TU WIEN Informatics

A collaborative multi-agent reinforcement learning approach to managing traffic light grids

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Data Science

eingereicht von

Mathias Halmetschlager, BSc
Matrikelnummer 01426370

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec. Schahram Dustdar
Mitwirkung: Univ.Ass. Pantelis Frangoudis, PhD

Wien, 17. August 2022

Mathias Halmetschlager	Schahram Dustdar

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

A collaborative multi-agent reinforcement learning approach to managing traffic light grids

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Data Science

by

Mathias Halmetschlager, BSc
Registration Number 01426370

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec. Schahram Dustdar
Assistance: Univ.Ass. Pantelis Frangoudis, PhD

Vienna, 17th August, 2022

Mathias Halmetschlager	Schahram Dustdar

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Mathias Halmetschlager, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 17. August 2022

Mathias Halmetschlager

Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die mich während der Verfassung dieser Masterarbeit unermüdlich unterstützt und motiviert haben. In erster Linie gebührt mein Dank Herrn Professor Schahram Dustdar und Herrn Pantelis Frangoudis für die Betreuung und Begutachtung der Arbeit und die angenehme Zusammenarbeit und ständige Bereitstellung konstruktiver Kritik. Ich danke auch meinem Partner von der EFS Unternehmensberatung GmbH, Herrn Roman Benedetto, der mir durch seine Flexibilität und die Bereitstellung einer unterstützenden und fördernden Arbeitsumgebung den kreativen Freiraum für die Verfassung dieser Arbeit ermöglichte. Ein spezieller Dank gilt auch dem Christchurch City Council Traffic Signals Team, für das Bereitstellen der Verkehrs- und Ampeldaten, welche diese Arbeit bereichert haben. Meinen Freunden Wilma Weixelbaum und Marcel Anderl danke ich für die ständige emotionale Unterstützung über die Dauer meines gesamten Studiums und besonders in der Zeit in der ich diese Arbeit verfasste. Abschließend möchte ich mich bei meinen Eltern bedanken, die mir mein Studium durch ihre Unterstützung ermöglicht haben.

Acknowledgements

At this point I would like to thank all those who tirelessly supported and motivated me during the writing of this master's thesis. First and foremost, I would like to thank Professor Schahram Dustdar and Mr. Pantelis Frangoudis for supervising and assessing the work and for the pleasant cooperation and constant provision of constructive criticism. I would also like to thank my partner at EFS Unternehmensberatung GmbH, Mr. Roman Benedetto, who gave me the creative freedom to write this thesis through his flexibility and the provision of a supportive and nurturing working environment. Special thanks also go to the Christchurch City Council Traffic Signals Team for providing the traffic and traffic light data that enriched this work. I would also like to thank my friends Wilma Weixelbaum and Marcel Anderl for their constant emotional support throughout my studies and especially during the time I spent writing this thesis. Finally, I would like to thank my parents who made my studies possible with their support.

Kurzfassung

Verkehrsstaus werden mit der ständig wachsenden Weltbevölkerung und Urbanisierung zu einem immer schwieriger zu lösenden Problem. Infolgedessen sind ampelgesteuerte Kreuzungen ein wichtiger Brennpunkt, der die Effizienz des Verkehrsflusses im gesamten Straßennetz einer Stadt beeinträchtigen und schnell zu Staus führen kann. In den letzten Jahren hat sich gezeigt, dass Reinforcement Learning großartige Ergebnisse bei der Verbesserung des Verkehrsflusses erzielt, die in der Lage ist, optimale Ampelphasen in Echtzeit auszuwählen. Während diese Lösungen sehr vielversprechend sind, steht die Forschung mit ihrem aktuellen Stand vor einem von zwei Problemen. Die state-of-the-art-Lösungen konzentrieren sich entweder auf eine einzelne Kreuzung, die die Schwierigkeiten des Verkehrsmanagements nicht angemessen abbildet, oder sie versuchen, komplexere Systeme als Ganzes zu lösen, was keinen skalierbarer Ansatz für ganze Städte darstellt. Diese Arbeit schlägt eine skalierbare Deep-Q-Learning-basierte Lösung für intelligente Ampeln vor, die sich auf die Zusammenarbeit von Ampeln mit ihrer unmittelbaren Nachbarschaft konzentriert, wodurch die Komplexität der Zusammenarbeit begrenzt wird und gleichzeitig die Modellierung transitiver Effekte ermöglicht wird, die sich über mehrere Kreuzungen auswirken, wie zum Beispiel der green wave Effekt. Diese Dissertation evaluiert mehrere verschiedene Stufen der Zusammenarbeit und verschiedene Synchronisationsschemata zwischen Agenten und misst die Auswirkungen dieser Designentscheidung auf ein kollaboratives System. Es wird gezeigt, dass vielversprechende state-of-the-art-Lösungen, die im Rahmen einer einzelnen Kreuzung evaluiert wurden, in einem Systemen mit mehreren Kreuzungen nicht mit optimierten fixen Zeitintervallen konkurieren können, was die Wichtigkeit der Nutzung von zusätzlichen Informationen bestätigt, die durch Kollaboration bereitgestellt werden. Diese Arbeit zeigt auch, dass der kollaborative Ansatz zu einer signifikanten Verringerung der Wartezeit in einem Netz von fünf Kreuzungen im Vergleich zu einer nicht kollaborativen state-of-the-art-Alternative führt. Schließlich wird auch gezeigt, dass die vorgeschlagene Lösung mit einer adaptiven und optimierten realen Lösung innerhalb einer Simulation von drei Kreuzungen basierend auf realen Verkehrsdaten und den entsprechenden Ampelprotokollen des beobachteten Zeitrahmens konkurrieren und diese bei geringem bis mittlerem Verkehrsaufkommen übertreffen kann. Diese Ergebnisse bilden die Grundlage für ein skalierbares, kollaboratives System, welches in großen Verkerhsnetzen und sogar ganzen Städten eingesetzt werden kann.

Abstract

Traffic congestion is becoming an increasingly difficult problem to solve with the ever growing world population and urbanization. As a result, traffic light controlled intersections are an important focal point that can make or break efficiency of traffic flow throughout a cities road network and can quickly cause congestion. In recent years, reinforcement learning has been shown to produce great results in improvement of traffic flow by using fully adaptive agent based traffic light control capable of choosing optimal light phases in real time. While these solutions show great promise, current literature faces one of two problems. The state-of-the-art solutions focus either on a single intersection which does not adequately represent the difficulties of traffic management or they attempt to solve more complex systems as a whole which is not a scalable approach for entire cities. This work proposes a scalable deep Q-Learning based solution for smart traffic lights that focuses on collaboration of traffic lights with their immediate neighborhood, thus limiting the complexity of the collaboration while still allowing modeling of transitive effects that span multiple intersections such as the green wave effect. This thesis evaluates several different levels of collaboration and different synchronisation schemes between agents measuring the impact of these design decisions in a collaborative system. It is shown that promising state-of-the-art solutions that were evaluated using the scope of a single intersection fall behind highly optimized fixed time intervals in systems of multiple intersections, which confirms the importance of utilizing the additional information provided by collaboration. This work also shows that collaboration results in significant reduction of wait time in a grid of five intersections when compared to a non-collaborative state-of-the-art alternative. Lastly it is also shown that the proposed solution can compete with and for low to medium traffic outperform an adaptive and optimized real world solution within a simulation of three intersections based on real traffic data and the respective traffic light logs of the observed time frame. These findings lay the ground work for a scalable, collaborative system deployable throughout large scale traffic systems or even entire cities.

Contents

CHAPTER 1

Introduction

Efficient management of traffic has always been a problem especially in densely populated areas but with the ever increasing population all around the world it is becoming more and more important to improve the way traffic is managed in order to keep the traffic grids from overloading. A key component for keeping traffic throughout an entire city fluid is the management of intersections using traffic lights. For many years the approaches for improving traffic flow through intersections have been either time-consuming and costly traffic studies to determine the best green/red light intervals for a given time of day, or more recently, semi-actuated and actuated controls which use sensors to detect oncoming traffic and cars waiting in front of a red light. The problems with these two approaches are apparent. Fixed time intervals can never really adapt to changing traffic conditions dynamically and can only be tuned based on empirical measurements. Actuated controls lay an important foundation for further improvements but they leave a lot of information unused. While switching the light phases based on whether or not cars are waiting to pass is adaptive, it does not take the whole intersection into account and it also cannot decide how to efficiently handle the intervals if cars are waiting in both directions. The work of Qadri et al. [QGÖ20] has shown with a comprehensive state-of-the-art analysis of current trends and advancements in this field, that with the ever increasing computational power of hardware as well as increasing availability of real time data, predictive machine learning approaches are becoming more feasible and powerful, making these solutions a realistic candidate for the future of traffic signal control. Several modern solutions using fuzzy logic as shown by Alam and Pandey [AP15], population based metaheuristic algorithms as seen in the works of Fleck et al. [FCG16] and machine learning approaches to the likes of the solutions proposed by Vidali et al. [VCVB19] and Gao et al. [GSL+17] have been proposed that make use of these technological advancements with promising results. While these solutions work well and have been extensively trained and tested for single intersections, there is a lack of work done in applying the approaches on multiple intersections making it effective for entire traffic grids as a wholistic solution. This thesis

expands a promising state-of-the-art solution proposed by Vidali et al. [VCVB19] to more complex grids, evaluating its performance in these complex settings. Furthermore, this thesis adapts the discussed solution with the goal to provide a scalable concept for self learning traffic light grids. This is achieved by extending the scope of each traffic light to its immediate neighborhood and enabling collaboration between the machine learning agents. By restricting the collaboration of each traffic light agent to its immediate neighbors, the communication costs can be kept low and the total complexity of the system is kept at a minimum while providing benefit from the additional information received from neighboring agents, resulting in a scalable solution for large grids. This thesis explores both the capabilities and limitations of the proposed system by rigorous testing within an experimental setup using ***S**imulation **O**f **U**rban **M**obility*[1] or *SUMO* for short which is an open source, highly portable, microscopic and continuous multi-modal traffic simulation package designed to handle large networks. Further tests of the full solution and implementation scheme on a real world example using traffic data provided by the City Council of Christchurch New Zealand are also performed. Lastly, this thesis also discusses the requirements, challenges and obstacles of constructing this system in the real world.

1.1 Research Questions and Contributions

This thesis thus answers the following three research questions:

1. *Does the performance improvement of state-of-the-art reinforcement learning solutions for smart traffic lights proposed for single intersections hold for more complex traffic light grids with multiple intersections? Specifically is there a statistically significant drop in performance improvement over fixed time intervals?*

   This question serves the purpose of confirming the need to further improve state-of-the-art solutions that are limited to the scope of a single intersection for them to be competitive in more complex real world settings. This is done by first confirming the results of the non-collaborative solution proposed by Vidali et al.[VCVB19] and applying it to a more complex simulation setup of five intersections. The results are compared against fixed time intervals with comparable traffic load within the system to measure if the improvements still hold within a grid of multiple intersections.

2. *Does collaboration among agents in a grid of traffic lights lead to a significant improvement in either cumulative wait time for the entire cluster of intersections or the average cumulative wait time per vehicle over non-collaborative agents?*

   By answering this research question it is shown that the proposed approach for collaboration can lead to a significant improvement of the wait time within a system

[1] https://www.eclipse.org/sumo/

of intersections when compared to both fixed time intervals and a non-collaborative Q-Learning solution. This is shown both within an experimental setup of five connected intersections and a real world example of three intersections using real traffic data.

3. *Can the proposed collaborative approach achieve significant improvement over highly optimized real world intervals in terms of total wait time or mean vehicle wait time?*

   To answer this final research question, we measure the performance of the proposed solution in a real world setting against optimized timings instead of the experimental setup with five intersections. Thus, it is shown that the collaborative solution can compete with modern systems and further reduce wait times and congestion in urban traffic. This is done by using a simulation model of three real intersections in Christchurch NZ for which the Christchurch City Council Traffic Signals Team provided the exact logs of active traffic light phases throughout the day and the traffic counts during this time. This provides a solid baseline of a competitive real world application as these intersections do not use regular fixed time intervals but are instead managed and constantly optimized by the Sydney Coordinated Adaptive Traffic System or *SCATS*[2] for short.

1.2 Thesis Organization

This thesis is structured as follows. In Chapter 2 we provide an overview of the historic background of signal based traffic management and a closer look at the technological advancements of recent years. The chapter also highlights enabling technologies that are needed for a real life implementation of the discussed solution and goes into detail on how we can benefit from them. Lastly it summarizes the contributions of Vidali et al.[VCVB19] that lay the ground work upon which the proposed solution is built.
In Chapter 3 we provide a detailed overview on deep Q-Learning in general and within the context of smart traffic lights. Furthermore the chapter explains the design contributions and the concrete implementations, that enable collaboration and synchronisation among reinforcement learning agents within the system.
Chapter 4 explains the setup of the simulation framework that is used to fully train and also subsequently test and evaluate the collaborative solution. It provides information on how the road networks used in this thesis are designed and how traffic within these systems is generated both for the training phase as well as for evaluation to answer the underlying research questions.
In Chapter 5 we show the concrete results of the proposed system compared to a non-collaborative alternative and optimized fixed time intervals evaluated on the previously defined simulation setups. On top of answering the research questions, the rigorous testing of specific design decisions of the collaborative aspects and external variables, such as for instance varying road lengths and varying traffic distributions, provides further

[2] https://www.scats.nsw.gov.au/

insight on the strengths and weaknesses of the discussed solution.
To showcase the feasibility of a real life implementation of the proposed system, Chapter 6 discusses the required steps of modeling, training and testing a given set of intersections and additionally, the required technologies for actually building the system in the real world are discussed as well.
Lastly, Chapter 7 concludes the thesis by summarizing the research questions and their respective answers as shown by this thesis. Furthermore, all other contributions that were made are revisited again and the final section of the chapter discusses potential future work to further improve the concepts discussed here and to solidify the understanding of the field.

CHAPTER 2

Related Work

This chapter gives an overview of traffic light management throughout history and how technological advancements are beginning to change the way the problem of efficient traffic management is viewed and subsequently solved. Furthermore, Section 2.3 summarizes the concepts proposed by Vidali et al. [VCVB19] which lays the ground work for the solution proposed in this thesis.

2.1 Historic Summary

The worlds first traffic light was implemented in 1868 and it was essentially composed of two mobile signs that could be interchanged by the use of a lever and a gas-lit semaphore so the signs were visible during the night. A few months after its implementation the police officer who operated the traffic light died when this first prototype exploded and thus the world had to wait for another 52 years until the first electricity powered traffic light was installed in Cleveland US, marking the start of the technology spreading across the urban areas of the world. Finally on March 30th, 1931 the first Convention on the Unification of Road Signals was signed in Geneva, standardizing the common three colored traffic lights known today. Within this last century, humanity has a come a long way not only regarding urbanization and an increase in demand for road infrastructure that could not have been imagined in 1930 but also with the digital revolution allowing for automation and optimization of every single step in the process of traffic management. From vehicle detection using machine learning based image recognition for monitoring to actuated traffic lights responding to vehicles present in the intersection and adjusting their cycles accordingly the possibilities are endless. In spite of these possibilities being available, the actual implementation is lagging behind as a large part of traffic lights still operate on fixed time intervals just as they did almost a hundred years ago. These fixed times are in no way selected arbitrarily and a lot of traffic studies and calculations factor into them as shown by the work of Gorodokin et al. [G17] but the drawbacks for

these solutions are apparent. In many cases the fixed cycle lengths are changed based on empirical data gathered for specific days and time periods to optimize for different demands but they can never truly adapt to the actual situation. While rush hours might be fairly consistent on week days, an expected low traffic situation can quickly change if for instance some special event is taking place in a certain area resulting in a rapid increase of traffic flow where it was not expected. *SCOOT*[1] and *SCATS*[2] are traffic control systems that aim to optimize traffic flow in cities by constantly monitoring traffic flow and optimizing traffic light timings accordingly. As shown by systems such as *SCOOT* [BC95] in the UK and *SCATS* [SD80] in Australia the market has recognized this potential for a long time as these adaptive systems have been around 35 and 40 years respectively. Examples such as the city of Sydney which heavily relies on *SCATS* show that adaptive systems have the potential to greatly improve traffic flow. These systems operate on real time traffic data and are able to adapt to situations accordingly and thus provide a great framework for the implementation of promising work from recent literature which has shown that reinforcement learning and fuzzy logic have the potential to further improve traffic management. The state-of-the-art summary of Quadri et al. [QGÖ20] shows that research is still largely focused on meta / heuristic algorithms despite these advancements in different technologies but machine learning is quickly catching up in this regard as shown in Figure 2.1. The aforementioned state-of-the-art summary also highlights two problems with the current state of research in this field which is on one hand the focus on single intersections without considering the problem in the larger scale and on the other hand the problem of the initially large investment for development and maintenance of these solutions. The problem of economic feasibility can be countered by the aforementioned systems that are already in place and being maintained with similar infrastructure as would be required by many literature solutions. In regard to those issues this thesis proposes a relatively light weight, scalable solution that can build upon known technologies for monitoring traffic in an intersection such as the one discussed by Collotta et al. [CBP15] while also considering the agents impact on neighboring traffic lights and thus the whole grid.

2.2 Enabling Technologies

In order to provide a better understanding of promising approaches, problems faced and work done in the field this section summarizes a selection of different solutions highlighting the variety of approaches that can be taken to tackle this issue and also discuss how the insights from these works are beneficial to the solution discussed in this thesis. The two papers summarized here are the the work of Collotta et al. [CBP15] which proposes a solution using a wireless sensor network and fuzzy logic controllers to determine the optimal phase duration for a given situation in real time and the concept proposed by Dhingra et al. [DMP+21] which elaborates on a concrete fog computing solution for real time traffic monitoring and congestion detection. Additionally this

[1] https://trlsoftware.com/products/traffic-control/scoot/
[2] https://www.scats.nsw.gov.au/

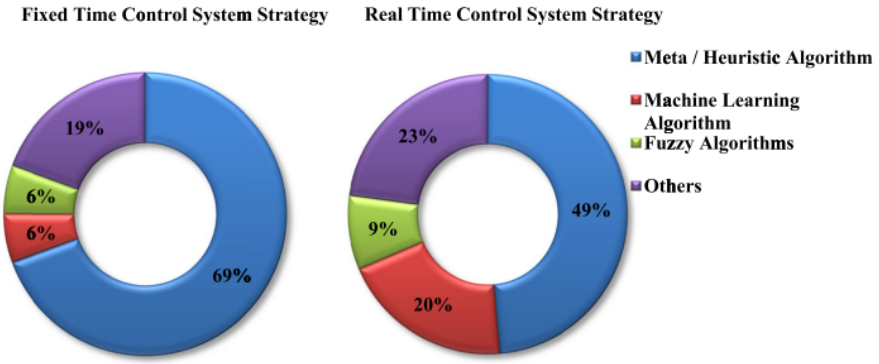

Figure 2.1: Current focus of research (2020, Quadri et al. [QGÖ20])

section goes over two promising solutions published very recently in 2020 and 2021 by Kumar et al. [KMGK21] and Zhou et al. [ZCL+20] and how these approaches differ from the collaborative solution proposed by this thesis.

Fuzzy Logic Controlled Traffic Lights

As mentioned above the traffic light management system proposed by Chavan et al. [CDR09] uses a technology which is widely used in research of traffic management which is fuzzy logic controllers. In essence these controllers can abstract input information to a more human-readable format which in this case is the conversion of numeric queue lengths to the categories of *normal*, *medium* and *long*, infer decisions within this domain and revert these abstract decisions back to a real world metric. This process is called fuzzification and defuzzification respectively. In this case the concept is utilized by using one fuzzy logic controller for each possible green phase which first categorizes the queue lengths for its respective lanes and then infers its own priority (i.e. how urgently its green phase needs to be executed) and the optimal length in seconds for its green phase. As an example the controller for north-south bound traffic would first calculate its phase priority as shown in Equation 2.1 then categorize the queue lengths and infer its green phase according to the function shown in Figure 2.2. Lastly a higher ranking phase selector determines which controller has the highest priority and selects the next phase to be executed.

$$phasepriority = queuelength_{northsouth} + queuelength_{southnorth} \tag{2.1}$$

Despite the simplistic approach the evaluation results have shown satisfying improvements over fixed intervals and also other single-controller fuzzy logic approaches. The main drawback of this solution is that the controllers which are working in a disjoint fashion cannot capture complex interactions and influences of other lanes in the intersection. Furthermore they cannot incorporate additional information from neighboring traffic lights and thus make future oriented decisions. While more complex solutions might outperform this approach the simplicity results in a very cost efficient and maintainable

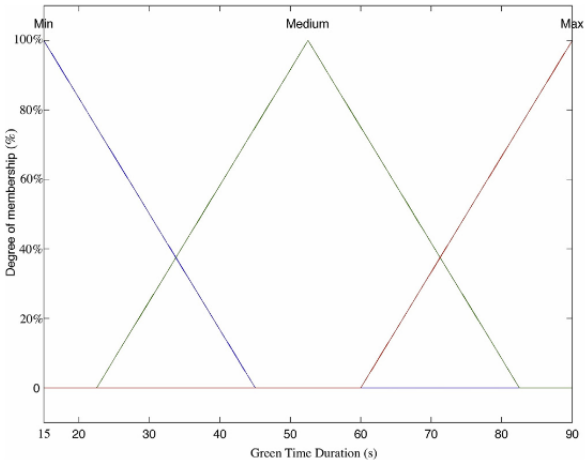

Figure 2.2: Fuzzylogic Green Phase Function (2015, Collotta et al. [CBP15])

system which is one of the most important driving factors for real world application. Chavan et al. [CDR09] propose a system of inexpensive wireless sensors for measuring the queue lengths in each lane which is the core of why this work is relevant for this thesis. While most reinforcement learning approaches work great within the simulation settings in which they are developed they often neglect the difficulty of gathering the data required to train and operate these models reliably. Traffic simulation frameworks like *SUMO* provide perfectly accurate real time data on each vehicle within the simulation at all times. In reality this information can be hard to acquire in sufficient quality which is why a system has to be designed around information that can be provided by the available technology. Figure 2.3 shows the concept of how these wireless sensors could be implemented in an intersection in order to provide the required information. As depicted the small ***R**educed **F**unction **D**evices* can detect the presence of cars and only communicate with their respective ***F**ull **F**unction **D**evice* which aggregates the received information and passes on the state of the entire lane to the traffic light controller. This intersects with the state representation used in the solution proposed by this thesis which requires precisely this information.

Fog Computing for Traffic Monitoring

Fog computing[YHQL15] is a well known, Internet of Things related concept which proposes the distribution of computing workload over smaller decentralised computing nodes instead of transmitting everything to one large server and computing it there generating potential bottlenecks. This concept is greatly relevant to the topic of large, complex traffic light grids that span entire cities. Transferring the traffic data of an entire city to a single computing center and computing decisions such as the next green phase of a specific traffic light there is infeasible especially if situation needs to be reevaluated on a second by second basis. Fog computing proposes an elegant solution to this by decentralizing the decision making process and dividing a given system into smaller

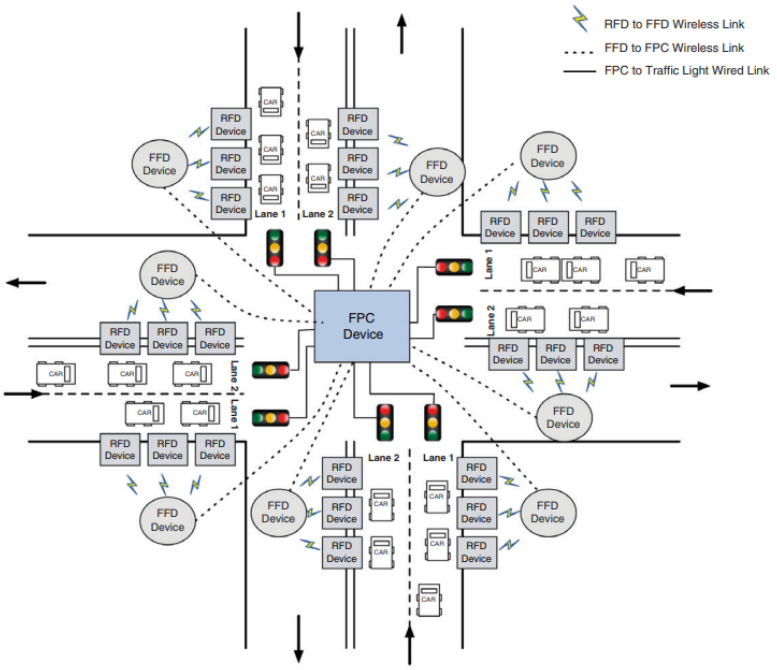

Figure 2.3: Wireless Sensor Network (2015, Collotta et al. [CBP15])

subsystems to the point where the communication costs are as low as possible while still providing all the necessary information for a given node. The work of Dhingra et al. [DMP+21] proposes a decentralized, scalable solution for traffic monitoring using ultrasonic sensors for the counting of vehicles at crucial points within a traffic network and utilizes fog computing to compute these traffic counts in real time with low latency. This is highly relevant to the topic of this thesis as on one hand fog computing is a viable solution to splitting up the computational load in a system of traffic light agents that require only the additional information of their immediate neighborhood and on the other hand the implemented traffic detection using ultrasonic sensors poses a possible real world solution for acquiring the data required as state representation input for the deep learning agent discussed in Chapter 3 in a very cost efficient way.

