Decentralised congestion control mechanisms for autonomous vehicles using distributed constraints optimisation

Huan Vu

Université Claude Bernard Lyon 1

THÈSE de DOCTORAT DE L'UNIVERSITÉ DE LYON

Opérée au sein de :

l'Université Claude Bernard Lyon 1

Ecole Doctorale 512
Informatique et Mathématiques

Spécialité de doctorat : Informatique

Soutenue publiquement le 14/04/2020, par :

Huan Vu

Decentralised congestion control mechanisms for autonomous vehicles using distributed constraints optimisation

Devant le jury composé de :

Bo An	**Rapporteur**
Professeur associé, Université de Technologie de Nanyang, Singapour	
Nicolas Maudet	**Rapporteur**
Professeur des Universités, Sorbonne Université	
Elise Bonzon	**Examinatrice**
Maître de Conférences, Université Paris Descartes	
Hamamache Kheddouci	**Examinateur**
Professeur des Universités, Université Lyon 1	
Leila Merghem-Boulahia	**Examinatrice**
Professeure des Universités, Université de Technologie de Troyes	
Samir Aknine	**Directeur de thèse**
Professeur des Universités, Université Lyon 1	

UNIVERSITÉ CLAUDE BERNARD LYON 1

Decentralised congestion control mechanisms for autonomous vehicles using distributed constraints optimisation

by

Huan Vu

A thesis submitted in partial fulfillment for the degree of
Doctor of Philosophy

in the

LIRIS - Laboratoire d'Informatique en Image et Système d'information

April 2020

UNIVERSITÉ CLAUDE BERNARD LYON 1

LIRIS - Laboratoire d'Informatique en Image et Système d'information

Abstract

Decentralised congestion control mechanisms for autonomous vehicles using distributed constraints optimisation

by Huan Vu

Autonomous vehicles are predicted to number several millions by 2025. Crucially, these vehicles will be able to communicate and coordinate with vehicles in range, opening up opportunities to mitigate congestion and the risk of accidents. This ability to communicate and coordinate underpins the notion of Connected Autonomous Vehicles (CAVs). This thesis presents a decentralised mechanism for traffic control of CAVs in settings where road intersections have to be managed and optimised. Against this background, we propose a solution based on the distributed constraint optimisation approach (DCOP). We first model the intersection and formulate the regulation problem as a DCOP. Following this, we evaluate the performance of different DCOP algorithms. Thereafter, we opt for an algorithm and adapt it to the traffic regulation problem, in order to improve performance and enhance security. In a multi-intersection setup, we propose an individual priority mechanism allowing road intersections to distribute vehicles while avoiding computational expensive global optimisation.

UNIVERSITÉ CLAUDE BERNARD LYON 1

LIRIS - Laboratoire d'Informatique en Image et Système d'information

Résumé

Mécanismes de régulation décentralisés pour les véhicules autonomes fondés sur l'optimisation sous contraintes distribuée

par Huan Vu

On prévoit que le nombre de véhicules autonomes atteindra plusieurs millions d'ici 2025. Ces véhicules pourront communiquer et se coordonner avec les véhicules à leur portée, ce qui permettra de réduire les encombrements et les risques d'accidents. Cette capacité de communication et de coordination sous-tend la notion de véhicules autonomes connectés (CAVs). Cette thèse présente un mécanisme décentralisé de régulation de traffic pour les CAVs dans des environnements où les intersections doivent être gérées et optimisées. Nous proposons une solution basée sur l'approche d'optimisation sous contraintes distribuée (DCOP). Nous modélisons d'abord l'intersection et formulons le problème de régulation comme un DCOP. Ensuite, nous évaluons les performances de différents algorithmes DCOP. Ensuite, nous optons pour un algorithme et l'adaptons au problème de la régulation du trafic afin d'améliorer les performances et la sécurité. Dans une configuration multi-intersections, nous proposons un mécanisme de priorité individuel permettant aux intersections de répartir les véhicules tout en évitant une optimisation globale coûteuse en calcul.

Acknowledgements

I would like to express my very great appreciation to my supervisor, Professor Samir Aknine, who single-handedly guided me through my four years of discovery. He has helped me a lot with his patience, support and encouragement. I am also thankful to him for his effort to organise my thesis defence during the Covid-19 pandemic.

I would also like to offer my special thanks to my co-authors, with whom I have spent wonderful time working. I owe my gratitude to Professor Sarvapali Ramchurn for his suggestions and guidance, and for hosting my two visits to his team in Southampton. I would like to thank Dr. Matthis Gaciarz for his valuable help during my first year, and Professor Alessandro Farinelli for his discussion and comments on DCOPs.

My special thanks goes to the reviewers, Bo An and Nicolas Maudet, and the examiners, Elise Bonzon, Hamamache Kheddouci and Leila Merghem-Boulahia, for their constructive comments and advice, before and during the defence.

I am particularly grateful for the support and good times given by my Vietnamese friends, in Lyon and everywhere else. Their friendship has surely made my years less stressful.

I appreciate many colleagues and friends, both in Lyon and in Southampton for every interesting discussion that we have had, inside and outside the work.

To my beloved family for all the support through my study. My special thanks goes to my life partner, Thu, for moving to France and for raising our children in France with me. To my two daughters, who have given me a lot of encouragement. I thank my parents and my sister for all their belief, their moral support and their unconditional love.

To my family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the growth in urbanisation and ownership of cars, most major cities around the world suffer from high rates of traffic congestion, with a significant impact on the economy and human well-being. In the US alone, urban congestion costs 8.8 billion hours of travel delay, and 3.3 billion gallons of wasted fuel per year (Schrank et al., 2019). In addition, pollution due to petrol and diesel cars at stand still at major traffic intersections can rise to more than 20 times than in normal free flow traffic conditions (Goel and Kumar, 2015).

Several attempts have been made to address this problem for the last decades. Solutions vary from the new designs of lanes, intersections and roundabouts to the optimisation of traffic light plans and traffic flow. With the arrival of new technologies that are developed in recent years, vehicles are now able to use intelligent devices on board (e.g. sensors, communication devices, auto pilot systems). These devices enable the possibility to conduct new approaches to coordinate vehicles in urban traffic.

In urban traffic, intersections are often in the core of congestion since it is the place where roads meet. Hence, reducing delays of vehicles in front of intersections is considered extremely crucial when optimising traffic conditions. In the first approaches, researchers take advantage of sensors installed on road networks to optimise traffic lights plans. The optimisation can either be offline (Robertson,

1969) (i.e. the plan is incapable of reacting to real-time information) or online (Hunt et al., 1982; Sims and Dobinson, 1980; Henry et al., 1984) (i.e. the light plan is optimised using the current traffic data). However, these approaches are still based on the traffic light technology, which only deals with vehicles as a flow instead of considering each vehicle individually.

Furthermore, autonomous cars are predicted to number several million by 2025. Crucially, these cars will be able to communicate and coordinate with vehicles in range, opening up opportunities to mitigate congestion and the risk of accidents. This ability to communicate and coordinate underpins the notion of Connected Autonomous Vehicles (CAVs). During the 1990s and 2000s, artificial intelligence enabled the investigation of new methods for traffic modelling and regulation, especially with multi-agent technologies that are able to solve various problems in a decentralised and/or distributed way (Bazzan and Klügl, 2014; Dresner and Stone, 2008, 2007). Today's communication technology has enabled the design of regulation methods based on real-time communication of accurate information. Each vehicle on a network has a traffic context, and the information provided can be useful for efficient regulation: the accumulated delay since the start of the vehicle's journey, its current position, its short and long-term intentions, etc.

The drawback of recent methods is that they focus only on the intersection level and often are not compatible with classic traffic flow optimisation solutions. Therefore, even though these solutions are proven to be effective at the intersection level, without the knowledge about the situation from neighbouring intersections, vehicles can be sent to a congested, or even deadlock situation. We discuss this in more detail in the Chapter 5.

On account of this, this thesis focus on designing a mechanism that optimises traffic at each intersection in a microscopic manner, while being able to improvise different situations in the neighbour intersections. In the following section, we will list all the requirements that such mechanism needs to fulfil.

1.1 Research Challenges

Designing a traffic regulation model for CAVs can be quite challenging. We can highlight some of the main challenges as follows.

- **Safety:** Before taking into account the efficiency of the regulation method, the model needs to be safe. Safety is, indeed, the key of any transportation system currently and in the future. This is the reason why traffic lights are adopted all over the world because if vehicles follow the rules, there would be no conflict between them. In the Intelligent Transportation System (ITS) where vehicles follow their individual crossing plan, safety would become even more crucial. Therefore, at any point in time, the system must be conflict free if vehicles follow their instructions. Thus, an agreement should be made in advance and the system must ensure that vehicles share the same agreement before they cross the intersection.

- **Efficiency:** The efficiency of the system must be considered. In a near future, CAVs are probably preferred over human drivers thanks to the quickness and precision of their reaction. Indeed, several researches have shown that even a basic lightless intersection can outperform classical traffic light system. Regarding the overall measurement of efficiency, many criteria can be taken into account, the first and most obvious being the traffic delays. Other criteria might also be considered, such as the energy efficiency and the comfort of passengers.

- **Robustness:** Since the traffic is a highly dynamic system, the regulation model must be robust to sudden changes. The examples of these changes can be: the arrival of emergency vehicles, buses, accidents and other infrastructure failures. Such events need to be dealt with in a quick and efficient way.

- **Compatibility:** The number of CAVs can only gradually increase over time. Thus, part of the traffic will still be conducted by human beings. In transition period, the system needs to be compatible with human drivers.

One possible solution is to give CAVs a dedicated lane. However, this would require extending existing roads, which is costly in urban area. In this regard, we believe that the best solution that is to build a mechanism that can take into account human drivers and CAVs at the same time.

In addition to these, there might be some minor problems such as communication range of vehicles and computational requirements from the infrastructure. These problems need to be addressed later when deploying such mechanism. Therefore in the scope of this work, we only give discussion in regard of these challenges.

Thereafter, we discuss the contributions of this thesis to the state-of-the-art.

1.2 Research contributions

Against these challenges, the contributions of this thesis are as follows.

- We model the traffic regulation problem at an intersection. In our model, we take into account a large amount of information, from vehicle speed, position and destination to traffic conditions. We then identify the rules to build a plan for all vehicles to cross the intersection.
- We formulate our model as a Distributed Constraint Optimisation Problem where vehicles can continuously exchange messages to find the optimal plan. This is a novel approach to microscopic transportation model. We then evaluate different DCOP algorithms in terms of solution quality and complexity to choose an algorithm that suits our case.
- We propose a safety enhancement to the algorithm. Note that DCOP algorithms may take a large amount of time to finish. Therefore, in case of failure, the system must be able to provide a backup plan that allows vehicles to cross the intersection instead of stopping them waiting for the algorithm.
- We propose some improvements to the chosen algorithms (the Max-Sum algorithm (Farinelli et al., 2008) and the Max-Sum_AD_VP algorithm (Zivan

and Peled, 2012)) and adapt them to our traffic problem, enhancing the efficiency and the computational speed.

- We develop our model to connect several intersections. In so doing, we create an individual priority level for vehicles that allows the system to deal with different traffic flows coming from several directions. Since optimising delays at a single intersection can sometimes result in sending the vehicles to a more congested area, this tool helps the system or the city authorities to regulate traffic in a macroscopic level.

- We discuss the choice of different methods proposed in regard to the equipment level at different intersections throughout the network, and also at the CAVs.

When taken together, this work proposes a novel congestion management model based on a DCOP representation, both at the intersection and the network level. The solution combines the efficiency of microscopic traffic regulation at the intersection level where each vehicle is considered individually, and the possibility of balancing traffic between the intersections. The proposed solution makes it possible to reduce congestion across the network, while avoiding computationally expensive global optimisation. At the intersection level, we propose two models with different precision levels. We then evaluate several DCOP algorithms, and opt for two of these algorithms, the Max-Sum algorithm (Farinelli et al., 2008) and the Max-Sum_AD_VP algorithm (Zivan and Peled, 2012). For each of the algorithms used, we propose an improvement to account for the particular structure of our problem. Furthermore, we empirically evaluate all our propositions, at the intersection level as well as at different multi-intersection scenarios, to show their effectiveness and drawbacks.

Parts of our work during the PhD has led to the publication of the following paper:

Huan Vu, Samir Aknine and Sarvapali Ramchurn (2018). A Decentralised Approach to Intersection Traffic Management. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI 2018)* (Vu et al., 2018). This paper presents our first cellular intersection model and a novel application of

DCOP algorithms to traffic regulation problem. The paper also illustrates different aspects such as safety and continuity of the solution. The work proposed in this paper is presented in Chapter 4.

Huan Vu, Samir Aknine, Sarvapali Ramchurn and Alessandro Farinelli (2020). Decentralised Multi-Intersection Congestion Control for Connected Autonomous Vehicles. In *Proceedings of the 17th European Conference on Multi-Agent Systems (EUMAS 2020)* (Vu et al., 2020). This paper presents some extensions to the cellular model presented in the previous one on several aspects, including a more space-efficient model, the use of a recent variant of the algorithm, and the multi-intersection solution. This work is detailed in Chapter 5.

During the PhD program, the knowledge achieved on intelligent transportation system and/or distributed constraints optimisation has also led to the publication of:

Matthis Gaciarz, Samir Aknine and Huan Vu (2017). A constraint-based coordination model to advantage buses in urban traffic. In *Proceedings of the 29th IEEE International Conference on Tools with Artificial Intelligence* (Gaciarz et al., 2017). This is a paper that introduced a coordination/negotiation policy for intersections to allow a bus to respect its timetable.

Sacha Lhopital, Samir Aknine, Vincent Thavonekham, Huan Vu and Sarvapali Ramchurn (2020). Decentralised Control of Intelligent Devices: A Healthcare Facility Study. In *Proceedings of the 17th European Conference on Multi-Agent Systems (EUMAS 2020)* (Lhopital et al., 2020). This paper presents a DCOP solution to deliver notifications for healthcare assistants to reduce their workload.

1.3 Thesis Outline

The remainder of this thesis is organised as follows:

In Chapter 2, we explore the existing solutions to traffic management. The chapter first gives the outline of classical methods used widely around the world, then

discusses different optimisation proposed to these methods. After that, it discusses the future of traffic using connected vehicles (CVs) and connected autonomous vehicles (CAVs), then surveys several existing solutions in both the transportation research area and the artificial intelligence one.

Chapter 3 gives a background on the method that we use throughout this thesis to address the challenges. We first give a definition of a DCOP framework and why it is a popular field of research in multi-agent systems. We then discuss different DCOP algorithms and their properties. Finally, we show some notable applications using the framework.

In Chapter 4, we first detail our proposed intersection model using a cellular representation. We then propose different ways of formulation as a DCOP to the model and evaluate some DCOP algorithms. Via this evaluation, we choose the suitable algorithm, namely the Max-Sum algorithm, and identify the remaining issues. We then propose some improvement to the Max-Sum algorithm to overcome these issues. Finally, we discuss safety and continuity of our approach and evaluate the performance of our system.

Chapter 5 proposes different extensions to the previous model. First, we outline the limits of a cellular model and propose a novel space-efficient model. After that, we propose the use of an individual priority to take into account traffic information at a higher level (e.g. density, congestion). We develop this tool and propose possible priority distributions regarding traffic conditions. In this chapter, we also study the use of a recent variant of the previously used Max-Sum algorithm and based on our priority level, we propose a better way to order nodes to speed up the computation. We conclude the chapter by evaluating these propositions in different traffic conditions.

Chapter 6 concludes the work in this thesis, and outlines possible improvements in the future.

Chapter 2

Background on Intersection Management Methods

Many efforts have been made throughout history to deal with traffic management issues. In this chapter, we will highlight several existing methods that were used or proposed to coordinate vehicles in the urban area, in order to keep a safe and efficient traffic flow. [1]

We first begin by identifying the compositions of an intersection in urban traffic in section 2.1. We then discuss in section 2.2 the classic regulation methods that are widely used in cities such as priority rules, priority signs and traffic lights. In this section, we also highlight some of the existing work on optimisation of a traffic light plan. After that, in section 2.3, we will discuss agent-based approaches for traffic regulation, which are designed for connected vehicles (CVs) or connected autonomous vehicles (CAVs). We put forward single intersection approaches, then coordination approaches between intersections.

[1] Part of this background research was done in collaboration with M. Garciarz in our internal report. (Gaciarz et al., unpublished)

2.1 Urban Traffic Composition

First, we define several compositions of an urban traffic intersection. An intersection is a place where two or more roads intersect. Each intersection may have one or several incoming and outgoing lanes in each direction. The zones of an intersection can be divided as follows (cf. Figure 2.1):

- The conflict zone: the area where several roads superimpose and where conflicts between vehicles are most likely to occur.
- The incoming lanes: upstream of the conflict zone, contains the vehicles likely to enter the conflict zone.
- The outgoing lanes: downstream of the conflict zone, through which the vehicles are evacuated.

2.2 Classic Regulation Methods

When crossing an intersection, the classical way to coordinate human drivers is through traffic laws. Different countries can use different laws. Here are some that are widely used around the world.

2.2.1 Priority Rules for Uncontrolled Intersection

When approaching an uncontrolled intersection (i.e. an intersection that has no other controlling method), vehicles must follow the priority rule. This rule is often called "Priority to the right" for countries where traffic keeps to the right. This system requires all vehicles to give way to the ones approaching from their right at intersections. This is a common law that is applied in most countries and is stipulated in the Vienna Convention on Road Traffic (United Nations Economic Commission for Europe, 1968).

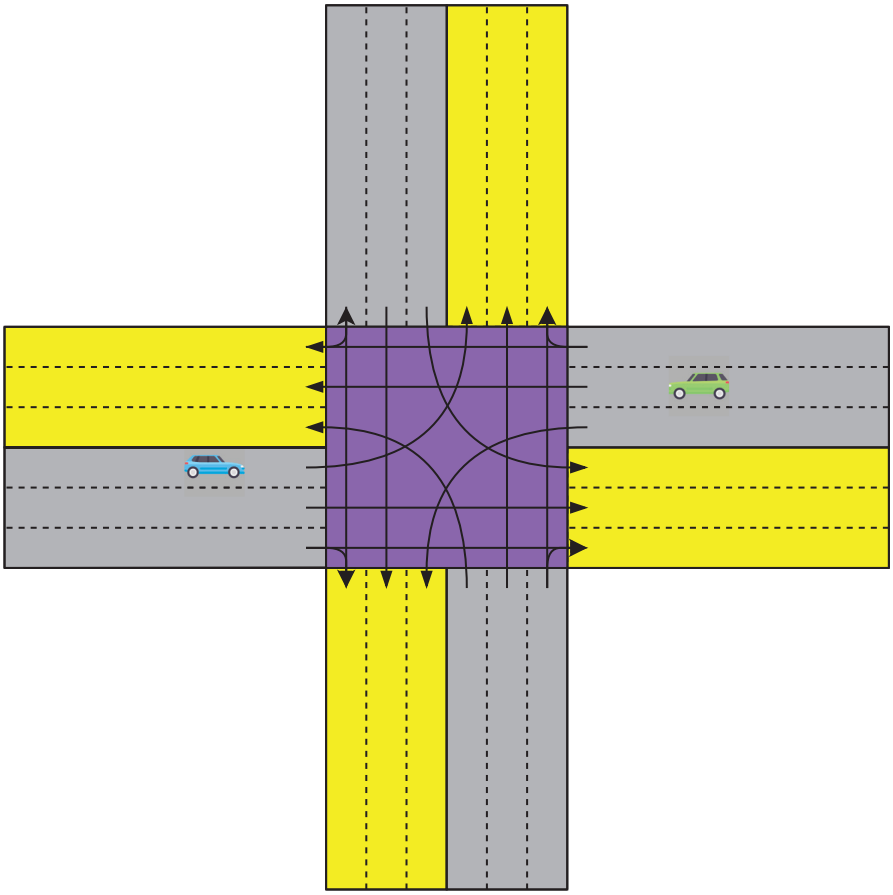

FIGURE 2.1: An intersection between two roads with 6 lanes each. Driving is on the right. The conflict zone is coloured in purple, the incoming lanes in gray and the outgoing lanes in yellow. The arrows represent the different possible vehicle streams. Thus the green vehicle or a vehicle on the same track can only go straight using the East-West current. The blue vehicle or a vehicle on the same track can only turn left using the West-North current.

Intersections are sometimes constructed by a different manner using roundabouts. Priority, in this case, should be given to traffic that is already inside the junction (e.g. traffic coming from the left in countries that drive on the right side). Modern roundabouts can have different designs that may enhance safety and capacity of roads in a different manner. (Kennedy, 2007) reviews the effect of different designs in various countries.

2.2.2 Priority Signs

Traffic laws used for uncontrolled intersections can be overridden by priority signs. Common examples are the Give Way sign (cf. Figure 2.2a) and the Stop sign (cf. Figure 2.2b). These signs are often used to indicate that vehicles must give way to other directions, regardless of their destination. The Stop sign also requires vehicles to stop before continuing their cross.

(A) (B)

FIGURE 2.2: Traffic sings: (A) Give Way sign (B) Stop sign, as in the Vienna Convention on Road Signs and Signals. (United Nations Economic Commission for Europe, 2006)

2.2.3 Traffic Lights and Optimising Traffic Light Plans with Classic Methods

One of the most commonly used regulation methods is traffic lights. Traffic lights allow road users, pedestrians or vehicles to be assigned the right to enter the conflict zone of an intersection or on a pedestrian crossing. For vehicles, this is represented by a colour code:

- Green allows vehicles to enter the conflict zone.
- Yellow is a transitional state from green to red, and vehicles should stop as far as possible.

- Red prohibits vehicles from entering the conflict zone.

The configuration of the intersection decides which group of vehicles is affected by the signal. A signal may relate to one or several channels of the same section. Sometimes the signal relates to a particular flow of the intersection, that is to say vehicles traveling from some lanes in the storage area to several lanes in the exit zone.

The conventional traffic light systems function as follows: time is divided into a series of cycles, which are themselves a series of phases. A phase is the period during which one or more flows are admitted into the intersection. These flows must be coherent, i.e. they must not overlap (they are said to be compatible), or be sufficiently used to reduce the risk of conflicts. At the end of a phase the signal changes to yellow, then to red. After a few seconds of full red during which no phase is active, in order to allow vehicles to evacuate out of the conflict zone, another phase begins. A cycle is therefore a succession of phases during which all the currents admitted into the intersection have been served at least once.

This type of conventional systems is implemented by TRANSYT (TRAffic Network Study Tool) (Robertson, 1969), one of the oldest systems proposed for traffic light management. This requires many parameters on the network on which it is deployed: network geometry, number of vehicles measured or expected, etc. This system optimises the traffic light plan in order to produce an optimal configuration. However, this optimisation stays an off-line solution, and is therefore incapable of adapting to the variations of the traffic in real time.

In contrary, some systems, called "adaptive", measure in real time certain traffic data and exert a strategy that adapts into the actual conditions of traffic.

Moreover, some transportation systems, whether they concern the issue of regulation or not, are called "cooperative". A cooperative transportation system is defined by the European Commission as a system in which "Road operators, infrastructure, vehicles, their drivers and other road users will co-operate to deliver the most efficient, safe, secure and comfortable journeys" (Bly, 2004). A co-operative system therefore requires communication and reaction capacities from its actors,

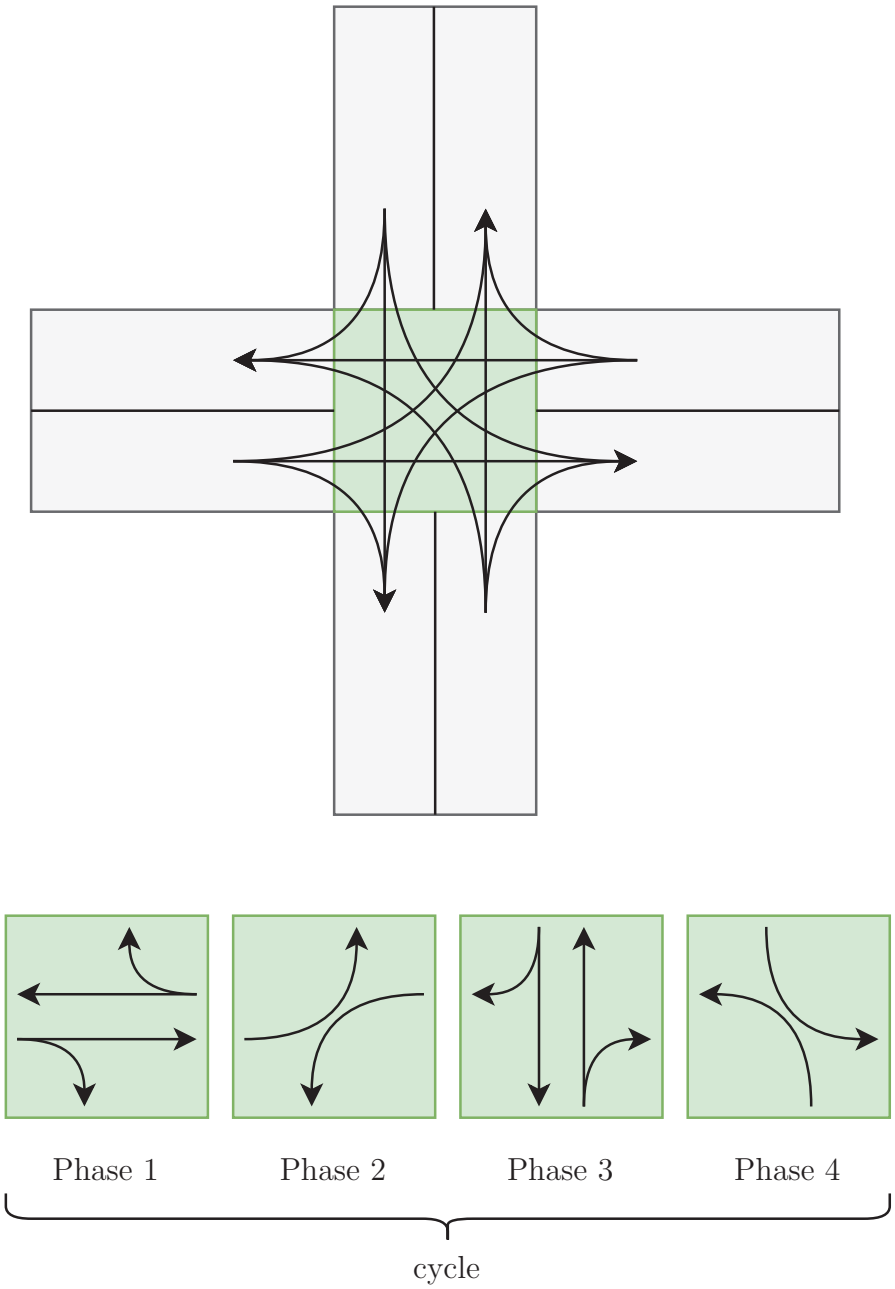

FIGURE 2.3: An intersection between two roads with 2 lanes each. 12 currents exist to cross this intersection. These currents can be decomposed into 4 groups of compatible currents (other cuttings are possible). The periods during which green and yellow are assigned to each of these groups of currents are the different phases forming the cycle.

in particular the vehicles. This can be obtained by the increasing equipment level of vehicles and drivers. A "connected vehicle" is a vehicle equipped for communication. This type includes autonomous vehicles, i.e. vehicles without a human driver, whose presence on the roads could become common in the near future. A cooperative system may have different objectives, such as travel safety or traffic quality.

