Vehicle-centric coordination for urban road traffic management: A market-based multiagent approach

Tesis presentada por

Matteo Vasirani

Departamento de Arquitectura y Tecnología de Computadores,
Ciencias de la Computación e Inteligencia Artificial

para la obtención del título de
Doctor en Informática

2009

D. Sascha Ossowski, Profesor Catedrático de la Universidad Rey Juan Carlos, con NIE X-1805919-M,

CERTIFICA: Que D. Matteo Vasirani, licenciado en Informática, ha realizado en el Dpto. de Arquitectura de Computadores y Ciencias de la Comunicación e Inteligencia Artificial, bajo su dirección, el trabajo correspondiente a la tesis doctoral titulada:

Vehicle-centric coordination for urban road traffic management: A market-based multiagent approach

Revisado el presente trabajo, estima que puede ser presentado al tribunal que ha de juzgarlo. Y para que conste a efecto de lo establecido en la normativa reguladora del tercer ciclo de la Universidad Rey Juan Carlos, autoriza su presentación.

Móstoles, 1 de Noviembre de 2009.

Fdo.: Sascha Ossowski

Acknowledgments

Completing this doctoral work has been a wonderful and challenging experience. It is hard to decide if the real learning experience was learning how to write a paper, give a talk or coding fast and intelligibly.

First of all, I'd like to thank my supervisor, Sascha Ossowski, for his help, his encouragement and his priceless ability of distinguishing a potentially good idea from something that someone had already done in the nineties.

I would also like to thank my colleagues at URJC, both past and present (Alberto, Ana, Cesar, Holger, Ingo, Jesús, Juanma, José Miguel, Moser, Ramón, Roberto, Rubén, Sergio), for the pleasant and relaxed working environment that we were able to create.

I'm also grateful to Rosaldo Rossetti and Eugenio Oliveira, who received me in Porto (Portugal) during my research stay, and to Ana Bazzan, who supervised my research stay in Porto Alegre (Brazil) and also took me to the doctor when I twisted my ankle playing football. Thanks also to Jürgen Dunkel for his comments on the thesis.

Going back in time, I cannot forget that my very first research training was given by Franco Zambonelli and Marco Mamei, during my master thesis in 2003, and by Andrea Omicini, with his list of fundamental readings and a fruitful exchange of philosophical emails about Popper, Kuhn and the Schrödinger's cat.

I want to thank my family, my mum Roberta, my dad Silvano, my brother Fabio, my nephew Luca and my niece Irene, who always supported me even when I decided to leave Italy (actually my nephew and my niece weren't born when I left Italy, but they supported me because they are so lovely).

Finally, I also want to give a special thanks to my friends, in Italy and in Spain. Nicola, for our delirious discussions about the future of the mankind, Stephen, for our even more delirious discussions about traffic lights that pay you if you are stopped at the intersection, Rosie, who always helped me refining my English grammar, Manuel, who still thinks that I'm studying how to make an invisibility tracksuit, Ceci, Betta, Floriana, Antonio, Marcus, Eva, Zory, Alberto and all the European taxpayers who funded my research grants.

To Dani, who has been my guiding light and
love over these last two years,
providing support, friendship and hugs.

Thesis advisor Author
Prof. Sascha Ossowski **Matteo Vasirani**

Abstract

Traffic congestion in urban road networks is a costly problem that affects all major cities in developed countries. For example, the Texas Transportation Institute estimated that traffic jams in the U.S. cost more than 78 billions dollars every year, in fuel consumption and productivity loss [1].

To tackle this problem, it is possible i) to increase the capacity of the network, adding more lanes or more roads, ii) to reduce the demand, restricting the access to urban areas at specific hours or to specific vehicles, or iii) to improve the efficiency of the existing network, by means of a widespread use so-called Intelligent Transportation Systems [41].

In line with the recent advances in telematic infrastructures, the traffic control and management problem has turned out to be a promising application field for multiagent system technology [115]. Multiagent systems (MAS) are the ideal candidates for the design and implementation of such systems, since many problems in this domain are inherently distributed and the actors fit perfectly the paradigm of autonomous agent [11].

In this thesis, several distributed, market-based, mechanisms have been studied and applied to the management of a (future) urban road network, where intelligent autonomous vehicles, governed by *driver agents*, interact with the infrastructure in order to travel through the network. Starting from the reservation-based intersection model proposed by Dresner and Stone in [35], this thesis studied how to implement a computational economy where the driver agents must acquire the necessary reservations to cross the intersections that compose their routes, while the agents in charge of managing the intersections (*intersection managers*) participate in the market as suppliers of such reservations.

Two scenarios have been studied, one with a *single intersection* and one with

a *network of intersections.* In the first case, we have developed different policies to control a reservation-based intersection, based on the *adversarial queueing theory* and the *combinatorial auction theory.* In the second case, we have studied two different models of computational economy to deal with the traffic assignment problem. The first one, $\mathcal{ECO}^+$, is a *cooperative* model, where the intersection managers learn to operate in the market to optimise a global profit measure for the society of intersection managers and, indirectly, the travel time of the driver agents. The second one, $\mathcal{ECO}^-$, is a *competitive* model, where the intersection managers compete with each other as suppliers of the reservations that are traded in the market, aiming at reaching the market equilibrium, that is, a situation where the amount of resources sought by buyers (driver agents) is equal to the amount of resources produced by suppliers (intersection managers). Finally, we combined the auction-based policy for traffic control and the competitive model for traffic assignment into an adaptive, integrated, strategy for full-fledged traffic management, $\mathcal{ECO}^-_{CA}$.

In parallel to the theoretical design of the market-based mechanisms, in this thesis we developed a simulation tool, called $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ (Multiagent Intelligent Transportation Environment), to evaluate the proposed mechanisms and to show how these mechanisms affect the driver agents' utility as well as the system utility. This simulator implements two validated traffic flow models (the mesoscopic model of Schwerdtfeger [89] and the microscopic model of Nagel and Schreckenberg [68]), and provide a powerful tool that enables the simulation of thousands of vehicles with high precision.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

It doesn't matter how beautiful your theory is,
it doesn't matter how smart you are.
If it doesn't agree with experiment, it's wrong.

Richard P. Feynman

"Bob woke up quite early this morning, today is the first day of work at his new company and he doesn't want to be late. A quick coffee and he gets into his car. As soon as he opens the car door, the dashboard quickly starts up, and his personal driver agent is up and running. Equipped with voice recognition software, the agent is ready to receive commands about the destination that Bob wants to reach. "125 Elm Street", Bob says. On the basis of Bob's profile, the driver agent selects a route and shows it to Bob on the dashboard. The route takes approximately 45 minutes to reach the destination and is almost free, only 10 cents to cross an intersection that connects the surrounding neighbourhoods with the downtown, near Oak Road. Usually Bob is not in a hurry, but this time is different, he wants to make a good impression with his new boss. Thus, using the touchscreen dashboard, he changes his profile for today and he sets it to <BusinessMode>, increasing by 3 euros the money he is willing to spend for this trip. The agent finds a new, faster and more expensive route, which takes only 25 minutes, and costs 2.5 euros, because it passes through a highly demanded intersection

Figure 1.1: Chevy Boss, winner of the Darpa Urban Challenge 2007

near the business district. Bob confirms the new route, and the driver agent starts the car and gently drives out of the garage. While Bob takes a look at some documents related to his new job position, the driver agent travels autonomously and smoothly towards the destination, paying the travel fees to each intersection manager agent that governs the intersection it passes through. The driver agent continuously consults the information that it spread throughout the road infrastructure, and it detects that the price of a previously quite expensive intersection has fallen. A quick replanning, and a new route is set up: it takes 2 minutes more than the previous one, but for 1.8 euros less, a good deal. The driver agent safely drives the car to the destination, 125 Elm Street, and notifies Bob with a gentle "Destination reached" message.

The above story sounds far-fetched, but such a scene may be closer than we think. Indeed, removing the human driver from the control loop by the use of *autonomous vehicles* and the integration of these with the *intelligent infrastructure* can be considered the ultimate, long-term, goal of the set of systems and technologies grouped under the name of Intelligent Transportation Systems (ITS) [41].

Autonomous vehicles are already a reality. For instance, two DARPA Grand Challenge and one DARPA Urban Challenge[1] have been hitherto held. This event, organised by the Defense Advanced Research Projects Agency (DARPA), required teams to build an autonomous vehicle capable of driving in traffic, performing complex manoeuvres such as merging, passing, parking and negotiating intersections. This event can be considered as the first time autonomous vehicles have interacted with both manned and unmanned vehicular traffic in an urban environment. An autonomous vehicle is a vehicle that navigates and drives entirely on its own with no human driver and no remote control. Through the use of various sensors and positioning systems, the vehicle determines all the characteristics of its environment required to carry out the task it has been assigned. An example of this kind of vehicles is Chevy Boss (figure 1.1), developed by the Carnegie Mellon University, winner of the DARPA Urban Challenge 2007. It used GPS, sonar and laser guidance to avoid obstacles and even negotiate intersections with other cars. Several car-makers expect the technology to be affordable (and less obtrusive) in about a decade.

Another initiative that fosters this vision is the Vehicle Infrastructure Integration (VII) initiative[2], which promotes research and development of technologies that are supposed to directly link road vehicles to their physical surroundings. The advantages of such integration span from improved road safety to a more efficient operational use of the transportation network. For example, Dresner and Stone in [35] introduced an infrastructure facility that allows for the control of intersections (see figure 1.2). In their model, an intersection is regulated by an intelligent agent (called intersection manager) that assigns reservations of space slots inside the intersection to driver agents that operate autonomous vehicles that intend to pass through the intersection. Such an approach has shown, in a simulated environment, several advantages, because it may drastically reduce delays with respect to traffic lights and it makes possible the adoption of fine grained, vehicle-centric, control policies.

With a widespread use of ITS technologies, it is reasonable to expect that future urban traffic control and management systems will reach a scale and a complexity

---

[1] http://www.darpa.mil/grandchallenge/index.asp

[2] http://www.intellidriveusa.org

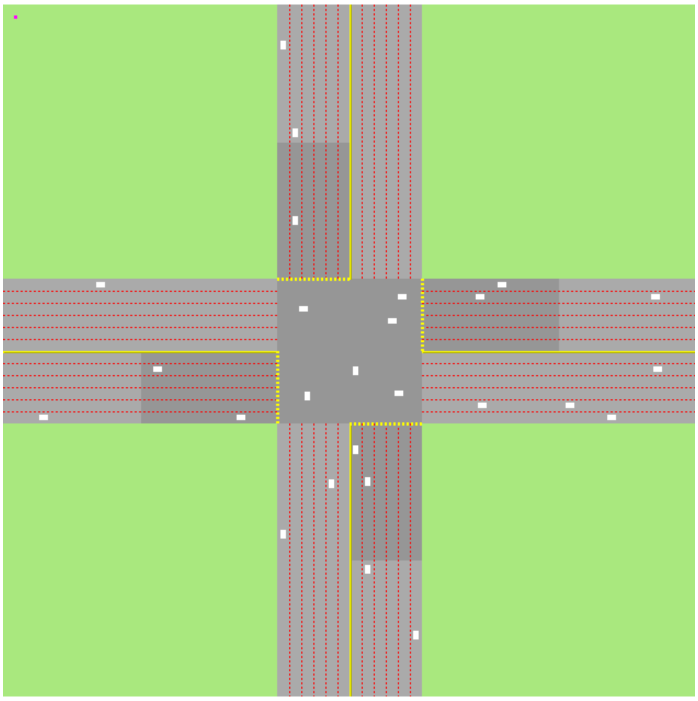


Figure 1.2: Autonomous intersection management

never seen before. The inherent distribution and the high degree of complexity allows for a natural decomposition of such systems into agents that interact so as to achieve their goals, selfishly as well as cooperatively. Therefore, (intelligent) traffic and transportation scenarios are extraordinarily appealing for multiagent technology [8][10][11]. In this kind of system we can clearly distinguish two types of agents: from one side we have the *infrastructure agents*, that is, entities that, provided with sensors and computing power, aim to control the system to alleviate traffic congestion, speed up the traffic flow and guarantee an overall quality of service; from the other side we have the *driver agents*, that is, entities that control the vehicles on behalf of their owners, make autonomous decisions about route choice and departure time selection, learn from their past experiences and influence each other in both positive and negative ways. In general, the system designer (or infrastructure designer) has control over the former, so that they can be modelled at will in terms of actions space, goals, mental attitude etc., while the latter cannot be directly controlled by the system designer. Thus, if we take the perspective of the infrastructure agents, we realise that they face a very complex problem, since they aim at controlling a system that provides only partial and noisy observations, with "weak" actuators in hand that are unable to directly intervene in the driver agents behaviour. Furthermore, the "agentification" of vehicles and infrastructure will make possible for next-generation integrated infrastructures to target vehicles *individually*, instead of whole *flows* of traffic as occur, for instance, with traffic lights.

1.1 Objectives

The main objective of this thesis is studying distributed mechanisms for the control and management of a (future) urban road network, where intelligent autonomous vehicles, controlled by driver agents, interact with the infrastructure in order to travel on the links of the network. This main objective can be broken down into the following, more specific, objectives:

1. This thesis is based on the reservation-based intersection control system proposed by Dresner and Stone in [35]. Thus, the first step is performing an in depth analysis of the aforementioned system for a *single intersection*, with the aim of discovering potential flaws, trying to improve the efficiency of the intersection control strategy and enforcing new desirable properties. We will tackle this task defining several new policies for the assignment of reservations to driver agents, relying on the adversarial queueing theory (AQT) [16] and on the theory of combinatorial auctions (CA) [59]. The policies based on AQT aim to maximise the intersection throughput, while the auction-based policy focuses on rewarding the driver agents that value the reservations the most.

2. Extending the model proposed by Dresner and Stone to a *network of intersections* opens many new interesting questions, apart from considerably complicating the scenario. To deal with this complexity, the traffic management problem must be broken down into two sub-problems: *traffic assignment* and *traffic control*. Traffic assignment can be seen as a resource allocation problem. Due to the large scale of the system at hand, we need a distributed formulation

and solution method. Therefore, it is very interesting to note that markets and economies in general perform distributed resource allocation also in rather complex environments. In fact, markets as a solution method to solve distributed resource allocation problems have been applied to several systems [25][104].

Thus, market-based mechanisms can be used to design an efficient traffic assignment system at urban level. The advantages of such an approach are multiple.

(a) The market dynamic provides the driver agents with incentives to explore different alternatives for the route choice.

(b) The intersection managers, participating in and (eventually) ruling the market, have more power to influence the driver agents behaviour.

(c) The pricing policies have different effects on different groups of driver agents, so that it is possible to apply strategies tailored to different collectives of driver agents.

(d) Reaching equilibrium prices enables the efficient allocation of the available resources (i.e., the capacity of the urban network) among the driver agents.

(e) The market rules (and the way benefits are calculated) can be designed so that the intersection managers, trying to maximise their benefits, "unintentionally" optimise performance measures at system level, such as impact of congestion, average travel time, etc.

3. In this context, computational markets for the traffic assignment can be built. In these markets, the driver agents must acquire the reservations to pass through the reservation-based intersections of the urban network. Since we have control, as system designers, on the behaviour and the goals of the intersection managers, these can be modelled as *cooperative learning agents* that learn which pricing policy optimises a global profit function. Furthermore, modelling appropriately the global profit function, it is possible to link the profit function with some performance measures of the underlying road network, so that the intersection managers, optimising the profit function, indirectly optimise the average travel time of the population of driver agents.

4. Markets in real environments are usually composed of agents that behave selfishly. Thus, another objective of this thesis is studying a *competitive pricing strategy* of the intersection managers, based on the theory of general equilibrium. In this model, each intersection manager competes with all the others for the supply of the resources that are traded, that is, the reservations at the intersections. In this case, our objective as system designers is reaching the market equilibrium, that is, a situation where the amount of resources sought by buyers is equal to the amount of resources provided by suppliers, accounting in this way for an efficient utilisation of the available resources.

5. Finally, another objective of this thesis is combining the traffic control policies that have been studied in the single intersection scenario with the market models for traffic assignment that have been studied in the network of intersections scenario, in order to develop an adaptive, integrated, strategy for full-fledged traffic management.

6. In order to evaluate the proposed mechanisms, a simulation tool is needed. Thus, in parallel to the theoretical design of the aforementioned mechanisms, the *pros* and *cons* of the traffic flow models proposed by the scientific community must be analysed, in order to develop a simulator that fits our needs.