State-of-the-Art Collaborative Approaches

The works of Kumar et al.[KMGK21], Zhou et al.[ZCL+20] and Chu et al.[CWCL19] are highly relevant to this thesis as they are three very recent papers published in 2022, 2021 and 2020 respectively on the topic of smart traffic light management using reinforcement learning. What makes them especially relevant is the fact that all three solutions extend the scope to multiple intersections and address the problem of collaboration and scalability which are the core contributions of this thesis. Thus, this subsection will provide a quick summary of the contributions of those three papers regarding these points and explain

how the proposed solution differs and why it provides important insight on top of these works. In essence the solution proposed by Kumar et al.[KMGK21] uses non-collaborative Q-Learning agents with a slightly more complex state space that not only observes vehicle positions but also their velocity. While this promises better results within a simulation framework, it makes real life application more difficult as these exact measurements need to be taken consistently for every decision step. The solution of Kumar et al. introduces collaboration by adding a congestion detection system on top of the non-collaborative agents which allows a form of collaboration by providing this congestion information to the agents in the system to which they can then react. The solution does not include any direct communication between agents or extension of the state space that allow agents to view more than the roads of their own intersection. Another important contribution of this work is the fact that the problem of heterogeneity of traffic is accounted for which is often not considered in model training and simulations of this kind. For this thesis we also simplified this aspect and all vehicles are assumed to be homogeneous. As such, Kumar et al. provides an important reference point for future work especially when considering this heterogeneity.

Zhuo et al.[ZCL+20] also propose a collaborative system that consists of multiple layers. Here the first layer which are single intersections operate on a threshold based algorithm that decides, given traffic parameters like halting vehicles and speed lag, whether to extend a given light phase or switch to the next phase in the cycle. Deep Q-Learning is introduced on the second level which oversees multiple intersections. This creates an important distinction to the solution proposed by us. Zhuo et al. proposes a solution in which the Q-Learning agent does not decide the optimal light phase per intersection but instead provides a vector which, for each intersection, contains the value 0 or 1 denoting whether to extend the given light phase or switch to the next phase. Finally there is another layer on top of this layer that groups several of this Q-Learning agents together and adjusts the learning rate based on the achieved results. While this is a form of collaborative system, it is an entirely different approach on an intersection basis, as a single intersection does not choose the best light phase for the given circumstances and instead prolongs certain phases within the cycle if necessary. It is also important to note that while this system scales linearly instead of exponentially, which would be the case if a single agent attempts to solve a system of multiple intersections, the solution proposed by Zhuo et al. follows a more modular approach that can be implemented intersection by intersection without ever facing a scaling problem beyond the immediate neighborhood. A core aspect that should be compared in future work is how well the system utilizes effects that span multiple intersections. The system of Zhuo et al. attempts to react to congestions as they arise and does so by applying the policy to a larger part of the grid while the solution proposed here attempts to carry these effects throughout the system from one intersection to the next as each agent communicates only with its immediate neighbor.

Out of the three discussed solutions the work of Chu et al. is closest to the concepts and contributions of this thesis. Chu et al. addresses the problem of real life feasibility of state representation by proposing a simple state which consists of the accumulative wait

time of only the first car in the lane combined with the total number of cars waiting in the given lane. While this is more complex than simply tracking occurrence of cars in the intersection, as proposed by our solution, it adds the additional waiting information without having to track the wait times of all vehicles. In terms of the reinforcement learning approach the solution of Chu et al. proposes collaborating agents, that share policy information among neighboring agents, similar to the solution proposed by this thesis. The main difference is that firstly Chu et al. proposes collaboration by also sharing the policy information of neighboring agents resulting in the agents policies influencing each other, whereas this thesis focuses on sharing only the chosen action and the available state information of neighboring agents, having each agent build their own policy. Secondly, as opposed to the evaluation of Chu et al. the emphasis of the research in this thesis is not solely focused on showing the efficiency of the resulting system but also on measuring the effect that collaboration has as a whole and how robust such a system is to varying traffic load and distances between intersections.
In conclusion it can be said that while all three solutions expand the scope to include collaboration in a system of multiple intersections the contributions of this thesis differs fundamentally from the works of Kumar et al. and Zhou et al. as it includes explicit communication of state information between agents, which allows the deep Q-Learning agents a broader view of the world around them, while still limiting complexity significantly. This is possible due to the simplicity of the state representation which results in low communication cost despite all of the relevant state information being shared among collaborating agents. The main difference to the work of Chu et al. is the fact that, in the solution proposed by this thesis, policies of neighboring agents do not influence each other and instead only share their observed environment and actions taken.

2.3 Deep Q-Learning Agent for Traffic Signal Control

This section summarizes the work of Vidali et al. [VCVB19] which provides the groundwork for this thesis and the concepts on which the discussed solution and experiment are built upon. Vidali et al. proposes a deep reinforcement learning approach which has become one of the most promising approaches for traffic light management in recent years as shown by a lot of research done in this field by [LLW16],[LDWH19],[GR16] and [GSL+17] to name a few recent publications. Among current research in the field of reinforcement learning for traffic light control the work of Vidali et al. stands out in the sense that it operates on a simple state space and does not require convolutional layers for a complex state representation and subsequently long and resource intensive training. Additionally these complex state representations require data in a quality that can often not be provided by current technology such as exact speed measurements of every car in the intersection at a given time. In essence, the core contributions of Vidali et al. are the following four points which are summarized in this section:

The state representation The algorithm is built on a binary representation of an intersection where cells are assigned to a road as shown in Figure 2.4. Each cell is either

assigned the value 1 if a car is present within its range or 0 if no car is present. Both Gao et al. [GSL+17] as well as Genders and Razavi [GR16] use a similar but decidedly more complex state representation which is shown in Figure 2.5. In the case of Gao et al. each cell is 7 meters long and each traffic lane consists of a separate row of cells. The simplified state representation proposed by Vidali et al. combines these cells for all lanes that allow straight crossing of the intersection and keeps the cells for the lane allowing for left turns only. This not only greatly reduces the complexity of the machine learning task but also decreases cost and complexity of a potential real world application as this is the information that has to be gathered for the fully trained algorithm to operate.

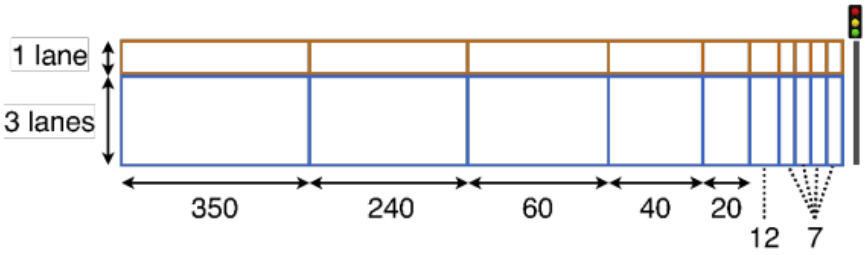

Figure 2.4: Design of the state representation (2019, Vidali et al. [VCVB19])

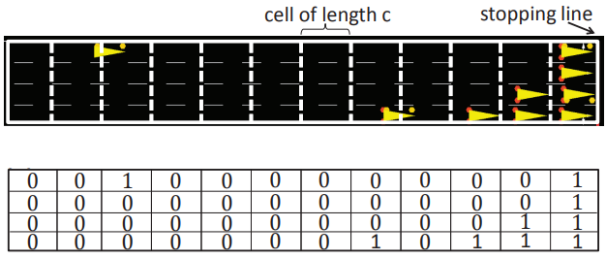

Figure 2.5: Design of the state representation (2017, Gao et al. [GSL+17])

The reward function The function proposed in the paper utilizes the cumulative wait time of all cars in the intersection as depicted in Equation 2.2.

$$atwt_t = \sum_{veh=1}^{n} awt_{(veh,t)} \tag{2.2}$$

awt denotes the wait time accumulated by vehicle veh at time step t in the given intersection. This metric is computed once for the time step at which a decision is made by the agent and once for the time step at which the agent makes its next decision. According to Equation 2.3 the reward is calculated with $atwt_t$ denoting the accumulated total wait time at timestep t.

$$r_t = atwt_{t-1} - atwt_t \tag{2.3}$$

Others such as Gao et al. [GSL+17] have also used this reward function with promising results and the work of Vidali et al. has shown the performance increase over using vehicle stop time which is defined as the time a vehicle has a speed below 0.1 m/s. The problem being that stop time does not take into account multiple stops in an intersection which can occur if the vehicle can not clear the intersection in a green cycle due to high traffic load.

The scheme of the deep neural network Another important contribution for this thesis is the scheme of the deep neural network proposed in the paper. As shown in Figure 2.6 the model consists of the input state which is shown in Figure 2.4 and 5 fully connected hidden layers of size 400 each. Lastly the output layer is of size 4 as there are 4 different green light phases in the example that was analyzed. Furthermore there was an additional project done by Hussain [Hus18] on the hyperparameter tuning of the model proposed by Vidali et al. which explored the learning rate α and the γ parameter which defines how much weight is put on maximizing immediate rewards versus maximizing future rewards. The insights of Hussain's analysis were also used to further improve both the model proposed of Vidali et al. as well as the collaborative solution evaluated by this thesis.

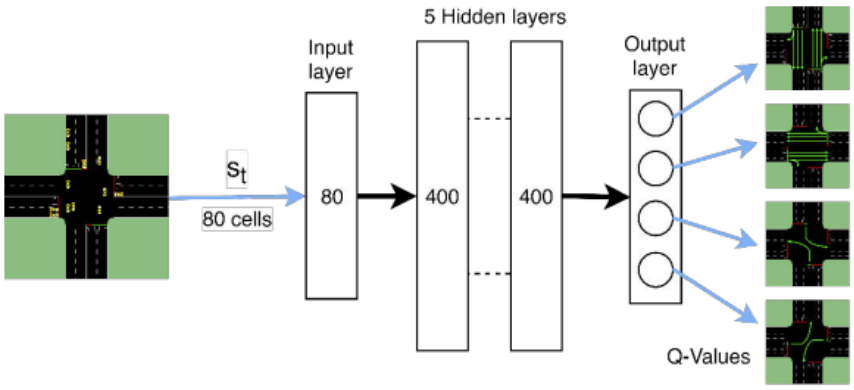

Figure 2.6: Scheme of the deep neural network (2019, Vidali et al. [VCVB19])

The experience replay function The basic concept of reinforcement learning describes an agent that is capable of observing its environment according to a given state representation and evaluate each state based on a predefined metric. This allows an agent to perform an action and observe how said action influenced the state of its environment and further evaluate exactly how good or bad the resulting change is in regard to the performance metric. Based on this concept an agent can learn from each action as they are taken. Experience replay is an extension of this concept where the gaining of experience is logically separated from the learning phase. Thus the agent does not attempt to immediately learn from its action but instead perform to the best of its

knowledge and store each resulting observations in a memory store in the form depicted in Equation 2.4 with s_t representing the state at time step t, a_t being the action taken at time step t and lastly r_{t+1}, s_{t+1} being the received reward and the resulting state respectively.

$$m = \{s_t, a_t, r_{t+1}, s_{t+1}\} \tag{2.4}$$

During the learning phase these 4-tuples are then randomly sampled from the memory store in batches in order to train the agent and improve its policy.

CHAPTER 3

Reinforcement Learning Based Traffic Light Management: Design and Mechanisms

This chapter provides an in depth explanation of the designed collaborative reinforcement learning algorithm for smart traffic light grids and the reasoning behind each design step that was chosen. The chapter is divided into an overview of a single Q-Learning agent and its design in Section 3.1 and the implementation of these agents in a collaborative setting in Section 3.2.

3.1 Deep Q-Learning for Smart Traffic Lights

The reinforcement learning framework can, in its simplest form, be summarized by Figure 3.1 which is a formalization of a *Markov Decision Process*. In accordance with this, the core components of each reinforcement learning problem are the learning agent itself, the state space which is the agent's view of the environment, the action space which defines the actions an agent can take within the environment and lastly the environment with which the agent interacts. Additionally the system requires a reward function in order to measure the reward r_t it receives for executing action a_t that leads to the transition of the state s_t to the state s_{t+1}. Based on this information there is an important distinction on how an agent attempts to maximize the received reward which is categorized as either model-based reinforcement learning or model-free reinforcement learning. Sutton and Barto [SB18] have described model-based RL to rely on *planning* based on understanding of the environment, while model-free RL relies on *learning* about the effects of their actions. This means that a model-based approach trains a complex model of the environment with which an agent can predict the outcome of certain actions ahead of time and make decisions based on those predictions. A model-free agent on

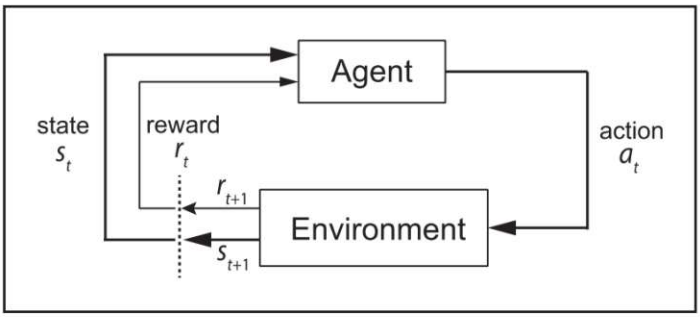

Figure 3.1: Basic view of the reinforcement learning framework (2018, Galatzer-Levy et al. [GLRC18])

the other hand does not require any knowledge of its environment. It can simply keep executing actions, observe the state changes and learn the best policy from experience. A clear advantage of a model-based agents is that given a sufficiently sophisticated model of the environment it can make relatively good decisions even when faced with states that have never been encountered before. A model-free approach on the other hand is entirely dependent on its experience but is a lot more flexible and less complex in its design. The development of both a model-free approach by Mnih et al. [MKS+13] and a model-based approach by Kaiser et al. [KBM+19] for popular atari games which is an important use case for reinforcement learning in literature showcase the advantages and disadvantages of both concepts. The advantage most important to the solution proposed in this thesis is the flexibility of model-free reinforcement learning at the cost of needing more data and time to train the policy. This is why Q-Learning which is a model-free RL approach is a fitting tool for the domain of traffic management in intersections. Given the ever increasing processing power, availability of real time traffic data and the development of microsimulation tools for traffic flow such as *SUMO*, there is an abundance of data to train from and resources to do so. In order for a machine learning problem to be solvable with Q-Learning, it has to be definable as a finite markov decision process. This means that both the state space as well as the action space need to be finite. In the case of a traffic light agent this is certainly true for the action space as the only available actions are the available light phases that can be activated. The state space defined by Vidali et al. [VCVB19] as described in Section 2.3 is also guaranteed to be finite as there is a finite number of cells and subsequent permutations which depict the presence of vehicles as shown in Figure 2.4. These two criteria must be satisfied because Q-Learning entails the construction of a finite Q-Table that assigns a fixed Q-value for each available action in a given state which can then be used to choose an based on wihch has the highest Q-Value for a given state. The Bellman equation shown in Equation 3.1 is used to calculate these Q-Values by considering the reward r_t received for action a_t in state s_t in combination with the maximum expected future reward achievable in the resulting state s_{t+1}. The future reward is weighted by the hyperparameter γ which specifies whether the agent tries to maximize immediate rewards or put more weight on the expected reward of

future states.

$$Q(s_t, a_t) = r_t + \gamma * max_A Q'(s_{t+1}, a_{t+1}) \tag{3.1}$$

In theory this is enough to start training an agent by continuously feeding it data and updating the Q-Values based on the resulting experiences. In reality this quickly becomes unfeasible for complex states. The Q-Table for a single intersection with four actions and a state representation of 80 binary cells is shown in Figure 3.2. While the state space is finite there are still $2^{80} = 1.21 * 10^{25}$ different states with four Q-Values assigned to each state which makes it impossible for an agent to collect sufficient experience for each state and action pair. To solve this problem deep Q-Learning has been introduced

Q Table		Actions			
		NS-Green	NSL-Green	EW-Green	EWL-Green
States	0	0	0	0	0
	.	.	.	.	.
	.	.	.	.	.
	.	.	.	.	.
	400	0	0	0	0
	.	.	.	.	.
	.	.	.	.	.
	.	.	.	.	.
	$1.21*10^{25}$	0	0	0	0

Figure 3.2: Q-Table for a single intersection with 4 actions and an input state of 80 binary cells

which combines the concept of Q-Learning with deep neural networks. Here the concept of the Q-Table is replaced by a deep neural network which receives a state as an input and produces Q-Values for each action as output as shown in Figure 3.3. With this approach even highly complex image input can be used for state representation within a Q-Learning framework with great results as shown by Mnih et al. [MKS+13] with the example of a Q-Learning agent learning to play popular atari games, easily outperforming humans.

Since the single agent is modeled directly after the Deep Q-Learning agent proposed by Vidali et al. [VCVB19] which was already discussed in Section 2.3 the following only shortly revisits the core components:

State Representation The concept of the state representation is the same as shown Figure 2.4 with the only difference being the exact size and total number of cells which varies depending on the size of the intersection and the available light phases. In order to allow for different distances between intersections while still building on the same model the number of cells is fixed at 10 per lane which are then spanned across the available lane space. Thus the agent can be trained using the same state representation whether the incoming road is 400m or 100m long. Furthermore, for a more generalized definition, a *lane* can span multiple physical lanes if they are all affected by the same green light phase. Figure 3.4 shows an example of state cell distribution in the case of a one-way setup. While there are four lanes, they are all affected by the same green light phase

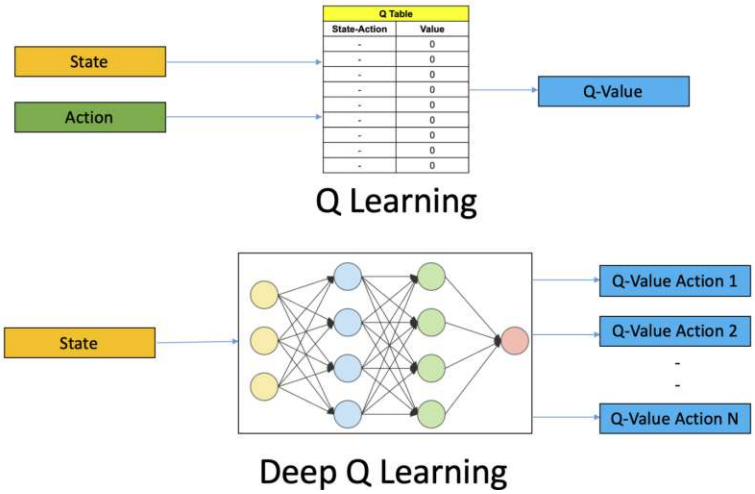

Figure 3.3: Deep Q-Learning Concept (graphic by A. Choudhary [Cho])

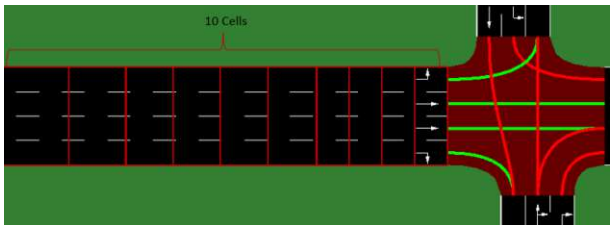

Figure 3.4: State cell distribution

and are combined into a single state lane represented by 10 cells. For the experiment conducted by Vidali et al. this results in 20 cells per incoming road for the intersection (since there is a separate green phase for left turns and straight/right turns) and a total of 80 binary cells of input for an intersection with four incoming roads as shown for the input layer in Figure 2.6. It is important to note that this concept is designed to be applicable to any given intersection but it might result in a differently sized input layer and thus might benefit from additional fine tuning of the size and number of the fully connected hidden layers.

Action Set The available action set for each agent is given by the number of different green light phases of the intersection. In the case discussed by Vidali et al. this set is described by 3.2 which contains NSA for NORTH-SOUTH bound traffic which either goes straight or right, NSLA for NORTH-SOUTH bound traffic which goes left and their EAST-WEST counterparts.

$$A = \{NSA, NSLA, EWA, EWLA\} \tag{3.2}$$

This action set is easily adaptable to any given intersection as it directly represents the green light phases. The only adjustment that needs to be made before training is defining

the output layer of the deep learning model to match the number of actions available.

Reward Function As discussed in Section 2.3 the reward function calculates the accumulated wait time awt_t of all vehicles in the lanes approaching the intersection (outgoing lanes are not observed) at timestep t and as shown in Equation 2.2 and Equation 2.3. The reward r_t for the action a_{t-1} is then calculated by comparing the sum of wait times $atwt_t$ with the wait times of the previous action step $atwt_{t-1}$. Within the scope of this thesis the system is designed to be trained in the simulation and then deployed to the actual intersection without any further need of training. This means that as long as the state representation is feasible for real world implementation the reward function can be very complex and only realistically usable within the simulation. Tracking the accumulated wait times for every vehicle is not technically impossible with modern image recognition as shown by Yudin et al. [YSK+19] but it might pose further challenges in its implementation and increase the overall cost. Due to these circumstances the evaluation phase also measures the performance of a simpler reward metric which is the total queue length of all incomming lanes for a given intersection denoted by Equation 3.3.

$$sum\_queuelengths = \sum_{lane=1}^{n} queuelength_{(lane,t)} \tag{3.3}$$

This metric can be more easily approximated in a real world application by first counting the cars that pass a threshold at an incoming lane and also the cars that leave the intersection at the traffic light. The comparative analysis conducted by Mandal and Adu-Gyamfi [MAG20] shows promising results regarding the reliability of traffic counting solutions. The benefit of having a reward metric that can be measured in real time after implementation is that the agent can constantly generate new data to learn from and can continuously improve its policy.

Deep Q-Learning Lastly, the deep Q-learning agent is implemented largely according to the scheme shown in Figure 2.6 with the only difference being the varying input sizes and output sizes which are more loosely defined to fit any given intersection. In a given grid of intersections each agent controls one traffic light which is training based on its own experience. The main difference for collaboration among these agents is timing and state representation which will be further discussed in Section 3.2. Aside from these changes for agent collaboration the models are trained exactly like the model proposed by Vidali et al. [VCVB19]. In line with this design the timing of a non-collaborative agent for consecutive decisions either spans 10 seconds if a green phase was extended or 14 seconds if the light phase is switched. These timings are derived from the green phase time $G = 10s$ and the yellow phase time $Y = 4s$ which are set to these fixed values for the sake of comparability of different solutions over the course of this thesis but can and should be fine tuned for real life application as required. As described in Section 2.3 the training of the deep neural network is not done directly after each action step. Instead the experience replay function is used to store all experiences in a memory store which can hold a maximum of 50.000 samples and requires a minimum of 600 samples before

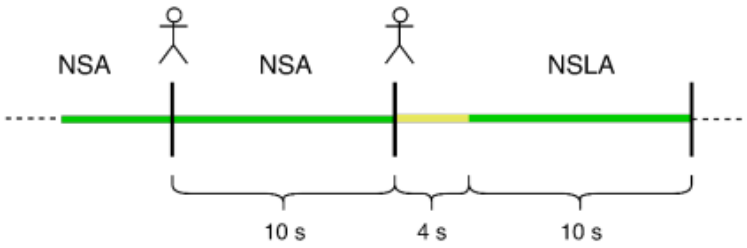

Figure 3.5: Simulation steps between actions (2019, Vidali et al. [VCVB19])

training of the deep neural network begins. Each training episode simulates an hour or 3600 time steps of traffic and generates one experience sample every 14 time steps (every time an agent chooses a new action and receives the reward for its previous action). After each simulation episode, if the memory store contains at least 600 samples the deep neural network is then trained for 500 training epochs with each epoch training on a randomly sampled batch of 100 samples.