2.2.3.1 Adaptive Cyclic Systems

Several adaptive systems remain close to the conventional traffic light systems with cycles. However, the traffic light plans are modified dynamically in order to adapt to the traffic conditions.

The most popular system of this kind is SCOOT (Split Cycle and Offset Optimisation Technique) (Hunt et al., 1982), developed in the United Kingdom in the 1980s. Sensors (electromagnetic loops) located on each section make it possible to measure the flow of vehicles, to detect the rise of queues due to congestion, and the presence of vehicles which may be stopped for another reason. All this information is used and processed by a traffic flow model to refine the phases of the lights for each intersection of the area supervised by the system in order to minimise the weighted sum of the number of stops by the vehicles and the length of queues in front of the intersections. This optimisation strategy is implemented by slight variations in the duration of the phases (up to 4 seconds) and / or the duration of the cycles (up to 8 seconds) of the lights in the area concerned.

SCATS (Sydney Coordinated Adaptive Traffic System) (Sims and Dobinson, 1980) was developed in the 1980s. Like SCOOT, this system retrieves information using electromagnetic loops. In fact, SCOOT relies on a traffic flow model to anticipate progression while SCATS operates on the data directly perceived. This strategy is based on several libraries: for cycle times, offsets and green light times. An algorithm helps select the most appropriate elements in different libraries to dynamically build a light plan. Different optimisation criteria are used depending on the context (night, fluidity, peak period, congestion). Implementation of this

strategy takes the same forms as the SCOOT strategy, namely the modification of cycle times, phases and offsets. SCATS performs an optimisation on a subsystem containing from an isolated intersection to a set of 10 intersections.

TUC (Traffic-Responsive Urban Control) (Diakaki, 1999) is a system developed in the 1990s in response to SCOOT and SCATS' lack of efficiency towards rapid traffic conditions changes, particularly congestion, and the algorithmic complexity of the acyclic systems described hereafter (OPAC, PRODYN, RHODES ...) which is an obstacle to their deployment on the scale of an entire urban network. The main strategy is based on an automatic control theory (Linear-Quadratic-Regulator) to dynamically modify the relative duration of the phases in a cycle. In this system, the collection of dynamic data can be based on electromagnetic loops as for SCOOT or SCATS, or on a video detection system.

2.2.3.2 Adaptive Acyclic Systems

Unlike the adaptive systems presented above, certain systems depart from the cycles of the traffic lights, eliminating this notion. In most cases, this is replaced by the choice of the switching time, that is to say the switch to the next phase.

One of the oldest systems of this type is PRODYN (DYNamic Programming) (Henry et al., 1984) developed in the 1980s. In this system, 2 to 3 electromagnetic loops are arranged on each section, and the assumed state of traffic on each path is estimated using a simple flow model to anticipate the progress of vehicles on the track. This system performs an optimisation on the "isolated" intersection. However some versions of this system allow communications between neighbouring intersections in order to anticipate incoming flows. The strategy used by this system consists in analysing at each time step (of 5 seconds) whether to switch the state of the light (i.e. to change phase) is the optimal decision (i.e. if it minimises the waiting time of vehicles in front of the intersection for the next 75 seconds according to the flow model used).

OPAC-RT (Real-Time Optimisation Policies for Adaptive Control) (Gartner et al., 2001), also developed in the 1980s, shares a number of similarities with PRODYN:

it is based on switching, and its strategy is also based on the minimisation of the waiting time of the vehicles by a flow model. There are several differences between OPAC-RT and PRODYN: the position of the sensors (only one at the beginning of the section), the flow model used, as well as the duration of the time horizon considered.

UTOPIA (Urban Traffic Optimisation by Integrated Automation) (Mauro and Di Taranto, 1990) was developed in the 1980s. One of its main objectives being the setting up of priorities for public transport. Sensors are located on each section near the lines of fire. This system operates on two levels. At the regional level, a macroscopic flow model is used to predict the vehicle progression and optimisation is carried out in order to minimise the travel time of vehicles on the network. Reference light plans as well as other parameters resulting from this optimisation are transmitted to the local level (i.e. to the intersections). Based on these recommendations, each intersection locally carries out an optimisation of the weighted sum of the waiting time and the number of stops by the vehicles, and the length of the queues. The two levels of regulation consider different time scales: the regional level is based on a time horizon of 30 minutes and the local level is based on a time horizon of 120 seconds.

2.2.3.3 Other Adaptive Systems

The systems presented in the previous two sections are based on baseline approaches and have all been effectively implemented in one or more cities. Several other systems with different approaches exist but have not been implemented, except for on-site experiments for some of them.

CRONOS (Control of Networks by Optimisation of Switchovers) (Boillot et al., 2006) is a different system that relies on a heuristic approach. It allows an optimisation that can be decentralised, where each intersection is treated locally and individually, or more centralised, where an overall optimisation is carried out over an area of up to 6-8 intersections. One of its peculiarities is to recover traffic data, not by electromagnetic loops but by cameras. Finally, CRONOS does not

have predetermined phases but relies on security constraints making it possible to prohibit states of light that may create conflicts.

RHODES (Real-time Hierarchical Optimised Distributed Effective System) (Mirchandani and Head, 2001) operates on three different levels: the first level takes into account the slowly shifting traffic characteristics (change of the network, evolution of the most frequented routes, etc.), the second level is based on the first one to estimate the number of vehicles per hour on each section and to deduce an approximate green time, which allows the last level to produce a control adapted to the demand at the scale of an intersection.

MOVA (Microprocessor Optimised Vehicle Actuation) (Vincent and Peirce, 1988) relies on another type of information, inter-vehicular distances, to determine the saturation state of traffic. Thus, the criteria taken into account in the regulation strategy vary according to this state.

The MOTION (On-line Controlled Networks) (Bielefeldt and Busch, 1994) strategy works as follows: each intersection determines a minimum cycle time as a function based on the estimated vehicle flows. Then a common cycle time at all intersections is determined, and coordination takes place as a phase offset to create green waves.

The CARS (Control Audoadaptativo para Redes Semaforizades) strategy (Barceló, 1991) is close to SCOOT, except that it relies on a sliding time horizon whose duration is permanently adjusted to correspond to the duration of the cycle.

2.2.3.4 Discussion on Traffic Light Optimisation Approaches

The systems presented above, whether TRANSYT or the adaptive systems, are implemented in many cities around the world. This can, of course, be explained historically. However these systems possess a number of interesting properties:

- Limited perception of the environment: these systems rely on data collected by simple sensors, usually electromagnetic loops, present between one and

three times per section. If the information used by these systems is limited, it does not require complex and costly equipment.

- Computational simplicity: the calculations made by these systems are most often carried out individually by intersections and are relatively simple, which facilitates their application in real time.

- Decentralisation and communication simplicity: some systems such as UTOPIA are structured at several levels and use regional centralisation, while others like PRODYN allows communication between neighbouring intersections. Although these communications are costly, they are relatively rare, making these systems less sensitive to a number of failures and do not require complex communication infrastructure.

These properties make it possible to discard a certain number of hypotheses, notably on the level of equipment of the environment (infrastructure and vehicles). However, recent technologies, especially those embedded in vehicles, make these assumptions less and less costly. Moreover, these adaptive systems make other hypotheses, which can be more or less simplifying.

These systems, in particular the cyclic systems, limit the actions to a certain number of possibilities, most often resulting from the traditional regulation: modification of phase times, cycles and phase shifts between neighbouring intersections.

However, other regulatory actions can be envisaged to allow more dynamic regulation. Moreover, the connected vehicles make it possible to envisage new possibilities of action which can be used for regulation, for example to communicate on the control strategy of an intersection so that the vehicle adapts its acceleration profile.

Some of these systems rely on flow models to predict changes in traffic. While it seems difficult to avoid any predictive model in an anticipatory approach, the simplicity of the perceptions of these systems does not allow a dynamic adjustment of the predictions realised. This criticism can be reinforced by the fact that the connected vehicles, because of their communication and coordination capacities,

exhibit different dynamic flows than classic vehicles. The diffusion of this type of vehicle could make the predictions realised by this type of system less and less relevant, and thus decrease the effectiveness of the regulation realised.

2.3 Using Multi-Agent Systems for Traffic Regulation

Various issues related to the regulation of urban traffic have been identified by the multi-agent community. We describe here several works, in their diversity, the way that they deal with regulation or coordination between vehicles, on the scale of an intersection or a wider area. In the first case, coordination and regulation at an intersection can be considered as a problem independent of the question of global regulation on a network, and the term "isolated" intersection means that the rest of the network is not taken in consideration. In the latter case, coordination takes place between intersections in order to achieve more coherent network-wide regulation.

2.3.1 Regulation on an Isolated Intersection

On an isolated intersection, various coordination problems and various approaches can be considered. Most of them concern real-time traffic regulation, and how the right-of-way is allocated to the vehicles. Some of these approaches imply a regulation agent that performs the regulation alone, others imply inter-vehicular coordination.

A first approach is tackled in (Zou and Levinson, 2003). In this paper, vehicles communicate their information to other vehicles in order to coordinate on the intersection. The vehicles have various trajectories that intersect at conflict points. To perform a right-of-way allocation and provide a crossing date to each vehicle, the agents' behaviour is based on a collaboration scheme. Without any coordination, the vehicles have conflicts. With a basic collision avoidance, the vehicles'

crossing dates are delayed, one at a time, to avoid these conflicts. In the collaboration scheme, the vehicles can change the order in which they are delayed in order to minimise the accumulated delay of the vehicles. However the authors do not provide details about the interaction mechanism performing this minimisation.

(Balan and Luke, 2006) uses the notion of fairness for traffic regulation, by proposing a control policy for intersections based on the history of the vehicles. This policy reduces the variance of the total time spent by the vehicles waiting at red lights during their journeys. Each intersection has a controller able to produce several traffic light patterns. A traffic light pattern is a combination of green and red lights duration for each approach, that avoids conflicts. The controller uses various score functions based on efficiency and fairness to evaluate each possible regulation pattern. The efficiency and the fairness of each pattern are evaluated, for various grid sizes and various traffic loads.

Some of the works on the isolated intersection concern vehicle coordination, and others concern the intersection regulation. AIM (Autonomous Intersection Management) aims to coordinate autonomous vehicles at an intersection. Coordinating these vehicles implies granting the right-of-way to the vehicles, so AIM also performs intersection regulation based on the vehicle information. The following articles treat the subject of AIM.

In (Dresner and Stone, 2008), K. Dresner and P. Stone propose a right-of-way awarding mechanism based on reservations for autonomous vehicles. It relies on a policy called FCFS (First Come First Served), granting the right-of-way to each vehicle requesting, as quickly as possible. This mechanism allows to take into account human drivers (Dresner and Stone, 2007) by using a classical traffic light policy for human drivers, and giving the right-of-way on red lights to automatic vehicles using the FCFS policy. Although this mechanism can accommodate human drivers, its main benefits are derived from the FCFS policy and the presence of autonomous vehicles.

(Grégoire et al., 2013) aims to perform coordination between vehicles approaching an intersection by constructing a priority (oriented) graph. This work proposes a characterisation of feasible priorities using a priority graph. The authors suggest,

the use of heuristics to build such graphs in an optimal way at a future stage of this work.

In (Yan et al., 2014), the different trajectories of the vehicles in the intersection are called streams. For example, all the vehicles coming from the south and going to the west form a stream. Groups of streams are formed, such that groups do not intersect in conflict points. Such streams are called "compatible". The streams of a group can have green lights at the same time. Then the right-of-way awarding problem is represented as a job scheduling problem. Groups of vehicles are formed based on the groups of compatible streams. These groups of vehicles are represented as groups of jobs, and based on this scheduling representation, the overall evacuation time of the vehicle is minimised using exact resolution (branch and bound, dynamic programming).

(Schepperle and Böhm, 2007) and (Schepperle et al., 2009) propose mechanisms to take into account the different valuations of time reduction for the drivers (for example, one minute is more important for a driver being late for a job interview than for a driver driving home from work). In these mechanisms, each vehicle has a budget and can buy or sell time slots. In (Schepperle and Böhm, 2007), an auction mechanism called ITSA (Initial Time Slot Auction) is proposed: while joining the neighbourhood of an intersection every vehicle has the ability to bid a part of its budget in order to get the first available time slot and thus cross the intersection. In (Schepperle et al., 2009), another mechanism is proposed: TSE (Time Slot Exchange). With TSE, vehicles can trade their respective time slots for credits. A hurried driver will be able to spend what he saved to gain time; other drivers will earn credits. A brokerage agent manages these exchanges according to the demands of each driver.

(Vasirani and Ossowski, 2009) also proposes a market-based approach for AIM. While choosing their itineraries, drivers are likely to choose the shortest path according to the estimated travel time of each path. In this model, the drivers have to purchase reservations from the intersection managers in order to cross the intersections. This reservation system provides incentives for drivers to explore alternative paths. In this mechanism, each intersection manager has to determine

its reservation fare in order to maximise its profit. With few vehicles the intersection manager would earn a low profit, but with numerous vehicles it would actually lose profits because of congestion, so it has to adjust the reservation fare to get an average number of vehicles. In order to perform a relevant joint action in fare adjusting, intersection managers use Q-learning.

(Tlig et al., 2014) proposes a synchronisation-based control to manage traffic at an intersection. In this work, the inner area of the intersection is managed by a control agent. This agent uses an alternating principle in order to determine right-of-way for vehicles from different directions through the intersection. The main idea is to compute the speed profile for each vehicle (i.e. acceleration and deceleration) so that it arrives at the intersection at the assigned time and speed.

In (Fayazi et al., 2017), the optimal scheduling for CAVs is dealt with using Mixed Integer Linear Program (MILP). In this mechanism, the constraints that vehicles have to follow in order to build a plan are formalised. These constraints are calculated based on the speed limit and maximum acceleration, as well as the safety gap between vehicles. The gaps are categorised into: gap between vehicles on the same trajectory and gap between vehicles having conflicting movements. Finally, after identifying all the constraints, the mechanism uses MILP to solve the formulated problem, in order to find the optimal solution.

2.3.2 Other Inter-Vehicular Coordination Problems

Regulation and right-of-way allocation is not the only coordination problem existing at an intersection. In the following approaches, vehicles use vehicular communication to perform a real-time coordination for a pre-existing regulation policy.

In (Champion, 2003), coordination between vehicles is used for traffic simulation. By modelling the intersection problem as a 2-player game where the players are the vehicles, and then as a n-player game, it's possible to simulate realistic human behaviours when a priority rule already exists. The moves of the players are "go" and "stop", and the payoff matrix is built by the players, allowing them to choose

the most relevant behaviour. However, the complexity of this method is high and it is difficult to use it for more than a few vehicles.

A common coordination problem for vehicles on an isolated intersection is Collision Avoidance (CA) (de Campos et al., 2013; Hafner et al., 2011). CA consists in adjusting the speed of autonomous vehicles approaching an intersection in order to avoid collisions, and the solution to this problem often involves the use of mechanical equations to find the appropriate speed for each vehicle. The CA therefore does not concern the policy of regulation although the adjustment of the speed of the vehicles can affect their admission date in the intersection. However this aspect is not tackled in detail. For example, (de Campos et al., 2013) states that the approach to the admission order of vehicles can be seen as a simple rule of priority in which the first agent in the sequence has the advantage of keeping its desired motion profile.

(Lee and Park, 2012) presents the right of way as a nonlinear constrained optimisation problem. In this method, the vehicles adjust their speed under various constraints (maximum acceleration and deceleration, maximum and minimum speed, minimum headway distance) in order to minimise the length of the overlapping trajectories of the vehicles in the conflict zone. The computation is performed in a centralised way by an Intersection Control Agent.

2.3.3 Coordinated Regulation for Several Intersections

The following works are based on coordination on the scale of several intersections. Allowing a larger scale coordination provides a better efficiency of the network, for example with green waves formation. A green wave is a phenomenon consisting in coordinating traffic lights in such a way that a group of vehicles can pass through a succession of green lights, reducing the time loss caused by stop-and-go traffic.

(France and Ghorbani, 2003) proposes a hierarchical multi-agent model for traffic regulation. In this model, Local Traffic Agents (LTA) perform a regulation at the intersection scale using sensory data. At a larger scale, an Information Traffic Agent stores information about the state of each intersection. At an intermediate

scale, Coordinator Traffic Agents (CTA) monitor the intersections of an area to provide information to LTAs about the state of their neighbours, particularly congestion, allowing the LTAs to adjust their behaviours and take into account larger scale information and goals.

In (Kosonen, 2003), groups of compatible streams are called "signal groups" and are represented by agents. On the intersection scale, each signal group negotiates with the others to get green light or green time extension according to the size of the queues for each signal group. Fuzzy logic is used to determine whether queues have to be considered as short or long in a non-boolean way. On the network scale, intersections are able to exchange their traffic and control data. This allows signal groups to take into account the neighbours' control decision to get green extensions and cause green waves.

(Camponogara and Kraus Jr, 2003) represents traffic control as a stochastic game. Traffic has various possible states, and various possible traffic policies are possible actions to be performed by an agent representing the intersection controller. A distributed Q-learning is performed to learn and apply the best traffic policy for each traffic state.

(Abdoos et al., 2013) proposes a holonic multi-agent system for traffic signals control. A holonic system is a multi-level system in which each "holon" is made of lower level holons, or atomic agents at the lowest level. In this model, the atomic agent is a signal controller for a single intersection and performs a local regulation based on a Q-learning. Higher level holons represent areas of the network, and their role is to restrict the action space of their sub-holons by giving them abstract actions to perform. Super-holons and sub-holons perform a common Q-learning and each level updates its own policy.

(Hausknecht et al., 2011) shows how an optimisation between several intersections is possible in an AIM context. On an individual scale, itinerary communication allows each intersection manager to produce an estimation of the crossing time for the vehicles. Then, this crossing time is given to the vehicles, allowing these to change their itineraries with realistic estimations of their travel times. On a larger scale, this work addresses Braess' Paradox, whereby opening additional travel

options for the vehicles reduces the efficiency of all vehicles in the system. Indeed, allowing the vehicles to perform a dynamic itinerary choice with self-interested goals leads to this suboptimal Nash equilibrium in which Braess' paradox occurs. Using dynamic lane reversal, the topology of the network is dynamically changed and avoids Braess' Paradox.

Some methods are based on drivers' behaviour, others on traffic light control. (Bazzan et al., 2008) discusses the co-adaptation of vehicles and traffic controllers. Various experiments are made, where only the drivers adapt themselves or only the controllers adapt themselves, or both adapt. They conclude that co-adaptation leads to traffic improvements, especially in large-scale situations involving hundreds of vehicles.

(Marsa-Maestre et al., 2015) presents an approach for congestion management in CSINs (Complex Self-Interested Networks) using negotiation between agents. The network is represented by a graph and divided into subgraphs, called "worlds". This division of the graph is based on graph properties. The main problem is divided into sub-problems, easier to solve, and the agents negotiate to decide where to place the "doors" between each world, allowing the agents to go from one world to another. At the end of this work, a transportation management scenario is succinctly shown, illustrating how this approach could be used for traffic management.

(Doniec et al., 2008) proposes a coordination model for multi-agent systems using anticipation. In this model, the agents compute the consequences of their actions using constraint networks to predict if their actions will cause "undesirable states" and avoid it. This algorithm is applied on a traffic regulation problem on the intersection scale. Vehicles are agents able to Go or to Stop, and have to cross an intersection. Gridlocks may happen in the intersection (because of left-turning vehicles), and are considered as an undesirable state that can be avoided using this algorithm.

(Morales et al., 2011) proposes an approach to build regulation plans in multi-agent traffic control using unsupervised machine learning. On a traffic intersection scenario, vehicles can move and have to avoid collisions. A traffic authority gathers

the information and performs a case-based reasoning, using past experience, to determine the solution to apply in order to solve the current case. Then a norm manager translates the solutions into norms for car agents, who apply these norms using a rule engine. A reduction of the number of norms necessary to accomplish the system goals (avoiding conflict) is also performed in order to reduce the number of norms the agents have to check.

(Dinanga and Pasin, 2014) proposes signals traffic regulation policies based on the real-time position of each vehicle around the intersection. These policies are based on the number of vehicles waiting in each queue around the intersection.

(Ashtiani et al., 2018) extends the MILP single intersection management in (Fayazi et al., 2017), proposing an optimisation over multiple intersections. Based on the access time of a vehicle through an intersection, the mechanism compute its desired access time through the next intersection in the trajectory. Using this mechanism, the intersection controllers have knowledge about both vehicles presented in the area, and vehicles that are about to arrive. The information is then taken into account in the MILP optimisation.

In (Junges and Bazzan, 2008), R. Junges and A.L.C. Bazzan propose a novel traffic light synchronisation problem using Distributed Constraint Optimisation (DCOP). In this work, intersections exchange messages using DCOP algorithms to optimise over the quality of the traffic lights plan. It shows that solution quality achieved with DCOP is of good quality, but communication overhead and computational time tends to be an issue for such system. In literature, DCOP is widely used and has been shown to be effective in task allocation and meeting scheduling problems (Macarthur et al., 2011; Farinelli et al., 2008; Modi and Veloso, 2004).

2.3.4 Discussion on Existing Multi-Agent Approaches

MAS-based approaches presented above rely on different assumptions and environment than the classic intersection control system. Classic approaches gather information through a number of loops and sensors installed in the infrastructure. They are used to measure macroscopic information about the traffic such as traffic

flow, velocity and density. In contrary, MAS-based approaches rely on accurate perception of the position, velocity and desired destinations from vehicles on the network, and therefore, can model the state of the network more precisely. This assumption is reasonable in an environment where vehicles are highly equipped and have significant communication capabilities. In a future state of transportation systems, the presence of CAVs can reasonably make this type of hypothesis because CAVs are proposed with these capacities.

MAS-based approaches can further be categorised into several types, based on their control variables. Some gather precise information from vehicles to optimise the traffic light plan (Balan and Luke, 2006; Kosonen, 2003; Junges and Bazzan, 2008). In contrary, others aim to regulate intersections without using traffic lights (Dresner and Stone, 2008; Vasirani and Ossowski, 2009; Tlig et al., 2014). Such systems require communicating to each vehicle approaching an intersection its desired crossing time. The signal received can be similar to an individual traffic light, where each vehicle has a small time window to cross. Therefore, when opting for such solutions, vehicles must be able to react precisely to their plan (i.e. acceleration, deceleration, cross).

Regarding the level of decentralisation, most approaches presented above are centralised (Dresner and Stone, 2008; Vasirani and Ossowski, 2009; Fayazi et al., 2017), i.e. they require a control agent that optimises over the situation and sends a solution to the traffic light system or to the vehicles. However, decentralised systems are believed to have some advantages over centralised systems when dealing with problems that are distributed by nature such as traffic regulation problems. For example, one can list the privacy by not transmitting all the data to the central agent, a better robustness due to not having a central point of failure and the parallelisation of certain computations. Furthermore, a decentralised approach would only require Vehicle-to-Vehicle communication and thus can be adopted in rural areas with minimal infrastructure level.

In terms of optimisation problems, most of traffic regulation mechanisms measure their performance using the average delay or average travel time of vehicles. This performance indicator has been widely considered one of the best since a lower

average delay leads to lower fuel consumption and greater comfort for passengers. However, when facing a realistic setup, the systems need to be able to consider more criteria. Indeed, in an urban traffic area, there are various types of vehicles, each with its own intention and each vehicle values its time differently based on their types (e.g. buses, road maintenance, other priority vehicles) or on the objectives of their travels (e.g. going to a hospital, going to work and others).

We also notice that most microscopic traffic regulation methods solve optimisation problems locally, at the scale of an intersection. These approaches can at times be greedy in certain scenarios because optimising traffic on each intersection locally does not guarantee an optimal result on a global scale. On the other hand, optimising traffic in a microscopic way over the network is highly computational expensive. Hence, a mechanism needs to be able to take into account global information, without raising additional complexity issues.

2.4 Summary

We surveyed the different approaches proposed from the AI community as well as from the transportation community to mitigate traffic congestion. Specifically, in Section 2.1, we denoted the areas in an urban traffic system. Section 2.2 presented the classic methods widely used for urban traffic regulation. These methods ranged from the basic one being traffic signs, to more advanced technologies that percept information from magnetic loops and sensors to optimise traffic lights.

After that, Section 2.3 highlighted several contributions from the multi-agent community to address traffic regulation problems. We discussed in this section both traffic lights optimisation approaches and microscopic approaches where each vehicle's crossing time is decided individually. Finally, we discussed these approaches to identify the differences and the limits of some notable ones.

Chapter 3

Background on Distributed Constraint Optimisation Problems

In this chapter, we give a brief literature on Distributed Constraint Optimisation Problems (DCOPs) and their applications in multi-agent systems. First we discuss different constraints handling frameworks then we define the DCOP framework (Section 3.1). We then survey in section 3.2 different DCOP algorithms. At the end of this chapter, we discuss the current applications in MAS that are proposed using DCOPs (Section 3.3).

3.1 Distributed Constraint Optimisation Problems

For several decades, coordination problems between agents have drawn considerable attention in literature. One of the techniques that are mostly used in the multi-agent community is by formulating these problems using a constraint handling framework. Using the framework often requires formulating the coordination problems with a constraint network. Generally, in a constraint network, agents

are represented with nodes, while constraints between agents are represented using edges. Constraints can either be *hard constraints* which return a binary value for each joint assignment or *soft constraints* which can return a real value based on the degree of satisfaction.

