.6.6 Conclusions on RL Design Parameters

The previous sections investigated the effect of the following design parameters on the RL-ATSC design: 1) exploration method, 2) TD learning method, 3) traffic signal phasing scheme, 4) traffic state representation, and 5) reward definition. The analysis of different RL-ATSC parameters led to the following guidelines for the selection of the best design parameters:

- The results showed that synergising the ε-greedy and softmax exploration methods using the ε-softmax method allowed for faster convergence and better performance;
- In terms of the RL methods, although it was found that the Q-learning approach performs at least as good as SARSA in terms of the average delay per vehicle, using Q-learning improves the convergence speed to the optimal policy;

- Although it was found that eligibility traces improve the learning speed for TD(λ), the higher the value of λ the worse the performance of the TD(λ) methods;
- Considering the queue lengths in the red phases and the arrivals to the green phase led to the best state representations across all experiments;
- It was found that VPS is better than FPS – especially in cases of higher variability in the arrival pattern – as it substantially outperformed FPS;
- The best reward function was found to be the reduction in cumulative delay as it resulted in the minimum average delay, average queue length, and average number of stops when compared to other reward functions.

5.6.7 RL-ATSC vs Benchmarks (Exp. No. 18 to 21):

In the previous sections several combinations of RL-based ATSC algorithms and parameter sets are compared to each other and the best combination was determined. In this section, the performance of the best-designed RL-ATSC (i.e. using ε-softmax Q-learning method with the best state, action, and reward definitions) is tested against two benchmarks traffic signal control methods: namely fixed-time control and actuated control.

The objective of the proposed RL-ATSC is to minimise the vehicle delay, improve the intersection capacity and reduce traffic congestion in general. Therefore the overall objective is two-fold; one is system efficiency, which for instance is represented by total vehicle delay on all approaches; and the other is system uniformity, which can be represented by the Std of average delay across the approaches. The lower the total delay and the lower the Std across intersection approaches, the better the performance of the control system. In addition to the overall performance measures, a set of movement-specific performance measures is also used.

5.6.7.1 Overall Performance Measures

The following MOE in the operation of an isolated intersection are used:

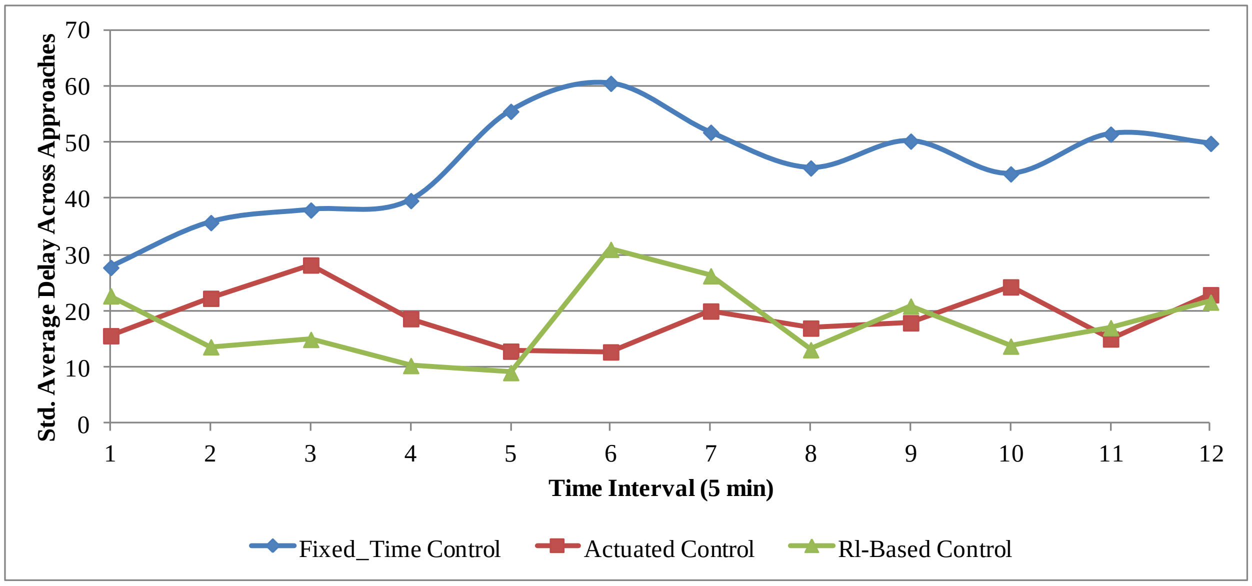
- Total Delay: the additional travel time of all vehicles relative to free flow travel time;
- Throughput: the total number of vehicles exiting the intersection, which represents the intersection capacity;
- Average Delay: the total delay divided by the number of vehicles exiting the intersection (Throughput);

- Ratio of Stopped Vehicles: the ratio of stopped vehicles to all vehicles travelling in the network;
- Average Number of Stops: the total number of stops of all vehicles approaching the intersection divided by the number of vehicles;
- Average Queue: the average number of vehicles waiting in queues per approach;
- Standard Deviations of Queue: the Stds of the average queues across approaches;
- Level of Service (LOS): LOS is a term used to qualitatively label the signalised intersection based on its performance. It converts the average delay for all vehicles to a letter grade between A to F in ascending order of delay values; with A representing the lowest delay (i.e. best operating conditions if delay<=10 sec) and F representing the highest delay (i.e. worst operating conditions if delay>=80 sec).

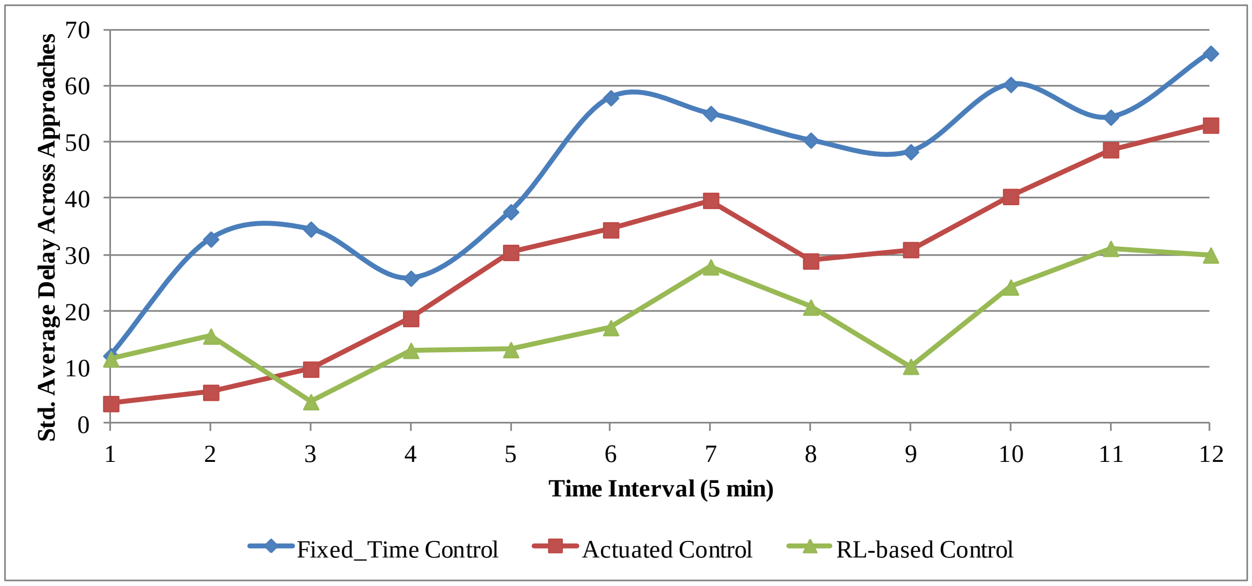
The simulation runs are replicated multiple times with different seeds (i.e. different stochasticities) to mimic the variability in traffic conditions from one day to another. Each seed number represents a different day in real-life. The average overall performance of 10 one-hour simulation replications is summarised in Table 5-8. The simulation results show that the RL-based ATSC (which is the Independent Mode version of MARLIN-ATSC) outperforms fixed-time and actuated control, regardless of the demand profile. Since the traffic flow is significantly high in all approaches, no considerable difference is reported between the fixed-time and the actuated control, especially in the case of variable profiles. However, the RL-based approach exhibits substantial savings in average delay, stops, and queue lengths, especially in the case of variable profiles. The RL-based adaptive control shows major reductions in both total vehicle delay and Std of average vehicle delay for different movements, which indicates the overall system efficiency and uniformity among the intersection approaches. To further investigate the improvements in RL-ATSC compared to the fixed-time and actuated control, Figure 5-16 shows – across the simulation hour – the Std of average delay across all movements approaching the intersection for the uniform and variable profile cases. Figure 5-17 shows the total cumulative delay within the simulation hour for each control system for the uniform and variable profile cases. Figure 5-16 shows that under a uniform demand profile, both actuated and RL-ATSC exhibit less variations in average delay across the different movements compared to fixed-time control; however RL-ATSC outperforms the other two systems in the case of the variable demand profiles. It is expected that the cumulative delay increases with the simulation hour as traffic is feeding into the network (Figure 5-17). It is interesting to compare the rate of increase (i.e. slope of the line) of the cumulative delay for different systems. It is clear that the RL-ATSC control system is more robust and efficient as it drains the network quickly such that the remaining intersection capacity can handle the generated demand later in the simulation; therefore it creates additional capacity at the intersection compared to the pre-timed and actuated control cases.

Table 5-8 Intersection Performance Measures for Different Control Systems under Different Demand Profiles

	Fixed-Time		**Actuated**		**Q-Learning**	
Profile / **Measure**	Uniform	Variable	Uniform	Variable	Uniform	Variable
Total Vehicle Delay (sec)	178482	246085	152349	187214	111729.5	122620.5
% Improvements	37.40%	50.17%	26.66%	34.50%		
Throughput (veh)	4355.20	4239	4343	4364	4388	4392
% Improvements	0.75%	3.61%	1.04%	0.64%		
Average Total Delay (sec/veh)	41.07	58.05	35.36	42.90	26.50	27.92
% Improvements	35.48%	51.91%	25.07%	34.92%		
% Stopped Vehicles (veh)	38.95%	43.25%	40.38%	42.69%	38.76%	38.09%
% Improvements	0.49%	11.95%	4.00%	10.79%		
Average No. Stops (stops/veh)	1.59	1.88	1.30	1.72	1.25	1.41
% Improvements	21.61%	25.00%	3.75%	17.90%		
Average Queue Length (veh/approach)	13.90	17.55	13.72	15.96	13.05	13.13
% Improvements	6.07%	25.22%	4.86%	17.75%		
Std. Queue Length (veh/approach)	4.95	5.48	3.66	4.47	3.65	3.64
% Improvements	26.20%	26.53%	0.34%	18.41%		
Level of Service	D	E	D	D	C	C



(a) Uniform Arrival Pattern



(b) Variable Arrival Pattern

Figure 5-16 Comparison of Standard Deviation of Average Delay across Approaches Throughout the Simulation Hour

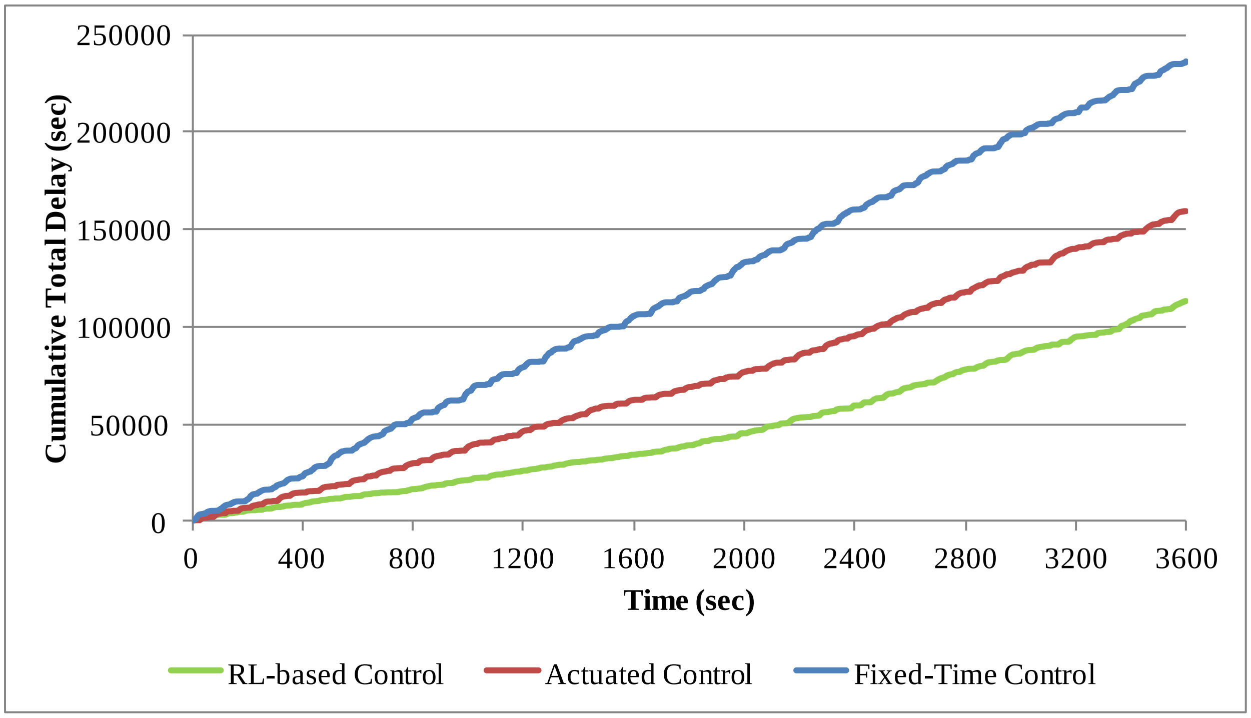


Figure 5-17 Comparison of Cumulative Total Delay

5.6.7.2 Movement-specific Performance Measures

Table 5-9 compares the results of the three types of signal control based on average vehicle delay and average queue length for each movement, averaged over 10 one-hour simulation runs. The analysis of the results in Table 5-8 leads to the following findings:

- In general, SB and WB contribute to the highest average delay and queue length. The high values associated with SB are expected because of the high demand associated with the SB and SBL movements (refer to Table 5-1). Although WB demand is not as high as EB, the demand for WBL is almost double the EBL demand which makes some of the LT vehicles block through vehicles before the LT bay, which is the reason behind the high value of average delay and queue lengths associated with WB;
- In general, the performance of the three control systems under variable demand profiles is unsurprisingly worse in terms of average delay and average queue length compared to the uniform profile, except for the SB movement using fixed-time control. This happens because SB and SBL have high volumes, so in the uniform profile case there might be some vehicles turning SBL that block others from going SB and *vice versa*. In the

variable profiles case, having different profiles for SB and SBL, as shown in Figure 5-5, allow more room for vehicles to avoid such blocking;

- As expected, and based on the network-wide findings above, the RL-ATSC is overall outperforming the other fixed-time and actuated controls in terms of average delay and average queue length. Overall, the improvements achieved by the RL-ATSC system are clearer in the variable demand case, which is expected because ATSC are best suited to adapt to stochastic variations in flow arrivals. This reflects the ability of the RL agent (when the entire state-action space is visited) to operate under abnormal traffic conditions (e.g. unusual surges in demand after concerts, games, or during emergency evacuation);
- A few exceptions to the above occur on some movements under some cases. Such exceptions occur on NB through movements in the average delay uniform case when the performance of RL-ATSC is slightly worse than the actuated control, but still much better than fixed control. A possible explanation for this exception is that the corresponding LT movement (i.e. NBL) shows relatively high savings compared to the loss in the NB through traffic delay in the uniform case. Therefore the RL decided to slightly favour the NBL movement delay to result in better overall performance (i.e. total average delay for NB traffic in RL = 13.9 + 6.1 (20 sec) as opposed to 10.2 + 11.0 (21.2 sec) in the actuated case). Similarly, although the performance of RL-ATSC, under variable profiles, on the WBL is slightly worse than the fixed-time and actuated controls, it is much better than both control systems on the WB through movement. RL decided to slightly favour the WB through movement delay to result in 50% better overall performance than actuated control (i.e. total average delay for WB traffic in RL = 31.9 + 4.1 (36 sec) as opposed to 67.4 + 3.6 (70 sec) in the actuated case). This finding might help explain why the queue length on the WB of RL-ATSC is worse than the actuated control despite the improvement in average delay for the same movements. This is primarily because queue length of WB might include vehicles that are turning WBL. It is worth noting that the ultimate goal for the RL agent is to learn the optimal control policy such that the overall intersection cumulative delay is minimised, which is clearly achieved;

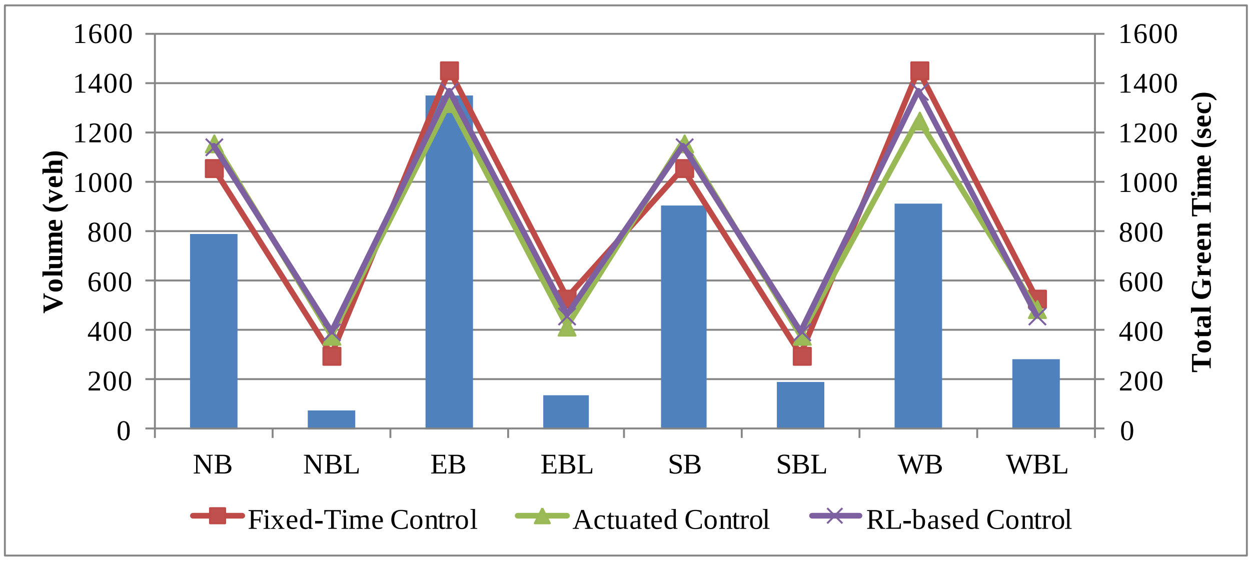
- It is interesting to show that the majority of savings in the RL-ATSC are in the EB, SB and WB directions, which are the movements that correspond to high traffic volumes. In other words, the RL is well trained to learn from the traffic pattern which movements are critical to the overall value of the intersection delay (reward), and therefore endeavours to improve their delays and queues as shown in the Table 5-9;
- Overall, it is shown that the Std in average delay and queue length across the simulation hour in RL-ATSC is less than the fixed and actuated control systems. This indicated that across the peak hour, the RL-ATSC approach is more stable and can therefore drain a higher number of vehicles, which in turn reduces the overall intersection delay.

Table 5-9 Performance Measures for Each Movement

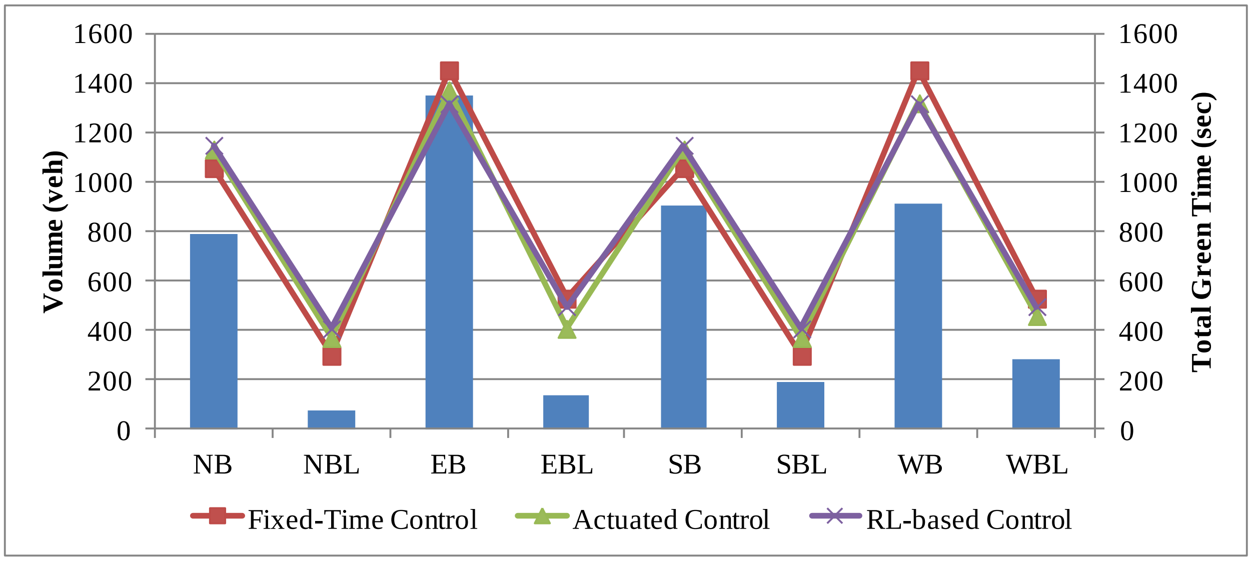
Measure		Average Delay						Queue Length					
Demand Profile		Unifrom			Variable			Unifrom			Variable		
Control / Movement		Fixed	Actuated	RL	Fixed	Actuated	RL	Fixed	Actuated	RL	Fixed	Actuated	RL
NB	AVG	20.8	10.2	13.9	22.5	13.6	14.6	10.4	7.8	8.0	9.9	9.2	7.7
	STD	4.5	6.1	6.5	8.9	10.5	16.7	2.1	1.9	1.9	5.0	4.0	3.0
NBL	AVG	11.5	11.0	6.1	11.0	10.6	8.4	3.8	3.8	3.6	3.6	3.3	3.8
	STD	4.0	4.8	3.5	5.8	8.3	7.7	0.4	0.5	0.5	0.5	0.7	0.4
EB	AVG	17.8	24.9	17.2	37.3	29.3	24.0	12.8	14.3	12.2	19.0	18.7	14.5
	STD	4.6	10.2	3.8	25.5	13.4	12.5	1.8	3.3	1.2	11.3	10.5	6.0
EBL	AVG	6.6	5.2	3.6	5.2	5.4	4.1	4.3	4.1	4.3	4.0	4.0	3.7
	STD	2.4	2.0	1.2	2.4	2.6	5.4	1.1	1.1	0.9	1.3	1.3	1.4
SB	AVG	91.0	46.1	49.6	83.6	54.3	47.8	21.8	14.0	13.7	21.2	16.5	13.5
	STD	21.3	18.3	15.4	23.7	32.2	19.9	1.3	4.0	5.4	4.0	5.9	4.1
SBL	AVG	6.9	4.1	4.3	6.8	5.2	4.0	2.7	2.7	3.0	3.4	3.1	2.7
	STD	1.4	1.0	1.7	2.4	4.0	2.2	0.9	1.0	1.4	1.7	2.0	1.1
WB	AVG	63.6	49.3	34.6	75.4	67.4	31.9	21.3	14.8	20.1	21.3	20.5	15.0
	STD	13.4	8.8	12.0	33.0	42.0	21.4	3.1	1.4	8.0	10.7	8.3	5.0
WBL	AVG	4.1	3.8	3.1	3.4	3.6	4.1	5.0	5.0	4.9	5.0	5.0	5.0
	STD	1.0	0.8	1.0	1.5	1.7	5.5	0.0	0.0	0.3	0.0	0.0	0.0

Figure 5-18 shows the total green time allocated during the simulation hour and the corresponding traffic volume for each movement. It is not surprising to find that the total green time follows the traffic volume pattern for each movement in the case of fixed-time and actuated control systems. This is simply because the fixed-time control is designed such that the effective green time for each phase is divided proportionally to the traffic volume. What is interesting though is to find that the RL-ATSC is learning that pattern by itself through the interaction with the environment by exploring all possible states. It is clear from Figure 5-18 that the three systems exhibit similar total green time patterns across the movements. What is even more interesting is to examine how that “same” total green time is distributed across the simulation hour. Therefore, Figure 5-19 and Figure 5-20 are generated to show an example of the distribution of green time for each five minutes during the simulation for the NB–SB and EB–WB movements. The figures compare the three control systems in both uniform and variable profile scenarios. The analysis of Figure 5-19 and Figure 5-20 leads to the following conclusions:

- Despite the fact that the total green time in the simulation is almost the same for the three control methods, the distribution across the simulation hour is substantially different. The RL-ATSC adapts to the change in traffic arrival pattern, which is actually what contributes the most to the outperformance of RL-ATSC over the fixed-time and actuated controls;
- It is clearly shown from the figures that the green splits of the RL-based control adapt to the demand profile. The fixed-time control remains constant regardless of the traffic arrival pattern;
- The actuated control fluctuates within the minimum and maximum green time window. The actuated control sometimes exhibits inconsistent patterns as in the case of the variable profile for the NB–SB movements (Figure 5-19). However, it shows the opposite for EB–WB (Figure 5-20), which is expected because actuated control always gives more priority to movements with highest volume within the minimum and maximum green window;
- Although the uniform arrival profile is somewhat fixed with time, the traffic is further fluctuating within the 5 minute time interval. Therefore the actuated and RL-ATSC green time allocated for each direction fluctuates according to the dynamics of the traffic conditions, even in the uniform arrival profile.

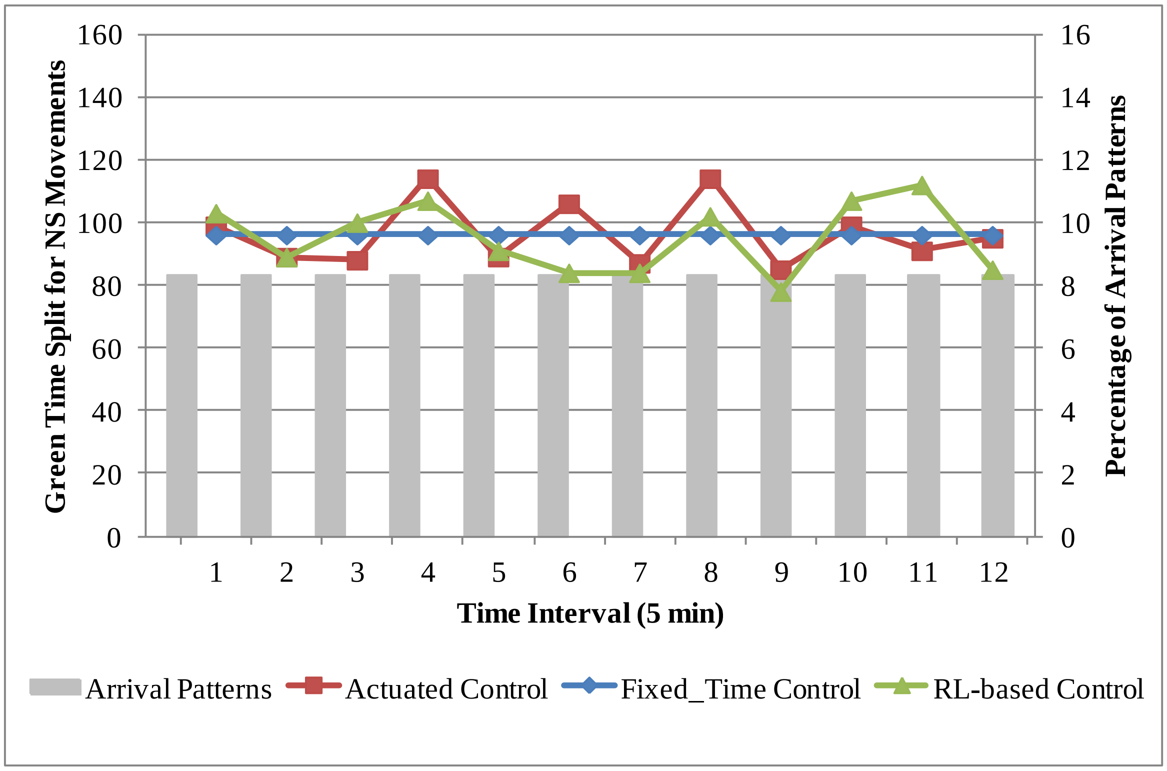


(a) Uniform Profile

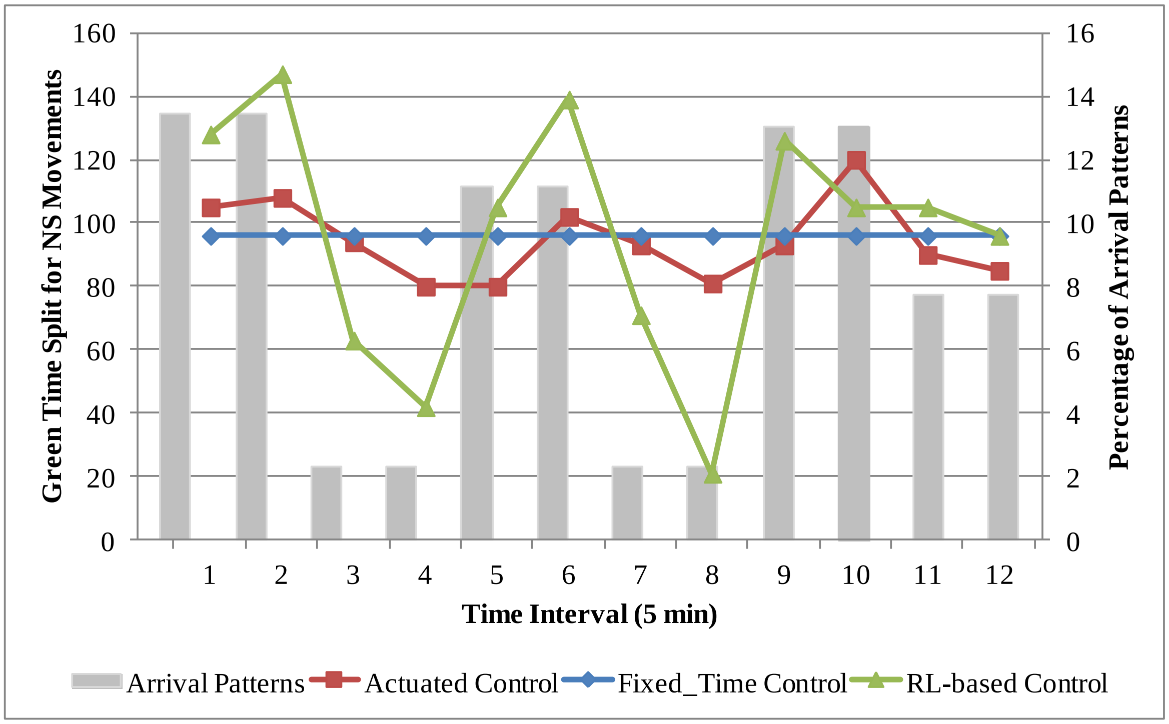


(b) Variable Profile

Figure 5-18 Comparison of Green Time Split by Movement

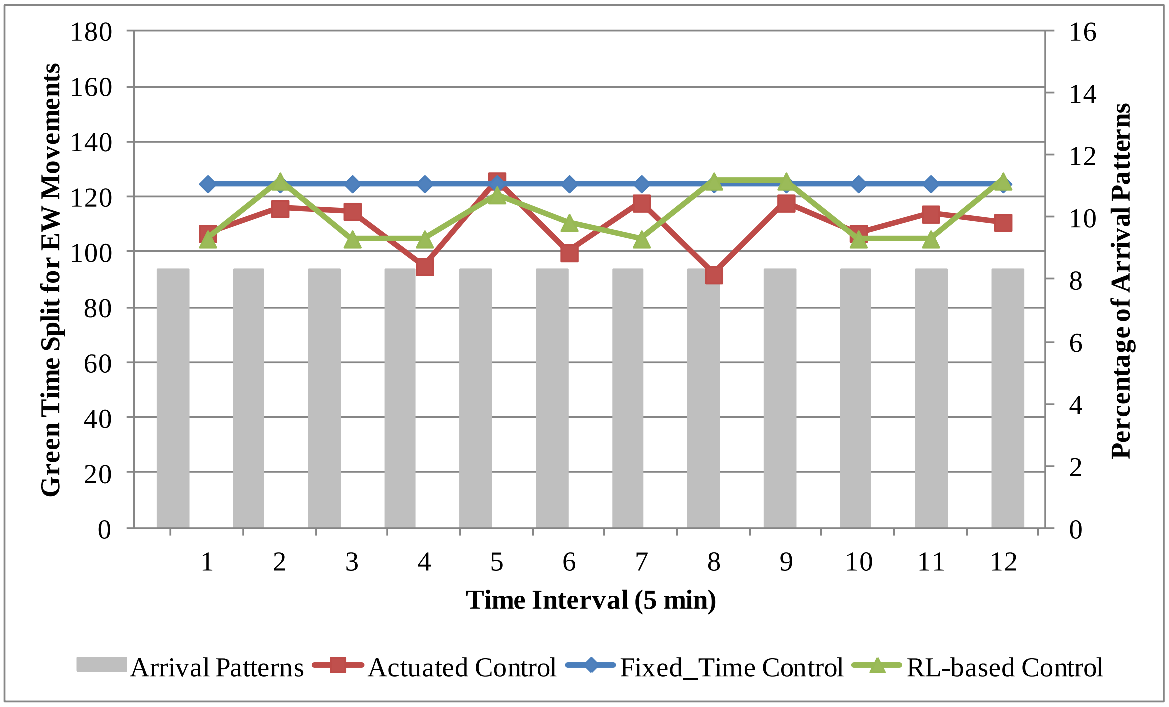


(a) Uniform Profile

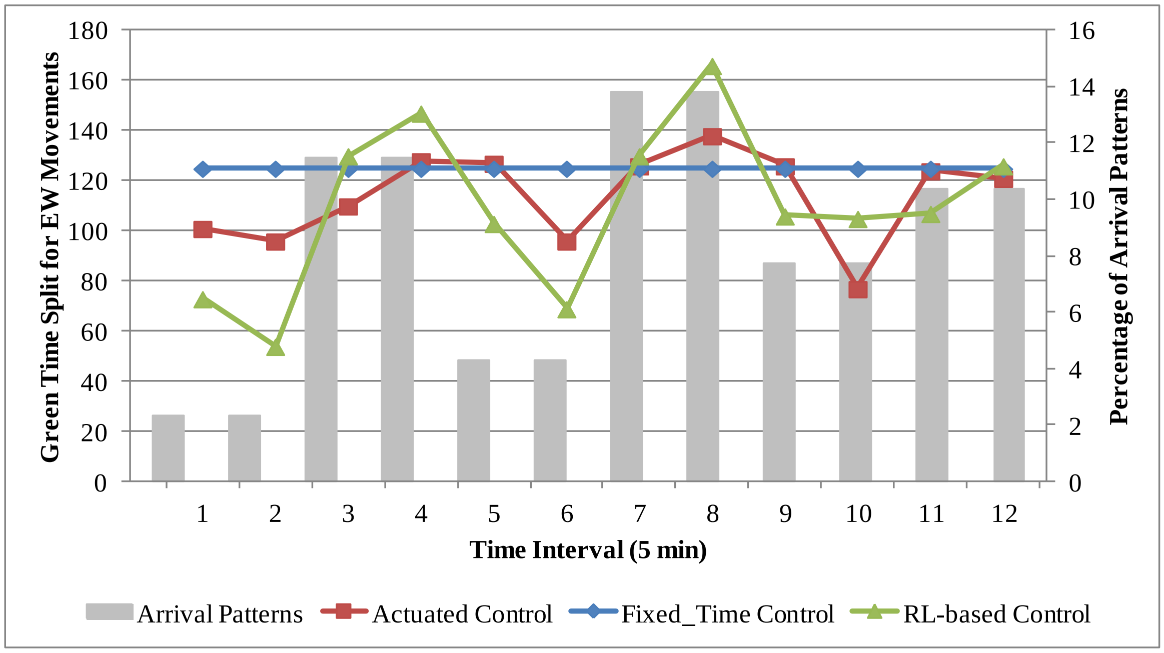


(b) Variable Profile

Figure 5-19 Allocated Green Time and Demand Arrival Percentages for NB–SB Movements



(a) Uniform Profile



(b) Variable Profile

Figure 5-20 Allocated Green Time and Demand Arrival Percentages for EB–WB Movements

5.7 Summary

In this chapter a single RL-based ATSC was applied to an isolated intersection. Paramics was used to evaluate the adaptive traffic control system on a major intersection in downtown

Toronto, using observed traffic data. This chapter endeavoured to identify the best design of the RL-based ATSC system for an isolated intersection. More specifically, the chapter investigated the following parameters/dimensions: 1) action selection method 2) RL learning method, 3) traffic state representation, 4) traffic signal phasing scheme, 5) reward definition, and 6) variability of flow arrivals to the intersection. The RL methods were compared with Webster-based fixed-time signal control and NEMA actuated control as a benchmarks. In general, the results showed that RL-based signal control consistently outperforms the fixed-time and actuated control approaches, in terms of savings in the cumulative delay per vehicle. Also it was found that RL-based control is more robust when compared to fixed-time or actuated controls, in which the same average delay was obtained regardless of the arrival profiles. The effectiveness of RL in signal control was more vivid in the variable demand profile scenario, which resulted in 52% and 35% saving in the average delay compared to fixed-time and actuated controls, respectively; which reflects its adaptability to fluctuation in traffic conditions. The results showed that synergising the ε-greedy and softmax action selection methods using the ε-softmax method allowed for faster convergence and better on-line performance. In terms of the RL methods, although it was found that Q-learning performs as well as SARSA in terms of average delay per vehicle, using Q-learning improved the convergence speed to the optimal policy. Eligibility traces on the other hand improved the learning speed for TD(λ); however the higher the value of λ the worse the performance (i.e. the average delay converges to a value). Although it was found that RL generally outperforms fixed-time control regardless of the state definition, consideration of the queue lengths in the red phases and the arrivals to the green phase was found to be the best state representation across all the experiments. VPS is recommended, especially in the case of the variable arrival rate, as it substantially outperformed the FPS. The best reward function was the reduction in cumulative delay, which resulted in the minimum average delay and average queue length. The findings of this chapter form the basis for designing RL-parameters in the ATSC problem. The next chapter extends the work to test the MARLIN-ATSC platform on a network of multiple intersections.