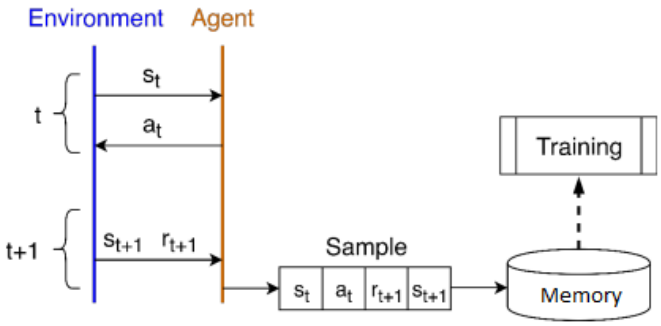

Figure 3.6: Experience replay of a single agent (2019, Vidali et al. [VCVB19])

3.2 Collaborative System Design

The deep Q-Learning solution proposed by Vidali et al. [VCVB19] has shown promising results for a single intersection, but in a real life application an intersection can rarely be viewed as an isolated problem and the complexity increases rapidly as traffic grids become larger. In order to expand the single agent reinforcement learning framework as depicted in Figure 3.1 it is important to have an understanding of the environment, how an agent can move and interact within said environment and how different agents might influence each other in their actions. For the example of a smart traffic light grid it is apparent that the spatial aspect which means the part of the environment visible to any given agent remains constant at all times as the traffic lights are stationary. This makes conceptualization easier as movement of an agent does not need to be accounted for and agents do not interfere directly with the observed space of other agents. As

described in Section 3.1 the part of the environment an agent can observe is limited to the available cells in the incoming roads of its own intersection but with a growing grid the actual environment and usable and potentially relevant information also increases. This poses the question of how much information is relevant for a given agent, how expensive the gathering and communication of this information is and whether or not it is feasible to compute the information within an acceptable time frame. It is obvious that a centralized solution where the decision for all traffic lights in a city is made by a single server that first needs to collect all available traffic data is not feasible. The trade-off is communication cost and complexity versus the added benefit of the additional information. This thesis proposes a solution that limits the collaboration and thus the complexity and cost of communication of each agent to its immediate neighborhood resulting in a decentralized, scalable grid of smart traffic lights. With this approach large, complex grids can be reduced to smaller sub problems with a maximum of four connected, collaborating agents as shown in Figure 3.7.

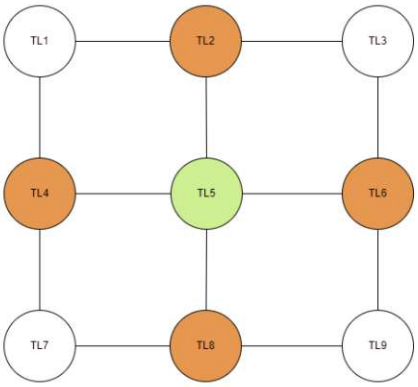

Figure 3.7: Simple Node Network

The simplest way of extending this solution to an entire grid with multiple agents is implementing the isolated Q-Learning agent separately for each intersection. While this solution is non-collaborative it is mentioned here as it is the easiest to implement with no need for the independent agent to communicate with other nodes and always provides a fallback solution in the case that communication amongst agents becomes impossible due to hardware failure or similar circumstances. Furthermore this disjoint, non-collaborative solution serves as a baseline to measure the benefit provided by different approaches evaluated in Chapter 5.
Communication between agents allows for two core concepts considered in the design of the proposed solution, which are the sharing of information and the synchronization of decision making. The scope for sharing of information is defined and limited by the knowledge each agent has by itself. This entails the last decision made by the agent which represents the currently active light phase and the observed state according to the state representation defined in Section 3.3. The resulting framework is shown in Figure 3.8 where the function constructing the state can access the observations made by neighboring

agents. The second concept, synchronization of decision making, is especially important

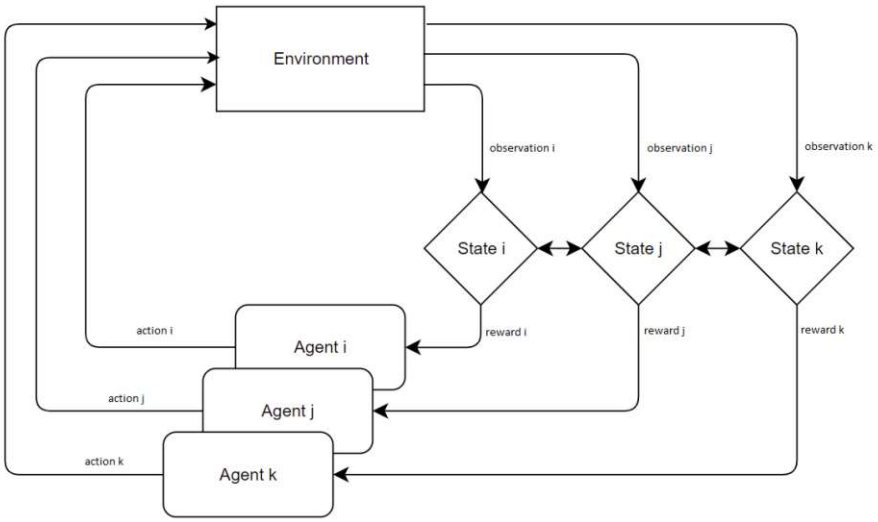

Figure 3.8: Multi Agent Approach

when considering for instance the green wave effect which would not be achievable by a non-collaborative system but becomes possible by sharing information on the currently active light phase of neighboring agents and timing their decisions correctly. As both the ideal amount of shared information as well as the synchronisation between agents is strongly dependent on the attributes of the intersections in question, Section 3.2.1 and Section 3.2.2 respectively propose an assortment of different schemes that are evaluated in Chapter 5. In order to explain the concept for state representation and synchronisation the following sections depict an implementation based on the non-collaborative state representation featuring 80 cells per intersection as proposed by Vidali et al. (20 cells per lane for four incoming lanes as shown in Figure 2.4) with the target agent (**TLC**enter) being connected to and collaborating with adjacent agents on all four lanes (**TLN**orth, **TLE**ast, **TLS**outh, **TLW**est) depicted in Figure 3.9. In the final implementation all synchronisation and state representation schemes can be applied for an arbitrary number of connected traffic agents and size of their non-collaborative state representation.

3.2.1 Collaborative State Representation

As briefly described above the state representation in the collaborative system discussed here can at most use all available information of its neighbors. This thesis explores three different state representations in regard to the amount of data shared between agents. The three approaches which utilize the available information to varying degree are referred to as **collaborative-complex**, **collaborative-simple** and **collaborative-optimal**.

Collaborative-Complex State The idea of the collaborative-complex state is relatively simple and straight forward. This state uses all the information of adjacent

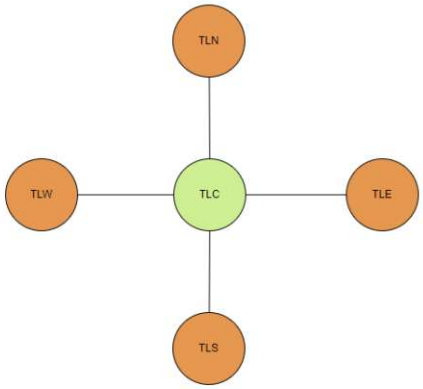

Figure 3.9: Fully Connected Traffic Light

traffic agents as is and concatenates the additional state information on to its own. As mentioned before, the available information here are the 80 cells an agent can observe within its own intersection plus the light phase that is currently active in the intersection. Figure 3.10 shows the resulting input array for the example of Figure 3.9 where the *TLC* agent receives the state and binary encoded light phase information of all surrounding traffic lights resulting in an input vector of size 408. In order to keep the same ratio of 1 : 4 for input size to width of hidden layers in the model this requires a hidden layer width of approximately 1600 resulting in a much more complex model. In deep learning however it is not guaranteed that simply keeping the same ratio results in equally optimized results and these parameters are often chosen based on experience and best practices. As such future work should include further research into the width and number of hidden layers for the complex model. A comprehensive overview on this topic is provided by the work of D. Stathakis [Sta09]. While the communication cost for 82 bit (80 bit state + 2 bit binary encoded light phase) per adjacent traffic light is certainly manageable, it must be evaluated whether or not the additional information provides enough utility and if the more complex model can achieve stable performance.

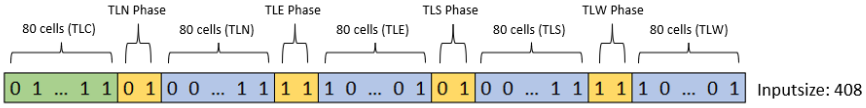

Figure 3.10: Input for Complex State Representation

Collaborative-Simple State The goal of the collaborative-simple state is to keep the model as simple as possible while still allowing the system to discover concepts like the green wave effect during the training phase. This is done by limiting the exchange of information to the binary encoding of the light phase of all adjacent traffic lights. While the agent is not provided information on the exact traffic situation of its neighbors it

can react to changes in light phases of surrounding traffic lights. Figure 3.11 depicts the resulting input vector with a size of 88 bits. The resulting communication cost is only 2 bit of information per adjacent traffic light and the increase of the state size is thus only 8 bit for four adjacent traffic lights.

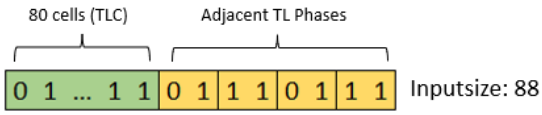

Figure 3.11: Input for Simple State Representation

Collaborative-Optimal State The collaborative-optimal state aims to combine the benefits of a simpler model while still providing the utility of traffic observations in surrounding intersections. This is achieved by simplifying the state representation of surrounding intersections and limiting the observed lanes to those that are relevant to the agent i.e. those that allow for traffic to enter the intersection. An example is shown in Figure 3.12 which depicts the observed lanes from an intersection to the east of the agent itself. All west-bound vehicles will enter the intersection of the agent and are thus observed. Furthermore the number of cells is reduced to 4 per relevant lane and the size of the cells is increased to cover a larger part of the lane with less granular information. In the cluster of intersections of the experiment setup TLC has 4 neighboring traffic

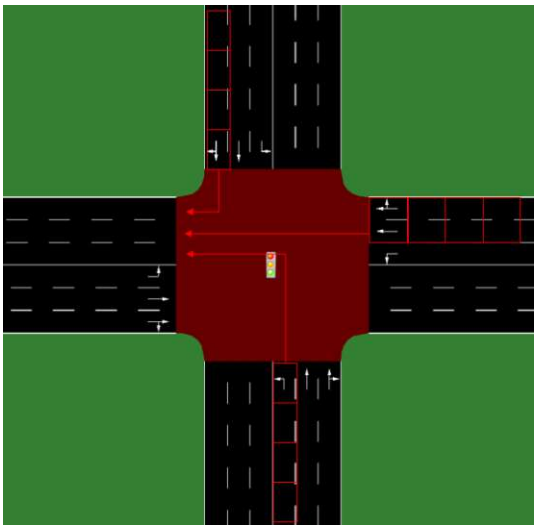

Figure 3.12: Optimal-State observing only the relevant lanes

lights (TLN, TLE, TLS, TLW) with 3 relevant lanes each. Additionally the currently active light phase of each adjacent traffic light is also binary encoded and added to the input state. The resulting input vector is shown in Figure 3.13 with a total size of 136 bits.

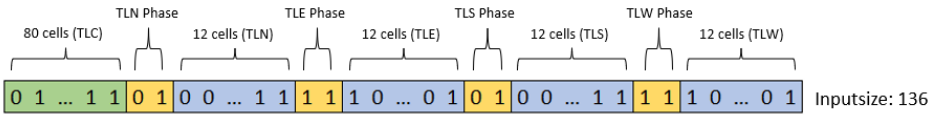

Figure 3.13: Input for Optimal State Representation

3.2.2 Synchronisation of Decision Making

With the added complexity of multiple intersections the timing of an agent's decision becomes very important. In order for a green wave effect to be possible, all traffic lights need to be synchronized and match their green light phases accordingly. This effect however is not always desirable especially if the traffic across all lanes is evenly distributed and there is no main road to benefit from this effect. The desired type of synchronization may thus vary based on the traffic load and structure of each intersection. In order to cater to these different circumstances this section proposes three different synchronization schemes for collaborating traffic light agents. The three approaches are **asynchronous** decision making, **synchronous** decision making and **cycled** decision making. The following depicts all synchronisation schemes for the example of green phase time $G = 10s$ and yellow phase time $Y = 4s$.

Asynchronous The asynchronous synchronisation scheme follows the concept proposed by Vidali et al. [VCVB19] and as discussed in their work may lead to different time intervals between agent decisions as the interval is either one green phase time $G = 10s$ if the agent chooses to not change the light phase or $G + Y = 14s$ if the phase is switched and requires a yellow light phase. These irregularities, while negligible for a single intersection, might have an adverse effect on collaborating agents as this allows the time delays between two adjacent agents to vary constantly. In other words if one agents changes light phases and a neighboring agent decides to remain in the current green phase, their next decision step will vary by exactly one yellow phase duration. This greatly changes the utility of the information on the currently active light phase in a neighboring traffic light, as this variance may constantly change if decision making is not synchronized. Depending on the current delay between the two agents the phase might remain active for an entire duration of $G = 10s$ or switch shortly after sending the information. This constantly changing utility can make it more difficult for the model to learn how to use this information correctly. The delay resulting from asynchronicity is shown in Figure 3.14 where TL1 changes the light phase and TL2 chooses to remain in the currently active phase. While the original decision was made at the same time, the second decision differs as TL2 reaches its next decision steps 4 seconds (or one yellow phase duration) before TL1.

Synchronous The synchronous scheme aims to stabilize the utility of the information received from neighboring agents. By having all agents decide their next light phase at the same decision step it is ensured that the resulting phase will be active for the next

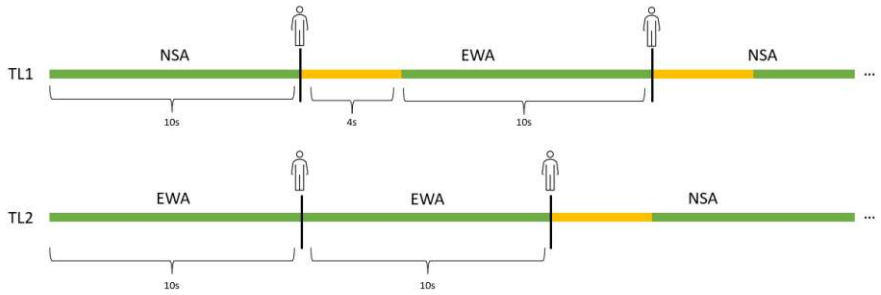

Figure 3.14: Asynchronous Scheme

$G + Y = 14s$. This also introduces an implicit penalty for switching light phases as a prolonged phase adds $14s$ of green light while switching results in $4s$ of yellow phase followed by $10s$ of green phase until the next decision step as shown in Figure 3.15. The most important benefit and drawback of this approach is that the order in which the agents make a decision also dictates who can use this information. The first agent to decide can thus not utilize the light phase information of any neighboring traffic light as they have not yet chosen the next phase but will potentially change it within the current time step. While this forces a hierarchy for the order of in which the agents decide their next step it can provide great utility in intersections where a decision hierarchy is sensible as the subsequent decision can rely on the fact that the light phase of its neighbor is active for the maximum duration of $14s$. A common situation would be a main road with commute traffic at rush hour where the majority of traffic is headed out of town. In this situation the hierarchy allows the agents to generate a green wave effect and also react accordingly if a preceding traffic light decides to switch to a different light phase and it gives each subsequent agent light phase information that will not change until the next decision step.

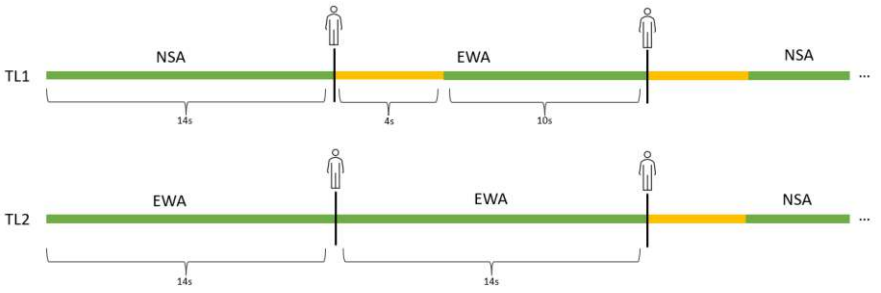

Figure 3.15: Synchronous Scheme

Cycled Lastly the cycled synchronisation scheme looks to stabilize the utility of light phase information in situations where a decision hierarchy can not be established. The experiment setup discusses in Chapter 4 assumes relatively equal distribution of traffic over

all incoming roads and intersection. Thus, disadvantaging certain agents for the benefit of others does not make sense. In order to keep the utility of light phase information stable while providing the same benefit to all agents the cycled synchronisation offsets the decision steps of neighboring agents by half a decision phase $(G + Y)/2 = 7s$. This results in an alternating pattern where the direct neighbors are always delayed by this offset and all agents within a 2-hop distance are making synchronous decisions as shown in Figure 3.16. Figure 3.17 depicts this scheme applied to the given experiment setup

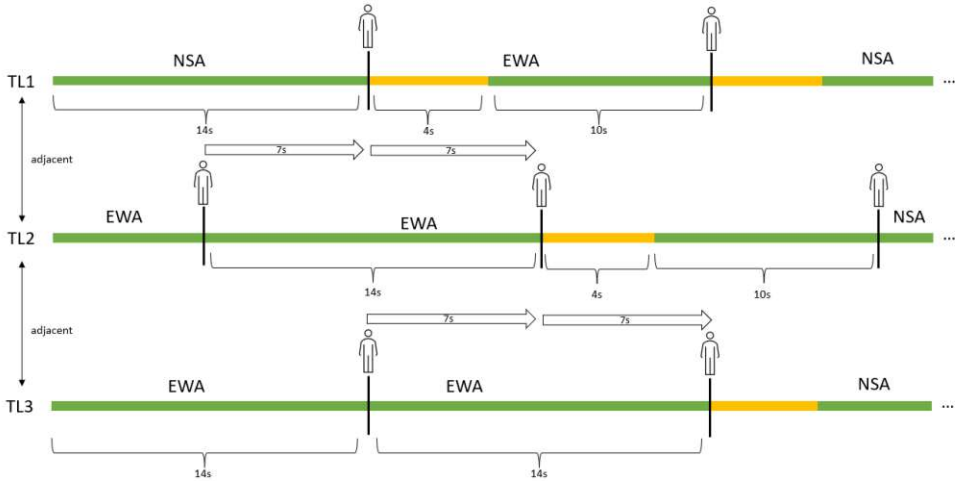

Figure 3.16: Cycled Synchronisation Scheme

plus additional neighboring nodes. It is important that as shown in Figure 3.17 real world application allows for connections that require exceptions to this pattern. In this case the system is forced to assign a delay which favors one neighboring intersection and disadvantages the other.

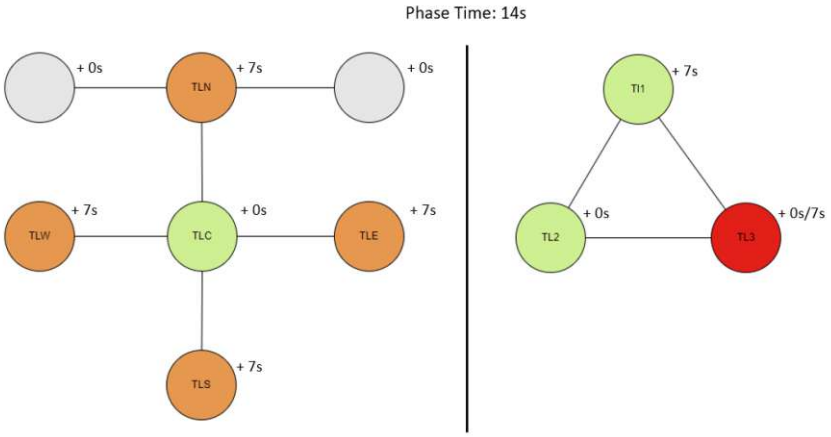

Figure 3.17: Cycled Synchronization Network

CHAPTER 4

Simulation Environment

Traffic simulation is not only a key methodology for evaluation of the solution proposed by this thesis, but also a core component of the conceptualisation and implementation of said solution itself. The reinforcement learning framework, and especially deep Q-Learning, requires a large amount of experience before an agent can make sensible decisions. Deploying an untrained agent in a real world intersection would result in a considerable amount of time in which the agent would operate solely on random decisions which would be unacceptable during day to day traffic. Modern traffic simulation tools make it possible to cope with this cold start problem by defining a state representation that can be implemented within the simulation as well as in the real world application. Thus, the agent can be trained within the simulation framework for countless hours of simulated traffic and be deployed to the real world once it achieves satisfactory performance. It is important to note that the technology used for traffic simulation is interchangeable as long as it allows for the state representation and the reward calculation required by the agent. The following sections give an overview of the simulation framework used and how the environment was set up for training and evaluation of the solution.

4.1 Basic Setup in SUMO

Both the reinforcement learning based training as well as the testing during the course of the experiment execution are implemented using ***S**imulation of **U**rban **Mo**bility*, which is an open-source microscopic traffic simulation framework. *SUMO* allows for full control over intersection- and traffic-design and also allows for step by step execution of any given simulation. This makes it possible to run each simulation on a second by second basis using self designed traffic networks, while the framework handles all the aspects of the actual traffic flow such as acceleration, deceleration, stopping at intersections and lane merging. Additionally the ***Tra**ffic **C**ontrol **I**nterface* or *TraCI* for short provides full control during the simulation using a python client to change traffic light behavior

and retrieve values of simulated objects such as vehicle speed, wait times, queue lengths in intersections and all other required environment information needed for the training of the traffic light model and for measuring the effectiveness of a given solution. This versatile toolset for realistic traffic simulations makes *SUMO* a popular framework used in many different areas such as analysis of congestion impact in the works of Malik et al. [MKAS19], simulation and analysis of CO^2 emissions in the case of Vidali et al. [VPA+20] or even analysis of energy consumption in wireless sensor networks in road networks as utilized by Kabrane et al. [KKE+17]. One important caveat that should be noted is that *SUMO* allows for a much more detailed representation of the state of a traffic network as opposed to the real world. This must be considered especially in the case of a machine learning model that is trained using the accurate information provided by the simulation but is expected to perform in a real life environment. This will be further discussed in Chapter 6. In summary, a full *SUMO*-Simulation consists of two parts. For one there is the road network file stored in XML format which models the actual network with its roads and intersections and also additional information such as traffic light cycles, priority rules, allowed turns and so on. Secondly, there is the route file which is also stored as XML stores all available routes (i.e. what sequence of lanes can be taken by a vehicle from entry to exit of the network) and it also contains all cars that are generated into the simulation with their assigned timestamp for entry and which of the predefined routes the car will take. The following sections explain the networks and routes used for training and performance measurement and the reasoning behind the setup in more detail.

4.2 Traffic Networks

As mentioned above, the first part of a *SUMO*-Simulation is the road network file. This section goes over the networks designed for training and testing of the algorithms. There are two simulation setups that are used to test the performance of the proposed algorithm and highlight its strengths and limitations. First is the experiment network which consists of five traffic lights with a center traffic light that is connected to the other four in the form of a cross in order to have a large amount of additional information that can be used from surrounding traffic lights. The second setup is a real world example consisting of three intersections in the inner city of Christchurch, New Zealand to test the proposed concept combined with the insights gained from the experiment phase in Chapter 5 on actual traffic data and in a realistic setting. The process of adapting the algorithm to this real world example not only allows for measurements of effectiveness on real data but also provides information on implementation steps and challenges faced in the application of the concept in the field.

4.2.1 Experiment Network

The experiment network is the main environment for performance measurement and for the comparison of the collaborative algorithm against the non-collaborative baseline and

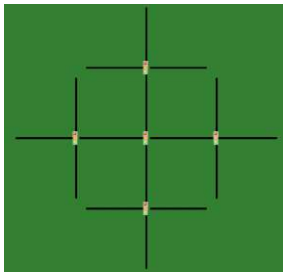

Figure 4.1: Experiment Network - Setup

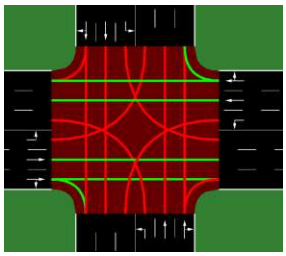

Figure 4.2: Experiment Network - Lanes

against commonly used fixed time intervals. As such it is designed to provide a stable setting in which the algorithms can be compared without a lot of additional variables that could influence the results (e.g. complex constellations of one way roads or difficult merging lanes). As stated above the experiment network is composed of a total of five traffic lights in a cross shape with the center traffic light in the middle being connected to the other four as shown in Figure 4.1. Each road contains three lanes in each direction for a total of six lanes with the left most lane allowing for left turns only and the right most lane allowing for right turns and straight crossing as shown in Figure 4.2.