1.2 Structure of the thesis

This thesis is structured in the following way:

1. Chapter 2 revises the state of the art of the research fields to which this thesis is related:

   (a) the set of hardware and software technologies framed by the name Intelligent Transportation Systems [41].

   (b) the multiagent systems, with special emphasis on the topics of coordination, multiagent learning, mechanism design and the application of multiagent systems to traffic and transportation.

2. Chapter 3 tackles the first objective of this thesis, analysing the reservation-based mechanism for autonomous intersection control in the single intersection scenario and evaluating different policies inspired by the adversarial queueing theory and the combinatorial auction theory.

3. Chapter 4 tackles the other objectives of this thesis, introducing the cooperative model and the competitive model for traffic assignment, as well as the integrated strategy for traffic management. Furthermore, the simulation tool $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ is presented, describing the implemented traffic model and the implementation details.

4. Finally, in chapter 5 we will describe the main conclusions of this thesis, the publications that have been produced during the development of this work, as well as some future research lines that are still open and deserve to be studied.

Chapter 2

State of the Art

Everything must justify its existence
before the judgement seat of Reason,
or give up existence.

Friedrich Engels

2.1 Intelligent Transportation Systems

The transport system plays a fundamental role in modern lives, and has a huge impact on our economy, environment and lifestyle. Intelligent Transportation Systems (ITS) [41] refer to the application of hardware and software technology to address and alleviate transportation problems. Traffic congestion has been increasing worldwide as a result of increased motorisation, urbanisation and population growth, causing reduction of the efficiency of the transportation infrastructure and increasing travel time, air pollution, and fuel consumption.

ITS encompasses the full scope of information technologies used in transportation, including control, computation and communication, as well as algorithms, databases, models and human interfaces. The emergence of these technologies as a new way to find solutions for the transportation problems is relatively new. ITS benefits can be quantified in terms of reduction in crashes, reduction in delays and travel times, as well as throughput increase. On the other hand, since many of the employed technologies are still relatively new, difficulties arise to assess the economic impacts of these technologies [13].

ITS includes a constellation of systems and, according to the U.S. department of transportation [3], it can be grouped into two main sub-fields: Intelligent Infrastructure and Intelligent Vehicles.

2.1.1 Intelligent Infrastructure

Arterial Management Systems

Arterial management systems manage traffic along arterial roadways, employing traffic detectors, traffic signals, and various means of communicating information to travellers. These systems make use of information collected by traffic surveillance devices, and provide important information about travel conditions to travellers via technologies such as dynamic message signs (DMS) or highway advisory radio (HAR).

- **Surveillance.** Traffic surveillance refers to the detection technologies, such as sensors or cameras, which aim at monitoring the traffic flow.

- **Traffic Control.** Traffic control systems on arterials optimise travel speeds and provide transit signal priority and signal preemption for emergency vehicles, as well as improve the safety of bicyclists and pedestrians.

  - **Traffic Signal Priority.** Traffic signal priority systems use sensors to detect approaching vehicles and alter signal timings to improve transit performance (e.g., extending the duration of green signals for public transportation vehicles).
  - **Emergency Vehicle Preemption.** Emergency vehicle preemption systems use sensors to detect an approaching emergency vehicle in order to provide it a green signal and so speed up its transit.

  - **Adaptive Signal Control.** Adaptive signal control systems coordinate control of traffic signals, adjusting the lengths of signal phases based on prevailing traffic conditions.
  - **Advanced Signal Systems.** Advanced signal systems include coordinated signal operations across neighbouring jurisdictions.
  - **Bicycle and Pedestrian.** Pedestrian detectors, pedestrian activated lighted crosswalks, specialised pedestrian signals, and bicycle-actuated signals can improve the safety of all road users at signalised intersections and unsignalised crossings.
  - **Special Events.** Arterial management systems can also smooth traffic flow during special events with unique operating schemes, incorporating elements such as special traffic signal operating plans, temporary lane restrictions, traveller guidance, and other measures.

- **Lane Management.** Lane management applications can promote the most effective use of available capacity during emergency evacuations, incidents, construction and a variety of other traffic and/or weather conditions.
  - **High Occupancy Vehicle (HOV) Facilities.** HOV facilities serve to increase the total number of people moved through a congested corridor by offering two kinds of travel incentives: a substantial savings in travel time, along with a reliable and predictable travel time. Sensors detecting the traffic conditions support the use of dynamic message signs and moveable barriers to control the operation of HOV facilities.
  - **Reversible Flow Lanes.** Traffic sensors and lane control signs can be used to implement reversible flow lanes allowing travel in the peak direction during rush hours.
  - **Pricing.** Traffic sensors, electronic payment, video, GPS, and automated enforcement technologies can support the implementation of congestion pricing strategies, varying the cost of transportation facilities based on demand or the time of day.

  - **Lane Control.** Lane control signs, supported by surveillance and detection technologies, allow the temporary closure of lanes to avoid incidents or construction on arterial roadways.
  - **Variable Speed Limits.** Variable speed limit systems use sensors to monitor prevailing traffic and/or weather conditions, posting appropriate enforceable speed limits on dynamic message signs.
  - **Emergency Evacuation.** Lane management applications such as reversible flow lanes and lane control can be used to support emergency evacuations. Such plans can also involve the implementation of special traffic signal timing plans, variable speed limits, and other measures.

- **Parking Management.** Parking management systems with information dissemination capabilities, most commonly deployed in urban centres or at modal transfer points such as airports, monitor the availability of parking and disseminate the information to drivers, reducing traveller frustration and congestion associated with searching for parking.

- **Information Dissemination.** Advanced communications have improved the dissemination of information to the travellers. Organisations operating ITS can share information collected by detectors associated with arterial management systems with road users. They are now able to receive relevant information on location-specific traffic conditions in a number of ways, including dynamic message signs (DMS), highway advisory radio (HAR), and in-vehicle signing (or specialised information transmitted to individual vehicles).

- **Enforcement.** Automated enforcement systems, such as speed enforcement and stop enforcement, improve safety, reduce aggressive driving, and assist in the enforcement of traffic signal and speed compliance. Using photo or video cameras, activated by detectors, they record vehicles travelling faster than the speed limit or vehicles travelling through a red signal. Furthermore, enforcement technologies can assist with the enforcement of high occupancy vehicle (HOV) restrictions. Enforcement personnel can trigger recording technology, such as cameras, to record vehicles that violate traffic regulation.

Freeway Management Systems

Freeway management systems manage traffic along highways, employing traffic surveillance systems, traffic control measures on freeway entrance ramps, such as ramp meters and lane management applications. Advanced communications have improved the dissemination of information to the travelling public. Motorists are now able to receive relevant information on location-specific traffic conditions in a number of ways, including dynamic message signs (DMS), highway advisory radio (HAR), in-vehicle signing, or specialised information transmitted only to a specific set of vehicles.

- **Ramp Control.** Traffic control measures on freeway entrance ramps, such as ramp meters, can use sensor data to optimise freeway travel speeds and ramp meter wait times.
  - **Ramp Metering.** Traffic signals on freeway ramp meters alternate between red and green signals to control the flow of vehicles entering the freeway. Metering rates can be altered based on freeway traffic conditions.
  - **Ramp Closures.** Surveillance and control technologies can allow for the temporary closure of freeway ramps to accommodate peak traffic conditions or inclement weather conditions.
  - **Priority Access.** Communication between ramp metering signal hardware or ramp closure gates, and emergency or common vehicles can allow priority access to these vehicles, providing a green signal or opening the gates to allow for passage of the approaching vehicle.
  - **Ramp Meter Enforcement.** Automated enforcement technologies can assist with the enforcement of ramp metering compliance. Still or video cameras, activated by detectors, can record vehicles travelling through a red signal.

- **Special Event Transportation Management.** Special event transportation management systems can help control the impact of congestion at stadiums or convention centres. In areas with frequent events, large changeable destination signs or other lane control equipment can be installed. In areas with occasional or one-time events, portable equipment can help smooth traffic flow.

Transit Management Systems

Transit ITS services include surveillance and communications, such as automated vehicle location (AVL) systems, computer-aided dispatch (CAD) systems, and remote vehicle and facility surveillance cameras, which enable transit agencies to improve the operational efficiency, safety, and security of the nation's public transportation systems.

- **Safety and Security.** Advanced software and communications enable data as well as voice to be transferred between transit management centres and vehicles for increased safety and security, improved transit operations, and more efficient fleet operations. Transit management centres can monitor in-vehicle and in-terminal surveillance systems to improve quality or service and improve the safety and security of passengers and operators.
  - **In-Vehicle Surveillance.** Video cameras monitor the interior of buses or cars. Wireless communication can make images available to transit dispatch or transit management centres. Automatic vehicle location systems often incorporate silent alarm features, allowing operators to report problems and vehicle location to dispatchers.
  - **Facility Surveillance.** Video and audio surveillance technologies can be deployed to enhance the security of train stations, bus depots, and transit stops.
  - **Employee Credentialing.** A variety of identification and access control systems can help maintain the security of public transportation management and support facilities.

   - **Remote Disabling Systems.** Vehicles in difficulty can be remotely shutdown via wireless communication and control, typically from dispatch centres.

- **Transportation Demand Management.** Transportation demand management service, such as ride sharing/matching, dynamic routing/scheduling, and service coordination, increase public access to transit resources where coverage is limited.

   - **Ride Sharing/Matching.** Computer database and Internet technologies can facilitate ride sharing (also called carpooling) matching services. Ride sharing is when people travel together in one car rather than driving their own cars, mostly used by people commuting to work.
   - **Dynamic Routing/Scheduling.** Automatic vehicle location, combined with dispatching and reservation technologies facilitate the implementation of flexible public transportation routing and scheduling.
   - **Service Coordination.** Vehicle monitoring and communication technologies facilitate the coordination of passenger transfers between vehicles or transit systems.

- **Fleet Management.** Fleet management systems improve transit reliability through implementation of automated vehicle location (AVL) and computer-aided dispatch (CAD) systems which can reduce passenger wait times. These systems may also be implemented with in-vehicle self-diagnostic equipment to automatically alert maintenance personnel of potential problems.

   - **AVL/CAD.** Automatic vehicle location (AVL) and computer aided dispatch (CAD) systems facilitate the management of transit operations, providing up-to-date information on vehicle locations to assist transit dispatchers as well as inform travellers of the bus status.
   - **Maintenance.** Maintenance monitoring technologies allow for the automatic collection and reporting of vehicle maintenance information. Infor-

mation can be uploaded at the end of a run, or while in service via wireless communication.

  - **Planning.** A variety of technologies, including records from AVL/CAD systems and automatic passenger counter systems, can assist in the planning of new and modified transit services.

- **Information Dissemination.** Transit agencies can disseminate both schedule and system performance information to travellers through a variety of applications, in-vehicle, wayside, or in-terminal dynamic messages signs, as well as the Internet or wireless devices. Information dissemination allows passengers to confirm scheduling information, improve transfer coordination, and reduce wait times. Electronic transit status information signs at bus stops help passengers manage time, and on-board systems such as next-stop audio annunciators help passengers in unfamiliar areas reach their destinations. Coordination with regional or multi-modal traveller information efforts can also increase the availability of this transit schedule and system performance information.

Electronic Payment and Pricing

Electronic payment systems employ various communication and electronic technologies to facilitate commerce between travellers and transportation agencies, typically for the purpose of paying tolls and transit fares. Pricing refers to charging motorists a fee or toll that varies with the level of demand or with the time of day.

- **Toll Collection.** Electronic toll collection (ETC) supports the collection of payment at toll plazas using automated systems to increase the operational efficiency and convenience of toll collection. Systems typically consist of vehicle-mounted transponders identified by readers located in dedicated and/or mixed-use lanes at toll plazas.

- **Transit Fare Payment.** Electronic transit fare payment systems, often enabled by smart card or magnetic stripe technologies, can provide increased convenience to customers and generate significant cost savings to transportation agencies by increasing the efficiency of money handling processes and improving administrative controls.

- **Parking Fee Payment.** Electronic parking fee payment systems can provide benefits to parking facility operators, simplify payment for customers, and reduce congestion at entrances and exits to parking facilities. These payment systems can be enabled by any of a variety of technologies including magnetic stripe cards, smart cards, in-vehicle transponders, or vehicle-mounted bar-codes.

- **Multi-use Payment.** Multi-use payment systems can make transit payment more convenient. Payment for bus, rail, and other public or private sector goods and services can be made using transit fare cards at terminal gates, or on check-out counters and phone booths of participating merchants located near transit stations. Multi-use systems may also incorporate the ability to pay highway tolls with the same card.

- **Congestion Pricing.** Congestion pricing, also known as value pricing, employs the use of technologies to vary the cost to use a transportation facility or network based on demand or the time of day. Pricing strategies include: variable priced lanes, variable tolls on entire roadways or roadway segments, cordon charging, area-wide charging and fast and intertwined regular (FAIR) lanes.

Traveller Information

Traveller information applications use a variety of technologies, including Internet websites, telephone hotlines, television, radio, as well as other wireless devices such as pagers and PDAs, to allow users to make more informed decisions regarding trip departures, routes, and mode of travel. The shared information could be pre-trip, en-route or tourism/special events-related. Information provided can include electronic yellow pages as well as transit and parking availability.

2.1.2 Intelligent Vehicles

Collision Avoidance Systems

To improve the ability of drivers to avoid accidents, vehicle-mounted collision warning systems (CWS) continue to be tested and deployed. These applications use a variety of sensors to monitor the vehicle's surroundings and alert the driver of conditions that could lead to a collision.

- **Intersection Collision Warning.** Intersection collision warning systems are designed to detect and warn drivers of approaching traffic at high-speed intersections.

- **Obstacle Detection.** Obstacle detection systems use vehicle-mounted sensors to detect obstructions, such as other vehicles, road debris, or animals, in a vehicle's path and alert the driver.

- **Lane Change Assistance.** Lane-change warning systems have been deployed to alert bus and truck drivers of vehicles, or obstructions, in adjacent lanes when the driver prepares to change lanes

- **Lane Departure Warning.** Lane departure warning systems warn drivers that their vehicle is unintentionally drifting out of the lane.

- **Rollover Warning.** Rollover warning systems notify drivers when they are travelling too fast for an approaching curve, given their vehicles operating characteristics.

- **Road Departure Warning.** Road departure warning systems have been tested using machine vision and other in-vehicle systems to detect and alert drivers of potentially unsafe lane-keeping practices and to keep drowsy drivers from running off the road.

- **Forward Collision Warning.** In the application area of forward-collision warning systems, microwave radar and machine vision technology help detect and avoid vehicle collisions. These systems typically use in-vehicle displays or audible alerts to warn drivers of unsafe following distances. If a driver does not properly apply brakes in a critical situation, some systems automatically assume control and apply the brakes in an attempt to avoid a collision.

- **Rear Impact Warning.** Rear-impact warning systems use radar detection to prevent accidents. A warning sign is activated on the rear of the vehicle to warn tailgating drivers of imminent danger.

Driver Assistance Systems

Numerous intelligent vehicle technologies exist to assist the driver in operating the vehicle safely. Systems are available to aid with navigation, while others, such as vision enhancement and speed control systems, are intended to facilitate safe driving during adverse conditions. Other systems assist with difficult driving tasks such as transit and commercial vehicle docking.

- **Navigation/Route Guidance.** In-vehicle navigation systems with GPS technology may reduce driver error, increase safety, and save time by improving driver's decision in unfamiliar areas

- **Driver Communication.** Integrated driver communication systems enable drivers and dispatchers to coordinate re-routing decisions on-the-fly and can also save time, money and improve productivity.

- **Vision Enhancement.** In-vehicle vision enhancement improves visibility for driving conditions involving reduced sight distance due to night driving, inadequate lighting, fog, drifting snow, or other inclement weather conditions.

- **Object Detection.** Object detection system warns the driver of an object (front, side or back) that is on the path or adjacent to the path of the vehicle. The most common application is for parking aids for passenger vehicles.

- **Adaptive Cruise Control.** Adaptive cruise control systems maintain a driver set speed without a lead vehicle, or a specified following time if there is a lead vehicle and it is travelling slower than the set speed.

- **Intelligent Speed Control.** Intelligent speed control systems limit maximum vehicle speed via a signal from the infrastructure to an equipped vehicle.