In problems that can be formulated only using hard constraints, finding value assignment for all variables can be referred as a Constraint Satisfaction Problem (CSP). In multi-agent systems, resources are often distributed among agents. In this case, a CSP can be extended as a Distributed Constraint Satisfaction Problems (DisCSPs).

In some MAS domains, problem formulations require using soft constraints. Hereby, the DisCSPs is generalised as a Distributed Constraint Optimisation Problems (DCOPs). In DCOPs, agents aim to choose values for their variables, to either minimise the cost or maximise the utility of a set of constraints.

Definition 3.1. A Distributed Constraint Optimisation Problem (or DCOP) is defined by a tuple $\langle \mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$, where:

- $\mathcal{A} = \{a_1, ..., a_n\}$ is a set of n agents.
- $\mathcal{X} = \{x_1, ..., x_n\}$ are variables owned by the agents, where variable x_i is owned by agent a_i. An agent can own a number of variables. In the scope of our problem, only single variable DCOP is needed and thus, for the sake of simplicity, we limit our literature review to this type of problem.
- $\mathcal{D} = \{\mathcal{D}_{x_1}, ..., \mathcal{D}_{x_n}\}$ is a set of finite-discrete domains. A variable x_i takes values in $\mathcal{D}_{x_i} = v_1, ..., v_k$.
- $\mathcal{C} = \{c_1, ..., c_m\}$ is a set of constraints, where each c_i defines a cost $\in \mathbb{R} \cup \{\infty\}$.

The constraints between one or several variables can naturally be represented by a constraint graph G (cf. Figure 3.1). Depending on the problem, a solution to the DCOP is an assignment to all variables that either minimise or maximise

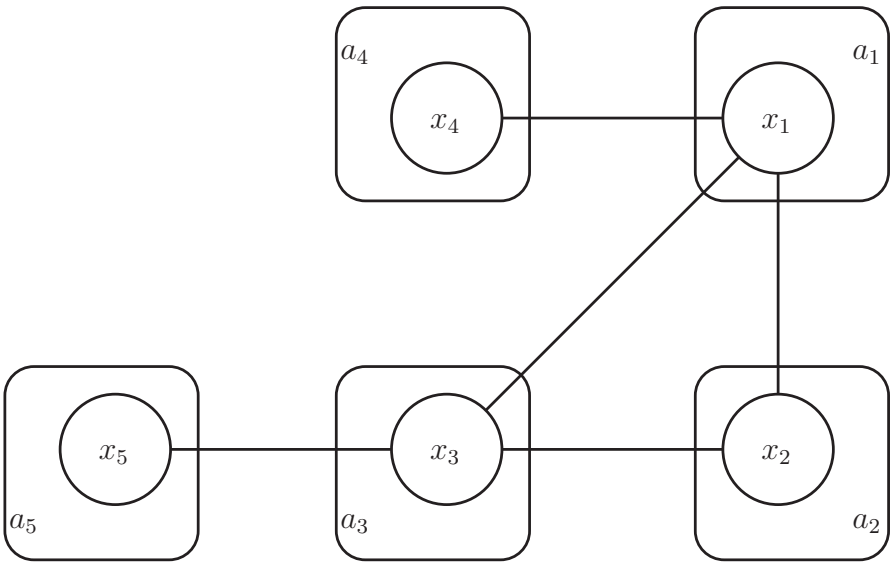

FIGURE 3.1: Example of a constraint graph representing five variables x_1 to x_5, each controlled by one agent, a_1 to a_5 respectively. A link between two variables represents a constraint between them.

the aggregated global objective function. Hence, the assigned values of all the variables, X^*, are produced as:

$$X^* = \arg\max_X \sum_{i=1}^{m} c_i \; OR \; X^* = \arg\min_X \sum_{i=1}^{m} c_i \tag{3.1}$$

3.2 DCOP Algorithms

In this section, we highlight some notable DCOP algorithms. DCOP algorithms are often categorised into two different categories: exact algorithms and non-exact algorithms. In addition, they can be further categorised into several groups based on their level of decentralisation and the way local information is updated.

First, algorithms can either be *fully decentralised* while some are classified as *partially centralised* algorithms. Partially centralisation is often proposed to allow agents to have a better local view and thus, improves the performance. The trade off is the loss of privacy since the central agent needs to have access to knowledge of other agents in the group.

Based on the synchronicity of the algorithms, they can be categorised as *synchronous* and *asynchronous*. Synchronous algorithms define the notion of cycle. Each cycle consists of several actions for each agent. One cycle needs to be finished before the algorithm advances to the next one. In asynchronous algorithms, agents can update their assignment solely based on their local view of the problem, without waiting for decisions of the other agents. This minimises the idle time of each agent and allows them to react quickly but was shown to have a negative impact on communication overhead and performance of the algorithms (Peri and Meisels, 2013).

3.2.1 Exact Algorithms

Exact algorithms guarantee to find a solution that optimises the objective function for a DCOP instance. Note that even though the constraint graph (cf. Figure 3.1) is the standard way to represent a DCOP instance, each algorithm performs some pre-processing steps and thus, operate on a slightly different version of the constraint graph. Thereafter, we present some well-known exact algorithms.

3.2.1.1 ADOPT and Its Variants

Asynchronous Distributed OPTimisation (ADOPT) (Modi et al., 2005) is a fully decentralised, asynchronous exact algorithm. ADOPT first organises agents into a Depth-First Search (DFS) tree (cf. Figure 3.2). When transforming the original constraint graph to a DFS tree, constraints are not allowed between agents in different branches of the tree. Therefore, these agents can search for solutions independently of each other. ADOPT then uses three kinds of messages: VALUE, COST and THRESHOLD. The goal of the algorithm is to update at each agent the lower and upper bounds on the solution cost at each subtree rooted by the agent itself. Upon receiving a message, an agent chooses the value with minimum cost, then sends these messages (VALUE messages to all of its descendants, COST message to its parent and THRESHOLD messages to its direct child nodes). The

process repeats until termination condition, i.e. when the lower bound meets the upper bound at the root node of the tree. The optimal solution is therefore found.

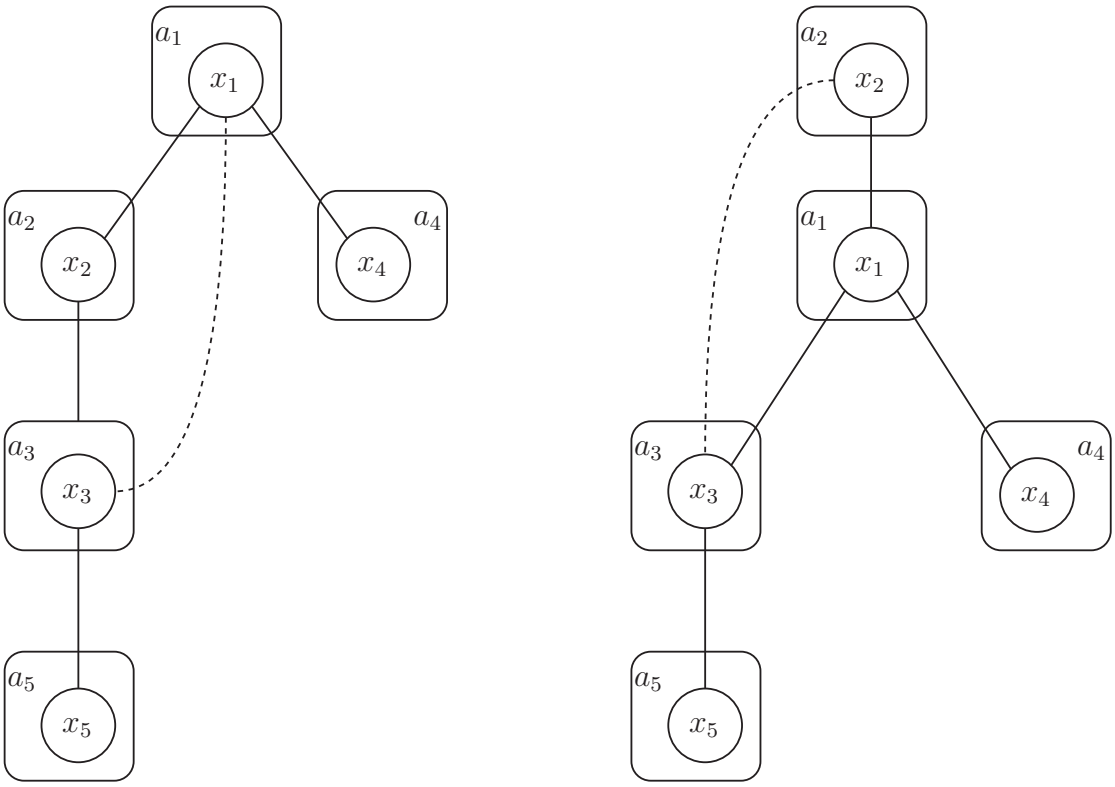

FIGURE 3.2: Two possible transformations of the Constraint Graph described in Figure 3.1 to a DFS tree. For example, on the left a_1 is the root node. Its descendants include every other nodes, while a_2 and a_4 are its direct childs. On the right is another possible way of transforming, using a_2 as the root node.

ADOPT was further extended in several ways. The most well-known version being BnB-ADOPT (Yeoh et al., 2008). This algorithm uses the same structure and message from ADOPT, while proposing the use of branch-and-bound strategy to improve computation. Other extensions are proposed in (Gutierrez et al., 2011), a combination of ADOPT and BnB-ADOPT, or BnB-ADOPT$^+$ (Gutierrez and Meseguer, 2010), a variant of BnB-ADOPT without redundant messages.

3.2.1.2 OptAPO

Optimal Asynchronous Partial Overlay (OptAPO) (Mailler and Lesser, 2004) is an exact, partially decentralised, asynchronous DCOP algorithm. In OptAPO,

mediator agents are used to solve conflicting assignment between neighbouring agents. In more details, OptAPO has three steps: initialisation, checking the agent view and mediation, with the mediation step being the most important part of the process. In this step, the mediator agents, decided earlier using a priority order, solve subproblems using a centralised branch-and-bound search. Petcu et al. (2006) shows that mediator agents occasionally solve overlapping problems, thus reducing their efficiency.

(Grinshpoun and Meisels, 2008) have improved OptAPO to address the incompleteness problem of the original version. In this work, the authors identified scenarios that can lead OptAPO to run infinitely due to the asynchronicity of the mediators, then proposed the complete version of OptAPO, named CompOptAPO.

3.2.1.3 DPOP and Its Variants

Distributed Pseudotree Optimisation Procedure (DPOP) (Petcu and Faltings, 2005b) is a fully decentralised, synchronous exact algorithm, In more details, DPOP operates using the DFS tree. Each agent in DPOP takes the role of the node which represents its own variable. The main process of DPOP consists in computing and exchanging messages between nodes. Each iteration is executed in 3 phases. In the first phase, agents exchange messages in order to generate a proper tree graph, which later serves as a communication structure for the next steps. After that, the second phase starts from the leaves of the tree. Each agent computes its $Util$ matrix depending on its value and on the values from its children and propagates the matrix in the upward direction. This $Util$ matrix summarises the influence of the sending agent and its neighbours on the next step. After the second phase, the third phase is started by the root. In this phase, the $Value$ messages are computed and propagated downward. Each agent then computes its optimal value based on its $Util$ matrix and the $Value$ message received from its parents.

Similar to ADOPT, DPOP has been extended in many different ways. In PC-DCOP (Petcu et al., 2006), the *Util* phase is partially centralised when the dimension of the *Util* matrix exceeds a certain threshold. Partially centralising improves the performance of DPOP, but leads to a loss of privacy. ADPOP (Petcu and Faltings, 2005a) is an approximate version of the algorithm. The solution quality achieved is therefore lower, in exchange for lower computational time.

3.2.2 Non-Exact Algorithms

Finding an optimal solution for a DCOP is known to be NP-hard (Modi et al., 2005). In practice, resources are often limited and thus, applying exact algorithms raises a serious issue regarding scalability. Non-exact algorithms are proposed to address this issue, trading off solution quality for computational and communication requirements. In this section, we will discuss several well-known non-exact algorithms that are used in different DCOP problems.

3.2.2.1 MGM

Maximum Gain Message (MGM) (Maheswaran et al., 2004a) is a non-exact, synchronous, fully decentralised algorithm that is based on a local greedy search strategy. Each agent assigns a random value to its variable, then sends the assignment to all its neighbours. Upon receiving all messages from its neighbours, an agent selects a better value that maximises its gain, then propagates the gain to neighbouring agents. If the local gain of an agent using the selected value is greater than the maximum gain of neighbouring agents, the selected value is assigned. The two-phase process repeats until a termination condition is met.

3.2.2.2 DSA

Distributed Stochastic Algorithm (DSA) (Zhang et al., 2005) is a non-exact, synchronous, fully decentralised algorithm. DSA is similar to MGM, without the gain propagating phase. Each agent only stochastically choses between values based on

its gain. DSA also terminated when a condition is met [1]. MGM and DSA both cannot provide a bound to the quality of the solution found.

3.2.2.3 Max-Sum and Its Variants

Unlike other algorithms, Max-Sum (Farinelli et al., 2008) and its variants operate on a factor graph (cf. Figure 3.3). In a factor graph, nodes are categorised into variable nodes and function nodes. Each variable node represents one variable, while each function node describes a constraint. One variable node is linked with one function node if and only if the variable is concerned in the constraint. Variable nodes are controlled by their controlling agents. The control of a function node is delegated to one of the agents controlling the neighbouring variable nodes.

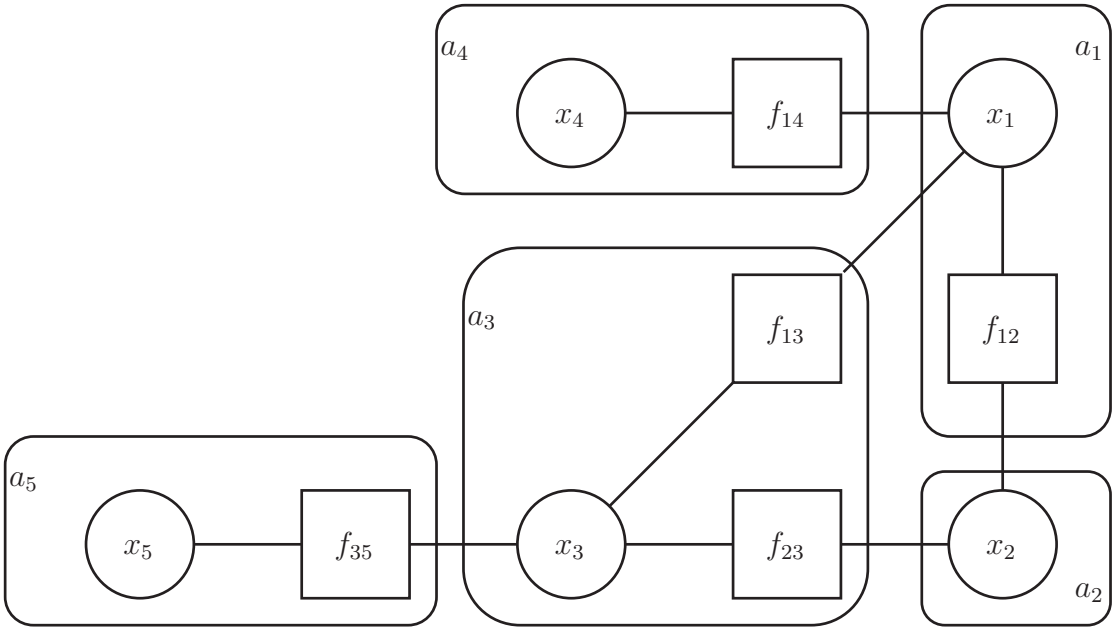

FIGURE 3.3: A factor graph describing the problem represented in Figure 3.1. A function node f_{ij} represents the constraint between x_i and x_j and is either controlled by a_i or a_j,

Max-Sum is a non-exact, synchronous, fully decentralised algorithm based on belief propagation. Messages are recursively propagated between variable nodes and function nodes. After each iteration, variable nodes can update their beliefs based

[1]The algorithms usually terminated after a fixed number of steps.

on messages received from neighbouring factor nodes. Max-Sum is guaranteed to converge in acyclic graph (i.e. graphs that do not contain cycles). When there is no ties among utilities, the algorithm will converge to the optimal solution. In cyclic graphs, convergence is not guaranteed but extensive empirical evidence shows that the algorithm generates good approximate solutions (Kschischang et al., 2001).

Max-Sum has many different variants. Fast Max-Sum (Ramchurn et al., 2010) improves Max-Sum in a RoboCup rescue environment by restricting the value exchanged by nodes, therefore reducing communication and computation. Bounded Max-Sum (Rogers et al., 2011) proposes a bounding technique to the quality of the solution by removing edges from cyclic factor graph, then run Max-Sum on the acyclic graph. Recently Max-Sum_AD_VP (Zivan and Peled, 2012) proposes a novel way to exchange messages by alternating the direction through phases. In this algorithm, the factor graph is transformed into a directed graph by ordering nodes. Max-Sum_AD_VP usually uses nodes index as the order (cf. Figure 3.4). The algorithm is proven to converge faster than standard Max-Sum on acyclic graph (Zivan and Peled, 2012; Chen et al., 2017).

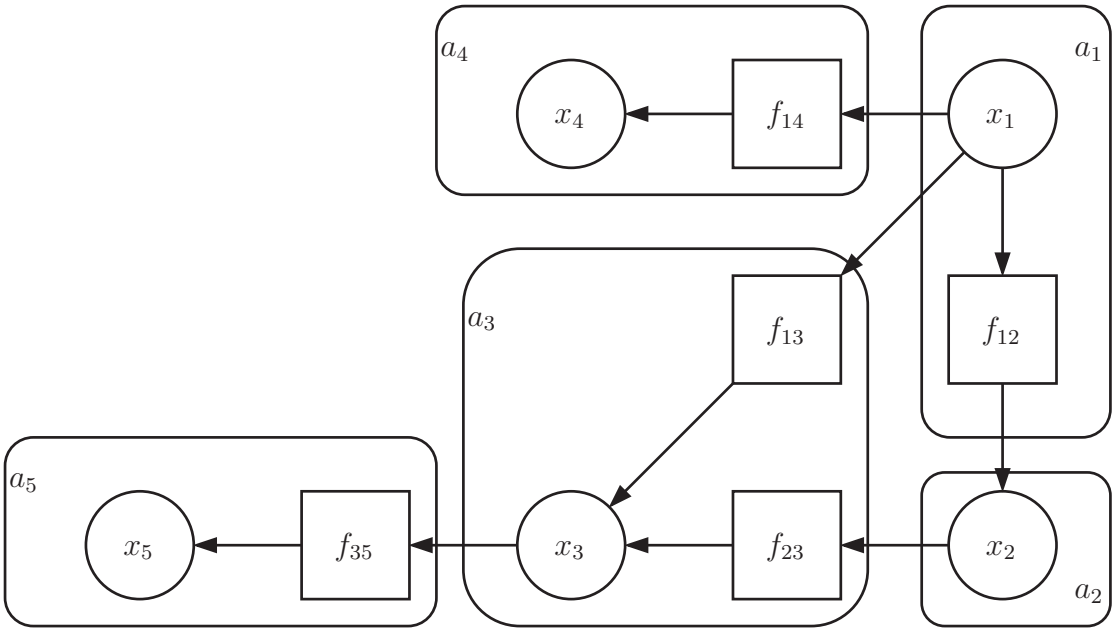

FIGURE 3.4: A directed factor graph used in Max-Sum_AD_VP describing the problem represented in Figure 3.1. Nodes are ordered using their index.

3.3 DCOP Applications

In recent decades, DCOP solutions have been proposed to various multi-agent system problems (Fioretto et al., 2018). Applications range from coordination problems for disaster respond (Carpenter et al., 2007), meeting scheduling (Maheswaran et al., 2004b), mobile sensor teams (Zivan et al., 2009; Yedidsion et al., 2018) to load distribution in smart grids (Davidson et al., 2009; Miller et al., 2012). Through these applications, DCOP is shown to be a flexible way to model distributed optimisation problems. Moreover, DCOP algorithms are regularly proposed in a general way and thus, modelling a problem as a DCOP can take advantage of a wide range of research, each with its own benefit (e.g. privacy, robustness, time complexity, communication overhead).

In regard to the intelligent transportation system, a first work using DCOP has been proposed for traffic light synchronisation problem (Junges and Bazzan, 2008). By using a DCOP formulation, this system aims to coordinate multiple intersections in order to create green waves, allowing vehicles to cross multiple intersections in an efficient way.

DCOP has not yet been adopted to solve right-of-way allocation problems at intersections. Indeed, there are several major challenges when applying DCOP in this highly dynamic environment. First, the runtime of DCOP algorithms tends to increase exponentially, which makes it hard to find a solution in rush hours. One can overcome this issue by applying incomplete algorithms, but the solution quality must be evaluated. Moreover, such system needs to be enhanced to ensure a solution is found even if algorithms failed to converge. Finally, the system needs to be robust when facing dynamic events that occur in traffic (e.g. emergency vehicles, incidents, road maintenance). In this thesis, we discuss the possibility of formulating the problem as a DCOP, then evaluate several DCOP algorithms. Based on the evaluation, we choose the most suited algorithm and propose improvements to the algorithm to respond to the formulated problem.

3.4 Summary

This chapter surveyed the DCOP approach that has been studied for over a decade in the multi-agent community. Specifically, we gave the definition of a DCOP in Section 3.1. We identified in Section 3.2 some notable DCOP algorithms and categorised them based on their completeness, their level of decentralisation and synchronicity. Finally, in Section 3.3 we gave some examples of applying DCOP for MAS coordination problems.

Through this survey, we can notice that DCOP is a popular framework used for distributed reasoning and can be applied in many MAS domains. However, when taking DCOP to microscopic traffic regulation problems, exact algorithms raise an issue due to their complexity which is usually exponential in either memory or communication overhead. On the other hand, non-exact algorithms are less complex but can converge to an awful solution, even an invalid one (i.e. a solution that has an infinite cost due to the violation of a structural constraint). In the next chapter, we propose a DCOP formulation of the right-of-way allocation problem at an intersection and discuss the use of DCOP algorithms to address this problem.

Chapter 4

Cellular Model for an Intersection

In this chapter, we first present in Section 4.1 a simple cellular model of an intersection. We then define a configuration, which we aim to build based on vehicles' positions and the optimisation problem that we are facing. Next, Section 4.2 proposes a formulation of our problem as a DCOP and evaluates the performance of some notable DCOP algorithms on the problem. We then discuss continuity of the solution in Section 4.3. In Section 4.4, we opt for an algorithm, the Max-Sum algorithm, then based on its properties, we propose another DCOP formulation, and an improvement to Max-Sum using a pruning technique. Finally, we evaluate the performance of each proposition in Section 4.5.

4.1 Problem Statement

We first model an intersection using a cellular automaton model (cf. Figure 4.1). This model is widely used in literature because it retains the main properties of a network while being relatively simple to use (Brockfeld et al., 2001; Maerivoet and Moor, 2005). An intersection is composed of several *incoming lanes*, several *outgoing lanes*, and a central zone called *conflict zone*. The path of a vehicle across the intersection is called a *trajectory*. Each incoming lane and trajectory is a succession of cells. A cell inside the conflict zone is called a *conflict spot*.

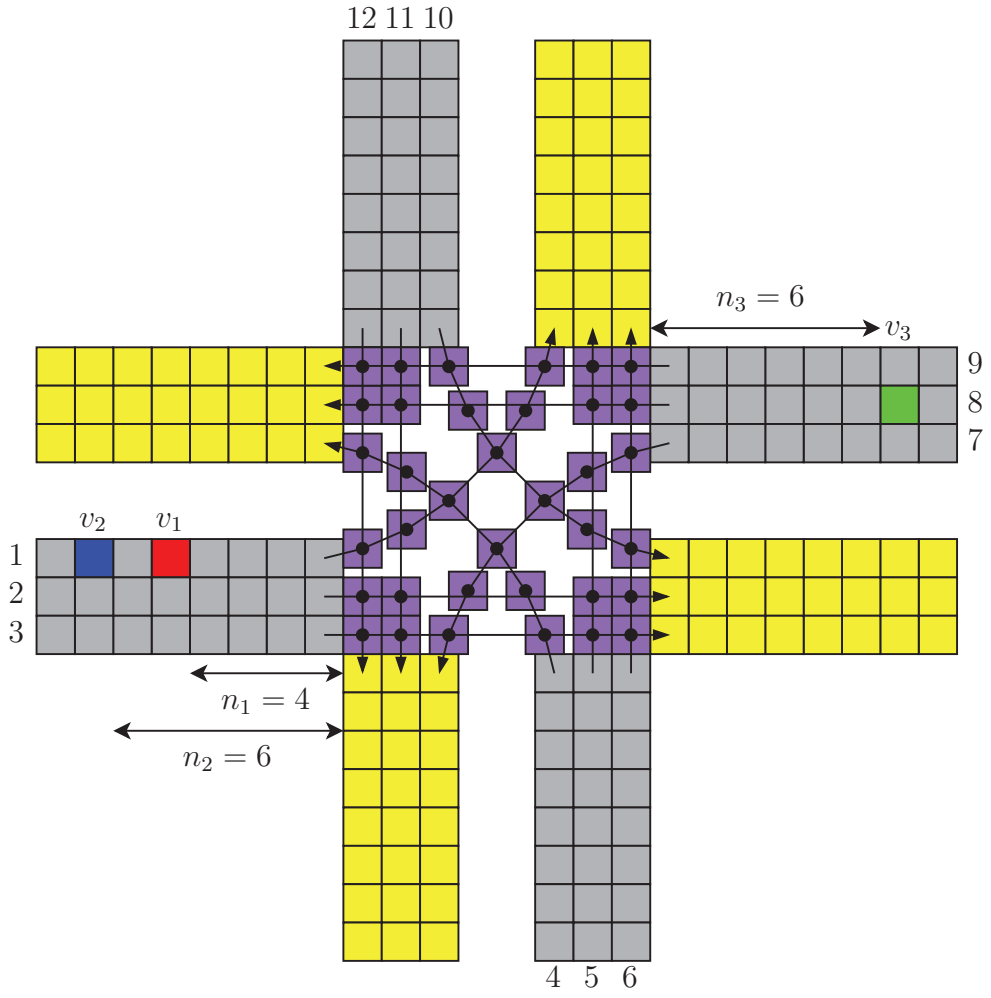

FIGURE 4.1: Intersection with 12 incoming lanes (in gray), 12 outgoing lanes (in yellow) and a conflict zone (in purple), all divided in cells. The incoming lanes are numbered from 1 to 12. The conflict zone is crossed by various trajectories. The cells belonging to several trajectories (every cell of the conflict zone in this case) are conflict spots. There are 3 vehicles v_1 (red rectangle), v_2 (blue rectangle) and v_3 (green rectangle). v_1 and v_2 are heading north, while v_3 is heading west.