Furthermore the roads in the network are all of identical length which is either 100, 200 or 400 meters based on which simulation type is chosen. This is implemented to analyse the impact of increasing distance between intersections and measure how much the value of information provided by neighboring traffic lights decreases with distance. While the corners of the roads appear to be connected in Figure 4.1, in the actual simulation the cars are spawned at the start of each respective road and also phase out the end without the corner turn being regarded or accounted for as valid routes.

Lastly the available light phases for each traffic light within the network are identical as well. The available phases are NORTH-SOUTH, NORTH-SOUTH-LEFT, EAST-WEST and EAST-WEST-LEFT with the 'LEFT' phases allowing for left turns only and the other two phases allowing for straight crossing as well as right turns as shown in Figure 4.3.

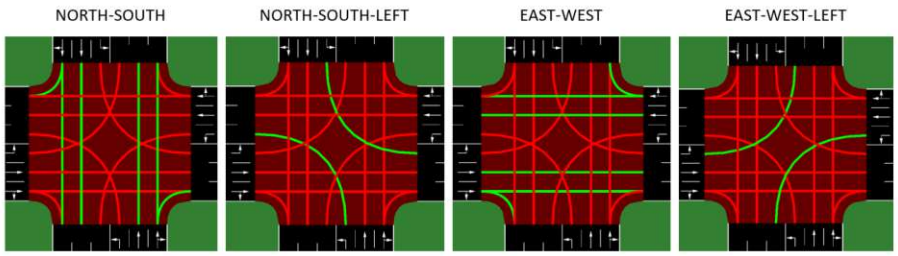

Figure 4.3: Experiment Network - Light phases of traffic lights

4.2.2 Real World Network - Intersections of Christchurch NZ

The second network is modeled after a series of three intersections on Montreal St. in Christchurch New Zealand. The exact intersections are *Montreal St and Hereford St*, *Montreal St and Worcester St* and *Montreal St and Gloucester St*. This specific case was chosen because the City Council of Christchurch provides a tool called *Intersection traffic counts database*[1] containing traffic counts of several intersections throughout the city from 2017-2021. This includes not only total traffic in the intersection but detailed counts per lane and turns taken in 15 minute intervals for the entirety of the 24 hour time frame as shown in Figure 4.4. Figure 4.4 also highlights the section used in the simulation. Furthermore the City Council of Christchurch provided values of the actual cycle times corresponding to the traffic count measurements upon request which are the timings used for the final real world experiment in Chapter 5.

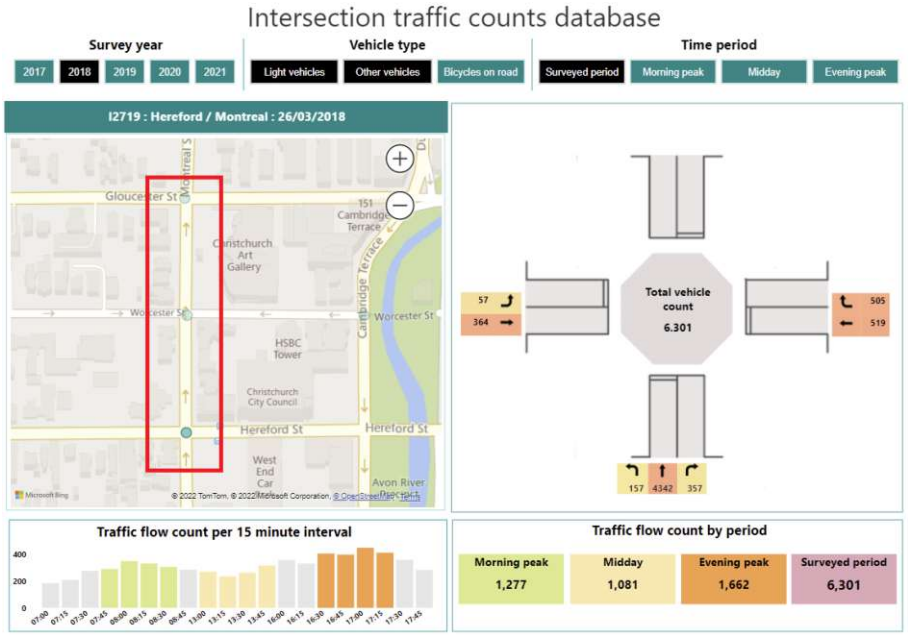

Figure 4.4: Selected intersection shown in the *Intersection traffic counts database*

Figure 4.5 shows the intersection of Montreal St. and Hereford St. and its *SUMO* equivalent. Since New Zealand traffic rules require left-hand traffic this detail was changed in the simulation to keep traffic rules in line with other simulations. Due to the nature of the network this has no impact on the functionality of the simulation as a whole and the provided real life data can be used almost as is. One required change is that in order to preserve right and left hand turns from the side lanes the counts have to be mirrored to the opposing lane. As shown in Figure 4.6 the resulting

[1] https://ccc.govt.nz/transport/improving-our-transport-and-roads/traffic-count-data/intersection-traffic-counts-database/

Figure 4.5: Intersection in Christchurch vs SUMO-Model

SUMO-Simulation thus has the lightphases NORTH-SOUTH-1 and EAST-WEST-1 for the intersections of *Montreal St and Hereford St* and *Montreal St and Gloucester St* and the phases NORTH-SOUTH-2 and EAST-WEST-2 for the intersection of *Montreal St and Worcester St* in accordance with their real life counterparts.

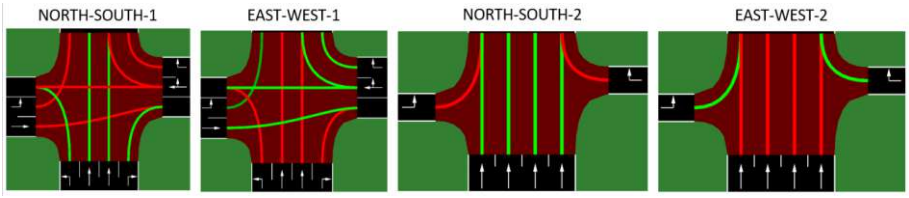

Figure 4.6: Available lightphases for the Christchurch SUMO-Model

4.3 Simulation Cases

The second part of each *SUMO*-Simulation is the route file which defines the available routes in the network and also stores at which point in time each car will spawn into the simulation and which route it will take. Both the route information as well as the specific car information is automatically generated by a separate algorithm based on the chosen simulation network, desired distribution, time frame and total number of cars for the episode. The four crucial aspects of each simulation case are the time frame of the case, the number of cars generated within that time frame, the distribution with which the cars are generated and the probability with which each route is assigned to a car.

4.3.1 Experiment Network

Each test case of the experiment network is designed to simulate an hour of traffic plus additional time steps for the agents to clear all intersections. The total number of time steps for an episode in the experiment network is 4000 steps $(3600 + 400)$ with the additional 400 steps providing a buffer for clearing all intersections which is especially important during the early training stages as the agent is expected to require more time to clear the entire intersection at the start of the training phase and it can potentialy

receive many positive rewards in these buffer steps where no additional vehicles are created. For the experiment network the main focus of the test cases is the evaluation of the collaborative feature of the traffic light agents. To this end, it is important to test how information provided by neighboring traffic lights factors into the decision making of an agent. The value of this information is expected to vary greatly depending on traffic load which is why the test cases are designed to create either low, medium or high traffic situations. The distribution of traffic over the available time steps is for the most part modeled after a Weibull distribution, as shown in Figure 4.7 which is generated using the *numpy.random.weibull* with the shape parameter $a = 2$. This distribution was chosen because it models the steep ramp up of rush hour traffic which then gradually declines towards the end of the rush hour. Since the experiment featured in Chapter 5 also tests

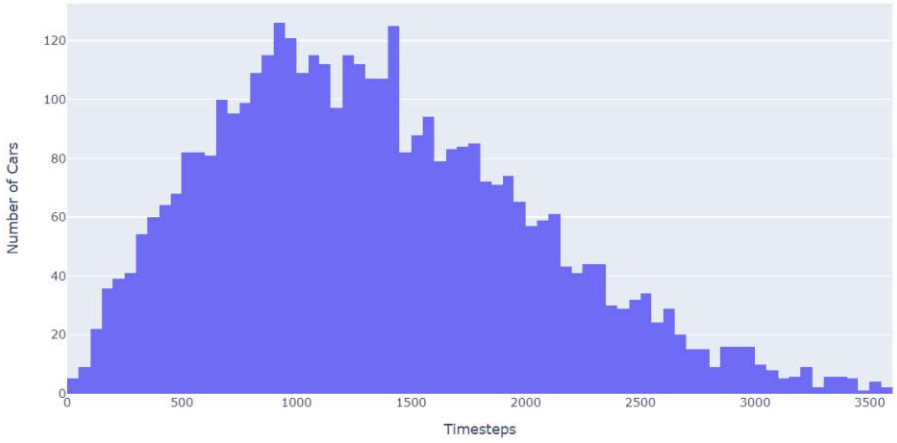

Figure 4.7: Weibull Distribution for High Traffic in 100m Experiment Simulation

for the robustness to alternate distributions, different robustness test cases are generated as well. These alternate distributions are firstly two separate Weibull distributions within the simulated hour to create two peaks in a short time span and secondly a uniform distribution to create a constant load of traffic as is the case for most rush hour phases. The resulting traffic load over the given time frame for both distributions is shown in Figure 4.8 and Figure 4.9 respectively. Both figures depict the generated load of 3000 vehicles in a 60 minute time frame.

The total number of cars generated for each simulation based on distance between intersections and chosen traffic load are listed in Table 4.1. The resulting traffic situations for the $100m$ distance-mode network with 1000, 3000 and 4000 cars are shown in Figures 4.10, 4.11 and 4.12 respectively.

Finally the routes which are taken by each car are assigned based on the predefined turn probabilities denoting whether a car goes straight, left or right at each intersection it encounters. For the experiment setup the starting point is chosen with equal probability for each edge of the simulation where cars spawn in and out of the simulation. From there a car has a 80% probability of going straight. If the car falls within the 20% that

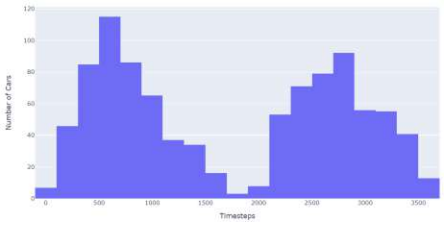

Figure 4.8: Double Weibull Distribution

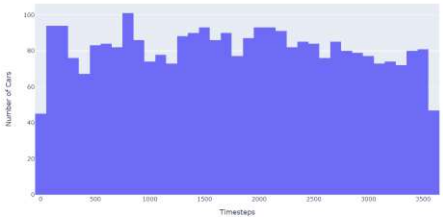

Figure 4.9: Uniform Distribution

Distance-Mode	Cars - Low Traffic	Cars - Medium Traffic	Cars - High Traffic
100m	1000/2000	3000	4000
200m	1500	3000	5000
400m	2000	3500	6000

Table 4.1: Final values for traffic scenarios - Experiment Network

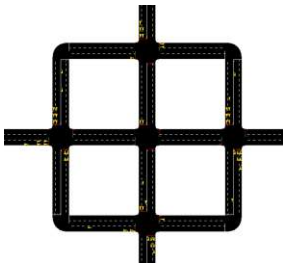

Figure 4.10: Low Traffic

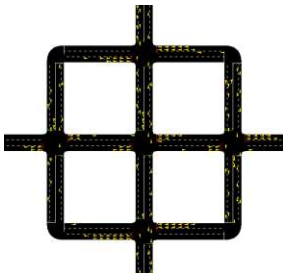

Figure 4.11: Medium Traffic

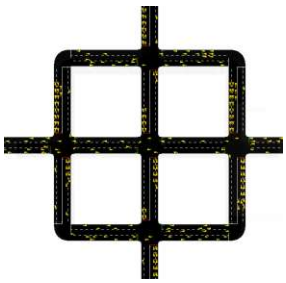

Figure 4.12: High Traffic

take a turn there is a 66.6% chance that it goes right and a 33.3% chance for a left turn. Based on these probabilities all routes are assigned for the generated cars.

4.3.2 Real World Network - Intersections of Christchurch NZ

For the second simulation network which is modeled after the intersections of Christchurch NZ the test cases are based directly on traffic counts provided by the traffic count database[2]

[2]https://ccc.govt.nz/transport/improving-our-transport-and-roads/traffic-count-data/intersection-traffic-counts-database/

Simulation Network	Cars - Low Traffic	Cars - Medium Traffic	Cars - High Traffic
Christchurch NZ	1100	1300	1700

Table 4.2: Final values for traffic scenarios - Real World Network

provided by the city council of Christchurch NZ. As shown in Figure 4.13 both the full daily counts of the observed cars as well as the actual counts per time of day are provided by the traffic count database.

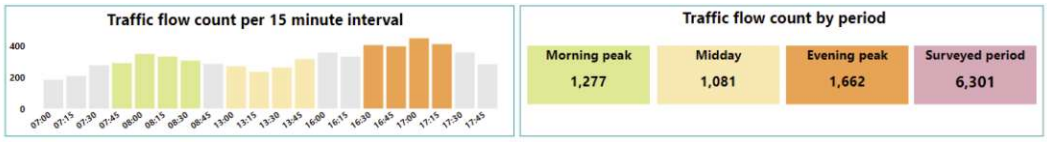

Figure 4.13: Traffic count for survey period in Montreal St and Hereford St

The resulting values for low, medium and high traffic are used as test cases and are shown in Table 4.2. For these scenarios, the evening peak represents the high traffic scenario, the morning peak is the medium traffic scenario and the midday measurements represent the low traffic scenario.

Lastly, the turn distribution which denotes with which probability a car spawns at a given entry point and whether it goes straight, left or right at each encountered intersection can be derived from the traffic counts. Figure 4.14 shows the counts for each lane which can be used to calculate the percentage values of cars in each lane and those can subsequently be used for the probability of the traffic routes in the final simulation. Thus all test cases are based on percentages derived from real life empirical values.

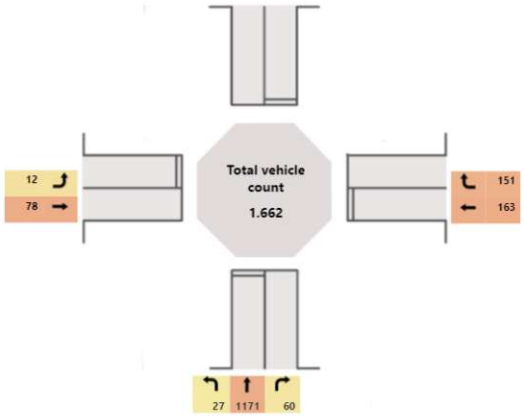

Figure 4.14: Turn distribution of Montreal St and Hereford St

CHAPTER 5

Evaluation

To preface the evaluation of the algorithm, an overview of the system which was used to develop, train and test the collaborative Q-learning solution is shown below.

Experiment Environment	
Operating System	Windows 10 Pro 64-Bit
CPUs	Intel Core i5-6600K | 4 cores @ 3.9GHz
Main Memory	16 GB DDR4 RAM
GPU	Nvidia GeForce GTX 980 | 4 GB GDDR5 RAM
Python Version	3.9.7
Tensorflow Version	2.5.0

The evaluation is structured into two different main sections, which is first an evaluation based on the experiment setup discussed in Chapter 4 which aims to test the strengths and weaknesses of the system plus the impact of certain design decisions and second an evaluation of the real world example of three intersections in Christchurch NZ to confirm the algorithm's performance on a real environment with real data. Furthermore, all comparisons among algorithms are structured in the same way which features a low, medium and high traffic scenario with 30 randomly generated test cases (one test case contains one hour of traffic) for each scenario. This is done to ensure that the results are reproducible and all algorithms are compared based on the same scenarios. The main metrics used for comparison are the total cumulative wait time over all intersections, which represents the efficiency of traffic handling as a whole, and the mean cumulative wait time per vehicle which represents the perceived efficiency for people in traffic.
Aside from answering the three research questions revisited below, this chapter also evaluates the algorithms strengths and weakness in regard to the three different state representation approaches (complex, simple, optimal), the three different synchronisation

Collaboration-Type	**Sync-Type**	**Reward**	**G/Y Phase**
disjoint / simple / complex / optimal	async /sync / cycled	cumwait / queue	10s | 4s
Training Episodes	**Training Epochs**	**Batch Size**	**Number of Cars**
100	500	100	4000
Simulation-Environment	**State Size**	**Hidden Layers**	**Max Time Steps**
experiment / christchurch / single	80	4x400	4000
Learning Rate α	**Gamma**	**Memory Size**	
0.001	0.75	600/50.000	

Table 5.1: Notation example for model-parameters

schemes (asynchronous, synchronous, cycled), impact of distance between intersections and robustness regarding a change in traffic load and distribution. Furthermore it aims to answer the three research questions revisited below.

1. Does the performance improvement of state-of-the-art reinforcement learning solutions for smart traffic lights proposed for single intersections hold for more complex traffic light grids with multiple intersections? Specifically is there a statistically significant drop in performance improvement over fixed time intervals?

2. Does collaboration among agents in a grid of traffic lights lead to a significant improvement in either cumulative wait time for the entire cluster of intersections or the average cumulative wait time per vehicle over non-collaborative agents?

3. Can the proposed collaborative approach achieve significant improvement over highly optimized real world intervals in terms of total wait time or mean vehicle wait time?

Since the full overview of parameters for each model is quite extensive and a large assortment of different models is discussed in this chapter, the notation shown in Table 5.1 will be used to document the exact attributes of each model. To provide a better understanding of this notation to the reader the following gives a short explanation of each parameter.

- **Collaboration-Type** refers to the type of state representation and information from adjacent traffic lights used. Disjoint refers to the non-collaborative baseline

solution proposed by Vidali et al. [VCVB19] and simple/complex/optimal refer to the state representations discussed in Chapter 3. To shortly revisit these types, the simple collaboration receives only the currently active traffic light phase of adjacent intersections, the optimal collaboration receives the active light phase and the state representation of the relevant lanes that contain traffic headed for the intersection and lastly, the complex collaboration receives the entire state representation of adjacent intersections as well as the currently active light phase.

- **Sync-Type** refers to the synchronisation schemes discussed in Chapter 3. The three synchronisation schemes are denoted asynchronous, synchronous and cycled. The asynchronous scheme does not sync up the decisions of agents and thus, each agent might change their light phase at different time steps within the simulation. The synchronous scheme ensures that all agents decide their next action at the same time step. Finally, the cycled scheme ensures that adjacent agents decide their action steps in an alternating pattern which is offset by half a cycle length (a light cycle consists of the yellow phase duration plus the green phase duration. In the experiment example green phase time $G = 10s$ and yellow phase time $Y = 4s$, the full cycle length is $C = 14s$).

- **Reward** denotes the reward metric used for training the model. While most models are built using the cumulative wait time function, this chapter also discusses queue lengths as an alternative reward function that can be implemented more easily in a real world application.

- **G/Y Phase** defines the green phase and yellow phase duration which, together define the total phase time between an agent's decisions.

- **Training Episodes** defines the number of 1-hour simulation episodes that are run during the training phase. 100 training episodes result in a total of 100 hours of simulated traffic generated for training. In order to have comparable results this parameter is set to 100 for all models and will thus be omitted in subsequent model overviews.

- **Training Epochs** refers to the number of training rounds done during the experience replay phase. With 500 training epochs the deep neural network is trained on 500 batches of randomized experience samples after each training episode. Just like training episodes, this parameter is fixed for all models and will be omitted in following model overviews

- **Batch Size** denotes the sample size that is pulled from the experience store during each training epoch to train the deep neural network. The batch size is also fixed for all following models and thus omitted.

- **Number of Cars** defines the total number of cars in the randomly generated traffic scenario of a training episode.

- **Simulation Environment** shows which intersection simulation was used to train the model. Possible values are *experiment*, *experiment-medium*, *experiment-long* for simulations with 5 intersections and varying road lengths, *christchurch* for the simulation of the real world example and *single* for the remodeled single intersection feature in the work of Vidali et al. [VCVB19].

- **State Size** is the exact size of the input vector according to the collaboration type that was used.

- **Hidden Layers** defines the number of hidden layers and their respective widths (i.e. number of neurons in each layer). Since the input size varies based on the selected **Collaboration-Type** it is necessary to fine tune the hidden layers accordingly. The choice of depth and width of hidden layers is often a difficult problem when optimizing a deep learning solution and as such it might require exhaustive parameter search and trial and error. D. Stathakis [Sta09] provides a comprehensive overview for best practices on this topic.

- **Max Time steps** defines at which time step the simulation is ended. This includes buffer steps on top of the 3600 (60 minutes) for which cars are generated so the model can experience the ramp down of clearing the intersection of all traffic.

- **Learning Rate** is the α parameter of the deep learning model and is fixed for all of the following models.

- **Gamma** denotes the weight with which the model regards possible future rewards as opposed to the reward received directly for a given action. This parameter is fixed for all models as well.

- **Memory Size** is the minimum and maximum number of samples stored in the experience replay store. The minimum defines a set number of experiences that need to be stored before the neural network starts the learning process. When the maximum number of experiences is reached the oldest experience is removed with each newly gained experience. This parameter is fixed for all of the following models and will be omitted.

5.1 Isolated Intersection

In order to have a benchmark of the single intersection problem discussed by Vidali et al. [VCVB19] with controlled parameters similar to those of the experiment setup discussed in Chapter 4 and to confirm the performance shown in the referenced work, this problem was first remodeled with some slight adaptations. The single intersection in the referenced paper features roads with four lanes in each direction and a length of $750m$ for each road. The number of lanes was reduced to three (removing one center lane which allowed for straight crossing only) and the length of incoming roads was reduced to $100m$. The shorter roads are more realistic in an urban setting and it is also a more feasible range to

Collaboration-Type	**Sync-Type**	**Reward**	**G/Y Phase**
disjoint	async	cumwait	10s | 4s
Number of Cars	**Simulation-Environment**	**State Size**	**Hidden Layers**
1500	single	80	4x400

Table 5.2: Single Intersection - Disjoint Model Parameters

monitor in a real world application as detection of vehicles over a span of $750m$ is not covered by literature working on traffic detection in intersections and can potentially be very difficult. The exact model parameters of the single intersection model implemented here are shown in Table 5.2. Figures 5.1 and 5.2 show the improvement of negative

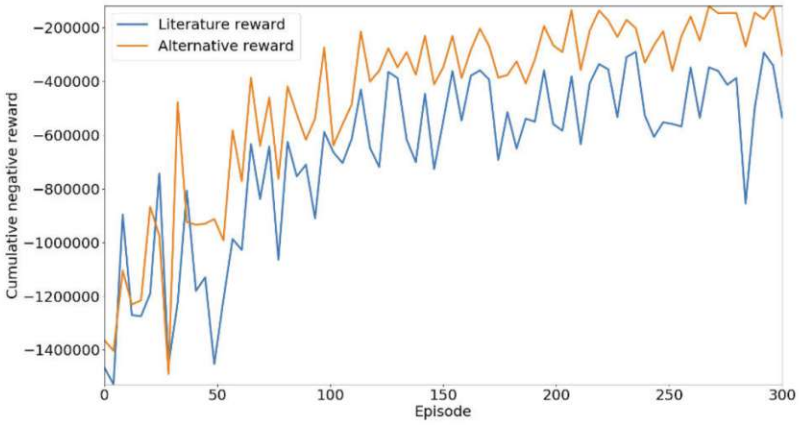

Figure 5.1: Training improvements in referenced paper (2019, Vidali et al. [VCVB19])

rewards received over the training of the algorithm from the works of Vidali et al. and the remodeled intersection implemented here respectively. Despite the adaptions made to the intersection, the performance is certainly comparable as in both cases the improvements are in the range of a factor of 6-7 from the start of the training phase versus the fully trained model. The exact values are expected to vary as the algorithm was trained on scenarios featuring 1500 cars within the hour instead of 4000 cars within 90 minutes (parameters used by Vidali et al.). The reduced number of cars is an approximation of the same scenario given the shortened simulation time and heavily reduced road length.