6 EXPERIMENTAL RESULTS 2: TWO-DIMENSIONAL PROTOTYPE NETWORK

In Chapter 5, the best set of design parameters for RL-ATSC is investigated on an isolated intersection. The RL-ATSC performance was compared to that of two benchmark traffic control methods: fixed-time control and actuated control. Although major savings are obtained for the isolated intersection test case, employing adaptive signal control strategies at the local level (isolated intersection) might limit their potential benefits. In urban traffic networks, traffic flows in two dimensions, from one intersection to the next. Traffic conditions at a given intersection affect and are affected by traffic conditions at the surrounding intersections. These effects propagate across the traffic network in a cascading fashion. For example, if a downstream intersection is congested and the queued vehicles are blocking the approaching links to that intersection, the operation of upstream intersections can be adversely affected by such a spillback. Also, the release of traffic from the upstream intersections affects the build up of traffic at the downstream intersections. Therefore the overall network would become even worse by feeding more traffic to that "critical" intersection, and as a result queues can very easily propagate upstream causing a network-gridlock that could last for hours. In essence, if intersections (i.e. agents) are made aware of the relative behaviour and performance of their neighbours; they can be trained (i.e. optimised) to ***coordinate*** their actions and operate more efficiently network-wide. Agents can aim to achieve their local objectives but also achieve a common objective for the entire network (e.g. minimise total travel time). Therefore, optimally controlling the operation of multiple intersections simultaneously can be synergetic and beneficial. However, such integration certainly adds more complexity to the problem. As discussed in sections 3.2.4.2 and 3.2.4.4, the MARLIN-ATSC system is designed to overcome the limitations in the state-of-the-practice and state-of-the-art systems by simultaneously handling multiple agents in a network of agents (intersections). The reader is cautioned not to confuse agent "coordination" in a multi-agent context with the more basic and traditional traffic signal coordination using offsets.

This chapter demonstrates the essence of the proposed MARLIN-ATSC system and builds on the lessons learned from the prototype single intersection discussed in Chapter 5. This chapter demonstrates the applicability and feasibility of the different modes of coordination of MARLIN-ATSC presented in Chapter 4 (section 4.2.2.1) in a prototype implementation on a network of 5 signalised intersections in downtown Toronto. We opted to examine a small network of only 5 intersections to be able to better isolate and understand the impact of coordination. Later, in the next chapter, we generalise the application to a much larger network of 50+ intersections in downtown Toronto. The structure of this chapter follows the roadmap shown in Figure 4-16.

6.1 Testbed Network Simulation Model

MARLIN-ATSC is tested on a simulated network of 5 intersections in the heart of the financial district of downtown Toronto (Figure 6-1). Paramics, a microscopic traffic simulator, is used to build the testbed network shown in Figure 6-2. The simulation model covers a 415400 m^2 area in the lower downtown core. The simulation model characteristics are presented in Table 6-1.

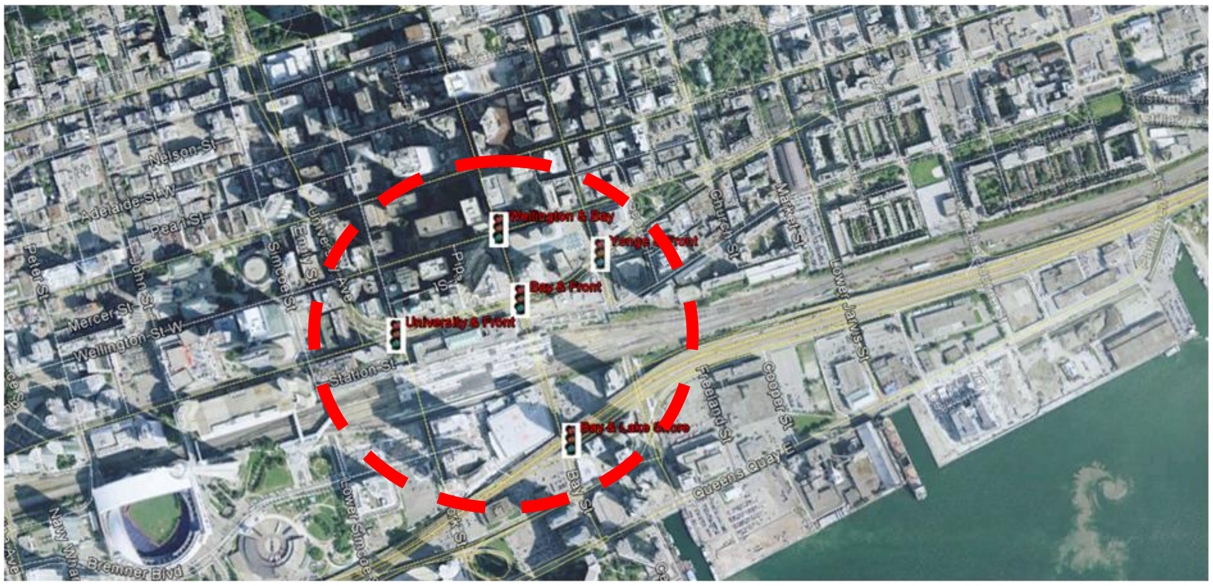


Figure 6-1 Google Satellite Image for the Testbed Network

Table 6-1 Simulation Model Characteristics

Area Covered (m^2)	670 m X 620 m = 415400
Perimeter (km)	2.85
No. of Zones	14
Length of Roads (km)	6.7
No of Signalised Intersections	5

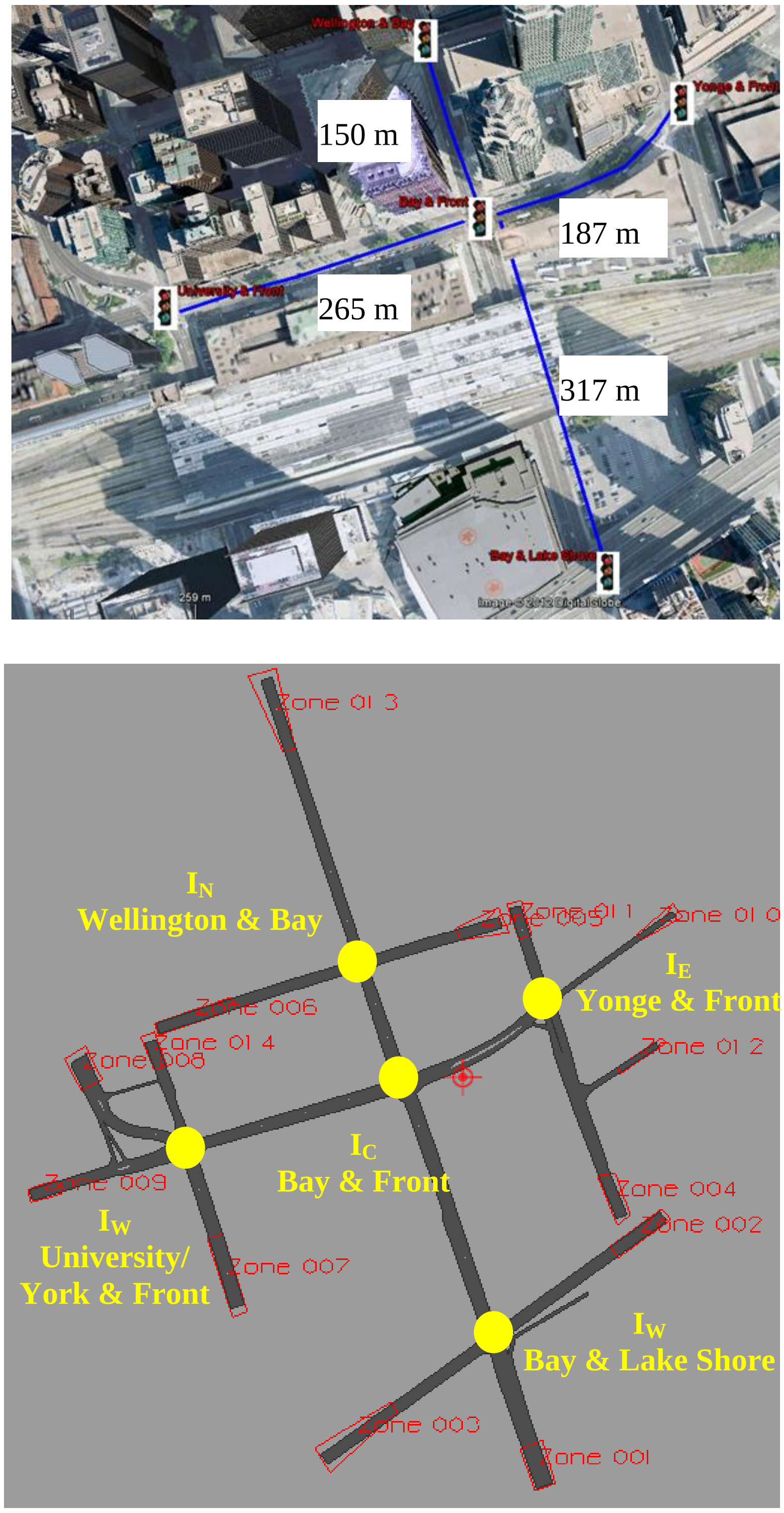


Figure 6-2 Paramics Simulation Model for the Testbed Network

6.2 Observed Traffic Counts

The latest available traffic counts from 2009 are obtained and examined for the AM and PM rush hours. It is found that the afternoon (PM) rush hour for 2009 forms the highest total demand (arrival flows) approaching the intersections, which is represented in Table 5-1. It is noteworthy that the data provided by the City of Toronto for different intersections were collected at different dates within 2009. The turning movement counts were a combination of available counts conducted during the months of April, August, September; December over different days of the week (Monday, Tuesday, Wednesday). While data discrepancy is inevitable given the different dates and days of the data provided, the turning movement counts data were scrutinised to ensure that the traffic volumes from the upstream intersections to the downstream intersection are balanced.

Table 6-2 Traffic Flow Patterns for the PM Peak Hour in 2009

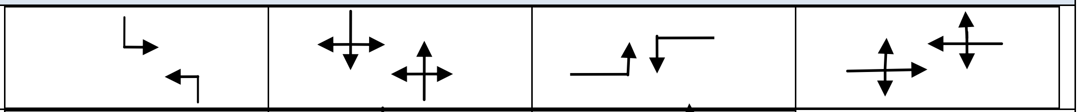
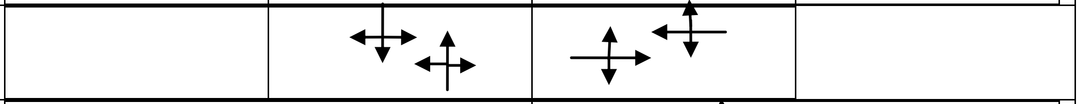
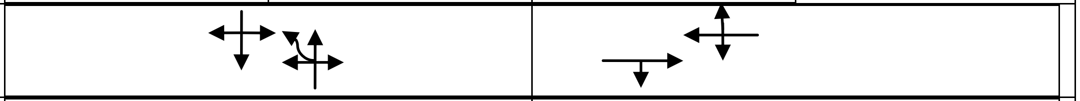
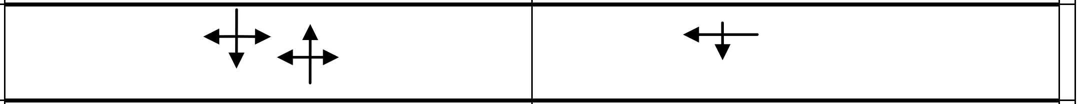
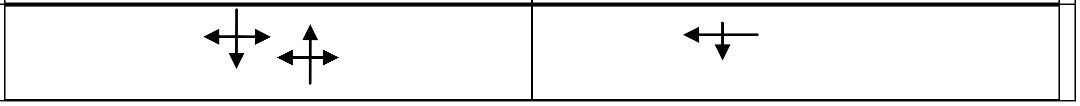
Intersection Number/ Name	Flow Parameter	Northbound (NB)	Eastbound (EB)	Southbound (SB)	Westbound (WB)
I_C Bay St. at Front St.	Through Volume	755	591	536	423
	Left-Turn Volume	92	88	105	244
	Right-Turn Volume	55	29	39	87
I_E Front St. E at Yonge St.	Through Volume	541	625	441	-
	Left-Turn Volume	130	84	43	-
	Right-Turn Volume	112	74	124	-
I_W Front St. at University Ave. & York St.	Through Volume	184	757	1,185	218
	Left-Turn Volume	423	41	0	0
	Right-Turn Volume	59	85	0	95
I_N Bay St. at Wellington St.	Through Volume	383	0	805	656
	Left-Turn Volume	41	0	0	124
	Right-Turn Volume	0	0	68	38

II_S Bay St. at Lake Shore Blvd. W	Through Volume	471	0	395	1,720
	Left-Turn Volume	105	0	0	55
	Right-Turn Volume	315	0	591	125

6.3 PHASING SCHEME DESIGN

Following the same procedure described in section 5.3, the phasing scheme for each intersection is designed as shown in Table 6-3.

Table 6-3 Phasing Scheme Design

Intersection	*Phasing Scheme*
I_C	
I_E	
I_W	
I_N	
I_S	

6.4 DEMAND MODELLING AND ORIGIN-DESTINATION MATRIX (OD) ESTIMATION

In order to employ any traffic simulation model an OD matrix that describes the trip pattern is required. In the absence of a measured OD matrix, an OD estimation process should be conducted. As briefly described in Chapter 4 (section 4.2.1) OD estimation is a collection of methods to estimate an OD matrix from a set of traffic counts and (optionally) a historical OD matrix. Although simple and straightforward to define, it is a complex and computationally very demanding multi-variable optimisation problem. The OD estimation problem is one of the most challenging problems in transportation engineering due to the fact that given a set of observed traffic data (inputs), one can obtain multiple solutions (OD matrices) that reproduce the same observed counts; i.e. there is no unique solution for the OD estimation problem. The problem can be stated as follows:

Find an OD matrix that minimises the "distance/error" between the trip values in the generated matrix and the corresponding values in the historical matrix, while simultaneously minimising the "distance/error" between the simulated traffic counts in the model and the observed counts in real-life.

In order to estimate an OD matrix in Paramics the following procedure is followed using Paramics Estimator (refer to Figure 4-6 in section 4.2.1):

1. Input Data (turning movement counts at each intersection from the field as shown in Table 5-1);
2. Model of the Environment (traffic network representation, i.e. Paramics network shown in Figure 6-2);
3. Optimisation Logic (minimise distance between modelled and observed counts, i.e. minimise GEH[2]);
4. Method of Traffic Assignment (dynamic traffic assignment employed by Paramics);
5. Convergence Criterion (number of iterations, 20 iterations were performed as the stopping criterion).

The average GEH after conducting the OD estimation process is found to be 4.1, which is satisfactory for measuring the distance between observed and modelled traffic counts (GEH of less than 5.0 is considered a good match (FHWA, 2004)). The output of the OD estimation process (i.e. the OD matrix) – shown in Figure 6-3 – resulted in around 8,300 vehicles being fed into the simulation model during the PM peak hour.

[2] The GEH is used as the statistical measure to judge on the accuracy of the estimated OD matrix when assigned to the simulation model. GEH is a statistical formula that measures the percent error with respect to the mean value of the observed (Y_{obs}) and simulated counts (Y_{sim}) for individual link (*i*) at peak period T as shown in this equation.

$$GEH = \sqrt{\frac{2(Y_{obs}(i, peak\_T) - Y_{sim}(i, peak\_T))^2}{Y_{obs}(i, peak\_T) + Y_{sim}(i, peak\_T)}}$$

From/To	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Total
1	0	0	94	48	0	0	0	0	0	0	0	0	431	0	**573**
2	50	0	1718	1	0	0	0	1	2	1	1	1	67	69	**1911**
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
4	204	0	2	0	0	0	0	0	28	58	1	1	1	1	**296**
5	0	0	0	1	0	622	0	0	2	19	7	1	19	48	**719**
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
7	0	0	0	0	0	0	0	599	0	0	0	0	39	1	**639**
8	0	0	0	1	0	0	1309	0	1	12	49	1	19	18	**1410**
9	0	0	0	1	0	0	89	1	0	465	14	143	2	10	**725**
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
11	7	0	1	2	0	0	0	0	181	50	0	436	25	0	**702**
12	0	0	0	1	0	0	0	0	1	56	554	0	1	0	**613**
13	30	0	567	1	0	56	0	0	1	82	1	1	0	0	**739**
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
Total	**291**	**0**	**2382**	**56**	**0**	**678**	**1398**	**601**	**216**	**743**	**627**	**584**	**604**	**147**	**8327**

Figure 6-3 OD Matrix

After running the simulation model using the estimated OD matrix it was found (from observing the simulation model) that the network is busy, but it did not offer the expected level of congestion during the PM rush peak. This observation may be attributed to the fact that some observed counts were conducted during the months of April and December which are relatively quieter than the rest of the months due to the end of the school year and the Christmas holidays. Also some counts were conducted on Mondays which may exhibit lower traffic volume compared to the remaining days of the week (with the exception of Friday).

Therefore, in order to advance the limit of the congestion levels in the prototype network, two demand levels were modelled. One demand level represents the actual observed demand, while the other represents a 50% increase in the total demand. The increased demand scenario reflects near future conditions given the constantly increasing population and employment and the Toronto condominium boom[3] in downtown Toronto; or it can reflect special events (e.g. sport events, parades, etc.). This scenario can also reflect construction and work zones, such as the Front St. re-reconfiguration from Bay St. to York St. currently under consideration (City of Toronto, 2012).

[3] *Dynamics of the Toronto Condominium Boom Accessed April 24, 2012, http://ellidavis.com/toronto-real-estate-news/2012/01/toronto-condominium-boom.*

6.5 Experimental Design

Similar to the isolated intersection test case, the experimental design includes the following three main tasks:

1) Choice of the parameters for the MARLIN-ATSC system
    a. exploration method
    b. MARL method
    c. state definition,
    d. action definition
    e. reward definition
2) Selection of benchmark control systems for comparison purposes
3) Setup of the experiments pool.

6.5.1 MARLIN-ATSC-Design Parameters

The choice of the following design parameters for RL-based signal control is an important task, including: state definition, action definition, reward definition, and exploration method. In Chapter 5, a comprehensive investigation of these key parameters in the RL-ATSC is conducted for an isolated intersection. Building on the lessons learned in Chapter 5, the most effective exploration method, action definition, state definition and reward definition are adopted for the MARL case as discussed in the following sections.

6.5.1.1 MARL-Methods

Since the main target of this chapter is to investigate the performance of the proposed MARLIN algorithm on a network of multiple intersections, integrated mode algorithms with indirect and direct coordination (MARLIN-IC and MARLIN-DC) are tested. The set of neighbours for each intersection are represented in Table 6-4.

Table 6-4 Neighbours

Intersection Number	Neighbours
Intersection I_C	$NB_1 = \{I_E, I_W, I_N, I_S\}$
Intersection I_E	$NB_2 = \{I_C\}$
Intersection I_W	$NB_3 = \{I_C\}$
Intersection I_N	$NB_4 = \{I_C\}$
Intersection I_S	$NB_5 = \{I_C\}$

6.5.1.2 *Exploration Method*

According to the results of the experiments in section 5.6.1, ε-softmax is chosen as the exploration method in the set of experiments presented in this chapter, where $\varepsilon = \exp(-0.05.SR)$

6.5.1.3 *Action Definition*

It was shown in section 5.6.3 that VPS is more efficient compared to the FPS. Therefore, VPS is considered for each intersection, in which case the action set represents the possible phases for the subject intersection. Table 6-5 shows the action set corresponding to each intersection.

Table 6-5 Action Set for Each Action definition

Agent	Action Set
Intersection I_C	$A_1 = \{1, 2, 3, 4\}$
Intersection I_E	$A_2 = \{1, 2, 3\}$
Intersection I_W	$A_3 = \{1, 2\}$
Intersection I_N	$A_4 = \{1, 2\}$
Intersection I_S	$A_5 = \{1, 2\}$

6.5.1.4 *State Representation*

In section 5.6.4, the overall system results of state representation 2 (arrivals and queues) outperformed that of the other two definitions. The state intervals corresponding to a certain phase are a function of the link lengths associated with that phase. Table 6-6 shows the intervals defining the state space for each intersection.

Table 6-6 State Discretization Intervals for Each Phase

Agent	State Intervals for Through Phases	State Intervals for Left-Turn Phases
Intersection I_C	{0, 1, 9, 17, >17}	{0, 1, 3, 6, >6}
Intersection I_E	{0, 1, 7, 14, >14}	{0, 1, 4, 8, >8}
Intersection I_W	{0, 1, 5, 12, >12}	-
Intersection I_N	{0, 1, 6, 14, >14}	-
Intersection I_S	{0, 1, 6, 14, >14}	-

6.5.1.5 *Reward Function*

As discussed in section 5.6.5, the reduction in the Total Cumulative Delay (see Equation 4-21) resulted in the best performance – in both the overall intersection performance and the balance among the intersection's movements – compared to other reward functions. Therefore, in this set of experiments the reward function is the reduction in the Total Cumulative Delay.

6.5.2 Benchmarks

While it is desirable to compare MARLIN-ATSC to state-of-the-practice systems such as SCOOT, the algorithmic details of SCOOT are proprietary and hence it is not possible to accurately reproduce. However, it is possible and desirable to compare the performance of MARLIN-IC and MARLIN-DC with independent MARL systems to assess the improvement to coordination. We also compare this with more traditional fixed-time and actuated control methods. Therefore the benchmarks used for evaluating the MARLIN methods are:

6.5.2.1 MARL-based Benchmarks (State-of-the-Art)

MARL systems extend RL-ATSC to a multi-agent networks but without coordination. Variants of MARL systems include:

- MARL for Totally Independent (MARL-TI) controllers system: each agent employs a Q-learning algorithm while solely considering its local state and action;
- MARL for Partially Independent (MARL-PI) controllers system: similar to MARL-TI except that each agent considers the state of the neighbouring intersections in addition to its local state. Considering the state of the neighbours without considering their actions or control policies may also be viewed as a weak form of coordination. While we do not consider this approach to be a MARLIN approach, we still use it as a benchmark for useful comparison.

It is worth noting that the design parameters of each of the above control systems (such as state representation, reward function, action definition, etc.) are the same as explained in section 6.5.1.

6.5.2.2 Traditional Benchmarks (State-of-the-Practice)

- Fixed Time control: the fixed-time signal plan is optimised using the Webster method (Webster, 1958). In fixed-time control, each demand level may result in different optimised fixed timing signal plans;
- Actuated control: fully-actuated control is applied for all the intersections in the network by following the design guidelines in section 5.5.2.2.

Progression is maintained in the tested network by employing Toronto’s offset values at each intersection. A protected LT phase in the SB and NB direction for intersection Ic – shown in Table 6-3 – is only considered in the high demand scenario because the actual demand case does not warrant a protected LT phase.

6.5.3 Experimental Setup

Table 6-7 shows the set of experiments conducted to investigate the effect of different control systems under normal and oversaturation conditions as discussed above.

Table 6-7 Design of Experiments

Demand Level	Control system	Exp. No.
Actual (Normal Traffic Conditions)	Fixed-Time	1
	Actuated	2
	MARL-TI	3
	MARL-PI	4
	MARLIN-IC	5
	MARLIN-DC	6
High (Over-Saturated Conditions)	Fixed-Time	7
	Actuated	8
	MARL-TI	9
	MARL-PI	10
	MARLIN-IC	11
	MARLIN-DC	12

6.6 RESULTS AND ANALYSIS

The results reported for each experiment are averaged over ten simulation runs after convergence. The performance of each control systems is evaluated based on three sets of MOE: overall network-wide performance measures, route-specific performance measures (as ATSC is often considered for busy routes), and intersection-specific performance measures. Since computation time is important in such complex systems, we also use MARL computational performance MOEs such as computation and convergence times.

- **Network-Wide Performance Measures**
  - Average Delay: the total delay for all vehicles in the network divided by the number of vehicles exiting the network in the simulation hour;
  - Throughput: the total number of vehicles exiting the network, which represents the network capacity;
  - Average Number of Stops: the total number of stops (either full stop of fractional stops) across all lanes for all vehicles traversing all links divided by the throughput;
  - Average Stop Time per Link: average time vehicles were stationary while traversing the link in the current interval;

- Average Travel Time per link: average travel time vehicles spend traversing the link;
- Average Maximum Queue Length: the average of the maximum queues that have formed on each of the approaches during the simulation hour;
- Standard Deviation of Queue Length across approaches
- Emissions: average CO2 emissions and fuel consumption measured by calculating the total CO2 emissions and fuel consumption in kg divided by the total distance travelled by all vehicles. An emission plugin (CMEM[4]) is used to extract CO2 emission and fuel consumption figures (Barth *et al.*, 1996).

▪ **Route-Specific Performance Measures**

- Average Travel Time: the average travel time per vehicle to travel through a specific route between defined start and end points.

▪ **Intersection-specific Performance Measures**

- Average Delay: the total intersection delay divided by the number of vehicles exiting the intersection in the simulation hour;
- Throughput: the total number of vehicles exiting the intersection, which represents the intersection capacity;
- Percentage of Green Allocation and Throughput Volumes: comparison of the percentage of green time allocated to each direction during the simulation hour and the corresponding throughput;
- Level of Service (LOS): LOS is a term used to qualitatively label the signalised intersection based on its performance. It converts the average delay for all vehicles to a letter grade between A to F in an ascending order of the delay values; with A representing the lowest delay (i.e. best operating conditions if delay<=10 sec) and F the highest delay (i.e. worst operating conditions if delay>=80 sec).

These three sets of performance measures together provide a comprehensive assessment of the performance of each control system.

▪ **MARL-Specific Performance Measures**

- Computation Time: the processing time spent in each learning step;

[4] CMEM: Comprehensive Model Emission Model was developed at UC Riverside. CMEM is a microscopic emission model that interacts with Paramics through an API. Two input files are required to define the vehicle characteristics and the reporting interval.

- Convergence Speed: the number of simulation runs required to converge.

6.6.1 Comparison of MARLIN to the State-of-the-Art MARL-based Approaches

In this section the performance of MARLIN-ATSC Integrated Mode algorithms (MARLIN-IC and MARLIN-DC) are compared with the performance of the MARL-ATSC Independent Mode algorithms (MARL-TI and MARL-PI) that represent the majority of the MARL-based ATSC systems in the literature.

6.6.1.1.1 Network Wide Performance Measures

Table 6-8 summarises the average delay and throughput for the actual and high demand cases for all MARL-based approaches, including the proposed MARLIN methods.

Table 6-8 Network Performance Measures for MARL Approaches

	Actual Demand Level		High Demand Level	
Control system	Average Delay (sec/veh)	Throughput (veh)	Average Delay (sec/veh)	Throughput (veh)
MARL-TI	17.14	7837	43.74	10862
MARL-PI	16.56	7842	42.20	11065
MARLIN-IC	15.81	7911	38.54	11241
MARLIN-DC	15.93	7834	39.11	11125

The analysis of the results in Table 6-8 leads to the following conclusions:

- In the actual and high demand cases, MARLIN-DC and MARLIN-IC exhibit almost the same performance in terms of average delay and vehicle throughput;
- In the actual demand case, MARLIN-based systems results in slightly better performance compared to the independent MARL-based systems. The savings range from 2% to 8% in average delay and from 0% to 1% in throughput. The small scale of improvement however, despite the methodically appealing and sound coordinated MARLIN approach, is not surprising. To the contrary it is even reassuring. One reason is the fact that the actual demand is relatively low compared to the available capacity – in which case the independent RL serves the traffic flow relatively well without the need for coordination among intersections/agents. In other words, traffic flow from one intersection rarely spills

back to the upstream intersections as shown in Figure 6-4 by the max queue length in the actual and high demand cases within the simulation hour. As can be shown from the figure, the maximum queue length in the actual demand case is around 20 vehicles, which corresponds to around 120 m – which is less than the minimum distance between adjacent intersections (shown in Figure 6-2). Therefore, congestion at downstream intersections is not affecting the actions of the coordinated agents, and hence it is not unexpected to obtain a similar average delay and throughput in the actual demand case. Theoretically, in the actual demand case the environment is almost stationary (i.e. the reward the agent receives after executing a certain action depends primarily on that action and is not directly affected by other agent's actions), therefore independent RL-based agents can converge to the optimal policies in these environments as mentioned in section 3.2.4.2;

- In the high demand case, MARLIN-based systems result in better performance compared to the independent MARL-based systems. The savings range from 9% to 11% in average delay and from 2% to 4% in throughput. In contrast to the low demand case, the maximum queue length in the high demand case, as shown in Figure 6-4, reaches the adjacent intersections, therefore the coordination among agents results in higher savings in average delay and throughput when compared to the actual demand case;
- Comparing the Independent Mode algorithms in the high demand level, it is found that considering the states of the neighbouring intersections in MARL-PI outperforms the totally independent operation MARL-TI with 4% savings in average delay and 2% improvement in throughput. This shows that even a limited level of coordination by sharing the states is beneficial compared to the agents acting in total isolation.

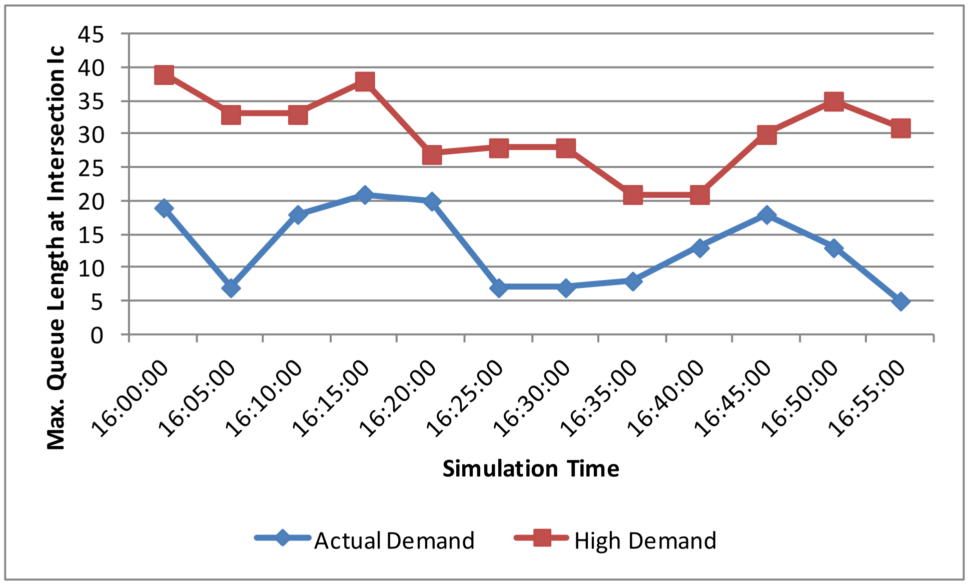


Figure 6-4 Maximum Queue Length Approaching Intersection Ic for Actual and High Demand Levels

6.6.1.1.2 Route-Specific Measures:

Although the average delay and throughput show the relative network-wide effectiveness of each traffic control system, examining route-specific measures would result in a better understanding of the effect of coordination-based control systems. Table 6-9 shows the travel time for five main routes in the testbed network under the high demand level: Route 1: Bay St. NB, Route 2: Bay St. SB, Route 3: Front St. EB, Route 4: Front St. WB, and Route 5: Yonge St. NB–Front St. WB–Bay St. SB. The effect of coordination in MARLIN-based methods is more distinct along Route 4 and Route 5 (see Table 6-9).

The analysis of the results shows that MARLIN-IC and MARLIN-DC exhibit similar route travel times. Furthermore, MARL-PI performs better than MARL-TI. Table 6-9 shows that MARLIN-IC methods have the potential to reduce route travel times by up to 26% on Route 4 and 35% on Route 5 when compared to the MARL-PI method. In addition, MARLIN-IC methods exhibit more savings in route travel times when compared to MARL-TI by around 43% on Route 4 and 49% on Route 5.

Table 6-9 also shows the Std of travel time along the five routes. It is shown from Table 6-9 that MARLIN generally exhibits stable travel times when compared to MARL-based systems. In particular, MARLIN shows substantial savings in travel time variations for Route 4 and Route 5 by 80% and 85%, respectively, when compared to the other MARL methods. This implies that

MARLIN methods can provide more reliable travel times, which is of equal (if not more) importance to travel time savings.

As shown in Table 6-9, MARLIN exhibits less variation in travel time across the simulation hour when compared to MARL methods, particularly in Route 5. After visualising the simulation and examining the demand pattern, it was found that a relatively high demand is assigned from Zone 4 to Zone 1 and other zones passing through the link connecting intersections I_E and I_C; which eventually translates to a high NBL volume as shown in Figure 6-5. Releasing such high demand relative to the available capacity on the link connecting I_E and I_C causes the link to reach its saturation capacity and results in queue spillback, and will therefore affect the travel time along Route 4 and Route 5. When such spillback occurs it becomes beneficial for the two intersections to coordinate their actions.

Table 6-9 Route Travel Times (min)

Demand Level	Traffic Control	Average Travel Time Statistical	Route 1	Route 2	Route 3	Route 4	Route 5
High Demand Level	MARL-TI	Avg	3.08	3.64	3.34	2.69	4.19
		Std	0.75	0.37	0.73	0.87	1.68
	MARL-PI	Avg	2.78	3.74	3.30	2.06	3.34
		Std	0.73	0.45	0.38	0.54	1.20
	MARLIN-IC	Avg	3.04	3.80	3.23	1.52	2.16
		Std	0.80	0.31	0.57	0.14	0.25
	MARLIN-DC	Avg	3.10	3.75	3.02	1.71	2.42
		Std	0.92	0.29	0.35	0.38	0.46

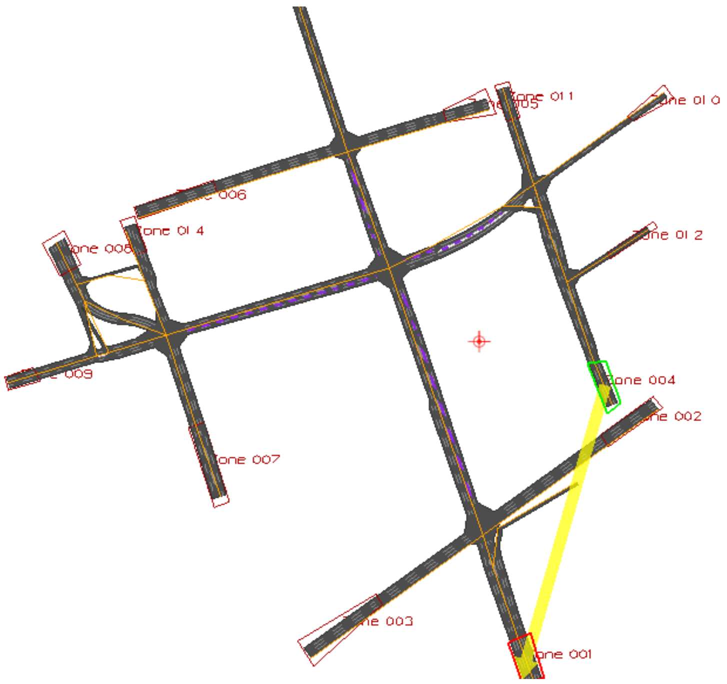


Figure 6-5 High Demand from Zone 4 to Zone 1

6.6.1.1.3 Intersection-Specific Measures

To further investigate the reasons that cause the average delay and travel time differences between MARL and MARLIN methods, intersection-specific measures are produced for the high demand case and presented in Table 6-10. It is shown from Table 6-10 that the efficiency of MARLIN is more profound at intersections I_C and I_E, which is expected because these intersections are largely affected by the high demand to capacity ratio of the two closely-spaced intersections.

Table 6-10 Intersection-Specific Average Delay and Throughput

Intersection	Measure	MARL-TI	MARL-PI	MARLIN-IC	MARLIN-DC
I_C	Avg. Delay (sec/veh)	75.69	71.44	67.98	68.03
	Throughput (veh)	3635	3716	3680	3680
I_E	Avg. Delay (sec/veh)	64.89	56.57	42.66	43.58
	Throughput (veh)	2902	3150	3382	3081
I_W	Avg. Delay (sec/veh)	30.43	30.41	29.04	30.83
	Throughput (veh)	4506	4629	4573	4597
I_N	Avg. Delay (sec/veh)	32.36	31.96	31.92	32.04
	Throughput (veh)	3041	3001	2994	3024
I_S	Avg. Delay (sec/veh)	28.03	28.80	26.76	27.35
	Throughput (veh)	5077	5045	5028	5054

It is interesting to find that these conditions confuse the MARL-based independent agent at intersection I_E, because the agent receives different rewards for the same state-action pair. To illustrate, assume the case when intersection I_E has a long queue for NBL, this could happen under one of the following scenarios: 1) when I_C has a spillback queue on the WB approach, or 2) when there is no spillback for I_C. In each of the previous scenarios, the reward will be different for the same action for agent I_E which is "switch to the NB LT phase". In scenario 1, vehicles could not cross the intersection due to the blocking queue spillback from the downstream intersection (I_C), hence the delay will increase and the reward (penalty) will result in a high negative value. However, in scenario 2, the reward will be positive due to the reduction in the delay associated with the vehicles that exit I_E without being held back as in the case of scenario 1. Figure 6-6 shows the average delay of intersection I_E for 20 simulation runs after convergence. As shown in Figure 6-6 MARL-TI exhibits higher variations in average delay for I_E compared to those of MARLIN-IC, which means that the independent agents fail to converge under oversaturated conditions at the downstream intersections (e.g. I_E).

These observations are interestingly identical to the theoretical fact discussed in section 3.2.4.2; that is the single agent RL is proven to only converge to the optimal policy if the environment is *stationary*. In the traffic control problem stationarity might exist in the case of low demand, but it is not the case in oversaturated conditions when each intersection is affected by the actions taken by neighbouring intersections, especially for closely-spaced intersections. In the high demand case, each agent is faced with a moving-target learning problem in which the agent's optimal policy changes as the other agents' policies change over time. This analysis applies for any independent MARL agents (both MARL-TI and MARL-PI). However, considering the states of the neighbouring agents helps improve the performance of MARL-PI compared to MARL-TI.