- **Lane Keeping Assistance.** Lane keeping assistance systems can make minor steering corrections if the vehicle detects an imminent lane departure without the use of a turn signal.

- **Roll Stability Control.** Roll stability control systems take corrective action, such as throttle control or braking, when sensors detect that a vehicle is in a potential rollover situation.

- **Drowsy Driver Warning.** Drowsy driver warning alerts the driver that he or she is fatigued which may lead to lane departure or road departure.

- **Precision Docking.** Precision docking systems automate precise positioning of vehicles at loading/unloading areas.

- **Coupling/Decoupling.** Intelligent cruise control, speed control, guidance and steering systems which help transit operators link multiple buses or train cars into trains each assist drivers with routine tasks that weight on driver workload.

- **On-board Monitoring.** On-board monitoring applications track and report cargo condition, safety and security, and the mechanical condition of vehicles equipped with in-vehicle diagnostics. This information can be presented to the driver immediately, transmitted off-board, or stored. In the event of a crash or near-crash, in-vehicle event data recorders can record vehicle performance data and other input from video cameras or radar sensors to improve the post-accident processing of data.

Collision Notification Systems

In an effort to improve response times and save lives, collision notification systems have been designed to detect and report the location and severity of incidents to agencies and services responsible for coordinating appropriate emergency response actions. These systems can be activated manually or automatically, with automatic collision notification (ACN). Advanced systems may transmit information on the type of crash, number of passengers, and the likelihood of injuries.

- **Mayday/ACN.** The typical Mayday/ACN application utilises location technology, wireless communication, and a third-party response centre to notify the closest Public Safety Answering Point (PSAP) for emergency response

- **Advanced ACN.** Advanced collision notification systems use in-vehicle crash sensors, GPS technology, and wireless communications systems to supply public/private call centres with crash location information, and in some cases, the number of injured passengers and the nature of their injuries.

Note that ITS includes many other systems and technologies [3], which are less relevant in respect to the problem that this thesis tackle, such as Incident Management Systems, Emergency Management Systems, Crash Prevention and Safety Systems, Roadway Operations and Maintenance Systems and Road Weather Management Systems.

2.2 Traffic control and assignment

According to Papageorgiou [75][76], most traffic control strategies fit the control loop described in figure 2.1. The basic elements are: the physical traffic network (and its model), the control devices (traffic signals, variable message signs, ramp metering, etc.), the surveillance devices (loop detectors), the demand model and the noise model (which can be detected by sensors or estimated). The key element in this loop is the control strategy, whose task is to specify the control inputs in real time, based on available measurements, estimations or predictions, in order to achieve the

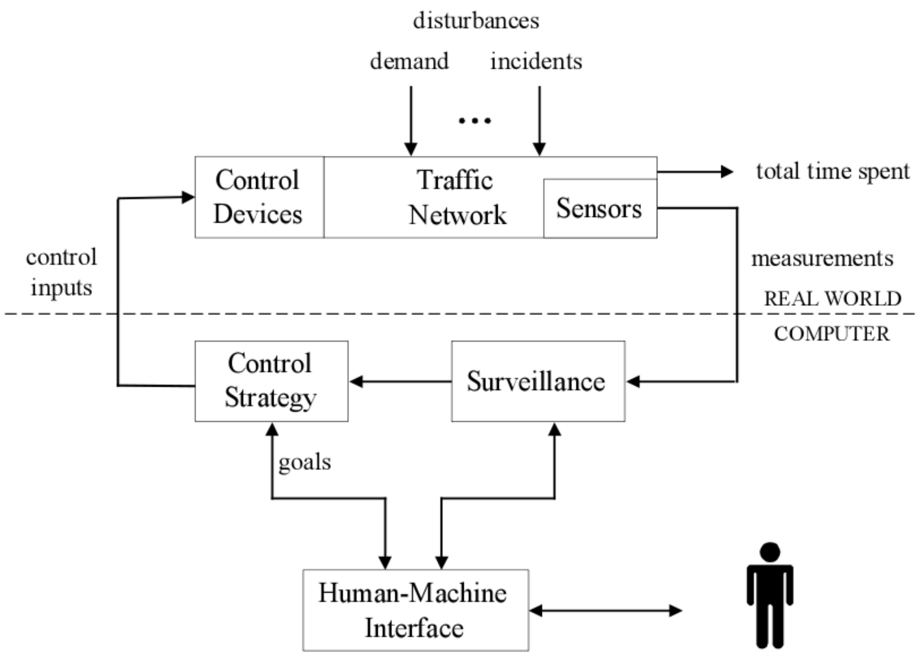


Figure 2.1: Traffic control loop [75][76]

pre-specified goals, such as the minimisation of total travel time. Techniques and methods from control theory are applied in traffic control, although only to small portions of the network (typically a single intersection), because, for big networks, real-time control seems impossible [75]. Depending on the characteristics of the traffic networks, the control problem can be classified as *road traffic control* and *freeway traffic control*.

Road traffic control

Traffic lights at intersections are the major control measure in urban road networks. An intersection consists of a number of incoming links and the crossing area. The incoming links are used by the corresponding traffic flows. Two traffic flows are compatible if they can safely cross the intersection at the same time (otherwise they are called antagonistic). A *signal cycle* is a repetition of the basic series of signal combinations at an intersection, whose overall duration is called *cycle time*. A *phase* is a part of the signal cycle, during which one set of flows can safely passes through

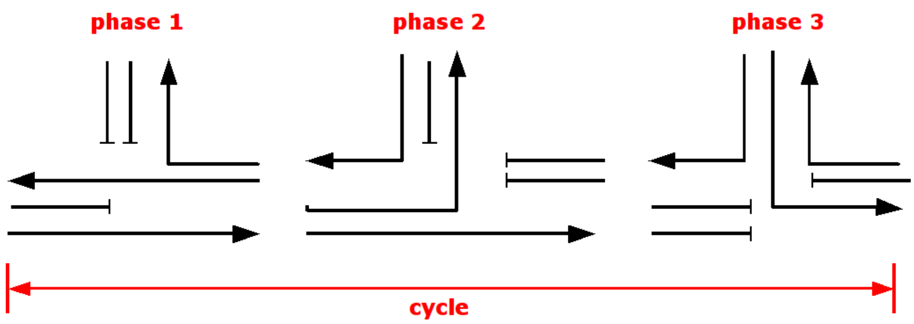


Figure 2.2: Traffic lights signal cycle

the intersection (see figure 2.2).

The operation mode (and consequently its effect on the traffic flow) of a traffic light is determined by the following three parameters:

1. Phase specification: for intersections with a complex geometry, the specification of the optimal number and structure of phases is not trivial, and it can have a great impact on the intersection's capacity and efficiency.

2. Split time: is the green duration of each phase, as a portion of the cycle time

3. Cycle time: is the duration of the whole cycle. Longer cycle times typically increase the intersection capacity because the proportion of the constant lost times (between phases) becomes smaller. On the other hand, longer cycle times may increase vehicle delays in under-saturated intersections due to longer waiting times during the red phase.

Control strategies for road traffic control may be further classified in *fixed-time* strategies and *traffic-responsive* (or actuated) strategies. Fixed-time strategies are defined for a given time of the day (e.g., morning peak hour) and they are formulated by off-line optimisation using historical data. Traffic-responsive strategies make use of real-time measurements (with inductive loops) to calculate the suitable signal settings in real time.

TRANSYT [80] is the most known and most frequently applied signal control strategy, and it is often used as a reference method to test improvements enabled by real-time strategies. For given values of the decision variables (i.e., number of splits, cycle time, etc.), the dynamic network model calculates the corresponding performance index. A heuristic, hill-climbing, optimisation algorithm introduces small changes to the decision variables and then the model is re-run, until a (local) optimum is found. Of course the main drawback of this method is that the signal plans are computed for a static situation, based on historical data. Unfortunately, demands are not constant, even within a time of the day, and may vary at different days. In other words, the signal plan is optimised for an average situation which never occurs exactly.

Among the traffic-responsive signal control strategies, SCOOT [50] is the most famous one, since it has been applied to over 150 cities in the United Kingdom and elsewhere. SCOOT uses real-time traffic volume and occupancy measurements from the upstream end of the network links, and then it runs, in a centralised control computer, an optimisation algorithm similar to TRANSYT.

Freeway traffic control

A freeway is a type of road, usually divided into at least two lanes in each direction, designed to enhance mobility through the elimination of intersections (regulated by traffic lights or stop signs). Nevertheless, the rapid increase of traffic demand led to increasingly severe congestion, during rush hours and due to incidents. Thus, the freeway network capacity is strongly underutilised on a daily basis, due to the lack of efficient traffic control systems, and ironically the nominal capacity that the network offers is not available (due to congestion) exactly at the time it is most urgently needed (during peak hours) [75].

In freeway networks, the control devices that are typically employed are:

- Ramp metering: is a device, usually a basic traffic light, that regulates the access to the freeway, according to current traffic conditions.

- Link control: it comprises a number of possibilities including lane control, variable speed limits, congestion warning, etc.

- Driver information and guidance systems: this system can be implemented either by variable message signs or via two-way communication with equipped vehicles.

Traffic assignment

Traffic assignment refers to the problem of the distribution of traffic in a network, considering demands between several locations, and the capacity of the network. Assignment methods must consider the distribution of traffic in a network as well as a set of constraints related to cost, time, and preferences of road users.

In normal situations, fixed direction signs at bifurcation nodes of the network typically indicate the direction that is time-shortest (in absence of congestion). However, the travel time on many routes changes substantially due to traffic congestion and alternative routes may become competitive. Drivers who have past experiences with the traffic conditions in a network, such as commuters, usually optimise their individual routes, thus leading to the user-equilibrium condition, first formulated by Wardrop [106].

However, demand may change in a non-predictable way, due to changing environmental conditions, exceptional events or accidents. This may lead to an underutilisation of the overall network's capacity, whereby some links are heavily congested while capacity reserves are available on alternative routes. Given the topological constraints, it is not possible to change the supply in a way which is flexible enough to match the demand. Therefore, several kinds of traffic management systems, involving both information broadcast as well as control and optimisation, must be employed. For example, route guidance and driver information systems (RGDIS) may be employed to improve the network efficiency via direct or indirect recommendation of alternative routes. These communication devices may be consulted by a potential road user to make a rational decision regarding whether or not to carry out (or postpone) the intended trip, the choice of transport mode (car, bus, underground, etc.), the departure time selection and the route choice. Although radio broadcasting services and variable message signs have been in use for a long time, individual route guidance systems, using two-way communication between vehicles and infrastructure and control centres, gain an increasing interest.

2.3 Economic models for resource allocation

Resource management refers to the activity of establishing a mutual agreement between a resource producer and a resource consumer. This agreement specifies that the provider must supply a resource that can be used by the consumer to perform some tasks. Conventional resource management techniques are based on relatively static and centralised models. One way to cope with the dynamism and the decentralisation of certain complex environments is using market-based approaches, which introduce money and pricing as the technique for coordination between consumers and producers of resources. In this way decentralisation is provided by distributing the decision-making process across users and resource suppliers.

There are many economic models to allocate resources among competing agents (see figure 2.3). Commodity markets usually perform allocations of resources by means of reaching some sort of equilibrium price. This can be done by bargaining [70], in which the consumer and the supplier of the resource dispute the price which will be paid and, eventually, come to an agreement, or by trading, in which several suppliers, in competition with each other, set the prices to their resources to acquire as many consumers as possible.

In auction markets, we can have two main classes of auctions: one-to-many or many-to-many. In one-to-many auctions, one agent (the auctioneer) initiates the auction protocol and several other agents (the bidders) place their bids. The English auction, the Dutch auction and the Vickrey auction belong to this class. In many-to-many auctions, several agents participate in the auction protocol as suppliers and several other agents participate as bidders. The double auction is the most used auction protocol for many-to-many auction. In a double auction potential buyers submit their bids and potential sellers simultaneously submit their ask prices to an auctioneer. The auctioneer chooses some price p that clears the market, and all the suppliers who asked for less than p sell and all the bidders who bid more than p buy

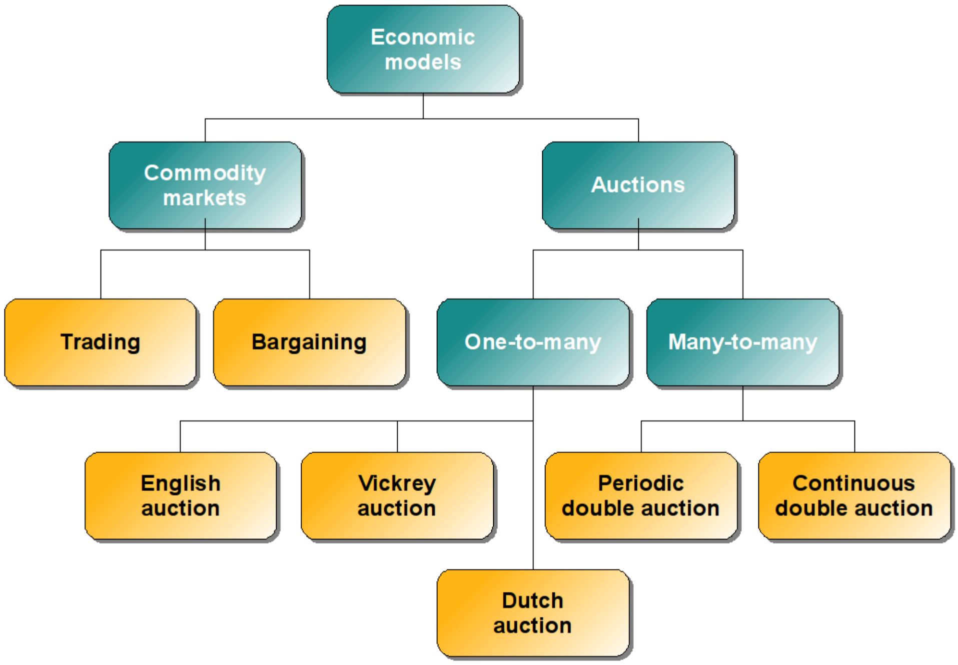


Figure 2.3: Economic models for resource allocation

at the price p. There are two types of double auctions: continuous double auctions clear the market and match buyers and sellers as soon as a match is detected. A periodic double auction instead collects bids over a specific interval of time and then clears the market.

The properties of the aforementioned economic models are studied by a discipline at the intersection between economics and game theory, called mechanism design [30][39][51][102]. Mechanism design can be viewed as reverse engineering of games or, equivalently, as the art of designing the rules of a game to achieve a specific desired outcome. The main focus of mechanism design is to design institutions or protocols that satisfy certain desired objectives, assuming that the individual agents, interacting through the institution, will act strategically and may hold private information that is relevant to the decision at hand.

Social choice function

Suppose that there are n agents, $i \in [1, n]$. Each agent i privately observes its preferences over a set $\mathcal{X}$, called the outcome set, which contains all the possible outcomes that a social planner may choose from. The private preference of agent i is condensed in the agent type $\theta_i \in \Theta_i$.

Given the agent type θ_i and an actual outcome $x \in \mathcal{X}$, each agent can evaluate "how good" this outcome is, according to a utility function $u_i : \mathcal{X} \times \Theta_i \to \mathbb{R}$. Each agent is modelled as rational and intelligent, i.e., it tries to maximise its utility u_i.

The agents types $\theta = [\theta_i, \ldots, \theta_n]$ are drawn according to a probability distribution $\phi \in \mathcal{P}(\Theta)$, where $\mathcal{P}(\Theta)$ is the set of all the probability distributions over the set $\Theta = \Theta_1 \times \cdots \times \Theta_n$. The probability distribution ϕ, the type sets Θ_1,...,Θ_n, and the utility functions u_is are common knowledge among the agents (still, each actual type θ_i is known only by the agent i).

In this setting, the social planner faces the problem of mapping each possible profile of the agents' types $\theta = [\theta_1, \ldots, \theta_n] \in \Theta = \Theta_1 \times \cdots \times \Theta_n$ to a collective choice (or outcome) $x \in \mathcal{X}$. This mapping is defined by a function $f : \Theta \to \mathcal{X}$, called *social choice function* (SCF) [67].

Example: allocation of a single unit of an indivisible good. Consider a set of n agents. One of them owns one unit of an indivisible good and is willing to trade this good by means of money. Suppose that this trade is mediated by an independent broker (i.e., the social planner). The problem for the broker is deciding which agent to allocate the good to and how much money each agent must pay (or receive).