The fixed time interval to which the algorithm was compared also slightly deviates from the one used in the referenced paper which was $\{30, 4, 15, 4, 30, 4, 15, 4\}$ for the action set $\{NSA, NSLA, EWA, EWLA\}$ with the 4 second yellow phases added between green phases. The adapted cycle used in this evaluation is $\{38, 4, 10, 4, 38, 4, 10, 4\}$ and it should be noted that the results of the fixed time interval are not fully optimized in this setting as it was designed to better utilize the green wave effect in the experiment setup with five intersections. To achieve this, the probability for left turns in intersections was reduced

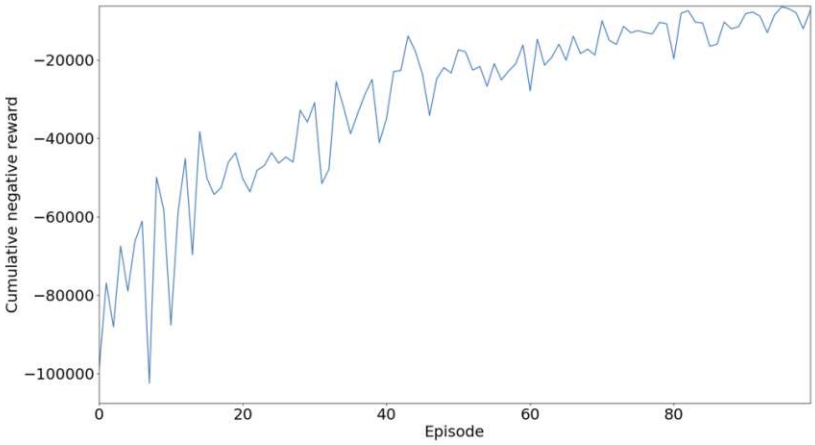

Figure 5.2: Training improvements for remodeled intersection

to 30% of turning vehicles going left instead of right (from 50% as used by Vidali et al.). Subsequently the straight advance settings which are NSA and EWA received longer green phases and the $NSLA$ and $EWLA$ green phases were shortened thus giving the fixed timings more of a green wave over all intersections. Figure 5.3 shows the resulting

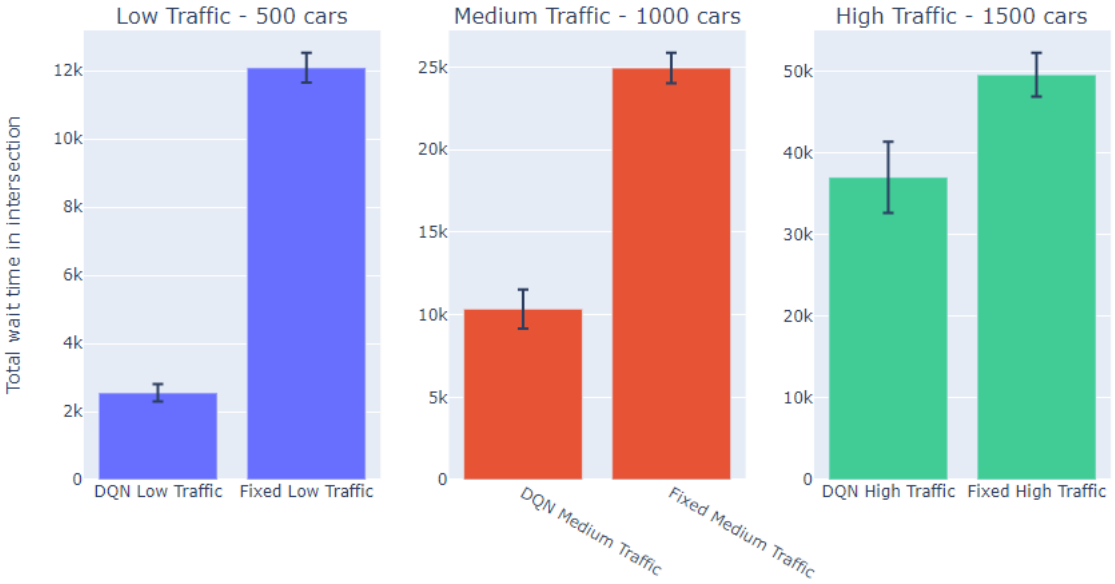

Figure 5.3: DQN Agent versus Fixed Time Intervals - Single Intersection

total wait times over all three traffic scenarios including the standard deviation on 30 1-hour episodes for each scenario. The findings of the repeat experiment are in line with the findings of Vidali et al. which shows the strength of the algorithm for low to medium traffic situations.

Collaboration-Type	**Sync-Type**	**Reward**	**G/Y Phase**
disjoint / optimal / complex / simple	cycled	cumwait	10s | 4s
Number of Cars	**Simulation-Environment**	**State Size**	**Hidden Layers**
4000	experiment	varying	varying

Table 5.3: Experiment Setup - Varying Collaboration Type Models

5.2 Experiment Setup - Evaluation

As mentioned previously the main goal of the experiment setup evaluation is the comparison of the proposed collaborative deep learning approach against the baselines of fixed time intervals and the non-collaborative solution developed by Vidali et al. [VCVB19]. The definition of the simulation environment used for this evaluation is discussed in Chapter 4. Due to the increased complexity of multiple intersections and the possible interacting effects they have on each other, the approach designed in Chapter 3 does not feature a single generalized solution but different design choices for collaboration and synchronisation between agents and as such these choices need to be considered for each implementation depending on their benefits and drawbacks. In order to provide a better understanding of these benefits and drawbacks and measure the actual impact they have on the solution the following section evaluates performance based on the different *collaboration types*, *synchronisation schemes*, impact of collaboration with *increasing distance between intersections* and the *robustness* of the solution based on the three different traffic distributions discussed in Section 4.3.1. Additionally the robustness section explores how the choice of traffic load during training effects the performance of the resulting model (i.e. how a model trained on mainly low traffic volumes performs in high traffic scenarios and vice versa). For all design choices models were trained with a given choice and the remaining model parameters fixed for all solutions. The results are compared to two baseline solutions. The first baseline solution is the fixed time interval agent which cycled through its phases based on the schedule $\{38, 4, 24, 4, 38, 4, 24, 4\}$. Secondly the non-collaborative solution by Vidali et al. is taken as a second baseline solution which also serves the purpose of evaluating in which situations collaboration does not provide sufficient improvements.

5.2.1 Collaboration Types - Evaluation

The collaboration types proposed in Section 3.2.1 define to what extent information of neighboring intersections is shared between agents. The three types *complex*, *simple* and *optimal* are compared to fixed time intervals and the non-collaborative solution, which are in the following denoted as *fixed* and *disjoint* respectively. Table 5.3 shows the training parameters set for all reviewed models. All four models were trained using identical synchronisation, reward function, green/yellow phases, and the number of cars of the

randomly generated training episodes was set to 4000. All models were trained for 100 training episodes of 4000 time steps each, which feature 500 training epochs for the deep neural network after each training episode for a total of 50.000 epochs. The resulting state sizes and width of hidden layers vary as each collaboration type has different input sizes depending on the type and number of neighboring intersections. For the center traffic light with four neighbors the input sizes for $[disjoint, simple, optimal, complex]$ are $[80, 88, 136, 408]$ according to the state representation of Section 3.2.1. Figure 5.4

Figure 5.4: Negative Reward per Episode - Collaboration Types

shows the negative rewards received per training episode over all 100 episodes. Despite the difference in collaboration all four models converge at roughly the same rate and reach similar negative reward values for their final training episode. This is to be expected as due to the training strategy the ϵ parameter which defines with which probability the agent chooses either a random action or uses the DQN to predict the best action starts off at 1 (resulting in exclusively random decision) is adjusted towards 0 (exclusively DQN predictions) with each training episode based on the total number of training episodes. For 100 training episodes this results in the ϵ parameter being adjusted by 0.01 towards 0 after each training episode. While the resulting models converge at about the same rate, the finale episode results of $[disjoint, simple, \mathbf{optimal}, complex] = [13621, 12113, \mathbf{7244}, 10843]$ show a slight improvement of all three collaboration types over the disjoint alternative. Especially the *optimal* collaboration type which ended the training phase with a substantial improvement of a factor 1.88 lower negative reward than the disjoint baseline.

While the training results show how well the design parameters allow for optimizing the reward function they do not guarantee performance in an actual traffic setting. To measure the performance of the models under traffic conditions that vary from the high traffic scenarios they were trained with, Figure 5.5 and Figure 5.6 show results of each model measured by the core metrics of total cumulative wait time per test case and mean vehicle wait time per test case. Measurements were taken based on a set of 30 randomly

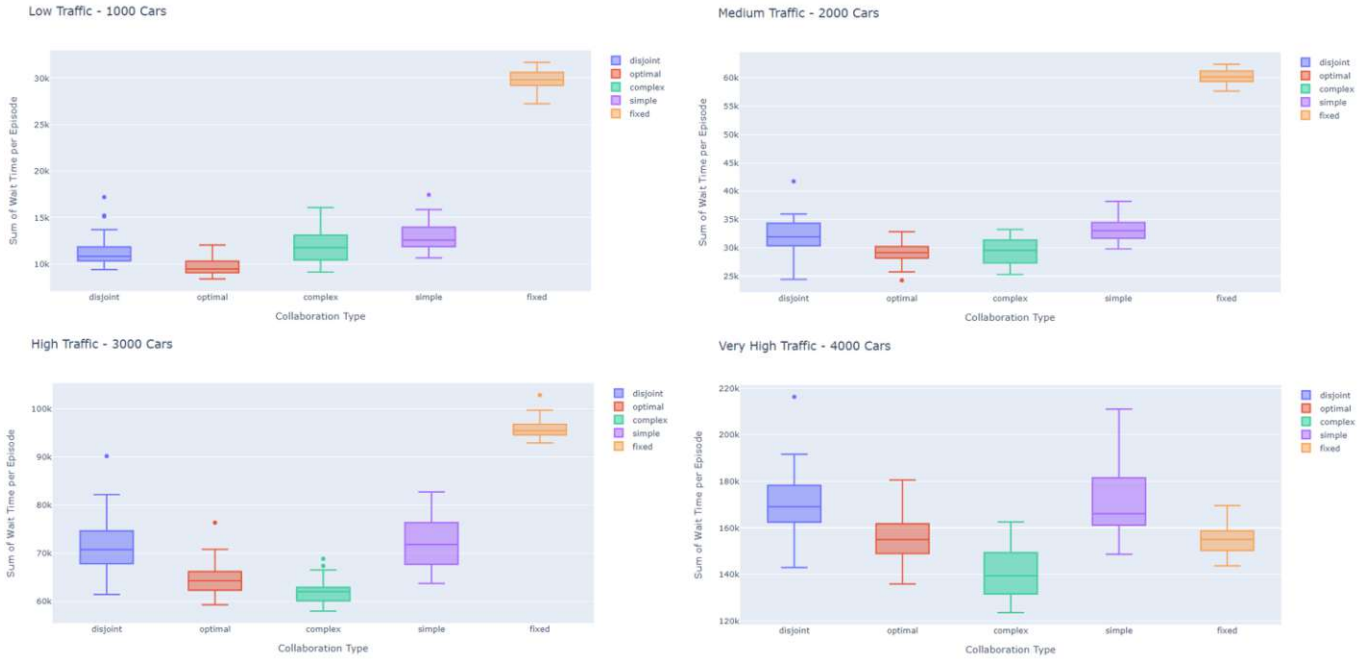

Figure 5.5: Total Wait Time over 30 Repeat Experiments - Collaboration Types

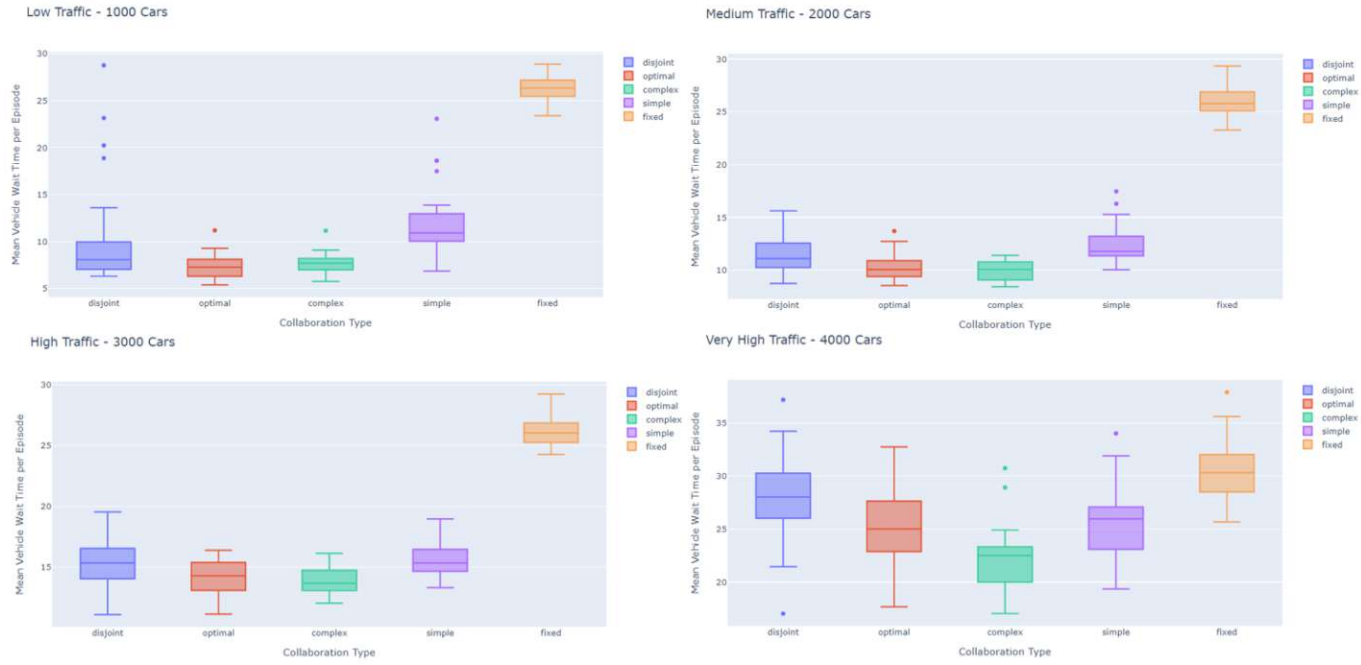

Figure 5.6: Mean Vehicle Wait Time over 30 Repeat Experiments - Collaboration Types

Collaboration-Type	**Sync-Type**	**Reward**	**G/Y Phase**
optimal	cycled / sync / async	cumwait	10s | 4s
Number of Cars	**Simulation-Environment**	**State Size**	**Hidden Layers**
4000	experiment	136	4x400

Table 5.4: Experiment Setup - Varying Synchronisation Scheme Models

generated test cases for each scenario (low traffic 1000, low traffic 2000, medium traffic, high traffic). Both for total wait times as well as mean wait time per vehicle, the smart traffic light agents outperform the fixed time interval by a large margin. While this is a promising result it should be noted that the phase times used were recommended by the *SUMO* framework and are not the result of a traffic study as would be the case in a real world example and it is thus expected to be less than optimal. The *simple* collaboration type which only encodes the currently active light phase of neighboring intersections appears to result in a decrease in performance even when compared to the disjoint model. While the *complex* model shows the most promising results for high to medium traffic situations it does drop off and become less stable as the traffic load decreases which might be an indication that the more complex model scheme results in overfitting to the high traffic scenario the model was trained with. Overall the most stable and well performing model is the one using *optimal* collaboration. The mean vehicle wait times have significantly less outliers over all scenarios when compared to the other models and it performs well in terms of total sum of wait time in all four evaluated scenarios improving on the disjoint baseline in all respects. Even if the complex collaboration type ahcieves the best performance, this comes at the expense of significant signalling overhead, model complexity and some overfitting making it less efficient for low traffic conditions.

5.2.2 Synchronisation Schemes - Evaluation

Synchronisation of agent decisions in a collaborative system such as the one discussed here is essential as it defines for how long a neighboring agent's decision remains valid and when the information becomes deprecated. The three synchronisation schemes discussed in Section 3.2.2 are *asynchronous*, *synchronous* and *cycled* decision making. Table 5.4 shows the training parameters for all models reviewed in this section. Again the models compared below are the baseline solutions using fixed time intervals and the *disjoint* solution which does not use collaboration among agents. All three models were trained using the *optimal* collaboration type and the same reward function, green/yellow phases and number of cars for training scenarios. Figure 5.7 shows the received negative rewards received during training. Again the models converge at roughly the same pace but since all collaborative agents use the *optimal* collaboration type, the disjoint solution performs

Figure 5.7: Negative Reward per Episode - Synchronisation Schemes

slightly worse as expected from the findings of Section 5.2.1. The exact values of the final training episode ($[disjoint, async, sync, \mathbf{cycled}] = [13621, 9222, 9250, \mathbf{7244}]$) show similar performance for *synchronous* and *asynchronous* decision making and further improvement for *cycled* synchronisation. Again the performance measurements shown

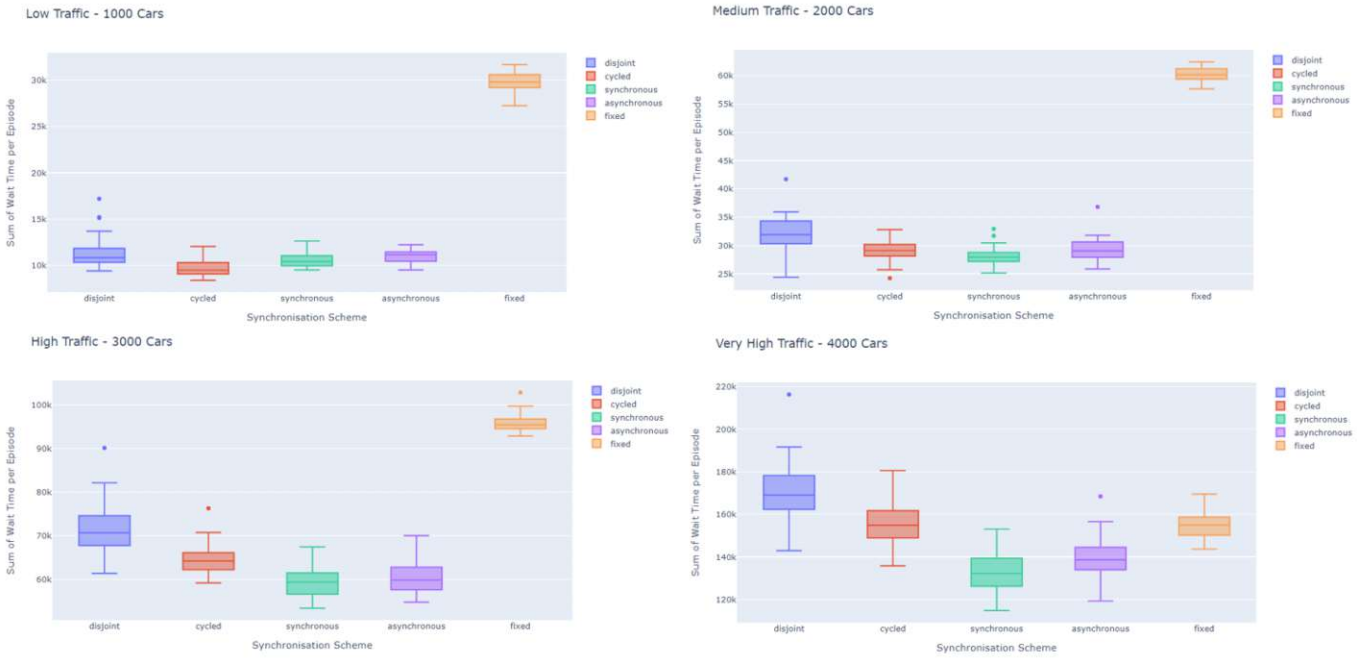

Figure 5.8: Total Wait Time over 30 Repeat Experiments - Synchronisation Schemes

in Figure 5.8 and Figure 5.9 were taken on 30 test cases for each traffic scenario. As expected, the mean wait times shown in Figure 5.9 are more stable than those shown

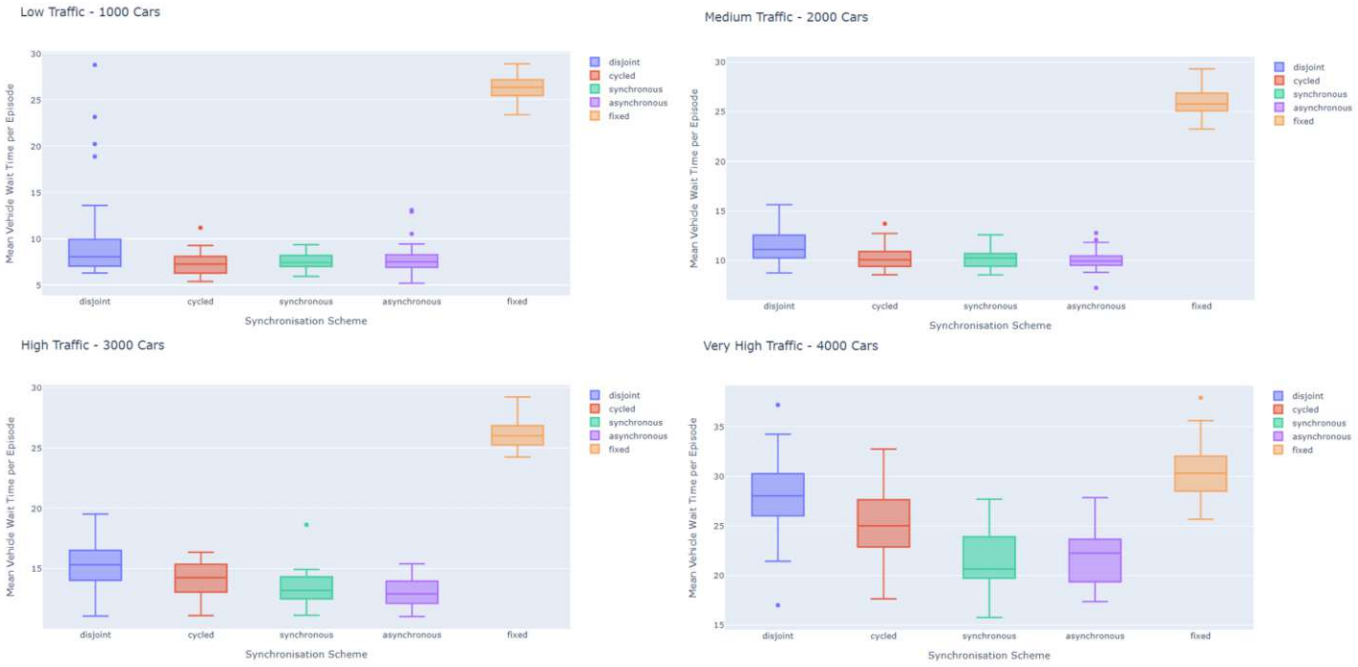

Figure 5.9: Mean Vehicle Wait Time over 30 Repeat Experiments - Synchronisation Schemes

in Figure 5.6 due to the collaboration type that was used. Analysis of the total sum of wait time in the cluster of intersections shows that for medium and high traffic the synchronous scheme is actually performing slightly better. This is potentially due to high traffic scenarios benefiting most from the green wave effect which is supported most by the synchronous scheme. Figure 5.8 also shows that for low traffic scenarios cycled synchronisation is preferable because adjacent traffic lights can react faster as they do not have to wait for a full cycle length before readjusting and can instead readjust after half a cycle of 7 seconds. Despite the fact that there is not really an imbalance in approaching vehicles in any direction, which is not often the case in the real world, synchronised decision making still performs really well and is expected to perform even better when there are certain imbalances in a given direction allowing for a decision hierarchy as discussed in Section 3.2.2.

5.2.3 Robustness - Evaluation

The robustness of the proposed solution is tested in two separate regards. First is the versatility of the algorithm to cope with traffic loads that differ from the scenarios it was trained on. To this end the comparison features two models. One was trained on the low traffic scenario (2000 cars per episode) and the second model was trained on the high traffic scenario (4000 cars per episode). Both models are tested for their downwards/upwards compatibility in dealing with scenarios that feature a much lower

Collaboration-Type	**Sync-Type**	**Reward**	**G/Y Phase**
optimal	cycled	cumwait	10s | 4s
Number of Cars	**Simulation-Environment**	**State Size**	**Hidden Layers**
4000 / 2000	experiment	136	4x400

Table 5.5: Experiment Setup - Varying Traffic Load during Training

and higher traffic load than what they were trained on respectively. The second part of the robustness evaluation explores robustness regarding traffic distribution differing from the Weibull distribution the model was trained on. This specifically refers to the double-Weibull and uniform distributions shown in Figure 4.8 and Figure 4.9 as opposed to the original Weibull distribution of Figure 4.7 on which the model was originally trained. The parameters of the two different models mentioned above are shown in Table 5.5. Both use the optimal collaboration type and cycled synchronisation with one having been trained exclusively on episodes featuring the low traffic scenario with 2000 cars over 4000 time steps, while the second one was trained on the very high traffic scenario with 4000 cars.