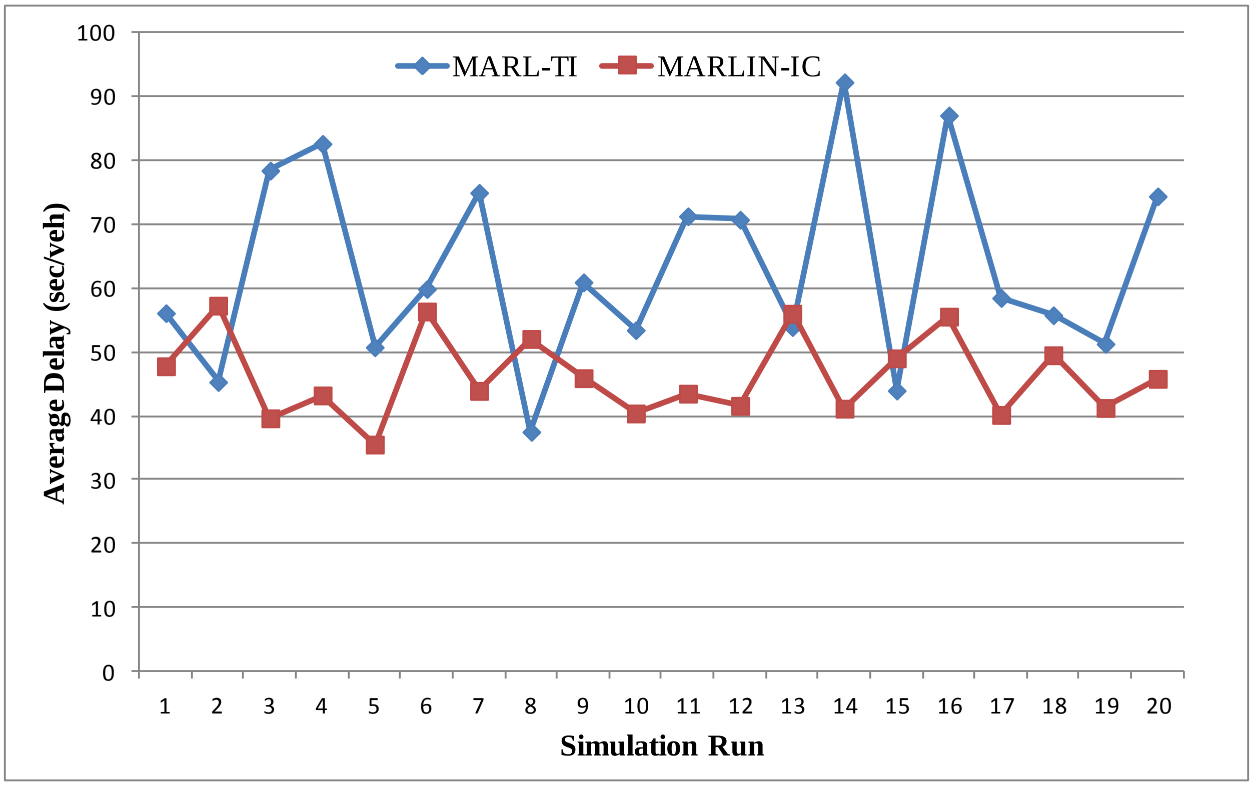


Figure 6-6 Average Delay for 20 Simulation Runs for MARLIN-IC and TI-MARL

As noted in section 6.5.1.3, VPS is considered in which the agent is not only optimising the green time for each phase but also optimising the phasing sequence. This is an additional reason behind the superior performance of MARLIN compared to independent MARL methods, especially at intersections I_E and I_C. In such cases the sequence of phases will largely affect the queue lengths (spillback) and could prevent such spillback by metering the traffic from the upstream intersections. This is more obvious at intersections I_E and I_C because they have more than two phases, so learning the optimal phasing sequence makes a substantial difference compared to the intersections with only two phases (I_W, I_N and I_S).

To better illustrate the difference in the operation of independent and coordinated agents that leads to the above-mentioned performance difference, Figure 6-7–Figure 6-11 show the aggregate percentage green time allocation and the throughput volume associated with each phase for each intersection using MARL-TI and MARLIN-IC.

It can be concluded from Figure 6-7–Figure 6-11 that MARLIN coordinated agents distribute the green time across the phases more wisely to discharge more vehicles (with lower average delay) compared to non-coordinated MARL-based agents. In such cases the agents learn to only activate green if the result – by coordinating this action with neighbouring intersections – would increase the throughput (i.e. allow more vehicles to pass the intersection) and therefore reduce delay by managing the cross-blocking delay. The following are some findings drawn from Figure 6-7–Figure 6-11.

- At intersection I_E, although high queues in the NBL lane are observed, the MARLIN agent does not switch to the corresponding NBL phase in cases of queue spillback from the downstream intersection (I_C) (18% in MARLIN-IC vs 26% in MARL-TI). This is known as ***metering*** the traffic to the downstream intersection which results in lower green time being allocated for the NBL phase at I_E, but with higher overall throughput (2573 in MARLIN-IC vs 2379 in MARL-TI, i.e. an 8% increase in intersection throughput) from the intersection compared to the independent MARL-based case as shown in Figure 6-8;
- At intersection I_C, MARLIN-IC learns to give priority those phases associated with closely-spaced upstream intersections such as I_E and I_N (4% increase in green time for NS and EW movements at the expense of NBL and EBL). If there is equal need for green from different approaches as shown in Figure 6-7, the MARLIN agent at I_C dedicates more green to the through NB and SB phases because of the short distances between I_C and I_N and I_E than the MARL-TI agent. From Figure 6-7, it can be shown that MARL-TI results in a lower ratio of throughput to green (veh/sec green), which indicates the higher frequency at which vehicles are blocked and are forced to come to a halt under a green light. Also the throughput of MARLIN-IC (3790 veh) is higher than that of MARL-TI (3708 veh).

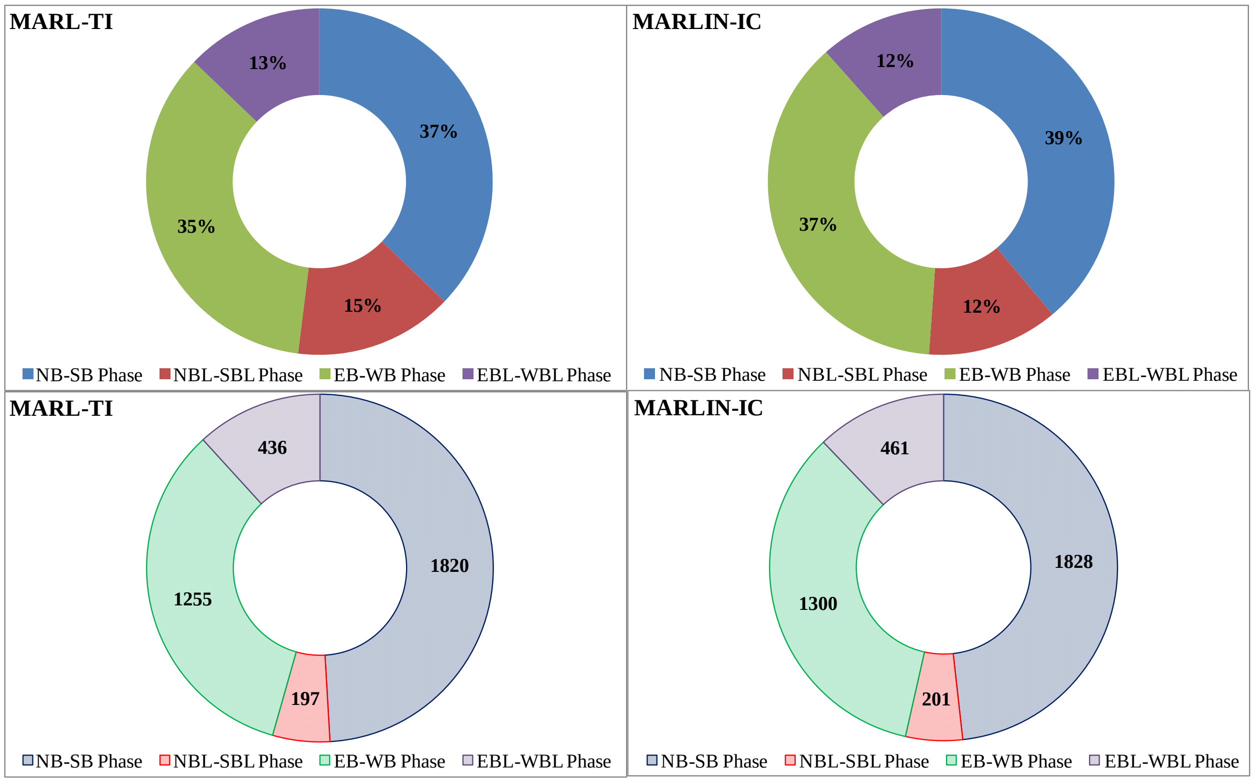


Figure 6-7 Percentage of Green Allocation and Volumes for Intersection I_C

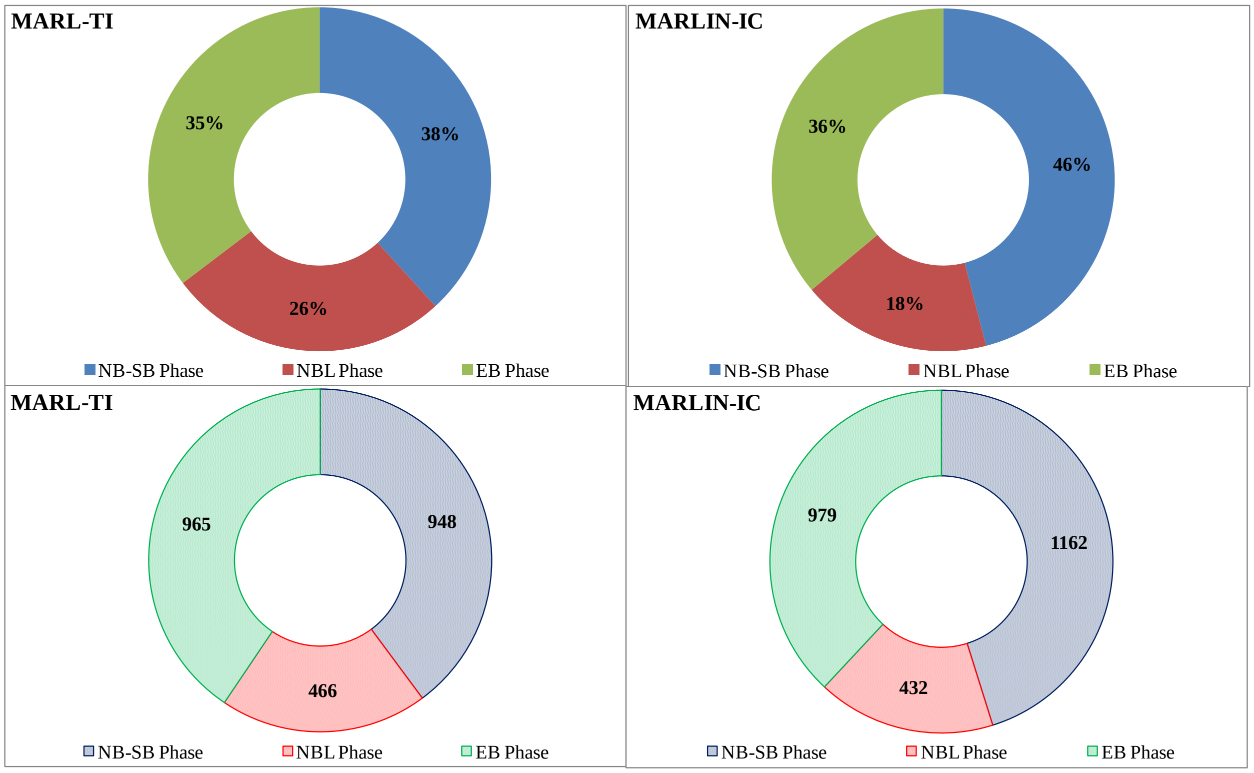


Figure 6-8 Percentage of Green Allocation and Volumes for Intersection I_E

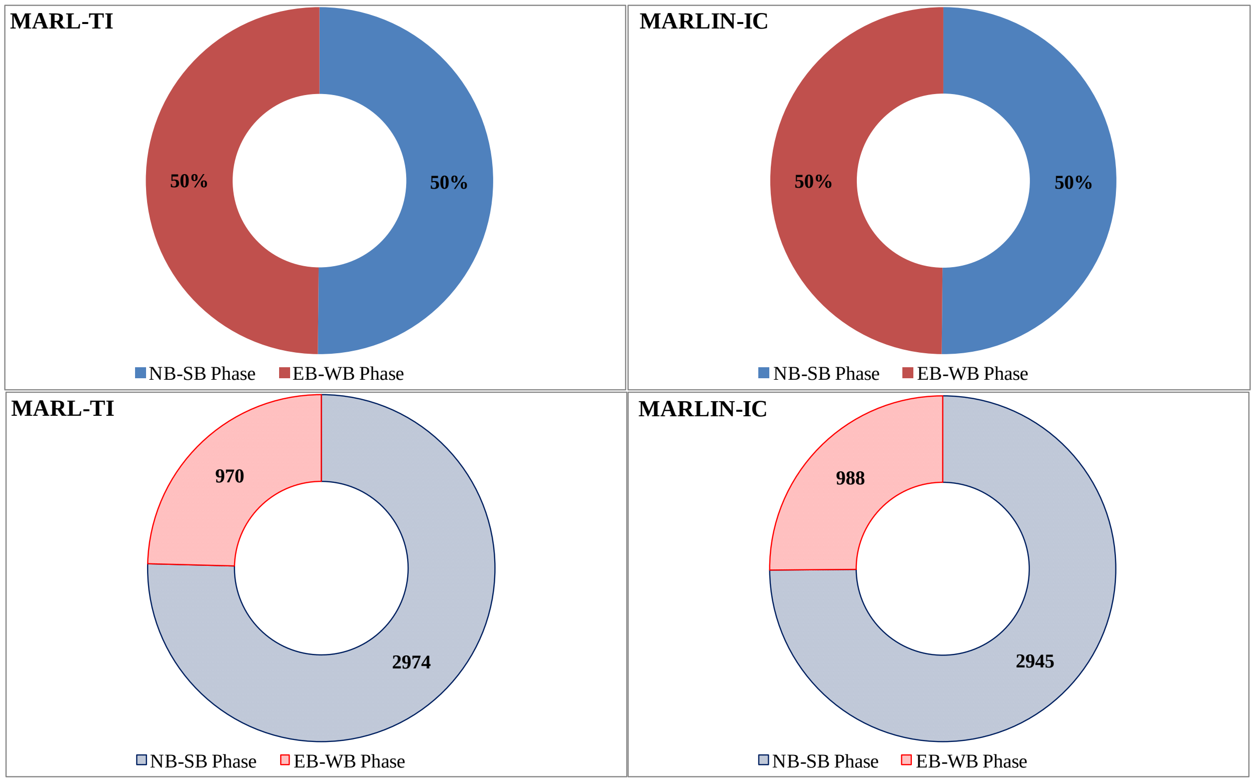


Figure 6-9 Percentage of Green Allocation and Volumes for Intersection I_W

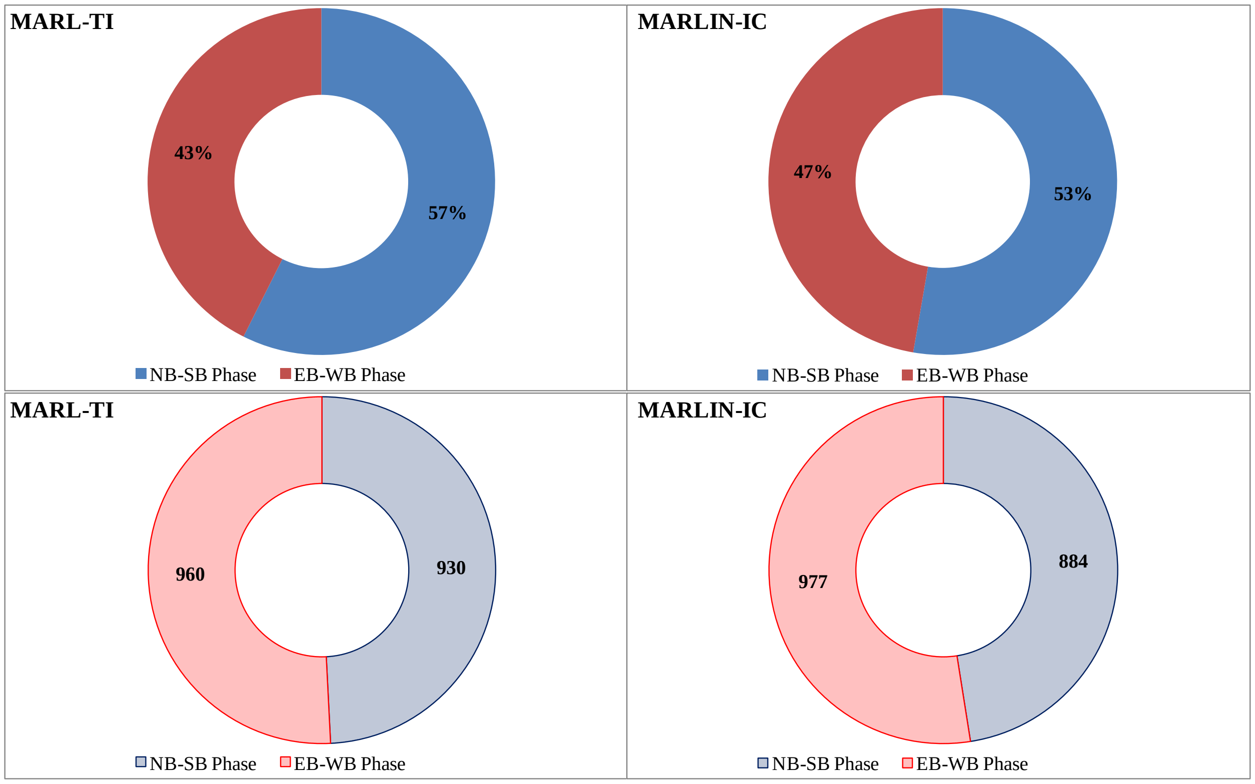


Figure 6-10 Percentage of Green Allocation and Volumes for Intersection I_N

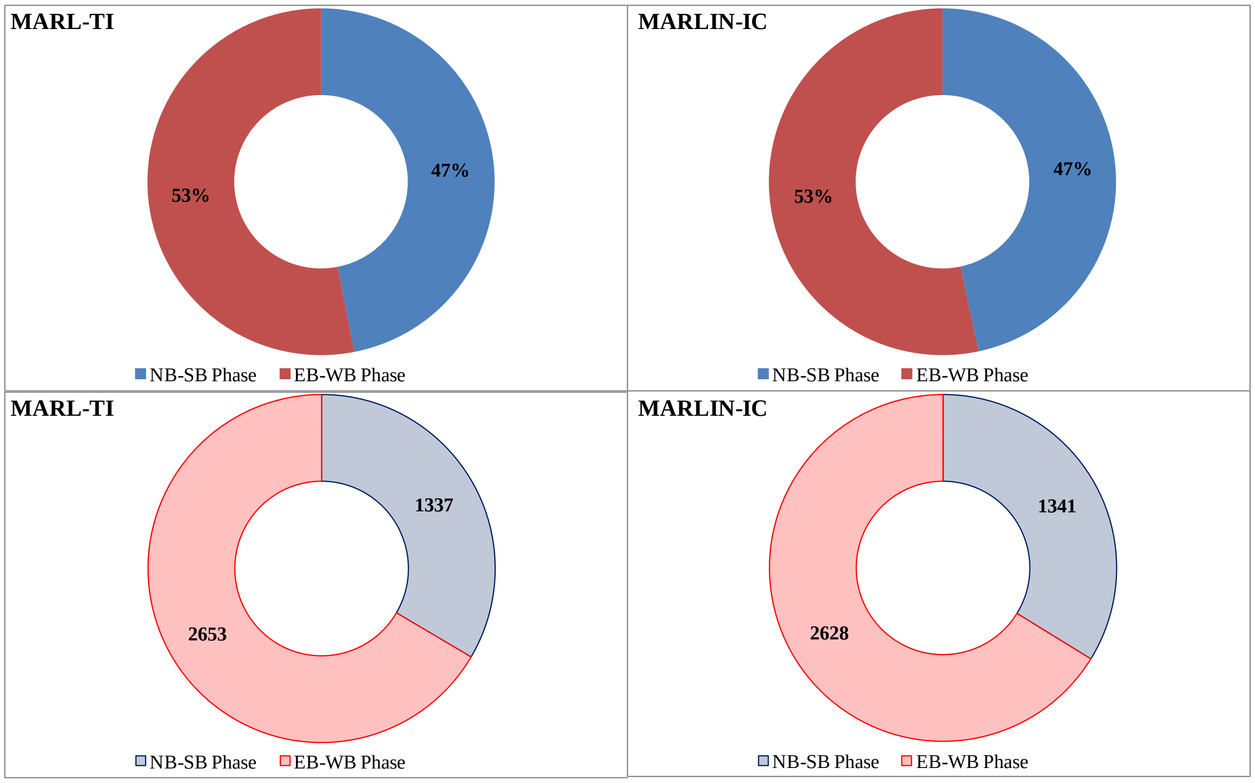


Figure 6-11 Percentage of Green Allocation and Volumes for Intersection I_S

In summary, the results above confirm the hypothesis that, under highly-saturated conditions, coordination is beneficial. Even under normal traffic conditions, MARLIN-based agents achieve the same performance (if not better) than the non-coordinated MARL-based agents, though it learns more slowly, as will be discussed later. Hence MARLIN-IC appears to be substantially more robust as it can perform well in a much broader range of scenarios and traffic conditions.

More specifically, the experiments presented above demonstrate a strong correlation between the number of phases, the distance between the intersections, and the need for coordinated learning. MARLIN methods consistently outperform both independent MARL methods for the intersections that have more phases and intersection that are tightly spaced. Therefore, the above results and the experimental findings should help when deciding under what circumstances/conditions coordinated methods are warranted and how much performance improvements such coordination may achieve. This is particularly important for urban traffic management centres and decision makers to prioritise their resources via identifying the areas/intersections that warrant investment in coordination.

6.6.1.1.4 MARL-Specific Computational Performance Measures

Table 6-11 shows the computation time and convergence speed for MARL and MARLIN algorithms. As expected, MARLIN convergences more slowely than MARL due to the larger state-action space that needs to be explored before convergence. It is found that for MARLIN the higher the demand, the lower the speed of convergence. Although MARLIN-IC and MARLIN-DC have similar average delays, throughputs, and route travel times (Table 6-8, Table 6-9, and Table 6-10) it is shown in Table 6-11 that MARLIN-DC requires more computation time per learning step in addition to a slower convergence speed. Accordingly, it can be concluded that when coordination between agents is warranted, it is plausibly sufficient to use MARLIN-IC since it requires 80% less learning time in each iteration and 18% less number of simulation runs to converge compared to MARLIN-DC while achieving very similar performance.

Table 6-11 Computation Time and Convergence Speed

Demand Level	Learning Method	Learning Step Computation Time (ms)	Convergence Speed (No. of Simulation Runs)
Actual	MARL-TI	3.69	35
	MARL-PI	3.53	77
	MARLIN-IC	4.24	75
	MARLIN-DC	22.34	90
High	MARL-TI	5.03	40
	MARL-PI	5.18	80
	MARLIN-IC	5.41	155
	MARLIN-DC	31.59	250

6.6.1.2 Comparison of MARLIN with Traditional Fixed Time and Actuated Methods

As shown in the previous section, MARLIN-IC outperforms the other MARL-based approaches in traffic-related performance measures and outperforms MARLIN-DC in computational and learning speed-related performance measures. Hence in this section, the performance of MARLIN-IC, as a representative for MARLIN-ATSC, is compared to the benchmark control systems: fixed-time control and actuated control.

6.6.1.2.1 Network Performance Measures

Table 6-12 demonstrates an average of 10 simulation runs (after convergence) for the network-wide results. At both demand levels. MARLIN-ATSC considerably outperforms fixed-time and actuated controls by improving average delay, throughput, stop time, number of stops, average queue length, and link travel times. It is noteworthy that the performance of MARLIN-ATSC is more noticeable in the high demand case compared to the actual demand case, because of the ability to relevantly coordinate actions across the network under highly-saturated conditions. It is also remarkable to note considerable savings in variations in queue length across approaches (average Stds of queue length), which reflect the ability of MARLIN-ATSC to equalise queue lengths across approaches compared to fixed-time and actuated controls. The savings in queue length variations are less marked in the high demand case due to the saturation conditions and the long queues in all approaches for MARLIN-ATSC, fixed-time, and actuated controls.

Table 6-12 Overall Performance Measures

Demand Level	Control System	Average Delay	Throughput	Avg. Link Stop Time	Avg. No. Stops	Avg. Max. Queue	Avg. Std Queue	Avg. Link Travel Time
Actual	Fixed-Time	20.11	9037.2	5.47	1.06	7.79	3.54	30.83
	Actuated	19.44	9119.2	5.30	1.05	7.69	3.00	30.20
	MARLIN-IC	15.81	9148.4	4.88	1.03	6.80	1.82	26.91
	% Improvement Vs. Fixed Time	21.4%	1.2%	10.7%	2.2%	12.6%	48.4%	12.7%
	% Improvement Vs. Actuated	18.7%	0.3%	7.9%	1.3%	11.5%	39.2%	10.9%
High	Fixed-Time	73.56	10937.4	10.11	1.19	12.53	2.01	87.55
	Actuated	63.95	11120.4	9.61	1.16	11.56	2.18	78.41
	MARLIN-IC	38.54	12327.6	6.11	1.14	9.73	1.81	51.98
	% Improvement Vs. Fixed Time	47.6%	12.7%	39.5%	3.6%	22.4%	10.3%	40.6%
	% Improvement Vs. Actuated	39.7%	10.9%	36.4%	1.6%	15.9%	17.0%	33.7%

Table 6-13 shows the average CO2 emissions and fuel consumption per kilometre travelled. As shown in the table, MARLIN-IC reduces CO2 emissions and fuel consumption in both actual and high demand cases. Similar to the overall performance findings above, the savings of MARLIN-IC compared to fixed-time and actuated signal controls are more vivid in the high demand case.

Table 6-13 Emissions Measures for Different Control Systems

Demand Level	Control System	Average CO2 Emissions (gm/km)	Average Fuel Consumption (gm/km)
Actual	Fixed-Time	36.13	13.01
	Actuated	36.51	13.18
	MARLIN-IC	35.98	12.94
	% Improvement Vs. Fixed Time	0.4%	0.5%
	% Improvement Vs. Actuated Time	1.5%	1.8%
High	Fixed-Time	64.88	21.73
	Actuated	62.51	20.98
	MARLIN-IC	50.95	17.41
	% Improvement Vs. Fixed Time	21.5%	19.9%
	% Improvement Vs. Actuated Time	18.5%	17.0%

6.6.1.2.2 Route -Specific Performance Measures

Table 6-14 shows the average and Stds of travel times along the five main routes in the network. Table 6-14 shows that, in general, MARLIN-ATSC outperforms the coordinated fixed-time and actuated controls along the 5 routes in terms of both the average travel time and variation in travel time, especially at the high demand level. For example, in Route 4 MARLIN-ATSC shows a range of improvements in travel time depending on the demand level from 18–76% and 0.5–78% compared to fixed-time and actuated controls, respectively. Similarly, in Route 5 MARLIN shows a range of improvement in travel time reliability depending on the demand level from 19–87% and 1–89% compared to fixed-time and actuated controls, respectively. The effect of MARLIN-ATSC in improving the average and variations in travel time is more apparent in the high demand case, and more specifically for Route 5. These high savings are attributed to: 1) the high demand going through two LT phases in Route 5 from Zone 4 to Zone 1 as explained earlier, and 2) because *progression* is only considered in the fixed-time and actuated cases along the (NB/SB) and (WB/EB) corridors, hence Route 5 has the worst travel time in the benchmark cases.

Table 6-14 Average and Standard Deviation of Route Travel Times for MARLIN, Fixed-Time, and Actuated Control Systems

Demand Level	Traffic Control	Average Travel Time Statistical	Route 1	Route 2	Route 3	Route 4	Route 5
Actual	Fixed-Time	Avg	1.68	2.20	1.61	1.18	1.74
		Std	0.12	0.92	0.23	0.13	0.13
	Actuated	Avg	1.60	2.10	2.04	1.26	1.86
		Std	0.11	0.70	0.77	0.18	0.17
	MARLIN-IC	Avg	1.55	1.71	1.51	1.28	1.83
		Std	0.11	0.31	0.20	0.17	0.16
High	Fixed-Time	Avg	4.85	4.63	4.86	4.63	9.33
		Std	1.34	0.39	0.53	0.29	1.99
	Actuated	Avg	2.78	4.18	4.68	4.69	9.77
		Std	0.73	0.34	0.42	1.12	2.22
	MARLIN-IC	Avg	2.77	3.80	3.23	1.52	2.16
		Std	0.70	0.31	0.42	0.14	0.25

6.6.1.2.3 Intersection-Specific Performance Measures

In order to quantify the local savings at the intersection level, Table 6-15 presents the average delay per vehicle and LOS for each intersection using MARLIN-IC, fixed-time and actuated control systems. It is found that MARLIN-ATSC consistently outperforms the fixed-time and actuated controls for all intersections. The effect of MARLIN-ATSC is more evident for the intersections at the boundaries, for closely spaced intersections, and in the high demand case. This is because of the ability of MARLIN-IC to "meter" traffic by optimising both the green time for each phase and the phasing sequence for each intersection that minimises the frequency of cross-blocking and hence minimises the average delay. This feature is important, particularly in high demand cases that could cause spillback. On the other hand, fixed-time and actuated controls typically provide green time that does not cope with the downstream saturation conditions, and consequently blocks vehicles upstream of the intersection.

Table 6-15 Intersection-Specific Average Delay

Demand Level	Control System	Average Delay - Level of Service									
		IC		IE		IW		IN		IS	
Actual	Fixed-Time	34.40	C	19.39	B	13.50	B	21.62	C	13.62	B
	Actuated	31.99	C	18.36	B	12.80	B	22.76	C	13.59	B
	MARLIN-IC	25.16	C	17.20	B	11.12	B	13.13	B	12.95	B
	% Improvement Vs. Fixed Time	26.9%		11.3%		17.6%		39.2%		4.9%	
	% Improvement Vs. Actuated	21.4%		6.3%		13.1%		42.3%		4.7%	
High	Fixed-Time	92.38	F	90.81	F	34.31	C	45.79	D	49.49	D
	Actuated	89.57	F	86.93	F	30.58	C	41.93	D	45.40	D
	MARLIN-IC	67.98	E	42.66	D	29.04	C	31.92	C	26.76	C
	% Improvement Vs. Fixed Time	26.4%		53.0%		15.4%		30.3%		45.9%	
	% Improvement Vs. Actuated	24.1%		50.9%		5.0%		23.9%		41.1%	

6.7 SUMMARY

This chapter presented the applicability of the two modes of operation of MARLIN-ATSC (i.e. Independent Mode and Integrated Mode) in a prototype network of 5 intersections in downtown Toronto using Paramics. This network featured all the components of a multi-agent control problem while being concise enough to allow for the proper interpretation of the results. To replicate realistic traffic conditions in the prototype implementation, two demand levels were modelled: one represented the actual demand observed, while the other represented a 50% increase in the total demand. The performance of MARLIN-ATSC modes were compared against fixed-time and traffic-responsive actuated controls. The analysis of these experiments led to the following conclusions: 1) MARLIN-ATSC Integrated Mode consistently outperformed MARLIN-ATSC Independent Mode regardless of the demand level, in terms of savings in average intersection delay and average route travel times, 2) the effectiveness of MARLIN was found to be more noticeable in the high demand level case. More specifically, the experiments showed that coordination is more effective under highly-saturated conditions when spillback occurs from one intersection to the upstream intersections. On the other hand, independent agents failed to converge to the optimal policy due to the non-stationary effect of the multi-agent learning problem. Investigating MARLIN-ATSC Independent Modes in the high demand level confirmed the need for considering the states of the neighbouring intersections, as MARL-PI approaches outperformed the totally independent operation (MARL-TI). In oversaturated conditions, MARLIN-ATSC Integrated Mode was found to be successful in “metering” traffic at critical intersections. This metering effect resulted in lower green time allocated for some phases of the upstream intersections. The effect of coordination in MARLIN-IC and MARLIN-DC was more distinct at intersections with more than two phases (e.g. with advanced protected LT phases) compared to the typical two-phase intersections (e.g. North/South, East/West), because agents are optimising both the phasing sequence and the split times for each phase. MARLIN-IC also showed substantial savings in travel time variations when compared to MARL-TI, which implies that the MARLIN-ATSC Integrated Mode can provide more reliable travel times, a performance measure that is of equal (if not more) importance to travel time savings. Indirect coordination mechanism (MARLIN-IC) is recommended over the direct coordination mechanism (MARLIN-DC) as it achieved similar performance (average delay and route travel times) but with faster convergence speed and less computation time. This prototype showed a proof-of-concept of MARLIN-ATSC controlling a network with multiple intersections. The next chapter offers insights into the scalability of the system in a large-scale network by applying the MARLIN-ATSC platform to a network of 59 intersections in downtown Toronto.

7 Experimental Results 3: Large-Scale Application In Lower Downtown Toronto

The utmost challenge is to control and coordinate traffic lights in large-scale two-dimensional urban networks, particularly in congested downtown cores of large cities such as Toronto. This chapter presents the performance of the MARLIN-ATSC framework on the lower downtown Toronto network (see Figure 7-1). The large-scale application demonstrates the essence of the proposed approach and builds on the lessons learned from the prototype implementations discussed in Chapters 5 and 6.

The findings from the single intersection case and the 5 intersection testbed demonstrated that MARLIN-ATSC can considerably reduce the average delay and queues at the individual intersections and at network level. In addition, MARLIN-ATSC showed superior performance when compared to MARL, fixed-time and actuated control methods under different demand and arrival profile scenarios. In this chapter we examine how the system scales up to larger networks. In addition, we offer useful insights that would help municipalities prioritise the roll-out of ATSC gradually in such large networks, and quantitatively pinpoint the intersections or groups of intersections that would benefit the most from MARL and MARLIN treatments. ATSC systems, although very promising, are generally costly to deploy, maintain and to operate (FHWA, 2005b) relative to the more basic fixed-time and actuated control systems. Therefore, a hasty decision to deploy ATSC where it is not warranted can be counterproductive, especially in the present tight economic conditions (NCHRP, 2010).

In North America, the average cost of a current ATSC installation is reported to be $65,000 per intersection. A modest corridor of a handful of intersections can cost over a million dollars in installation, communication and initial operational expenses, not to mention the long-term commitment to the system. On the other hand the benefits, in delay reduction for instance, can be as rewarding as 50~60% or more or as disappointing as 0~5% if the ATSC is deployed in an unwarranted location. Therefore, the careful assessment and quantification of benefits prior to deployment decisions cannot be overstated. Therefore, in this chapter we demonstrate a wide range of measures and evaluation criteria to not only identify the best candidate intersections to receive adaptive control treatment but also to help prioritise the roll-out of adaptive control with coordination/communication (e.g. MARLIN-IC) vs without communication (e.g. MARL-TI). The structure of this chapter follows the roadmap shown in Figure 4-16.

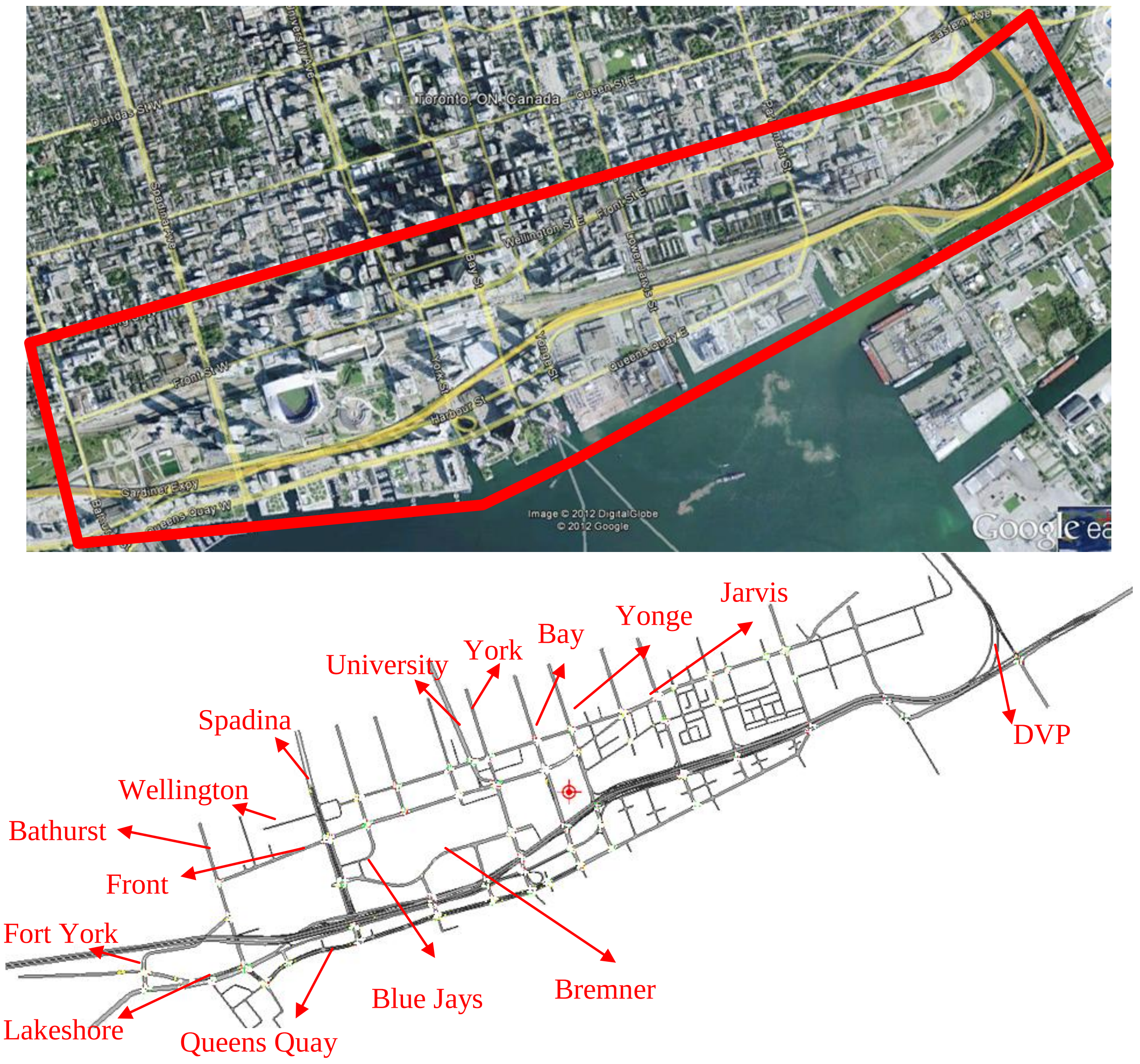


Figure 7-1 Study Area

7.1 Testbed Network Simulation Model and Demand Modelling

The lower downtown of Toronto is the core of the City of Toronto. The lower downtown of Toronto in this study is bounded to the South by the Queens Quay corridor, to the West by Bathurst St., to the East by the Don Valley Parkway (DVP) and to the North by Front St. Toronto is the oldest, densest, most diverse area in the region and its downtown core contains one of the highest concentrations of economic activity in the country.

This chapter demonstrates the large-scale application of MARLIN-ATSC on a simulated replica of the lower downtown core. A base-case simulation model for the lower downtown core was originally developed in the ITS Centre and Testbed at the University of Toronto in 2006. In this application, the model is further refined to reflect the signal timing sheets provided by the City of Toronto[5]. The analysis period considered in this application is the AM peak hour, which has around 25,000 vehicular trips. Similar to the models used in Chapters 5 and 6, the testbed simulation model was built and refined using the Paramics microsimulation (Quadstone Paramics, 2012), as detailed in section 4.2.2.3.1.