Outcome set $\mathcal{X}$: the outcome set $\mathcal{X}$ is the set of vectors $x = [y_1 \ldots y_n \; t_1 \ldots t_n]$, where $y_i = 1$ if the agent i receives the good, $y_i = 0$ otherwise, and t_i is the monetary transfer received by the agent i (i.e., if $t_i < 0$, agent i pays t_i, if $t_i > 0$, agent i receives t_i. The set of feasible alternatives is then

$$\mathcal{X} = \left\{ [y_1 \ldots y_n \; t_1 \ldots t_n] \mid y_i \in \{0, 1\}, \; t_i \in \mathbb{R} \; \forall i, \; \sum_{i=1}^{n} y_i = 1, \; \sum_{i=1}^{n} t_i \leq 0 \right\}$$

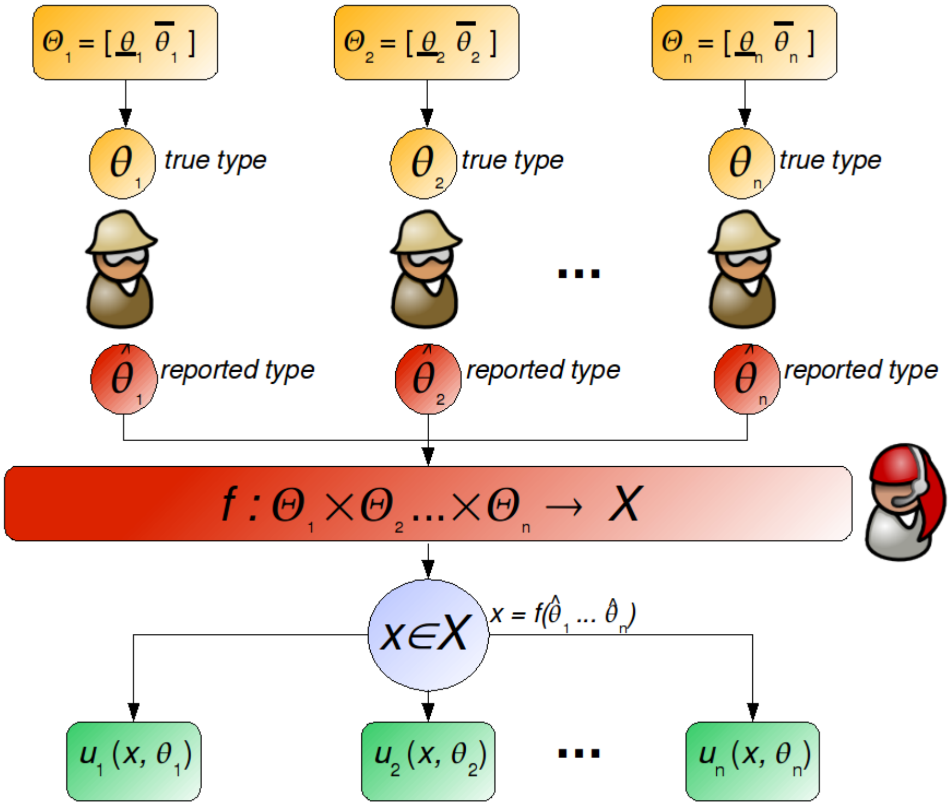


Figure 2.4: Social choice function

Type set Θ_i**:** in this example, the type θ_i of an agent can be viewed as its valuation of the good. We can take the set of possible valuations for agent i to be $\Theta_i = [\underline{\theta_i}, \overline{\theta_i}] \subset \mathbb{R}$, where $\underline{\theta_i}$ is the lowest valuation and $\overline{\theta_i}$ is the highest valuation the agent may have for the item.

Utility function $u(\cdot)$**:** the utility function of agent i can be given by

$$u_i(x, \theta_i) = u_i([y_1 \dots y_n \; t_1 \dots t_n \; \theta_i]) = \theta_i \cdot y_i + t_i$$

In other words, the utility is 0 if the agent does not get the item, otherwise it is equal to the difference between its private valuation, θ_i, and the money it paid for the good, t_i.

Social choice function $f(\cdot)$: The general structure of the social choice function for this case is

$$f(\theta) = [y_1(\theta) \dots y_n(\theta)\ t_1(\theta) \dots t_n(\theta)] \quad \forall \theta \in \Theta$$

Mechanisms

Given the model described in section 2.3, the social planner faces two problems:

1. **Preference Aggregation Problem**: for a given type profile $\theta = [\theta_1 \dots \theta_n]$ of the agents, which outcome $x \in \mathcal{X}$ should be chosen?

2. **Information Elicitation Problem**: how do we extract the agent's true type θ_i, which is private information of agent i?

A mechanism $\mathcal{M} = (\{S_i\}_{i \in [1,n]}, g(\cdot))$ is a collection of action sets $\{S_1, \dots, S_n\}$ and an outcome function $g : S \to \mathcal{X}$, where $S = S_1 \times \dots \times S_n$.

The set S_i for each agent i describes the set of actions available to that agent. Based on its actual type θ_i, each agent i will choose some action, say $s_i \in S_i$. Once all the agents have chosen their actions, the social planner uses this profile of the actions $s = [s_1 \dots s_n]$ to pick a social outcome $x = g(s)$.

The trivial scheme of asking the agents to reveal their types becomes a special case, called a *direct revelation mechanism* (DRM). Formally, $\mathcal{D} = (\{S_i\}_{i \in [1,n]}, g(\cdot))$ where $S_i \equiv \Theta_i\ \forall i$ and $g(\cdot) \equiv f(\cdot)$.

After discovering the mechanism $\mathcal{M}$ chosen by the social planner, each agent i starts making an analysis regarding which action s_i will result in the most favourable outcome for agent i, and comes up with a strategy $s_i : \Theta_i \to S_i$. This phenomenon leads to a game among the agents. A mechanism $\mathcal{M}$ combined with possible types of the agents $\Theta_1, \dots, \Theta_n$, probability density ϕ, and utility functions $u_1(\cdot), \dots, u_n(\cdot)$ defines a game of incomplete information. Given that, the social planner now worries about whether or not the outcome of the game matches the outcome of the social choice function, that is if the mechanism implements the social function.

A mechanism $\mathcal{M}$ implements the social choice function $f(\cdot)$ if there is a pure strategy equilibrium $s(\cdot) = [s_1^*(\cdot) \dots s_n^*(\cdot)]$ of the game induced by $\mathcal{M}$ such that $g([s_1^*(\theta_1) \dots s_n^*(\theta_n)]) \equiv f(\theta_1 \dots \theta_n)\ \forall [\theta_1 \dots \theta_n] \in \Theta$.

Properties of a SCF

We have seen that a mechanism provides a solution to both the problem of information elicitation and the problem of preferences aggregation if it can implement the desired social choice function $f(\cdot)$. However, not all the SCFs are implementable. Thus it is important to know which social choice function a social planner would ideally prefer to be implemented. Note that a social planner would always like to use a SCF which satisfies as many desirable properties from the perspective of fairness as possible. The properties of a SCF which a social planner would ideally wish the SCF to have are: *ex-post efficiency*, *non-dictatorialship* and *incentive compatibility*. Ex-post efficiency means that the SCF always selects an outcome on the Pareto frontier, non-dictatorialship means that no agent is a dictator, i.e., it always gets the highest utility from the outcomes selected by the SCF, while incentive compatibility means that revealing the true type θ_i is an equilibrium strategy for all the agents.

Given the properties described above, one would define a SCF that is ex-post efficient, non-dictatorial and dominant strategy incentive compatible. Unfortunately, according to the Gibbard-Satterthwaite impossibility theorem [42][87], such a SCF does not exist. Still, there is a special and much studied class of environments, called quasi-linear environments, where the Gibbard-Satterthwaite theorem does not hold true. In a quasi-linear environment, an alternative $x \in \mathcal{X}$ is a vector of the form $x = [k\ t_1 \dots t_n]$, where k is an element of a closed and bounded set $\mathcal{K}$ and $t_i \in \mathbb{R}$ is the monetary transfer. If $t_i > 0$, then agent i will receive the money and if $t_i < 0$, then agent i will pay the money. The system is assumed to be closed and that the n agents have no outside source of funding, i.e., $\sum_{i=1}^{N} t_i \leq 0$.

A social choice function in this quasi-linear environment takes the form $f(\theta) = [k(\theta)\ t_1(\theta) \dots t_n(\theta)]$, while the agent i's utility function takes the form $u_i(x, \theta_i) = v_i(k, \theta_i) + m_i + t_i$, where $v_i(k, \theta_i)$ is the agent i's valuation of choice k and m_i is

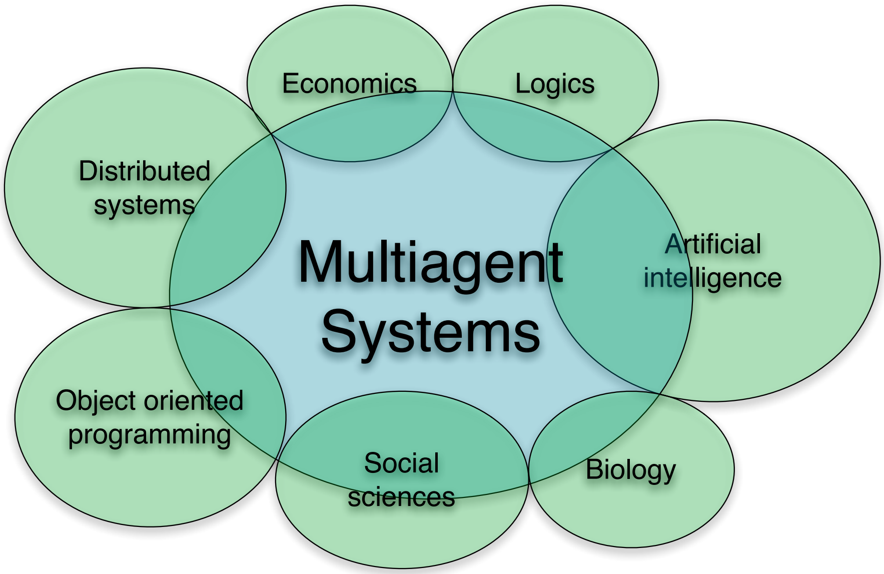


Figure 2.5: Multiagent systems

the initial endowment. This kind of social choice function is characterised by two important properties, *allocative efficiency* and *budget balance*. An SCF is allocative efficient if for every agents profile it selects the choice k that is most valued by all the agents, while is budget balanced if the sum of the payments is zero.

According to [39], all the SCFs in quasi-linear environment are non-dictatorial. Furthermore, allocative efficiency and budget balance imply ex-post efficiency, so that in these kind of environments a social planner can focus only on choosing a SCF that is incentive compatible.

2.4 Multiagent systems

The modern approach to artificial intelligence is based on the concept of agent. An *agent* is an entity that can perceive its *environment* through *sensors* and act upon that environment through *actuators* [85]. An agent is said to be *rational* if it always tries to optimise an appropriate performance measure or utility function. Very rarely are agents isolated systems. In many situations agents coexist and interact with each other, so that the system as a whole can be considered as a *multiagent system* (MAS). Examples of such systems include software agents on the Internet, robotic soccer agents or e-commerce agents.

2.4.1 Elements of a multiagent system

Agent design

If the agents of the system have been designed by different designers or they have different capabilities, such agents are said to be *heterogeneous*. Agent heterogeneity can affect all functional aspects of an agent, from perception to decision making. On the contrary, agents that are designed in an identical way and have *a priori* the same capabilities are said to be *homogeneous*. This distinction is not always clear, because even agents with the same capabilities that implement different behaviours can also be viewed heterogeneous.

Agent rationality

An autonomous agent must face the problem of optimal decision making, that is, choosing the best possible action in every situation, given what it knows about the world around it. An agent is said to be *rational* if it always selects an action that optimises an appropriate performance measure, given what the agent knows so far. The performance measure is typically defined by the designer of the agent and reflects what the user expects from the agent in the task at hand.

Environment

Agents exists in an environment that can be either *static* or *dynamic*. Static environments are easier to handle and allow for a more rigorous mathematical formulation. Unfortunately, in a MAS, the mere presence of multiple agents makes the environment appear dynamic from the point of view of a single agent. This can often be problematic, for instance in the case of learning agents, where non-stable behaviours (co-learning) may arise.

Perception

The agent observes data that are spatially, temporally and semantically distributed, so that, in general, a single agent cannot access the full environment state. This fact has various consequences in the decision making of the agents, because it must determine which action has to be executed on the basis of a conditional probability distribution over possible environment states.

Control

The control in a MAS is typically *decentralised*, which means that there is no central entity that gathers all the available information from each agent and then decides what action each agent should take. The decision making of each agent is usually autonomous, so that, in competitive settings, game-theoretic issues arise. Furthermore, if the agents are cooperative, problems of coordination arise, to ensure that the individual decisions of the agents result in good joint decisions for the group.

Knowledge

In a MAS, the knowledge that is available to each agent about the current environment state can differ substantially. For example, in a cooperative setting each agent may know the available action set of the other agents, all agents may know (by communication) their current perceptions, or they can infer the intentions of each other based on some shared prior knowledge. On the other hand, in a competitive setting an agent is typically unaware of the action set and current perceptions of the competing agents. Nevertheless, in a MAS each agent should also consider the knowledge of each other agent in its decision making.

Communication

Direct interaction is often associated with some form of communication. Communication can be used, for instance, for coordination among cooperative agents or for negotiation among competitive agents. MASs are sometimes characterised by indirect interaction, such in biological-inspired or stigmergic systems [12]. Communication raises issues related with protocols and languages to understand each other.

2.4.2 Coordination

The problem of *coordination* is the effort of governing the space of interaction of a MAS [19]. In the case of a MAS with a common objective, coordination refers to the activity of a group of agents that must find individual actions that result in good joint decisions for the group. A distinguishing feature of a multiagent system is the fact that the decision making of the agents can be distributed. This means that there is not a central controller agent that decides what each agent must do at each time-step, but each agent is autonomous and responsible for its own decisions. Such a decentralised approach has the advantage of being efficient, due to the asynchronous and parallel computation, and robust, in the sense that the functionality of the whole system does not rely on a single agent. Nevertheless, in order for the agents to be able to take their actions in a distributed fashion, appropriate coordination mechanisms must be additionally developed. This is particularly true in the case of cooperative agents that form a team, and through this team they make joint plans and pursue common goals. In this case, coordination is needed to ensure that the agents do not obstruct each other when taking actions, and moreover that these actions serve the common goal. Coordination can also appear as an emergent phenomenon in a population of competing and adaptive agents [17].

Several approaches that tackle this problem can be found in the literature, shaping the interaction space either directly, by making assumptions on agent behaviours and knowledge. For example, social conventions (or social laws) are used to constrain the space of possible actions of the agents. Given that the convention has been established and is common knowledge among agents, no agent can benefit from not respecting it. For example, in [18], a general convention that achieves coordination in a large class of systems is proposed. The convention assumes a unique ordering scheme of joint actions that is common knowledge among agents. In a particular situation, each agent first computes the optimal joint actions, and then selects the first joint action according to this ordering scheme.

Coordination laws can be also explicitly defined and embedded into a coordination medium, such as a tuple centre, whose behaviour can be programmed by defining reactions to the basic communication events [33]. Coordination by social conventions relies on the assumption that an agent can compute all the optimal joint actions before choosing a single one. However, computing these joint actions can be expensive when the action sets of the agents are large. Therefore one would like to reduce the size of the action sets first, so that the joint action selection is simplified. A natural way to reduce the action sets is by assigning roles to the agents. In practical terms, if an agent is assigned a role at a particular state, then some of the agents actions are deactivated at this state.

From the point of view of an individual agent, the problem of coordination essentially boils down to finding the sequence of actions that best achieves its goals. This is a not trivial problem even in cooperative systems, because often each agent has only partial knowledge of its environment and uncertainty about the effect of a specific action. In this case, a solution to the coordination problem is *learning* to select the best joint action that yields to the highest utility for the entire group of agents.

2.4.3 Learning

In this section we describe the basic concepts, methods, notations and models for sequential decision making in stochastic domains for both single-agent and multiagent systems, with particular emphasis on cooperative multiagent reinforcement learning.

Sequential decision-making refers to the problem of an agent that repeatedly interacts with its environment with the aim of optimising a performance metric based on the rewards that it receives. Sequential decision-making is quite a difficult problem for an agent, because actions may not be reversible or may have long-term effects, the environment may not be completely observable or it may be non-stationary.