In order to measure the downwards compatibility of a model facing much lower traffic

Figure 5.10: Robustness on Lower Traffic Scenario

than what it was originally trained on, Figure 5.10 shows the results for the low traffic scenario featuring 2000 cars. In this graph *DQN - High Traffic Training* refers to the model trained on 4000 cars and *DQN - Low Traffic Training* refers to the one trained on 2000 cars. Interestingly the performance here is almost identical despite one having been trained on exactly this scenario while the high traffic DQN was trained on scenarios featuring twice the traffic load. Based on this the solution seems to be highly robust in regard to lower traffic scenario.

This robustness does not hold up in the opposite direction as the model trained on

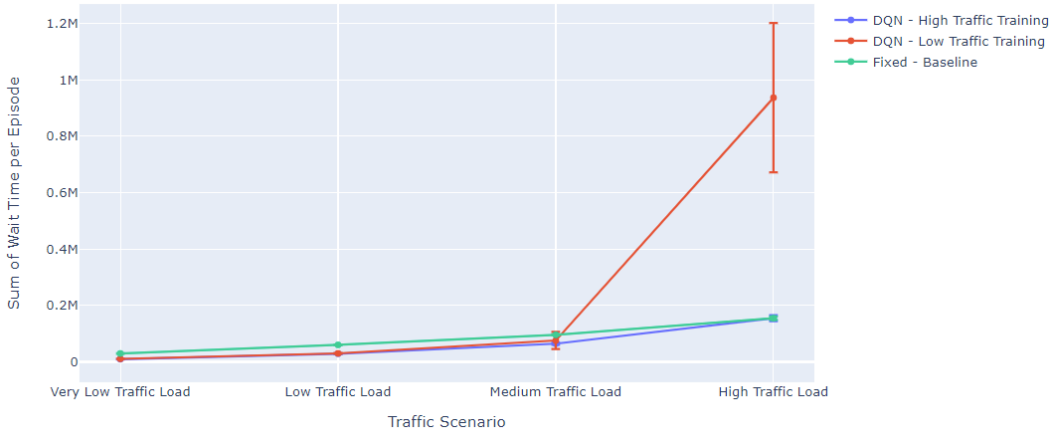

Figure 5.11: Comparison of varying Traffic Scenarios

low traffic can not cope with traffic drastically higher than what it experienced during training. Figure 5.11 visualizes this problem as the models performance completely breaks down on the high traffic scenario featuring 4000 cars (denoted *High Traffic Load* in Figure 5.11). Upon further investigation, the breakdown of performance is attributed to the agent facing state representations it has never experienced during training which results in more or less random decisions in these situations. In the worst case scenario this can lead to a complete deadlock where the state stops changing because no car can pass the intersection and the agents decision for the deadlock state does not allow for any cars to pass. An example of such a deadlock situation is shown in Figure 5.12 where the

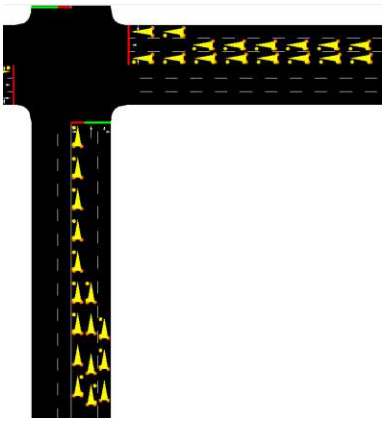

Figure 5.12: Possible Deadlock Situation

lanes are completely blocked resulting in no further state changes and the agents decision for NORTH-SOUTH-Advance does not result in any vehicle clearing the intersection. While this situation should never occur with a properly trained agent, it is important

to account for this problem by implementing a fail safe to clear potential deadlocks. A simple solution to preventing full deadlocks is checking if an agent decision resulted in no state change and the new decision is identical to the one before. In this case an agent can choose the action with the second highest Q-Value to unblock the intersection. The main insight however is the importance of having an agent experience the maximum expected traffic load during the training phase for it to learn how to handle them effectively as the training does not appear to result in overfitting for high traffic scenarios and instead still performs very good on low traffic scenarios as well. To evaluate the robustness in

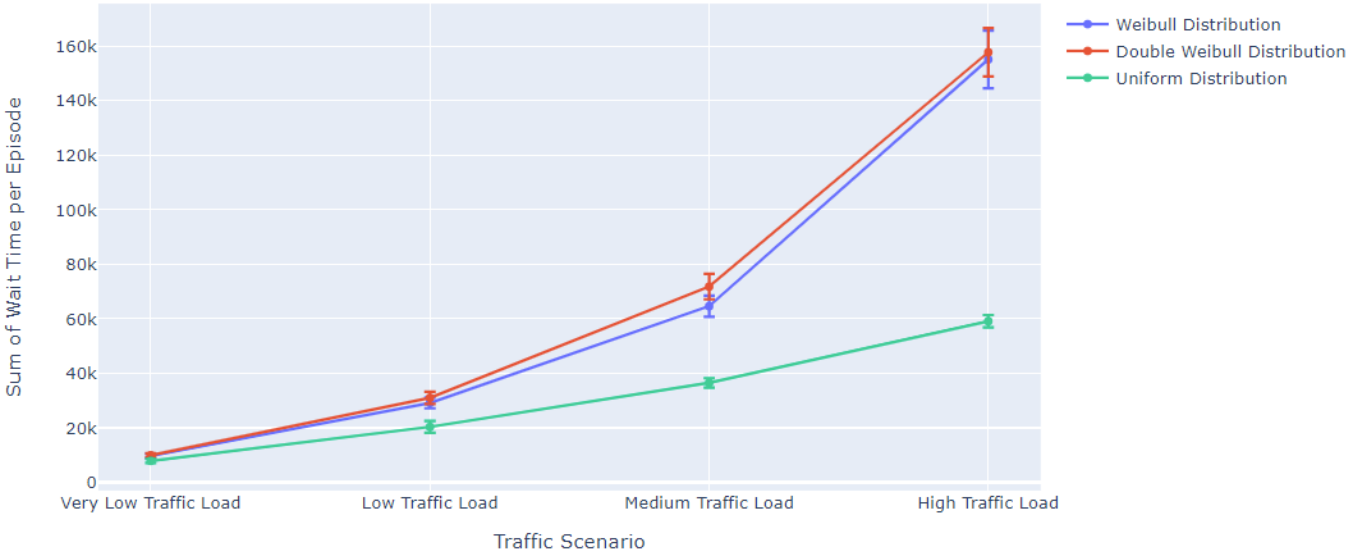

Figure 5.13: Robustness on varying Traffic Distribution

regards to the traffic distribution, the traffic scenarios with their respective number of vehicles were recreated with different underlying distributions. Thus for each of the four traffic scenarios (1000, 2000, 3000, 4000) 30 test cases were generated for both alternative distributions in order to compare the results and evaluate the performance under these changing circumstances. Figure 5.13 clearly shows that there are no performance drop offs due to changes in the distribution. For the uniform test cases, the more even distribution of traffic with lower peak numbers even resulted in an improvement of total wait time for the same number of vehicles over all traffic scenarios.

5.2.4 Distance between Intersections - Evaluation

In order to apply the concept of collaborative reinforcement learning agents to entire traffic grids it is important to measure the drop off in effectiveness with increasing distance between intersections to decide at which point parts of the grid can be viewed as disjoint due to collaboration no longer being effective. To evaluate this effect this section compares the improvement of the collaborative optimal solution over the non-collaborative disjoint solution for three experiment setups with distances between intersections of 100m, 200m

Collaboration-Type	**Sync-Type**	**Reward**	**G/Y Phase**
optimal	cycled	cumwait	10s | 4s
Number of Cars	**Simulation-Environment**	**State Size**	**Hidden Layers**
4000 / 5000 / 6000	experiment: **100m / 200m / 400m<**	136	4x400

Table 5.6: Experiment Setup - Varying Distance Between Intersections

and 400m respectively. The exact model parameters are listed in Table 5.6. All models are trained on their respective high traffic scenario as defined in Table 4.1 and tested for low,medium and high traffic load. As there are two low traffic scenarios defined for the 100m experiment setup, the one chosen here features 1000 cars. In addition a second model is trained for each experiment setup featuring the same parameters except for the collaboration type which is the disjoint solution of Vidali et al. Figure 5.14 features the

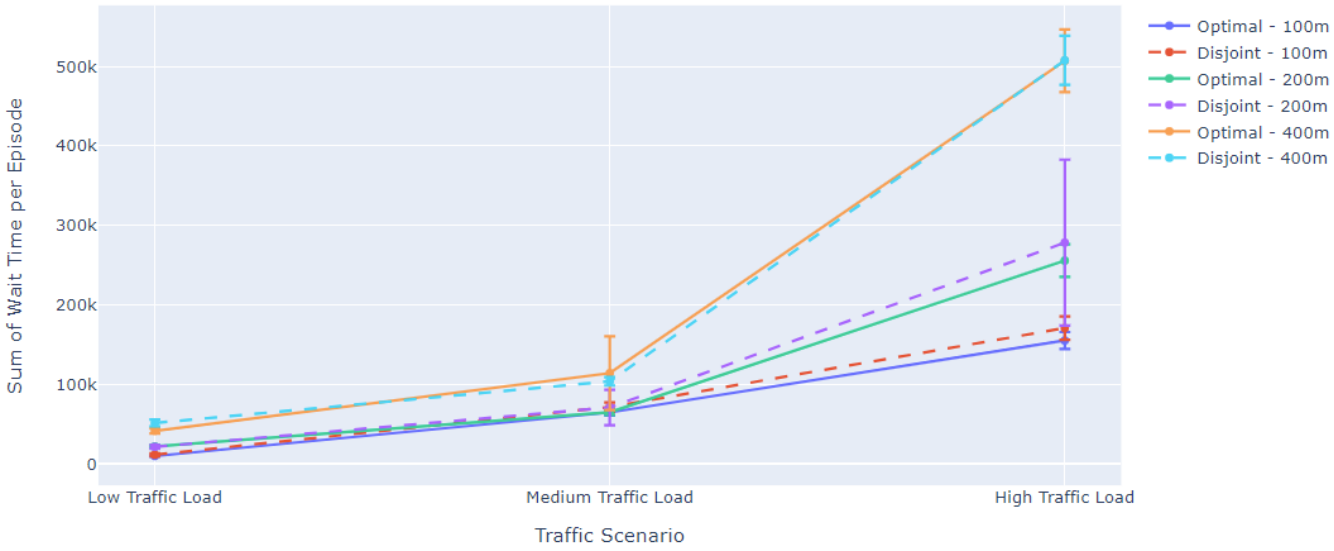

Figure 5.14: Collaborative vs. Disjoint Solution - Varying Intersection Distance

results of all three distance modes with the dotted lines showing the non-collaborative results. The evaluation clearly shows that while the optimal solution is better for the 100m and 200m case they are almost identical for 400m. This indicates that given the maximum allowed speed of the simulation which is $50km/h$ 400m is the cut off where the utility of collaboration between agents becomes negligible. To better visualize the exact factors by which the improvement drops off, Figure 5.15 shows these factors calculated for each of the 30 test cases. For the low traffic scenario the collaborative solution improved

Collaboration-Type	**Sync-Type**	**Reward**	**G/Y Phase**
optimal	cycled	**cumwait / queuelength**	10s | 4s
Number of Cars	**Simulation-Environment**	**State Size**	**Hidden Layers**
4000	experiment	136	4x400

Table 5.7: Experiment Setup - Queue Length Reward Function

the total wait time on average by approximately 14% over the disjoint solution which dropped to 8% for 200m and 4% for 400m. Respectively for medium and high traffic load these percentage values are $10\%, 5\%, 0\%$ and $9\%, 3\%, 0\%$. These values are highly

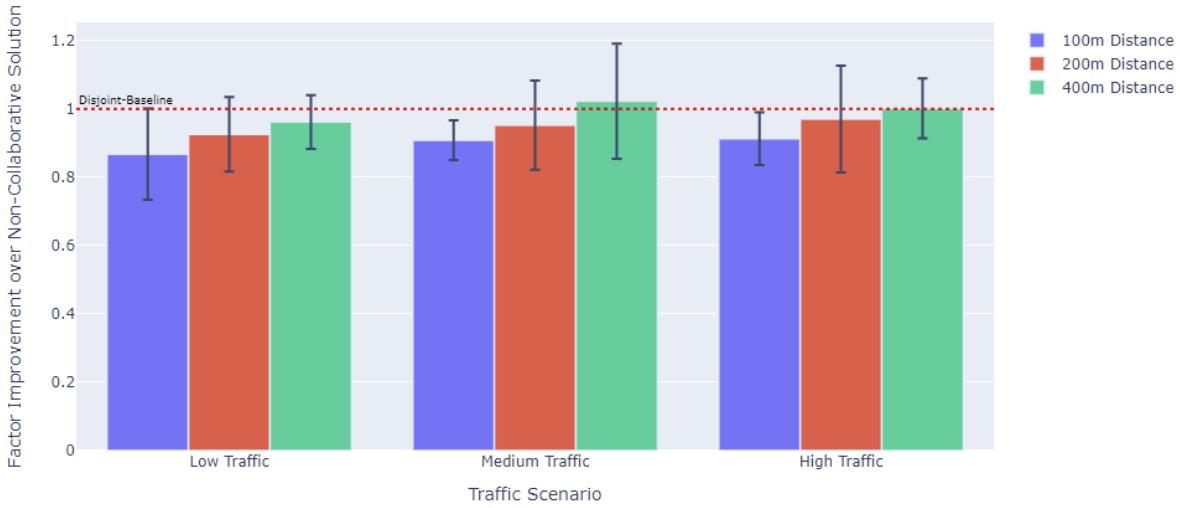

Figure 5.15: Improvement over Baseline Disjoint Solution

dependent on the phase time between agent decisions and on the speed limit of the roads between intersections but it shows the importance of considering this effect when implementing a collaborative system.

5.2.5 Alternative Reward Function - Evaluation

This subsection explores the viability of using a simpler reward function which uses the queue lengths as denoted in Equation 3.3 as opposed to the cumulative wait time per vehicle. The measurement of exact cumulative wait times for each vehicle waiting in a given intersection can become difficult in a real world setting which is why this is discussed as an alternative if the former function proves infeasible in certain scenarios.

The two models that are compared here use cycled synchronisation, the optimal collabora-

tion type as defined in Table 5.7 and only differ in the reward function that is used during training. Since the two models use different reward functions the negative reward per

Figure 5.16: Wait Times of Training Episodes - Alternate Reward Function

episode is not comparable and Figure 5.16 shows the total sum of wait time per training episode instead. It is important to note that this is the exact metric for which the the primary reward function attempts to optimize its policy which makes the similarity in resulting performance surprising. The queue length metric encodes less information in the sense that it does not punish for instance a single vehicle waiting indefinitely as this car will only represent one vehicle in the queue on each decision step. This problem is solved by using cumulative wait time as a vehicle waiting for a long time will keep increasing the negative reward until it is allowed to pass the intersection. Despite this the alternate metric achieved very similar results during training with the small caveat that the model using the queue length metric was more unstable during its final 20 training episodes where performance dropped on a few episodes. While the results look promising in regard to the training reward the actual evaluation on randomised test data shows a crucial problem of the alternate reward function which is the deadlock problem shortly discussed in Section 5.2.3. As shown in Figure 5.17 the results of the alternate reward function show a very large standard error due to complete deadlocks occurring in certain test cases. These deadlocks occur even for the high traffic scenario on which the model was trained.

Further research is required to determine how to alleviate this deadlock problem, but if the test cases where the model entered a deadlock are filtered out the results as shown in Figure 5.18 while comparable and below the fixed time interval baseline still falls behind the model using the cumulative wait time reward. Ultimately the results indicate that the benefit provided by the simplicity of the queue length is outweighed by the drop in performance. It is important to note that the complexity of the reward function only factors in when training the model further. It is important to note again that if a model is considered fully trained it can operate in any environment that can provide the state representation it was trained on and does not require the information of the reward

Figure 5.17: Alternate Reward Function - Results with Deadlocks

function. The main benefit of having a reward function that can be measured in a real life implementation is to continuously generate training experiences and further train the model.

Figure 5.18: Altenrate Reward Function - Results without Deadlocks

5.3 Real World Example - Evaluation

As discussed in Chapter 4 the real world example consists of three adjacent traffic lights in the city of Christchurch NZ which are the intersections of *Montreal St and Hereford*

Collaboration-Type	**Sync-Type**	**Reward**	**G/Y Phase**
optimal / disjoint	sync	cumwait	16s | 4s
Number of Cars	**Simulation-Environment**	**State Size**	**Hidden Layers**
1700	christchurch	43 / 30	4x200

Table 5.8: Christchurch Setup - Optimal and Disjoint Model

St, *Montreal St and Worcester St* and *Montreal St and Gloucester St*. In addition to providing the exact traffic counts for all three intersections on the 26.03.2018 through the intersection traffic counts database[1] the Christchurch City Council Traffic Signals Team upon request also provided the exact phase time logs for the entire 24 hours span on 26.03.2018 given the information shown in Figure 5.19 as the traffic lights are not completely fixed but may also react as actuated agents. This information was used to remodel the intersections and generate the test cases and run the experiments on the resulting simulation.

	A	B	C	D
1	Phase	Duration (s)	Start	End
2	?	22	0:00:00	0:00:22
3	B	20	0:00:22	0:00:42
4	<A>	27	0:00:42	0:01:09
5	B	19	0:01:09	0:01:28
6	<A>	24	0:01:28	0:01:52
7	B	20	0:01:52	0:02:12
8	<A>	26	0:02:12	0:02:38
9	B	19	0:02:38	0:02:57

Figure 5.19: Phase Time Logs - Christchurch NZ

For the evaluation of the real world example, the parameters selected for the collaborative model were based on the insights of the experiment evaluation. As shown in Table 5.8 chosen state representation is the optimal collaboration type as it showed the most stable results across all traffic scenarios. Furthermore for the synchronisation scheme the synchronised type was chosen as the Christchurch example features a large main road which only allows for one-way traffic. This means that there is a sensible decision hierarchy that can be used allowing for the algorithm to utilize the green wave effect. The one-way traffic is allowed in the SOUTH-NORTH direction and thus the synchronized decisions are made first by the southernmost agent and with minimal delay by the following neighbor and so on. For the green/yellow phase time the green phase time was increased by 6 seconds allowing for better utilization of the green wave effect with less phase changes overall. Lastly the hidden layer width was adjusted to 200 as the input size for the experiment simulation was a lot smaller with 43 bits for the collaborative solution

[1] https://ccc.govt.nz/transport/improving-our-transport-and-roads/traffic-count-data/intersection-traffic-counts-database/

and 30 for the disjoint solution. The number of cars generated in the training phase are 1700 in line with maximum traffic load measured by the the Christchurch City Council Traffic Signals Team. Additionally the underlying distribution for traffic generation was switched from the Weibull distribution depicted in Figure 4.7 to a uniform distribution in accordance with the traffic distributions depicted in Figure 4.13. Similar to the experiment

Figure 5.20: Negative Reward per Episode - Christchurch Simulation

setup where the collaboration already showed decent improvements in the negative reward received in the training phase this was also the case for the real world example. The exact values of the last training episode respectively was: $[disjoint, \mathbf{cycled}] = [7284, \mathbf{6165}]$. It is important to note that the real life experiment features a system of intersections that is simpler than the fully connected experiment network with equally distributed traffic load. For all three traffic lights there are only two light phases and the main road is a one-way street. Furthermore the intersections timings are optimized and regulated by the Sydney Coordinated Adaptive Traffic System[SD80]. Due to these circumstances this highly optimized semi-fixed implementation performs a lot better than the fixed time solution used in Section 5.2. This is confirmed by Figure 5.21 which shows the sum of wait times per test case. Here the fixed time solution actually outperforms the disjoint alternative on all three evaluated traffic scenarios. The collaborative solution shows promising results for the low and medium traffic scenarios where the total waiting time was reduced significantly when compared to the fixed time solution but the evaluation clearly indicates that the fixed time interval becomes increasingly efficient as the traffic load increases. This is in line with the findings of Vidali et al. [VCVB19] where the fixed time solution outperformed the algorithm on very high traffic load in a single intersection. In regard to perceived efficiency measured by mean wait time per vehicle as shown in Figure 5.22 there is not really any significant improvement. While the collaborative approach resulted in more stable wait means the fixed time solution resulted in a lower median wait time over the 30 experiments. In order to better visualize the results, Figure 5.23 shows the improvements by factor over the fixed time results. For

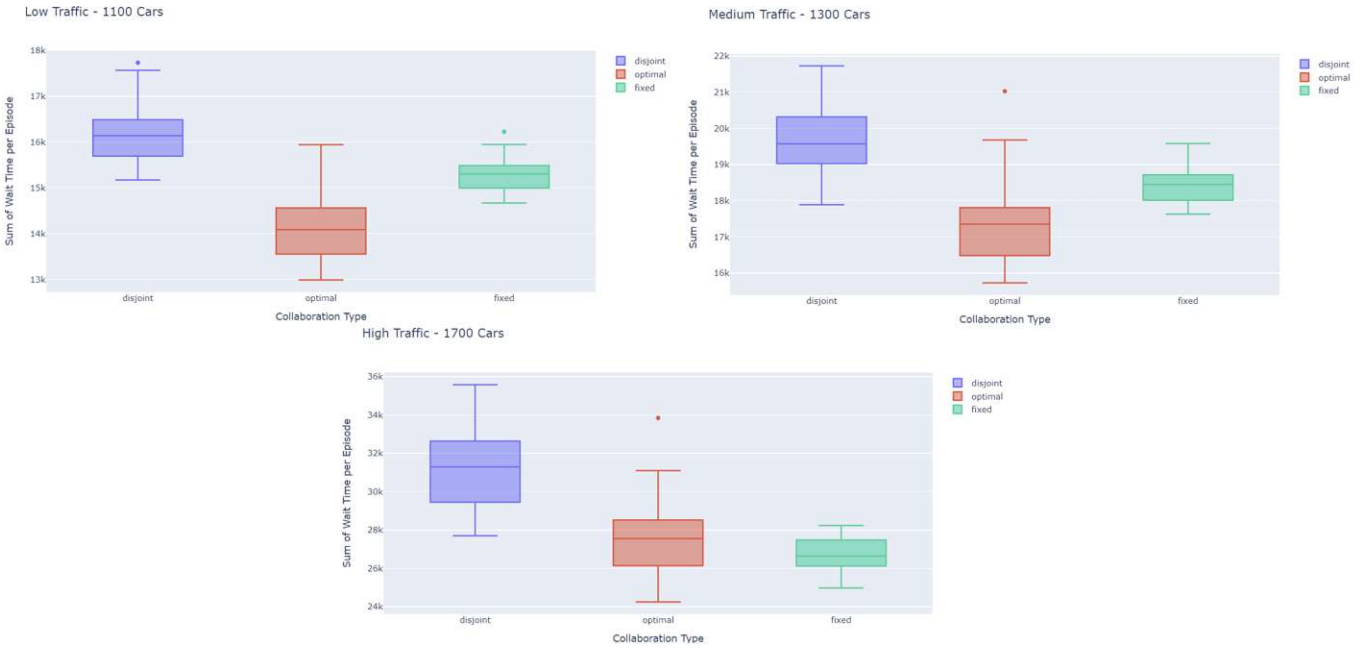

Figure 5.21: Total Wait Time over 30 Repeat Experiments - Christchurch Simulation

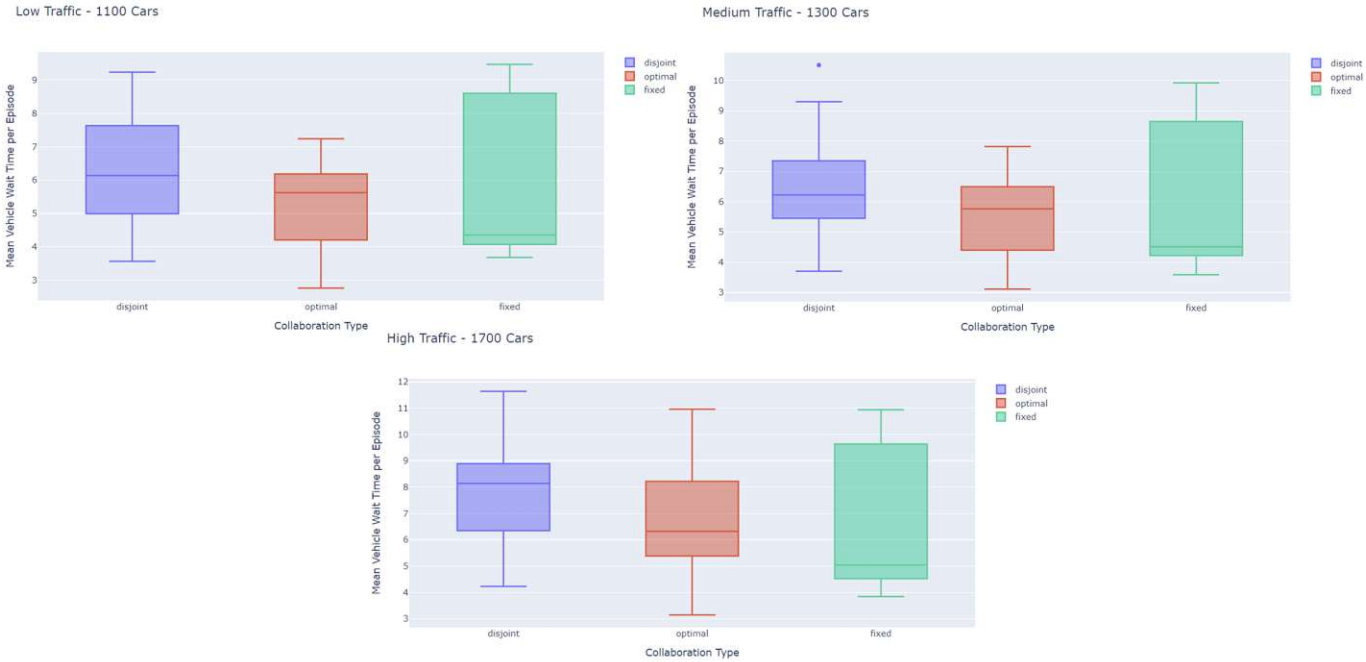

Figure 5.22: Mean Vehicle Wait Time over 30 Repeat Experiments - Christchurch Simulation

this comparison each test case result from the collaborative and the disjoint solution was compared to that specific result with fixed time intervals. This shows that on average the collaborative solution improves the total wait time by $10\% - 13\%$ for low and medium traffic while the non-collaborative solution performs $30\% - 32\%$ worse. In case of the high traffic scenario the collaborative algorithm performs 0.5% worse on average.