The following provides the detailed characteristics of the simulation model and the study area:

- In the simulation model, Traffic Analysis Zones (TAZs) form the origin/destination zones within the study area. TAZs can be either gateway zones at the boundaries of the study area or intermediate zones from where trips originate or are destined to (see Figure 7-2);
- Unlike the centroid-connector approach of generating traffic in macroscopic transportation planning models, in microsimulation models traffic is generated on roadway segments in proportional to the capacity and length of these road segments. Traffic from parking lots, parking garages, residential areas, etc. is defined, especially if these access points attract a large number of trips. Figure 7-2 shows an example for the parking garages/lots zones coded in the simulation model;
- The traffic demand is then assigned to the TAZs according to a pre-defined OD matrix. The demand matrix is dynamically generated according to a predefined *profile* in which the first half-hour of the simulation model is used as a warm-up period followed by the full AM peak hour traffic. As expected, the traffic demand pattern in this network exhibits a large number of trips destined to the downtown core from the West (32%) and East boundaries (23%) of the study area. Also traffic demand generated from the North boundary of the study area forms a combination of trips destined to and exiting the downtown core (28%), with the remaining demand forming the intra-city trips (17%). Figure 7-3 shows the demand pattern described above by the display of arrows/lines from the originating zone to all destination zones in the area;

[5] *Contact person is Rajnath Bissessar, City of Toronto – Transportation Services, Manager of the UTCS*

- The traffic demand is then assigned to the TAZs and traffic navigates through the network according to a dynamic stochastic traffic assignment method employed in Paramics as discussed in section 4.2.2.3.1. The most important parameters that govern the driver behaviour/response in the traffic assignment in Paramics are: feedback interval, perturbation, and driver familiarity;
- In this network, the Gardiner Expressway is a major corridor running across the East/West ends of the study area. The Gardiner Expressway carries a large number of trips that are destined to the downtown core through the Spadina St., York St./Yonge St., and Jarivs St. off-ramps. These locations show congestion accumulation to a level that may block the off-ramp of the expressway; an example is the Spadina St. off-ramp as shown in Figure 7-4. This demand pattern creates a constant traffic flow being fed to the downstream intersections at the off-ramp locations, resulting in queue spillbacks and queue propagation as shown in Figure 7-4;
- The simulation model covers a relatively small area, 3.2 km^2 area in the lower downtown core, however it contains around 59 signalised intersections. The simulation characteristics are presented in Table 7-1.

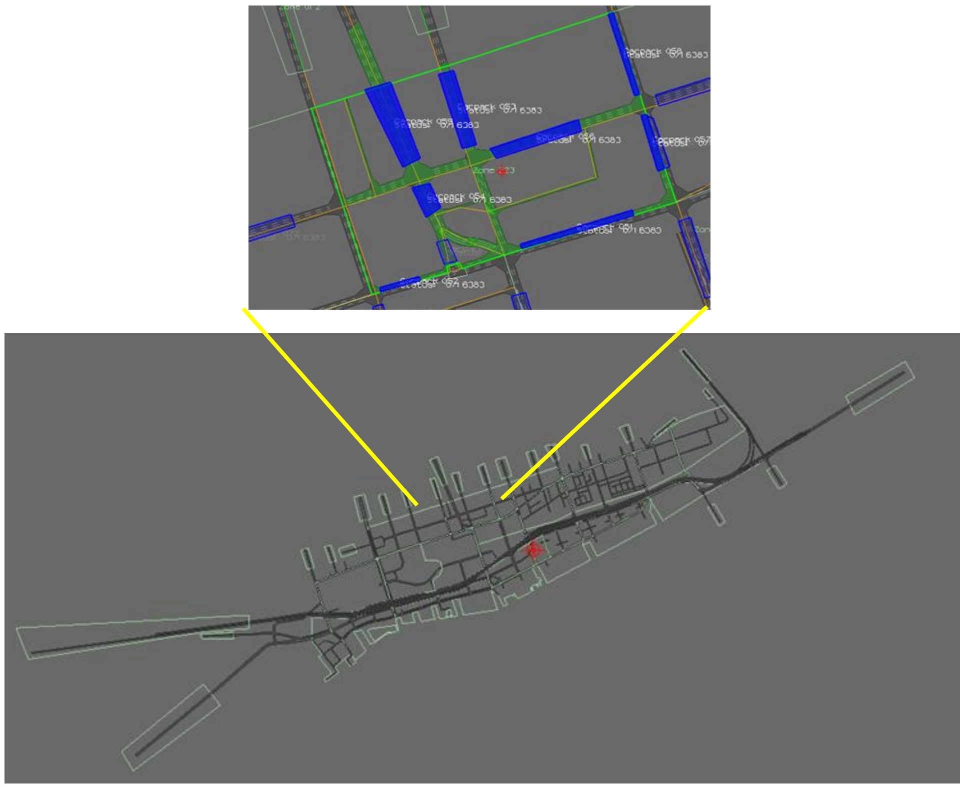
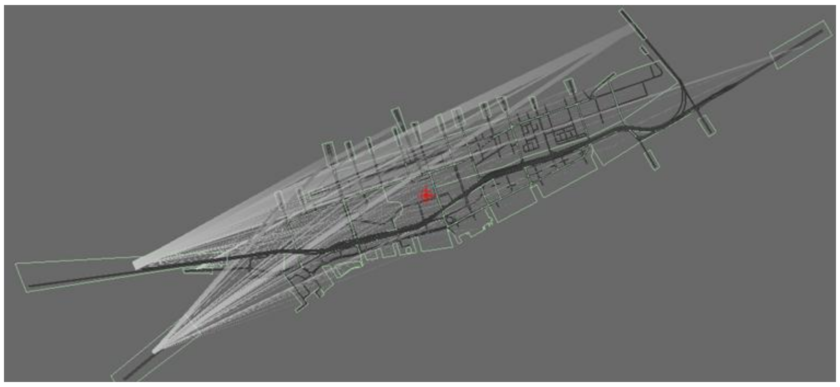
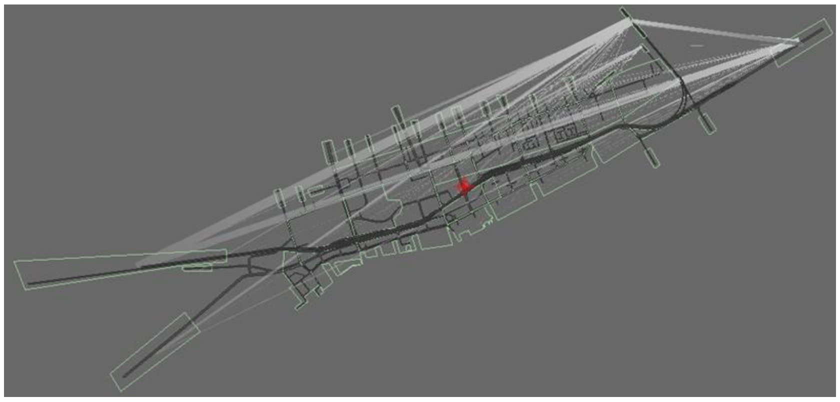
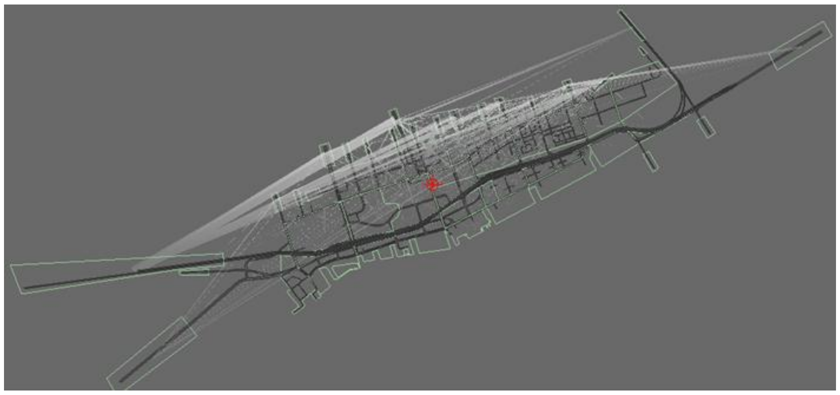


Figure 7-2 Study Area and Traffic Analysis Zones (TAZ)


(a) Demand lines from the West End (7582 trips → 32%)


(b) Demand lines from the East End (5510 trips → 23%)


(c) Demand lines from the North End (6640 trips → 28%)

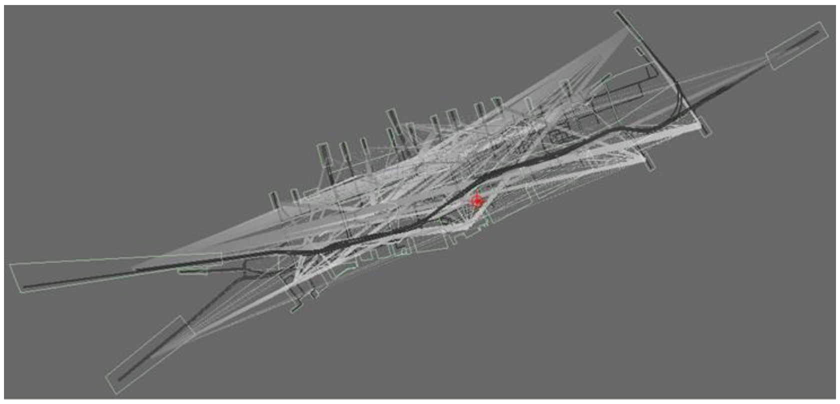

(d) Demand lines from Intra-City Trips (3929 trips → 17%)

Figure 7-3 Demand Pattern in the Study Area

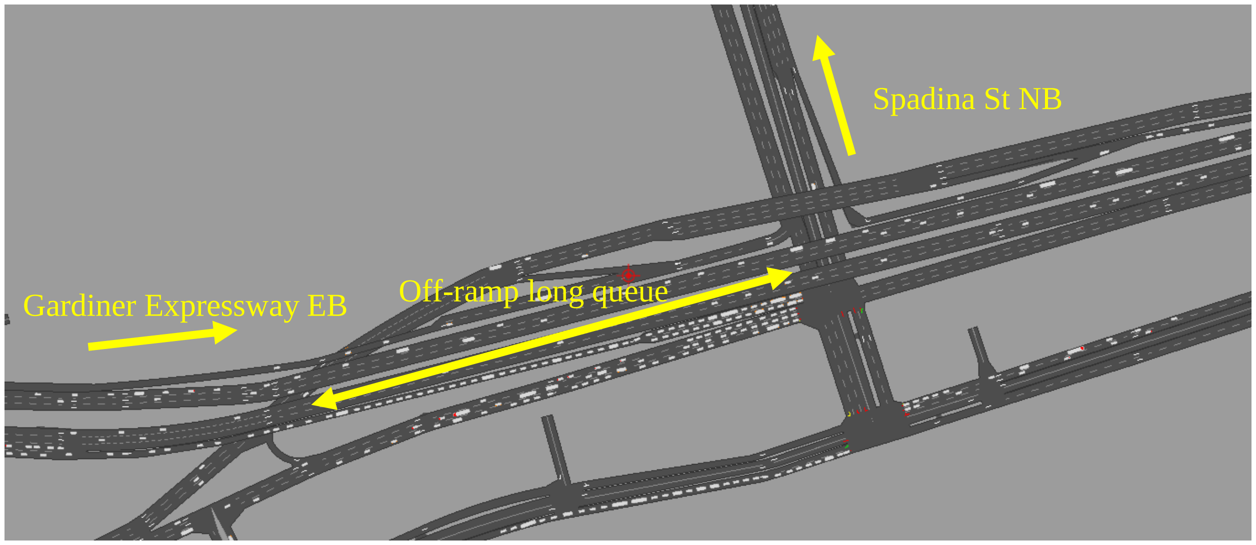


Figure 7-4 Queue Spillback on Spadina Off-Ramp

Table 7-1 Simulation Model Characteristics

Area Covered (km^2)	0.70 km X 4.5 km = 3.2
Perimeter (km)	11,846
No. of Nodes	561
No. of Links	1100
No. of Zones	53
Length of Roads (km)	293
No of Signalised Intersections	59

In this application, two demand arrival profiles are investigated: uniform profile and variable profile, as shown in Figure 7-5.

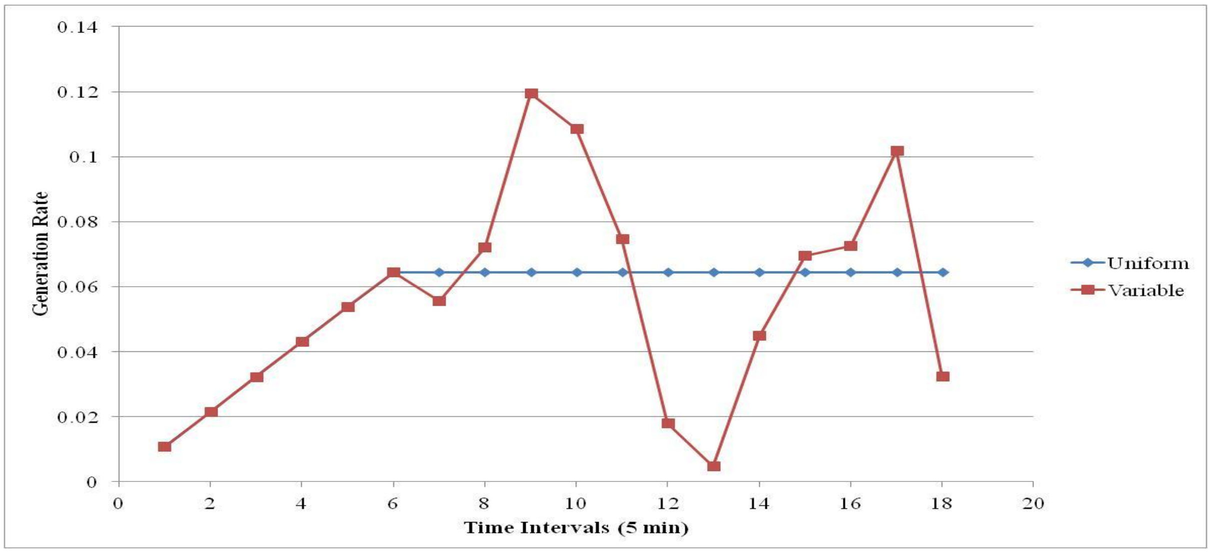


Figure 7-5 Demand Profiles

7.2 EXPERIMENTAL DESIGN

7.2.1 Benchmarks

Two scenarios are investigated: arterial control (along Lakeshore Boulevard), and network control as shown in the experimental design in Table 7-2. In each of the above scenarios, the Base Case (BC) scenario is tested in which traffic signals, as defined and operated by the City of Toronto, show a mix of different traffic control systems including: SCOOT (21 signals), Main Traffic Signal System (MTSS, 24 signals), Arterial Master Signal System (AMSS,7 signals), and TransCore Signal System (TransSuite,7 signals), as shown in Figure 7-6.

7.2.1.1 Control Systems

MTSS: This system is based on a main computer located at the Traffic Management Centre (TMC) that controls the individual operation of each signalised intersection. MTSS provides different timing plans for different times of day and allows the transportation staff to monitor its operation.

TransSuite: This system provides a more responsive control of signalised intersections by including extendable LT phases at specific intersections. TransSuite also provides improved monitoring of signalised intersections from a central location and seamless operation during communication loss between the TMC and the signalised intersection. However there is no major difference in the control logic between MTSS and TransSuite.

AMSS: This system only runs a small proportion of traffic signals on a specifically required system, such as the 7 signals serving the Harbourfront LRT as shown in Figure 7-6. The major difference between AMSS and previous systems is that this system is acyclic (no fixed cycle length, and hence no offset).

SCOOT: This system uses a central computer to adapt the traffic signal timings to detected traffic volumes to minimise an overall system (or corridor) delay as explained in section 2.3.2.2. As shown in Figure 7-6, SCOOT is implemented on the intersections along two major roads, Lakeshore St. and Spadina St., where substantial variations in traffic flows are anticipated. It is worth noting that due to the limited technical details about the operation of SCOOT, it is approximated in this thesis as an enhanced fully-actuated control in which loop detectors are placed on all approaches and the extension times are conducted second-by-second.

7.2.1.2 Modes of Control

Each of the systems above can operate under one of the following modes of control as indicated in the timing sheets provided by the City of Toronto (Figure 7-6):

Fixed-Time Control (FXT) in which the signal timings for all phases, and the offset are fixed values and assigned to the phases regardless of the demand, with the exception of the SCOOT-FXT control mode where the green for the phase is started with minimum green and extended as long as there is demand for the corresponding phase up to a maximum green.

Semi-Actuated with Presence loops (SAP) in which one or more phases are only activated and assigned the corresponding fixed timings if there is demand for these phases.

Semi-Actuated 2 - Vehicle Minimum Green (SA2) in which one or more phases are only activated and assigned minimum green if there is demand for these phases. The green time can be extended as long as there is a demand for the corresponding phase.

In Paramics, the signal timings are hardcoded using the signal timings provided by the City of Toronto, while the aforementioned control systems/modes of control are modelled as illustrated in Figure 7-7.

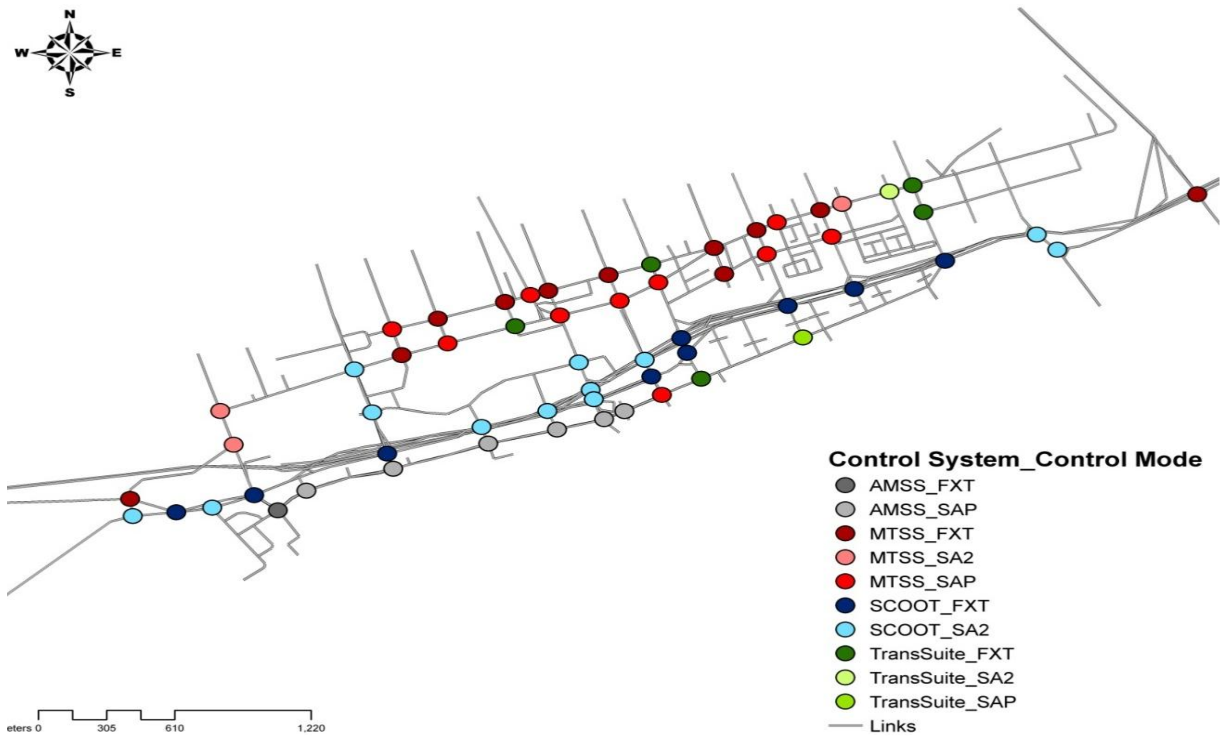


Figure 7-6 Signal Control Systems Currently Implemented

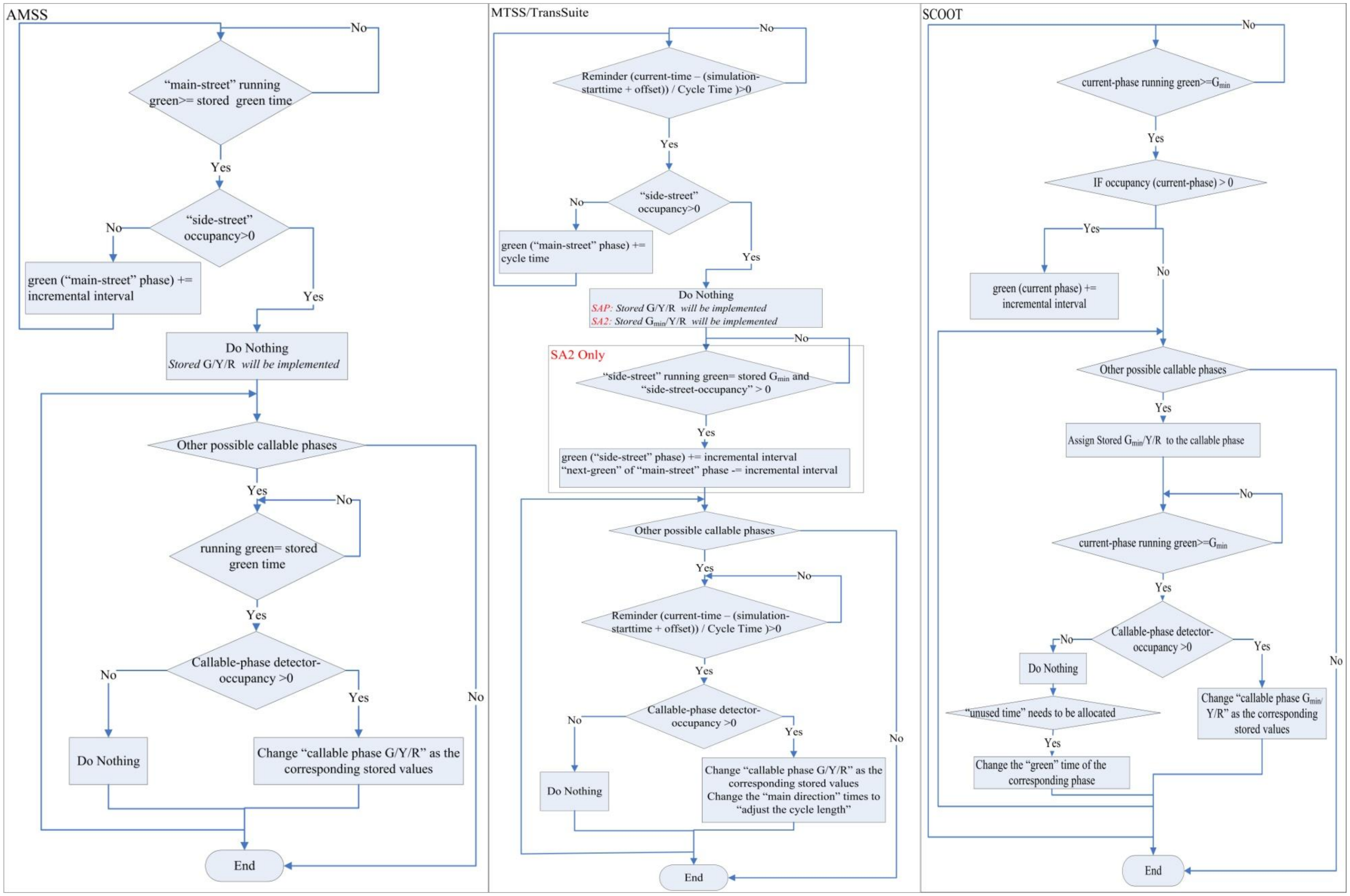

AMSS
"main-street" running green>= stored green time
No
Yes
"side-street" occupancy>0
No
green ("main-street" phase) += incremental interval
Yes
Do Nothing
Stored G/Y/R will be implemented
Other possible callable phases
Yes
running green= stored green time
Callable-phase detector-occupancy >0
Do Nothing
Change "callable phase G/Y/R" as the corresponding stored values
End
MTSS/TransSuite
Reminder (current-time – (simulation-starttime + offset)) / Cycle Time)>0
"side-street" occupancy>0
green ("main-street" phase) += cycle time
Do Nothing
SAP: Stored G/Y/R will be implemented
SA2: Stored Gmin/Y/R will be implemented
SA2 Only
"side-street" running green= stored Gmin and "side-street-occupancy" > 0
green ("side-street" phase) += incremental interval
"next-green" of "main-street" phase -= incremental interval
Other possible callable phases
Reminder (current-time – (simulation-starttime + offset)) / Cycle Time)>0
Callable-phase detector-occupancy >0
Do Nothing
Change "callable phase G/Y/R" as the corresponding stored values
Change the "main direction" times to "adjust the cycle length"
End
SCOOT
current-phase running green>=Gmin
IF occupancy (current-phase) > 0
green (current phase) += incremental interval
Other possible callable phases
Assign Stored Gmin/Y/R to the callable phase
current-phase running green>=Gmin
Callable-phase detector-occupancy >0
Do Nothing
"unused time" needs to be allocated
Change the "green" time of the corresponding phase
Change "callable phase Gmin/Y/R" as the corresponding stored values
End

Figure 7-7 Conceptual Description of Currently Implemented Signal Control Systems

7.2.2 Experimental Setup

The MARLIN-ATSC large-scale application is conducted to examine two types of coordination: 1) arterial (corridor)-specific coordination, and 2) network-wide coordination. MARLIN-IC is proven to be the most efficient system among those that require communication with neighbouring agents, while MARL-TI does not require any communication with neighbouring agents. Therefore an investigation of the benefits of both methods is conducted in this chapter.

- Arterial Control

Building on the lessons learned from previous chapters, in the arterial control case MARL-TI and MARLIN-IC are tested on the Lakeshore Arterial, an arterial where considerable variations in traffic are anticipated because it runs parallel to the Gardner Expressway. MARLIN-IC is used to test two cases: 1) arterial-specific coordination with agents coordinated only with their neighbours on the specific arterial (called MARLIN-IC-A), and 2) network-wide agents coordinated with all neighbours (called MARLIN-IC (Network Control)).

- Network Control

In the Network Control case, MARL-TI, and MARLIN-IC are applied to the 59 intersections in the network. Two scenarios are examined to study the effect of the following variables:

- Demand Profile: uniform and variable;
- Drivers Familiarity: high familiarity percentage, and low familiarity percentage.

The familiarity cases are meant to investigate the effect of Traveller Information System (TIS) provision and/or the drivers' reaction and rerouting in response to congestion levels (related to the performance of intersection control) and how this behaviour may impact on network performance. Low familiarity percentage is an indication of drivers who are not receiving updates (or aware) of the traffic characteristics and congestion build up in the network; therefore they are less likely to change route even if the travel time to destination along that route increases. High familiarity percentage on the other hand is indicative of drivers' awareness of traffic conditions and alternative routes in the case where their preferred route (e.g. shortest in time) becomes congested. It is expected that with a low driver familiarity percentage, drivers will be less aware of other routing options and therefore congestion can quickly spear in the network.

This should form a challenging test case to compare the performance of the MARLIN-ATSC integrated mode (MARLIN-IC) against the independent mode using MARL-TI.

Table 7-2 Experimental Design

Experiment No.	**Control System**	**Test Area**		**Demand Profile**		**Familiarity**	
		Arterial	**Network**	**Uniform**	**Variable**	**60%**	**30%**
1	Base Case	√		√		√	
2			√	√		√	
3			√	√			√
4			√		√	√	
5	MARL-TI	√		√		√	
6			√	√		√	
7			√	√			√
8			√		√	√	
9	MARLIN-IC	√		√		√	
10			√	√		√	
11			√	√			√
12			√		√	√	

7.3 Results and Analysis

7.3.1 Arterial Control Experiments

In these experiments, Lake Shore Boulevard (LS Blvd) is selected as the testbed arterial. LS is a major East/West arterial in Toronto's waterfront. It is an alternative route for the Gardiner Expressway, and therefore it is a candidate route for many trips destined to downtown Toronto. Consequently it is one of the busiest and key corridors in the network, where substantial variations in traffic flows are anticipated. The modelled section of the arterial is around 7 km long and contains 20 intersections, as shown in Table 7-3. In the field, the intersections along LS are operated using the SCOOT system as shown in Figure 7-6. However due the proprietary nature of SCOOT, the control logic of SCOOT is approximated as enhanced fully-actuated, as discussed in section 7.2.1.

Table 7-3 Intersections of the Arterial Control Experiments

Intersection Name
Cherry_St_N and Lake_Shore_Blvd
Cherry_St_S and Lake_Shore_Blvd
Lake_Shore_Blvd and Parliament_St_&_Queens_Quay
Lake_Shore_Blvd and Sherbourne_St
Lake_Shore_Blvd and Lower_Jarvis_St
Lake_Shore_Blvd_N_Tcs and Yonge_St
Harbour_St and Lake_Shore_Blvd_&_Yonge_St
Lakeshore_Blvd_WB and Bay_St
Lakeshore_Blvd_EB and Bay_St
Lakeshore_Blvd_WB and York_St
Lakeshore_Blvd and Simcoe
Lakeshore_Blvd and Rees
Lakeshore_Blvd_EB and York_St
Don_Rdwy and Lake_Shore_Blvd
Lakeshore_Blvd and Spadina
Lakeshore_Blvd and Bathurst_St
Lakeshore_Blvd and Stadium
Lakeshore_Blvd and Fleet
Lakeshore_Blvd and Fort_York_Blvd
Queens_Quay and Yonge_St

The experiments are conducted using the following systems:

- BC control as currently in the field (but with SCOOT approximation)
- MARL-TI system
- Two implementations for the MARLIN-IC system:
    - MARLIN-IC along arterial neighbours only (MARLIN-IC-A)
    - MARLIN-IC with two-dimensional network neighbours (MARLIN-IC).

It is worth noting that MARL-TI and MARLIN-IC-A are only applied to the signalised intersections along the arterial, while the remaining intersections in the network are operated with the same control as the BC. It is also important to note that MARLIN-IC is applied over the entire network and will be explained in more detail in the next section; but it is important to investigate it here to study the effect of two-dimensional (network) coordination vs one-dimensional (arterial) coordination.

The following performance measures are extracted and analysed for the LS arterial:

- Average number of vehicles served by the arterial in the simulation hour
- Average stop time experienced per vehicle
- Average and Std of travel time
- Major and minor street average delay per intersection.

Table 7-4 summarises the overall performance for the arterial control using different control systems including the BC system. Figure 7-8 shows the average delay per intersection for major and minor streets. Figure 7-9 and Figure 7-10 show the variations in travel time for lakeshore EB during the AM peak hour.

Overall, it is found that the LS EB is more congested and exhibits longer travel times and higher travel time variability when compared to LS WB. This is not surprising as EB traffic destined to downtown Toronto is higher during the morning rush hour. For LS EB, MARLIN-IC generally outperforms the BC by around 13%, 40%, 28%, 63% in the total number of vehicles served, average stop time, average travel time, and Std of travel time, respectively. In addition, MARLIN-IC-A outperforms BC but not to the level achieved by MARLIN-IC, which shows the importance and effect of network-wide (i.e. two-dimensional) coordination. MARL-TI on the other hand still shows better performance compared to the BC but not to the level of any MARLIN system.

It is shown in Figure 7-8 that MARL-TI, MARLIN-IC-A and MARLIN-IC outperform the BC by around 10%, 29%, 55% in terms of average delay/intersections along the major street approaches (i.e. EB and WB) and by around 49%, 63%, 69% in terms of average delay/intersections along the minor street approaches (i.e. NB and SB). This indicates that the system not only minimises the delay along major street approaches, but also along those of the minor streets. The disaggregate average delay for selected intersections are shown in Appendix IV.

Table 7-4 MOE for Arterial Control on Lakeshore EB and WB

System \ MOE	Served Vehicles (veh)	Avg. Stop Time (min)	Avg. Travel Time (min)	Std. Travel Time (min)
BC	4750.61	10.49	16.81	3.70
MARL-TI	4766.29	10.26	15.51	2.92
MARLIN-IC-A	4936.33	9.24	14.39	2.89
MARLIN-IC	5351.29	6.33	12.10	1.37
% Improvments MARL-TI Vs. BC	0.3%	2.2%	7.7%	21.1%
% Improvments MARLIN-IC-A Vs. BC	3.9%	11.9%	14.4%	21.8%
% Improvments MARLIN-IC Vs. BC	12.6%	39.7%	28.0%	63.1%
BC	2893.13	5.10	10.81	1.13
MARL-TI	3253.94	4.25	9.52	1.00
MARLIN-IC-A	3386.37	4.50	9.05	0.76
MARLIN-IC	3530.01	3.27	8.46	0.50
% Improvments MARL-TI Vs. BC	12.5%	16.8%	11.9%	11.1%
% Improvments MARLIN-IC-A Vs. BC	17.0%	11.8%	16.3%	32.5%
% Improvments MARLIN-IC-N Vs. BC	22.0%	35.9%	21.7%	55.4%

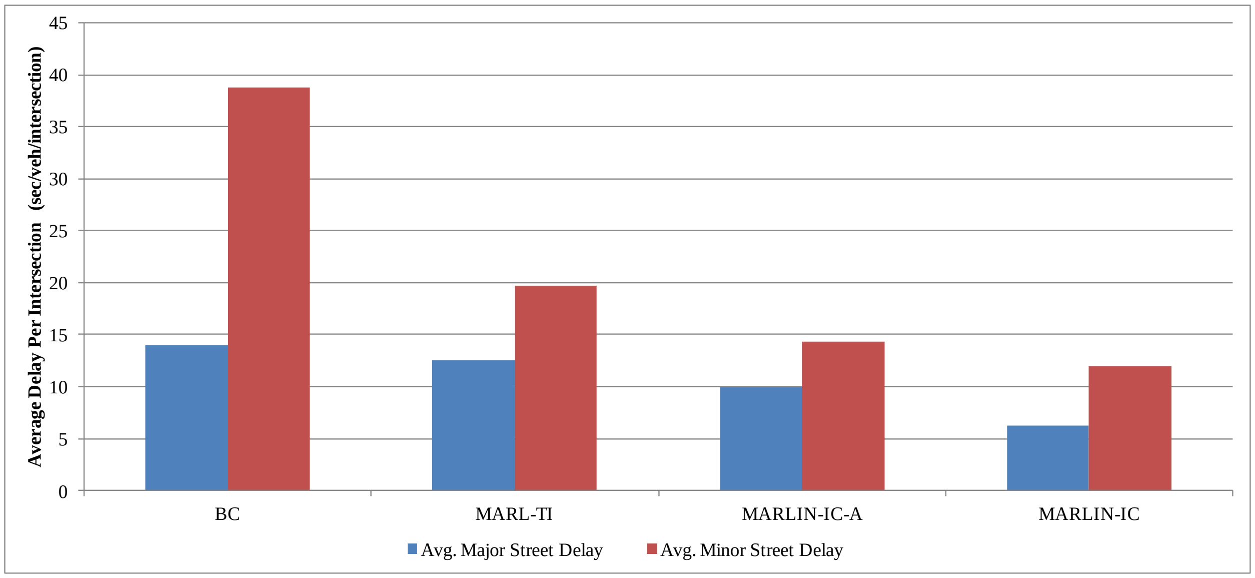


Figure 7-8 Comparison of Major and Minor Street Average Delay Per Intersection

To illustrate the route travel time variability in the morning rush hour, Figure 7-9 and Figure 7-10 have been produced. MARLIN-IC outperforms the other systems with respect to travel time variability, which is expected because in the MARLIN-IC all the signalised intersections in the network are operating using MARLIN. MARLIN-IC-A on the other hand still exhibits less travel time variability compared to the BC, but not to the level achieved by MARLIN-IC. Overall MARL and MARLIN systems exhibit less travel time variability, which is indicative of travel time reliability. Reliable travel times reflects a robust system and less cost due to congestion.

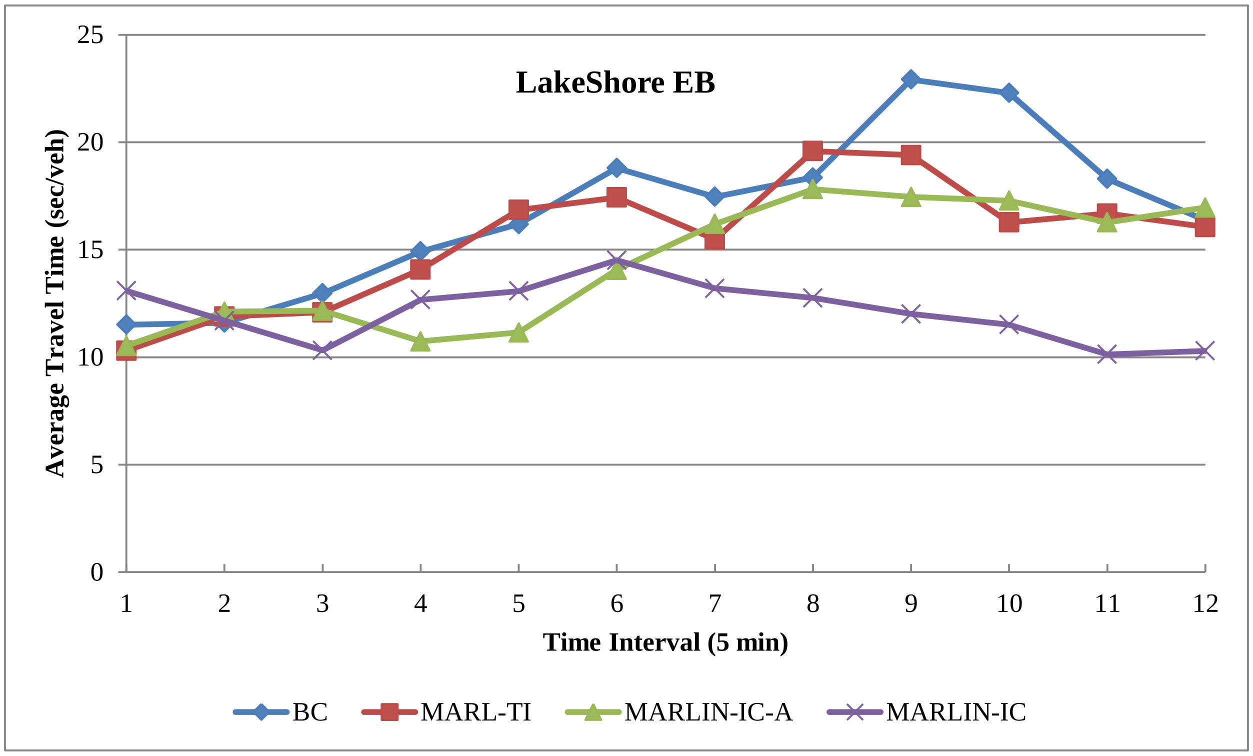


Figure 7-9 Lake Shore EB Travel Time Variations for: Base Case, MARL-TI, MARLIN

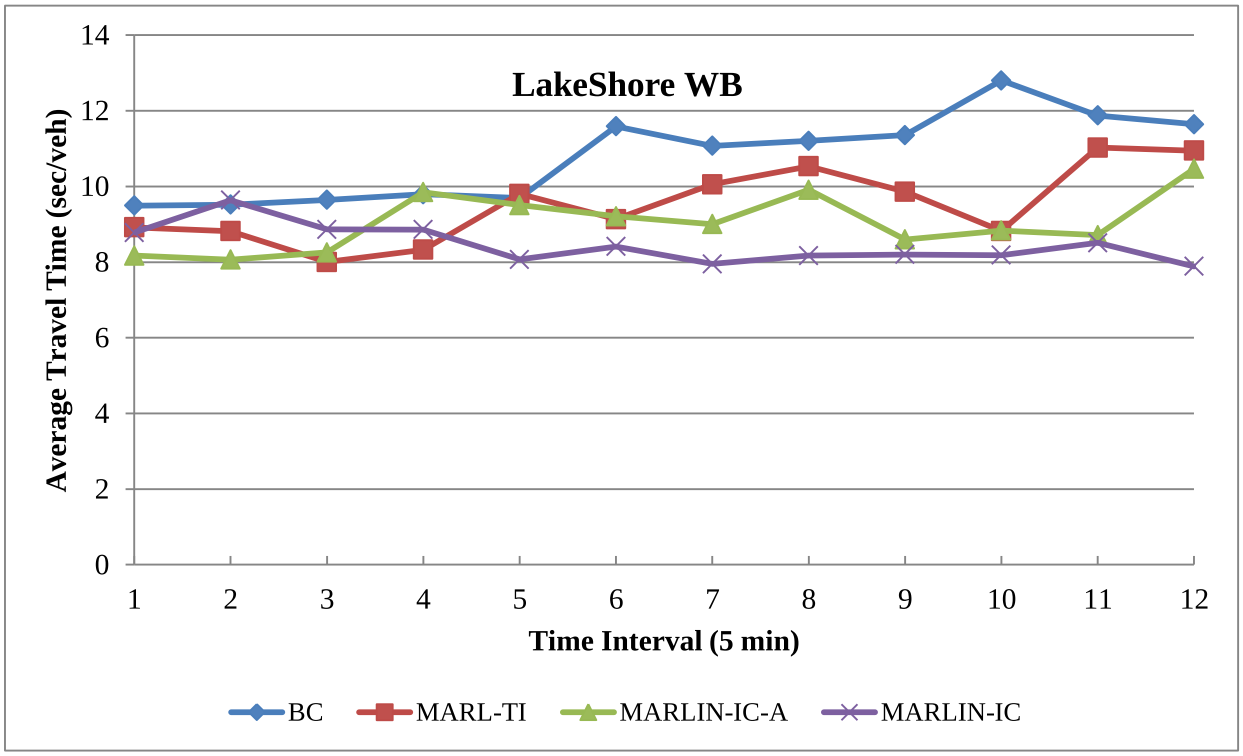


Figure 7-10 Lake Shore WB Travel Time Variations for: Base Case, MARL-TI, MARLIN

7.3.2 Network Control Experiments

In the network control experiments three scenarios are investigated; uniform demand profile, variable demand profile, and unfamiliar drivers.