If the current situation of a sequential decision-making problem provides a complete description of the past, and previous information is not relevant for making a

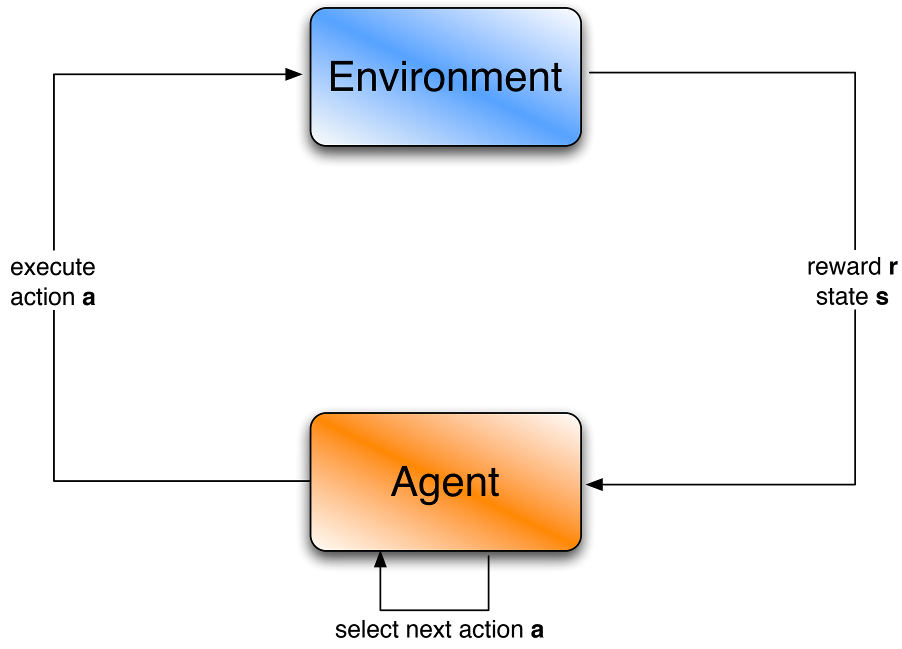


Figure 2.6: Single-agent decision-making loop

decision, we say that the problem obeys the *Markov property*.

The sequential decision-making problem may be *single-agent* or *multiagent*. In single-agent problems, the decision maker is unique and the environment only responds to the decision maker's actions. In multiagent problems, there are several agents that interact with the environment and in general the environment responds to the joint actions of the group of agents. Having multiple agents interacting with the environment and with each other causes severe consequences on the complexity of the problem. In this chapter we focus on *cooperative* multiagent systems in which the group of agents collaborates to make a collective decision and to achieve a common goal.

Single-agent models

As stated earlier, a sequential decision-making problem refers to the problem of an agent that repeatedly interacts with its environment with the aim of optimising a performance metric based on the rewards that it receives. Usually the interaction with the environment consists of the following steps:

1. Environment observation: the agent observes the environment through its sensors and perceives an observed state s. In general, the observed state s is not the "true" state of the environment, because the agent's sensors may be noisy or because different states may be considered as the same observed state (aliasing). Nevertheless, the state s is the only information about the environment that is relevant for the decision maker.

2. Action selection: based on the current perceived state s, the agent selects the action that at that time is the "best" action that it may take, according to its policy and its knowledge.

3. State transition: according to a stochastic transition model, which is a function of the current state and the selected action, the environment transits to a new state s'

4. Reward observation: after having transited to state s', the environment provides the agent with a feedback to evaluate the new situation, usually a real scalar reward value.

Figure 2.6 shows the general structure of an agent interacting with its environment. This process is broken down into three phases: sensing, deliberating, and acting. The key issue of the decision-making problem is of course the deliberation phase, in which the intelligence of the agent is implemented.

Environment. The environment is called *stationary* if the transition probability of moving from state s to state s' after executing action a does not change with time. That is, the action executed by an agent always has the same probabilistic effect on the environment. An environment is called *non-stationary* when the transition probabilities change over time. In this section we focus on stationary environments.

The decision maker may eventually have at its disposal a *model* of the environment, which mimics the behaviour of the environment. In this case, the decision maker can use this model for *planning*, considering possible future situations before they are actually experienced. On the other hand, if such model is not available, the decision maker can only *learn* by interacting with the environment (trial and error).

Rewards. The goal of the decision maker is to select actions that optimise the expected return R_t. If the sequential decision-making can be divided into episodes of finite length, the expected return is defined as:

$$R_t = r_t + r_{t+1} + r_{t+2} + \cdots + r_T = \sum_{k=t}^{T} r_k \tag{2.1}$$

After T steps, or when a goal state is reached, the episode ends and the system resets to a starting state.

In continuous sequential decision-making, there are no goal states and the system "lives" indefinitely. In this case the expected return is defined as:

$$R_t = r_t + \gamma \cdot r_{t+1} + \gamma^2 \cdot r_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \tag{2.2}$$

where $\gamma \in [0, 1)$ is the discount rate. If $\gamma = 0$, the agent is "myopic" and it tries to maximise only the immediate received reward. If $\gamma > 0$, the rewards received in the near future are considered more valuable than the rewards received later.

Solution techniques. Given the environment state s, the agent must select an action following its policy π. A deterministic policy is a function that maps the state s to a single action a. A stochastic policy is a function that maps the current state s to a probability distribution over all possible actions.

Optimising the expected return R_t is equivalent to computing an optimal policy π^*, which for every possible state s returns the action a that maximises the performance measure. If a model of the environment is not available, the most common technique to find an optimal policy is *reinforcement learning*.

Q-learning is the most widely used reinforcement learning technique [108]. Using Q-learning, the decision maker represents its policy π in terms of a special action-value function, $Q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, which for a given state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$ returns the expected future discounted reward that can be obtained. Thus, the objective of the decision maker is to learn the optimal action-value function, $Q^*(s,a)$.

Before learning has started, the action-value function Q returns an initial estimate, chosen by the designer of the learning agent. Then the agent enters the main loop depicted in figure 2.6. It observes the environment state s, it executes an action a, and finally it observes the environment next state s' and the reward r. The agent uses these quantities to update the current action-value function, using the formula:

$$Q(s,a) \leftarrow \overbrace{Q(s,a)}^{\substack{old \\ estimate}} \;+\; \alpha \cdot [\overbrace{r \;+\; \gamma \cdot max_{b \in \mathcal{A}} Q(s',b)}^{\text{expected discounted reward}} - \overbrace{Q(s,a)}^{\substack{old \\ estimate}}\,] \tag{2.3}$$

where γ is the *discount rate*, and $\alpha \in (0,1]$ is the *learning rate*, which controls the contribution of the new experience to the current estimate.

At each time-step, given the state s, the learning agent selects its next action on the basis of the actual estimation of the action-value function. The simplest action selection rule is to select the action with the highest estimated action value, $argmax_{a \in \mathcal{A}} Q(s,a)$. This greedy strategy always exploits current knowledge and does not spend time selecting apparently inferior actions to see if they might really be better.

A simple alternative is to behave greedily most of the time, but, with small probability ϵ, selecting an action uniformly at random. This near-greedy rule is called ϵ-greedy action selection. This ensures that all actions, and their effects, are experienced, guaranteeing that the action-value function Q converge to the optimal Q^* [109]. Although ϵ-greedy action selection is an effective technique of balancing exploration and exploitation, one drawback is that when it explores it chooses equally among all actions. This means that it is as likely to choose the worst-appearing action as well as the next-to-best action. The obvious solution is selecting action a with a probability proportional to the estimated action value. The greedy action is still given the highest selection probability, but all the others are ranked and weighted according to their value estimates. This rule is called soft-max action selection. The most common

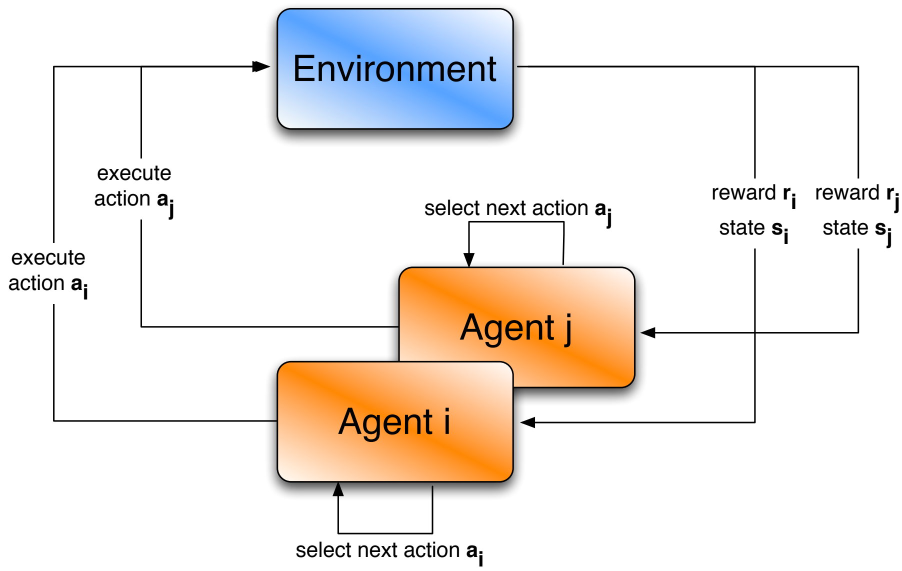


Figure 2.7: MAS decision-making loop

soft-max method uses a Gibbs, or Boltzmann, distribution, where the probability of choosing action a when in state s is given by:

$$P(a) = \frac{e^{Q(s,a)/\tau}}{\sum_{b \in \mathcal{A}} e^{Q(s,b)/\tau}} \tag{2.4}$$

where τ is a positive parameter called temperature. High temperatures cause the actions to be all (nearly) equiprobable (exploration). Low temperatures cause a greater difference in selection probability for actions that differ in their value estimates (exploitation). The main drawback is that is not easy to define high or low temperatures in absolute terms, since τ is related to the order of magnitude of $Q(s, a)$, which may not be known *a priori*.

Multiagent models

If single-agent models were characterised by a unique decision maker interacting with the environment, in a multiagent system (MAS) multiple agents are all executing actions and influencing their environment [95][110][115].

Although each agent observes its environment and selects actions individually and autonomously, it is the resulting *joint action* which affects the environment and produces the perceived reward (figure 2.7).

The multiagent setting has severe consequences on the complexity of the decision making problem. The action space and the state space grow significantly in the multiagent case. For instance, if the MAS is composed of N agents, each of them with M actions to choose from, the size of the joint action space is M^N (i.e., it scales exponentially with the number of agents).

Environment. In the multiagent case the environment is no longer stationary, because the state transition probability as well as the perceived rewards depend on the actions selected by the other agents. Dynamic environments are more challenging than stationary environments since the same action can have very different effects. Thus, tracking the best action to execute in every state becomes more difficult.

Furthermore, the problem of observing and identifying the current environment state arises. Different agents may observe different parts of the environment, or the same environment state may be mapped to different perceived states. In [79] are defined four models of observability:

1. *Individual observability.* Every agent observes the same, complete environment state.

2. *Collective observability.* Every agent observes a part of the environment state, and the complete environment state is given by the combined observations of all agents.

3. *Collective partial observability.* Every agent observes a part of the environment state, but there are no assumptions about the combined observations of the agents, which may or not coincide with the complete environment state.

4. *Non-observability.* The agents do not observe any aspect of the environment.

Agent structure. Agents in a MAS may be either *homogeneous* or *heterogeneous*. Homogeneous agents are usually constructed by the same designer, and they have identical capabilities and goals. On the other hand, heterogeneous agents have different capabilities and different goals, either because they are constructed by different designers or because they fulfil different roles within the system.

Another important issue to consider is whether the different agents are *cooperative* or *competitive*. Cooperative agents aim at solving the same decision-making problem and they are willing to help each other to achieve this goal. On the other hand, the agents may be selfish and only consider their own goals when acting. In the extreme, the agents may be involved in a zero-sum situation so that they must actively oppose other agents' goals in order to achieve their own. This last case describes competitive agent systems.

In this chapter we focus on cooperative multiagent learning systems. Cooperative MAS are usually composed of homogeneous agents. Nevertheless, it could be possible to have cooperative MASs of heterogeneous agents, as in robotic soccer. In this case, the cooperation resides in the global reward function that the team of agents aims at maximising.

Communication. The agent may improve their performance by sharing knowledge, information and experiences [96], which can help agents with similar tasks to learn faster and better. Agents may communicate in different ways, such as *broadcasting* a message to all agents at once, directly sending a message to a specific agent using *direct communication*, or using *blackboard communication* [110].

Rewards. In single-agent reinforcement learning, the goal of the agent is formalised in terms of a special *reward function*. The reward function is the way to communicate to the learning-agent *what* we, as agent designer, want it to achieve. It assigns a real value to the last action executed by the agent, informing it about how good that particular action was with respect to the goal to achieve.

When dealing with multiple cooperative learners, the designer is faced with the task of dividing the reward of a joint action among the learners. This problem is called the *credit assignment problem*. Hence, in multiagent cooperative reinforcement learning we can distinguish two levels of reward functions: the *global reward* and the *agent reward*. The *global reward* is a signal that rates the usefulness of a full *joint action* with respect to the global goal that the collective of learning agents pursues. On the other hand, the *agent reward* is the signal that aims at rating the *individual agent action*, i.e., the contribution of the individual agent to the global reward.

The simplest solution is to assign the whole global reward to each learner. In this way, whenever a learner's reward increases (resp. decreases), all learners' rewards increase (resp. decrease). Albeit simple, such an approach may not scale well to increasingly difficult problems because the learners do not have sufficient feedback tailored to their own specific actions [114]. In fact, the global reward is a "noisy" signal, which may lead the learners to make wrong decisions.

The complementary solution is to reward each agent with its individual local reward. Such rewards may lead to faster learning rates, but not necessarily to better results (compared to rewarding the learners with the entire global reward [7]). It may happen that agents do not have any rational incentive to help other agents, and greedy behaviours may develop. Still, local rewards may speed up the learning process, reducing the number of examples necessary for learning [5]. Other approaches aim at providing the learning agents with more informative and less noisy signals, using a Kalman filter to compute the true contribution to the global reward [20].

In general there is no general and definitive approach to dealing with the complex problem of designing reward functions for collectives of agents. Still the theory of COllective INtelligence (COIN) [114] gives some guidelines as to the design of agent reward functions. In fact, an agent reward function should show *factoredness* and *learnability*: an agent reward is meant to be factored if it is aligned with the global reward (i.e., if the agent reward increases, the global reward does the same); furthermore, an agent reward should enable the agent to distinguish its contribution to the global reward from that of the other agents. For example, if we rate the agent action with the global reward, the agent reward function is trivially aligned, but is poorly learnable. In fact, if an agent takes an action that actually improves the global reward, while all the other agents take actions that worsen the global reward, the agent wrongly believes that its action was bad. Purely local agent reward functions are usually less noisy and so highly learnable, but may not be necessarily aligned with the global reward.

In [114] Wolpert and Tumer defined a fully factored and highly learnable agent reward function, called the *Wonderful Life Reward* (WLR), defined as follows:

$$WLR_j(x) = G(x) - G(x \mid x_j \leftarrow c_j) \tag{2.5}$$

where x if the joint action, $G(x)$ is the global reward derived from such joint action, and $G(x \mid x_j \leftarrow c_j)$ is the global reward evaluated under the joint action where all the components of x affected by agent j are replaced by a constant factor c_j. If this constant is the null action, the WLR is equivalent to the global reward minus the global reward that would have arisen if the agent j had been removed from the system. Unfortunately the counterfactual term $G(x \mid x_j \leftarrow c_j)$ is not always possible to compute, specially if the function G does not have a know functional form.

Solution techniques. As in the single-agent case, the goal of a cooperative multi-agent learning system is finding an optimal *joint policy* π^*, which for every possible state $\mathbf{s}$ returns the *joint action* $\mathbf{a}$ that maximises the performance measure, that is, the global reward function. Besides the similarities with the single-agent case, difficulties arise due to the decentralised nature of the problem. Each agent receives observations and selects actions individually, but it is the resulting joint action that influences the environment and generates the reward.