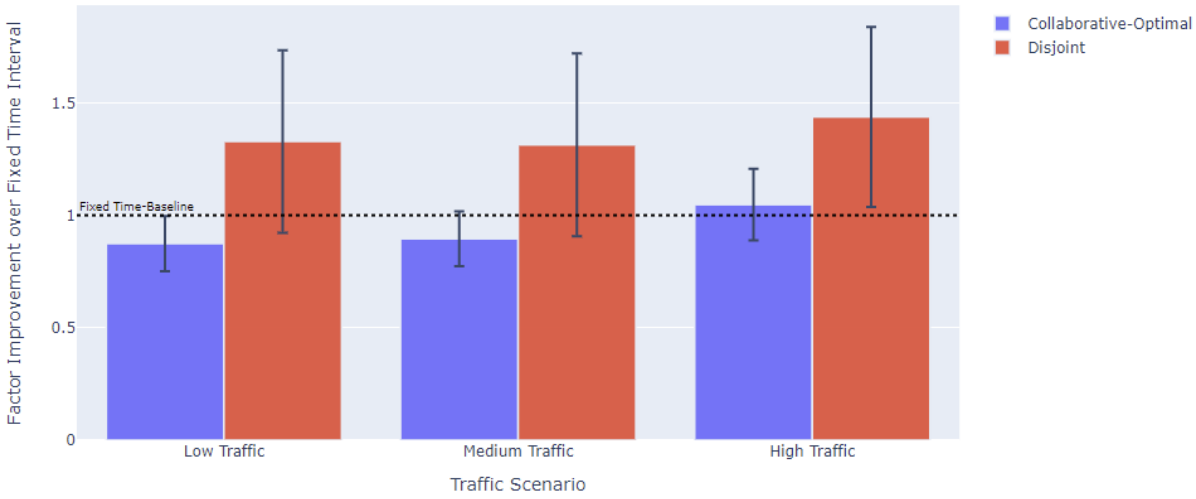

Figure 5.23: Improvement over Optimized Fixed Time Intervals

Overall it can be said that the collaborative solution improves the total wait time of the intersection for low to medium traffic situations over fixed time intervals while still achieving similar results for very high traffic load even for the case of the optimized Christchurch NZ intersections. Here the non-collaborative solution proposed by Vidali et al. falls behind showing that an approach that does not utilize collaboration in closely connected intersections can not compete with an optimized fixed time interval even for low traffic scenarios.

CHAPTER 6

Real Life Applicability

The aim of this chapter is to summarize and discuss the real life applicability of the proposed algorithm by looking at the implementation steps and the technology required to migrate the system from the simulation framework to the actual intersections it is supposed to manage. A core issue in machine learning solutions is the famous cold start problem which is most prominent in recommender systems[LKH14]. In the context of traffic management this poses a problem if the agent is deployed before training and is supposed to learn within a real life environment resulting in terrible traffic management until the agent improves. One of the main benefits of the proposed solution is that due to modern traffic micro-simulation frameworks such as *SUMO* this is a non-issue here, as the initial training can be exclusively done within the simulation before the system is migrated to the real world. Thus the applicability for real life implementation is structured into two main parts which are first the conceptualisation and training within the simulation framework and secondly the migration of the pre-trained solution to the real world.

6.1 Conceptualisation and Training

This section summarizes the steps required for conceptualizing and implementing a solution for a given set of intersections on the exemplary implementation of the Christchurch NZ intersection, which has not only provided performance measurements of the proposed solution against a real optimized system of traffic lights, but also proved the collaborative Q-Learning solution can quickly be adapted to a given set of real intersections. The conceptualisation and training of the system consists of three core components which are the construction of the simulation environment as closely to the real world intersections as possible, the definition of the traffic distribution and traffic load either based on actual traffic counts or approximation and lastly the selection of collaboration type and preferred synchronisation for each agent in the system. The selection of the collaboration type

here also includes the adaption of the state and action space based on the light phases, number of lanes and allowed turns as discussed in Chapter 3.

The construction of the simulation , which is the first step, is straight forward and only requires knowledge of the number of lanes, light phases, exact road lengths and the exact traffic rules of the given network which includes allowed turns per lane and also the speed limit. For the intersections of Christchurch NZ the information of road measurements and traffic rules was retrieved using google maps and google streetview and the light phases were confirmed by the Christchurch City Council Traffic Signals Team. This information is used to construct the system in a given micro simulation tool like *SUMO*. In essence, any micro simulation tool can be used that meets the two core requirements which are identification of exact vehicle positions at any given time step and tracking of wait times of each vehicle to allow for calculation of the cumulative wait time reward metric.

Traffic distribution and maximum traffic load is the second requirement for the conceptualisation phase. As shown by the evaluation in Chapter 5 it is important to train the agent on the maximum expected traffic load for optimal results since the trained agent scales well to lower than expected traffic loads but can not handle scenarios that go far beyond the maximum traffic load it experienced during training. Furthermore the simulation requires approximate probabilities for each possible turn to randomly generate traffic for training and testing. Idealy these probabilities come from traffic counts taken in the real intersection but the evaluation has shown that the agent is robust to changes in the traffic distribution which promises decent results even when the simulated traffic is only based on a rough estimate. Figure 4.4 for instance contains all the information required for randomly generated traffic in the Christchurch NZ example.

Lastly, the non-fixed design decisions of the collaborative system need to be defined. These design decisions include the adaptation of the state space, the adjustment of the agent's hidden layer width and the choice of collaboration type and synchronisation scheme. While the state space of a single agent in the experiment setup featured 20 cells per lane (10 for straight and right-turn traffic and 10 for left-turn traffic) this number is not fixed and has to be adapted to the road length, desired granularity and most importantly the available light phases. The choice of discretization into state-cells should be made based on which lanes are relevant to certain light phases. In the example of the experiment setup the intersections have two separate light phases which required discretization into two cell-lanes as shown in Figure 6.1. The example of Christchurch NZ on the other hand had only one light phase for NORTH-SOUTH bound traffic as shown in Figure 6.2 which is why a single cell-lane is sufficient to convey the required information of the state space. The final design decisions are the selection of the collaboration type and the synchronisation scheme. For the collaboration types, the evaluation has shown very promising results for the complex and the optimal collaboration type, but over all there is no clear cut answer to which combination of collaboration type and synchronisation

scheme is the best choice and they should instead be viewed as a tool set with different strengths and weaknesses. To this end, the evaluation done in this thesis provides an overview of potential benefits and drawbacks to help with these design decisions, but further research is certainly required on these topics.

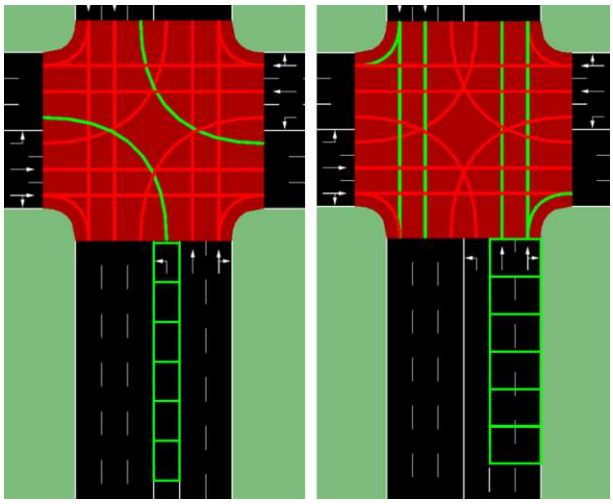

Figure 6.1: Cell Discretization - Experiment Setup

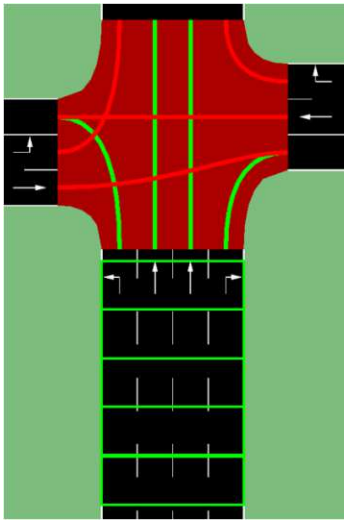

Figure 6.2: Cell Discretization - Christchurch NZ

6.2 Real World Migration

The solution proposed by this thesis keeps the conceptualisation, development and training fully within the given simulation framework, which means that this first phase

produces a fully trained agent without any hardware requirements, that can later be deployed to an intersection as is. This section gives an overview of the challenges and requirements for deploying the trained agents in the real world with possible technologies that could be used to solve these issues. While this thesis does not discuss a solution that continuously improves based on experiences after deployment in the field, this could be added in future work and the necessary requirements are also briefly mentioned here. For the migration of the trained agents to the real world there are three core problems that need to be addressed. The first challenge is recreating the state-representation in a real world setting which means detecting cars based on the lane discretization which splits the lane into cells and detects the presence of vehicles in these cells. The second challenge is allowing communication between this mechanism and the actual agent which controls the light phases. With these two problems solved, the system can be migrated to the real world for the non-collaborative solution proposed by Vidali et al.. For the collaborative solution proposed by this thesis, the final requirement is communication between adjacent intersections in order to share the currently active light phase and state representation in real time. For the sake of this discussion the setting of the implementation is limited to traffic with a speed limit of $50km/h$. The evaluation has shown that for the given speed limit, collaboration becomes ineffective at around $200m - 300m$, which means that a system can be viewed as disjoint and communication is no longer required beyond that range. Thus, a fully operational collaborative solution has to be capable of communicating with adjacent traffic lights within this $200m - 300m$ range.

State-Representation is the first and possibly the most challenging problem for a real world implementation of this system. It should be noted that this is also one of the strong points of the solution proposed in this thesis as the state-representation was kept simple compared to other state-of-the-art reinforcement learning approaches, such as the solution proposed by Kumar et al.[KMGK21] which not only requires vehicle position for cell discretization but also exact speed measurements of each vehicle within the system. Thus, the only required information is presence of a vehicle in a given cell. In recent years a lot of research has been done in the field of image based vehicle recognition, and as shown in the work of Wang et al.[WZSZ09] which was done in 2009, this problem could already be considered partially solved in terms of vehicle recognition from a camera mounted sufficiently high above a given intersection to provide a birds eye view. While AI based image recognition is capable of providing the required information, there are two main weaknesses of such an implementation which is on one hand a decrease in accuracy in poor lighting or weather conditions, for instance at night or during heavy rain, and on the other hand the possible difficulty to find a spot where the cameras can be mounted to provide a sufficiently wide range of view to cover all of the required cell discretization. An alternative solution could be proposed using small and cost efficient wireless magnetic sensors as discussed by Sifuentes et al.[SCPA11]. These small sensors could be used to directly represent each cell with a single sensor per cell. Liangliang et al.[LZXJ19] proposed a wireless cloud based solution that utilizes these sensors to measure occupancy in parking spaces as shown in Figure 6.3 which in essence solves

the exact problem of cell discretization for the proposed algorithm. The main benefits of this solution are its resistance to poor lighting and weather conditions and the fault tolerance provided by a modular system, as a single faulty sensor will only result in missing information of a single cell. As a drawback, in a city wide solution, maintenance of these sensors could become very expensive which is why further research is required on their life cycle and feasability.

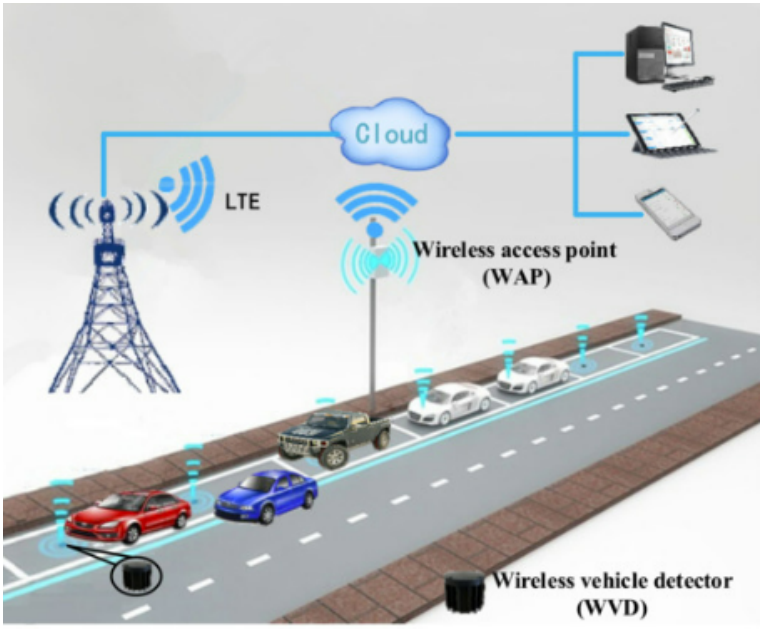

Figure 6.3: Wireless Vehicle Detection (image by Liangliang et al.[LZXJ19])

Intersection Communication poses a less complex problem as this has been discussed in length in literature for traffic regulation ([DMP+21],[LZXJ19],[ZYC09]) and is used for instance in common adaptive traffic light systems that use inductive loops to detect vehicles in the intersection or similar systems that require communication of wireless components in a network. This problem is also addressed by the actual solution implemented in the real world example of Christchurch NZ where SCATS (Sydney Coordinated Adaptive Traffic System) already solves this problem. The system shown in Figure 6.3 also covers this problem using wireless sensors. The total communication cost of information gathered within the intersection is also not problematic as the discussed solution uses at most 20 cells or 20 bits of information per incoming lane.

Collaboration Communication is the third core problem of a potential real life implementation. Due to the design of the system, collaborative communication can be restricted to only the immediate neighborhood which keeps the total load of received and transmitted communication small. Even for the complex collaboration type which utilizes all of the available state information from neighboring intersections this only requires

sending of 82 bit of information to each neighboring agent (or 328 bit of transmitted information in the case of four neighboring agents) and 328 bit of received information from four neighboring agents at each decision step, which in the case of the experiment discussed in this thesis are 14 second intervals. Thus, the total cost of communication required for the most expensive collaboration type is a total of 656 bits every 14 seconds. Also the size of the fully trained complex model is around $75MB$ which easily fits on common microcontrollers. The optimal collaborative model is even more efficient with a size of just $8MB$. A fully collaborative system could thus be constructed from a single microcontroller in each intersection that communicates with neighboring controllers with a built in LTE module or in case of physically close intersections even a standard 802.11ax wireless LAN connection with repeaters. Alternatively the system can also be built on already existing infrastructure which in the case of Christchurch NZ would already be fully provided by SCATS[1]. Here communication in a distributed system of all intersections has already been implemented based on the scheme shown in Figure 6.4. Overall, SCATS generally provides all of the required infrastructure with the only exception being the range of the inductive loop technology used for vehicle detection. While the system does utilize this technology it only covers the front of the queue to detect the presence of a waiting vehicle in a given lane. For further research it would be sufficient to extend this infrastructure to allow for full lane discretization to build a real life prototype.

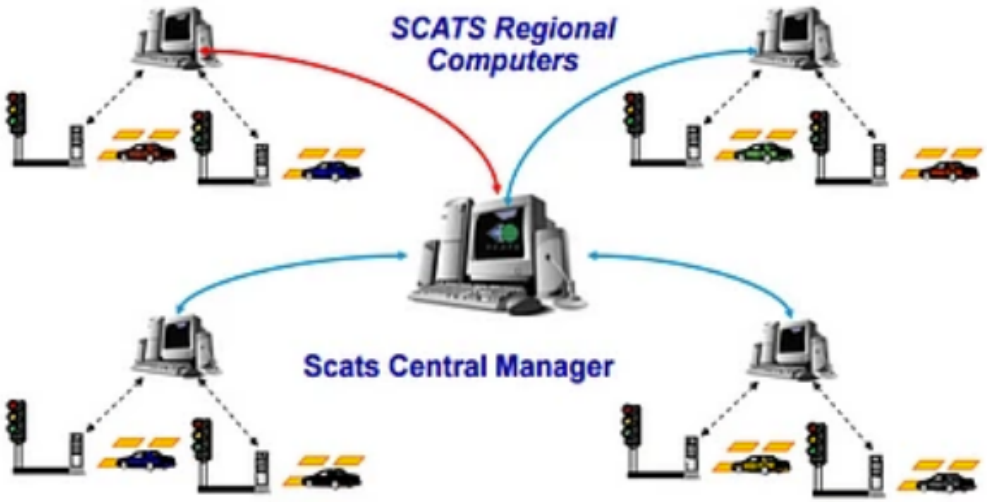

Figure 6.4: Sydney Coordinated Adaptive Traffic Systems

[1] https://www.scats.nsw.gov.au/

CHAPTER 7

Conclusion and Discussion

In this final conclusion chapter the three research questions will each be revisited and answered in detail based on the insight gained from the evaluation of the algorithm. Lastly, all additional findings regarding benefits and drawbacks of the proposed solution will be summarized and discussed along with possible future work and improvements.

7.1 Research Question 1 - Non-Collaborative Approach

Does the performance improvement of state-of-the-art reinforcement learning solutions for smart traffic lights proposed for single intersections hold for more complex traffic light grids with multiple intersections? Specifically is there a statistically significant drop in performance improvement over fixed time intervals? The goal of this research question is to highlight a weak point of state-of-the-art solutions that are implemented and tested only for single intersections and confirm the need to further improve on them by utilizing the potential for collaboration, as an intersection can very rarely be seen as an isolated system. To answer this question the Q-learning approach proposed by Vidali et al.[VCVB19] was implemented, the findings for a single intersection were confirmed and the solution was applied to a more complex grid of five intersections to measure if the improvement over fixed time intervals holds in this system. While the results of Chapter 5 show great improvement over the fixed time intervals for both the single intersection as well as the the grid of the experiment setup these results are not a direct indicator for real life results. The purpose of the fixed time interval used in the experiment setup is to provide a baseline by which the results of a single intersection and the selected grid of five intersections can be compared. Unlike with the real life example discussed in Section 7.3 the fixed time intervals used here are not optimized for the grid and as such the magnitude of improvement is not expressive. Figure 7.1 shows the factor by which the total wait time is decreased over the fixed time interval across the 30 test cases per traffic scenario with the error bar

Traffic Scenario	Fixed Intervals	Disjoint - Single Intersection	Disjoint - Five Intersections	Performance Drop-Off (from Single to Five)
Low Traffic	1	0.14 ± 0.01	0.27 ± 0.04	**92.85**%
Medium Traffic	1	0.27 ± 0.03	0.48 ± 0.04	**77.78**%
High Traffic	1	0.42 ± 0.06	0.63 ± 0.06	**50**%

Table 7.1: Factor Improvement: Disjoint over Fixed Time

depicting the standard deviation. For instance a factor of 0.3 on medium traffic can be interpreted as the solution achieving on average $0.3 * waittime_{fixedtime}$ for the 30 test cases generated with the medium traffic scenario. The exact values of the improvement

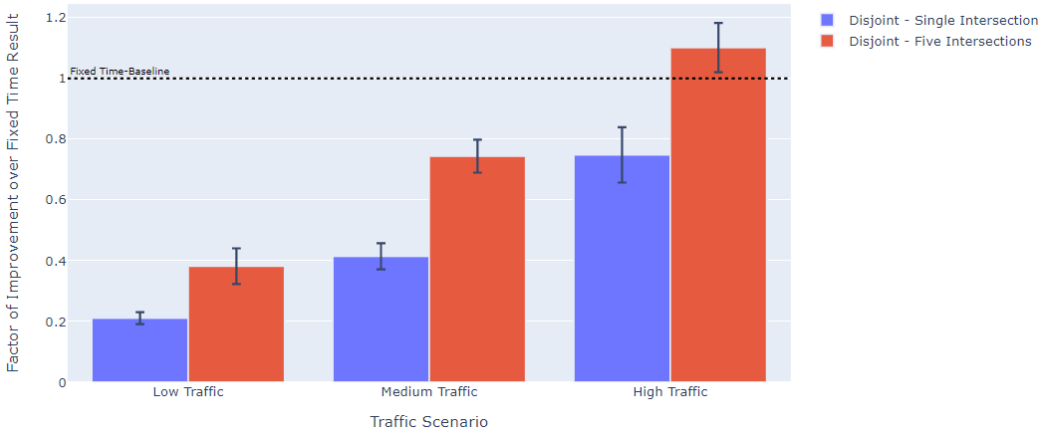

Figure 7.1: Improvement over Fixed Time Interval with Disjoint Solution

are listed in Table 7.1 and it is shown that the loss in performance improvement for all three traffic scenarios is certainly significant. The research question is answered by the fact that the performance improvement of the disjoint algorithm does not hold up in a more complex system in regard to the metric of total wait time. This loss in performance decreases with increasing traffic load but it is certainly significant for all scenarios. It could be argued that simply approximating low, medium and high traffic scenarios for comparison of a single intersection to five intersections is not accurate in measuring the drop in performance but it should nonetheless be considered as an indicator that the performance does not hold. Additionally this finding is reinforced by the evaluation of the real world example in Section 5.3 which has shown that the state-of-the-art solution of Vidali et al.[VCVB19] can not compete with highly optimized fixed time intervals in multiple intersections contrary to the findings of Vidali et al. in a single intersection.

Traffic Scenario	Disjoint	Optimal	Complex	Simple
Very Low Traffic	2279.61 ± 865.71	**1937.79** ± 408.63	2392.61 ± 898.48	2585.15 ± 947.16
Low Traffic	6400.77 ± 1334.86	**5823.16** ± 888.58	5880.28 ± 885.54	6657.17 ± 1285.20
Medium Traffic	14267.23 ± 2600.15	12918.40 ± 1669.07	**12393.26** ± 1370.95	14392.04 ± 2618.69
High Traffic	34151.01 ± 5573.055	31030.95 ± 4691.27	**28172.29** ± 4124.42	33925.44 ± 5951.16

Table 7.2: Average Sum of Wait Time per Intersection - Collaboration Types

7.2 Research Question 2 - Improvement by Collaboration

Does collaboration among agents in a grid of traffic lights lead to a significant improvement in either cumulative wait time for the entire cluster of intersections or the average cumulative wait time per vehicle over non-collaborative agents?