- Normal: in this scenario uniform demand profile is used. The percentage of familiar drivers is 60% and the feedback interval is 2 min;

- Variable: in this scenario variable demand profile is used (as shown in Figure 7-5) to replicate any expected highly variable traffic conditions;

- Unfamiliar Drivers: in this scenario 30% familiar drivers and a 4 min feedback interval are considered to investigate the case when drivers are less aware of other routing options.

In each of the three above scenarios, results are reported for BC control systems (existing conditions), MARL-TI (represents MARLIN-ATSC Independent Mode with no communication between agents), and MARLIN-IC (represents MARLIN-ATSC Integrated Mode with coordination between agents).

The performance of each control system is evaluated based on the following MOEs:

- Average Delay Per Vehicle (sec/veh)
- Average Max. Queue Length Per Intersection (veh)
- Average Standard Deviation of Queue Lengths Across Approaches (veh)
- Number of Completed Trips
- Average CO2 Emissions Factors (gm/km)
- Average Travel Time for Selected Routes (min).

7.3.2.1 Normal Scenario

Table 7-5 compares the performance of the BC against the MARLIN-ATSC system with and without communication among agents, i.e. MARLIN-IC and MARL-TI respectively.

Table 7-5 Network-Wide MOE in the Normal Scenario

System / MOE	BC	MARL-TI	MARLIN-IC	% Improvments MARL-TI Vs. BC	% Improvments MARLIN-IC Vs. BC	% Improvments MARLIN-IC Vs. MARL-TI
Average Intersection Delay (sec/veh)	35.27	25.72	22.02	27.06%	37.57%	14.41%
Throughput (veh)	23084	23732	24482	2.81%	6.06%	3.16%
Avg Queue Length (veh)	8.66	6.60	5.88	23.77%	32.07%	10.88%
Std. Avg. Queue Length (veh)	2.12	1.62	1.47	23.37%	30.74%	9.61%
Avg. Link Delay (sec)	9.45	8.50	5.04	10.07%	46.73%	40.76%
Avg. Link Stop Time (sec)	2.74	2.57	2.02	5.95%	26.06%	21.38%
Avg. Link Travel Time (sec)	16.81	15.81	12.32	5.97%	26.70%	22.05%
CO2 Emission Factor (gm/km)	587.28	421.34	412.21	28.26%	29.81%	2.17%

The analysis of the results shown in Table 7-5 leads to the following findings:

- The two MARLIN-ATSC algorithms result in lower average delay, higher throughput, and shorter queue length and stop time compared to those from the BC. The most notable improvements are the average delay (38% MARLIN vs BC), Std of average queue length (31% MARLIN vs BC), CO2 emission factors (30% MARLIN vs BC). Appendix 3 presents a simple economic benefit analysis of using MARLIN vs BC;
- These substantial improvements are not only due to the intelligence of the RL algorithm, but also as a result of the coordination mechanism between the agents to reach a network-wide set of actions that minimise the long-term delay. This coordination results in the so-called "metering" effect from the upstream intersection to the downstream intersection while accounting for the queues and delays at the downstream intersection. In fact, the tangible savings in the Std in the queue length are interesting because this means there are balanced queues among all intersection approaches;
- MARL-TI outperforms the BC in all the MOEs, most notable are the average intersection delay (27%) and the C02 emission factor (28%). However, comparing MARLIN-IC to MARL-TI it is found that the latter experiences relatively higher delays because in MARL-TI the actions are only based on locally collected data, and thereby results in more vehicles being retained in the network at the end of the simulation (6% throughput improvement in MARLIN vs 2.8% throughput improvement in MARL-TI).

To further understand which intersections contribute the most to the above noted savings the spatial distribution of delay of the BC – normal scenario – is plotted in Figure 7-11. It is interesting to note that some intersections encounter delay in the range of 0–10 sec/veh while others encounter 70–110 sec/veh in the BC scenario.

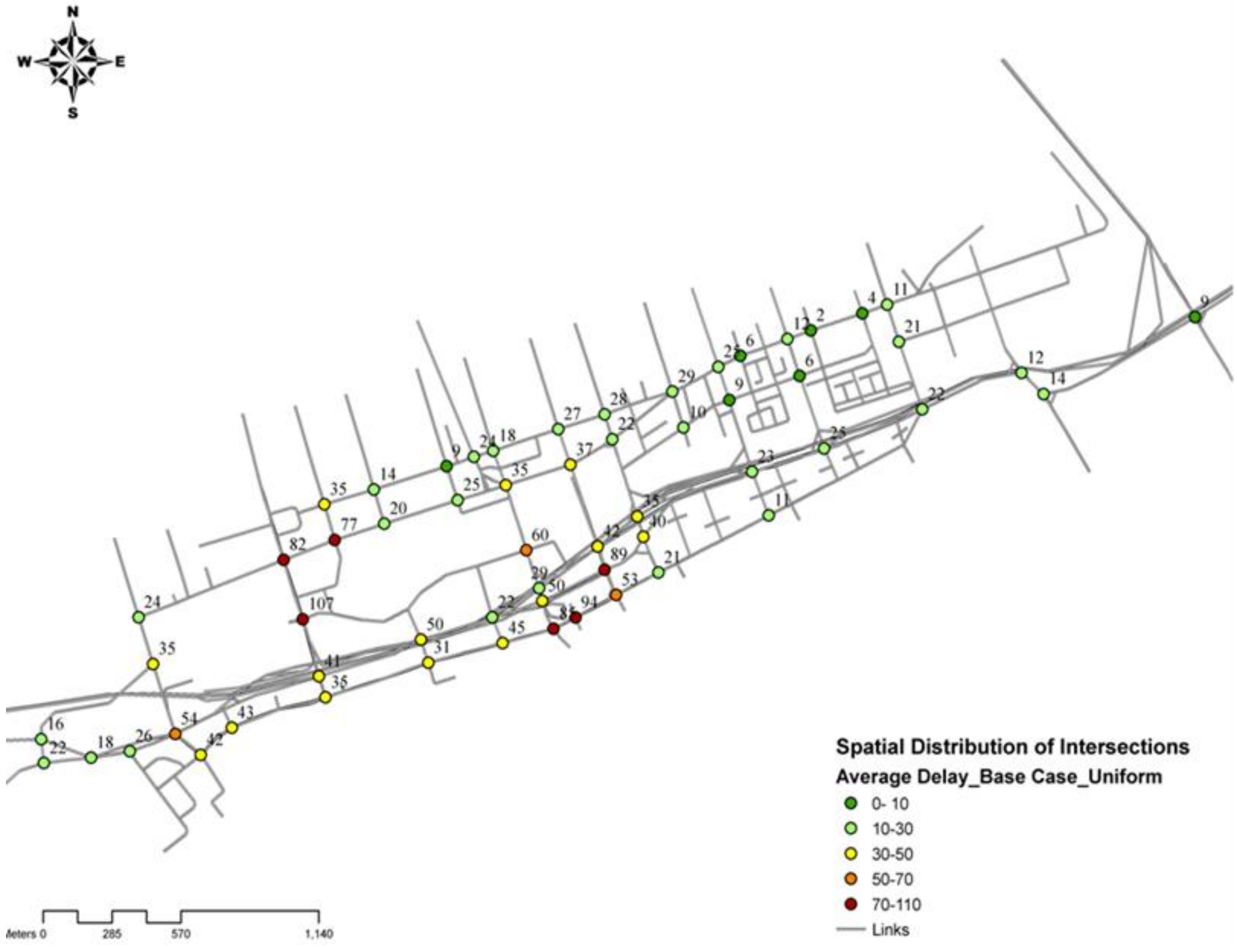

N
W
E
S
Spatial Distribution of Intersections
Average Delay_Base Case_Uniform
0- 10
10-30
30-50
50-70
70-110
Links
Meters 0
285
570
1,140

Figure 7-11 Spatial Distribution of Average Delay for the Base Case Normal Scenario

Table 7-5 shows a very promising overall performance of MARLIN. However, as shown in the wide range of average delays among intersections, the improvements at some intersections are much higher than the network averages. Therefore the spatial distribution of percentage improvement and histogram of these improvements are presented in Figure 7-12 and Figure 7-13. The analysis of Figure 7-12 and Figure 7-13 leads to the following conclusions:

- In the MARL-TI case, most of the savings occur, interestingly, at the chronically busy intersections such as Lake Shore & Yonge Sts., Lake Shore & York Sts., Bathurst & Lake Shore Sts., and University & Wellington Sts.;
- Similar to MARL-TI, in MARLIN-IC most of the savings are found at the busiest downtown intersections, but what is interesting to note is the formation of a “corridor of savings” in the MARLIN-IC case as shown in the dotted shaded areas in Figure 7-12. To illustrate the

intersections along York St. (from Queens Quay St. to Bremner St.) exhibit substantial savings, which are not observed in the MARL-TI case. This is primarily due to coordination among agents. This observation is extremely important as this is the only methodical way to determine which intersections warrant MARLIN-ATSC with communication between agents as opposed to MARLIN-ATSC without communication. It is clear from this example that these intersections are perfect examples for such case. A similar coordination is warranted along Yonge St. and to a lesser extent along Spadina Ave. and Jarvis St.;

- It is even more interesting to observe the formation of an “area control” effect in some chronic areas in the downtown core. The off-ramps at Spadina St., Yonge St., and York St. form major access points to the City’s core, therefore if traffic is heavily congested at these locations (and their downstream intersections) the entire network could easily deteriorate into a grid-lock condition. The MARLIN-IC algorithm is found to be intelligent enough to capture such a pattern while controlling these “critical” intersections, hence the major savings compared to the BC. As a result, the noted “Area 1” in Figure 7-12 warrants coordination among all the intersections within this area. Similarly, “Area 2” is known to be chronically congested in the morning peak due to the dense business and economic activities near Front St., Wellington St., York St., and Simco St. In addition, in the West end of the network, LS and Fort York St. are two alternative routes for a high number of trips destined to Bathurst St., which is also observed to be congested in the morning peak. This area (“Area 3”) exhibits considerable savings that warrant “area control” using MARLIN-IC;
- It is worth noting that there are few intersections that exhibit substantial savings (81%), such as the case of Front St. and Princess St.; however this saving is not that important because the BC delay is observed to be minimal (2 sec/veh);
- In Figure 7-12, the histogram of percentage savings for different control systems is shown. It is found that the majority of savings (40–60%) achieved by MARLIN-IC over BC are within the defined “area of control” intersection. Comparing MARL-TI to MARLIN-IC, it is interesting to find that the majority of savings (10–20% and 20–30%) are actually for the same “area of control” noted above; which confirms that the additional “average” 15% outperformance (in average delay) of MARLIN-IC over MARL-TI – reported in Table 7-5 – is mostly concentrated in and attributed to this area of the network;

- In Figure 7-13, the spatial distribution of the percentage savings of MARLIN-IC over MARL-TI is shown. It is found that the majority of savings (40–60%) achieved by the coordination are on the intersections of University & Front Sts., Queens Quay & Bay Sts., and Queens Quay & Jarvis Sts.

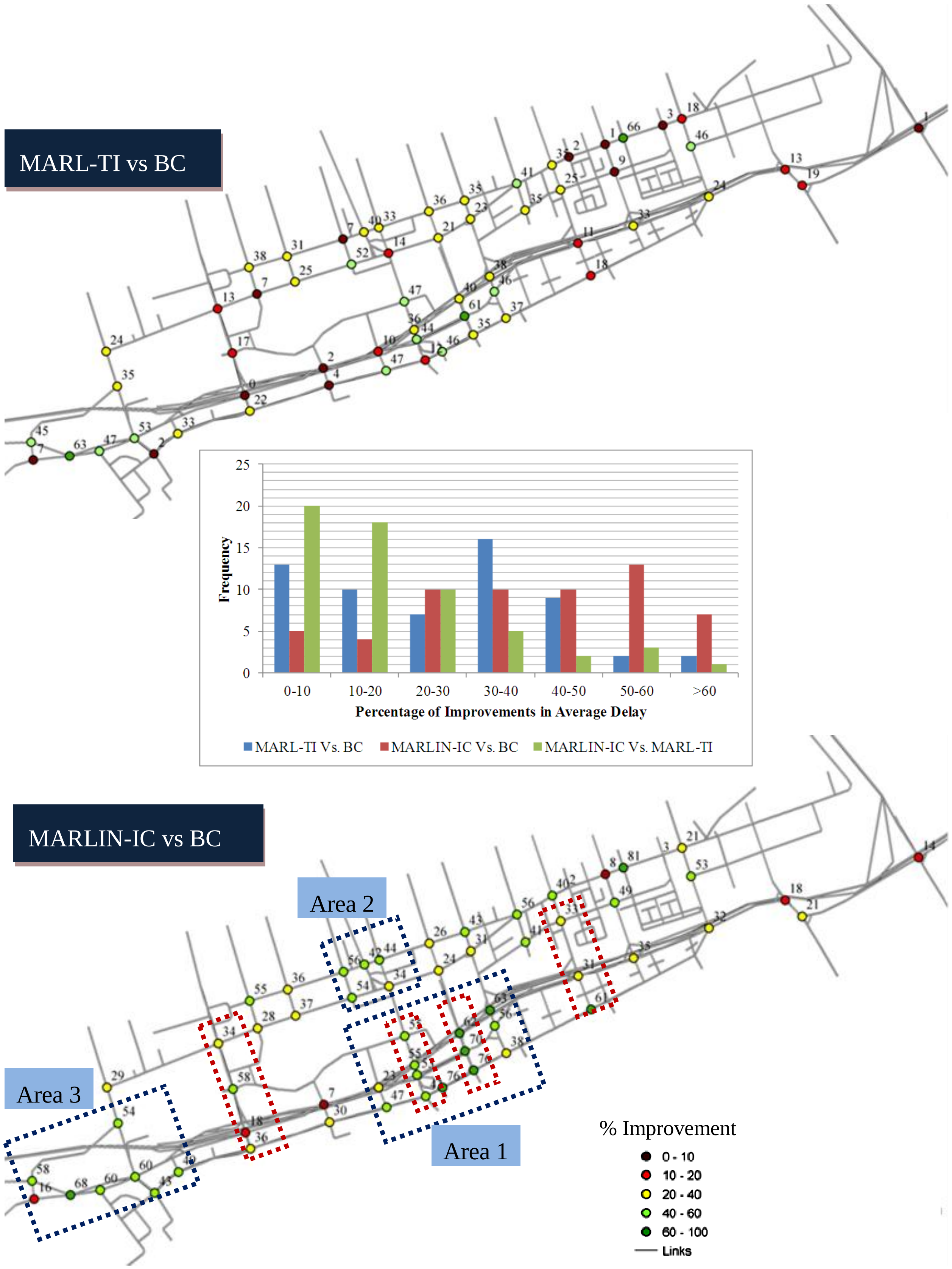
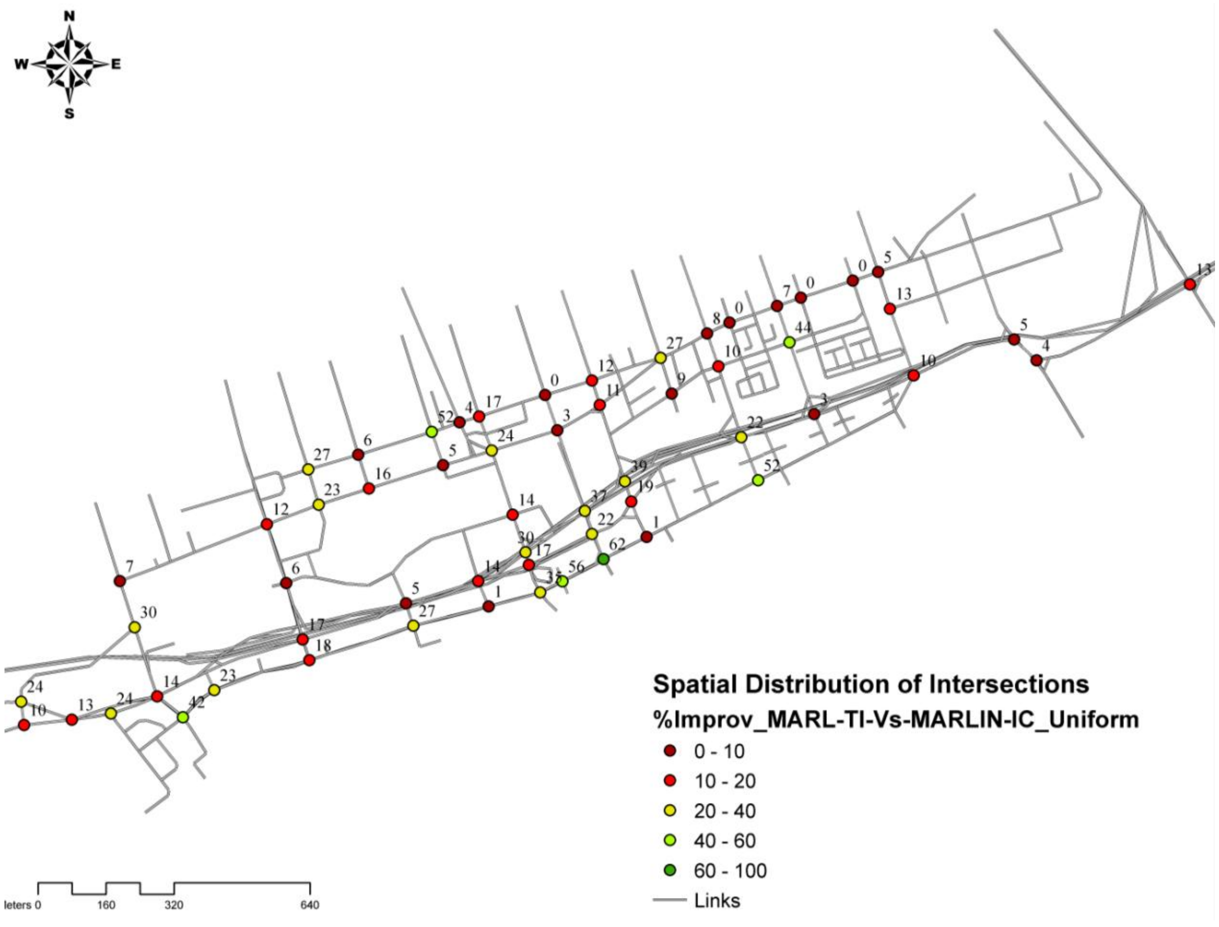


Figure 7-12 Spatial Distribution of Average Delay Improvements for the Normal Scenario



N
W
E
S
Spatial Distribution of Intersections
%Improv_MARL-TI-Vs-MARLIN-IC_Uniform
● 0 - 10
● 10 - 20
● 20 - 40
● 40 - 60
● 60 - 100
Links
Meters 0
160
320
640

Figure 7-13 Comparison of Average Delay Improvements for MARLIN-IC vs MARL-TI in the Normal Scenario

The emission levels presented in Table 7-5 are aggregated for all the links in the network. To further understand which links contribute the most to the noted savings in CO2 emission factors, the spatial distribution of the emission factors is plotted in Figure 7-14 for the BC, MARL-TI, MARLIN-IC cases. It is interesting to show that the emission levels for some links range between 0–250 gm/km while others are > 1500 gm/km. The analysis of Figure 7-14 leads to the following conclusions:

- In the BC scenario, unsurprisingly, the majority of emissions are concentrated within the busy area of the network as noted in Figure 7-14. Also large emission levels are observed along the Gardiner Expressway, especially near the off-ramps of Spadina St., York St. and Jarvis St.; which is expected given the limited downstream capacity for traffic exiting the freeway to the urban street network with signalised intersections;

- It is found the MARL-TI and MARLIN-IC reduce the emission factors compared to the BC, which indicates that more distances are travelled on each link within the same period of time, with a fewer number of stops, and less average delay. This is indicative of better performance and less congestion in the network. MARLIN-IC results in the least emission levels when compared to MARL-TI and BC, which is not surprising given the savings reported in Table 7-5;
- It is interesting to find that the emission levels on the Gardiner Expressway are reduced under MARLIN-IC when compared to the BC scenario. This confirms the fact that the downstream capacity of urban networks can certainly affect congestion on the freeway off-ramps and consequently the freeway mainline.

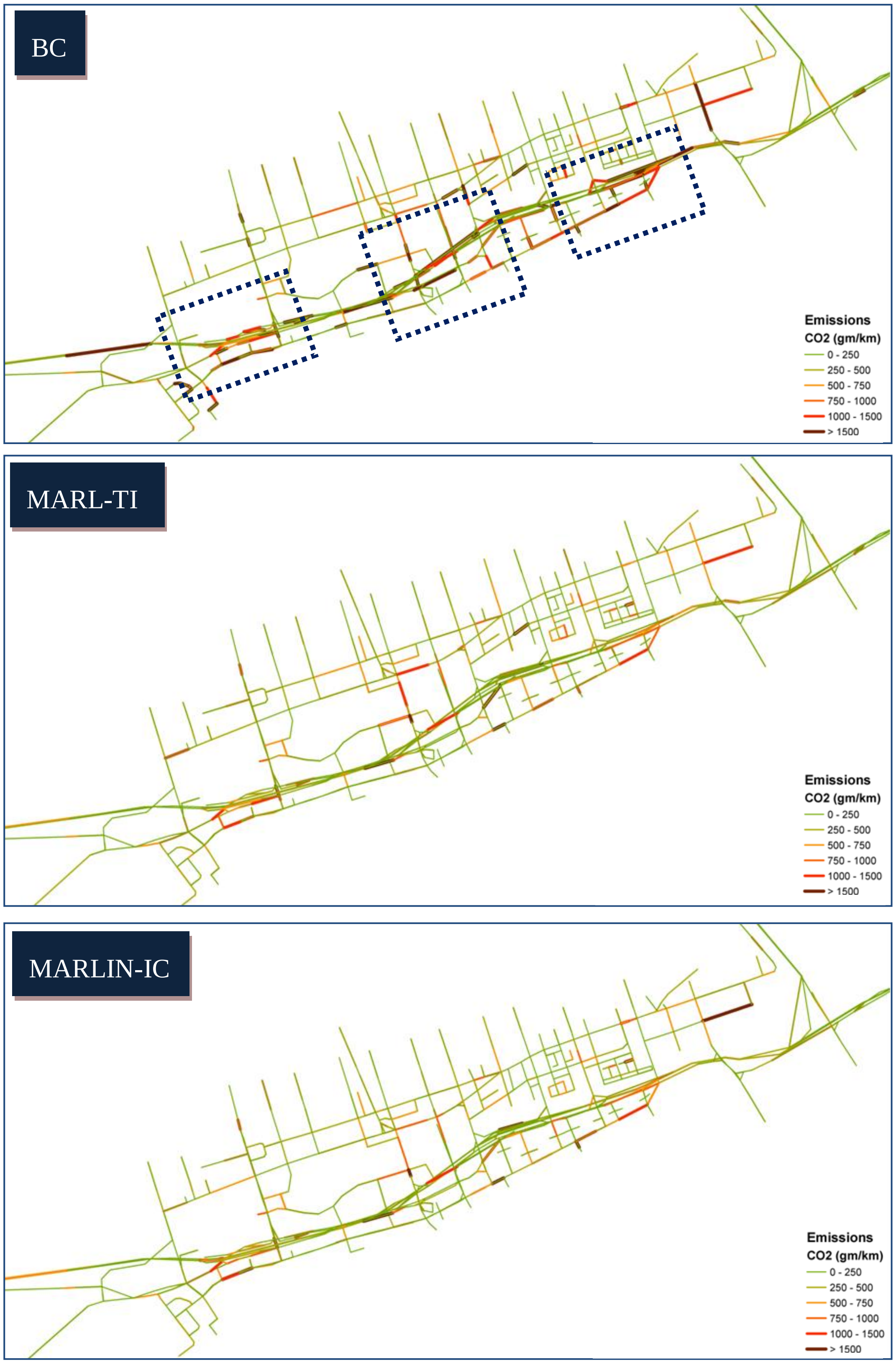
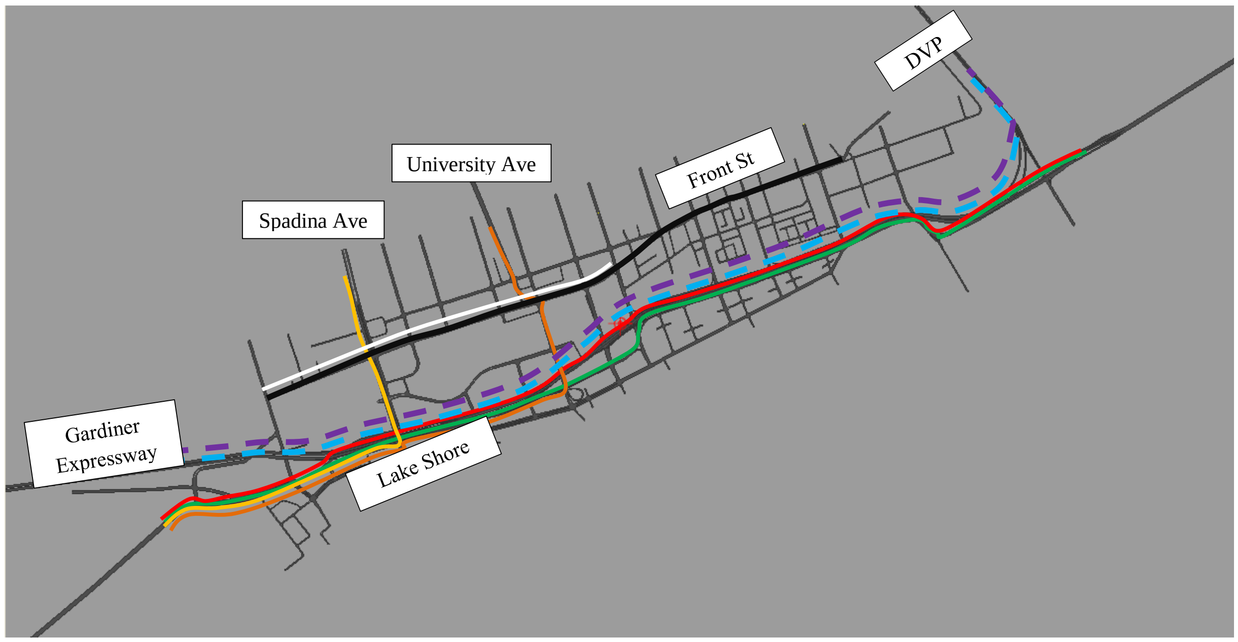


Figure 7-14 Emission Factors Spatial Representation

It is important to study the effect of various control systems on the travel time and travel time variability for selected key routes in the lower downtown core of Toronto. Eight key routes are defined as shown in Figure 7-15.



🟦	1- Gardiner EB	Length: 6.2 km	Free Flow Speed: 90 kph
🟪	2- Gardiner WB	Length: 6.1 km	Free Flow Speed: 90 kph
⬛	3- Front EB	Length: 3.5 km	Free Flow Speed: 50 kph
⬜	4- Front WB	Length: 2.1 km	Free Flow Speed: 50 kph
🟩	5- Lake Shore EB	Length: 6.2 km	Free Flow Speed: 60 kph
🟥	6- Lake Shore WB	Length: 5.5 km	Free Flow Speed: 60 kph
🟨	7- Lake Shore EB to Spadina NB	Length: 2.7 km	Free Flow Speed: 50 - 60 kph
🟧	8- Lake Shore EB to University NB	Length: 3.9 km	Free Flow Speed: 50 - 60 kph

Figure 7-15 Selected Busiest Routes in the AM Peak Hour

Route travel times and Std in travel time for the BC, MARL-TI, and MARLIN-IC scenarios are presented in Table 7-6. To further study the route travel times within the simulation hour, the travel time for the eight routes are plotted in Figure 7-16. The analysis of Table 7-6 and Figure 7-16 leads to the following conclusions:

- It is clear that MARLIN-IC outperforms MARL-TI and BC in all routes. The % improvements range from 4% in route 2 to 30% in route 7. MARL-TI outperforms BC in almost all cases; the % improvements range from 3% in route 1 to 15% in route 5 with the exception of route 7, the BC scenario performs better than MARL-TI;

- It is interesting to find that the Gardiner Expressway EB traffic (inbound) travel time improves by 19% in the MARLIN-IC scenario. Alleviating the congestion on the Spadina St. and York St. off-ramps contributes the most to these savings. This clearly shows the effect of the downstream capacity on the freeway performance. The Gardiner WB traffic was not as congested as the EB, but MARLIN-IC still achieves a 4% improvement in average route travel times;
- The most congested routes appear to be routes 7 and 8, through which traffic originated at the West end of the study area and is destined to the downtown core (Spadina St. and University Ave.). MARLIN-IC achieves 30% and 26% improvements in route 7 and 8, respectively, which reflects the superior effect of the two-dimensional coordination between agents. MARL-TI improves route 8 by around 15%, however the performance of route 7 is worse. From observing the simulation it is found that the intersection of Spadina & Front Sts. and Spadina & Bremner Sts. are those contributing to that deterioration in the route 7 travel time. It is also worth noting that this condition only occurs at the beginning of the simulation hour, and then with the course of the simulation hour the performance of MARL-TI is improved. This is clearly shown in Figure 7-16;
- From observing the temporal distribution of route travel time across the simulation hour it is generally found that MARLIN-IC is stable and exhibits less variation compared to the BC and MARL-TI scenarios. While the BC scenario exhibits the highest variability in travel time (as shown in the Std values in Table 7-6), MARL-TI still shows some variations, most notably in the most two congested routes (routes 7 and 8). MARLIN-IC shows stable route travel times along all routes;
- From observing the route travel times along route 5, which shares the first section of routes 7 and 8, it is found that the great variability in routes 7 and 8 is due to the sections in the downtown core (i.e. Spadina NB and University NB), while the Lake Shore EB section shows less variations in travel times. Therefore it is highly recommended to coordinate the signals along Spadina Ave. and University Ave. Although the BC scenario shows high variability in travel time for route 5, MARL-TI is close to the performance of MARLIN-IC, therefore there is no urgent need for coordination along LS EB.

Table 7-6 Route Travel Times for BC, MARL-TI, and MARLIN-IC

System / Route	BC	MARL-TI	MARLIN-IC	% Improvments MARL-TI Vs. BC	% Improvments MARLIN-IC Vs. BC	% Improvments MARLIN-IC Vs. MARL-TI
1- Gardiner EB	5.14	4.98	4.15	3.18%	19.30%	16.65%
St Dev	1.15	0.86	0.12	0.25	0.90	0.86
2- Gardiner WB	4.42	4.27	4.23	3.35%	4.35%	1.04%
St Dev	0.20	0.15	0.06	26.52%	68.03%	56.49%
3- Front EB	10.65	9.13	7.88	14.28%	13.69%	13.69%
St Dev	2.15	1.22	0.60	43.26%	72.27%	51.13%
4- Front_WB	5.55	5.34	5.10	3.81%	8.15%	4.51%
St Dev	0.92	0.79	0.49	13.39%	47.10%	38.93%
5- LakeShore EB	16.31	13.28	12.10	18.60%	25.77%	8.82%
St Dev	3.74	1.37	1.37	63.38%	63.49%	0.31%
6- LakeShore WB	10.31	9.07	8.46	12.02%	17.91%	6.70%
St Dev	1.03	0.69	0.50	33.30%	51.09%	26.67%
7- LakeShore EB to Spadina NB	10.94	11.86	7.70	-8.40%	29.59%	35.05%
St Dev	3.75	3.07	1.75	18.25%	53.44%	43.05%
8- LakeShore EB University to Ave NB	12.05	10.24	8.87	15.04%	26.38%	13.36%
St Dev	2.81	1.66	1.64	40.80%	41.75%	1.62%

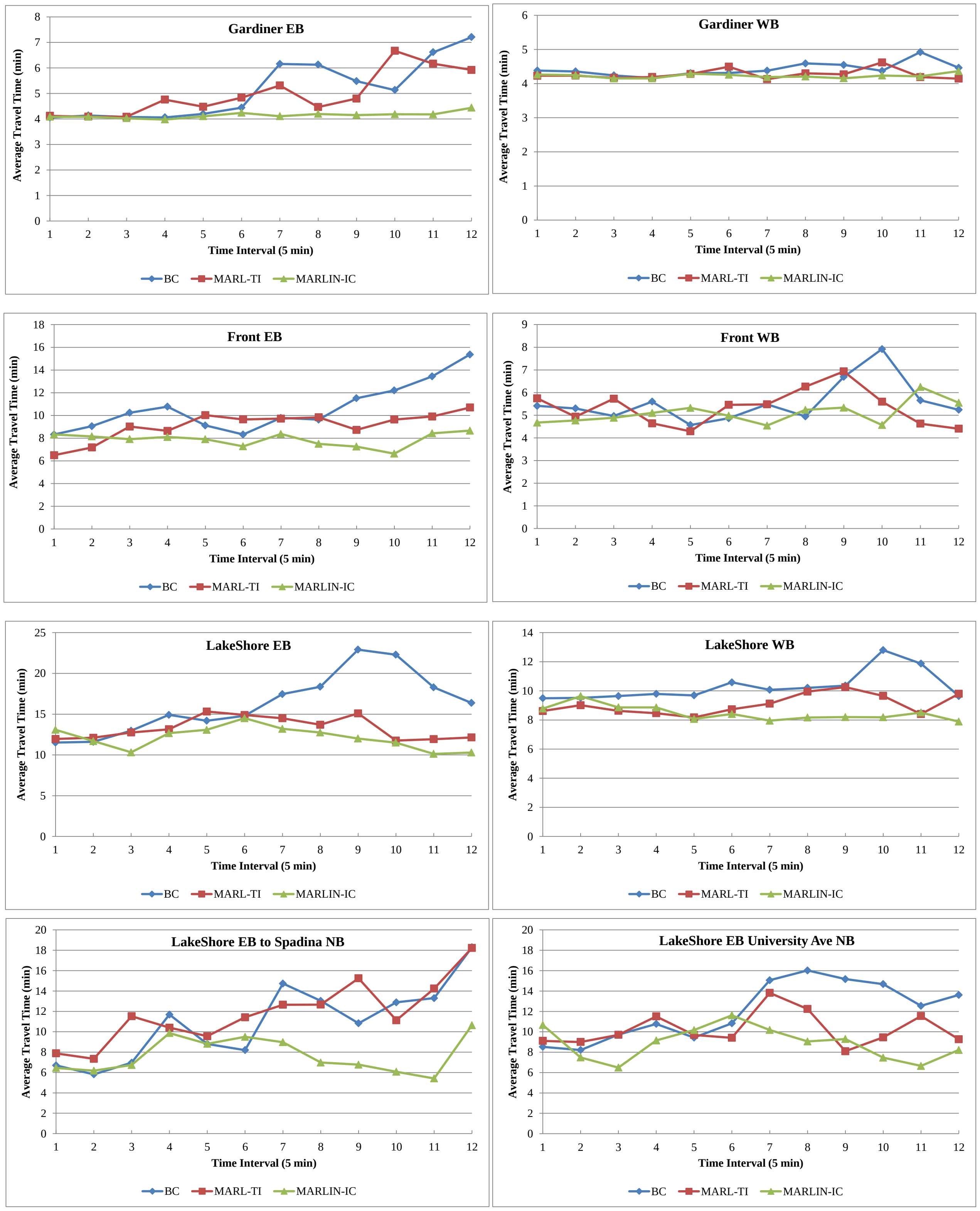


Figure 7-16 Average Route Travel Times for the Selected Routes

7.3.2.2 Variable Demand Profile

Due to the large-scale size of the network and for the sake of conciseness, only average delay, throughput, and average and Std of queue length are presented for the variable profile test case. While the main focus in this section is to investigate the performance of MARLIN-ATSC under variable conditions, a comparison of these results against the uniform profile case is desirable. Table 7-7 shows the network-wide MOE for the uniform profile case (discussed in section 7.3.2.1) and the variable profile case.