Besides the challenges inherited from single-agent learning (action space and state space dimension, the exploration-exploitation trade-off), several new challenges arise in the multiagent scenario, such as the difficulty of specifying a learning goal [92] and the non-stationarity of the learning problem.

Given the complexity of the decision-making problem in the multiagent setting, many techniques are available, which differ in the way the learning process is performed by each agent, and which knowledge is available to each agent etc. Agents may learn independently of the others, they could learn to solve a part of the whole learning task, or they can interact in a cooperative, negotiated search for a solution of the learning task [111].

If the reward signal is the unique information that an agent has at its disposal, the learning agents are called *independent learners*. Independent learners are unaware of the existence of other agents, and only perceive the reward that is associated with each joint action. Thus, independent learners must estimate the value of their individual actions based solely on the rewards that they receive for their actions. On the other hand, the learning-agents may be made aware of the actions of other agents, not only the reward. In this case, the learning agents are called *joint-action learners*. Since a joint-action learner can also perceive the actions of the others, it can maintain a model of the strategy of other agents and choose its actions based on the other participants' perceived strategy.

Joint-action learners may potentially make more informed decision, since they somewhat predict what the other agents will do and behave accordingly. If each agent is aware of the other agents and knows the entire joint action, it can model the other agents strategy, for instance estimating the probability for agent j of executing action a_j when in state $\mathbf{s}$:

$$P^i_j(\mathbf{s}, a_j) = \frac{c(a_j)}{\sum\limits_{b_j \in \mathcal{A}_j} c(b_j)} \tag{2.6}$$

where $c(a_j)$ counts the number of times agent i observed agent j taking action when in state $\mathbf{s}$. In [24] several heuristics are proposed to increase the learner's Q-values for the actions with a high likelihood of getting good rewards. In very large domains with hundreds of agents, modelling all the other agents it is not computationally viable. Another issue that may arise with joint-action learners is co-adaptation. Since each agent model the others, the action selection is influenced by the perceived strategy of the other agents, resulting in co-adaptation among the concurrent learning agents. Co-adaptation can drive the team towards suboptimal solutions because agents tend to select those actions that are rewarded better, without any consideration for how such actions may affect the search of their teammates. To counter balance this effect, agents should prefer actions that *inform* their teammates about the structure of the joint search space in order to help them choose among various available actions [74].

With the Team Q-learning algorithm [65], all the agents learn the same common action-value function in parallel using the formula:

$$\mathbf{Q}(\mathbf{s},\mathbf{a}) \leftarrow \mathbf{Q}(\mathbf{s},\mathbf{a}) + \alpha \cdot [r(\mathbf{s},\mathbf{a}) + \gamma \cdot max_{\mathbf{b}\in\mathcal{A}}\mathbf{Q}(\mathbf{s}',\mathbf{b}) - \mathbf{Q}(\mathbf{s},\mathbf{a})] \tag{2.7}$$

where $\mathcal{A}$ is the joint action space, $\mathcal{A} = A_1 \times \cdots \times A_N$. The Team Q-learning algorithm assumes that each agent perceives the same full environment state $\mathbf{s}$ and is aware of the entire joint action $\mathbf{a}$ that the team of agents executes at each time-step. Furthermore it assumes that the optimal joint actions are unique (which is rarely the case), so as any coordination to break ties is unnecessary.

Similar to the Team Q-learning algorithm, the Distributed Q-learning algorithm [62] solves the cooperative learning task without coordination. Each agent maintains a local policy $\pi_i(\mathbf{s})$ and a local action-value function $Q_i(\mathbf{s}, a_i)$, which depends exclusively on its own action. The local Q-values are updated only when the update leads to an increase in the Q-value:

$$Q_i(\mathbf{s}, a_i) \leftarrow max[\ Q_i(\mathbf{s}, a_i),\quad r(\mathbf{s},\mathbf{a}) + \gamma \cdot max_{b\in\mathcal{A}_i} Q_i(\mathbf{s}', b)\ ] \tag{2.8}$$

If the update leads to an improvement in the Q-values, the local policy $\pi_i(\mathbf{s})$ is updated to the action a_i which has improved the Q-value.

With the Independent Q-learning algorithm [24], each agent stores and updates an individual action-value function Q_i and the global action-value function $\mathbf{Q}$ is defined as a linear combination of all individual contributions, $\mathbf{Q}(\mathbf{s},\mathbf{a}) = \sum_{i=1}^{N} Q_i(\mathbf{s}, a_i)$. Each local action-value function Q_i is updated independently of the other agents using the formula:

$$Q_i(\mathbf{s}, a_i) \leftarrow Q_i(\mathbf{s}, a_i) + \alpha \cdot [r_i(\mathbf{s},\mathbf{a}) + \gamma \cdot max_{b_i\in\mathcal{A}_\rangle} Q_i(\mathbf{s}', b_i) - Q_i(\mathbf{s}, a_i)] \tag{2.9}$$

Although the learning agents are independent, they are coupled by the reward function $r_i(\mathbf{s},\mathbf{a})$, which depends on the entire joint action $\mathbf{a}$. Furthermore, the agents need to perceive the same full environment state **s** (collective observability) to successfully update the local action-value function. The standard convergence proof for Q-learning does not hold anymore, because the environment becomes non-stationary from the perspective of the individual agent. Still, Q-learning is resilient to moderate non-stationarity and this method has been applied successfully in multiple cases [90][96].

All the solution methods for multiagent cooperative learning presented here assume the exact and unique observation of the environment state, which in many domains would be a strong assumption, due to, for instance, the spatial distribution of the learning agents (e.g., robotic soccer). Some of them also need to know the entire joint action exactly. Communication might help relax these strict requirements, by providing a way for the agents to exchange needed data, like state observations or portions of Q-values. Furthermore, many algorithms proposed in literature (either for cooperative tasks or competitive ones), often use game-theoretic stateless tasks to test the approach. Complex domains such as distributed control of dynamic processes (traffic systems, power networks, sensor networks, etc.) are disregarded, so that the application of these algorithms to real-life multiagent problems is still an open question. For instance, *scalability* is the central concern for multiagent learning algorithms, because many of them require explicit tabular storage of the agents' action-value functions, which limits the applicability of the algorithms to problems with a relatively small number of discrete states, actions and agents. Also the problem of defining a *learning goal* is a difficult open issue. Usually goals are typically formulated in terms of static games, while their extension to dynamic tasks is not always clear or even possible. *Stability* and *adaptation* are two desired properties of a good multiagent learning system, still they seem to be antithetical, so how to reach both is not clear.

The importance of reward functions

Reward functions are crucial to a part of this thesis. So, in order to reach a better understanding of behaviour of different reward functions for cooperative multiagent learning, we have performed several experiments, using a variant of the Minority Game [4]. The problem is formulated as follows. Let be $\mathcal{O}$ a set of options where each agent can choose from. At each time-step, an agent can choose only one option $O_j \in \mathcal{O}$. Depending on the number of agents that chose option $O_j \in \mathcal{O}$, an *option reward* ρ_i is calculated. The goal for the team of agents is maximising an aggregation G of the option rewards.

Each agent is modelled as an independent learner, which updates its action-value function using the formula:

$$Q_i(O_j) \leftarrow Q_i(O_j) + \alpha \cdot [\ r_i - Q_i(O_j)\ ] \tag{2.10}$$

where α is the learning step, set to 0.5 in the experiments. The problem is an iterated single-stage game, so that there is no notion of state. The reward r_i depends on the form of the aggregation G of the option rewards, and its functional form is defined in the following experiments. Each learner uses the ϵ-greedy action selection policy, with $\epsilon = 0.05$.

Experiment 1. In the first experiment, the option reward ρ_i is calculated as:

$$\rho_i = \begin{cases} w_i \cdot k_i \cdot e^{-1} & if\ k_i \leq c_i \\ w_i \cdot k_i \cdot e^{-k_i/c_i} & otherwise \end{cases} \tag{2.11}$$

where w_i is the weight (or desirability) of option O_i, k_i is the number of agent that chose option O_i, and c_i is the option threshold. Basically, the option reward increases with the number of agents if it does not exceed the threshold c_i, and decreases if the number of agents exceeds the threshold c_i.

The goal G that we assign to the team of agents is maximising the function:

$$G = \sum_{O_i \in \mathcal{O}} \rho_i \tag{2.12}$$

In the experiment, we simulate 500 agents and 9 available options. We set $c_i = 125$ for every $O_i \in \mathcal{O}$, and we use the vector of weights [1 5 10 15 20 15 10 5 1].

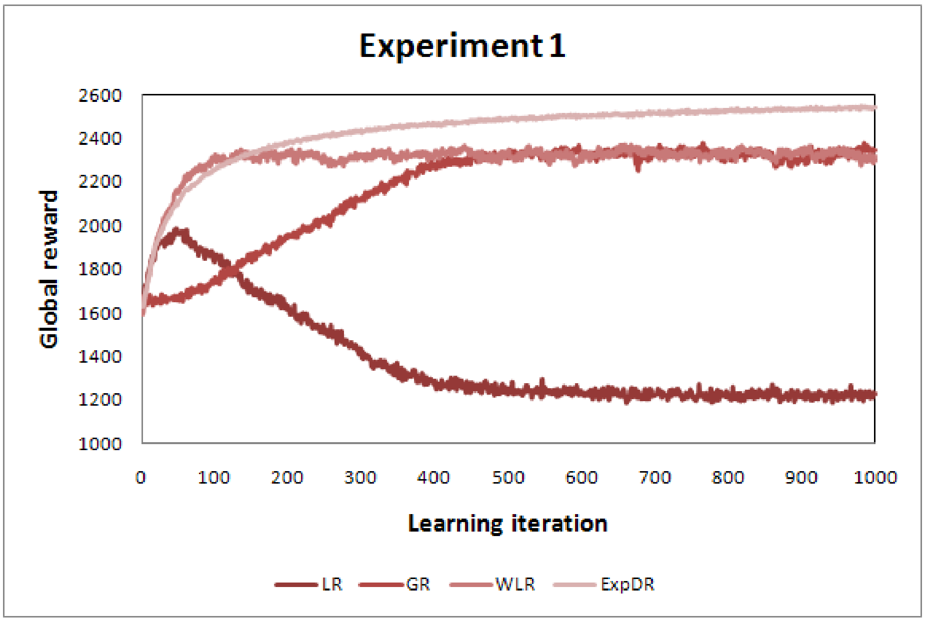


Figure 2.8: The importance of reward functions: experiment 1

We evaluated 4 different agent reward functions, namely the *local reward* (LR), the *global reward* (GR), the *wonderful life reward* (WLR) and the *expected difference reward* (ExpDR).

The local reward LR is defined as the reward of the option that agent j chose. Formally:

$$LR_j = \rho_j \tag{2.13}$$

Using the global reward GR as agent reward, each agent j is rewarded with the value returned by the function G. Formally:

$$GR_j = G = \sum_{i=1}^{9} \rho_i \tag{2.14}$$

The wonderful life reward WLR is defined as the difference between the global reward and the global reward that would have arisen if the agent j had been removed from the system. Formally:

$$WLR_j = \rho_j(k_j) - \rho_j(k_j - 1) \tag{2.15}$$

since agent j affects exclusively the reward of the option it chose, and removing agent j from the system means evaluating the reward that would have been obtained if option O_j had been selected by $k-1$ agents.

Finally, the expected difference reward ExpDR is defined as the difference between the expected global reward that agent j obtains when it selects option O_j and the expected global reward:

$$ExpDR_j = \mathbb{E}[G \mid O_j] - \mathbb{E}[G] \tag{2.16}$$

Such expectation values can be calculated by averaging the global rewards that an agent observes along the learning episodes, so that they become more and more precise with time.

The average results of 1000 trials of the first experiment are plotted in figure 2.8. We can clearly distinguish that the reward functions with global information (GR, WLR, ExpDR) perform better than the reward function with strictly local information (LR). Using LR, the global reward quickly increases but then converges to values far away from the optimum. The behaviour of the agents that arises is clearly greedy and poorly cooperative. Using WLR the global reward rapidly converges and settles around a good value. This is because in this experiment the WLR contains much information that each agent uses to drive its search for the best option to choose. Using GR, the convergence is slower but in the end the agents are able to obtain a global reward comparable of that obtained using WLR. The agent learns to cooperate, but slower than with WLR, due to the bigger quantity of noise carried by the reward signal. The best results are obtained using ExpDR: the convergence is fast as with WLR and the agents maximise with more efficacy the global reward.

Experiment 2. In the second experiment, the option reward ρ_i is calculated as in equation 2.11, but we change the goal that the team of agents pursues. In this experiment we introduce a factor of *equality* between option rewards, so that those joint

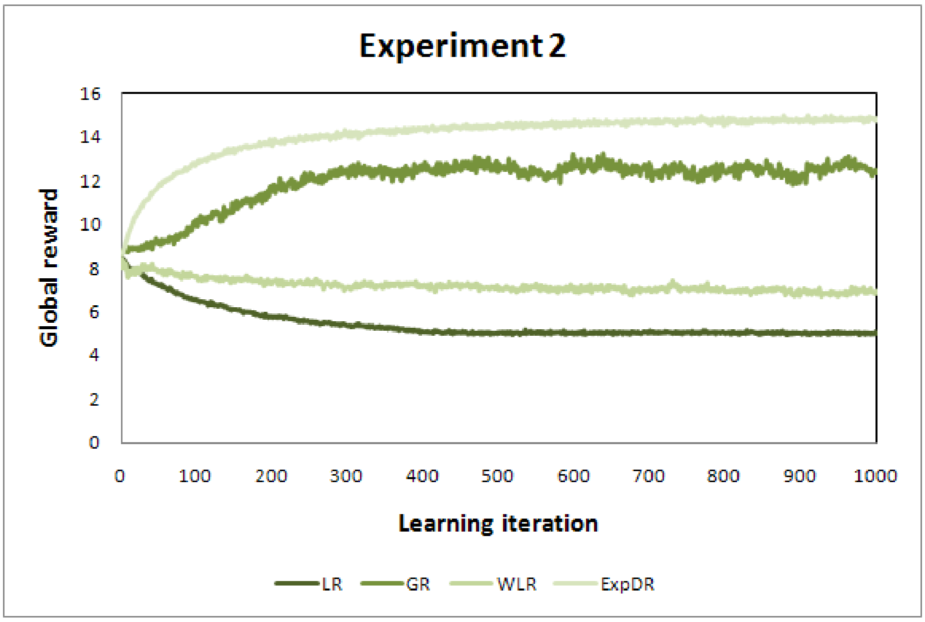


Figure 2.9: The importance of reward functions: experiment 2

actions that give rise to almost equal option rewards are highly rewarded. Formally:

$$G = \frac{\sum_{O_i \in \mathcal{O}} \rho_i}{1 + \sigma} \tag{2.17}$$

where σ is the standard deviation of the option rewards, defined as:

$$\sigma = \sqrt{\frac{\sum_{O_i \in \mathcal{O}} (\mu - \rho_i)^2}{N}} \tag{2.18}$$

where μ is the mean of the option rewards:

$$\mu = \frac{\rho_1 + \cdots + \rho_N}{N} \tag{2.19}$$

In the experiment, we simulate 500 agents and 9 available options as usual. The threshold c_i is set to 125 for every $O_i \in \mathcal{O}$, and we use the vector of weights [1 5 10 15 20 15 10 5 1]. Again we evaluated the local reward (LR), the global reward (GR), the wonderful life reward (WLR) and the expected difference reward (ExpDR).

With the local reward, an agent is aware exclusively of the consequences of its actions, so that the standard deviation σ that it perceives is always equal to 0. Thus, the local reward is again defined as the reward of the option that agent j chose. Formally:

$$LR_j = \rho_j \tag{2.20}$$

where ρ_j is the reward of the option selected by agent j.