To answer the second research questions the results for total wait times within the intersections and the mean wait times per vehicles are again summarized in Figure 7.2 and Figure 7.3 respectively. As discussed in Chapter 5 the best performing collaboration type for very low and low traffic was the optimal type while the complex type showed the best results for medium to high traffic scenarios. Overall the most stable model was the one built using the collaborative optimal type which is why this was chosen to be compared to the disjoint solution in terms of statistically significant improvements in this section. The exact values of the test results are shown in Table 7.2 and Table 7.3.

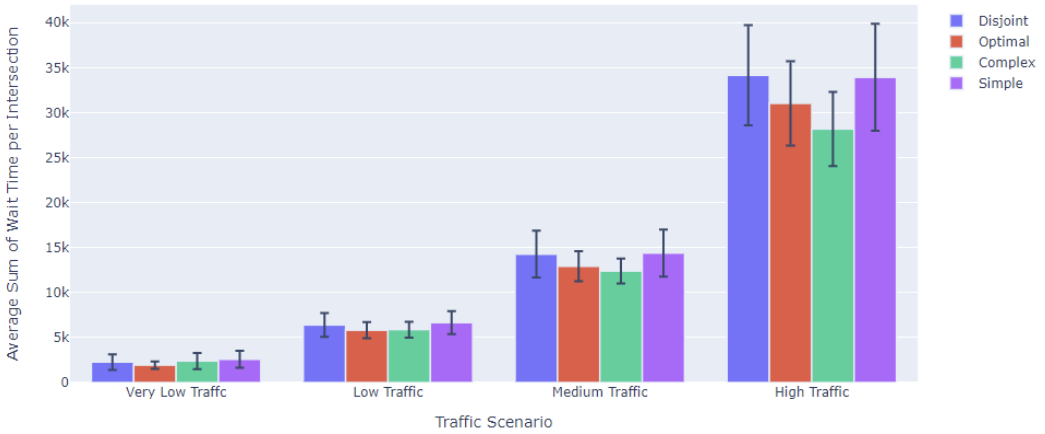

Figure 7.2: Average Sum of Wait Time - Disjoint Solution and Collaboration Types

Since the criterion used for this research question is a statistically significant improvement

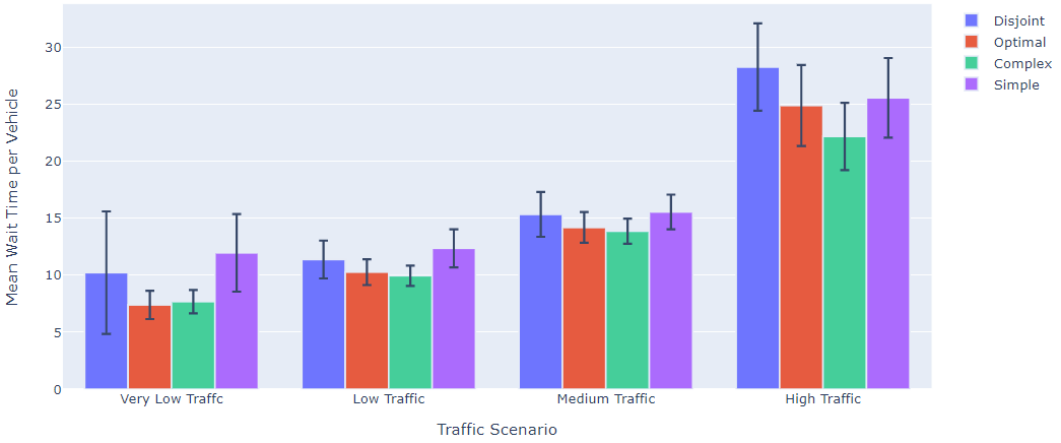

Figure 7.3: Mean Wait Time per Vehicle - Disjoint Solution and Collaboration Types

Traffic Scenario	Disjoint	Optimal	Complex	Simple
Very Low Traffic	10.22 ± 5.36	$\mathbf{7.39} \pm 1.24$	7.67 ± 1.02	11.95 ± 3.40
Low Traffic	11.96 ± 1.66	10.27 ± 1.13	$\mathbf{9.94} \pm 0.89$	12.35 ± 1.67
Medium Traffic	15.32 ± 1.97	14.18 ± 1.35	$\mathbf{13.85} \pm 1.10$	15.54 ± 1.52
High Traffic	28.24 ± 3.83	24.87 ± 3.54	$\mathbf{22.15} \pm 2.95$	25.54 ± 3.49

Table 7.3: Mean Wait Time per Vehicle - Collaboration Types

in terms of total wait time or mean wait time per vehicle, the result of the test cases were evaluated using a one-tailed t-test, $\alpha = 0.05$ and the following hypotheses:

$$H_0 : Sample_{collaborative} >= Sample_{disjoint}$$
$$H_a : Sample_{collaborative} < Sample_{disjoint}$$

In order to reject the null hypothesis in favor of the alternative hypothesis $p-value < 0.05$ is required. Statistical significance testing of the collaborative optimal results versus the disjoint results reported above yielded a $p - value < 0.001$ for all four traffic scenarios in terms of sum of wait time as well as mean vehicle wait time which clearly shows that the performance improvement by collaboration is statistically significant.

7.3 Research Question 3 - Real World Example

Can the proposed collaborative approach achieve significant improvement over highly optimized real world intervals in terms of total wait time or mean vehicle wait time?

The third research question is geared towards measuring the performance of the solution in a competitive real world setting. In order to answer this question the chosen benchmark are the intersections of Christchurch NZ which were remodeled in *SUMO*. As mentioned

previously the intersections in Christchurch NZ are managed by the Christchurch City Council Traffic Signals Team which kindly provided the traffic count data as well as phase time logs for the measured intervals. The intersections phase times are optimized by SCATS[1] which is used for monitoring, managing and optimizing traffic flow for Christchurch's intersections resulting in much more efficient pseudo-fixed timings than those featured in the experiment setup.

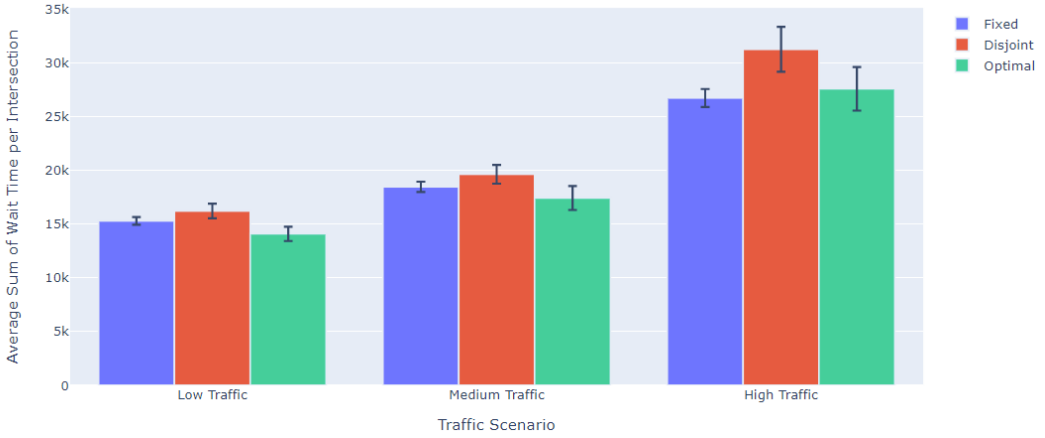

Figure 7.4: Average Sum of Wait Time - Christchurch Intersections

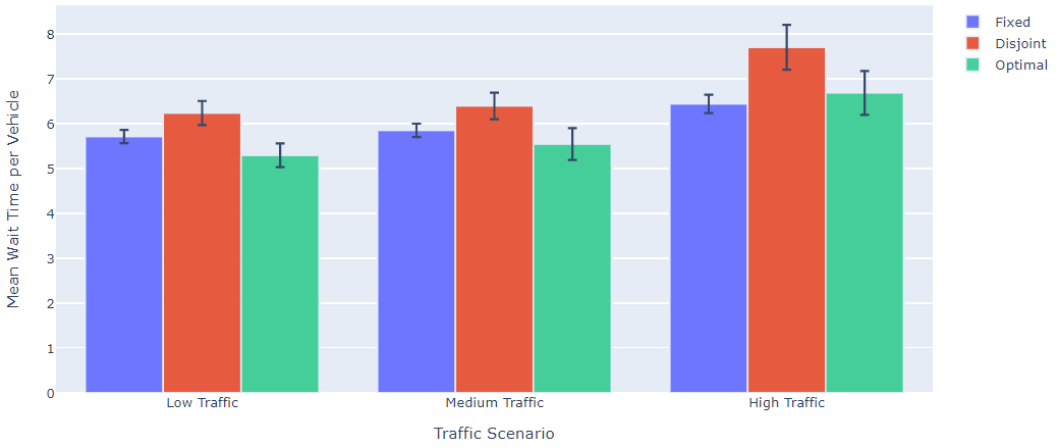

Figure 7.5: Mean Wait Time per Vehicle - Christchurch Intersections

To quickly summarize the setting again the disjoint solution and the collaborative solution was evaluated against the optimized fixed time interval for the three intersections shown in Figure 4.4. The low, medium and high traffic scenarios are based on the midday traffic, morning peak and evening peak shown in Figure 4.13 respectively. Section 5.3

[1] https://www.scats.nsw.gov.au/

Traffic Scenario	Fixed	Disjoint	Optimal
Low Traffic	15284.53 ± 351.97	16205.27 ± 687.12	**14081.47** ± 668.36
Medium Traffic	18451.73 ± 480.53	19624.27 ± 876.48	**17409.9** ± 1111.55
High Traffic	**26716.2** ± 833.56	31242.10 ± 2096.27	27566.17 ± 2028.36

Table 7.4: Average Sum of Wait Time per Intersection - Christchurch Intersections

Traffic Scenario	Fixed	Disjoint	Optimal
Low Traffic	5.71 ± 0.15	6.24 ± 0.27	$\mathbf{5.23} \pm 0.26$
Medium Traffic	5.85 ± 0.15	6.40 ± 0.30	$\mathbf{5.55} \pm 0.36$
High Traffic	$\mathbf{6.44} \pm 0.21$	7.70 ± 0.50	6.69 ± 0.49

Table 7.5: Mean Wait Time per Vehicle - Christchurch Intersection

summarized the results of this evaluation showing that the highly optimized fixed time intervals are very efficient for the available real world data as they outperformed the disjoint non-collaborative solution for all three traffic scenarios. The collaborative solution did however show very promising results for low to medium traffic and comparable results for the high traffic scenario.

In order to provide a concrete answer to the research question at hand, the results for the sum of wait time and mean wait time per vehicle shown in Figure 7.4 and Figure 7.5 and summarized in Table 7.4 and Table 7.5 are tested for statistical significance based on the same testing scheme used in Section 7.2. Thus a one-tailed t-test with $\alpha = 0.05$ and the following hypotheses is used to confirm significance:

$$H_0 : Sample_{collaborative} >= Sample_{fixed}$$
$$H_a : Sample_{collaborative} < Sample_{fixed}$$

For the sums of wait times per intersection, both the small and medium traffic scenario results yielded a $p-value < 0.001$ which shows a significant improvement in performance. For the high traffic scenario the hypotheses were reversed to check if the model is performing significantly worse than the fixed time interval and with a resulting $p-value = 0.041$ it can be said that for $\alpha = 0.05$ the resulting total wait time is indeed significantly worse.
While the results for total wait time are all significant in either a positive or negative sense the significance testing for the mean vehicle wait time yielded $p - values = [0.055, 0.13, 0.23]$ for low, medium and high traffic scenarios which shows that for $\alpha = 0.05$ the reduction in mean vehicle wait time is neither significantly better nor worse than that of optimized fixed time intervals. It should be noted that the results for the low traffic scenario almost reached the threshold for significance. The definitive answer to the research question at hand is that the collaborative Q-learning solution was shown to

bring significant improvement over the intervals of a highly optimized solution regarding the total wait time in the given intersections. However the experiment could not confirm a significant improvement for very high traffic loads and for mean vehicle wait times in general.

7.4 Summary and Discussion

7.4.1 Contributions

In summary, the answers to the three research question posed in this thesis provide three core contributions to the field of traffic management with reinforcement learning technologies. The first research question confirmed the drop in performance of state-of-the-art solutions in more complex traffic light grids and thus the need for improvement through for instance collaboration among agents to provide a competitive approach in this field. The answer to the second research question confirmed that collaboration and synchronisation among agents leads to a significant improvement on these non-collaborative state-of-the-art solutions, which warrants further research in this direction. Lastly the answer to the third research question shows that the benefit of collaboration amongst agents as proposed by this thesis results in a solution that can compete and for low to medium traffic load even outperform a highly optimized system taken from a real world example of three intersections with real traffic data. Furthermore the evaluation of the proposed solution has provided additional insights on the impact of various design decisions in a collaborative system and on the robustness of such a system to varying traffic loads and road lengths. Due to the nature of urban traffic, there are many parameters that are not constant and vary from intersection to intersection and this results in necessary adjustments that need to be made to the proposed solution. The most essential of these design decisions were discussed and evaluated by this work and as such this thesis provides knowledge on how a concrete solution should be adjusted based on the underlying variables. This includes the road length at which collaboration is no longer effective, which for $50km/h$ was shown to be at around $300m - 400m$. In a larger system or with different speed limits this threshold might vary, but the evaluation does strongly suggest that this threshold exists in every system and should be considered. In terms of collaboration and synchronisation the thesis provides insight on the strengths and weaknesses of each approach. Here, synchronous decision making and the complex collaboration type showed the most promising results for very high traffic load, while cycled synchronisation and the optimal collaboration type showed to be most stable and effective solution for low to medium traffic. In order for the collaboration types to function as intended, the state space also needs to be adjusted according to the circumstances in the given intersection. This work proposed a general guideline to define the cell discretization for any given intersection by linking the cells to the corresponding light phases as discussed in Chapter 6. While further research should be done into deriving a fixed scheme for discretizing lanes in an optimal way, the concept discussed here is a first step and provides a viable strategy that is applicable to any given intersection. Another

crucial finding is the algorithm's robustness to low traffic scenarios. The evaluation has shown that agents with the optimal collaboration type that are trained on the maximum expected traffic load using a Weibull distribution for traffic ramp up do not show any signs of overfitting and perform very well on low traffic scenarios as well, while agents that are faced with much higher traffic scenarios than those experienced during training can not properly manage traffic resulting in a massive drop in performance or even deadlocks. This highlights the importance of either estimating the maximum possible load for a system or generally overloading it during model training. While all of the design decisions summarized here can not simply be generalized and need to be made individually for a given system, it can be said that the work done during the evaluation of the algorithm provides a first overview of the strengths and weaknesses which is crucial to making an informed decision in the design process and for future work in this field.

7.4.2 Future Work and Discussion

This final section of the thesis will discuss possible future work and in which aspects the research conducted in this work can be expanded upon to provide further insight and understanding of efficient traffic management using the collaborative multi agent reinforcement learning approach proposed here. In summary, there are three separate key areas of the implementation that can benefit greatly from future work. These three areas being the conceptualisation phase of a concrete solution, the real life implementation and lastly further research into the strengths and weaknesses of the solution in larger and more complex systems or even entire city grids. While the thesis explored the effects of different design steps which included the three collaboration types, the three synchronisation schemes and the adjustment of cell discretization and subsequently the state and action space and also the width and depth of the hidden layers of the reinforcement learning model, there is no deterministic decision process for making these decisions. This means that currently the optimal combination needs to be chosen based on best practices or if the resources allow it, by simulating all possible combinations in a grid search. To this end, it would be interesting for future work to further explore the impact of these design decisions, possibly proposing a process that chooses these parameters based on certain features of the underlying system of intersections thus providing an informed recommendation for the design of these systems. The second area is the real life application of the solution which has been discussed in Chapter 6. While the conceptualisation and the training of the model can be contained within the simulation framework, the final implementation has to be migrated to the actual intersections that are supposed to be controlled by the system. Possible solutions for the challenges of this endeavor have been shown in Chapter 6 but it should be noted that ultimately the usefulness of the proposed system hinges on the feasibility of a real life implementation with all the pitfalls and difficulties that come with moving from the simulation framework to the real world. The last point of discussion for future work that is brought up in this section is further research into the impact this scalable concept of immediate neighborhood collaboration has in larger systems and how it compares to solutions that aim to fully optimize entire systems. Due to resource limitation and the defined scope of this work, the collaboration was limited to

the experiment setup featuring five intersections which was chosen to answer the research questions and provide further insight on the impact of certain design decisions on the systems behavior. For further research, it is important to expand this setup to larger grids in order to measure if the shown improvements hold up there and also look deeper into large scale effects such as the green wave effect over longer distances with multiple intersections. This is essential to confirm if limiting the collaboration to the immediate neighborhood has an adverse effect on these large scale concepts or if the solution can actually reproduce these effects. Since the process of extending this solution to large grids is iterative and can slowly be deployed throughout the entire city intersection by intersection, it is also potentially interesting to look further into how well a collaborative agent can utilize the consistency of neighboring traffic lights that operate on fixed time intervals instead of assuming a fully collaborative system. This is important as some intersections might not require further optimization or simply will not be upgraded due to the limited resources available for urban planing.

List of Figures

List of Tables

Bibliography

[AP15] Javed Alam and Manoj K Pandey. Design and analysis of a two stage traffic light system using fuzzy logic. *J. Inf. Technol. Softw. Eng*, 5(03), 2015.

[BC95] Brian Bing and Alan Carter. Scoot: The world's foremost adaptive traffic control system. *TRAFFIC TECHNOLOGY INTERNATIONAL'95*, 1995.

[CBP15] Mario Collotta, Lucia Lo Bello, and Giovanni Pau. A novel approach for dynamic traffic lights management based on wireless sensor networks and multiple fuzzy logic controllers. *Expert Systems with Applications*, 42(13):5403–5415, 2015.

[CDR09] Shilpa S. Chavan, R. S. Deshpande, and J. G. Rana. Design of intelligent traffic light controller using embedded system. In *2009 Second International Conference on Emerging Trends in Engineering Technology*, pages 1086–1091, 2009.

[Cho] Ankit Choudhary. A hands-on introduction to deep q-learning using openai gym in python. `https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/`. Accessed: 2022-05-16.

[CWCL19] Tianshu Chu, Jie Wang, Lara Codecà, and Zhaojian Li. Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1086–1095, 2019.

[DMP+21] Swati Dhingra, Rajasekhara Babu Madda, Rizwan Patan, Pengcheng Jiao, Kaveh Barri, and Amir H Alavi. Internet of things-based fog and cloud computing technology for smart traffic monitoring. *Internet of Things*, 14:100175, 2021.

[FCG16] Julia L. Fleck, Christos G. Cassandras, and Yanfeng Geng. Adaptive quasi-dynamic traffic light control. *IEEE Transactions on Control Systems Technology*, 24(3):830–842, 2016.

[GLRC18] Isaac Galatzer-Levy, Kelly Ruggles, and Zhe Chen. Data science in the research domain criteria era: Relevance of machine learning to the study of

stress pathology, recovery, and resilience. *Chronic Stress*, 2:247054701774755, 01 2018.

[GR16] Wade Genders and Saiedeh Razavi. Using a deep reinforcement learning agent for traffic signal control. *arXiv preprint arXiv:1611.01142*, 2016.

[GSL+17] Juntao Gao, Yulong Shen, Jia Liu, Minoru Ito, and Norio Shiratori. Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network. *arXiv preprint arXiv:1705.02755*, 2017.

[G17] Vladimir Gorodokin, , and Vladimir Shepelev. Procedure for calculating on-time duration of the main cycle of a set of coordinated traffic lights. *Transportation Research Procedia*, 20:231–235, 12 2017.

[Hus18] Timmy A Hussain. Combining deep q-networks and double q-learning to minimize car delay at traffic lights. 2018.

[KBM+19] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

[KKE+17] M. Kabrane, S. Krit, L. Elmaimouni, K. Bendaoud, H. Oudani, M. Elasikri, K. Karimi, and H. El Bousty. Smart cities: Energy consumption in wireless sensor networks for road trafile modeling using simulator sumo. In *2017 International Conference on Engineering MIS (ICEMIS)*, pages 1–7, 2017.

[KMGK21] Neetesh Kumar, Sarthak Mittal, Vaibhav Garg, and Neeraj Kumar. Deep reinforcement learning-based traffic light scheduling framework for sdn-enabled smart transportation system. *IEEE Transactions on Intelligent Transportation Systems*, 23(3):2411–2421, 2021.

[LDWH19] Xiaoyuan Liang, Xunsheng Du, Guiling Wang, and Zhu Han. A deep reinforcement learning network for traffic light cycle control. *IEEE Transactions on Vehicular Technology*, 68(2):1243–1253, 2019.

[LKH14] Blerina Lika, Kostas Kolomvatsos, and Stathes Hadjiefthymiades. Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4, Part 2):2065–2073, 2014.

[LLW16] Li Li, Yisheng Lv, and Fei-Yue Wang. Traffic signal timing via deep reinforcement learning. *IEEE/CAA Journal of Automatica Sinica*, 3(3):247–254, 2016.

[LZXJ19] Liangliang Lou, Jinyi Zhang, Yong Xiong, and Yanliang Jin. An improved roadside parking space occupancy detection method based on magnetic sensors and wireless signal strength. *Sensors*, 19(10), 2019.

[MAG20] Vishal Mandal and Yaw Adu-Gyamfi. Object detection and tracking algorithms for vehicle counting: A comparative analysis. *Journal of big data analytics in transportation*, 2(3):251–261, 2020.

[MKAS19] Fehda Malik, Hasan Ali Khattak, and Munam Ali Shah. Evaluation of the impact of traffic congestion based on sumo. In *2019 25th International Conference on Automation and Computing (ICAC)*, pages 1–5, 2019.

[MKS+13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[QGÖ20] Syed Shah Sultan Mohiuddin Qadri, Mahmut Ali Gökçe, and Erdinç Öner. State-of-art review of traffic signal control methods: challenges and opportunities. *European Transport Research Review*, 12(1):55, Oct 2020.

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[SCPA11] E Sifuentes, O Casas, and R Pallas-Areny. Wireless magnetic sensor node for vehicle detection with optical wake-up. *IEEE Sensors journal*, 11(8):1669–1676, 2011.

[SD80] Arthur G Sims and Kenneth W Dobinson. The sydney coordinated adaptive traffic (scat) system philosophy and benefits. *IEEE Transactions on vehicular technology*, 29(2):130–137, 1980.

[Sta09] D. Stathakis. How many hidden layers and nodes? *International Journal of Remote Sensing*, 30(8):2133–2147, 2009.

[VCVB19] Andrea Vidali, Luca Crociani, Giuseppe Vizzari, and Stefania Bandini. A deep reinforcement learning approach to adaptive traffic lights management. In *WOA*, pages 42–50, 2019.

[VPA+20] Aso Validi, Nicole Polasek, Leonie Alabi, Michael Leitner, and Cristina Olaverri-Monreal. Environmental impact of bundling transport deliveries using sumo : Analysis of a cooperative approach in austria. In *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–5, 2020.

[WZSZ09] Yiyan Wang, Yuexian Zou, Hang Shi, and He Zhao. Video image vehicle detection system for signaled traffic intersection. In *2009 Ninth International Conference on Hybrid Intelligent Systems*, volume 1, pages 222–227. IEEE, 2009.

[YHQL15] Shanhe Yi, Zijiang Hao, Zhengrui Qin, and Qun Li. Fog computing: Platform and applications. In *2015 Third IEEE workshop on hot topics in web systems and technologies (HotWeb)*, pages 73–78. IEEE, 2015.

[YSK+19] DA Yudin, A Skrynnik, A Krishtopik, I Belkin, and AI Panov. Object detection with deep neural networks for reinforcement learning in the task of autonomous vehicles path planning at the intersection. *Optical Memory and Neural Networks*, 28(4):283–295, 2019.

[ZCL+20] Pengyuan Zhou, Xianfu Chen, Zhi Liu, Tristan Braud, Pan Hui, and Jussi Kangasharju. Drle: Decentralized reinforcement learning at the edge for traffic light control in the iov. *IEEE Transactions on Intelligent Transportation Systems*, 22(4):2262–2273, 2020.

[ZYC09] Fuqiang Zou, Bo Yang, and Yitao Cao. Traffic light control for a single intersection based on wireless sensor network. In *2009 9th International Conference on Electronic Measurement Instruments*, pages 1–1040–1–1044, 2009.