The main objective of the system is to minimise travel time. The travel time of a vehicle consists of the time it needs to travel through its journey at its speed, and a waiting time. Thus, to minimise the travel time, we must minimise the waiting time of vehicles. Our objective is to assign to each vehicle an admission time to the conflict zone. A vehicle's admission time is the time that this vehicle can begin crossing the intersection, similar to an individual traffic light system. We define, for each time step t a *configuration* Φ_t as the set of admission times of vehicles in front of the conflict zone.

This configuration must satisfy the following rules:

- The configuration must ensure that vehicles can cross the intersection at their admission time safely and without stopping inside the conflict zone.
- A vehicle must have only one admission time at a time.
- The current configuration must be accessible by all vehicles so they share the same agreement any time.

In order to build this configuration, we model the right-of-way allocation problem as follows.

Definition 4.1. Let t be the current time step and V_t the set of all vehicles approaching the intersection. A configuration is a set $\Phi_t = \{\varphi_1, ..., \varphi_k\}$ where each φ_i is the admission time in the conflict zone assigned to each $v_i \in V_t$.

Let L be the set of incoming lanes and $l_k \in L$ be the lane k. For each $v_i \in V_t$, let $l_{v_i} \in L$ be the lane in which the vehicle v_i is present, n_i be the distance (in number of cells) between v_i and the conflict zone, and τ_i be v_i's trajectory inside the conflict zone. Let e be one of the cells in trajectory τ_i, $pos(e, \tau_i)$ is the distance, in number of cells, between the cell e and the first cell of τ_i. The position of the first cell of τ_i is 0. Let s_i be the speed of the vehicle v_i in cells per time step.

We aim to build, for each time step t, a configuration Φ_t for all vehicles in V_t that minimises their total waiting time. The input is the set of vehicles V_t presented

in the system at the current time step and the configuration at the last time step Φ_{t-1}. Let w_i be the waiting time of the vehicle v_i and Φ be the set of all possible configurations (this waiting time can be changed to weighted waiting time to take into account a vehicle's priority). Thus our goal is to search for a minimisation:

$$f : (t, V_t, \Phi_{t-1}) \mapsto \arg\min_{\Phi_t \in \Phi} \sum_{v_i \in V_t} w_i \tag{4.1}$$

To ensure that the configuration Φ_t satisfies the rules described above, the admission times of vehicles in the configuration must follow some structural constraints.

c1. Distance constraint A vehicle has to cross the distance separating it from the conflict zone before entering it:

$$\forall v_i \in V, \varphi_i > t + \frac{n_i}{s_i} \tag{4.2}$$

c2. Anteriority constraint In our model, we consider that no overtaking is possible when vehicles are close to the intersection. Thus a vehicle v_j cannot enter the conflict zone before the vehicles v_j preceding it on its lane. This constraint should be modified in a more complex model that takes into account overtaking. We have:

$$\forall v_i, v_j \in V_t^2, l_{v_i} = l_{v_j}, n_i < n_j \Rightarrow \varphi_i < \varphi_j \tag{4.3}$$

c3.a Simple conflict constraint Two vehicles cannot be in the same cell at the same time in the conflict zone. If the vehicles belong to the same lane, the anteriority constraint covers this case. However, if two vehicles v_i and v_j coming from different lanes, having a conflict spot in their trajectories, their admission times must ensure that they are not present in the conflict spot at the same moment. Thus, we have:

$$\begin{aligned} &\forall v_i, v_j \in V_t^2, \forall e \in \tau_i, e \in \tau_j \Rightarrow \\ &(\varphi_i + \tfrac{pos(e,\tau_i)}{s_i}) \neq (\varphi_j + \tfrac{pos(e,\tau_j)}{s_j}) \end{aligned} \tag{4.4}$$

c3.b Conflict constraint with safety lapse We can further restrict constraint c3.a for safety reasons. Indeed, adding a time lapse t_{safe} between the passing of a vehicle v_i on a cell c and the passing of another vehicle v_j, in a conflicting trajectory on this cell, enhances the drivers' safety. Thus, v_j can only occupy this cell after a t_{safe} duration of v_i's occupation. The simple conflict constraint can be replaced by the following:

$$\begin{aligned} &\forall v_i, v_j \in V_t^2, \forall e \in \tau_i, e \in \tau_j \Rightarrow \\ &|(\varphi_i + \tfrac{pos(e,\tau_i)}{s_i}) - (\varphi_j + \tfrac{pos(e,\tau_j)}{s_j})| > t_{safe} \end{aligned} \tag{4.5}$$

Example 4.1. *Consider the scenario presented in Figure 4.1. Assuming the speed of all vehicles is 1 cell/time step. Let $e(i,j)$ be the conflict cell between two trajectories τ_i and τ_j, the distance (in cell) from the the first cell of τ_i to the conflict spot is presented in the table 4.1. Therefore, the structural constraints can be described as:*

c1: *v_1 has 4 cells to travel before entering the conflict zone, thus $\varphi_1 > 4$. By the same logic, $\varphi_2 > 6; \varphi_3 > 6$.*

c2: *v_2 cannot overtake v_1, therefore $\varphi_2 > \varphi_1$.*

c3.b: *There is a conflict spot between the trajectory of v_1 and that of v_3. The conflict spot is the cell number 4 in v_1's trajectory and the cell number 2 in v_3's trajectory. Let the safety lapse be 1 time step, we have:*

$$|(\varphi_1 + 4) - (\varphi_3 + 2)| > 1 \tag{4.6}$$

v_2 has the same conflict spot with v_3, we also have:

$$|(\varphi_2 + 4) - (\varphi_3 + 2)| > 1 \tag{4.7}$$

We next formalise the right-of-way allocation problem as a distributed constraint optimisation problem.

TABLE 4.1: Distance to conflict point between two trajectories τ_i and τ_j. The cell is blank if there is no conflict between them.

τ_i \ τ_j	Lane: 1 Left turn	Lane: 2 Straight	Lane: 3 Straight	Lane: 3 Right turn	Lane: 4 Left turn	Lane: 5 Straight	Lane: 6 Straight	Lane: 6 Right turn	Lane: 7 Left turn	Lane: 8 Straight	Lane: 9 Straight	Lane: 9 Right turn	Lane: 10 Left turn	Lane: 11 Straight	Lane: 12 Straight	Lane: 12 Right turn
Lane: 1 Left turn					2					4	5		3	1	0	
Lane: 2 Straight					3	4	5		2					1	0	
Lane: 3 Straight					3	4	5		2					1	0	
Lane: 3 Right turn															0	
Lane: 4 Left turn	3	1	0						2					4	5	
Lane: 5 Straight		1	0						3	4	5		2			
Lane: 6 Straight		1	0						3	4	5		2			
Lane: 6 Right turn			0													
Lane: 7 Left turn		4	5		3	1	0						2			
Lane: 8 Straight	2					1	0						3	4	5	
Lane: 9 Straight	2					1	0						3	4	5	
Lane: 9 Right turn							0									
Lane: 10 Left turn	2					4	5		3	1	0					
Lane: 11 Straight	3	4	5		2					1	0					
Lane: 12 Straight	3	4	5		2					1	0					
Lane: 12 Right turn											0					

4.2 DCOPs for Intersection Management

Centralised solutions to traffic regulation result in high computational requirements for one agent. Moreover, centralised approaches create a single point of failure and have a lack of scalability and adaptability to dynamic events such as accidents or the arrival of an emergency vehicle. In such a dynamic context, using a decentralised approach allows to be proactive to any change in traffic control. This is particularly relevant in the light of connected vehicles capable of advanced computations. In this section, we present a decentralised formalisation of traffic regulation model using a DCOP. This formalisation allows every agent to coordinate by exchanging messages with their neighbours, thus reduces the computational requirements for each agent.

A Distributed Constraint Optimisation Problem (or DCOP), as defined in Definition 3.1, is a tuple $\{\mathcal{A}, \mathcal{X}, \mathcal{D}, \mathcal{C}\}$, where: $\mathcal{A} = \{a_1, ..., a_n\}$ is a set of n agents; $\mathcal{X} = \{x_1, ..., x_n\}$ are variables owned by the agents, where variable x_i is owned by agent a_i; $\mathcal{D} = \{\mathcal{D}_{x_1}, ..., \mathcal{D}_{x_n}\}$ is a set of finite-discrete domains. A variable x_i takes values in $\mathcal{D}_{x_i} = \{v_1, ..., v_k\}$; $\mathcal{C} = \{c_1, ..., c_m\}$ is a set of constraints, where each c_i defines a cost $\in \mathbb{R}_+ \cup \{\infty\}$. A solution to the DCOP is an assignment to all variables that minimises $\sum_i c_i$.

There are several ways to formalise a problem as a DCOP, depending on what agents, variables and constraints represent. The most natural way to model our problem as a DCOP is to model each vehicle using an agent. Here we name it the vehicles-based approach.

4.2.1 Vehicle-based Approach

The vehicle-based approach consists of modelling all the vehicles as agents. The number of agents is also the number of vehicles arriving at the intersection. Each agent holds a variable which corresponds to the vehicle's admission time to the intersection. The domain of the variables varies from $t + \frac{n_i+1}{s_i}$, which is the earliest possible admission time of this vehicle taking into account its distance to the

conflict zone, to $t + \frac{n_i+1}{s_i} + p$. p is the time window for the waiting time of each vehicle. A small window may limit the search and makes it impossible to find a solution, while a large window adds unnecessary complexity to the problem. The value of this time window will be detailed in Section 4.4.3. Since the domain of the variables already takes into account the distance constraint described in Equation 4.2, we map the other structural constraints described in Equation 4.3 and Equation 4.5 as follows:

Anteriority constraint

$$c_1(\varphi_i, \varphi_j) = \begin{cases} \infty & \text{if } l_{v_i} = l_{v_j}, n_i < n_j \text{ and } \varphi_i > \varphi_j \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

Conflict constraint with safety lapse

$$c_2(\varphi_i, \varphi_j) = \begin{cases} \infty & \text{if } \exists e \in \tau_i, e \in \tau_j \text{ and} \\ & |(\varphi_i + \frac{pos(e,\tau_i)}{s_i}) - (\varphi_j + \frac{pos(e,\tau_j)}{s_j})| \leq t_{safe} \\ 0 & \text{otherwise} \end{cases} \tag{4.9}$$

In order to formalise our objective (Equation 4.1) as a DCOP, each vehicle holds a cost constraint, which directly links to its waiting time. Thus, we also have:

Waiting constraint

$$c_3(\varphi_i) = \varphi_i - (t + \frac{n_i + 1}{s_i}) \tag{4.10}$$

The objective of a DCOP is to minimise $\sum_{c_i(.) \in \mathcal{C}} c_i(.)$. This optimisation represents the goal of the system (minimise the global waiting time of vehicles without violating any structural constraint).

4.2.2 Evaluating the Performance of DCOP Algorithms

After formulating our problem as a DCOP, we evaluate the performance of some notable DCOP algorithms. We do that by randomly generating traffic scenarios,

then we use the algorithm to solve the generated problem. DCOP algorithms are evaluated using Frodo (Léauté et al., 2009), a DCOP solver. We test each algorithm on a varied number of vehicles using the intersection presented above (cf. Figure 4.1). The algorithms terminate whenever convergence is achieved or the timeout is reached.

In this evaluation, we measure the performance of five DCOP algorithms (ADOPT, DPOP, Max-Sum, DSA, MGM) on different factors (e.g., solution quality, runtime, communication overhead).

First, Figure 4.2 shows the success rate of each DCOP algorithm in giving a valid solution in time (a valid solution is a solution that does not violate any structural constraint). For the sake of simplicity, we set the timeout of the process at 6 seconds. In this evaluation, we observe that greedy incomplete algorithms (DSA, MGM) always provide a feasible solution. This is due to the fact that constraints between two agents (i.e. anteriority constraints and conflict constraints) are hard constraints, which makes the problem similar to a classic satisfaction problem. On the other hand, the success rate of complete algorithms (ADOPT, DPOP) decreases drastically from 10 agents. This suggests that the use of a complete algorithm is rather impossible in our problem.

Figure 4.3 shows the solution quality of each algorithm. ADOPT and DPOP provide the optimal solution and thus, are of the best quality. For the sake of comparison between incomplete algorithms, the average solution quality is only taken for problem instances where all the 3 algorithms provide a feasible solution. Max-Sum does not perform well compared to DSA and MGM as it fails about 25-30% of the time and when they all converge, the solution quality from Max-Sum is lower. Overall, the solutions computed using any DCOP algorithm are better than the ones using a simple FCFS mechanism as DCOP algorithms try to optimise solution quality.

Regarding communication overhead, the messages size of the algorithms are consistent (cf. Figure 4.5), except for DPOP which is known to grow exponentially with the complexity of the pseudo tree (Fioretto et al., 2018). Due to the termination condition, DSA and MGM agents, who have less computational requirement

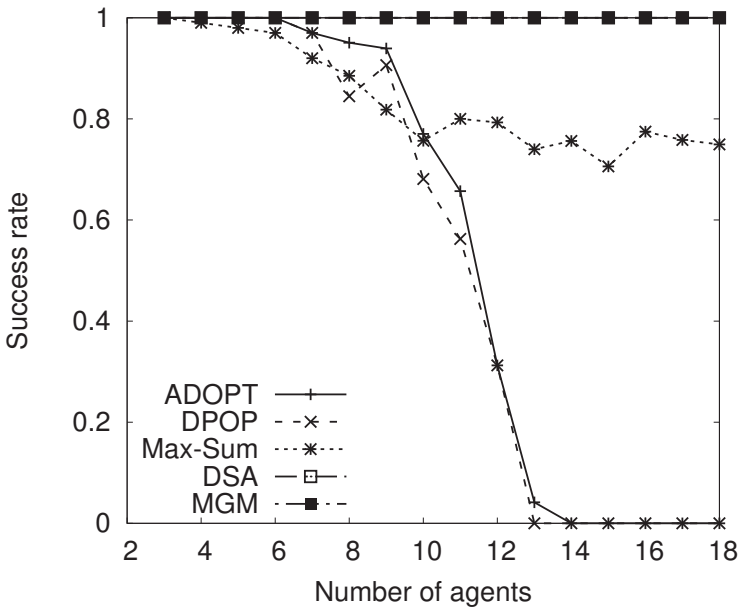

FIGURE 4.2: Average success rate of each algorithm.

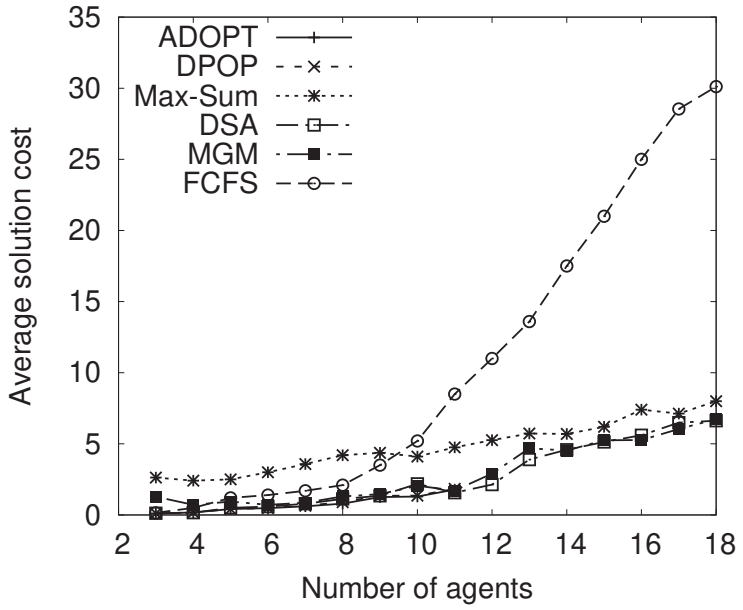

FIGURE 4.3: Average solution quality of each algorithm. The results of ADOPT and DCOP from 12 agents are not shown due to their low success rate.

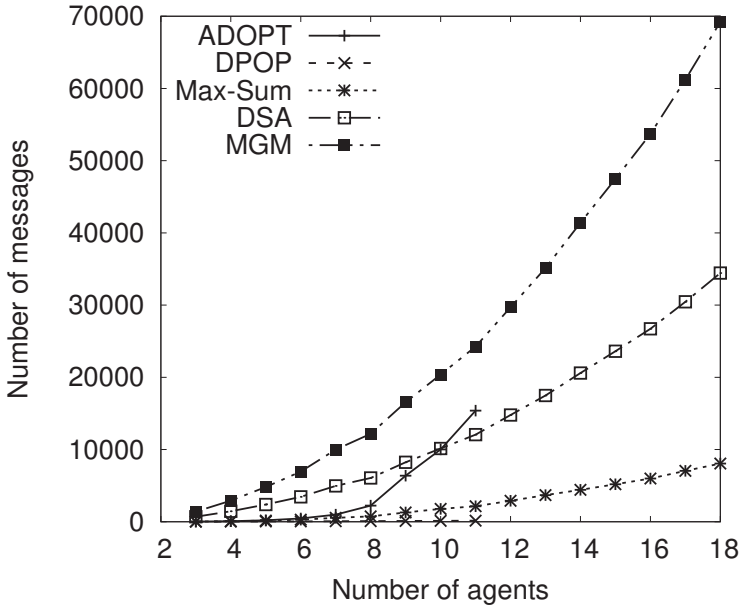

FIGURE 4.4: Average number of messages of each algorithm. The results of ADOPT and DCOP from 12 agents are not shown due to the low success rate.

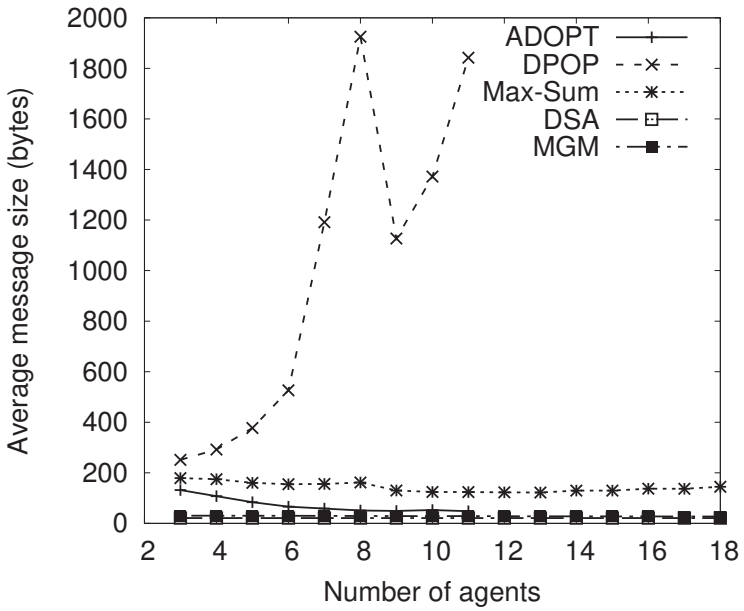

FIGURE 4.5: Average message size of each algorithm. The results of ADOPT and DCOP from 12 agents are not shown due to the low success rate.

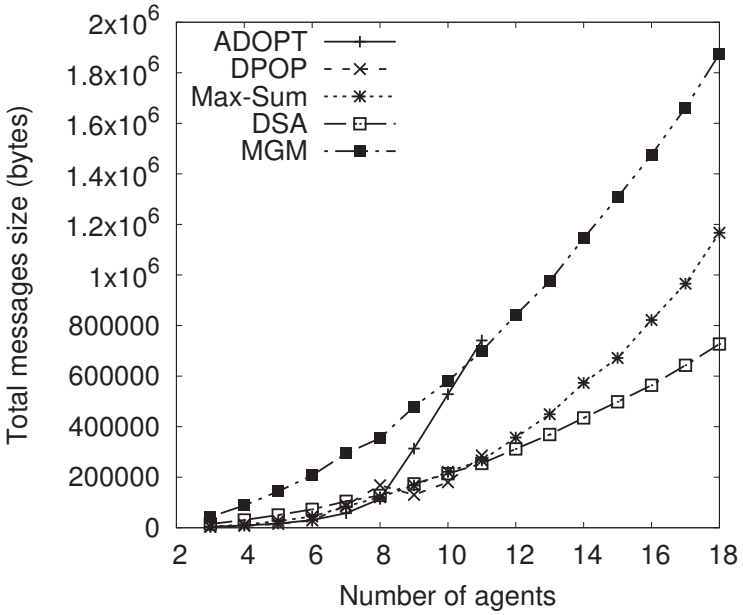

FIGURE 4.6: Total messages size of each algorithm. The results of ADOPT and DCOP from 12 agents are not shown due to the low success rate.

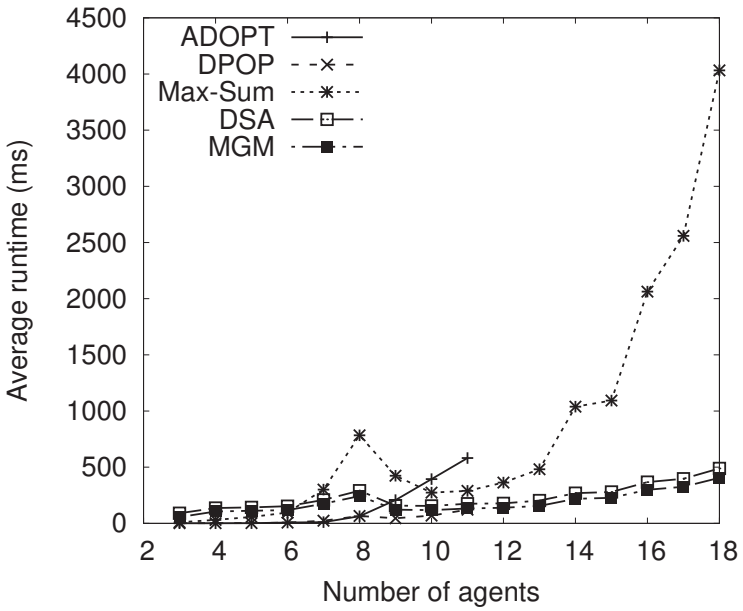

FIGURE 4.7: Average runtime of each algorithm. The results of ADOPT and DCOP from 12 agents are not shown due to the low success rate.

can run more iterations than Max-Sum. This is why they exchange a lot of small messages while Max-Sum agents exchange a lower number of larger messages (cf. Figure 4.4). Figure 4.6 shows that the communication cost of MGM tends to be much higher than the other incomplete algorithms. At this point the computational and communication capabilities of real CAVs, as well as the robustness of the message passing system are not known yet. Once these information is available, further studies can look at the incomplete algorithms in order to choose the most suitable one.

Figure 4.7 shows the average runtime of each algorithm. In a simple scenario, complete algorithms have surprisingly low runtime. However, the runtime grows exponentially with the number of agents, and regularly timed out from 12 agents. As we ran the algorithms until convergence or timed out, the runtime of Max-Sum also grows quickly because the algorithm failed to converge and is at about 4 seconds when reaching 18 agents. MGM and DSA always converge in a fairly low amount of time.

Based on this evaluation, we can notice that only incomplete algorithms can be used for our traffic regulation model. The solution quality achieved by DSA and MGM seems to be the best. Max-Sum did not provide the best solution quality as we use the standard version, but opting for this algorithm has several advantages. The algorithm has a large number of variants that can be used to improve solution quality, convergence rate or preserve privacy. Furthermore, the algorithm is shown to be robust to message loss. However, there are several issues that we have to address when opting for this algorithm. When the number of agents is low, Max-Sum fails to escape from local minimum and provides bad quality solutions (even FCFS solutions are close to the optimal one). Furthermore, the runtime of Max-Sum tends to grow rapidly with the number of agents. In the next sections, we first discuss the continuity problem and the solution we propose. After that, we propose some improvements to the Max-Sum algorithm to address the issues presented above.

4.3 Continuity of the Solution

Now that we have formalised the problem as a DCOP, it is also necessary to discuss the continuity of the solution to deal with the continuous flow of vehicles. Since vehicles continuously approach the intersection, at each time step, we must define the vehicles that take part in the DCOP, the vehicles for which the DCOP will provide an admission time, and the conditions under which an admission time of a vehicle can be revised.

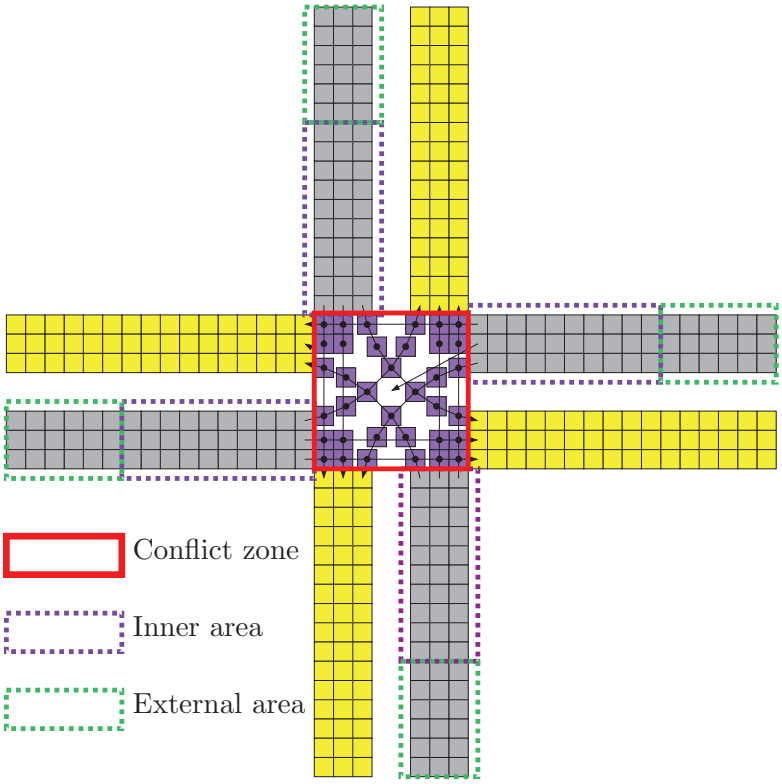

FIGURE 4.8: Inner and external areas of the intersection.