Similarly, using the global reward as agent reward function, each agent j is rewarded with the value returned by the objective function G. Formally:

$$GR_j = G = \frac{\sum_{i=1}^{9} \rho_i}{1+\sigma} \tag{2.21}$$

The wonderful life reward is defined as the difference between the value returned by the objective function and the value returned by the objective function if we remove agent j from the system. Removing j from the system affects not only the sum of the option rewards, but also the standard deviation σ. Formally:

$$WLR_j = \frac{\sum_{i=1}^{9} \rho_i}{1+\sigma} - \frac{\sum_{i=1}^{9} \widehat{\rho_i}}{1+\widehat{\sigma}} \tag{2.22}$$

where $\widehat{\rho_i}$ and $\widehat{\sigma}$ are the option rewards and the standard deviation when agent j is removed from the system. Since agent j affects only the option it chooses, O_j, $\widehat{\rho_i}$ is defined as:

$$\widehat{\rho_i} = \begin{cases} \rho_i(k_i - 1) & if\ i = j \\ \rho_i(k_i) & otherwise \end{cases} \tag{2.23}$$

while $\widehat{\sigma}$ is defined as:

$$\widehat{\sigma} = \sqrt{\frac{(\widehat{\mu} - \rho_1)^2 + \cdots + (\widehat{\mu} - \rho_j(k_j - 1))^2 + \cdots + (\widehat{\mu} - \rho_9)^2}{9}} \tag{2.24}$$

where $\widehat{\mu}$ is the mean of the option reward when agent j is removed from the system:

$$\widehat{\mu} = \frac{\rho_1 + \cdots + \rho_j(k_j - 1) + \cdots + r_9}{9} \tag{2.25}$$

Finally, the expected difference reward ExpDR is defined as usual as the difference between the expected global reward that agent j obtains when it selects option O_j and the expected global reward:

$$ExpDR_j = \mathbb{E}[G \mid O_j] - \mathbb{E}[G] \tag{2.26}$$

The average results of 1000 trials of the second experiment are plotted in figure 2.9. Once again, the LR is the worst performing reward function. The agents are not able to jointly maximise the global reward, since it tends to decrease during the learning of the agents, and also the initial improvement as in experiment 1 disappears. Surprisingly, using WLR the agents perform quite a lot worse than in experiment 1. The global reward basically remains constant over all the learning episodes, approximately equal to the global reward that can be obtained by a team of agents that act randomly. The GR shows a similar dynamics as in experiment 1, characterised by a slow convergence, while the ExpDR is again the best performing reward function.

Experiment 3. In the third experiment, the option reward ρ_i is calculated as:

$$\rho_i = \begin{cases} k_i/N & k_i > 0 \\ 1 & otherwise \end{cases} \tag{2.27}$$

where k_i is the number of agent that chose option O_i, and N is the total number of agents. The collective goal is defined by the function:

$$G = \prod_{O_i \in \mathcal{O}} \rho_i \tag{2.28}$$

Given that $\rho_i \in (0, 1]$, the global reward G is maximised when all the agents select the same option. In the experiment we again simulate 500 agents and 9 available options, evaluating the four agent reward functions, LR, GR, WLR and ExpDR. The local reward LR is defined as the reward of the option that agent j chose. Formally:

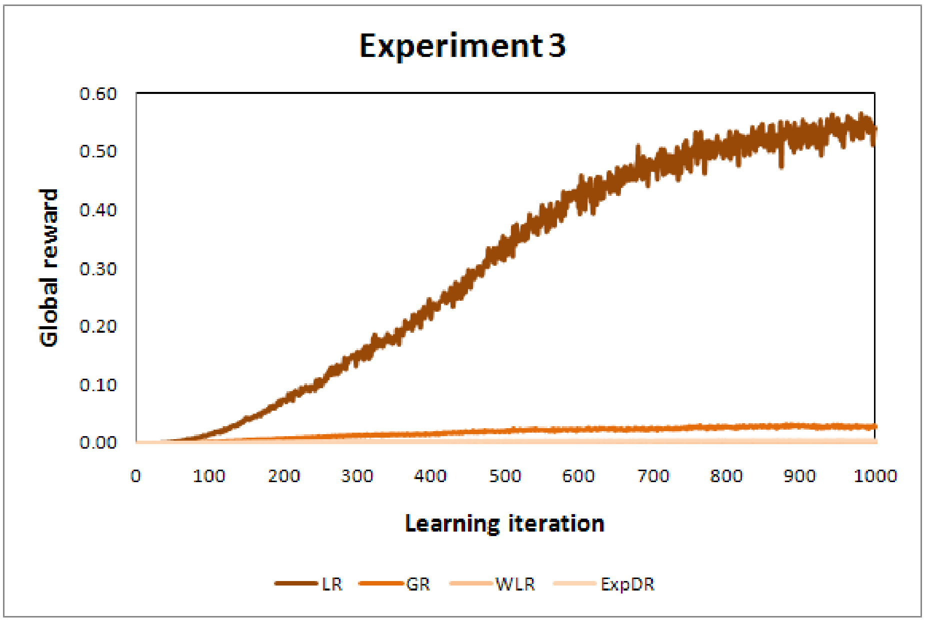


Figure 2.10: The importance of reward functions: experiment 3

$$LR_j = \rho_j = k_j/500 \tag{2.29}$$

where k_j is the number of agent that selected option O_j.

Using the global reward GR as agent reward, each agent j is rewarded with the value returned by the function G. Formally:

$$GR_j = G = \prod_{i=1}^{9} \rho_i \tag{2.30}$$

The wonderful life reward WLR is defined as the difference between the global reward and the global reward that would have arisen if the agent j had been removed from the system. Formally:

$$WLR_j = [\rho_j(k_j) - \rho_j(k_j - 1)] \cdot \prod_{\substack{i = 1 \\ i \neq j}}^{9} \rho_i \tag{2.31}$$

Finally, the agents calculate the expected difference reward ExpDR as the difference between the expected global reward that agent j obtains when it selects option O_j and the expected global reward:

$$ExpDR_j = \mathbb{E}[G \mid O_j] - \mathbb{E}[G] \tag{2.32}$$

The average results of 1000 trials of the third experiment are plotted in figure 2.10. Surprisingly, in this specific scenario all the reward functions that use global information (GR, WLR, ExpDR) perform quite a lot worse than the reward function with strictly local information (LR). If with GR, WLR and ExpDR, the agents are basically unable to agree on a single option, with LR they reach on average a global reward of 0.5 (recall that 1 is the optimum). This is because the local information is not only enough to drive the agents toward the selection of a common option, but also it is less noisy, because it clearly distinguishes the "good" agents from the "bad" agents. For instance, if $N-1$ agents select the same option, while only one agent selects a spare option, the $N-1$ "good" agents are rewarded with $N-1/N \simeq 1$, while the "bad" agent is rewarded with $1/N \simeq 0$. So that the former are incentivised to confirm the selected option, while the latter is incentivised to change its previously selected option and select another one. On the other hand, in the same situation the GR would have rewarded *all the agents* with $N-1/N^2 \simeq 1/N \simeq 0$, which means that the $N-1$ "good" agents would have observed the same low reward as the "bad" agent, wrongly believing that the option they had chosen was bad.

2.5 MAS in traffic and transportation

To achieve the goals pursued by the Intelligent Transportation Systems (ITS) there is an increasing need to understand, model, and govern such systems at both the individual (micro) and the society (macro) level. Transportation systems may contain thousands of autonomous entities that need to be controlled, raising significant technical problems. The inherent distribution of problems in traffic management and control, the high degree of complexity, and the fact that the actors in a traffic and transportation system (the driver, the pedestrian, the infrastructure component, etc.) fit the concept of autonomous agent very well, allows for a natural break down of the system into agents that interact so as to achieve their goals, selfishly as well as cooperatively. Therefore, traffic and transportation scenarios are extraordinarily appealing for multiagent technology [8][10][11].

Parunak [77] listed the characteristics of an ideal application suited for agent technology:

- **Modular**. Each entity (agent) defines a set of state variables that is clearly distinct from those of its environment, and it is possible to clearly identify the boundary between the agent and the environment.
- **Decentralised**. The application can be broken down into self-contained and decoupled software processes capable of performing useful tasks with a certain degree of autonomy.
- **Changeable**. The structure of the application is dynamic and may change quickly and frequently.
- **Ill-structured**. The knowledge of the application is not available when the system is being designed.
- **Complex**. The system exhibits a large number of different behaviours which may interact in sophisticated ways.

As most traffic and transportation logistics applications actually fit Parunak's characterisation rather well, this would suggest that agent technology indeed is a promising approach for this area.

Problem space. Figure 2.11 shows how to organise the space of multiagent problems [103]. There are two dimensions: the x-dimension represents whether or not the designer knows the agents' decision-making function (δ_i), the y-dimension represents whether or not the designer knows the mechanism ($\mathcal{M}$) that maps the collective (or joint) actions of the agents into an outcome.

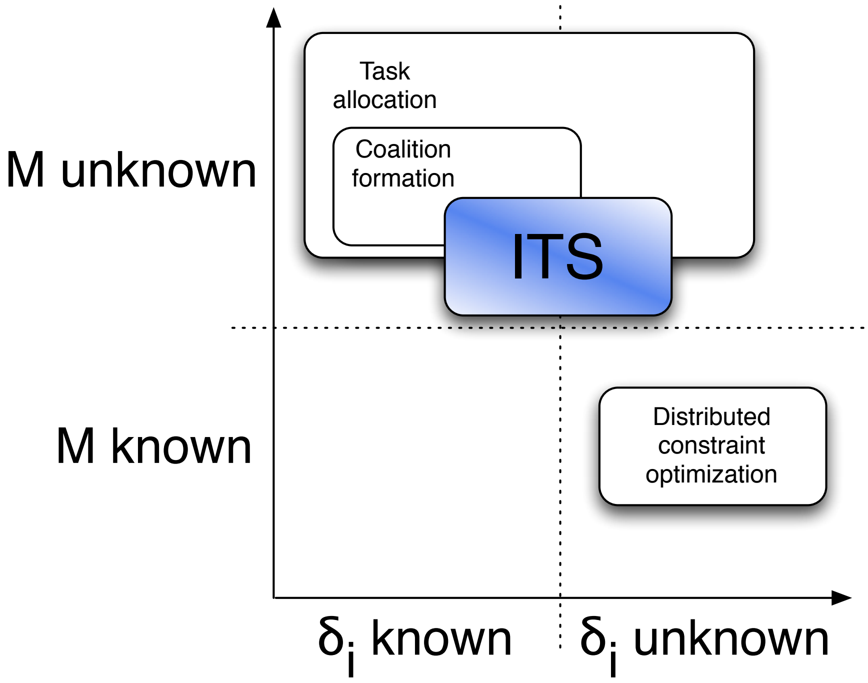


Figure 2.11: Space of all multiagent problems [103].

In the bottom left quadrant fall all the problems where the agents behaviour and the outcome for a vector of actions is known to the designer. Since everything is known, these problems are not so interesting from the multiagent perspective.

The bottom right quadrant represents the problems where the agent behaviour is not known but the mechanism that generates the outcome of a collective action is known. Distributed constraint optimisation and satisfaction problems [118] also reside in this quadrant. In these problems, the designer defines a global utility function so that, for a given joint action, it is possible to calculate the value of such global utility. Thus, the designer must determine how the individual agents must behave (δ_i) in order to maximise the global utility.

The top left quadrant represents systems where the agents' behaviour is known, so that the designer must define a mechanism that maps collective actions to a final outcome. This kind of system is usually composed of selfish agents, which act to maximise their utility: Problems that fall in this quadrant are, for instance, coalition formation [38] and task allocation [91].

Problems related with ITS fall between the top left and the top right quadrant. In fact, the behaviour of the driver agents is not known *a priori* by the designer (although some reasonable assumptions can be made), while the designer is free to define and engineer the agents of the infrastructure. Still, at design time it is not known which outcome will emerge from a collective action of both the driver agents and the infrastructure agents, due to the complexity and large-scale of the problem.

ITS dimensions. According to [31], we can define the following dimensions of a traffic and transportation application:

- **Domain**. The domain can be *transport*, *traffic* and *terminal*. Transport is the activity of moving something between point A and point B by one or several modes of transport. Traffic refers to the flow of different transports within a network (air, road, rail etc.). Within a transport chain operated by different modes, there are interfaces, referred to as terminals.

- **Transport mode**. There are five basic modes of transportation: *road*, *rail*, *air*, *water*, and *pipeline*.

- **Time horizon**. Time horizon refers to at what stage in the decision-making process the application is used. There are three levels of time perspective of the decision-making process: *strategic*, *tactical* and *operational* level. Strategic decision-making typically involves long-term decisions, tactical decision-making deals with medium-term issues and operational decision-making refers to how to actually do the work, i.e., short term issues.

- **Usage**. The applications can be classified as *automation systems* or *decision-support system* (DSS) [99]. An automation system is a self-acting mechanism that performs a required act in response to certain conditions. On the contrary, a decision-support system has an indirect impact on the decision-making, since the user makes decisions by taking (or not) the suggested decision into consideration.

- **Control, structure and attitude**. The control performed by the agents can be either *centralised* or *distributed*. The structure refers to the agents constituting the MAS, their roles, and the communication topology. The structure is either *static* or *dynamic*. Finally, the agent *attitude* captures the behaviour of agents, which is classified as either *cooperative* or *selfish*.

In this section, we will review the literature of multiagent applications in the *road traffic* domain, with special emphasis on the reservation-based intersection control mechanism proposed by Dresner and Stone [35]. We will highlight the following application dimensions: time horizon, usage, control, structure and attitude. The results with respect to these dimensions are summarised in table 2.1.

In [48], two multiagent systems, $InTRYS$ and $TRYSA_2$, which perform decision support for real-time traffic management in the urban motorway network around Barcelona are presented. Both systems make use of traffic management agents that apply knowledge-based reasoning techniques to deal with local traffic problems. In this work, two coordination mechanisms are employed, one centralised ($InTRYS$) and one distributed ($TRYSA_2$). The $InTRYS$ system uses a special coordinator agent endowed with knowledge on how to integrate local control strategies (proposed by traffic control agents) into a coherent global plan for the whole traffic network. In $TRYSA_2$, spatial problem areas are controlled by autonomous, selfish, traffic agents that coordinate using a mechanism called structural cooperation [73].

In [71] a design method for the construction of agent-based decision support systems (DSS) is presented. DSS are interesting in the traffic domain because they support the storage of large amounts of decision-relevant data and assist human operators for tactical decision-making. The authors take advantage of multiagent technology to construct the DSS, since multiagent systems reduce design complexity and also support a dialogue-based stance on decision support interactions. Starting from a generic organisational and communicative model for decision support environments, the authors present an abstract architecture for multiagent DSS, composed of connection agents (which provide data and execute actions), management agents (which advise the human operators and explain the decisions they suggest), user interface agents and peripheral agents (which represent the support infrastructure for the DSS, such as directory facilitators). The architecture has been instantiated for real-world problems by means of two prototypes for transportation management.

In [22] a cooperative, hierarchical, multiagent system for real-time traffic signal control is presented. The control problem is divided into various sub-problems, each of them handled by an intelligent agent that applies fuzzy neural decision-making. The multiagent system is hierarchical, since decisions made by lower-level agents are mediated by their respective higher-level agents. The multiagent architecture also adapt itself to the dynamically changing problem domain, using an on-line reinforcement learning process for each agent.

Adler et al. [2] explore the use of cooperative MAS to improve dynamic routing and traffic management. From one side, real-time control over the transportation network is performed by system operator agents. On the other, information service providers aim at advising the drivers and distributing the traffic, resulting in a better allocation of the network capacity. The agents involved use negotiation to seek a more efficient route allocation.

In [36] is presented a hierarchical multiagent system that consists of several agents that act locally, representing an intersection. These local traffic agents (LTAs) aim to optimise the performance of their assigned intersection, under the supervision of a coordinator traffic agent (CTA). Each LTA calculates the optimal local signal plan, using a basic expert system, and sends it to the supervising CTA, which adjusts the local plans it receives, taking into consideration the plans of the LTAs governing the neighbouring intersection. Nevertheless, many details are not mentioned or discussed, such as how the CTA adjusts the local plans or what a global solution in this case is.

In [101] a test bed for multiagent control systems in road traffic management is presented. As the complexity of traffic control on a network grows, dividing the coordination problem into smaller coherent sub-problems that can be solved with a minimum of interaction becomes necessary. The work focuses on the development of a test bed to evaluate different configurations and coordination mechanisms for a traffic managing multiagent system.

In [32] an approach based on swarm intelligence is proposed. Each intersection behaves like a social insect that receives stimuli to perform or to change tasks (i.e., signal plans) from the vehicles in form of “pheromone”. Therefore the queued vehicles may trigger some signal plan switching. Using the swarm approach, the system behaves as well as a central decision support system, although the time needed to converge to a stable coordination can be high.