Table 7-7 Network-Wide MOE for Variable Profile Scenario

Profile	MOE \ System	BC	MARL-TI	MARLIN-IC	% Improvments MARL-TI Vs. BC	% Improvments MARLIN-IC Vs. BC	% Improvments MARLIN-IC Vs. MARL-TI
Uniform	Average Delay (sec/veh)	35.27	25.72	22.02	27.06%	37.57%	14.41%
	Throughput (veh)	23084	23732	24482	2.81%	6.06%	3.16%
	Avg Queue Length (veh)	8.66	6.60	5.88	23.77%	32.07%	10.88%
	Std. Avg. Queue Length (veh)	2.12	1.62	1.47	23.37%	30.74%	9.61%
Variable	Average Delay (sec/veh)	**46.26**	**29.41**	**24.51**	**36.41%**	**47.01%**	**16.67%**
	Throughput (veh)	**18452**	**21600**	**24518**	**17.06%**	**32.87%**	**13.51%**
	Avg Queue Length (veh)	**14.20**	**9.97**	**6.26**	**29.84%**	**55.91%**	**37.16%**
	Std. Avg. Queue Length (veh)	**3.19**	**2.22**	**1.55**	**30.45%**	**51.45%**	**30.20%**

The analysis of the results shown in Table 7-7 leads to the following findings:

- Compared to the uniform profile case, considerable deterioration in the performance of BC is observed (24% increase in average delay, 25% decrease in throughput, 39% increase in average queue length, and 34% increase in Std of average queue lengths);
- The two MARLIN-ATSC algorithms considerably outperform the BC in all the MOEs. The most notable improvements are in average delay (47% MARLIN-IC vs BC), throughput (33% MARLIN-IC vs BC), average queue length (56% MARLIN-IC vs BC), and Std of average queue length (51% MARLIN-IC vs BC);
- MARL-TI outperforms the BC in all the MOEs, most notably the average intersection delay (36%). Additionally, comparing MARLIN-IC to MARL-TI it is found that the latter experience higher delays because in MARL-TI the actions are based on local traffic states with no coordination between other agents, therefore MARL-TI results in more vehicles retained in the network at the end of the simulation (14% throughout improvement in MARLIN vs MARL-TI). It is found that MARL-TI exhibits higher average queue length values than MARLIN (37%) and less balancing for the queue lengths across the approaches (30%);

- This deterioration in the performance of the BC in the variable profile case is due to the fact that high variations in the traffic stream are causing long queues to form and blockages at some intersections. These cases result in substantial performance degradation in a cascading fashion (especially in this grid-like urban network) as the traffic control system in the BC could not respond to the change in traffic patterns in an adequately adaptive manner. This observation is a clear indication that the BC is not robust, as its performance degrades considerably with slight changes in the traffic conditions. The key to the good performance achieved by MARLIN-ATSC, on the other hand, is that agents could promptly adapt the signal timings, by skipping unnecessary phases and by providing optimised timing per phase to any variations in the traffic stream.

To further understand which intersections contribute the most to the above noted savings, the spatial distribution of average intersection delay is plotted in Figure 7-17. It is interesting to show that in the BC model some intersections encounter delay in the range of 5–10 sec/veh, while others encounter 110–170 sec/veh in the BC scenario.

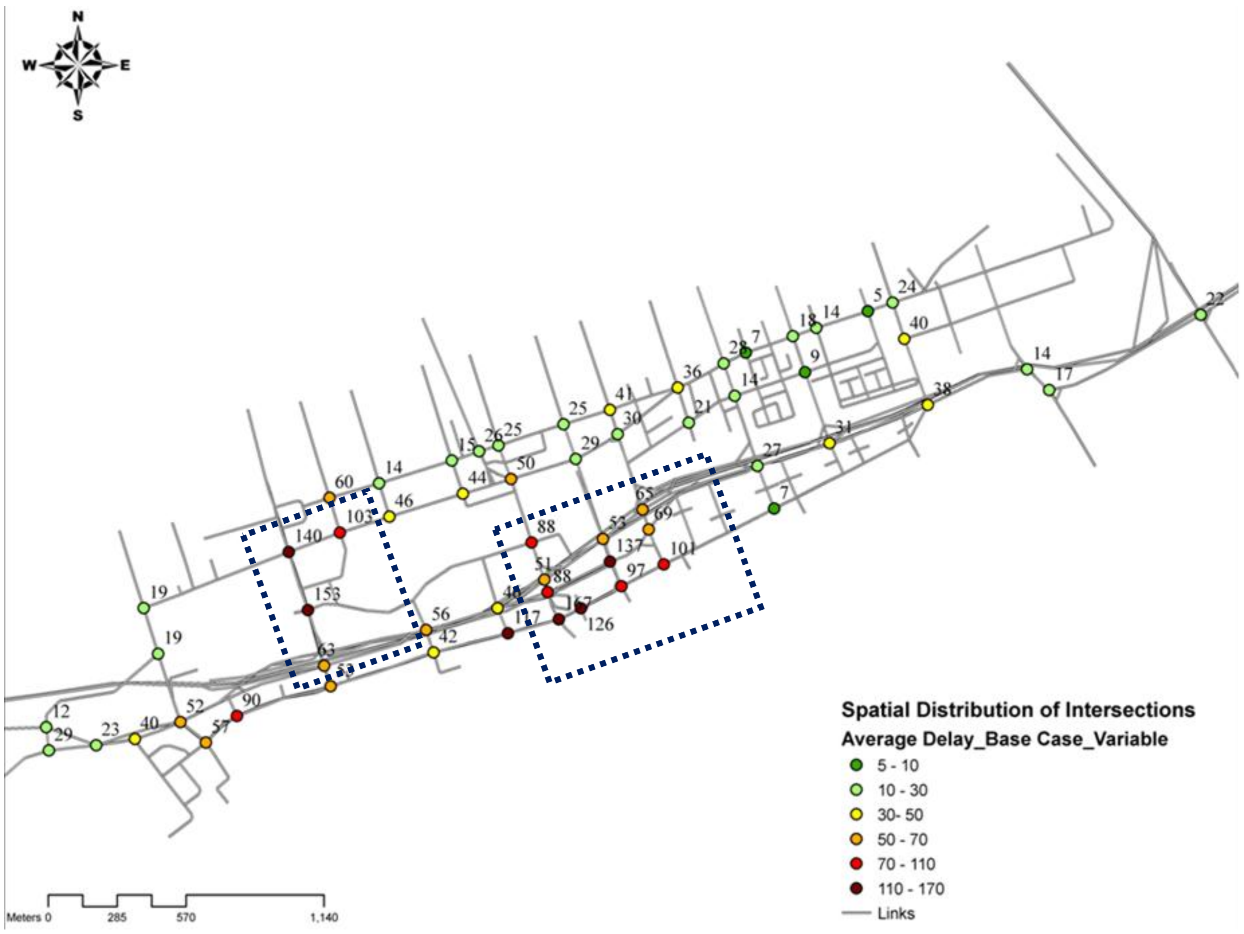


Figure 7-17 Spatial Distribution of Average Delay Using Base Case Under Variable Demand Profile

As shown in Figure 7-17, the intersections experiencing the highest delay are highlighted by the dotted shaded areas. These are the same intersections that contribute to the highest delay in the normal scenario but with higher values of delay due to the variations in arrival profiles that affect the intersections' performance in a cascading fashion.

Table 7-7 shows a very promising overall performance of MARLIN. However, as shown from the wide range of average delays among intersections, the improvements in some intersections are much higher than the network averages. Therefore the spatial distribution of percentage improvement and a histogram of these improvements are presented in Figure 7-18 and Figure 7-19.

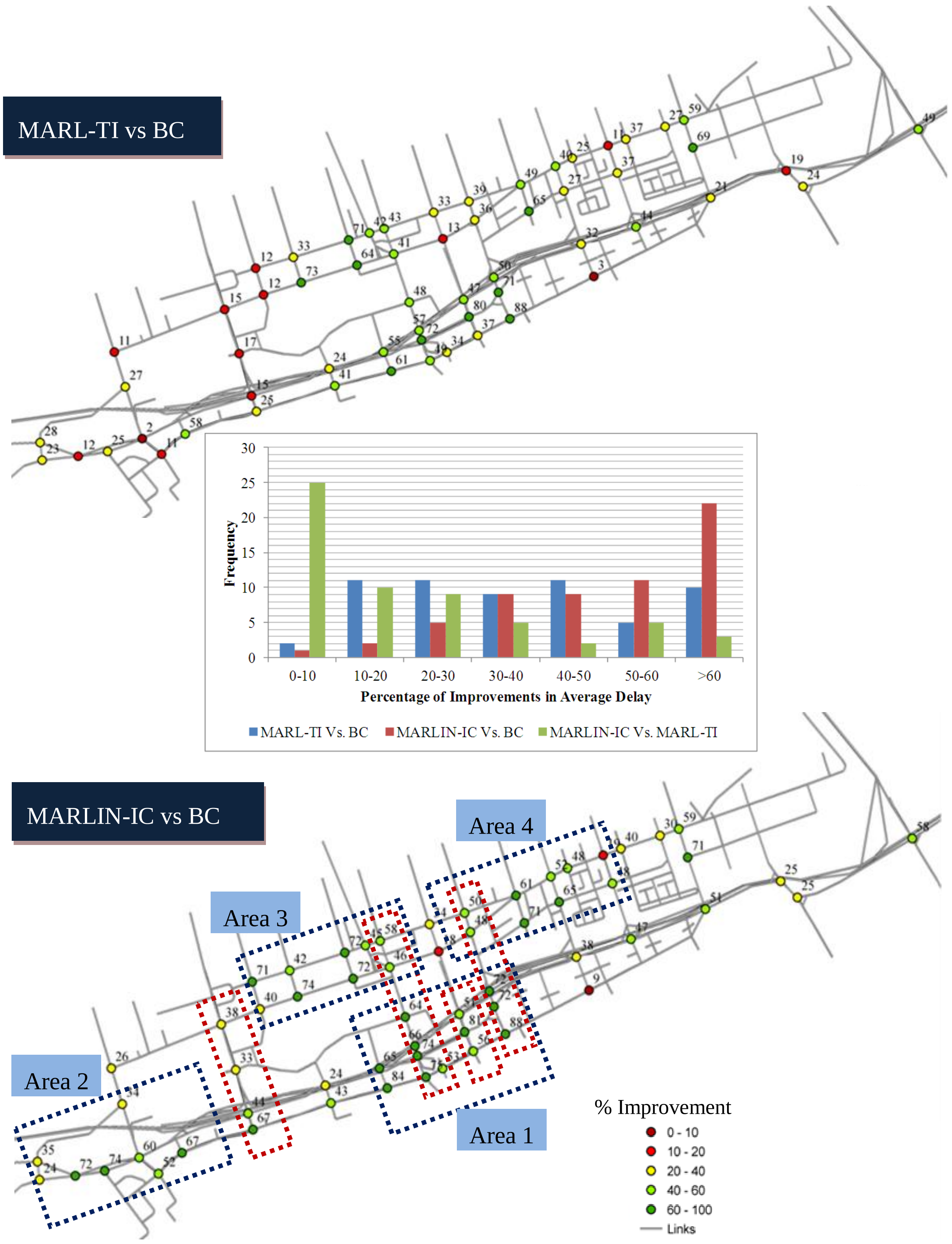


Figure 7-18 Spatial Distribution of Average Delay Improvements for the Variable Profile Scenario

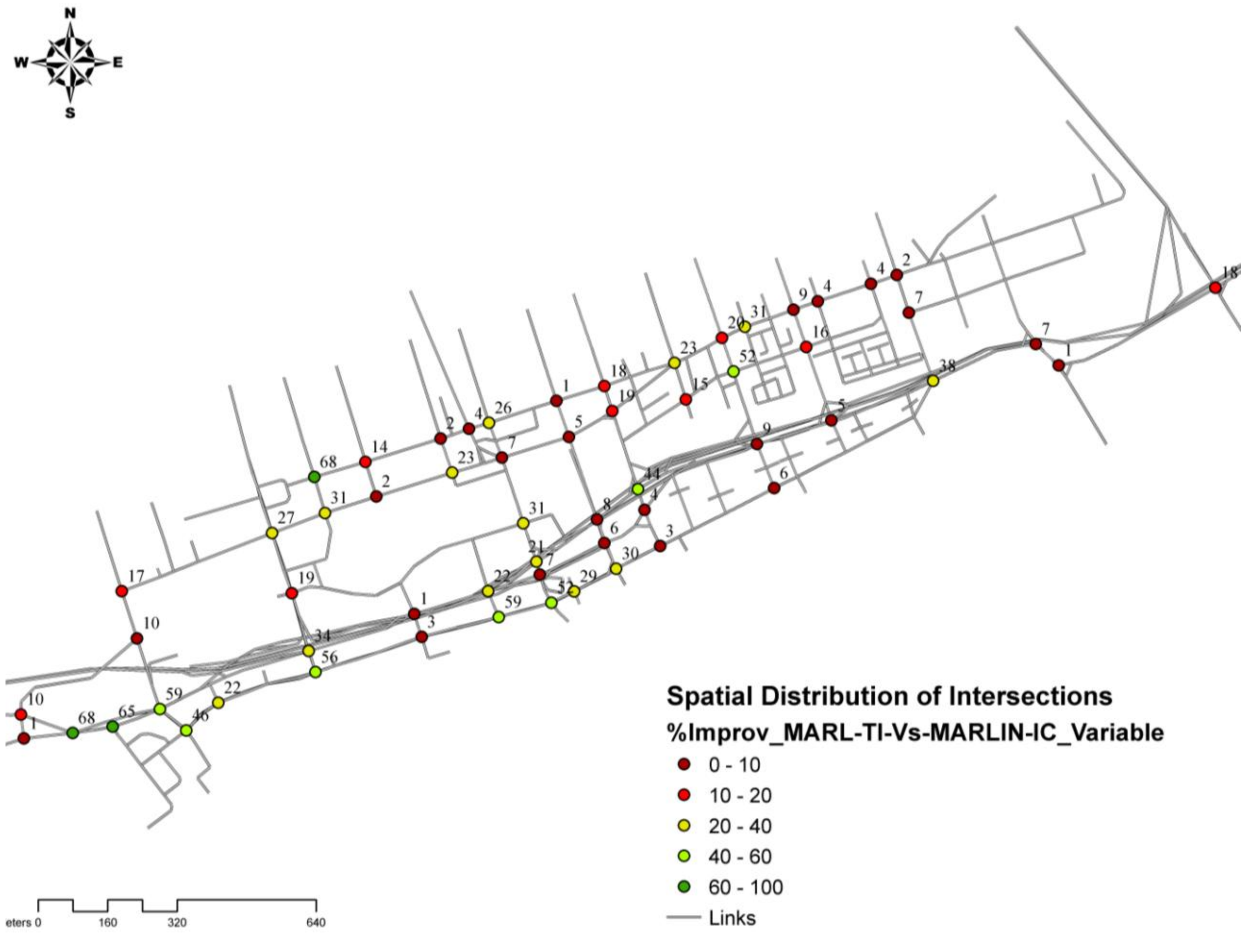

N
S
W
E
Spatial Distribution of Intersections
%Improv_MARL-TI-Vs-MARLIN-IC_Variable
0 - 10
10 - 20
20 - 40
40 - 60
60 - 100
Links
eters 0
160
320
640

Figure 7-19 Comparison of Average Delay Improvements for MARLIN-IC vs MARL-TI in the Variable Profile Scenario

The analysis of Figure 7-18 and Figure 7-19 leads to the following conclusions:

- In the MARL-TI case, most of the savings occur, interestingly, at chronically busy intersections such as intersections along Lake Shore, Queens Quay, York and University Sts.;
- Similar to MARL-TI, in MARLIN-IC most of the savings are found at the busiest downtown intersections, but what is interesting to note is the formation of a "corridor of savings" and "area of savings" in the MARLIN-IC case as shown in the dotted shaded areas in Figure 7-18. The effect of the coordination among agents in MARLIN-IC is shown to be more distinct in the variable profile compared to the normal scenario. To illustrate, the intersections at "Area 2" and "Area 3" and along the corridors of York St. and Spadina St. (from Queens Quay St. to Front St.) exhibit large savings, which are not observed in the MARL-TI case. This is primarily due to the coordination effect among agents. This observation is extremely

important as this is the only methodical way to determine which intersections warrant MARLIN-ATSC with communication between agents as opposed to MARLIN-ATSC without communication. It is clear from this example that these intersections are perfect examples for such a case;

- It is interesting to observe the effect of the “area control” effect in some chronic areas in the downtown core. The off-ramps at Spadina St., Yonge St., and York St. form major access points to the city’s core, therefore if traffic is at halted at these locations (and their downstream intersections) the entire network could easily deteriorate into a grid-lock condition. The MARLIN-IC algorithm is found to be intelligent enough to capture such a pattern while controlling these “critical” intersections, and therefore provides substantial savings compared to both BC and MARL-TI;
- As a result, the area noted “Area 1” in Figure 7-18 warrants coordination among all the intersections within this area, especially under variable arrival patterns during the morning peak. From observing the simulation model and the network characteristics, two main factors are contributing to the superior performance of MARLIN-IC compared to MARL-TI: 1) closely-spaced intersections within this area, and 2) the high demand for LT movements (EB to NB traffic) in most of these intersections which creates oversaturation conditions in which MARL-TI fails to converge to the optimal control policy, as explained in section 6.6.1;
- One interesting observation is the intersection of Yonge St. and Queens Quay St. as shown in Figure 7-20. This is an intersection that shows the high average delay in the BC and high savings in the average delay using MARLIN vs both BC and MARL-TI. From observing the simulation model it is found that due to congestion downstream of the York St. off-ramp, vehicles are willing to exit at the Yonge St. off-ramp as a better alternative. For these vehicles to reach their destination, a sequence of turning movements is required: RT (Yonge St. and Lake Shore East), RT (Yonge St. and Queens Quay St.), and LT (York St. and Queens Quay). In such cases the coordination among these three intersections plays a crucial role in MARLIN-IC compared to the BC;
- Figure 7-19 shows the spatial distribution of the percentage savings of MARLIN-IC over MARL-TI under a variable profile. It is found that the majority of savings (40–60%) achieved by the coordination effect are on the intersections on the access points to the downtown area from Lakeshore West;

-

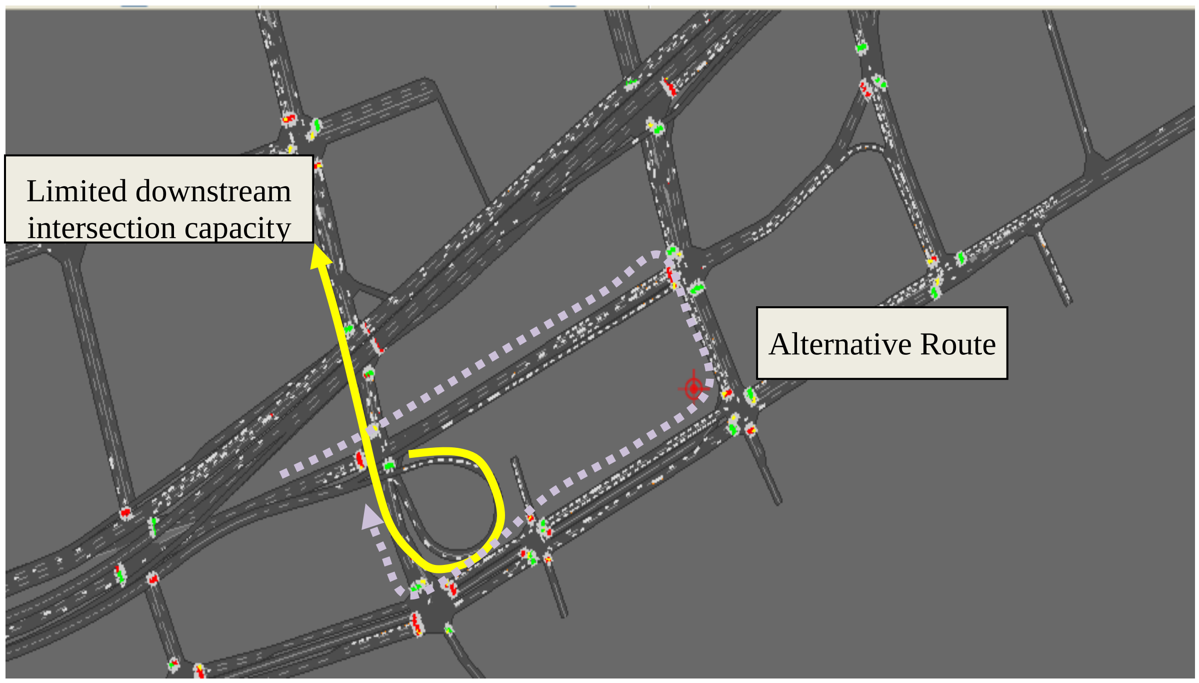
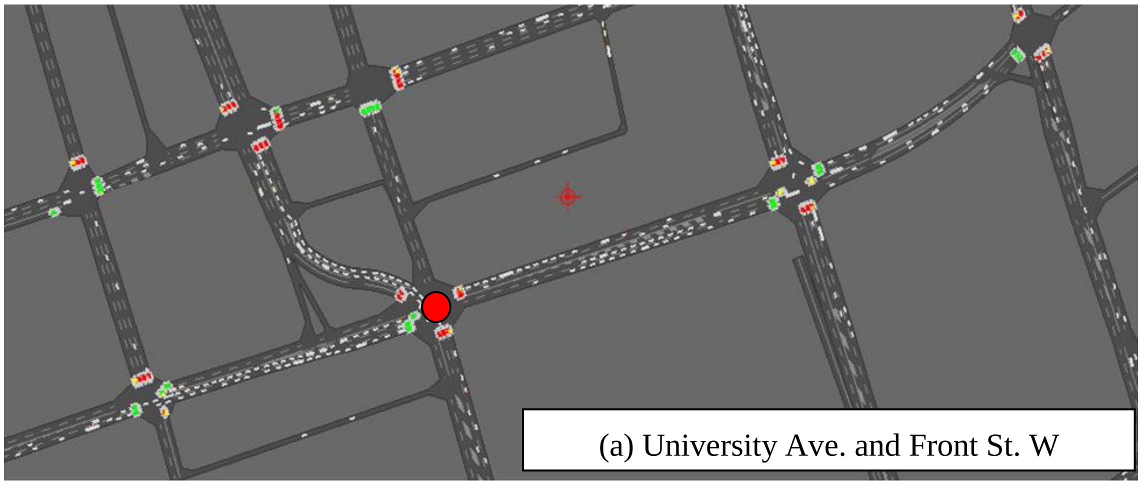


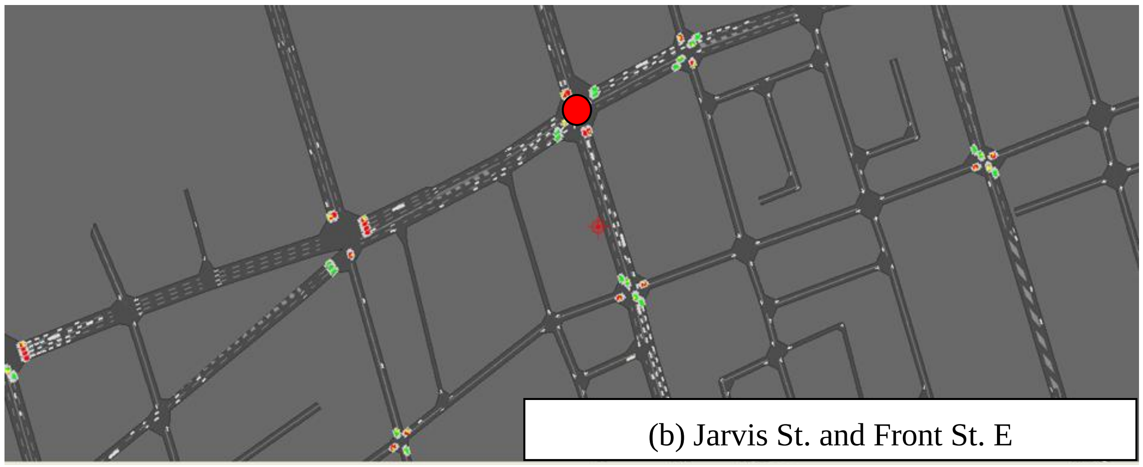
Figure 7-20 Snapshot for "Area 1" During the Simulation for BC Scenario

- Similarly, “Area 3” is known to be chronically congested in the morning peak due to the dense business and economic activities near Front St., Wellington St., York St., and Simco St. In addition, in the West end of the network, LS and Fort York St. are two alternative routes to Bathurst St., which is also observed to be congested in the morning peak. This area (“Area2”) exhibits major savings that warrant “area control” using MARLIN-IC;
- In Figure 7-18 the histogram of percentage savings for different control systems is shown. It is found that the majority of savings (40–60%) achieved by MARLIN-IC over BC are within the defined “area of control” intersections. Comparing MARL-TI to MARLIN-IC it is interesting to find that the majority of savings (0–10%, 10–20% and 20–30%) are actually for the same “area of control” noted above, which confirms that the additional “average” 17% outperformance (in average delay) of MARLIN-IC to MARL-TI – reported in Table 7-5 – are mostly concentrated and attributed to these areas of the network;
- Another intersection that contributes to the results above is the intersection of Bay St. and University Ave. As shown in Figure 7-21 (a), at this intersection a high EB LT is observed in addition to the high NB, EB, and WB demands, which result in long queues approaching the upstream intersection (Simco St. and Front St. W). MARLIN-IC handles such a situation

more efficiently by “metering” the traffic from the upstream intersection to not exacerbate the congestion of the downstream intersection. This confirms the findings discussed in Chapter 6, which is that the intersections with a high LT demand benefit the most from the coordination mechanism between the intersections operated by MARLIN-IC. Similar observations are noted at the intersections of Jarvis St. and Front St. East, Spadina St. and Bremner St. as shown in Figure 7-21 (b) and Figure 7-21 (b);

- The intersection of Spadina St. and Front St. W is always congested and exhibits high values of delay as shown in Figure 7-22. An alternative route for traffic destined EB on Front St., is Bremner St.–Blue Jays St. (u-shape). When the Spadina St. NB approach is congested, considerable traffic volume shifts to Bremner St.–Blue Jays St., which results in congestion in the latter case and high delays at the intersection of Blue-Jays St. and Front St. W. In MARLIN-IC the coordination mechanism between the two intersections prevents such congestion. Comparing the average delay of MARLIN-IC to MARL-TI, Spadina St. and Front St. saves 13%, and Blue-Jays St. and Front St. saves 28%.





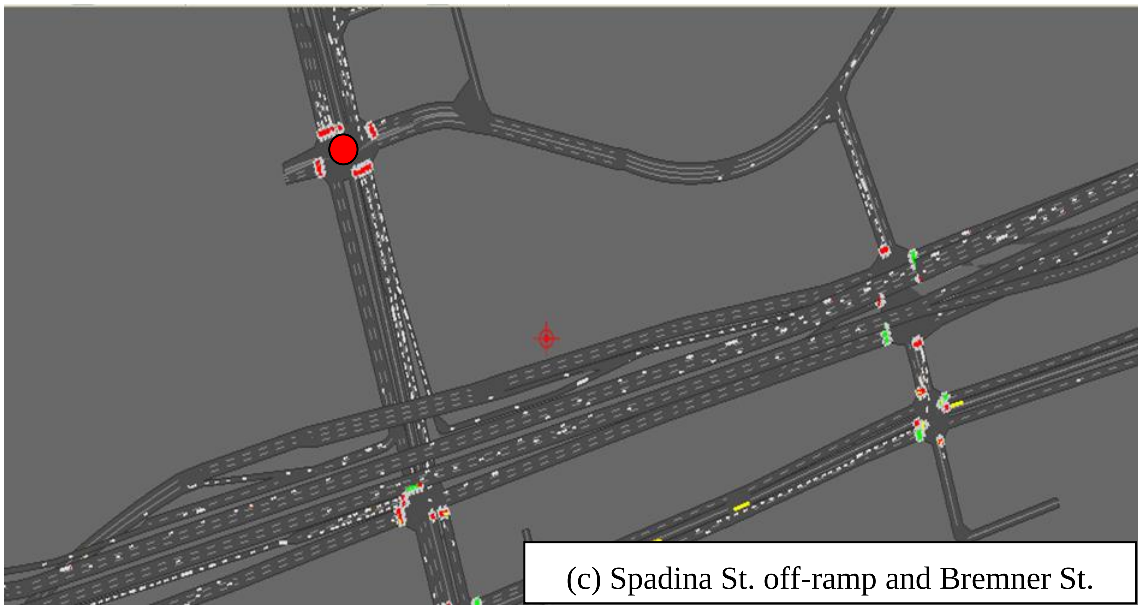


Figure 7-21 Snapshots to Illustrate the Performance of MARLIN-IC vs MARL-TI for the Variable Profile Scenario

-

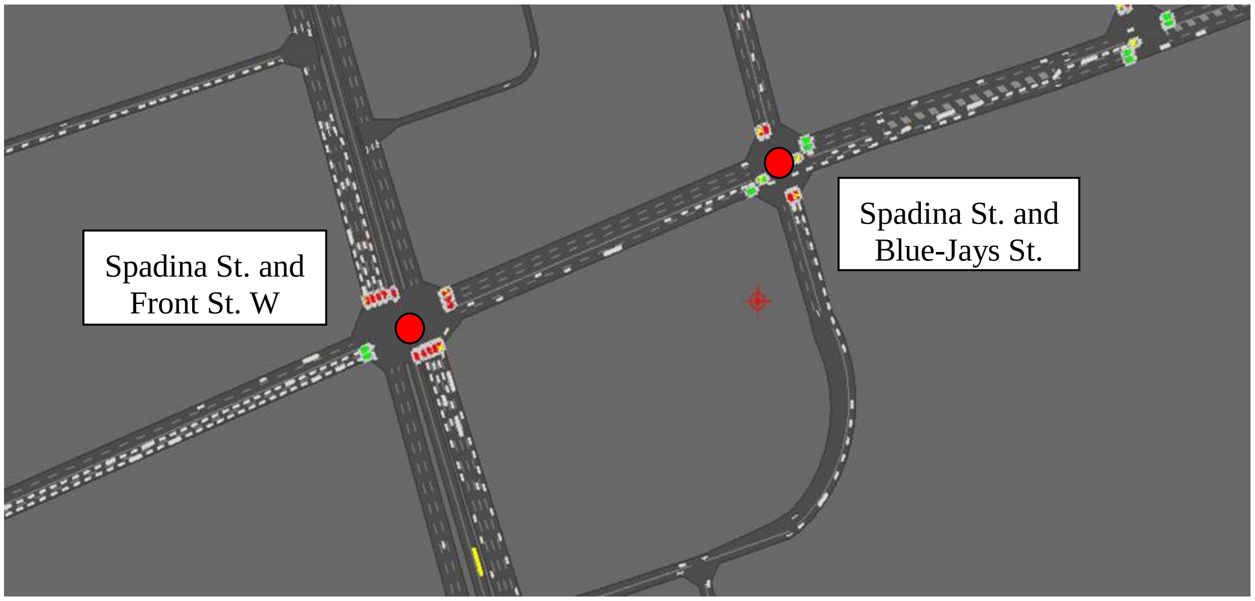


Figure 7-22 Effect of MARLIN-IC on Congestion and Routing Options

7.3.2.3 Drivers Unfamiliarity

While the main focus in this section is to investigate the effect of TIS and the unfamiliarity of drivers on the performance of different control systems, a comparison with the normal case profile case is desirable. Table 7-8 shows the network-wide MOE for the familiar drivers case (i.e. normal scenario), that is presented in section 7.3.2.1, and the unfamiliar drivers case.

Table 7-8 Network-Wide MOE for the Driver Familiarity Scenario

Drivers Familiarity	System / MOE	BC	MARL-TI	MARLIN-IC	% Improvments MARL-TI Vs. BC	% Improvments MARLIN-IC Vs. BC	% Improvments MARLIN-IC Vs. MARL-TI
High	Average Delay (sec/veh)	35.27	25.72	22.02	27.06%	37.57%	14.41%
	Throughput (veh)	23084	23732	24482	2.81%	6.06%	3.16%
	Avg Queue Length (veh)	8.66	6.60	5.88	23.77%	32.07%	10.88%
	Std. Avg. Queue Length (veh)	2.12	1.62	1.47	23.37%	30.74%	9.61%
Low	Average Delay (sec/veh)	**42.37**	**39.20**	**23.47**	**7.48%**	**44.60%**	**40.12%**
	Throughput (veh)	**11601**	**13739**	**22664**	**18.43%**	**95.36%**	**64.96%**
	Avg Queue Length (veh)	**13.13**	**10.43**	**6.82**	**20.53%**	**48.06%**	**34.64%**
	Std. Avg. Queue Length (veh)	**2.96**	**2.43**	**1.81**	**17.93%**	**38.71%**	**25.32%**

The analysis of the results shown in Table 7-8 leads to the following findings:

- Compared to the uniform profile case, a substantial deterioration in the performance of BC is observed using the low percentage of familiar drivers (16% increase in average delay, 34% increase in average queue length, and 28% increase in Std of queue lengths);

- The two MARLIN-ATSC algorithms considerably outperform the BC in all the MOEs. MARL-TI outperforms the BC in all the MOEs, most notably the average queue length (20%). However, comparing MARLIN-IC to MARL-TI it is found that the latter experiences higher delays and congestion (64% throughout improvement in MARLIN vs MARL-TI);
- This deterioration in the performance of the BC in the low driver familiarity is attributed to the fact that a fewer number of travellers are receiving updated travel times (and costs in general) to their destination, therefore they are less likely to switch routes in the case where their route travel time to destination increases. This effect creates oversaturation conditions in some areas of the network, where MARL-TI agents fail to converge to the optimal policy (section 6.6.1).

Figure 7-23 shows the average intersection delay for the low familiarity case and the intersections that contribute the most to the average delay. Unfamiliar drivers are typically confined to less routing options from their origin to destination. In recurrent congestion cases and typical rush hour commutes this may lead to typical rush hour travel times. However, in cases of non-recurrent congestion or queue spillback in a grid-like network, the lack of traveller information may result in a substantial drop in the network throughput.

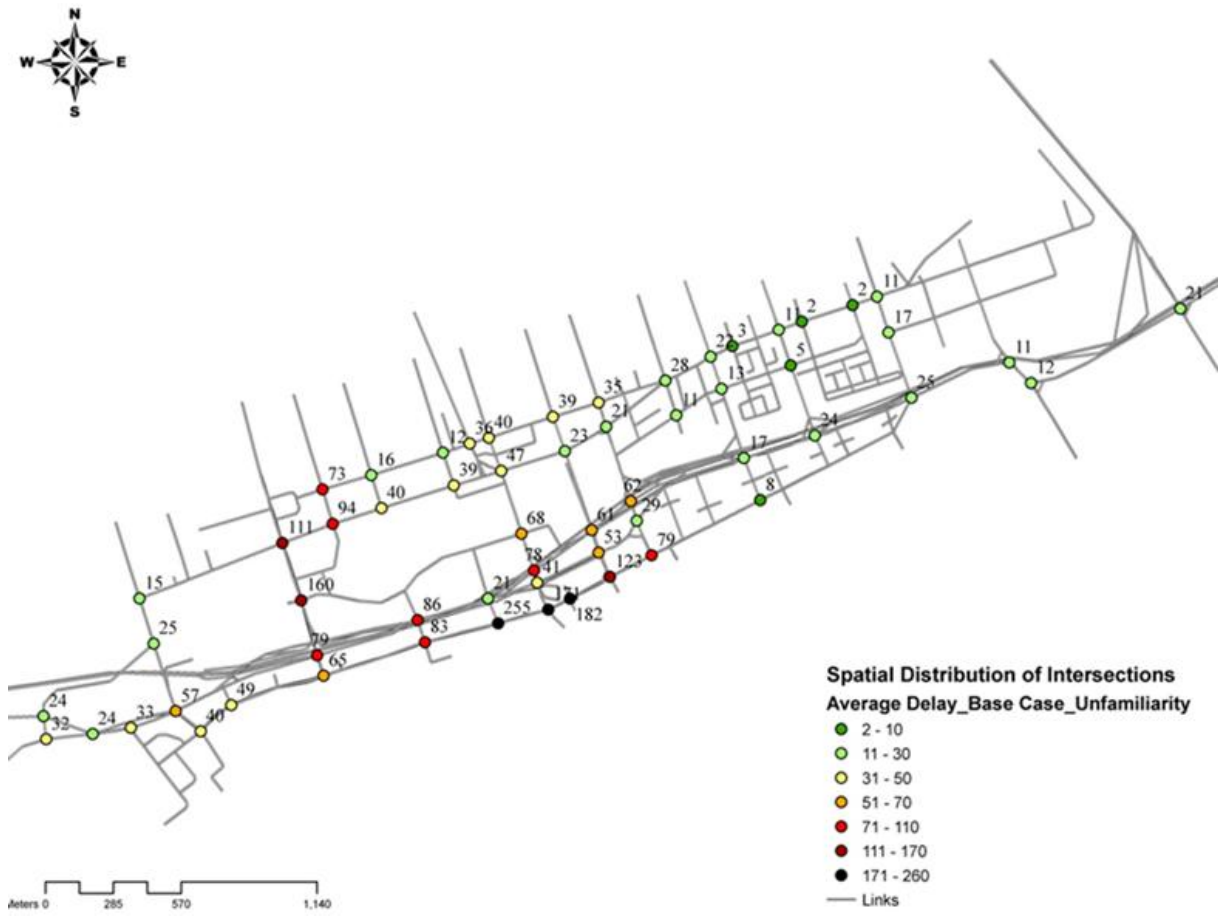
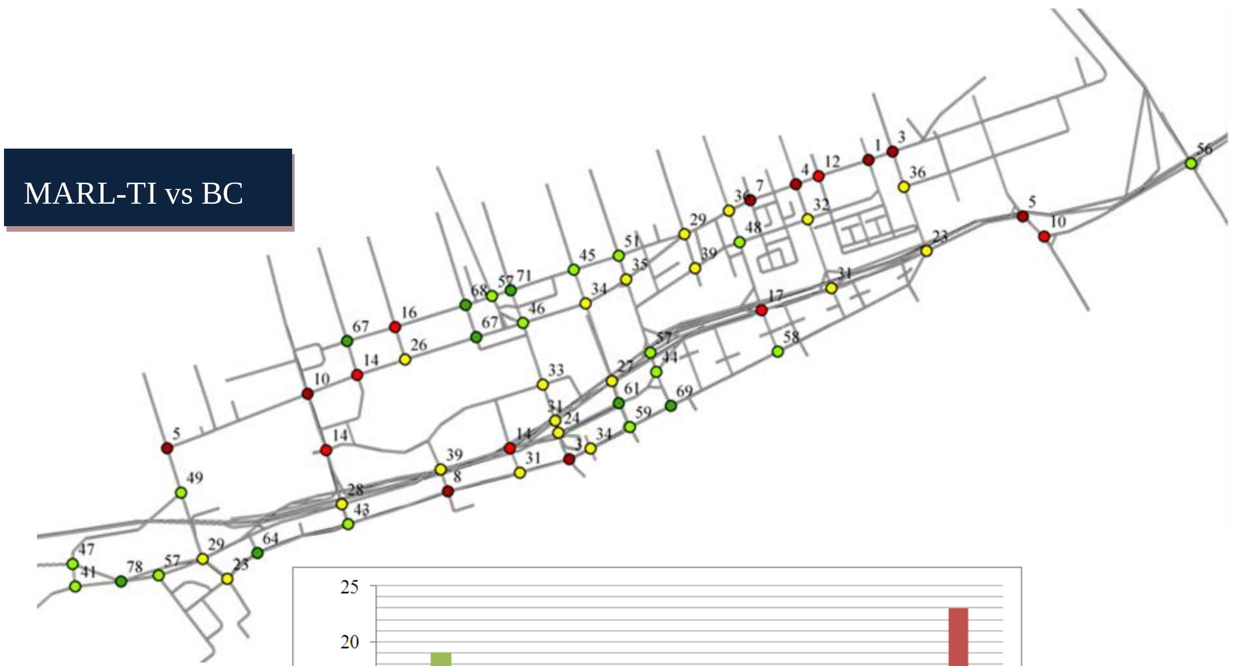
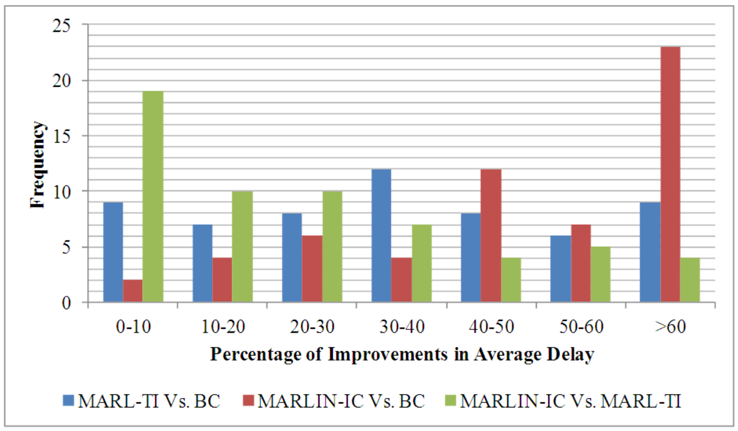


Figure 7-23 Spatial Distribution of Average Delay for Unfamiliar Drivers

The spatial distribution of percentage improvement and a histogram of these improvements are presented in Figure 7-24 and Figure 7-25.