We propose several policies to manage the continuity problem. First, we distinguish two areas on the approaches of the intersection: the inner area, where all the vehicles are about to reach the conflict zone in a short term, and the external area, where the vehicles will reach the conflict zone in a slightly longer term (cf. Figure 4.8). The size of each area depends on the intersection. At each time step, the set V_t of the incoming vehicles is divided into two subsets: V_t^{in} the vehicles inside the inner area and V_t^{ext} the vehicles in the external area. Thus we have:

$$V_t = V_t^{in} \cup V_t^{ext}, V_t^{in} \cap V_t^{ext} = \emptyset \tag{4.11}$$

Let V_t^{par} be the subset of vehicles participating in DCOP at the current time step, i.e. vehicles whose admission time can be revised. The intersection can choose to apply several policies as follows:

Iterated Policy (IP) Each vehicle in V_t^{in} participates once and only once in finding the solution. Once an admission time is chosen, it cannot be changed in the next time steps. Thus we have:

$$V_t^{par} = V_t^{in} \backslash V_{t-1}^{in} \tag{4.12}$$

This policy continues to iterate and to produce new admission times for the next vehicles in the inner area without revising those of the vehicles that already were in it.

Continuous Policy (CP) All vehicles in V_t^{in} participate in the DCOP and the admission time of every vehicle can be revised at any time step. Thus $V^{par} = V^{in}$. For safety reasons, we also note that it is risky to change the admission time of a vehicle at the last moment because of the delay in the reaction of the drivers. To avoid this, we define a safety threshold t_{low}. An admission time lower than t_{low} cannot be modified. Let V^{low} be the set of vehicles v_i having $\varphi_i - t \leq t_{low}$, we have:

$$V^{par} = V^{in} \backslash V^{low} \tag{4.13}$$

Compared to the CP, the IP has fewer vehicles whose admission time will be assigned or modified. This leads to a lower number of agents to take part in the DCOP algorithm, reducing its computational and communication complexity. In addition, CP revises the admission time of all the vehicles, which results in a larger search space. Therefore, we expect a better quality of the solution provided using the CP (as we show later in Section 4.5).

4.4 A Max-sum Solution for the Traffic Management Problem

Based on the evaluation above (cf. 4.2.2), we choose to use the Max-Sum algorithm to solve our problem. Despite the fact that our formalisation is compatible with any *complete* or *incomplete* DCOP algorithm, we chose to use max-sum as it is one of the most efficient algorithms in high density traffic and it has been applied in many multi-agent domains (Macarthur et al., 2011; Ramchurn et al., 2010; Stranders et al., 2009). In this section, we present the basic Max-Sum process and propose solutions to overcome the issues that Max-Sum had during the evaluation (low quality results in low density, high runtime in high density).

In more details, max-sum operates on a factor graph: a bipartite, undirected graph, that contains a variable node x_i for each variable, a factor node c_j for each constraint, and an edge connecting a variable node x_i with a factor node c_j if and only if x_i is involved in c_j. Each agent in max-sum takes the role of the variable node which represents its own variable. The function node's role is taken by one of the agents whose variable is involved in the constraint. Figure 4.9 shows the factor graph of the vehicle-based approach of the scenario presented in Figure 4.1.

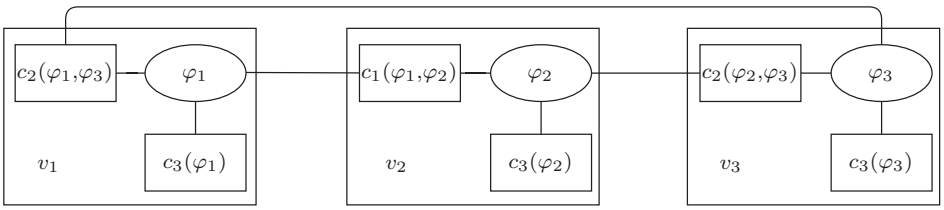

FIGURE 4.9: Vehicle-based factor graph for the scenario presented in Figure 4.1. There are 3 agents (v_1, v_2, v_3), each holds an admission time as a variable node $(\varphi_1, \varphi_2, \varphi_3)$, 1 anteriority factor node $c_1(\varphi_1, \varphi_2)$, 2 conflict factor nodes ($c_2(\varphi_1, \varphi_3)$ and $c_2(\varphi_2, \varphi_3)$), and 3 waiting time factor nodes ($c_3(\varphi_1)$, $c_3(\varphi_2)$, and $c_3(\varphi_3)$).

The main routine of max-sum is the repetition of computing and exchanging messages between variable nodes and factor nodes. At each iteration i of the process, a message is sent from each variable node x to a factor node c, including for each value $d \in \mathcal{D}_x$, the sum of the costs for this value she received from all factor node

neighbours apart from c in iteration $i-1$. Formally, for each value $d \in \mathcal{D}_x$ the message $Q^i_{x\to c}(d)$ is:

$$Q^i_{x\to c}(d) = \sum_{c' \in C_x \setminus c} cost(f'.d) - \alpha \tag{4.14}$$

C_x is the set of factor neighbours of variable x and $cost(c'.d)$ is the cost for value d included in the message received from c' in iteration $i-1$. α represents a scalar to prevent the message to increase endlessly in cyclic factor graphs.

To search for minimisation, the message sent from a factor node c to a variable node x contains for each possible value $d \in \mathcal{D}_x$ the minimum cost that can be achieved from any combination of other variables involved in c. Formally, for each value $d \in \mathcal{D}_x$, the message $R^i_{f\to x}(d)$ is:

$$R^i_{f\to x}(d) = min_{PA_{-x}} cost(\langle x, d\rangle, PA_{-x}) \tag{4.15}$$

In this message PA_{-x} is a possible combination of assignments to all variables involved in c except x. The cost of an assignment $a = (\langle x, d\rangle, PA_{-x})$ is $c(a) + \sum_{x' \in X_f \setminus x} cost(x'.d')$. $c(a)$ is the original cost in the constraint c for the assignment a and $cost(x', d')$ is the cost which was received from the variable node x' during iteration $i-1$, for the value d' which is assigned to x' in a.

Example 4.2. *To give an example of the messages sent, consider the factor graph presented in Figure 4.9. Let $\mathcal{D}_{\varphi_1} = \{5, 6, 7\}$, $\mathcal{D}_{\varphi_2} = \{7, 8\}$. The message that the variable node φ_1 sends to the factor $c_1(\varphi_1, \varphi_2)$ at iteration i for the value $d = 5$ is the following:*

$$Q^i_{\varphi_1\to c_1(\varphi_1,\varphi_2)}(5) = R^{i-1}_{c_2(\varphi_1,\varphi_3)\to\varphi_1}(5) + R^{i-1}_{c_3(\varphi_1)\to\varphi_1}(5) \tag{4.16}$$

The message sent from the factor node $c_1(\varphi_1, \varphi_2)$ to the variable node φ_1 at iteration i is the following:

$$R^i_{c_1(\varphi_1,\varphi_2)\to\varphi_1}(5) = min(cost(\{5, 7\}), cost(\{5, 8\})) \tag{4.17}$$

where:

$$cost(\{5,k\}) = c_1(\{5,k\}) + Q^{i-1}_{\varphi_2 \to c_1(\varphi_1,\varphi_2)}(k) \tag{4.18}$$

During the propagation of messages, an agent is able to calculate locally its admission time that minimises the sum of the costs over all neighbour functions. Standard max-sum often terminates after the solution converges, or after a fixed number of iterations per agent. We have to note that the factor graph of the problem is not cycle free. Therefore, there is no guarantee of convergence with max-sum but extensive empirical evidence demonstrates that the algorithm generates good approximate solutions (Kschischang et al., 2001). In our model, the time complexity is also an issue because a solution that is found after the end of the time step is not useful. Thus, we have to optimise the algorithm to reduce computation.

We observe that, the convergence time of Max-Sum increases quickly to the number of agents. When evaluating Max-Sum using the Vehicle-based approach (cf. Section 4.2.2), we notice that the runtime in high density traffic is costly compared to MGM and DSA. To overcome the issue, we next propose the Lane-based approach, a partially centralised approach that is compatible with the structure of our model.

4.4.1 Lane-based Approach

Instead of considering each vehicle as an agent, we can consider each incoming lane as an agent. The lane agents can either be a part of the traffic control system, or be one of the vehicles in the lane that has the highest computational capability. We consider that there is an agent per incoming lane that has the knowledge of all vehicles in it. As a lane agent, it holds an array variable ϕ_l that contains the admission time of every vehicle in the lane l. By having the knowledge on all these vehicles, the lane agent can build its own domain, respecting both distance constraints and anteriority constraints. These are defined as follows:

Conflict constraint

$$c_2(\phi_i, \phi_j) = \begin{cases} \infty & \text{if } \exists \varphi_k \in \phi_i, \exists \varphi_m \in \phi_j, \\ & \exists e \in \tau_k, e \in \tau_m \text{ and} \\ & |(\varphi_k + \frac{pos(e,\tau_k)}{s_k}) - (\varphi_m + \\ & \frac{pos(e,\tau_m)}{s_m})| \leq t_{safe} \\ 0 & \text{otherwise} \end{cases} \tag{4.19}$$

Waiting constraint

$$c_3(\phi_i) = \sum_{\varphi_j \in \phi_i} \varphi_j - (t + \frac{n_j + 1}{s_j}) \tag{4.20}$$

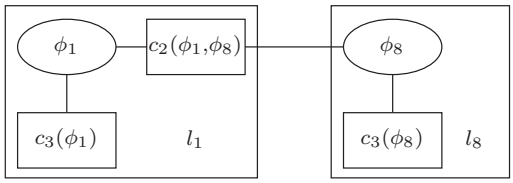

FIGURE 4.10: Lane-based factor graph for the scenario presented in Figure 4.1. Only two lanes have vehicles approaching the intersection so there are only two agents (l_1 and l_8). Each agent holds an array variable node, ϕ_1 contains φ_1 and φ_2, ϕ_8 contains φ_3. There are 2 waiting time factor nodes ($c_3(\phi_1)$ and $c_3(\phi_8)$), and 1 conflict factor node ($c_2(\phi_1, \phi_8)$).

Figure 4.10 shows the factor graph corresponding to the lane-based approach of the scenario presented in Figure 4.1. Clearly, the lane-based approach has a lower number of variables and factors compared to the vehicle-based approach. The lane-based approach considers, as an agent, a lane which contains at least 1 vehicle, thus the worst-case number of agents is the maximum number of incoming lanes ($\mathcal{O}(|L|)$), while the number of agents in the vehicle-based approach grows with the number of vehicles ($\mathcal{O}(|V|)$). The number of factors is also reduced from $\mathcal{O}(\frac{|V|\times(|V|-1)}{2})$ for the vehicle-based approach to $\mathcal{O}(\frac{|L|\times(|L|-1)}{2})$ for the lane-based approach. This reduction leads to a smaller number of messages, in exchange for a growth in the average size of messages due to a larger domain. For the vehicle-based approach the domains grow at $\mathcal{O}(p)$, while for the lane-based approach the domains grow at $\mathcal{O}(p^k)$ where k is the number of vehicles presented in the lane.

Since the lane-based approach may lead to tighter search space and less cycles in the factor graph, which increase the chance for the algorithm to converge, we expect a better performance using this approach.

For safety reasons, we need to ensure each vehicle is assigned an admission time before entering the intersection. However, the DCOP solver may not provide a solution in time. In the next section, we detail the role of the intersection agent which guarantees configurations.

4.4.2 Guaranteeing Safe Configurations

As the traffic conditions change dynamically, we have to ensure that every vehicle that enters the inner area at time step t is assigned an admission time before time step $t+1$. To deal with this problem, we design an intersection agent. This agent has two roles: to hold the current configuration so that the vehicles are synchronised every time there is a change, and to assign to each vehicle that enters the intersection at the beginning of the time step a precalculated admission time. This admission time can be calculated easily by giving to each vehicle (in a random order) the earliest possible admission time, respecting all the other vehicles' admission times, including those whose admission times were just assigned. Despite not being the optimal solution for the system, this solution has two advantages:

- it helps ensure that no vehicle in the inner area is found without an admission time at any time step, even if the DCOP solver fails to terminate in time.
- it gives the DCOP algorithm an upper bound UB (i.e. the total waiting time of the vehicles on the precalculated solution) to run a pruning algorithm as a preprocessing step.

Example 4.3. *Consider the scenario presented in Figure 4.1. Let $t = 0$. At the time step $t-1$, consider having only v_1 in the inner area. The configuration of $t-1$ is $\{\varphi_1 = 5\}$. At the beginning of the current time step, v_2 and v_3 enter the inner area. The precalculated admission times for v_2 and v_3 are: (1) For v_2 the earliest admission time respecting $\{\varphi_1 = 5\}$ is $\varphi_2 = 7$ (2) For v_3 the earliest admission time respecting $\{\varphi_1 = 5, \varphi_2 = 7\}$ is $\varphi_3 = 11$. Therefore, we have $UB = 4$.*

4.4.3 Pruning the Domains

The complexity of max-sum is known to be exponential in the number of agents where the base is the domain size and the exponent is the number of variables involved (Macarthur et al., 2011). Thus, one solution to reduce the calculation time of max-sum is to prune the search space. The pruning technique was implemented using a modified version of the preprocessing method proposed in (Stranders et al., 2009) to reduce the size of the variables' domains by detecting values that are dominated. The values are detected as follows:

1. The intersection agent notifies other agents about UB.
2. The variable nodes calculate the lower bound (LB) of the cost of the value assignment, for each value assignment in their domains.
3. The variable nodes remove dominated values. A dominated value is one whose LB is higher than UB.
4. The variable nodes propagate their updated domains to the factor nodes. The factor nodes recalculate the costs and propagate them further.
5. The steps 2,3,4 are repeated until no more elimination found.

We note that the total cost of the solution cannot exceed UB. As mentioned in Section 4.3, depending on the policy, there are vehicles whose admission time cannot be changed. Let V_t^u be the set of these vehicles. Thus the cost of the admission time for each vehicle in V_t^{par} cannot exceed $UB - \sum_{v_j \in V_t^u} c_3(\varphi_j)$. Thus we have:

$$\forall v_i \in V_t^{par}, \varphi_i - (t + \frac{n_i + 1}{s_i}) \leq UB - \sum_{v_j \in V_t^u} c_3(\varphi_j) \tag{4.21}$$

Therefore, the value of p which is the range of each domain before pruning can be predetermined as:

$$p = UB - \sum_{v_i \in V_t^u} c_3(\varphi_i) \tag{4.22}$$

Example 4.4. *Following the scenario presented in Example 4.3. Let the precalculated configuration for* $\{v_1, v_2, v_3\}$ *be* $\phi = \{5, 7, 11\}$*. Thus we have* $UB = 4$ *and* $p = 4$*, initially we have* $\mathcal{D}_{\varphi_1} = \{5, 6, 7, 8, 9\}, \mathcal{D}_{\varphi_2} = \{7, 8, 9, 10, 11\}, \mathcal{D}_{\varphi_3} = \{7, 8, 9, 10, 11\}$*. After completing the pruning process, we obtain the following pruned domains:* $\mathcal{D}_{\varphi_1} = \{5, 6, 7\}, \mathcal{D}_{\varphi_2} = \{7, 8, 9, 10, 11\}, \mathcal{D}_{\varphi_3} = \{7, 9, 11\}$.

After running max-sum on the vehicle-based scenario presented above, we obtain the following solutions:

- *With IP (the admission time of* v_1 *cannot be revised):* $\Phi_t = \{5, 9, 9\}$*, total waiting time of all vehicles: 4s.*
- *With CP (any admission time can be revised):* $\Phi_t = \{7, 8, 7\}$*, total waiting time of all vehicles: 3s.*

4.5 Empirical Evaluation

In this section, we evaluate the performance of our method using max-sum algorithm. All experiments were performed using an Intel Core i5-4690 3.5 GHz, 8 GB RAM, under Ubuntu 16.04. Max-sum algorithm is implemented using Frodo (Léauté et al., 2009). All compared values are averages over at least 50 simulations, with 95% confidence interval as error bars. All algorithms are evaluated according to the insertion rate of vehicles. The insertion rate varies from 0.1 (off-peak) to 0.5 (rush hour) (Junges and Bazzan, 2008). An insertion rate of 0.5 consists of adding 5 vehicles to a lane every 10 time steps. We ran our experiments with the vehicle-based and lane-based approaches, using both IP and CP.

4.5.1 Benchmarking

First we compare our methods with the state of the art FCFS algorithm (Dresner and Stone, 2008) where each vehicle sends a request for an admission date and the intersection handles these requests using a First come First served policy to test the quality of the solution, the computational time and the number of messages exchanged between agents.

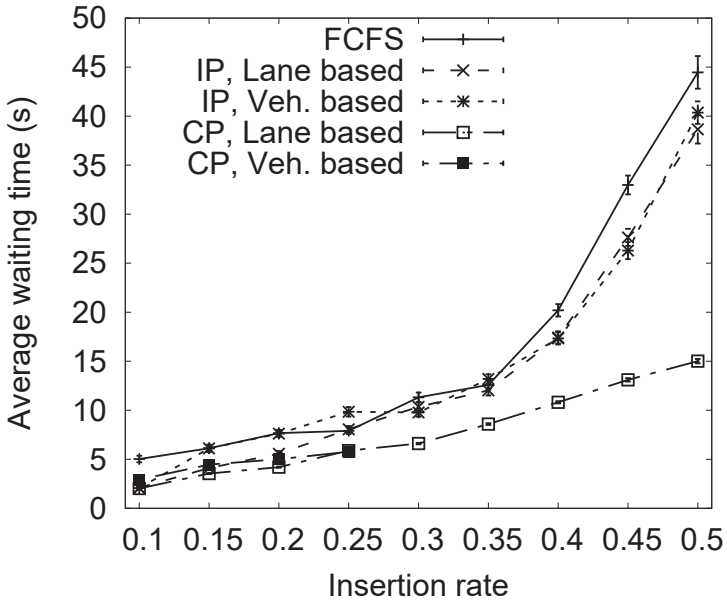

FIGURE 4.11: Average waiting time of vehicles.

In terms of quality of the solution, IP did not provide a significantly better solution compared to the FCFS policy. On the other hand, CP performs better than all the other policies, reducing the waiting time by about 60% in rush hours (cf. Figure 4.11). We also note that IP consumed more resources than FCFS, but less than CP. In rush hours, vehicle-based IP exchanged in average 2200 times more messages, while lane-based IP exchanged about 660 times more messages than FCFS. Lane-based CP used even more resources, exchanging 7880 times more messages than FCFS (cf. Figure 4.12 and 4.13).

To compare the vehicle-based approach and lane-based approach, we note that both provided the same solution quality. The lane-based approach reduced the

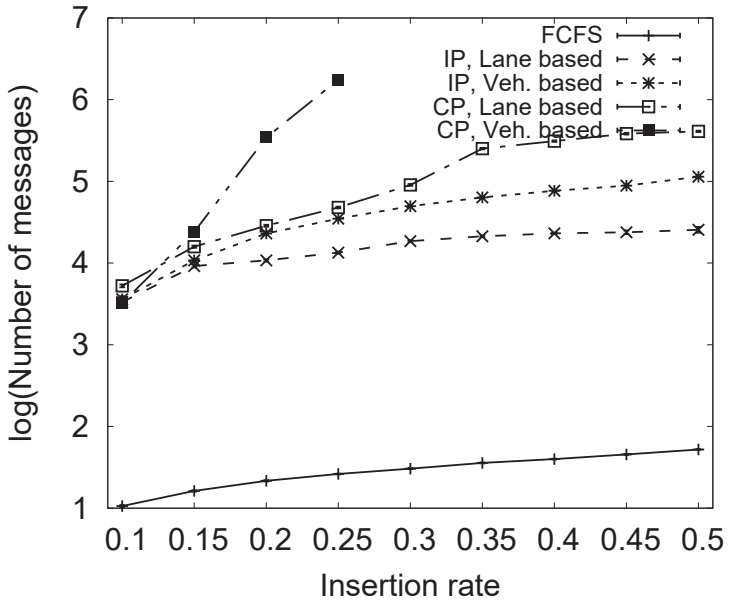

FIGURE 4.12: Average communication overhead at each iteration.

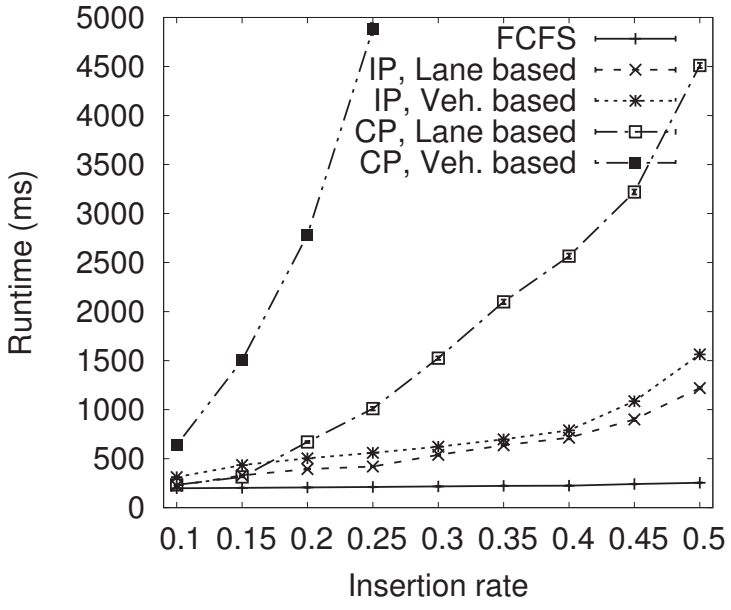

FIGURE 4.13: Average computational time at each iteration.

number of messages and calculation time, as its level of decentralisation is lower. In our experiments, due to the limit in computational capability of our system, we fix the time out of the DCOP solver at 6000 ms. When using the max-sum algorithm, as the convergence time grows drastically with the number of agents, the vehicle-based approach quickly failed to converge to a feasible solution in time, while the lane-based approach using CP continued to generate solutions at the insertion rate of 0.5.

4.5.2 Pruning Efficiency

To measure the performance of the pruning algorithm, Figures 4.14 and 4.15 show the difference in number of messages exchanged and calculation time (in milliseconds) between the pruned and unpruned versions of the lane-based approach using CP. For the unpruned version, we just fix

$$p = UB - \sum_{v_i \in V_t^u} c_3(\varphi_i) \tag{4.23}$$

at every time step. We note that the pruned algorithm reduces about 25% - 30% of the messages exchanged and calculation time because convergence was achieved in less iterations.

4.5.3 Dynamic Events

The DCOP formalisation of microscopic traffic regulation is also adaptable to dynamic events. We have done 20 simulations and got the average results to compare how the CP, lane-based approach and FCFS react on the arrival of emergency vehicles. We simulate the traffic over 500 time steps, with an emergency vehicle added to a random lane on time step 200. The emergency vehicle is defined in the system as a vehicle with an extremely high cost per second of waiting time. This forces the DCOP solver to look for a solution which minimises the waiting time of the emergency vehicle. This solution often leads to the immediate evacuation of the vehicles in front of the emergency vehicle in its lane. We observe that

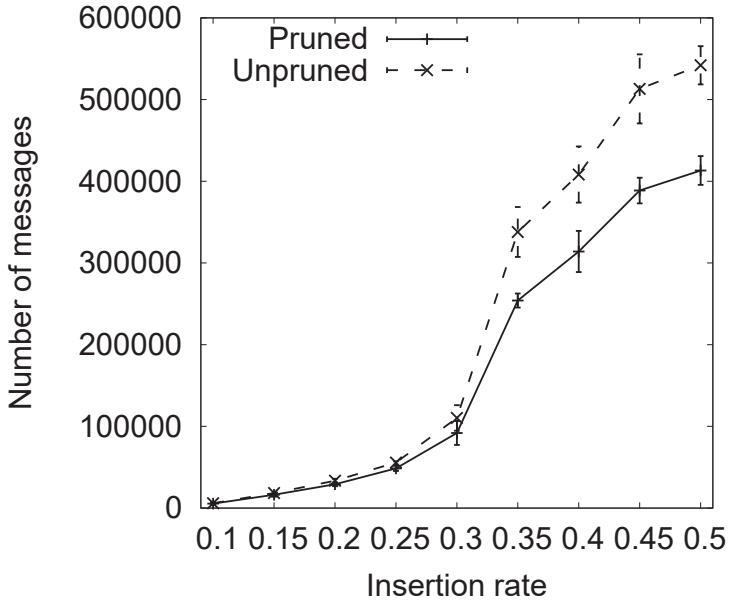

FIGURE 4.14: Average communication overhead comparison between the pruned and the standard version.

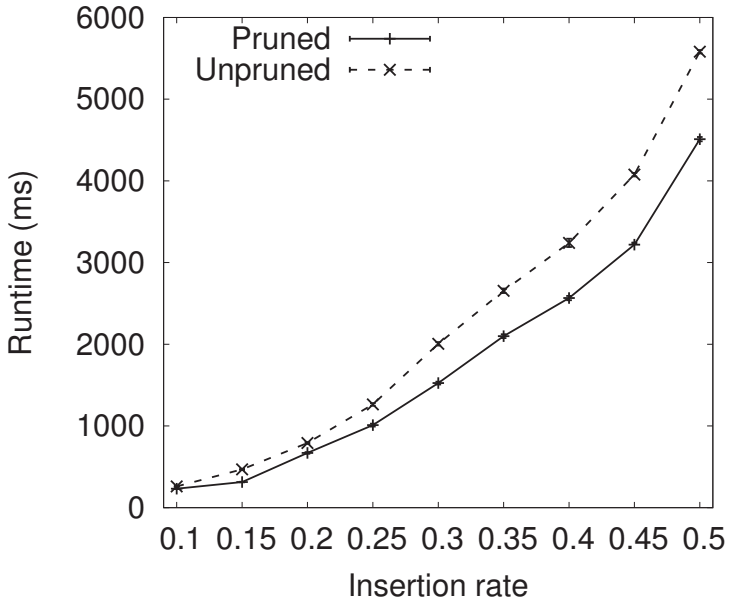

FIGURE 4.15: Average computational time comparison between the pruned and the standard version.