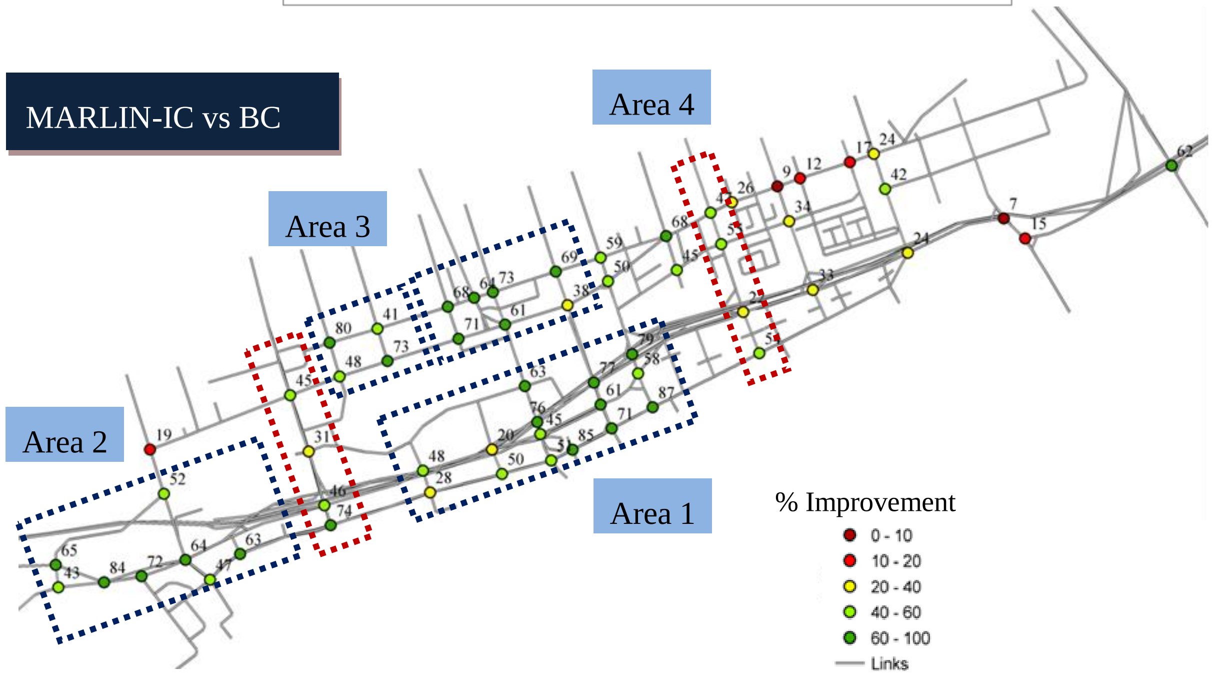


Figure 7-24 Spatial Distribution of Average Delay Savings for the Unfamiliar Drivers Scenario

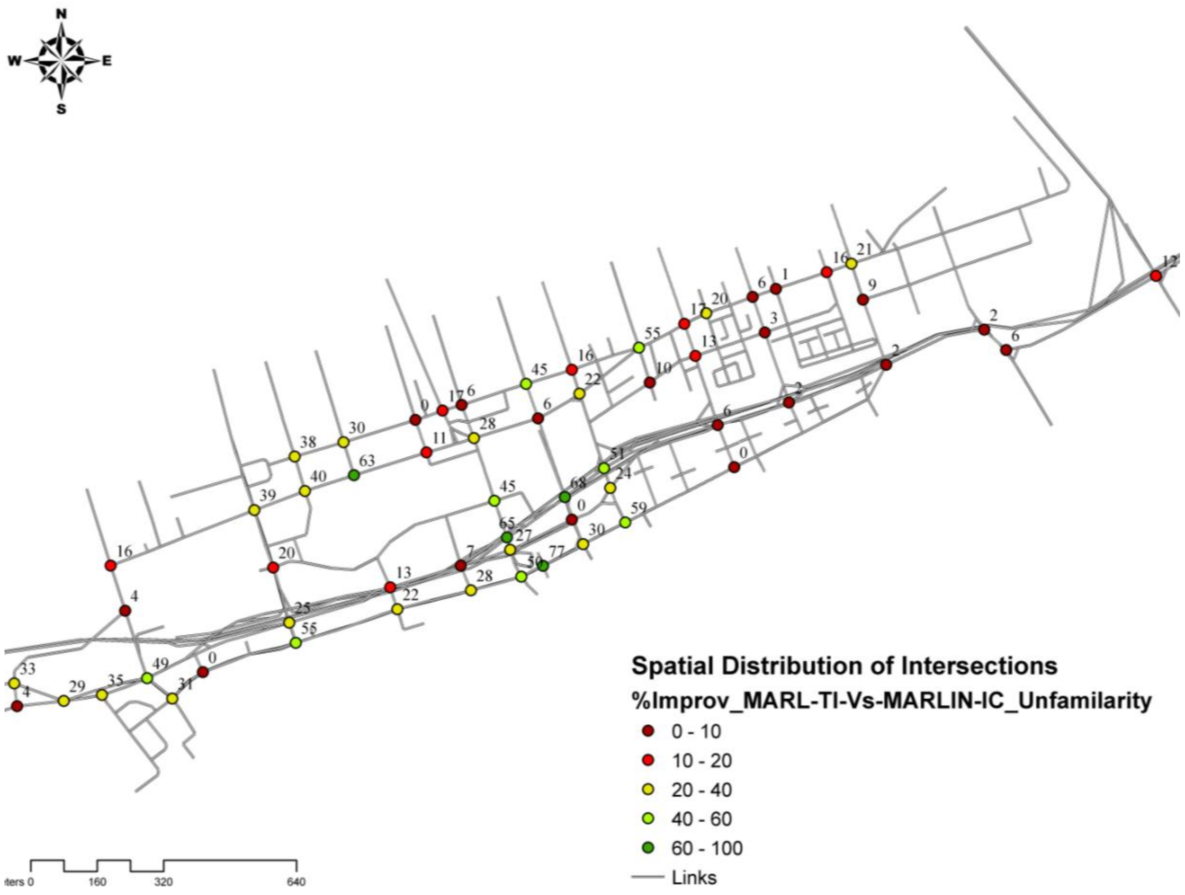
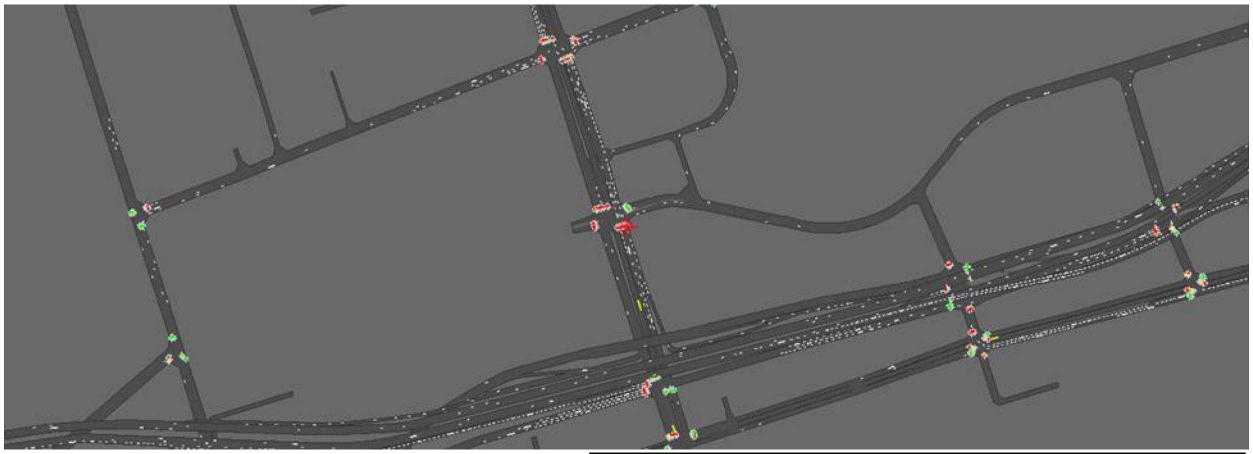

Spatial Distribution of Intersections
%Improv_MARL-TI-Vs-MARLIN-IC_Unfamilarity
● 0 - 10
● 10 - 20
● 20 - 40
● 40 - 60
● 60 - 100
Links
N
S
E
W
eters 0
160
320
640

Figure 7-25 Comparison of Average Delay Improvements for MARLIN-IC vs MARL-TI in the Unfamiliar Drivers Scenario

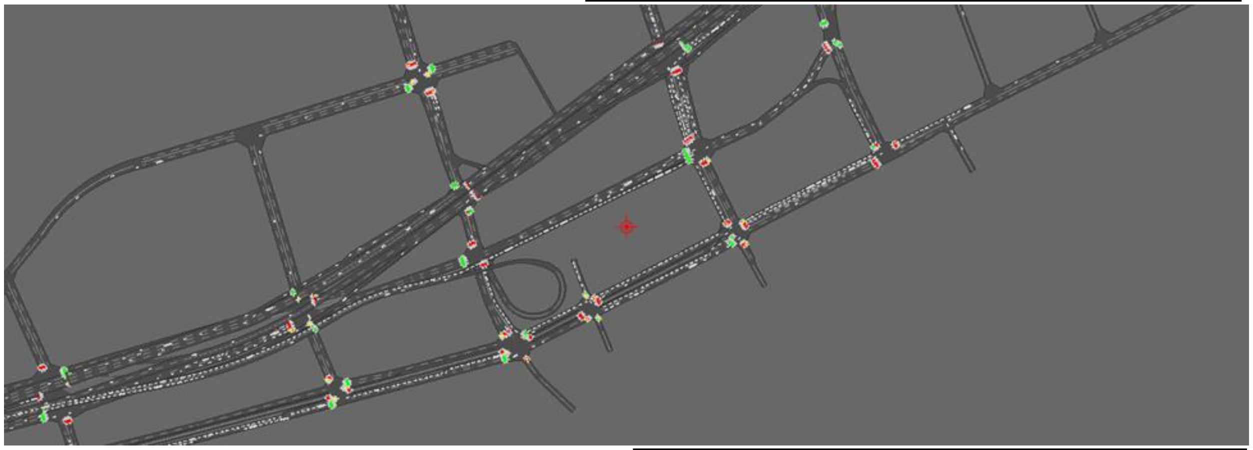
The analysis of Figure 7-24 and Figure 7-25 leads to the following conclusions:

- In the MARL-TI case most of the savings occur, interestingly, at chronic intersections, such as Lake Shore & Yonge Sts., Lake Shore & Bay Sts., York & Wellington Sts., York & Front Sts., University & Wellington Sts., and intersections along the access points to the downtown area from Lakeshore West;
- Similar to MARL-TI, in MARLIN-IC most of the savings are found at the busiest downtown intersections, but what is interesting to note is the formation of a "corridor of savings" in the MARLIN-IC case as shown in the dotted shaded areas in Figure 7-24. In the case of unfamiliar drivers the role of ATSC becomes more paramount due to the lack of any traveller information system, therefore congestion can quickly propagate and cause grid-lock. This is clearly shown by the footprint of the area control. In essence, most of the intersections in the downtown core (areas 1–4) warrant ATSC with coordination between agents (MARLIN-IC);

- In Figure 7-24, the histogram of percentage savings for different control systems is shown. It is found that the majority of savings (> 60%) achieved by MARLIN-IC over BC are within the defined “area of control” intersection. Comparing MARL-TI to MARLIN-IC, it is interesting to find that the majority of savings (0–10%, 10–20%, 20–30% and 30–40%) are actually for the same “area of control” noted above, which confirms that the additional “average” 40% outperformance (in average delay) of MARLIN-IC to MARL-TI – reported in Table 7-5 – are mostly concentrated and attributed to these areas of the network;
- Figure 7-25 shows the spatial distribution of the percentage savings in MARLIN-IC over MARL-TI under a variable profile. It is found that intersections along Lakeshore and York Sts. and Spadina and Queens Quay Sts. contributed to the majority of the savings (40–60%) achieved by the coordination effect;
- EB Traffic approaching the downtown core can either exit at the Spadina St. off-ramp or the York St./Yonge St. off-ramp. In the absence of TIS on traffic conditions downstream, if one off-ramp is congested due to limited to downstream capacity, travellers would still choose that route resulting in severe congestion. Figure 7-26 (a) shows the congestion build up along the Gardiner Expressway due to the limited capacity downstream at Spadina St. Similarly Figure 7-26 (b) shows the congestion build up on York St. off-ramp due to the grid-lock conditions observed downstream of the ramp. In high driver familiarity percentages, if the Spadina St. off-ramp is congested travellers would seek the next off-ramp (i.e. York St.) and *vice versa*. In such cases MARLIN-IC outperforms the BC and MARL-TI as it endeavours to prevent grid-lock conditions downstream of the off-ramps by properly coordinating actions with the neighbouring intersections;
- The findings in the normal and variable scenarios indicate that Bremner St. and Blue-Jays St. are candidate alternate routes to Spadina St. in cases where long queues are formed in the NB direction. However in the case of low familiarity, drivers are not aware of those options and continue in their predefined route (in this case Spadina NB and then EB in Font St. West), exacerbating the situation upstream until reaching the Gardiner Expressway off-ramp (see Figure 7-26 (a)).



(a) Spadina St off-ramp and Congestion Build Up



(b) York St off-ramp and Congestion Build Up

Figure 7-26 Snapshots to Illustrate the Performance of MARLIN-IC vs MARL-TI in the Unfamilarity Scenario

7.4 SUMMARY

Chapter 7 demonstrated the essence of MARLIN-ATSC on a large-scale urban network of 59 intersections, and built on the lessons learned from the prototype implementations described in Chapters 5 and 6. A MARLIN-ATSC large-scale application was conducted to examine two cases: 1) corridor-specific agent coordination, and 2) network-wide agent coordination. In the corridor experiments, the LS Blvd was selected as the testbed arterial as it is one of the most important corridors in downtown Toronto. In the network-wide experiment, the lower downtown of the Toronto network of 59 signalised intersections was selected as the testbed. In the network-wide coordination experiments, three scenarios were investigated: uniform demand profile, variable demand profile, and unfamiliar drivers (using a low percentage of familiar drivers). In each of the three scenarios above, the results were reported for BC control systems from the field (simulated using signal timing sheets provided by the City of Toronto, MARL-TI (represents MARLIN-ATSC Independent Mode with no communication between agents), and MARLIN-IC (represents MARLIN-ATSC Integrated Mode with coordination between agents). In the arterial experiment, MARLIN-IC generally outperformed the BC in terms of average stop time, average travel time, and Std of travel time. In the normal scenario of network-wide coordination experiments, MARL-TI outperformed the BC in all the MOEs. However, comparing MARLIN-IC to MARL-TI it was found that the latter experienced relatively higher delays. Compared to the uniform profile case, substantial deterioration in the performance of BC in the variable profile scenario was observed. Although MARL-TI also outperformed the BC in all the MOEs in the variable profile scenario, comparing MARLIN-IC to MARL-TI it is found that the latter produced higher delays because in MARL-TI the actions are based on locally sensed states with no coordination with other agents. Thereby MARL-TI resulted in more vehicles being retained in the network at the end of the simulation. In the unfamiliar drivers scenario a large deterioration in the performance of BC was observed. The two MARLIN-ATSC algorithms substantially outperformed the BC in all the MOEs. However, comparing MARLIN-IC to MARL-TI it was found that the latter produced higher delays and congestion. It was found that the deterioration in the performance of the BC in the low drivers familiarity case is attributed to the fact the unfamiliar drivers are less likely to switch routes in cases where their routes to the destination became congested, which created oversaturation conditions in some areas of the network and hence also deteriorated the performance of MARL-TI. MARLIN-ATSC was able to identify the best candidate intersections to receive adaptive control which would help municipalities prioritise the roll-out of MARLIN-ATSC with coordination (MARLIN-IC) vs MARL-TI, starting with most beneficial intersections. Such ability to quantify benefits prior to costly deployment and to prioritise investment is paramount, especially in light of the frequently tight budgets of today.

8 SUMMARY, CONCLUSIONS AND FUTURE RESEARCH

The problem of coordinated ATSC is a challenging due to the exponential growth in the number of joint timing plans to be explored as the network size grows. This thesis focused on the development of self-learning-based approaches to coordinate the behaviour of control agents in a multi-agent traffic control system with an emphasis on decentralised and scalable methods for large-scale urban traffic networks. This research investigated the application of MARL and GT to ATSC. A novel Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC) system was designed, developed and evaluated at the following levels: isolated intersection, a prototype network of 5 intersections, an arterial, and a large-scale urban network.

The core concept of MARLIN-ATSC is that each traffic signal controller is represented by an intelligent software agent (at each signalised intersection) that interacts with its environment (e.g. traffic network) in a closed-loop system. It also interacts with neighbouring control agents at surrounding intersections. The agent iteratively observes the *state* of the environment (e.g. queue lengths), takes an *action* accordingly (e.g. switches to another phase), and receives a feedback *reward* signal (e.g. delay reduction) for the actions taken. The agent adjusts the control policy until it converges to the optimal mapping from states to optimal actions (optimal *policy*) that maximises the cumulative reward (e.g. total delay). Accumulating the maximum reward not only requires the traffic signal control agent to reinforce the best-experienced actions, but also explores new actions to discover better action selections in the future.

Achieving coordination between agents in GT is proven to be unfeasible for a large number of players (agents) because the state-action space increases exponentially and the learning speed decreases dramatically with the number of agents in the system, which is known as the curse of dimensionality problem. MARLIN-ATSC achieves coordination-based decentralised control without suffering from the curse of dimensionality. Each agent plays a game with all its adjacent intersections in its neighbourhood, in which the agent not only learns its control policy but also considers the policies of the neighbours and acts accordingly.

The MARLIN-ATSC platform is developed to allow the investigation of different coordination levels and different RL design parameters in the ATSC problem (*MARLIN-ATSC Platform*). In MARLIN-ATSC, agents can implement one of the following two control modes:

- *Independent Mode:* In this mode each controller has an RL agent working independently of the other agents using Multi-Agent Reinforcement Learning for Independent controllers (MARL-I), two MARL-I are developed in the MARLIN-ATSC platform:
    - MARL-TI: Multi-Agent Reinforcement Learning for Totally Independent controllers;
    - MARL-PI: Multi-Agent Reinforcement Learning for Partially Independent controllers.
- *Integrated Mode:* In this mode, each controller coordinates the signal control actions with the neighbouring controllers using either MARLIN-IC or MARLIN-DC:
    - MARLIN-IC: Multi-Agent Reinforcement Learning for Integrated Network of Indirect Coordinated controllers;
    - MARLIN-DC: Multi-Agent Reinforcement Learning for Integrated Network of Direct Coordinated controllers.

The platform consists of two main layers; the first layer is an input configuration layer that is responsible for configuring and providing the necessary inputs to the second layer. The second layer is the control layer that includes three interacting components:

- ***Agent***: The agent is the learner and the decision-maker that implements the control algorithms and interacts with the environment by first receiving the system's state and reward and then selecting an action accordingly;
- ***Agent-Environment Interface***: The interface component facilitates the interaction between the agent and the traffic environment by exchanging the state, reward, and action;
- ***Traffic Environment***: The simulation-based environment component models the traffic environment in Paramics and executes the actions, monitors vehicles, constructs the states, and calculates the rewards.

8.1 SUMMARY

Chapter 1 of the dissertation started with a brief description of the ATSC problem. It also highlighted the major limitations of existing approaches and outlined the motivation and research objectives. Chapter 1 also provided a high-level description of the proposed framework.

Chapter 2 provided an overview of state-of-the-practice and state-of-the-art approaches in the literature on traffic signal control. It summarised the major challenges and gaps in the existing state-of-the-practice literature. It also highlighted the history and trend of the adaptive signal control research and development. In Chapter 2, the state-of-the-art studies that applied RL and MARL in traffic control applications were reviewed and their limitations were highlighted.

Chapter 3 reviewed the theoretical foundation of the stochastic control problem, which represents the foundation of the ATSC problem. Chapter 3 reviewed the problem from single agent and multi-agent perspectives. It provided a brief description of the theoretical framework, solution approach, optimal methods, and RL/MARL methods for solving the stochastic control problem in both single agent and multi-agent settings. The challenges and limitations of the MARL approaches were discussed in Chapter 3.

Chapter 4 discussed the development of the MARLIN-ATSC platform, while highlighting how it addresses the limitations/gaps in the traffic and control literature discussed in Chapters 2 and 3. It started with providing a mathematical formulation for the MARLIN control system. Then it provided a high-level description of the framework components.

Chapters 5, 6, and 7 demonstrated the applicability and feasibility of the approaches presented in Chapter 4 in a set of simulation-based experiments. Chapter 5 focused on identifying the best set of design parameters of the MARLIN-ATSC. The experimental setup and performance details of MARLIN-ATSC Independent Mode (RL-ATSC) were presented through a prototype implementation on a simulation of an isolated intersection in downtown Toronto. More specifically, the chapter comprehensively investigated the following parameters/dimensions: 1) exploration methods, 2) RL learning methods, 3) traffic state representations, 4) traffic signal phasing schemes, 5) reward definitions, and 6) variability in traffic flow arrival patterns to the intersection.

Chapter 6 presented the applicability of the two modes of operations of MARLIN-ATSC (i.e. Independent Mode (MARL methods) and Integrated Mode (MARLIN methods)) in a prototype network of 5 intersections. This network featured all the components of a multi-agent control problem while being concise enough to allow for the proper interpretation of the results. This prototype showed a proof-of-concept of MARLIN-ATSC controlling a network with multiple intersections. It did not, however, represent a large-scale implementation of the system and hence could not offer insights into the scalability of the system in large networks

The application of MARLIN-ATSC in a large-scale urban network that included 59 intersections in downtown Toronto was presented in Chapter 7. The large-scale application comprehensively demonstrated the essence of the proposed approach and built on the lessons learned from the prototype implementations described in Chapters 5 and 6.

The remainder of this chapter summarises the major findings of the experimental testbed networks and the main contributions of this thesis. This chapter ends with proposing areas for future research to further improve the MARLIN-ATSC system.

8.2 MAJOR FINDINGS

The analysis of the isolated intersection experiments (refer to Chapter 5) led to the following findings:

- In general, the results showed that the MARLIN-ATSC system consistently outperformed the optimised fixed-time as well as the traffic-responsive actuated control approaches. Savings in cumulative delay per vehicle are in the range of 35% and 25% compared to fixed-time and actuated controls, respectively;
- It was shown that the MARLIN-ATSC system performed more robustly than the fixed-time and actuated controls because it attained the same average delay regardless of the variations in arrival patterns (i.e. variable demand profile);
- The improvements achieved by the MARLIN-ATSC system were more apparent in the variable demand profile scenarios, resulting in 52% and 35% saving in average delay compared to the fixed-time and actuated controls, respectively, which reflects its better adaptability to fluctuations in traffic conditions;
- The results showed that synergising the ε-greedy and softmax exploration methods using the ε-softmax method allowed for faster convergence and better performance;

- In terms of the RL methods, although it was found that the Q-learning approach performs at least as well as SARSA in terms of average delay per vehicle, using Q-learning improves the convergence speed to the optimal policy;
- Although it was found that eligibility traces improve the learning speed for TD(λ), the higher the value of λ the worse the performance of the TD(λ) methods;
- Although it was found that RL generally outperforms the fixed-time control regardless of the state definition, considering the queue lengths at the red phases and the arrivals to the green phase led to the best state representations across all experiments;
- It was found that VPS is better than FPS, especially in cases of higher variability in the arrival pattern, as it substantially outperformed the FPS;
- The best reward function was found to be the reduction in the cumulative delay as it resulted in the minimum average delay, average queue length, and average number of stops when compared to the other reward functions.

The findings of Chapter 5 furnished important information for designing the parameters of MARLIN-ATSC in Chapters 6 and 7. Chapter 6 extended the research to investigate the MARLIN-ATSC platform with different coordination levels: Independent Mode (MARL-TI and MARL-PI) and Integrated Mode (MARLIN-IC and MARLIN-DC). To replicate realistic traffic conditions in the prototype implementation, two demand levels are modelled. One represented the actual observed demand, while the other represented a 50% increase in total demand.

The performance of various MARLIN-ATSC modes were tested on a prototype network of 5 intersections and compared against the fixed-time and traffic-responsive actuated controls (refer to Chapter 6). The analysis of these experiments leads to the following findings:

- In the actual demand case, MARLIN-ATSC Integrated Mode (MARLIN-IC and MARLIN-DC) resulted in slightly better performance compared to the Independent Mode (MARL-TI and MARL-PI). The savings ranged from 2% to 8% in network-wide average delay reduction and from 0% to 1% in total vehicle throughput. The modest improvement from coordination are attributed to the fact that demand levels did not cause queue spillbacks from one intersection to the next, i.e. MARL was found to be sufficient in cases where congestion was not severe;
- In the high demand case, in contrast to the above, MARLIN-ATSC Integrated Mode approaches resulted in better performance compared to the Independent Mode. The

savings ranged from 9% to 11% in network-wide average delay and from 2% to 4% in total vehicle throughput;

- Investigating MARLIN-ATSC Independent Modes (MARL) in the high demand level confirmed the need for considering the states of the neighbouring intersections, as MARL-PI approaches outperformed the totally independent operation (MARL-TI) with 4% savings in network-wide average delay and 2% improvement in total vehicle throughput;
- MARLIN-IC methods exhibited high savings in route travel times when compared to MARL-TI that ranged from 2% to 48% for different routes with an average of 20%;
- MARLIN-IC also showed substantial savings in travel time variations when compared to MARL-TI that ranged from 0% to 85% for different routes with an average of 40%. This implies that MARLIN-ATSC Integrated Mode can provide more reliable travel times, a performance measure that is of equal (if not more) importance to travel time savings;
- It was found that MARLIN-ATSC Integrated Mode is essential in the cases of oversaturation when spillback occurs from one intersection to the upstream intersections;
- Agents implementing MARLIN-ATSC Independent Mode struggled to converge under oversaturated conditions due to the violation of the stationary property of the environment associated with the multi-agent learning problem;
- In oversaturated conditions, MARLIN-ATSC Integrated Mode was found to be successful in “metering” traffic to critical intersections. This metering effect resulted in lower green time being allocated for some phases of the upstream intersections (18% in MARLIN-IC vs 26% in MARL-TI, i.e. MARLIN-IC allows 8% less green time for upstream intersections); but interestingly with higher overall vehicle throughput (2573 in MARLIN-IC vs 2379 in MARL-TI, i.e. MARLIN-IC allows 8% more vehicles to leave the network). Metering is an established practice in signal control but is often conducted manually by an experienced operator. MARLIN-IC not only automates this metering process but also achieves optimality;
- The effect of coordination in MARLIN-IC and MARLIN-DC was more noticeable in intersections with more than two phases (e.g. with advanced protected LT phases) compared to the typical two phase intersections (e.g. North/South, East/West) because agents are optimising both the phasing sequence and the split times for each phase;

- MARL-TI and MARL-PI methods unsurprisingly require less communication and computational resources, but they are not as efficient as MARLIN-IC and MARLIN-DC;
- The indirect coordination mechanism (MARLIN-IC) is recommended over the direct coordination mechanism (MARLIN-DC) as it achieves similar performance (average delay and route travel times) but with faster convergence speed and less computation time;
- In the actual demand case. MARLIN methods considerably outperforms fixed-time and actuated control by improving average delay (by 21% and 19%, respectively), throughput (1.2% and 0.3%, respectively), stop time (11% and 8%, respectively), number of stops (2% and 1%, respectively), average queue length (13% and 12%, respectively), link travel time (13% and 11%, respectively), and CO2 emissions (1.5% and 0.4%, respectively);
- The effectiveness of the MARLIN-ATSC was more evident in high traffic demand cases, which confirmed that coordination is more effective under highly-saturated conditions. Compared to fixed-time and traffic-responsive actuated controls, MARLIN-ATSC improved average delay (by 48% and 40%, respectively), throughput (13% and 11%, respectively), stop time (40% and 37%, respectively), number of stops (4% and 2%, respectively), average queue length (22% and 16%, respectively), link travel time (41% and 34%, respectively) and CO2 emissions factor (22% and 19%, respectively);
- It was also significant that considerable savings in variations in queue length across approaches (average Stds of queue length) were achieved, which reflected the ability of MARLIN-ATSC to equalise queue lengths across approaches compared to the fixed-time and actuated controls by 48% and 39%, respectively.

Chapter 7 demonstrated the essence of MARLIN-ATSC on a large-scale urban network of 59 intersections, and built on the lessons learned from the prototype implementations described in Chapters 5 and 6. The MARLIN-ATSC large-scale application was conducted to examine two cases: 1) corridor-specific agent coordination, and 2) network-wide agent coordination. In the corridor experiments LS Blvd was selected as the testbed arterial as it is one of the most important corridors in downtown Toronto. In the network-wide experiment, the lower downtown of Toronto network of 59 signalised intersections was selected as a testbed (refer to Chapter 7). In the network-wide coordination experiments three scenarios were investigated: uniform demand profile, variable demand profile, and unfamiliar drivers (using a low percentage of familiar drivers). In each of the three scenarios above the results were reported for BC control systems from the field (simulated using signal timing sheets provided by the City of Toronto as discussed in Section 7.2.1), MARL-TI (represents MARLIN-ATSC Independent Mode with no communication between agents), and MARLIN-IC (represents MARLIN-ATSC Integrated Mode with coordination between agents). The analysis of these experiments led to the following findings:

- In the arterial experiment, MARLIN-IC generally outperformed the BC by around by 40%, 28%, 63% in average stop time, average travel time, and Std of travel time, respectively;
- In the normal scenario of network-wide coordination experiments:
    - MARLIN-ATSC algorithms resulted in lower average delay, throughput, queue length, and stop time compared to those from the BC. The most notable improvements over the BC were reductions in average delay (38%), Std of average queue length (31%), and CO2 emissions (30%);
    - MARL-TI outperformed the BC in all the MOEs in the normal scenario, most notably were the reductions in the average intersection delay (27%) and the C02 emissions (28%). However, in comparing MARLIN-IC to MARL-TI it was found that the latter experienced relatively higher delays because in MARL-TI the actions are only based on local states, thereby resulting in more vehicles being retained in the network at the end of the simulation (6% throughout improvement in MARLIN-IC vs 2.8% throughput improvement in MARL-TI);
    - Compared to the uniform profile case, substantial deterioration in the performance of BC in the variable profile scenario were observed (24% increase in average delay, 25% decrease in throughput, 39% increase in average queue length, and 34% increase in Std of average queue lengths);
- In the variable profile scenario:
    - The two MARLIN-ATSC algorithms (MARLIN-IC and MARL-TI) considerably outperformed the BC in all the MOEs. The most notable improvements over the BC were the reductions in average delay (47%), increase in throughput (33%),

reduction in the average queue length (56%), and Std of average queue length (51%);

    - o Although, MARL-TI also outperformed the BC in all the MOEs in the variable profile scenario, comparing MARLIN-IC to MARL-TI it is found that the latter produced higher delays, because in MARL-TI the actions are based on locally sensed states with no coordination with other agents; thereby MARL-TI resulted in more vehicles being retained in the network at the end of the simulation (14% throughout improvement in MARLIN-IC vs MARL-TI). It was found that MARL-TI exhibited higher average queue length values than MARLIN-IC (37%) and less queue lengths balancing across the approaches (30%);

- In the unfamiliar drivers scenario:
    - o A large deterioration in the performance of BC was observed (16% increase in average delay, 34% increase in the average queue length, and 28% increase in Std of queue lengths);
    - o The two MARLIN-ATSC algorithms substantially outperformed the BC in all the MOEs. However, comparing MARLIN-IC to MARL-TI it was found that the latter produced higher delays and congestion (64% throughout improvement in MARLIN-IC vs MARL-TI);
    - o It was found that the deterioration in the performance of the BC in the low driver familiarity case is attributed to the fact the unfamiliar drivers are less likely to switch routes in cases where their routes to destination became congested, which created oversaturation conditions in some areas of the network, and hence also deteriorated the performance of MARL-TI;
- MARLIN-ATSC was able to identify the best candidate intersections to receive adaptive control which would help municipalities prioritise the roll-out of MARLIN-ATSC with coordination (MARLIN-IC) vs MARL-TI, starting with most beneficial intersections. Such ability to quantify benefits prior to costly deployment and to prioritise investment is paramount, especially in light of present tight budgets.

8.3 SUMMARY OF CONTRIBUTIONS

The research presented in this thesis represents important contributions in many areas as outlined in the following points:

1. The development of a new taxonomy of current approaches and studies in traffic signal control using RL techniques;
2. The introduction of a comprehensive critical review of the theoretical background of RL, both single agent and multi-agent perspectives. This review crystallised the connection between MDP, DP, and RL and the extension to the Markov game, GT, and MARL;
3. Comprehensive investigation of key design parameters in RL-based signal control for isolated intersections by offering a rigorous analysis of the effect of the following:
    a. exploration method,
    b. learning method,
    c. traffic signal phasing scheme,
    d. traffic state representation,
    e. reward definition,
    f. variability of flow arrivals to the intersection;
4. A novel approach for a decentralised and coordinated adaptive real-time traffic signal control system named MARLIN-ATSC. The combination of coordinated MARL-based methods, locality of interaction principle, and modular Q-learning approach, which represents the methodological contribution in this research, allows MARLIN-ATSC to achieve coordination without suffering from the curse of dimensionality. This approach offers the following capabilities compared to the existing ATSC systems employed in the literature (both state-of-the-practice and state-of-the-art):
    a. Lower Capital Cost: MARLIN-ATSC requirements can be satisfied using inexpensive communication networks, such as wireless networks between intersections, which considerably reduces the capital cost of the MARLIN-ATSC system compared to other ATSC systems such as SCOOT.
    b. Lower Operation Cost: MARLIN-ATSC has three features that help reduce the operating cost:
        i. Scalability: due to the decentralisation nature, MARLIN-ATSC is scalable and can be easily expanded by inserting new controllers into the system.

   ii. Robustness: robustness is guaranteed in a decentralised control system, because if one or more controller fails the remaining controllers are not affected.

   iii. Reduced Human Intervention: self-learning concepts reduce the complexity of the system and as a result the traffic operators will focus more on high-level monitoring and supervisory roles instead of the low-level management of traffic systems operations. Maintaining and retaining highly-skilled staff to operate systems such as SCOOT has been reported to be a significant issue for municipalities. Self learning systems alleviate this issue.

c. It maintains a coordination mechanism between agents without compromising the dimensionality of the problem.
d. It achieves coordination, not only along arterials, but also coordinates the operation of intersections in two-dimensional road networks (e.g. grid networks) by implementing mode 2 (integrated mode). To the best of my knowledge, agent coordination in two-dimensional networks is new.
e. In addition to a reduced delay at each intersection, MARLIN-ATSC was found to be intelligent enough to automatically protect critical intersections by "holding back" or "metering" approaching traffic at upstream intersections to prevent progressive network gridlock.
f. It lessens the curse of dimensionality problems associated with the traditional DP and RL methods when applied to a large number of agents.
g. MARLIN is theoretically designed to control any type of connected network of agents to suit a wide range of applications beyond traffic signals. Untested examples include:
   i. <u>Freeway Control</u>: to enhance the freeway performance by intelligently controlling the freeways on-ramps, speed, and changeable message signs.
   ii. <u>Wireless Networks Control</u>: to improve the performance of wireless networks by intelligently assigning users to the network's access points (APs).
   iii. <u>Hydro Power Generation Control</u>: to make the best use of available water resources by intelligently controlling the amount of water released from reservoirs and the amount of energy traded.

iv. Wind Energy Control: to balance the load frequency in interconnected networks of wind turbines.

v. Voltage Control: to provide a desirable voltage profile in a network of voltage controller devices;

5. The concept of locality of interaction was introduced into the direct coordinated-based algorithm by Yagan and Tham (2007). However, to the best of my knowledge, MARLIN-IC is the first attempt to use an indirect coordinated model-free MARL to solve a distributed stochastic control problem, while considering the locality of the interaction of agents;
6. The unprecedented introduction of a MARL system with indirect coordination (MARLIN-IC) for traffic signal control application;
7. In depth investigation into the effect of the following coordination strategies in a multi-agent system on a simulated real-work network of 5 intersections:
    a. totally independent agents,
    b. partially independent agents,
    c. direct coordinated agents,
    d. indirect coordinated agents;
8. Large-scale application of MARLIN-ATSC in a simulation of a real-world network of 59 intersections in downtown Toronto with the following variations:
    a. different arrival patterns,
    b. different driver familiarity levels;
9. MARLIN-ATSC provides a basis for understanding the characteristics and performance of a traffic network under agent-based coordinated ATSC;
10. Utilising state-of-the-art microscopic Paramics modelling as a simulation testebd that offers the following capabilities: dynamic and stochastic traffic assignment, Application Programming Interface (API), and strong visualisation capabilities. Most studies in the literature did not use a comprehensive microscopic simulation model, and therefore most of these features were largely missing in most of these studies. Self-developed simulation platforms give users more control over the simulation process, but the simulated traffic environment may not be as close to the true traffic conditions as those from comprehensive and well-calibrated simulation tools such as Paramics;

11. A comprehensive assessment of MARLIN-ATSC against the most commonly-used control methods in practice: fixed-time and traffic responsive actuated controls. We also attempted to approximate the operation of different traffic control systems in downtown Toronto, including some intersections under SCOOT.