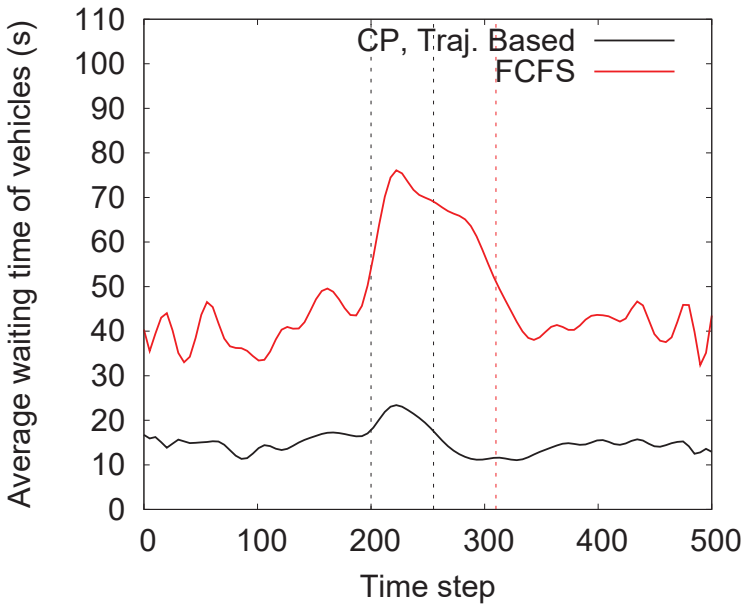

FIGURE 4.16: Average waiting time of vehicles at each time step. The emergency vehicle arrives at $t = 200$. The CP stabilises at $t \approx 250$ while FCFS stabilises at $t \approx 310$.

the arrival of the emergency vehicle leads to a high average waiting time on the other vehicles. FCFS succeeds in giving the emergency vehicle a reasonably low waiting time (2.7 seconds) by prioritising the emergency vehicle's lane, but this policy takes on average 110 time steps to evacuate the other lanes to return to a stable state. The DCOP approach stabilises after 50 time steps (about half the amount of time compared to FCFS) and returns to the normal condition, giving the emergency vehicle a waiting time of 2.4 seconds.

4.5.4 A Note on the Lane-Based Approach

As shown in the empirical results, the lane-based approach can sometimes be outperformed by the vehicle-based approach, especially in the lower density settings. However, there is also another aspect we should take a closer look at, namely the communication range of the vehicles. Communication between vehicles can be achieved via the infrastructure installed at the intersection level. However,

there are always areas where intersections may not have any computation capability (e.g., in rural areas or non-urban settings). This can be important in mass evacuations following fires or floods (Ramchurn et al., 2010).

Figures 4.17 and 4.18 show the communication required for the vehicle-based and the lane-based approaches. We can see that the lane-based approach also helps reducing communication range, because only the vehicle representing the lane is required to communicate with vehicles representing conflicting lanes. Hence, in the lane-based approach, the lane agent should be the first vehicle in the lane due to the reduction of communication range.

In an intersection such as ours, even the lane-based approach will have 12 agents negotiating in dense traffic (i.e. one for each lane). As we have shown in the Section 4.2.2, complete algorithms have a very low success rate with 12 vehicle agents. It is then unlikely that this type of algorithm can be used for such intersection. However, in less complex scenario (e.g. intersections with 4 lanes), they can be opted to produce the true optimal solution.

4.6 Summary

In this chapter we have modelled the traffic management problem at an intersection using constraints (cf. Section 4.1). We then provided in Section 4.2 a DCOP formalisation of the problem. In this section, we also evaluated the performance of two notable complete algorithms (ADOPT, DPOP) and three incomplete algorithms (Max-Sum, DSA, MGM). Through the experiments, we used different factors to identify the strengths and weaknesses of each algorithm. We then highlighted the different issues that we have to address with our chosen algorithm. Section 4.4 shown in more details the use of the Max-Sum algorithm and discussed how we overcome the issues identified. Finally, Section 4.5 showed the performance of the complete mechanism to manage traffic at a single intersection. The mechanism is empirically shown to outperform the state of the art solution in terms of reductions in waiting time and robustness to dynamic events.

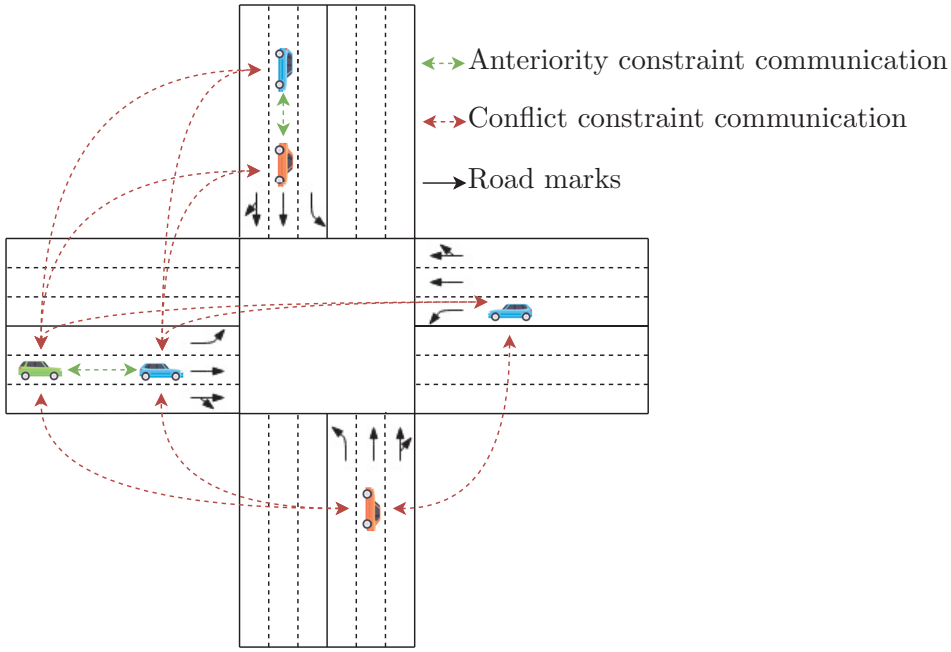

FIGURE 4.17: Communication range required for vehicle-based approach. Two vehicles need to be able to communicate with each other if they are constrained (i.e. either they are on the same lane, or they have conflicting trajectories). Therefore, the green vehicle on the western side has to communicate with the blue one on the north side. Such problem leads to higher communication range required.

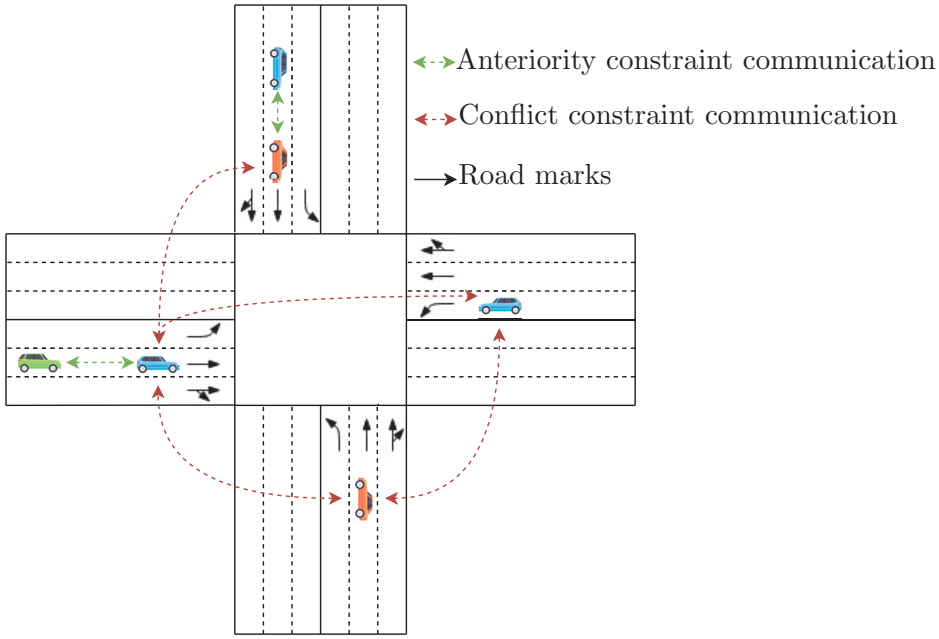

FIGURE 4.18: Communication range required for lane-based approach. Each lane is represented by a vehicle. To reduce communication range, the lane should be represented by the first vehicle on it. The communication is required between lane agents to solve conflicting trajectories. Other vehicles only need to communicate with their lane agent. The range required is therefore lower.

While this chapter has shown the potential of DCOPs to solve traffic management problems, it can be further extended in many ways. First, cellular model keeps the presentation simple, but seems to result in a loss of space when dealing with vehicles with different sizes. Furthermore, not taking into account multi-intersection setup can lead to the solution being worse at a larger scale. In the next chapter, we extend this model in several ways. First, we propose a space-efficient model of the intersection, which is more space-efficient compared to the cellular model. We then propose an individual priority level which can be used to guide traffic based on global information. Finally, we evaluate these propositions using a recent variant of the Max-Sum algorithm, with some improvements in regard to the structure of our problem.

Chapter 5

The Space-Efficient Model, the Use of Max-Sum_AD_VP and the Multi-Intersection Problem

In this chapter, we extend the model proposed in the previous one (cf. Chapter 4). We first discuss the limits of the cellular model used widely in traffic regulation research. We overcome these limits by proposing a precise model, where vehicles use their exact position and velocity to compute their possible conflicts. After that, we propose a multi-intersection approach where global information can be used to guide DCOP algorithm, in order to balance traffic between different areas. We then opt for a recent variant of the Max-Sum algorithm, namely the Max-Sum_AD_VP algorithm, and propose a better way to order nodes when computing. Finally, we evaluate each of our proposition and discuss further possible studies.

5.1 Limits of the Cellular Model

When opting for intelligent intersection management, one crucial step is to model the intersection area and define rules for vehicles crossing this area. In most works proposed earlier in multi-agent systems, cellular-based presentations are often the

authors' choice (Dresner and Stone, 2008; Vasirani and Ossowski, 2009), or the one in Chapter 4. However, using cellular-based model might lead to a higher use of space than necessary (e.g. in the model proposed by (Dresner and Stone, 2008), the area that a vehicle reserved is always higher than its exact length and width), or a lack of precision (e.g. in the model proposed in the previous chapter, each vehicle is counted as one cell, regardless of their length). Figure 5.1 shows situations where a car can occupy more than one cell and thus, leads to an inefficient way of using space. Furthermore, using the space-efficient model makes the mechanism compatible with different types of intersections, for example, the ones with an uneven number of lanes in each direction.

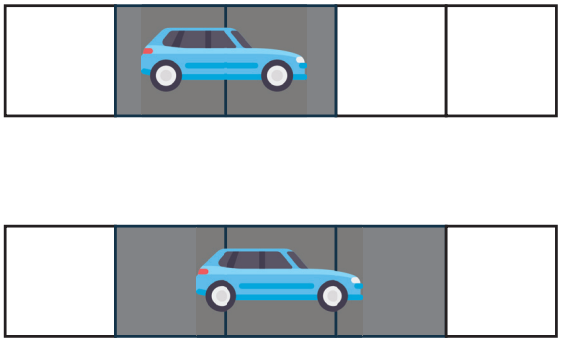

FIGURE 5.1: A car with a longer size can occupy two, or even three cells at a time, depending on its position.

In every existing intersection model, rules for vehicles are the same. They aim to give each vehicle a reservation (Dresner and Stone, 2008; Vasirani and Ossowski, 2009), which is a set of cells for each time step or an admission time, which is the time that the vehicle can enter the intersection. The rules for reservations or admission times to be accepted is that vehicles can cross the intersection without stopping and without any conflict between them. Conflicts are often detected if vehicles try to use the same cell at the same time.

In reality, depending on the infrastructure installed at the intersection level, vehicles might be able to know their exact position. They also have information about their velocity and their length. Thus, instead of using a cellular model and checking for conflict between vehicles using their reserved cells, they might be able to

overcome the limits of the cellular model by applying the exact formula computed based on this information. In the next section, we describe a novel space-efficient model and redefine the constraints.

5.2 A Space-Efficient Intersection Model

As mentioned above, vehicles are regularly not precisely represented in a cellular model and can occupy more space than necessary. This may lead to an inefficient way of using space and thus can reduce the performance of the model. In this section, we will present a precise way to model trajectories of vehicles to avoid conflict, while being more space-efficient.

Definition 5.1. Let t be the current time step and V_t the set of all vehicles approaching the intersection. Each vehicle $v_i \in V_t$ is modelled with: its relative distance to the intersection d_i, its velocity s_i and its length ℓ_i.

Definition 5.2. An intersection is modelled with several *incoming lanes*, several *outgoing lanes*, and a central zone called *conflict zone*. The path of a vehicle across the intersection is called a *trajectory*. The shared area between two trajectories is called a *conflict spot* (cf. Figure 5.2).

5.2.1 Structural Constraints

We model each intersection using a DCOP model described in Chapter 4. This model aims to find, for each time step t, a configuration Φ_t, which consists of one admission time for each vehicle. Vehicles are able to cross the intersection at a constant speed at their admission time. The conflict-free property is guaranteed. As mentioned earlier, we extend the existing model by using the exact information about vehicle location, velocity and length. Thus, the rules in that model can be rewritten as follows:

Let L be the set of incoming lanes and $l_k \in L$ be lane k. For each $v_i \in V_t$, let $l_{v_i} \in L$ be the lane in which the vehicle v_i is present and τ_i be v_i's trajectory inside

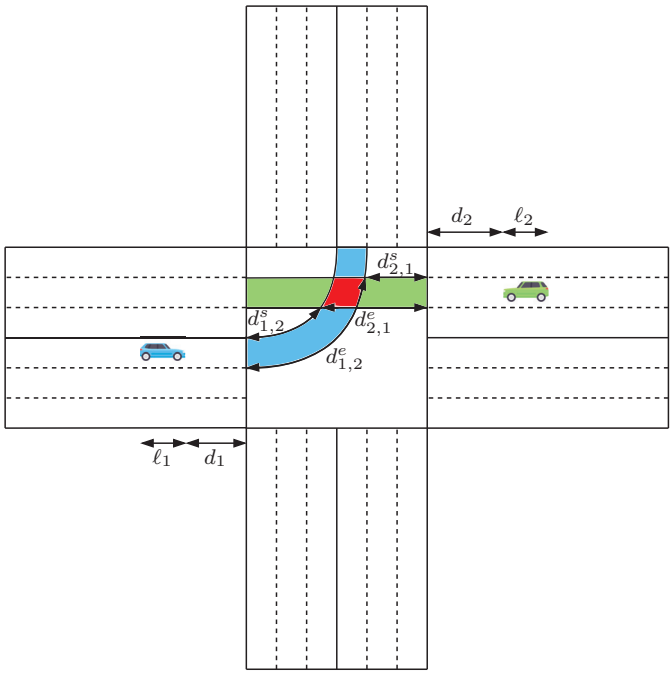

FIGURE 5.2: Intersection with 12 incoming lanes, 12 outgoing lanes and a conflict zone. *Incoming lanes* are numbered from 1 to 12. The conflict zone is crossed by various trajectories. There are 3 vehicles v_1 (light blue), v_2 (green) and v_3 (orange). The trajectories τ_1 of v_1 and τ_3 of v_3 are the same and are coloured in light blue, and τ_2 of v_2 in green. The conflict spot between the two trajectories is coloured in red.

the conflict zone. Let $d^s_{i,j}$ the distance between the beginning of τ_i and the starting point of the *conflict spot* between τ_i and τ_j, and $d^e_{i,j}$ the distance between τ_i and the end of this *conflict spot*.

c1. Distance constraint A vehicle has to cross the distance separating it from the conflict zone before entering it:

$$\forall v_i \in V, \varphi_i > t + \frac{d_i}{s_i} \tag{5.1}$$

c2. Anteriority constraint A vehicle v_i cannot enter the conflict zone before the vehicle v_j, preceding it on its lane, completely enters the *conflict zone*. In our model, we consider the area close to an intersection. Therefore, no overtaking is possible. Thus a vehicle v_i cannot enter the conflict zone before the vehicle v_j preceding it on its lane completely enters the *conflict zone*. We

have:

$$\forall v_i, v_j \in V_t^2, l_{v_i} = l_{v_j}, d_i > d_j \Rightarrow \varphi_i > \varphi_j + \frac{\ell_j}{s_j} \tag{5.2}$$

c3. Conflict constraint Two vehicles must not be present at the same time in their conflict spot. Given all the information, if the trajectories of v_i and v_j have a *conflict spot*, v_i has to leave it before v_j arrives or vice-versa [1]. Note that the time it takes for v_i to completely leave the *conflict spot* is the time it travels the distance $d^e_{i,j} + \ell_i$. We have:

$$\forall v_i, v_j \in V_t^2, (\varphi_i + \tfrac{d^s_{i,j}}{s_i}) > (\varphi_j + \tfrac{d^e_{j,i}+\ell_j}{s_j}) \vee (\varphi_j + \tfrac{d^s_{j,i}}{s_j}) > (\varphi_i + \tfrac{d^e_{i,j}+\ell_i}{s_i}) \tag{5.3}$$

c3b. Conflict constraint with safety lapse When adding a safety lapse t_{safe} for the constraint **c3**. We have:

$$\forall v_i, v_j \in V_t^2, (\varphi_i + \tfrac{d^s_{i,j}}{s_i}) > (\varphi_j + \tfrac{d^e_{j,i}+\ell_j}{s_j} + t_{safe}) \vee (\varphi_j + \tfrac{d^s_{j,i}}{s_j}) > (\varphi_i + \tfrac{d^e_{i,j}+\ell_i}{s_i} + t_{safe}) \tag{5.4}$$

For the scenario presented in Figure 5.2, assuming that the width of each lane is 3 meters (i.e. the conflict area of the intersection has a dimension of 18x18 meters), the distance between each trajectory τ_i and the conflict spot between it and a trajectory τ_j (i.e. $d^s_{i,j}$) is shown precisely in Table 5.1. The distance between each trajectory τ_i and the end of the conflict spot between it and a trajectory τ_j (i.e. $d^e_{i,j}$) is presented in Table 5.2.

5.2.2 Objective of Each Intersection and Discussion

The average delay of vehicles has been the common performance indicator of a system at the intersection level (Dresner and Stone, 2008; Vasirani and Ossowski,

[1] This solution aims to work for settings with a large number of CAVs. In transitional periods where non-autonomous vehicles are presented, this constraint can be extended by adding a time lapse between the two vehicles to keep a safe distance.

TABLE 5.1: Distance in meters from a trajectory τ_i to the conflict point between two trajectories τ_i and τ_j. The cell is blank if there is no conflict between them.

τ_i \ τ_j	Lane: 1 Left turn	Lane: 2 Straight	Lane: 3 Straight	Lane: 3 Right turn	Lane: 4 Left turn	Lane: 5 Straight	Lane: 6 Straight	Lane: 6 Right turn	Lane: 7 Left turn	Lane: 8 Straight	Lane: 9 Straight	Lane: 9 Right turn	Lane: 10 Left turn	Lane: 11 Straight	Lane: 12 Straight	Lane: 12 Right turn
Lane: 1 Left turn					0						7.570	11.079		8.429	3.032	0		
Lane: 2 Straight					6.708	12	15		6.381					3	0			
Lane: 3 Straight					8.485	12	15		6					3	0			
Lane: 3 Right turn															0			
Lane: 4 Left turn	8.429	3.032	0						0					7.570	11.079			
Lane: 5 Straight		3	0						6.708	12	15		6.381					
Lane: 6 Straight		3	0						8.485	12	15		6					
Lane: 6 Right turn			0															
Lane: 7 Left turn		7.570	11.079		8.429	3.032	0						0					
Lane: 8 Straight	6.381					3	0						6.708	12	15			
Lane: 9 Straight	6					3	0						8.485	12	15			
Lane: 9 Right turn						0												
Lane: 10 Left turn	0				7.570	11.079			8.429	3.032	0							
Lane: 11 Straight	6.708	12	15		6.381					3	0							
Lane: 12 Straight	8.485	12	15		6					3	0							
Lane: 12 Right turn											0							

TABLE 5.2: Distance in meters from a trajectory τ_i to the end of the conflict point between two trajectories τ_i and τ_j. The cell is blank if there is no conflict between them.

τ_i \ τ_j	Lane: 1 Left turn	Lane: 2 Straight	Lane: 3 Straight	Lane: 3 Right turn	Lane: 4 Left turn	Lane: 5 Straight	Lane: 6 Straight	Lane: 6 Right turn	Lane: 7 Left turn	Lane: 8 Straight	Lane: 9 Straight	Lane: 9 Right turn	Lane: 10 Left turn	Lane: 11 Straight	Lane: 12 Straight	Lane: 12 Right turn
Lane: 1 Left turn					8.672					15.817	14.137		14.137	6.567	3.058	
Lane: 2 Straight					11.619	15	18		11.292					6	3	
Lane: 3 Straight					12	15	18		9.515					6	3	
Lane: 3 Right turn															4.712	
Lane: 4 Left turn	14.137	6.567	3.058						8.672					15.817	14.137	
Lane: 5 Straight		6	3						11.619	15	18		11.292			
Lane: 6 Straight		6	3						12	15	18		9.515			
Lane: 6 Right turn			4.712													
Lane: 7 Left turn		15.817	14.137		14.137	6.567	3.058						8.672			
Lane: 8 Straight	11.292					6	3						11.619	15	18	
Lane: 9 Straight	9.515					6	3						12	15	18	
Lane: 9 Right turn							4.712									
Lane: 10 Left turn	8.672					15.817	14.137		14.137	6.567	3.058					
Lane: 11 Straight	11.619	15	18		11.292					6	3					
Lane: 12 Straight	12	15	18		9.515					6	3					
Lane: 12 Right turn											4.712					

2009). Let w_i be the waiting time of v_i, this minimisation can be described as finding the minimum value for $\sum_{v_i \in V_t} w_i$. However, from a network point-of-view, simply evacuating vehicles in front of the intersection as quickly as possible can create high density traffic in the *outgoing lanes*. Indeed, several studies (Lighthill and Whitham, 1955a,b; van Rijn, 2014) have shown that traffic flow speed in a lane is not linear in the lane's density, but rather follows complex rules. Hence, in a road network, continuing to send vehicles to a lane that has a high density may result in a significant slowdown. In the market-based regulation system (Vasirani and Ossowski, 2009), authors have introduced a dynamic pricing policy to improve the performance of the network. Building upon that policy, we will next introduce a priority setting technique that can be used to regulate traffic in a multi-intersection settings.

5.3 Priority Levels for Multi-Intersection Settings

To date, intersection management algorithms have mainly been shown to optimise traffic flow for individual intersections. However, they do not acknowledge the fact that it is not always possible to evacuate vehicles through the outgoing lanes as they might be the neighbouring intersections' incoming lane and thus might have a long queue. This leads to the fact that optimising traffic at an intersection might lead to further conflict at another intersection. In this section, we present a novel dynamic individual priority level, which can be used to distribute vehicles among intersections, or even guide vehicles to a better trajectory.

Similar to a dynamic pricing problem (Faruqui and Sergici, 2010; Do Chung et al., 2012) where resources might have different costs each time, a vehicle's delay should be continuously evaluated using several criteria. Formally, we define for each vehicle v_i a strictly positive real value priority level ρ_i. This priority level is updated using traffic information such as the vehicle's past trajectories, the state of its destination and the nature of the vehicle. A priority is defined by its type (e.g., emergency vehicles, buses, road maintenance vehicle) whilst a dynamic factor

is added using the other information (e.g., trajectory, destination, delay). Next we propose two ways to update vehicles' priority level, namely the *Priority by history* and the *Priority by destination*.

5.3.1 Calculating Priority Levels

The priority of a vehicle represents the contribution of its delay in the solution (i.e. a vehicle with higher priority contributes more to the quality of the solution). Therefore, dynamically updating this priority can guide the mechanism to different solutions as time progresses. In this chapter, we propose two ways to calculate and update a vehicle's priority based on its information and on the global traffic conditions.

Priority by history: *Priority by history* is computed based on the total delay of a vehicle from the beginning of its trajectory. We assume that every ordinary vehicle that enters the road network has the same priority level and each intersection will try to minimise average delays, the method would favour the more crowded lanes. This makes vehicles that travel in a less crowded trajectory wait for an extremely long time in dense traffic. To be able to balance a vehicle's waiting time and the global objective of the intersection, we dynamically change vehicles' *priority by history*. In this chapter, we consider the distribution of *priority by history*, ranging from 0 for vehicles that recently entered the system to 10 for vehicles that are suffering lots of delay (cf. Figure 5.3). In certain cases, this priority can also help evacuating vehicles from a congested area as they tend to have higher delays and thus, higher priority than others.

Priority by destination: *Priority by destination* is computed based on the density of the next destination of a vehicle to avoid sending vehicles to a congested area. In a simple intersection model, it is often assumed that the *outgoing lanes* are always free and capable of taking vehicles. However, this assumption breaks down in real-world settings as the conditions at the neighbouring intersections will determine how fast cars can move along. For example, if an intersection cannot

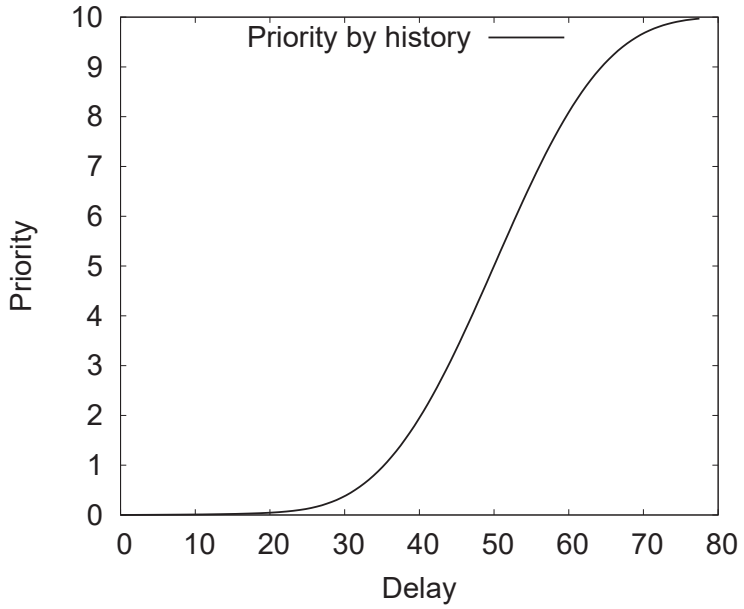

FIGURE 5.3: Priority by history. A vehicle with higher delay in since the beginning of its journey should be more prioritised over others.

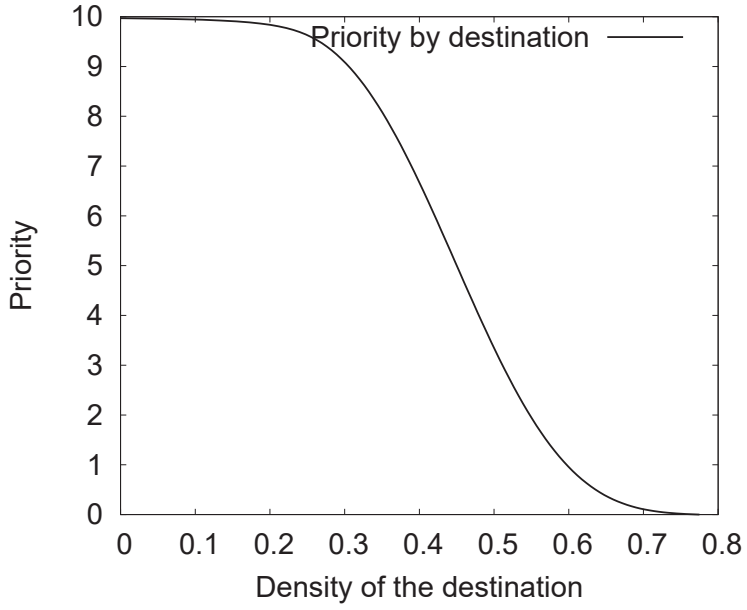

FIGURE 5.4: Priority by destination. A vehicle willing to enter a more congested area should be less prioritised than others.

shift vehicles from one of its *incoming lanes*, a neighbouring intersection cannot and should not shift more vehicles to this lane. Such situation can also create a deadlock if the first CAV in the lane has to stop because its destination doesn't have enough free space. Hence, redistributing priority so that an intersection can avoid sending vehicles to a more congested intersection can also be useful. Furthermore, giving a priority bonus to a certain direction also encourages vehicles to take a less congested route when they have multiple options to complete their journey. The bonus *priority by destination* is also distributed from 0 to 10, according to the density of a vehicle's destination communicated by the neighbouring intersection (cf. Figure 5.4). Furthermore, intersections can exchange information with neighbouring intersections in case of blocked lanes due to unpredictable events so that traffic flows can be eased.

5.3.1.1 Optimising Weighted Delay

Since each vehicle has its priority level, we will build, for each time step t, a configuration Φ_t for all vehicles in V_t in front of the intersection that minimises their total weighted delay whilst being able to satisfy all the structural constraints described above. The input is the set of vehicles V_t presented in front of the intersection at the current time step and the configuration at the last time step Φ_{t-1}. Let w_i be the waiting time of vehicle v_i (i.e. the difference between the admission time of v_i in fluid condition and its actual admission time) and Φ be the set of all possible configurations, our goal can be expressed as follows:

$$f : (t, V_t, \Phi_{t-1}) \mapsto \arg\min_{\Phi_t \in \Phi} \sum_{v_i \in V_t} w_i * \rho_i \tag{5.5}$$

We next discuss the formalisation of the model using a DCOP and show how existing DCOP solution algorithms can be optimised to consider the parameters of our problem.

5.4 DCOPs for Intersection Management

We have presented two possible ways to formalise the intersection management problem as a DCOP, depending on what we choose to represent with agents, variables and constraints. The choices have an impact in both computational load and communication overhead of agents. In this chapter we continue to evaluate our model using two formalisations, namely the vehicle-based approach where each vehicle is considered as an agent and the lane-based approach where the sub-problem of the lane is solved before the global optimisation problem.

As a reminder, in the vehicle-based approach, each vehicle participate in the DCOP formulation as an agent. They each have one variable representing their admission time. The vehicles then perform a fully decentralised process in order to find a global solution that does not cause any conflict, and that optimises the overall delay.

On the other hand, in the lane-based approach, each lane is represented by an agent. This solution sacrifices some decentralisation in exchange for less computational time. Vehicles in the same lane are affected with the anteriority constraint, and may often cross the intersection using the same trajectory. Thus, the lane agent can solve the sub-problem and exchanges solutions that do not violate the anteriority constraint. The lane agent uses a pseudo-variable which is the Cartesian product of the admission times of all the vehicles in the lane.

The lane-based approach has been shown to outperform the vehicle-based approach in standard Max-sum setting (cf. Chapter 4). However, when switching to a recent variant of the algorithm, the Max-sum_AD_VP (Zivan and Peled, 2012), the success rate becomes higher and thus we reevaluate their performances and notice that each approach is preferred in different traffic densities.

In the space-efficient model, the agents and variables are the same as the cellular model. The changes in the DCOP formalisation are the size of the domains and the equations that describe the constraints. As we know, Max-Sum has a linear complexity to the domain size of variables. Hence we can continue to use the Max-Sum family algorithms to exploit the two models presented above. In the next

step, we give a discussion on another variant of the Max-Sum algorithm, named the Max-Sum_AD_VP (Zivan and Peled, 2012), on its difference and performance compared to the standard version.

5.4.1 The Max-sum_AD_VP Algorithm and the Importance of Node Ordering

Max-sum_AD_VP is a recent variant of Max-sum and is empirically proven to converge faster and to a better solution than the standard version (Zivan and Peled, 2012). It operates on a directed factor graph (c.f. Figures 5.5 and 5.6) to avoid cycles. The transformation between these two graphs is produced by giving each agent a unique index to create an order.

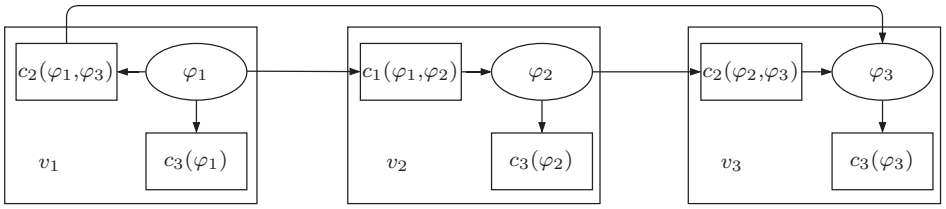

FIGURE 5.5: Vehicle-based directed factor graph for the scenario presented in Figure 5.2. There are 3 agents (v_1, v_2, v_3), each holds an admission time as a variable node ($\varphi_1, \varphi_2, \varphi_3$), 1 anteriority factor node $c_1(\varphi_1, \varphi_2)$, 2 conflict factor nodes ($c_2(\varphi_1, \varphi_3)$ and $c_2(\varphi_2, \varphi_3)$), and 3 waiting time factor nodes ($c_3(\varphi_1)$, $c_3(\varphi_2)$, and $c_3(\varphi_3)$).

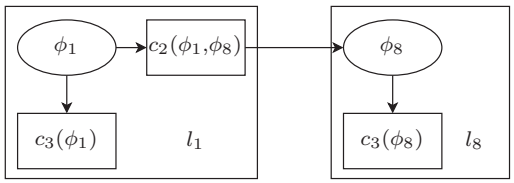

FIGURE 5.6: Lane-based directed factor graph for the scenario presented in Figure 5.2. Only two lanes have vehicles approaching the intersection so there are only two agents (l_1 and l_8). Each agent holds an array variable node, ϕ_1 contains φ_1 and φ_2, ϕ_8 contains φ_3. There are 2 waiting time factor nodes ($c_3(\phi_1)$ and $c_3(\phi_8)$), and 1 conflict factor node ($c_2(\phi_1, \phi_8)$).

Max-Sum_AD was first proposed to address the shortcoming of standard Max-Sum not converging or producing low-quality solutions on a cyclic factor graph. In more

details, Max-Sum_AD uses the same messages as Max-Sum (cf. Equations 4.14 and 4.15). At each phase, messages are sent only on one direction (e.g. upstream direction in odd phases and downstream direction in even phases). Messages are computed by considering all received messages (i.e. messages from upstream neighbours in the current phase, and messages received in the previous phase from downstream neighbours).

From the third phase, Max-sum_AD_VP adds value propagation to Max-Sum_AD. In this technique, each variable node selects a currently optimal assignment and sends it alongside the standard message. Factor nodes then, based on the value chosen, compute messages by minimising only over assignments that are consistent with the value chosen.

To transform the original factor graph into an acyclic directed graph, Max-sum_AD_VP has no preference and often uses the variable indices. Since the solution quality of Max-sum_AD_VP is highly related to the initial assignments, we aim to find a good way to organise nodes to improve its performance. In our system, vehicles come with different priorities and we can see that the optimal solution is more likely to favour vehicles with high priority. Thus, we conjecture that by arranging the nodes in the priority order the algorithm can converge faster to a better solution. This is due to the fact that during the value propagation phases, the nodes with higher priority propagate their values first. In section 5.5.2 we will evaluate the performance of ordering nodes in different traffic conditions.

In the lane-based approach, instead of the priority of the vehicles, lane agents use the sum of the priorities over the vehicles presented in the lane.

5.5 Empirical Evaluation

In this section, we evaluate the performance of our mechanism and the efficiency of the improvements that we proposed for the Max-sum_AD_VP algorithm. The experiments were performed using an Intel Core i5 clocked at 2.9 GHz with 32 GB RAM, under Ubuntu 16.04. The Max-sum_AD_VP algorithm is implemented

as per (Zivan and Peled, 2012). We compare values from at least 50 simulations, with 95% confidence interval as error bars. The insertion rate of vehicles to the intersection ranges from 0.1 (off-peak) to 0.5 (rush hour) (Junges and Bazzan, 2008).

5.5.1 Evaluating Space Efficiency at Individual Intersections

In this first benchmark, we aim to compare the performance of our model and the standard cellular model used in Chapter 4. The intersection evaluated is the one from Figure 5.2. Each incoming lane has a width of 3 meters. We decided to use such intersections as they are one of the most complicated scenarios in urban settings. Vehicles are generated without any priority and both models are evaluated using the lane-based approach with the same standard Max-sum algorithm. A time step is set at 2 seconds and is also the timeout of the DCOP algorithms. If the algorithm fails to provide a solution before timing out, the intersection will automatically apply the FCFS solution as it is very simple to compute and advance to the next time step. Based on the results in Figure 5.7, we observe an improvement in dense traffic only from using space more efficiently, without changing the algorithm.

5.5.2 Evaluating the Efficiency of the Max-sum_AD_VP Algorithm at Individual Intersections

Next, we evaluate in detail our mechanism at a single intersection. Here we evaluate all combinations of the approaches: Vehicle-based approach with node ordering (VB-NO), Lane-based approach with node ordering (LB-NO), standard vehicle-based approach (VB) and standard lane-based approach (LB). Vehicles are generated with a random priority ranging from 1 to 10. We measure the quality of the solution (i.e. the total weighted delay of vehicles) during off-peak and rush

hours. For reference, we also put the results from the model proposed in the previous Chapter on the same weighted delay problem (i.e. using a cellular, standard Max-sum resolution). The intersection and timeout conditions stay the same as the first experiment.

Figure 5.8 shows the average success rate of each approach (i.e. the percentage of iterations where the algorithm converges to a better solution than the one provided by FCFS). We can see that in dense traffic, VB fails to respond to the 2-second timeout and thus, has the worst success rate of about 24% whilst LB converges about 80% of the time with node ordering and 70% of the time without. Figure 5.9 shows the average solution quality when successful. We note that VB tends to converge to a better solution in off-peak conditions. In VB, the solution is less likely to favour vehicles with high priority since it depends more on the number of vehicles in the lane. Therefore, using node ordering with this approach does not always result in a better outcome, and at times pushes Max-sum_AD_VP to greedily pick a worse solution. For LB, since lanes with more vehicles or with vehicles of higher priority are more likely to have shorter delays, using node ordering causes Max-sum_AD_VP to converge faster with higher success rate, especially in dense traffic. Figure 5.10 shows the overall quality of the solution, i.e. the average weighted delay of all vehicles. VB is the solution that gives the best performance in off-peak conditions. In dense traffic, since it often has to take the FCFS solution when it fails to converge, its overall cost is higher than the cost of LB. LB-NO provides a fairly good result in dense traffic and is the best one in rush hours. It outperforms the existing approach by up to 32%. Hence, switching between approaches for different traffic conditions could lead to a better solution for single intersection traffic management. Figure 5.12 shows the anytime quality of the solution to compare the performance between the ordered and the standard versions of the lane-based approach Max-sum_AD_VP. We can clearly observe a better convergence when ordering nodes using priority levels.

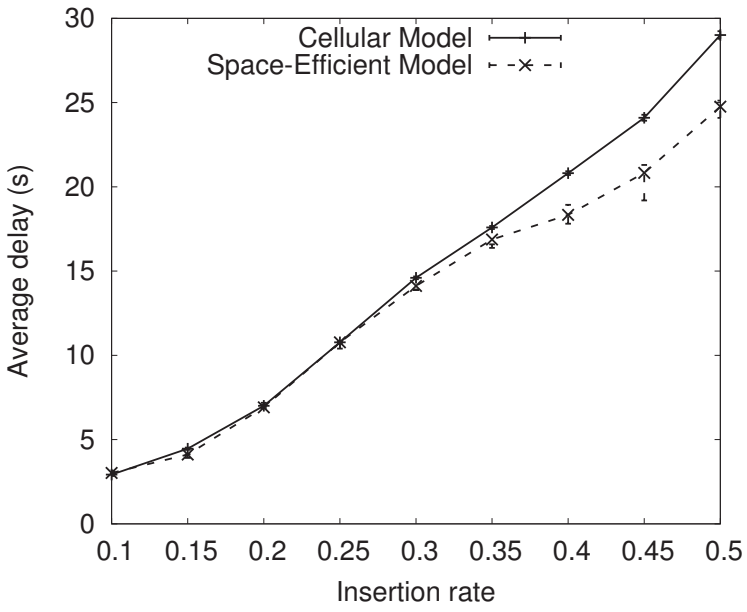

FIGURE 5.7: Performance of the space-efficient model compared to the cellular model.

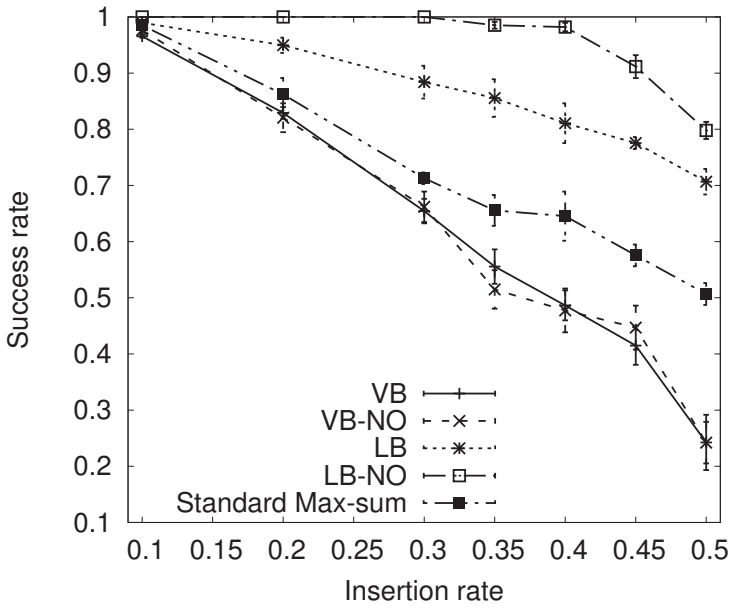

FIGURE 5.8: Success rates of each approach on a weighted delay problem.

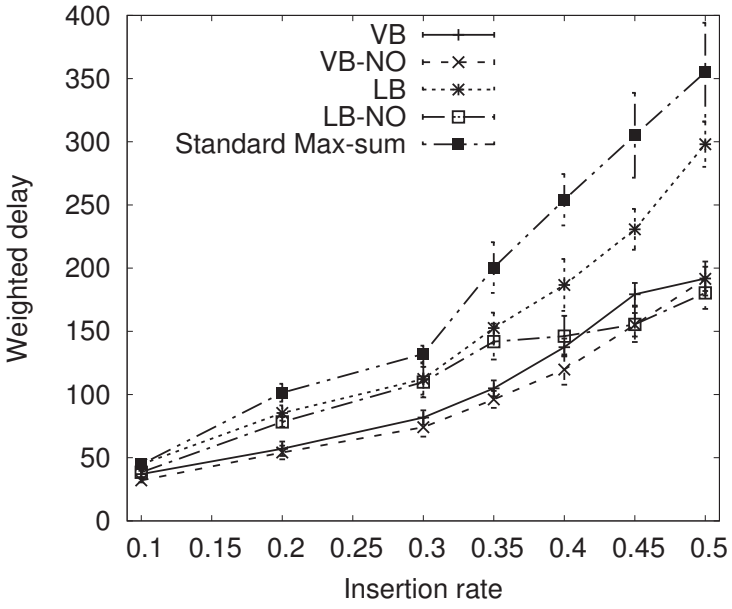

FIGURE 5.9: Average solution quality when successful.

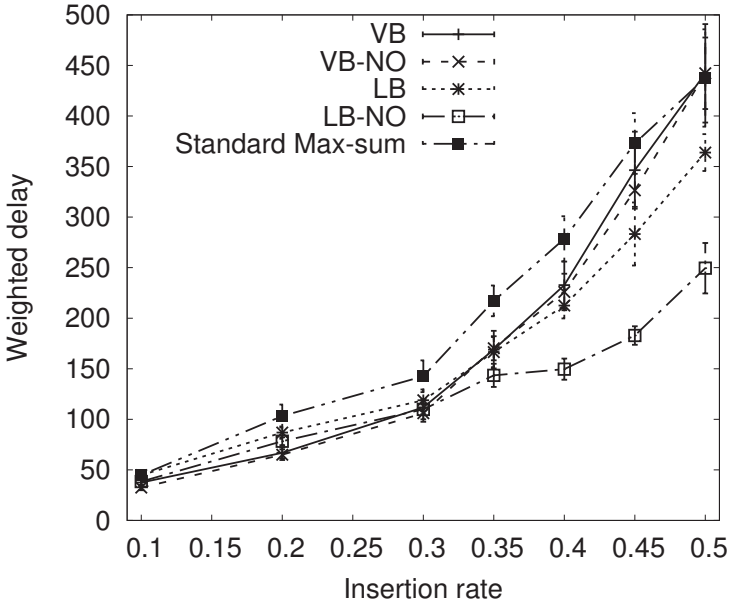

FIGURE 5.10: Average overall weighted delay.

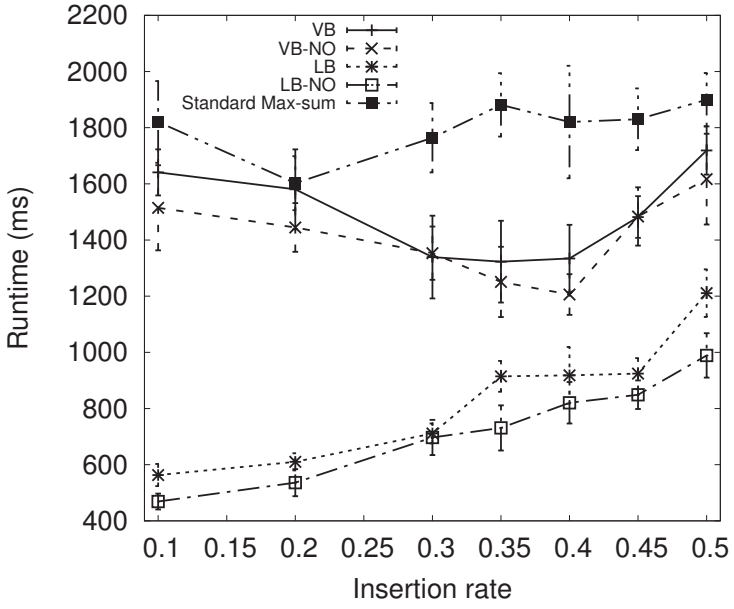

FIGURE 5.11: Average runtime when successful.

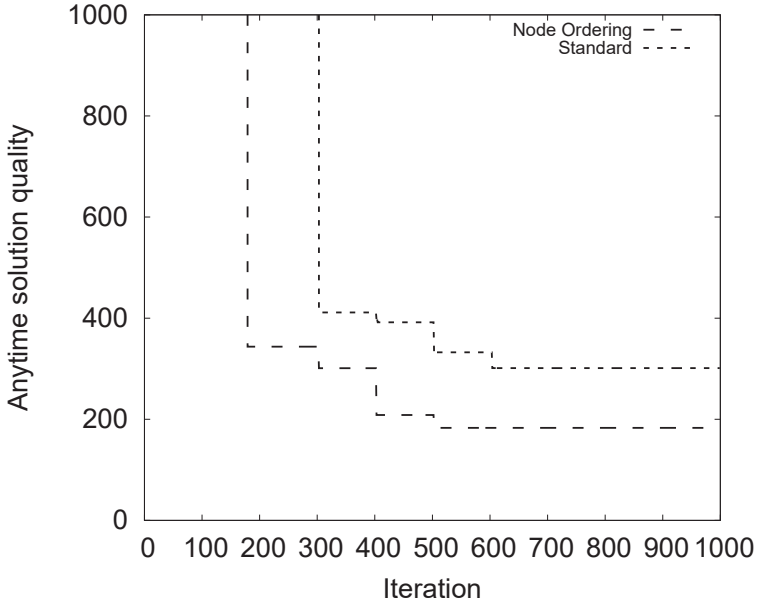

FIGURE 5.12: Anytime cost of ordered and standard versions of Max-sum_AD_VP running on the lane-based approach.

5.5.3 Multi-Intersection Efficiency

To be able to measure the effect of dynamic vehicle priorities, we evaluate our mechanism in two different scenarios using a 2x2 intersection model. In the scenario A, we consider the East-West direction through I_1 and I_2 in the rush hour conditions whilst the other directions in normal conditions (cf. Figure 5.13). This is a common scenario during rush hours in urban traffic. In the scenario B, we consider the east and south *outgoing lanes* of I_4 can only evacuate 1 vehicle every 3 time steps and get crowded (cf. Figure 5.14). Table 5.3 shows results achieved using each individual priority, a combined version using the sum of the priorities, and the standard version.

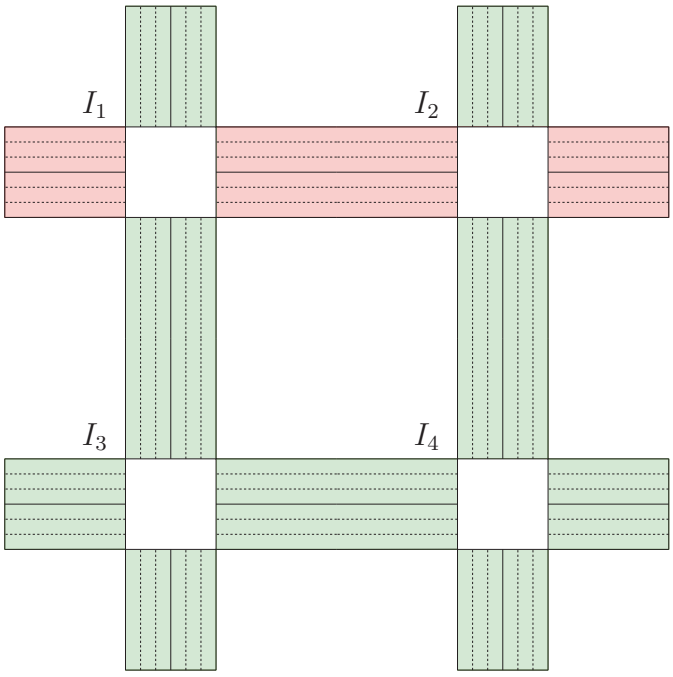

FIGURE 5.13: Scenario 1: The east-west direction through I_1 and I_2 (in red) is more crowded than the other directions.

	Priority by history only	Priority by destination only	Combined priority	No priority
Scenario A	24.74 ± 3.13	27.88 ± 3.86	**21.25 ± 2.25**	32.19 ± 6.29
Scenario B	21.98 ± 3.01	**12.85 ± 2.66**	13.12 ± 3.84	21.16 ± 4.42

TABLE 5.3: Average delay of vehicles in different scenarios.

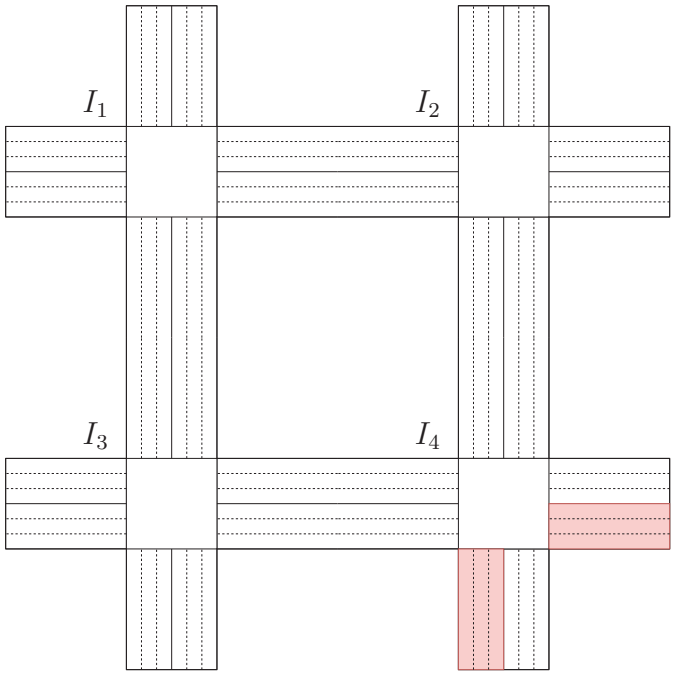

FIGURE 5.14: Scenario 2: The east and south *outgoing lanes* of I_4 (in red) have a limited capacity.

In the scenario A, both priorities contribute to the improvement of the overall solution. Indeed, when we take a closer look at the intersections I_3 and I_4, their north lanes often have to evacuate more vehicles. The *priority by history* speeds up this evacuation since the vehicles in these lanes have suffered from higher delays. On the other hand, the *priority by destination* prevents I_3 and I_4 to send vehicles to the north, since the northern *outgoing lanes* might not be able to evacuate a large number of vehicles.

In the scenario B, we noticed that the *priority by destination* contributes much more to the congestion avoidance. In fact, without the *priority by destination*, vehicles continue to be sent to the intersection I_4, creating a congested situation. This congestion further leads to the impossibility of sending vehicles from I_2 and I_3 to the east and south directions respectively, thus blocked vehicles from entering I_2 and I_3. The average delay grows rapidly due to deadlocks. The *priority by history* makes the performance slightly worse (but not significant) while sending unnecessary vehicles to I_4. In this simulation, we consider that vehicles have a fixed trajectory before entering the network but to extend the model, vehicles

might choose to go from I_2 to I_3 or vice-versa through I_1 instead of I_4 to reduce their delays and optimise the use of traffic network.