8.4 FUTURE RESEARCH

Although very encouraging results were obtained in this research and many questions were answered in the development and testing stages of MARLIN-ATSC, more research questions have been triggered after completing this thesis. Abundant opportunities exist to further extend and enhance the MARLIN-ATSC. The following are suggestions for future research:

1. **Transit Signal Priority Integration and Pedestrian Consideration**: In this research the transit vehicles are treated as normal vehicles, and pedestrians consideration was only included in the calculation for the minimum green times to ensure adequate time for pedestrians to cross the intersection. However it is necessary to conduct future research that explicitly takes transit vehicles and pedestrians into consideration in the optimisation/control logic. Transit Signal Priority (TSP) is a common practice that assigns intersection right of way to approaching transit vehicles. The underlying philosophy of TSP is based on giving priority to high occupancy vehicles including transit vehicles, as such vehicles carry large numbers of travellers. Rightfully so, TSP gives priority to people throughput. However, if a transit vehicle is not full or not behind schedule then TSP may be counterproductive as it creates traffic delays without benefiting people throughput. Therefore the proper integration of ATSC and TSP is a logical next step for MARLIN to achieve the synergy of both techniques. In order to integrate the transit priority logic within MARLIN-ATSC, the state vector for each intersection should be extended to include components to reflect the transit state at the intersection. Instead of minimising the traditional reward for each signalised intersection which is the average traffic delay, the reward for each intersection should combine traffic and transit delays. To accommodate transit, MARLIN-ATSC should be trained to minimise delay for vehicles, weighted by the number of passengers in each. In other words, MARLIN-ATSC would ultimately target minimising total passengers delay regardless of the vehicle type approaching the intersection. This will however require Automatic Passenger Counters on transit vehicles and wireless communication to the traffic controller to register the approaching number of passengers. Pedestrians can be

accommodated in MARLIN-ATSC by including an exclusive right-turn phase if the pedestrian demand is extremely high (>1700) or by including a scramble phase if the pedestrian demand warrants such an installation. To implement the above within MARLIN-ATSC a pedestrian actuation button should be modelled to receive calls from pedestrians and included as one of the parameters in MARLIN-ATSC. Newer technologies are also emerging that count the number of pedestrians waiting to cross. With such technologies in place, priority could be assigned to pedestrians depending on pedestrian demand levels.

If the number of vehicles approaching an intersection and their occupancy are known and if the number of pedestrians can also be measured, MARLIN can be adjusted to optimise "people throughput" or minimise "people's delay" regardless of whether they are in automobiles, on buses or on foot;

2. **Comparison to SCOOT:** To quantify the benefits of MARLIN-ATSC relative to existing ATSC systems such as SCOOT, the following approaches could be used:
    - **a.** Measure SCOOT's Performance in the Field: compare the simulation-based measures of MARLIN with the real-life observations and benefits of SCOOT for SCOOT-controlled intersections.
    - **b.** Replicate the Logic of SCOOT: For instance, if SCOOT is to be evaluated and compared to MARLIN using microsimulation, a version of the SCOOT software is needed which is not readily available, and the underlying logic is proprietary. An alternative is to use *hardware-in-the-loop simulation (HILS) methodologies*. HILS is one of the most advanced forms of microscopic simulation for traffic signal control systems that has been experimented with in many cities including Virginia and Idaho[6]. HILS physically links a simulation software to an actual traffic controller, in which an actual traffic controller operates a traffic signal within the simulation software such as Paramics, VISSIM or similar. Figure 8-1 shows a schematic of the HILS configuration;

[6] http://cts.virginia.edu/facilities_tol.htm

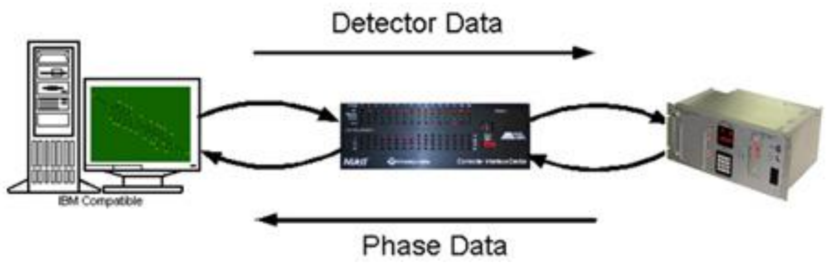


Figure 8-1 Hardware in the Loop System Testbed (HILS) (source: University of Virginia, 2012)

3. **Continuous State-Action Space and Function Approximation RL:** In this thesis MARLIN can be considered as an extension of Q-learning, which in its conventional form uses a Q-table to represent discrete Q values. This approach limits the applicability of Q-learning to more complex problems with a large number of states. This issue becomes more complicated in problems with continuous state space. In these problems there is always a trade-off between system performance and learning time, because a finer discretization will result in a better performance but at the cost of intractable state space and a longer learning time. One approach to overcome this limitation is to use a general function approximator instead of the discretized table. Although Neural Networks (NN) have been used in many RL implementations of real-world problems, there have been reports of divergence and sub-optimality of the these methods (Gosavi, 2003), in addition to the typical challenge of designing the parameters of the NN (or similar approaches). Another class of algorithms that can incorporate the continuous state space representation in Q-learning can be achieved through the use of local regression methods;
4. **Multi-Objective Reward Function:** A new reward function can take into account multiple factors such as delay, queue lengths, and number of stops;
5. **Partially Observable Markov Decision Processes (POMDP):** In this thesis a number of simplifying assumptions – that essentially makes the problem "easier" than it would be in practice – are made. First, the environment is assumed to be fully observable. In practice, this would hardly be the case as noisy inputs from sensors are expected as well as noisy communication among agents. Therefore these methods can be extended to deal with cases where only partial observations are available or when parts of the state are not visible to the agents. POMDP is an MDP with belief states; in POMDP the agent estimates and maintains a probability distribution function of the agent belief about the currently not observable states.

The distribution the agent maintains can be considered as the state of the decision process itself and as a result the process is again a MDP;

6. **Unexpected Traffic Conditions:** in order to minimise learning time in the field and increase the reliability of the system, the agent can learn a number of control policies for different traffic situations (e.g. incidents, special events, constructions, etc.) and allow the controller to occasionally choose the policy that is most suitable for the conditions at hand. This could be plausibly achieved by training the agent using data obtained from previous roadway closures, incidents, and special events;
7. **Extending the Neighbourhood Limits:** In this thesis we assumed the dependencies among agents to be fixed, i.e. a dependence relationship is assumed to be only between agents that are directly and physically connected (i.e. connected by a roadway segment). It should be noted that for certain problems or certain cases, different dependencies may lead to better solutions. Therefore, a possible extension of the current implementation is to find the optimal set of dependencies for different problems/cases;
8. **Further Investigation of Field Deployment Needs:** To ensure the smooth and successful deployment of MARLIN-ATSC we recommend the following investigations:
    - **a.** A review of operating and maintaining the communication infrastructure currently employed by municipalities/cities,
    - **b.** To scan and assess the feasibility and compatibility of the current controllers, cabinet needs, and functional requirements of the MARLIN-ATSC system considering the industry standards NEMA TS2 controller and the NTCIP communications protocol;
9. **Distributed Control Simulation through Parallel Computing:** Another possible extension is to implement a distributed version of the MARLIN-ATSC algorithm. The current version of MARLIN-ATSC, as used in this thesis, is maintained by a single computer station and operates using iterations. In order to simulate the decentralised operation of field controllers, a distributed implementation of MARLIN-ATSC could be achieved using high-performance computing clusters, however this implementation necessitates the development of functionalities and protocols not presently supported by the Paramisc simulator;
10. **Integrated Freeway Control with Urban Street Control:** Although the MARLIN control system is designed as a generic tool it is only tested in this thesis for signal control. In

addition to surface streets' signalised intersections, other promising control tools include Ramp Metering (RM) to control the entrance flow to the freeway, Variable Message Signs (VMS) to divert traffic at bifurcation points and Variable Speed Limit (VSL) control. Synergising the aforementioned strategies in one platform is the ultimate challenge and goal to alleviate traffic gridlock in metropolitan areas such as the GTA, and optimally utilise the existing system capacity. Due to the effect of freeway operations on surface streets and *vice versa*, future work should be conducted to implement the MARLIN control system on a network that includes RM, VMS, and VSL control, in which these elements are modelled as interacting agents with the signalised intersections agents. However, the proper choice for the RL-design parameters for each type of control is not an easy task.

In conclusion, the novel MARLIN-ATSC framework developed in this thesis outperforms prior methods and techniques by introducing a more robust and scalable solution, especially for highly-saturated, diverse, and increasingly larger networks. Therefore, this thesis offers a major contribution to the state-of-the-art and state-of-the-practice traffic signal control. The resulting system is also promising for other control applications beyond ATSC.

REFERENCES

Abdulhai, B. and L. Kattan (2003). Reinforcement learning: Introduction to theory and potential for transport applications. *Canadian Journal of Civil Engineering* 30(6): 981–991.

Abdulhai, B., R. Pringle and G. J. Karakoulas (2003). Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering* 129(3): 278–285.

Andrews, C. M., S. M. Elahi and J. E. Clark (1997). *Evaluation of New Jersey Route 18 OPAC/MIST Traffic-Control System*. Transportation Research Board, Washington D.C.

Arel, I., C. Liu, T. Urbanik and A. G. Kohls (2010). Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transport Systems* 4(2): 128–135.

Armstrong, M. D., J. Marchand, J. R. Rhodes, A. MacNab, P. V. Godfrey and S. Cass (1974). I*mproved Operation of Urban Transportation Systems, Volume 1: Traffic Signal Control Strategies – A State-of-the-Art*. Ministry of Transport, Canada.

Bakker, B., M. Steingrover, R. Schouten, E. Nijhuis and L. Kester (2005). Cooperative multi-agent reinforcement learning of traffic lights. *Workshop on Cooperative Multi-Agent Learning, European Conference on Machine Learning (ECML'05).*

Bakker, B., S. Whiteson, L. Kester and F. C. A. Groen (2010). *Traffic Light Control by Multiagent Reinforcement Learning Systems.* Interactive Collaborative Information Systems. Springer, Berlin, Germany, pp. 475–510.

Barriere, J. F., J. L. Farges and J. J. Henry (1986). Decentralization vs. hierarchy in optimal traffic control. *Proceedings of the 5th IFAC/IFIP/IFORS Symposium on Control in Transportation Systems*.

Barth, M., F. An and J. Norbeck (1996). Modal emissions modeling: A physical approach. *Transportation Research Record: Journal of the Transportation Research Board* 1520: 81–88.

Basar, T. and G. J. Olsder (1999). *Dynamic Noncooperative Game Theory*. Classics in Applied Mathematics. London, U.K.

Bazzan, A. L. C. (2009). Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. *Autonomous Agents and Multi-Agent Systems* 3(18): 342–375.

Bell, M. C. and R. D. Bretherton (1986). Ageing of fixed-time traffic signal plans. *IEE 2nd International Conference on Road Traffic Control*.

Bertsekas, D. P. (1976). *Dynamic Programming and Stochastic Control*. Academic Press, New York.

Bertsekas, D. P. (2007). *Dynamic Programming and Optimal Control*. Athena Scientific.

Boillot, F., S. Midenet and J.-C. Pierrelée (2006). The real-time urban traffic control system CRONOS: Algorithm and experiments. *Transportation Research Part C: Emerging Technologies* 14: 18–38.

Boutilier, C. (1996). Planning, learning and coordination in multiagent decision processes. *6th Conference on Theoretical Aspects of Rationality and Knowledge.*

Brown, G. W. (1951). Iterative solutions of games by fictitious play. In: T. C. Koopmans (ed.) *Activity Analysis of Production and Allocation*. Wiley, New York, pp. 374–376.

Buckholz, J. P. (2007). *The Real-Time Estimation of Delay at Signalized Intersections*. University of North Florida, Jacksonville, Florida.

Busoniu, L., R. Babuska and B. De Schutter (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man and Cybernetics* 38(2): 156–172.

Camponogara, E. and W. Kraus Jr. (2003). Distributed learning agents in urban traffic control. *11th Portuguese Conference on Artificial Intelligence.*

Chen, B. and H. H. Cheng (2010). A review of the applications of agent technology in traffic and transportation systems. *IEEE Transactions on Intelligent Transportation Systems* 11(2): 485–497.

City of Toronto (2012). Changes to Front Street at Union Station. http://www.toronto.ca/involved/projects/frontunion/pdf/front-st-ea-pic2-all-comp4-sec-orient.pdf, Retrieved 20 April 2012.

Claus, C. and C. Boutilier (1998). The dynamics of reinforcement learning in cooperative multiagent systems. *15th National Conference on Artificial Intelligence and 10th Conference on Innovative Applications of Artificial Intelligence, Madison, US.*

Daskalakis, C., P. W. Goldberg and C. H. Papadimitrio (2006). The complexity of computing a Nash equilibrium. *38th Annual ACM Symposium on Theory of Computing.*

Dayan, P. and T. J. Sejnowski (1994). TD (lamda) converges with probability 1. *Machine Learning* 14(3): 295–301.

De Oliveira, D., A. L. C. Bazzan, B. C. da Silva, E. W. Basso, L. Nunes, R. Rossetti, E. de Oliveira, R. da Silva and L. Lamb (2006). Reinforcement Learning-based Control of Traffic Lights in Non-stationary Environments: A Case Study in a Microscopic Simulator. *Fourth European Workshop on Multi-Agent Systems, Lisbon, Portugal* (*EUMAS06).*

de Queiroz, M. S., R. C. de Berredo and A. de Pádua Braga (2006). Reinforcement learning of a simple control task using the spike response model. *Neurocomputing* 70(1–3): 14–20.

El-Tantawy, S. and B. Abdulhai (2010). Towards multi-agent reinforcement learning for integrated network of optimal traffic controllers (MARLIN-OTC). *Transportation Letters: The International Journal of Transportation Research* 2(2): 89–110.

Ethier, S. N. and T. G. Kurtz (1986). *Markov Processes: Characterization and Convergence.* Wiley, New York.

Farges, J. L., J. J. Henry and J. Tufal (1983). The PRODYN real-time traffic algorithm. *4th IFAC/IFIP/IFORS Symposium on Control in Transportation Systems, Baden-Baden, Germany.*

Farges, J. L., I. Khoudour and J. B. Lesort (1990). PRODYN: On site evaluation. *3rd International Conference on Road Traffic Control, London, England.*

FHWA (2004). *Traffic Analysis Toolbox Volume III: Guidelines for Applying Traffic Microsimulation Software, Report FHWA-HRT-04-040*. http://ops.fhwa.dot.gov/trafficanalysistools/tat_vol3/Vol3_Guidelines.pdf, Retrieved 28 September 2012.

FHWA (2005a). *Traffic Control Systems Handbook: Chapter 7. Local Controllers*. http://www.ops.fhwa.dot.gov/publications/fhwahop06006/chapter_7.htm, Retrieved 20 April 2012.

FHWA (2005b). *Traffic Control Systems Handbook: Chapter 8. System Control*. http://www.ops.fhwa.dot.gov/publications/fhwahop06006/chapter_8.htm, Retrieved 28 September 2012.

Filar, J. and K. Vrieze (1997). *Competitive Markov Decision Processes*. Springer-Verlag, New York.

Friedrich, B., I. Matschke, E. Almasri and J. Mück (2003). Data fusion techniques for adaptive traffic signal control. *10th IFAC Symposium on Control in Transportation Systems, Tokyo, Japan.*

Gartner, N. H. (1983). OPAC: A demand-responsive strategy for traffic signal control. *Transportation Research Record: Journal of the Transportation Research Board* 906: 75–81.

Gartner, N. H., S. F. Assmann, F. Lasaga and D. L. Hout (1990). MULTIBAND - A variable bandwidth arterial progression scheme. *Transportation Research Record* 1287: 212–222.

Gartner, N. H., S. F. Assmann, F. Lasaga and D. L. Hout (1991). A multi-band approach to arterial traffic signal optimization. *Transportation Research Part B* 25: 55–74.

Gartner, N. H., C. J. Messer and A. K. Rathi (1992). *Traffic Flow Theory: A State of the Art Report – Revised Monograph on Traffic Flow Theory*. Transportation Research Board, Washington, D.C.

Gartner, N. H., F. J. Pooran and C. M. Andrews (2001). Implementation of the OPAC adaptive control strategy in a traffic signal network. *Proceedings of IEEE Intelligent Transportation Systems Conference*.

Gartner, N. H., C. Stamatiadis and P. J. Tarnoff (1995). Development of advanced traffic signal control strategies for intelligent transportation systems: Multilevel design. *Transportation Research Record* 1494: 98–105.

Gartner, N. H., C. Stamatiadis and P. J. Tarnoff (1996). Development of advanced traffic signal control strategies for intelligent transportation systems: Multilevel design. *Transportation Research Record* 1494: 98–105.

Gartner, N. H., P. J. Tarnoff and C. M. Andrews (1999). Evaluation of the optimized policies for adaptive signal control strategy. *Transportation Research Record* 1683: 105–114.

Gosavi, A. (2003). *Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning*. Springer, Netherlands.

Hale, D. K. (2006). *Traffic Network Study Tool*, TRANSYT-7F, United States Version. Mc-Trans Center, University of Florida, Gamesville, Florida, pp. 32611–6585.

Head, K. L., P. B. Mirchandani and D. Sheppard (1992). Hierarchical framework for real-time traffic control. *Transportation Research Record* 1360: 82–88.

Henry, J. J. (1989). PRODYN tests and future experiments on ZELT. *Vehicle Navigation and Information Systems, IEEE Conference, Toronto (VNIS '89).*

Henry, J. J. and J. L. Farges (1989). PRODYN. *6th IFAC/IFIP/IFORS Symposium on Control, Computers, and Communications in Transportation, Paris, France.*

Hunt, P. B., D. I. Robertson, R. D. Bretherton and R. I. Winton (1981). *SCOOT - A traffic responsive method of coordinating signals*. Technical Report. Transport and Road Research Laboratory, Crowthorne, England.

Isa et al. (2006)

Jacob, C. (2005). *Optimal, Integrated and Adaptive Traffic Corridor Control: A Machine Learning Approach*. Department of Civil Engineering, University of Toronto, Toronto.

Jang, J. S. R., C. T. Sun and E. Mizutani (1997). *Neuro-Fuzzy and Soft Computing*. Prentice Hall, Upper Saddle River.

Kaelbling, L. P., M. L. Littman and A. W. Moore (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence* 4(1): 237–285.

Kalai, E. and E. Lehrer (1993). Rational learning leads to Nash equilibrium. *Econometrica* 61(5): 1019–1045.

Kok J. R. and Vlassis, N. (2005). Using the max-plus algorithm for multiagent decision making in coordination graphs. In RoboCup-2005: Robot Soccer World Cup IX, Osaka, Japan.

Kok, J. R., M. T. J. Spaan and N. Vlassis (2005). Non-communicative multirobot coordination in dynamic environment. *Robotics and Autonomous Systems* 50(2/3): 99–114.

Koller, D. and A. Pfeffer (1997). Representations and solutions for game theoretic problems. *Artificial Intelligence* 94(1): 167–215.

Kuyer, L., S. Whiteson, B. Bakker and N. Vlassis (2008). Multiagent reinforcement learning for urban traffic control using coordination graph. *19th European Conference on Machine Learning.*

Leng, J., C. Fyfe and L. C. Jain (2009). Experimental analysis on Sarsa(λ) and Q(λ) with different eligibility traces strategies. *Journal of Intelligent and Fuzzy Systems* 20(1): 73–82.

Little, J. D. C., M. D. Kelson and N. H. Gartner (1981). MAXBAND: A program for setting signals on arteries and triangular networks. *Transportation Research Record* 796: 40–46.

Littman, M. (1994). Markov games as a framework for multi-agent reinforcement learning. *11th International Conference on Machine Learning.*

Lu, S., X. Liu and S. Dai (2008). Incremental multistep Q-learning for adaptive traffic signal control based on delay minimization strategy. *7th World Congress on Intelligent Control and Automation.*

MacGowan, J. and I. J. Fullerton (1979). UTES: Development and testing of advanced control strategies in the urban traffic control system. *Public Roads* 43**:** 97–105.

Mauro, V. and C. Di Taranto (1989a). UTOPIA. *2nd IFAC-IFIP-IFORS Symposium on Traffic Control and Transportation Systems.*

Mauro, V. and C. Di Taranto (1989b). UTOPIA. *6th IFAC/IFIP/IFORS Symposium on Control, Computers, and Communications in Transportation.*

McShane, W. R., R. P. Roess and E. S. Prassas (1998). *Traffic Engineering*. Prentice Hall.

Medina, J. C. and R. F. Benekohal (2012). Q-learning and approximate dynamic programming for traffic control – A case study for an oversaturated network. *Transportation Research Board Annual Meeting.*

Mendelson, E. (2004). *Introducing Game Theory and Its Applications.* CRC Press.

Metrolinx (2008). *The Big Move: Transforming Transportation in the Greater Toronto and Hamilton Area*. Metrolinx, Toronto.

Mirchandani, P. and L. Head (2001). A real-time traffic signal control system: Architecture, algorithms and analysis. *Transporation Research Part C* 9: 415–432.

Mirchandani, P. and D. E. Lucas (2001). *RHODES - ITMS TEMPE FIELD Test Project: Implementation and Field Testing of RHODES - A Real-Time Traffic Adaptive Control System*. FHWA-AZ01-447. Federal Highway Administration, Washington, D.C.

Nair, R., P. Varakantham, M. Tambe and M. Yokoo (2005). Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. *20th National Conference on Artificial Intelligence.*

Nash, S. and A. Sofer (1996). *Linear and Nonlinear Programming*. McGraw-Hill, New York.

NCHRP (2010). *NCHRP Synthesis 403: Adaptive Traffic Control Systems: Domestic and Foreign - State of Practice*. Transportation Research Board, Washington D.C.

Ono, N. and K. Fukumoto (1996). Multi-agent reinforcement learning: A modular approach. *Second International Conference on Multi-Agent Systems.*

Osborne, M. J. (2004). *An Introduction to Game Theory.* Oxford University Press, New York.

Papageorgiou, M. (2003). Traffic control. In: R. W. Hall (ed.) *Handbook of Transportation Science*. Kluwer Academic Publishers, Dordrecht, pp. 243–277.

Parzen, E. (1967). *Stochastic Processes*. Holden-Day.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, New York.

Quadstone Paramics (2012). *Paramics Microscopic Traffic Simulation Software*. http://www.paramics-online.com.

Richter, S., D. Aberdeen and J. Yu (2007). Natural actor-critic for road traffic optimisation. *Advances in Neural Information Processing Systems* 19.

Robertson, D. I. (1969). *TRANSYT: A Traffic Network Study Tool*. Report LR 253. Road Research Laboratory.

Roberston, D. I. (1979). Traffic models and optimum strategies of control: A review. Institute of Transportation Studies and US Department of Transportation. *Proceedings of International Symposium on Traffic Control Systems* 1: 262–288.

Salkham, A., R. Cunningham, A. Garg and V. Cahill (2008). A collaborative reinforcement learning approach to urban traffic control optimization. *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*.

Sen, S. and K. L. Head (1997). Controlled optimization of phases at an intersection. *Transportation Science* 31: 5–17.

Sen, S., M. Sekaran and J. Hale (1994). Learning to coordinate without sharing information. *12th National Conference on Artificial Intelligence.*

Shapley, L. S. (1953). Stochastic Games. *Proceedings of the National Academy of Sciences of the United States of America* 39(10): 1095–1100.

Shoham, Y., R. Powers and T. Grenager (2003). *Multi-agent reinforcement learning: A critical survey*. Technical Report. Computer Science Department, Stanford University, California, US.

Shoufeng, L., L. Ximin and D. Shiqiang (2008). Q-learning for adaptive traffic signal control based on delay minimization strategy. *IEEE International Conference on Networking, Sensing and Control.*

Sims, A. G. and K. W. Dobinson (1979). SCAT - The Sydney co-ordinated adaptive traffic system - Philosophy and benefits. *International Symposium on Traffic Control Systems.*

Spaan, M. T., J. N. Vlassis and F. C. A. Groen (2002). High level coordination of agents based on multiagent Markov decision processes with roles. *IEEE/RSJ International Conference Intelligent Robots Systems.*

Sutton, R. S. and A. G. Barto (1998). *Introduction to Reinforcement Learning*. MIT Press Cambridge, MA.

Texas Transportation Institute (1998). *PASSER^{TM}III-98 Application and User's Guide.* Texas Transportation Institute.

Thorpe, T. (1997). *Vehicle Traffic Light Control Using SARSA*. Masters Project Report. Computer Science Department, Colorado State University, Fort Collins, Colorado.

Tian, Z. and M. Abbas (2007). Models for quantitative assessment of video detection system impacts on signalized intersection operations. *Transportation Research Record* 2035: 50–58.

Trafficware (2012). *Signal Timing*. http://www.trafficware.com/assets/pdfs/Signal%20Timing%20Background.pdf, Retrieved 28 September 2012.

Trafficware (2012). *Synchro: Real-Time Adaptive Control System.* http://synchrogreen.com, Retrieved 25 March 2012.

University of Virginia (2012). *Traffic Operations Laboratory, Center for Transportation Studies*. http://cts.virginia.edu/facilities_tol.htm, Retrieved 20 April 2012.

Venglar, S., P. Koonce and T. Urbanik II (2000). *PASSER V*. Texas Transportation Institute.

von Neumann, J. and O. Morgenstern (1944). *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ.

Wahba, M. (2008). *MILATRAS: MIcrosimulation Learning-based Approach to TRansit ASsignment*. Department of Civil Engineering, University of Toronto, Toronto.

Wang, X. and T. Sandholm (2002). Reinforcement learning to play an optimal Nash equilibrium in team Markov games. *Advances in Neural Information Processing Systems.*

Watkins, C. and P. Dayan (1992). Q-learning. *Machine learning* 8(3): 279–292.

Webster, F. V. (1958). *Traffic Signal Settings. Road Research Technical Paper No 39*. HMSO, London.

Weinberg, M. and J. S. Rosenschein (2004). Best-response multiagent learning in non-stationary environments. *3rd International Joint Conference on Autonomous Agents and Multiagent Systems.*

Weiss, G. (1999). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence.* MIT Press.

Wiering, M. (2000). Multi-agent reinforcement learning for traffic light control. *17th International Conference on Machine Learning*

Wiering, M., J. Van Veenen, J. Vreeken and A. Koopman (2003). Intelligent traffic light control. *ERCIM News, European Research Consortium for Informatics and Mathematics* 53: 40–41.

Wood, K. (1993). *Urban Traffic Control - Systems Review.* Project Report 41. Transport Research Laboratory, Crowthorne, U.K.

Yagan, D. and C. Tham (2007). Coordinated reinforcement learning for decentralized optimal control. *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning.*

Zhang, L., H. Li and P. D. Prevedouros (2005). Signal control for oversaturated intersections using fuzzy logic. *84th Annual Meeting of the Transportation Research Board.*

APPENDIX 1

Webster Method

Webster developed the following procedure to calculate the optimum cycle time and splits such that the overall intersection delay is minimised:

1. A critical lane is first defined for each signal phase. A critical lane (or critical movement) is the lane with the highest ratio of flow to saturation flow (or flow ratio). Suppose that f_p is the critical flow ratio for phase p. By definition:

$$\boldsymbol{f_p = max_{l \in L_p}\left(\frac{q_l}{q_l^{sat}}\right)} \quad (A1\text{-}1)$$

where L_p is the set of lanes of phase p, and q_l and q_l^{sat} are the flow rate and saturation flow rate, respectively, for lane l.

2. The optimum cycle length, C_{opt}, can be approximated as:

$$\boldsymbol{C_{opt} = \frac{1.5L+5}{1-F}} \quad (A1\text{-}2)$$

where L is the total intersection lost time per cycle and F is the summation of all critical flow ratios corresponding to each of the P phases in the cycle:

$$\boldsymbol{F = \sum_{p=1}^{P} f_p} \quad (A1\text{-}3)$$

3. For each phase p, its optimum green time (split), g_p, is calculated by proportionally distributing the total available green time, i.e. $C_{opt} - L$, to its critical flow ratio:

$$\boldsymbol{g_p = \frac{f_p}{F}(C_{opt} - L)} \quad (A1\text{-}4)$$

APPENDIX 2

Policy Iteration

A sequence of policy evaluation and improving processes is called the *policy iteration* method. The state-value function for an arbitrary policy, π, V^{π}, is first computed. This is called *policy evaluation. Policy improvement* is the process of producing a new policy by taking the *greedy* action, the best action in the short term, according to V^{π}. Each policy is guaranteed to be a strict improvement over the previous policy unless it is already optimal. Since a finite MDP has only a finite number of policies, the policy iteration process must converge to an optimal policy π^* and an optimal value function V^{π^*} in a finite number of iterations. Policy evaluation uses the Bellman equation for policy π as expressed as follows in the discounted reward context:

$$V^{\pi}(s)=\sum_{s^{k+1}\in S} T(s,\pi(s),s^{k+1})(r(s,\pi(s),s^{k+1})+\gamma V^{\pi}(s^{k+1})) \qquad \forall\, s\in S \qquad \text{(A2-1)}$$

where γ is the discount factor and the improved policy is obtained as follows:

$$\pi^{k+1}(\mathrm{s}) \in \arg\max_{a\in A}[\sum_{s^{k+1}\in S} T(s,a,s^{k+1})(r(s,a,s^{k+1})+\gamma V^{\pi}(s^{k+1}))] \quad \forall\, s\in S \qquad \text{(A2-2)}$$

Example A2-1: Signalized Intersection Control Using Policy Iteration Dynamic Programming

This example shows the solution of Example 3.3 using DP Policy Iteration algorithm; assuming $\boldsymbol{\pi_0}\,(s_1)=a_1$ $\boldsymbol{\pi_0}\,(s_2)=a_1$

Iteration 0	$\boldsymbol{V^{\pi_k}(s) = \sum_{s^{k+1}\in S} T(s,a,s^{k+1})(r(s,a,s^{k+1})+\gamma V^{\pi_k}(s^{k+1}))}$ Start with $\boldsymbol{\pi_0\,(s_1)=a_1}$ $\boldsymbol{\pi_0\,(s_2)=a_1}$
Iteration 1	$\boldsymbol{V^{\pi_0}(s_1) = [-1*0.5-16*0.5] + 0.8\,[\,0.5\,V^{\pi_0}(s_1) + 0.5\,V^{\pi_0}(s_2)]}$ $\boldsymbol{V^{\pi_0}(s_2) = [-10*0.1-50*0.9] + 0.8\,[\,0.1\,V^{\pi_0}(s_1) + 0.9\,V^{\pi_0}(s_2)]}$ $\boldsymbol{\begin{bmatrix}0.6 & -0.6\\ 0.2 & -0.72\end{bmatrix}\begin{bmatrix}V^{\pi_0}(s_1)\\ V^{\pi_0}(s_2)\end{bmatrix} = \begin{bmatrix}-8.5\\ -46\end{bmatrix}}$ $\boldsymbol{\begin{bmatrix}V^{\pi_0}(s_1)\\ V^{\pi_0}(s_2)\end{bmatrix} = \begin{bmatrix}-153\\ -208\end{bmatrix}}$ $\boldsymbol{\pi^1(s_i) = \arg\max_{a\in A}\left[\sum_{s^{k+1}\in S} T(s,a,s^{k+1})(r(s,a,s^{k+1})+\gamma V^{\pi}(s^{k+1}))\right]}$ $\boldsymbol{\pi_1(s_1) = \arg\max_{u}[(-1*0.5-16*0.5)}$ $\boldsymbol{+0.8\,[0.5\,(-153) + 0.5\,(-208)], (-10*0.1-50*0.9)}$ $\boldsymbol{+0.8\,[0.1\,(-153) + 0.9\,(-208)]}$ $\boldsymbol{= \arg\max_{u}[(-153), (-140)]}$ $\boldsymbol{\pi^1(s_1) = a_2}$ Similarly

	$\boldsymbol{\pi^1(s_2) = a_2}$
Iteration 2	Similarly $\boldsymbol{\pi^2(s_1) = a_1}$ $\boldsymbol{\pi^2(s_2) = a_2}$
Iteration 3	Similarly $\boldsymbol{\pi^3(s_1) = a_1}$ $\boldsymbol{\pi^3(s_2) = a_2}$

In order to balance the queue lengths in this intersection, the following control policy has to be followed:

- If the queue length on the minor street is "low" (s_1), the optimal action is to extend the green time of the major street phase for the next time interval (e.g. 30 sec) (a_1)
- If queue length on the minor street is "high" (s_2), the optimal action is to switch to the minor street phase for the corresponding minimum green time (e.g. 15 sec) and then turn on the green again to the major street for the corresponding minimum green time (e.g. 15 sec) (a_2).

Computational Complexity

As shown in Examples 3.4 and A2-1 the value iteration method requires 20 iterations to converge, and in each iteration two quantities need to be calculated (*V(s)* one for each state), and in each quantity two values need to be calculated (one for each action), and the maximum value will be considered. Then, a comparison with the previous V(s) values should be performed. On the other hand, the policy iteration method requires only 3 iterations to converge. However, in each iteration, there is a 2x2 system of linear equations to be solved (where the dimensions represent the number of states and the actions, respectively).

To better illustrate the effect of the problem dimensions on the computational complexity of DP over RL, assume that the number of states is |S| and the number of actions is |A|.

As shown in the above example, the value iteration method required many more iterations to converge. However, in each iteration one value is calculated for each state (|S| computations). To calculate *V(s),* |A| values would be compared to find the maximum (|A| comparisons).

On the other hand, the policy iteration method required a fewer number of iterations to converge. However, in each iteration there is a system of linear equations that needs to be solved with the number of variables equal to |S| (solving a |S|x|S| system of equations in each iteration). In addition, in each iteration a comparison between |A| values should be performed to update the policy (|A| comparisons).

APPENDIX 3

Simplified Economic Analysis

The large-scale application of MARLIN-ATSC was conducted on a simulation testbed of 59 intersections in the lower downtown core of Toronto during the morning rush hour, resulting in 25,000 trips. MARLIN-ATSC was compared against the pre-timed and actuated timing plans provided by the City of Toronto. The daily economic benefits (i.e. travel time savings) were estimated to be around $53,000. MARLIN-ATSC would cost approximately $1.2 M to implement across a network of 59 intersections. Consequently the payback period is 23 days!

The numbers calculated above are based on the following assumptions:

- The daily economic benefit is calculated as follows: average delay per vehicle per hour (13 sec) x the demand per hour (25,000 veh) then summed for all hours in a day (considering a daily profile for the demand to proportionally estimate the savings during the off-peak and night hours) x $40 (estimated value of time/hour) = $ 53 K savings/day;

- Cost of implementing MARLIN is based on the estimated cost for the decentralized system installation ($10,000 to $30,000 per intersection, an average value of $20 K is used) and multiplied by 59 intersections = $1.2 M;

- Therefore the payback period is $1.2 M / $53 K = 23 days.

Appendix 4

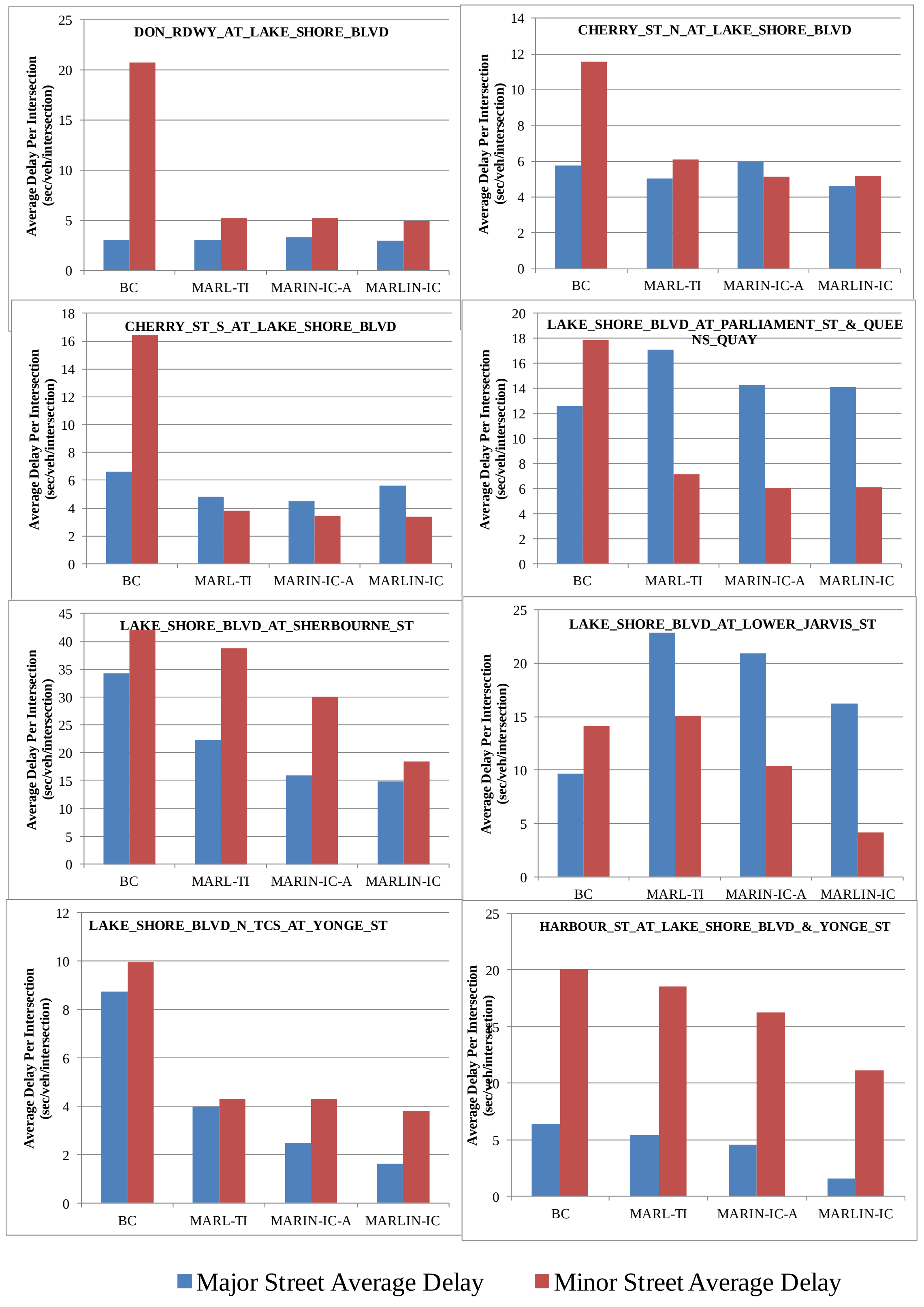


Figure 0-1 Comparison of Major and Minor Street Average Delay Per Intersection