5.6 Summary

In this chapter, we proposed several extensions to the model previously used (cf. Chapter 4). First, Section 5.1 discussed the limits of the cellular models. A space-efficient model was introduced in Section 5.2, along with the associated constraints. In regard to a multi-intersection setup, we introduced in Section 5.3 a novel way to coordinate intersections, while avoiding the computationally expensive of network optimisation. This solution was inspired by the market-based solution (Vasirani and Ossowski, 2009) where, right-of-way can be purchased with a variable price, and the distributed weighted graph colouring problem, a common benchmark in DCOP literature in which each agent has its own preferences and priority over its choices. After that, we discussed in Section 5.4 the use of a recent variant of the Max-Sum algorithm and proposed a node ordering mechanism. Finally, the approaches were empirically evaluated in Section 5.5. The results showed that the new variant of the algorithm and the space-efficient model improved the overall solution. The dynamic priority assignment technique is proven to be efficient in multi-intersection settings.

Since the combined version might not be the best in some cases, future work will look at a detailed evaluation of combination between several priority distribution functions to adapt to traffic conditions. This evaluation should include extreme scenarios. One can list several scenarios as followings:

- Vehicles with high priorities coming from every direction.
- Platoons of vehicles taking the same trajectory.
- Different emergency vehicles (e.g. police cars, ambulances, firefighters) approaching an intersection from different directions.

Furthermore, vehicles might start their journeys with a different priority, based on their delays from the past, or based on the cooperation level (e.g. a vehicle that violates their crossing plan might be penalised). Finally, other performance metrics such as fuel consumption and comfort of passengers (due to acceleration, deceleration and stop-and-go) can also be used for evaluation.

Chapter 6

Conclusions and Future Work

This chapter summarises the thesis and discusses possible improvements. In more details, in Section 6.1, we give the results achieved of this thesis while, in Section 6.2, we discuss the challenges in implementing such system and directions for future work.

6.1 Conclusions

In this thesis, we sought to propose a novel traffic regulation mechanism based on a DCOP representation. Our proposal advanced the existing techniques of mitigating congestion in future urban area with the existence of CAVs. To do that, we proposed to distribute computation over vehicles, and coordinate them in order to find the optimal crossing plan through an intersection. We chose DCOP as a solution because of its flexibility and its rich background researches shown in numerous MAS-based applications (Fioretto et al., 2018). When applying DCOP to the microscopic intersection management problem, we had to address certain challenges, mainly due to the computational and communication complexity and we have successfully adopted a DCOP algorithm. We have empirically proven the possibility of applying such framework in a highly dynamic environment, while respecting road safety.

In Chapter 4, to design and implement our proposal, we first modelled the intersection in the traditional cellular way. This kept the problem simple and made it possible to evaluate the performance of a DCOP formalisation. Thereafter, based on the performance of notable DCOP algorithms, we have chosen the Max-Sum algorithm (Farinelli et al., 2008). Thanks to the structure of our problem, we found a better partially centralised approach to speed up the computation and reduce communication overhead. Then, we enhanced safety during the continuous optimisation process by proposing a bound to the solution quality. This helped guarantee a solution at any time step and reduce the search space for the algorithm, thus reduced computational time. Finally, we evaluated the performance of the approach in different traffic densities, as well as in dealing with dynamic event.

Following this, in Chapter 5, we developed several extensions to the previous mechanism. First, we discussed the limits of the standard cellular model. We then modelled the intersection in a more space-efficient way and redefined the associated constraints. The second extension was proposed to address the lack of multi-intersection coordination in microscopic intersection management methods. This extension helped the mechanism deal with traffic conditions at a larger scale, while keeping the same complexity as the single-intersection optimisation problem. The third extension was the use of a recent variant of the Max-Sum algorithm, the Max-Sum_AD_VP algorithm (Zivan and Peled, 2012). When studying the node ordering problem in Max-Sum_AD_VP, we discovered a better way to use our individual priority policy to order nodes and improve the runtime performance of the algorithm. Through the experiments, we showed that the space-efficient model outperformed the cellular version when applying to vehicles with different sizes, and that node ordering can improve the success rate of the algorithm. We also highlighted the needs for a multi-intersection coordination in two common urban traffic scenarios.

When taken together, this thesis presented a first use of DCOP formalisation to address the traffic regulation problem. We have designed a flexible solution that can be applied to optimise traffic at an intersection, as well as at the network

scale. In the next section, we briefly describe how this work can be improved in the future.

6.2 Future Work

6.2.1 Single Intersection Model

In the scope of this thesis, we simulated our mechanism using an intersection with 12 incoming lanes and 12 outgoing lanes, which is one of the most complicated scenarios in the urban area. Regarding the intersection model, future work will first look at different forms of the intersection and evaluate the impact of the mechanism on these forms (e.g. the ones listed in Figure 6.1). These experiments are important as they validate the efficiency of our approach compared to classic traffic regulation methods.

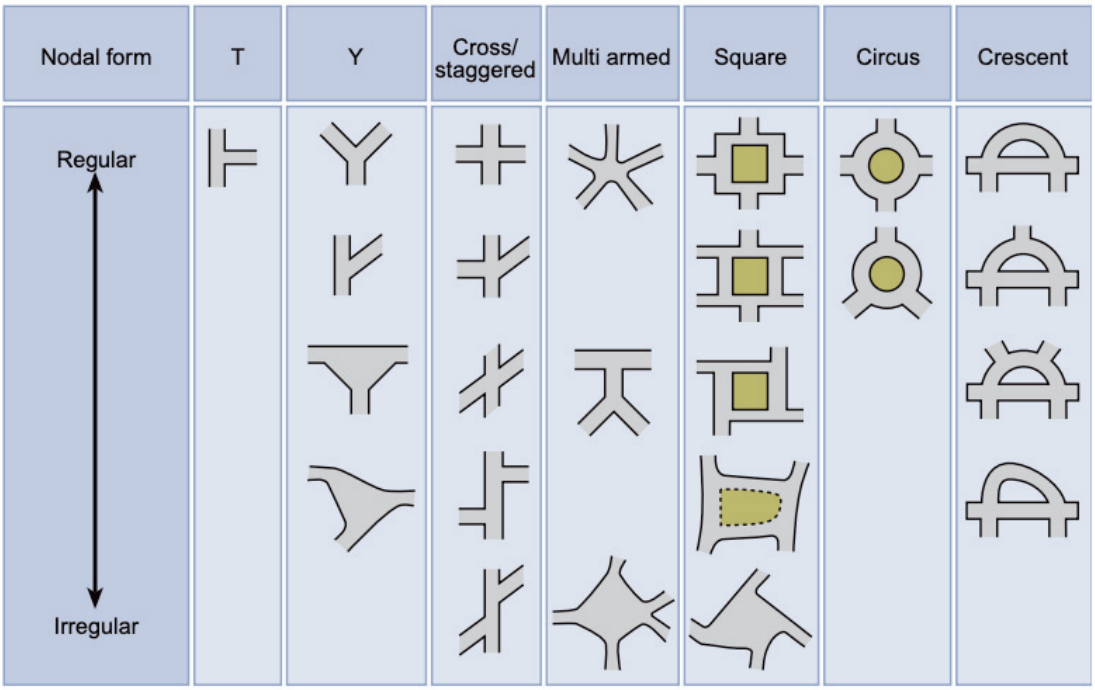

FIGURE 6.1: Different types of intersections in the urban area (Donnelley, 2010).

We assumed, in our thesis, that vehicles travel with a fixed speed. However, vehicles' speed might differ from one to another. This leads to the need of a way to optimise over the current speed of vehicles, in order to build a speed profile allowing them to cross the intersection at their desired crossing time, minimising the negative effects of stop-and-go. This can be done by combining the algorithm with another work that addressed the speed profile problem (e.g., (Zhang et al., 2018)).

In Chapter 4, we evaluated the performance of our mechanism with a dynamic event to test its robustness. However, it is noteworthy that, to validate the mechanism, additional events might be simulated. These events include the arrival of buses, blocked lanes or reduced speed due to maintenance and incidents.

6.2.2 DCOP Algorithms

DCOP is a field of research that had a lot of improvements during the last decade, and new algorithms are continuously proposed. While our formalisation can be solved with any algorithm, a more exhaustive comparison can be studied to choose a suitable algorithm for a realistic system. This evaluation should take into account the communication range of vehicles, as well as their computational capability, memory and the robustness of the message passing protocol. Such research would require the knowledge about the capabilities of CAVs in the future.

When taking DCOP algorithms dynamically, it can be interesting to look at the changes between iterations. Indeed, from an iteration to the next one, the constraints between vehicles that participate in both iterations stay the same (e.g., if two vehicles are constrained with a conflict constraint, since they cannot change the lane, they stay constrained in the next iteration). Therefore, when taking into account that non-changing part in the factor graphs, we can avoid redundant messages by saving knowledge that other agents have achieved during their last iterations. Furthermore, if we can predict certain changes (e.g., vehicles arriving or leaving the area), we can improve the performance of DCOP by only exchanging messages between vehicles that can be affected by these changes.

6.2.3 Multi-Intersection Setup

When dealing with a multi-intersection setup, different cities might have different preferences (e.g., some might prefer to prioritise vehicles with a higher number of occupants, ordinary vehicles that are going to a hospital or prioritise electric vehicles to encourage the use of this type of energy). This leads to a need of studying the priority distribution with multiple objectives. Indeed, any function can be applied to distribute this priority. Hence, finding the optimal way for priority distribution might requires simulation and data collection in a larger scale.

Furthermore, simulating traffic on a larger scale (e.g. a city) using real data might be interesting. In a real city, the traffic contains many complex factors, each of them contributes differently to traffic congestion. By evaluating the mechanism on real data, we can be able to measure the performance of the system in different realistic conditions.

In addition, a future intelligent transportation system may include many types of vehicles, each has to be dealt with differently. We can list some of them as follows:

- **Public transportation:** Current solution in cities to favour public transportation (e.g., buses) usually implies giving them a dedicated lane. In our system, buses can be dealt with using a different manner, by giving them additional priority over other vehicles. However, priority for buses must be different than other types of vehicles because unlike other vehicles which want to arrive as fast as possible, buses have to respect their timetable, and, arriving to their destinations too soon or too late should both be penalised. Since the arrival of a bus can be predicted, we can further develop a more complex strategy where the lane of the bus might be dynamically reserved before its arrival, while can still be used for other vehicles outside of this reservation.

- **Car sharing:** A person that is willing to share his/her car can be prioritised. Furthermore, this type enables the possibility to predict traffic beforehand because the departure time and the arrival time of the car to pick up the

passengers are often fixed in advance. Such prediction leads to the need of studying how to regulate traffic efficiently using these data.

- **Autonomous taxi:** The system might also include autonomous taxi, which can be used to pick up passengers. This type of vehicles needs an efficient mechanism to assign each vehicle to its passengers, in order to minimise cost (e.g. minimise trajectory length, number of passengers on board or fuel consumption). Since passengers are informed of their position and might be able to wait for its arrival, the regulation method can also consider these vehicles differently.

Finally, our work can be adapted to fit alongside other policies that addressed other challenges in urban traffic, such as optimising lane changing decision (e.g., (Cao et al., 2017)). When combined altogether, these approaches can provide a complete autonomous driving policy in the urban area.

Bibliography

Abdoos, M., Mozayani, N., and Bazzan, A. L. C. Holonic multi-agent system for traffic signals control. *Engineering Applications of Artificial Intelligence*, 26: 1575–1587, 2013.

Ashtiani, F., Fayazi, S. A., and Vahidi, A. Multi-intersection traffic management for autonomous vehicles via distributed mixed integer linear programming. In *2018 Annual American Control Conference (ACC)*, pages 6341–6346, 2018.

Balan, G. and Luke, S. History-based traffic control. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 616–621, 2006.

Barceló, J. Cars, a demand-responsive traffic control system. In *Applications of advanced technologies in transportation engineering : proceedings of the second international conference*, 1991.

Bazzan, A. L. C. and Klügl, F. A review on agent-based technology for traffic and transportation. *The Knowledge Engineering Review*, pages 375–403, 2014.

Bazzan, A. L. C., De Oliveira, D., Klügl, F., and Nagel, K. To adapt or not to adapt–consequences of adapting driver and traffic light agents. In *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*, pages 1–14. 2008.

Bielefeldt, C. and Busch, F. MOTION - a new on-line traffic signal network control system. *Road Traffic Monitoring and Control, 1994., Seventh International Conference on*, pages 55–59, 1994.

Bly, P. e-Safety - Co-operative Systems for Road Transport (IST Work Programme 2005-2006). *European Commission, Tech. Rep.*, 2004.

Boillot, F., Midenet, S., and Pierrelee, J.-C. The real-time urban traffic control system CRONOS: Algorithm and experiments. *Transportation Research Part C : Emerging Technologies 14.1*, pages 18–38, 2006.

Brockfeld, E., Barlovic, R., Schadschneider, A., and Schreckenberg, M. Optimizing traffic lights in a cellular automaton model for city traffic. *Phys. Rev. E*, 64, 2001.

Camponogara, E. and Kraus Jr, W. Distributed learning agents in urban traffic control. In *Progress in Artificial Intelligence*, pages 324–335. 2003.

Cao, P., Hu, Y., Miwa, T., Morikawa, T., and Liu, X. An optimal mandatory lane change decision model for autonomous vehicles in urban arterials. *Journal of Intelligent Transportation Systems*, 2017.

Carpenter, C. J., Dugan, C. J., Kopena, J. B., Lass, R. N., Naik, G., Nguyen, D. N., Sultanik, E., Modi, P. J., and Regli, W. C. Intelligent systems demonstration: Disaster evacuation support. In *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 2*, pages 1964–1965, 2007.

Champion, A. Mécanisme de coordination multi-agent fondé sur des jeux: applicationa la simulation comportementale de trafic routier en situation de carrefour. *Ph.D thesis, Université de Valenciennes et du Hainaut-Cambrésis*, 2003.

Chen, Z., Deng, Y., and Wu, T. An iterative refined max-sum_ad algorithm via single-side value propagation and local search. In *AAMAS*, 2017.

Davidson, E. M., Dolan, M. J., McArthur, S. D. J., and Ault, G. W. The use of constraint programming for the autonomous management of power flows. *2009 15th International Conference on Intelligent System Applications to Power Systems*, pages 1–7, 2009.

de Campos, G. R., Falcone, P., and Sjöberg, J. Autonomous cooperative driving: a velocitybased negotiation approach for intersections crossing. In *16th International IEEE Conference on Intelligent Transportation Systems*, 2013.

Diakaki, C. Integrated control of traffic flow in corridor networks. *Ph.D Thesis, Technical University of Crete*, 1999.

Dinanga, E. K. and Pasin, M. Toward equitable vehicle-based intersection control in transportation networks. *8th International Workshop on Agents in Traffic and Transportation (ATT-2014)*, 2014.

Do Chung, B., Yao, T., Friesz, T. L., and Liu, H. Dynamic congestion pricing with demand uncertainty: a robust optimization approach. *Transportation Research Part B: Methodological*, 46:1504–1518, 2012.

Doniec, A., Mandiau, R., Piechowiak, S., and Espié, S. Anticipation based on constraint processing in a multi-agent context. *Autonomous Agents and Multi-Agent Systems*, 17:339–361, 2008.

Donnelley, R. R. Designing streets: A policy statement for Scotland, 2010.

Dresner, K. and Stone, P. Sharing the road : autonomous vehicles meet human drivers. In *IJCAI*, 2007.

Dresner, K. and Stone, P. A multiagent approach to autonomous intersection management. *JAIR*, 31:591–656, 2008.

Farinelli, A., Rogers, A., Petcu, A., and Jennings, N. R. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *AAMAS '08*, pages 639–646, 2008.

Faruqui, A. and Sergici, S. Household response to dynamic pricing of electricity: a survey of 15 experiments. *Journal of regulatory Economics*, pages 193–225, 2010.

Fayazi, S. A., Vahidi, A., and Luckow, A. Optimal scheduling of autonomous vehicle arrivals at intelligent intersections via milp. *2017 American Control Conference (ACC)*, pages 4920–4925, 2017.

Fioretto, F., Pontelli, E., and Yeoh, W. Distributed constraint optimization problems and applications: A survey. *Journal of Artificial Intelligence Research*, 61:623–698, 2018.

France, J. and Ghorbani, A. A. A multiagent system for optimizing urban traffic. In *Intelligent Agent Technology, 2003. IAT 2003. IEEE/WIC International Conference on*, pages 411–414, 2003.

Gaciarz, M., Aknine, S., and Vu, H. A constraint-based coordination model to advantage buses in urban traffic. *ICTAI*, pages 1144–1152, 2017.

Gaciarz, M., Aknine, S., Vu, H., and Bhouri, N. A continuous negotiation based model for traffic regulation at an intersection. Technical report, unpublished.

Gartner, N. H., Pooran, F. J., and Andrews, C. M. Implementation of the opac adaptive control strategy in a traffic signal network. In *Intelligent Transportation Systems, 2001. Proceedings.*, pages 195–200, 2001.

Goel, A. and Kumar, P. Characterisation of nanoparticle emissions and exposure at traffic intersections through fast–response mobile and sequential measurements. *Atmospheric Environment*, 107:374 – 390, 2015.

Grégoire, J., Bonnabel, S., and de La Fortelle, A. Optimal cooperative motion planning for vehicles at intersections. *arXiv preprint arXiv:1310.7729*, 2013.

Grinshpoun, T. and Meisels, A. Completeness and performance of the APO algorithm. *Journal of Artificial Intelligence Research*, 33:223–258, 2008.

Gutierrez, P. and Meseguer, P. Saving redundant messages in BnB-ADOPT. In *AAAI*, 2010.

Gutierrez, P., Meseguer, P., and Yeoh, W. Generalizing ADOPT and BnB-ADOPT. In *IJCAI*, 2011.

Hafner, M. R., Cunningham, D., Caminiti, L., and Del Vecchio, D. Automated vehicle-to-vehicle collision avoidance at intersections. In *Proceedings of World Congress on Intelligent Transport Systems*, 2011.

Hausknecht, M., Au, T.-C., and Stone, P. Autonomous intersection management: Multi-intersection optimization. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4581–4586, 2011.

Henry, J.-J., Farges, J. L., and Tuffal, J. The prodyn real time traffic algorithm. *IFAC/IFIP/IFORS conference on control*, 1984.

Hunt, P. B., Robertson, D. I., Bretherton, R. D., and Royle, M. C. The scoot on-line traffic signal optimisation technique. *Traffic Engineering & Control 23.4*, 1982.

Junges, R. and Bazzan, A. L. C. Evaluating the performance of dcop algorithms in a real world, dynamic problem. In *AAMAS '08*, pages 599–606, 2008.

Kennedy, J. International comparison of roundabout design guidelines. *Transport Research Laboratory*, 2007.

Kosonen, I. Multi-agent fuzzy signal control based on real-time simulation. *Transportation Research Part C: Emerging Technologies*, 11:389–403, 2003.

Kschischang, F. R., Frey, B. J., and Loeliger, H.-A. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, pages 498–519, 2001.

Léauté, T., Ottens, B., and Szymanek, R. FRODO 2.0: An open-source framework for distributed constraint optimization. In *IJCAI'09 (DCR)*, pages 160–164, 2009.

Lee, J. and Park, B. Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment. *Intelligent Transportation Systems, IEEE Transactions on*, 13(1):81–90, 2012.

Lhopital, S., Aknine, S., Thavonekham, V., Vu, H., and Ramchurn, S. D. Decentralised control of intelligent devices: A healthcare facility study. In *EUMAS*, 2020.

Lighthill, M. J. and Whitham, G. B. On kinematic waves I. Flood movement in long rivers. In *Proc. R. Soc. Lond. 229A*, 1955a.

Lighthill, M. J. and Whitham, G. B. On kinematic waves II. A theory of traffic flow on long crowded roads. In *Proc. R. Soc. Lond. 229A*, 1955b.

Macarthur, K., Stranders, R., Ramchurn, S. D., and Jennings, N. R. A distributed anytime algorithm for dynamic task allocation in multi-agent systems. In *AAAI '11*, pages 701–706, August 2011.

Maerivoet, S. and Moor, B. D. Cellular automata models of road traffic. *Physics Reports*, 419:1 – 64, 2005.

Maheswaran, R. T., Pearce, J. P., and Tambe, M. Distributed algorithms for DCOP: A graphical-game-based approach. In *ISCA PDCS*, 2004a.

Maheswaran, R. T., Tambe, M., Bowring, E., Pearce, J. P., and Varakantham, P. Taking dcop to the real world: efficient complete solutions for distributed multi-event scheduling. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, 2004. AAMAS 2004.*, pages 310–317, 2004b.

Mailler, R. and Lesser, V. Solving distributed constraint optimization problems using cooperative mediation. In *AAMAS*, 2004.

Marsa-Maestre, I., de la Hoz, E., Jonker, C. M., and Klein, M. Using graph properties to enable better negotiations in complex self-interested networks. In *The Eighth International Workshop on Agent-based Complex Automated Negotiations (ACAN2015)*, 2015.

Mauro, V. and Di Taranto, C. UTOPIA. *Control, computers, communications in transportation*, 1990.

Miller, S., Ramchurn, S. D., and Rogers, A. Optimal decentralised dispatch of embedded generation in the smart grid. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, pages 281–288, 2012.

Mirchandani, P. and Head, L. A real-time traffic signal control system: architecture, algorithms, and analysis. *Transportation Research Part C : Emerging Technologies 9.6*, pages 415–432, 2001.

Modi, P. J. and Veloso, M. Multiagent meeting scheduling with rescheduling. In *DCR '04*, 2004.

Modi, P. J., Shen, W.-M., Tambe, M., and Yokoo, M. Adopt: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161:149–180, 2005.

Morales, J., López-Sánchez, M., and Esteva, M. Using experience to generate new regulations. In *IJCAI*, pages 307–312, 2011.

Peri, O. and Meisels, A. Synchronizing for performance - DCOP algorithms. In *ICAART*, 2013.

Petcu, A. and Faltings, B. Approximations in distributed optimization. In *CP*, 2005a.

Petcu, A. and Faltings, B. A scalable method for multiagent constraint optimization. In *IJCAI*, 2005b.

Petcu, A., Faltings, B., and Mailler, R. PC-DPOP: A new partial centralization algorithm for distributed optimization. In *IJCAI*, 2006.

Ramchurn, S. D., Farinelli, A., Macarthur, K., and Jennings, N. R. Decentralized coordination in robocup rescue. *Comput. J.*, pages 1447–1461, 2010.

Robertson, D. I. Transyt: a traffic network study tool. 1969.

Rogers, A., Farinelli, A., Stranders, R., and Jennings, N. R. Bounded approximate decentralised coordination via the max-sum algorithm. *Artificial Intelligence*, 175:730–759, 2011.

Schepperle, H. and Böhm, K. Agent-based traffic control using auctions. In *Cooperative Information Agents XI*, pages 119–133. 2007.

Schepperle, H., Böhm, K., and Forster, S. Traffic management based on negotiations between vehicles–a feasibility demonstration using agents. In *Agent-Mediated Electronic Commerce and Trading Agent Design and Analysis*, pages 90–104. 2009.

Schrank, D., Eisele, B., and Lomax, T. 2019 urban mobility report. 2019.

Sims, A. G. and Dobinson, K. W. The sydney coordinated adaptive traffic (SCAT) system philosophy and benefits. *IEEE Transactions on Vehicular Technology 29.2*, pages 130–137, 1980.

Stranders, R., Farinelli, A., Rogers, A., and Jennings, N. R. Decentralised coordination of mobile sensors using the max-sum algorithm. In *IJCAI'09*, pages 299–304, 2009.

Tlig, M., Buffet, O., and Simonin, O. Decentralized traffic management: A synchronization-based intersection control. In *ICALT'14*, 2014.

United Nations Economic Commission for Europe. Vienna convention on road traffic, 1968.

United Nations Economic Commission for Europe. Convention on road signs and signals of 1968, 2006.

van Rijn, J. Road capacities. *Indevelopment*, 2014.

Vasirani, M. and Ossowski, S. A market-inspired approach to reservation-based urban road traffic management. In *AAMAS '09*, pages 617–624, 2009.

Vincent, R. A. and Peirce, J. R. Mova: Traffic responsive, self-optimising signal control for isolated intersections. *Rapp. tech.*, 1988.

Vu, H., Aknine, S., and Ramchurn, S. D. A decentralised approach to intersection traffic management. In *IJCAI*, 2018.

Vu, H., Aknine, S., Ramchurn, S. D., and Farinelli, A. Decentralised multi-intersection congestion control for connected autonomous vehicles. In *EUMAS*, 2020.

Yan, F., Wu, J., and Dridi, M. A scheduling model and complexity proof for autonomous vehicle sequencing problem at isolated intersections. In *Service Operations and Logistics, and Informatics (SOLI), 2014 IEEE International Conference on*, pages 78–83, 2014.

Yedidsion, H., Zivan, R., and Farinelli, A. Applying max-sum to teams of mobile sensing agents. *Engineering Applications of Artificial Intelligence*, 71:87–99, 2018.

Yeoh, W., Felner, A., and Koenig, S. BnB-ADOPT: an asynchronous branch-and-bound DCOP algorithm. In *AAMAS*, 2008.

Zhang, W., Wang, G., Xing, Z., and Wittenburg, L. Distributed stochastic search and distributed breakout: Properties, comparison and applications to constraint optimization problems in sensor networks. *Artificial Intelligence*, 161:55–87, 2005.

Zhang, Y., Chen, H., Waslander, S. L., Yang, T., Zhang, S., Xiong, G., and Liu, K. Toward a more complete, flexible, and safer speed planning for autonomous driving via convex optimization. In *Sensors*, 2018.

Zivan, R. and Peled, H. Max/min-sum distributed constraint optimization through value propagation on an alternating dag. In *AAMAS*, pages 265–272, 2012.

Zivan, R., Glinton, R., and Sycara, K. Distributed constraint optimization for large teams of mobile sensing agents. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 02*, pages 347–354, 2009.

Zou, X. and Levinson, D. Vehicle-based intersection management with intelligent agents. In *ITS America Annual Meeting. Minneapolis, Minnesota*, 2003.