In [54], traffic signal plans are coordinated through distributed constraint optimisation (DCOP), using cooperative mediation. The approach is intended to be a compromise between totally autonomous coordination and the classical centralised solution, like in TRANSYT [80] or SCOOT [50]. Each agent is assigned to one ore more variables of the DCOP, which have inter-dependencies and conflicts (e.g., two neighbouring intersections want to coordinate in different traffic directions.). A mediator agent is in charge of resolving these conflicts when they occur, recommending values for variables associated with the agents involved in the mediation.

In [81], a multi-layered architecture is proposed. In the bottom layer, traffic detectors sense changes in the traffic patterns and select appropriate signal plans consequently. The upper layer deals with previously unknown situations by searching for a signal plan based on off-line, evolutionary, optimisation.

In [6], history-based controllers gather knowledge about the trip plans of the drivers and recent performance, in term of travel times. The authors base their approach on the notion of historical fairness by allowing vehicles to store credits they receive when waiting at red lights, and spend credits when passing through intersections. The main drawback of such an approach is that drivers need to report the average waiting time over all intersections at the end of its trip, in order to assess the efficiency of the control.

In [58], multi-agent control and fuzzy inference are mixed to control a group of phases. Each group is modelled as an agent that can set the lights of the group to green when requested by traffic demand and when permitted by other agents. Thus, agents need to negotiate about how to operate together. Agents exchange their local traffic and control data to negotiate control decisions, which are to extend the green time or to terminate it.

In [28], a reinforcement learning system copes with the dynamism of the environment by incrementally building new models. When the traffic pattern changes, a new model is created and the learning in the new model starts. The changes in the traffic pattern are estimated according to the types of transitions and rewards observed. This means that the system models the flow patterns as non-stationary but divided in stationary, separated, dynamics that do not need to be known *a priori*. The creation of new models is controlled by a continuous evaluation of the prediction errors generated by each partial model.

In [113], traffic light agents use reinforcement learning in order to minimise the overall waiting time of vehicles in a small grid. Those agents learn a value function that estimates expected waiting times of vehicles given different settings of traffic lights. Value functions are also learnt to compute policies to select optimal routes for vehicles. The traffic light agents can shift from red to green and opposite at each time-step, although in the practice of traffic engineering changes are introduced only in a smooth way.

In [94], a similar reinforcement learning-based method for controlling traffic lights is presented. The traffic light agents learn to minimise the total travel time of all vehicles in the network. The control objective is global, although actions are local to the agents. The state of the learning task is represented as an aggregation of the waiting times at the intersection of individual vehicles.

The work presented in [9] investigates the effect of co-adaptation between drivers and traffic light agents, each having its own goal and learning algorithm. The objective of local traffic light agents is obviously to minimise queues around the intersection, whilst the objective of drivers is to minimise their travel times. The control is done via decentralised traffic signals. The traffic light agents use single agent Q-learning [108] to learn the effectiveness of a signal plan. Results show that in general co-evolution improves both travel time and occupancy, especially in large-scale situations.

In [69], heterogeneous groups of agents communicate to improve their learning skills. Each agent controls an intersection in a certain area, using information from several sources as learning input, and communicating with agents in the same area or in different areas. The goal of each agent is formalised as a weighted sum of two terms, which represent the compromise between improving the individual performance at the single intersection and the overall quality in the area. The work evaluates different types of agents, such as evolutionary neural agents, Q-learning agents and heuristic agents.

Reaching the equilibrium, as in the aforementioned learning-based approaches it not the only goal that can be pursued. Adaptation, for instance, may improve traffic flow as well. In [40], traffic lights self-organise by means of three methods, with no direct communication between them. It is shown that the adaptation to traffic conditions reduces waiting times and number of stopped vehicles.

Also Lämmer et al. [61] proposed a self-organising approach, inspired by the observation of self-organised oscillations of pedestrian flows at bottlenecks. The local interactions between neighbouring traffic lights lead to emergent coordination patterns such as “green waves” and achieve an efficient, decentralised traffic light control. The self-organised control is a combination of two rules, one that aims at optimising the flow and one that aim at stabilising it. Simulation results have shown a considerable reduction not only of the average travel times, but also of their variation.

Reservation-based intersection control.

The reservation-based system proposed in [35] assumes the existence of two different kind of agents: *intersection managers* and *driver agents*. The intersection manager controls the space of an intersection and schedules the transit of each vehicle. The driver agent is the entity that autonomously operates the vehicle.

Each driver agent, when approaching the intersection, contacts the intersection manager and requests a reservation to safely cross the intersection. Such a request contains the necessary information to simulate the vehicle trajectory through the intersection, such as the vehicle’s properties (vehicle’s ID, vehicle’s size, etc.) as well as some properties of the proposed reservation (arrival time, arrival speed, arrival lane, arrival road segment, type of turn, etc.). The intersection manager simulates the vehicle trajectory through the intersection and informs the driver agent whether or not its request is in conflict with the already confirmed reservations. If there is

	Time horizon			**Usage**		**Control**		**Structure**		**Attitude**	
	Strat.	Tact.	Op.	Aut.	DSS	Centr.	Distr.	Stat.	Dyn.	Coop.	Self.
Ossowski et. al. [71]		X			X		X	X		X	
Hernandez et. al. [48]			X		X	X	X	X		X	X
Choy et. al. [22]			X	X			X	X		X	
Adler et. al. [2]			X	X			X		X		X
France et. al. [36]			X	X		X		X		X	
Katwijk et. al. [101]			X	X			X		X	X	
Oliveira et. al. [32]			X	X			X		X	X	
Junges et. al. [54]			X	X			X	X		X	
Rochner et. al. [81]			X	X			X	X		X	X
Balan et. al. [6]			X	X			X	X		X	
Kosonen et. al. [58]			X	X			X	X			X
Silva et. al. [28]			X	X		X		X			X
Wiering et. al. [113]			X	X			X	X		X	
Steingrover et. al. [94]			X	X			X	X		X	
Bazzan et. al. [9]			X	X			X	X		X	
Nunes et. al. [69]			X	X			X	X		X	
Gershenson et. al. [40]			X	X			X		X		X
Lammer et. al. [61]			X	X			X		X		X
Dresner et. al. [35]			X	X		X		X			X

Table 2.1: State-of-the-art with respect to ITS dimensions

no such conflict, the driver stores the reservation details and tries to meet them; otherwise it may try again at a later time.

The reservation-based system gives also the possibility to the driver agent to change the parameters of a confirmed reservation as well as to cancel a confirmed reservation that it holds. For example, suppose that a driver agent realises that the traffic conditions have slightly changed and that it will be a bit late at the intersection with respect to the reserved arrival time. In this case the driver agent might change the reservation parameters, trying to get a new updated reservation providing the new, more accurate, arrival time.

Consider also the case that there is a crashed vehicle in front of the vehicle operated by the driver agent. In this case, the reservation is useless for the driver agent, because it will not be actually able to make use of it, so that the driver agent can cancel the reservation and make a new one.

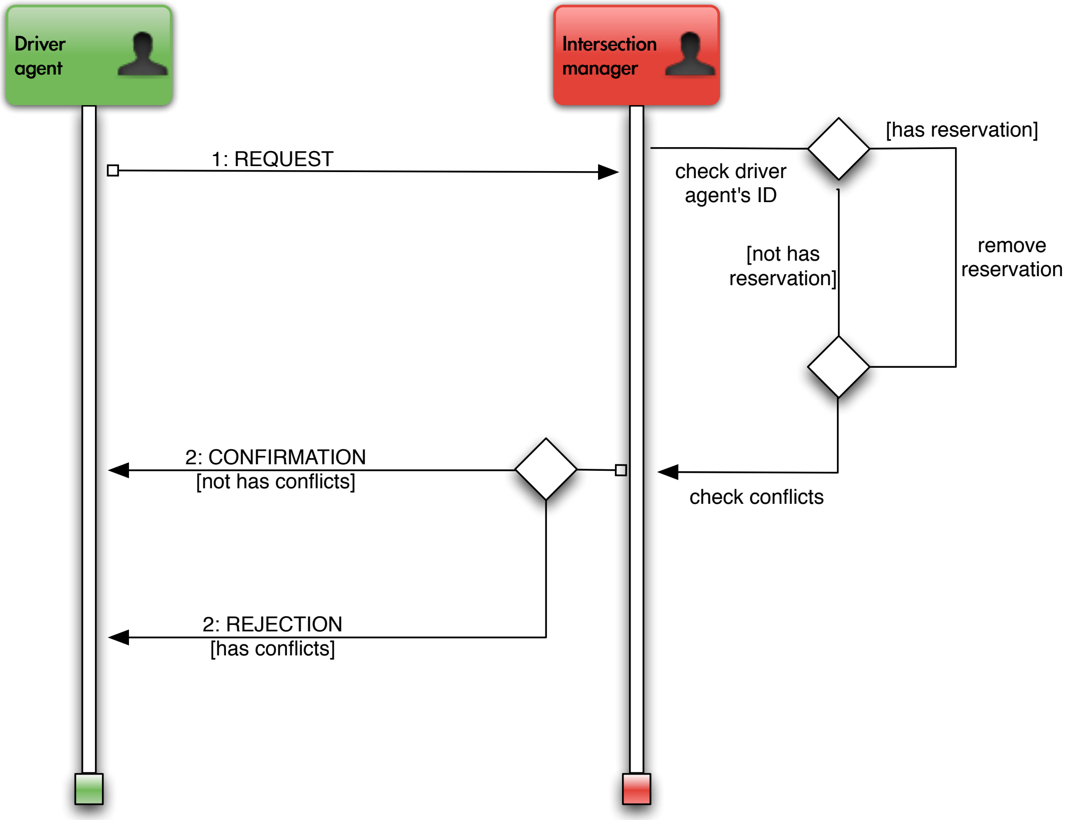


Figure 2.12: Protocol: requesting a reservation

Vehicle-infrastructure protocol. The reservation-based system relies on a tight integration between vehicles and intelligent infrastructure. The driver agent initiates the protocol sending a REQUEST message (see figure 2.13). The message contains the vehicle's ID, the arrival time, the arrival speed, the lane occupied by the vehicle in the road segment before the intersection and the type of turn.

The intersection manager, with the information contained in the REQUEST message, simulates the vehicle trajectory, calculating the space needed by the vehicle over time. If the transit does not have conflicts with the confirmed reservations, the intersection manager replies with a CONFIRMATION message, which implies that the driver agent accepts the reservation parameters. On the other hand, if the transit is not feasible, the intersection manager replies with a REJECTION message.

```
(request reservation
          :sender  D-3548
        :receiver  IM-05629
        :content(
                   :arrival_time   08:03:15
                   :arrival_speed  23km/h
                   :lane           2
                   :type_of_turn   LEFT
                 )
 )
```

Figure 2.13: Example of a REQUEST message

One of the norms of the control mechanism is that a driver agent is allowed to hold *only one* confirmed reservation at a particular intersection. If a driver agent has a confirmed reservation but it wants to change or cancel it and make a new one (e.g., with a more accurate arrival time), it simply starts again the reservation request protocol, sending a REQUEST message with the desired parameters. When the intersection manager receives the request, it firstly determines whether the driver agent already has a confirmed reservation, using the driver agent's ID. If so, the intersection manager implicitly assumes that the driver agent wants to replace the confirmed reservation and make a new one, so that it removes the reservation stored in its internal database and evaluates the new one as in the original protocol (i.e., if the new reservation request is feasible, then the intersection manager sends a CONFIRMATION message, otherwise it sends a REJECTION message). Figure 2.12 summarises the interaction protocol between driver agent and intersection manager.

Reservation distance. The protocol described above may generate deadlock, especially when many vehicles are approaching the intersection. Consider the following scenario: a vehicle A, following a vehicle B, requests and obtains a reservation *before* vehicle B. Vehicle B requests a reservation but it is rejected, due to conflicts with

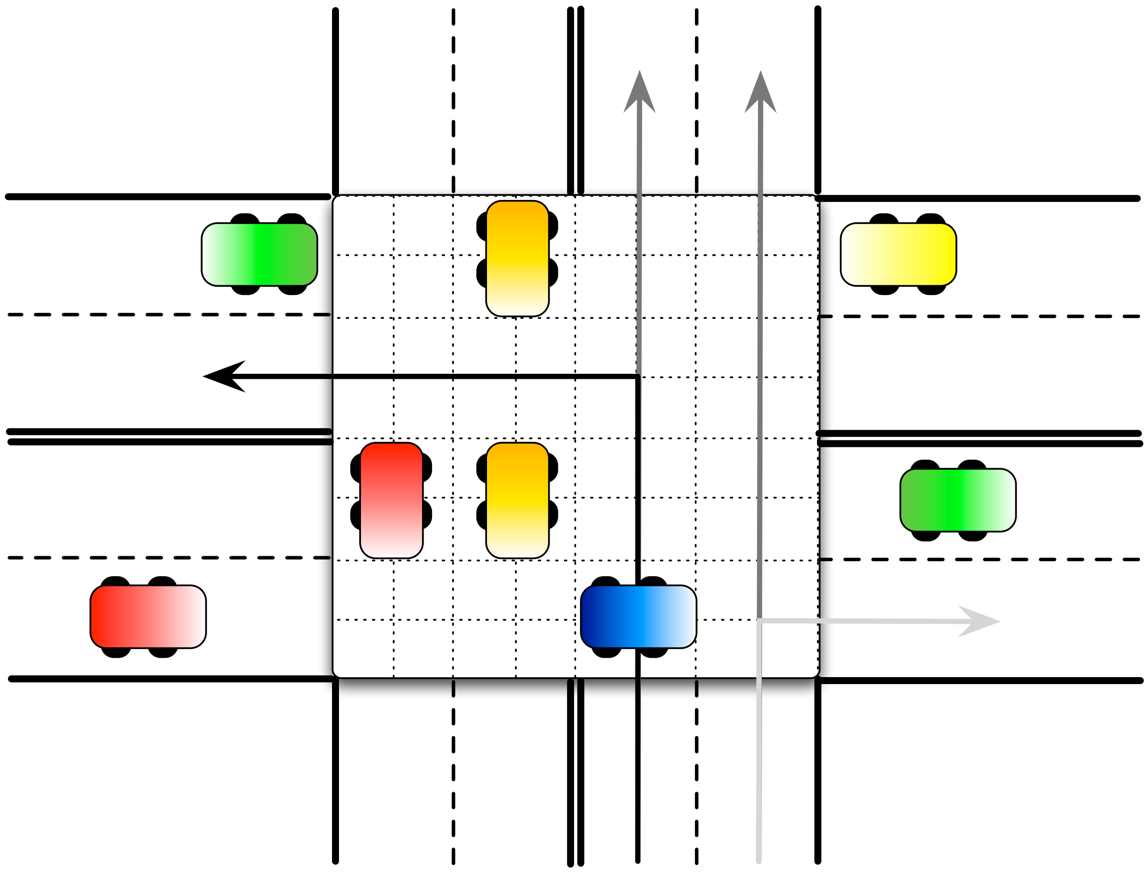


Figure 2.14: Example of a 4-links-3-lanes intersection

the reservation granted to vehicle A. Vehicle B, not holding a valid reservation to safely cross the intersection, stops at the intersection, thus impeding vehicle A to use its confirmed reservation and cross the intersection. To avoid (or at least minimise) these deadlock situations, Dresner and Stone proposed the use of the *reservation distance* as a criterion for filtering out reservation requests that could generate deadlock situations. Since the vehicles communicate the time at which they plan to arrive at the intersection, as well as what their speed will be when they get there (quantities which the vehicles have no incentive to misrepresent), it is possible to approximate a vehicle's distance from the intersection, given a reservation request by that vehicle. This approximation, called the *reservation distance*, is $v_a \cdot (t_a - t)$, where v_a is the proposed arrival speed of the vehicle, t_a is the proposed arrival time of the vehicle, and t is the current time. This approximation assumes the vehicle is maintaining a constant speed. The reservation processing policy uses such approximation as follows. For each lane i, the policy has a variable d_i, initialised to ∞. For each reservation request r in lane i, the policy computes the reservation distance, $d(r)$. If $d(r) > d_i$, r is rejected. If, on the other hand, $d(r) \leq d_i$, r is processed as normal. If r is rejected after being processed as normal, $d_i \leftarrow min(d_i,\ d(r))$. Otherwise, $d_i \leftarrow \infty$. While the use of the reservation distance does not guarantee that vehicles only get reservations if all vehicles in front of them already have reservations, it makes it more likely.

Model of the physical intersection. An intersection manager needs a model of the physical intersection that it governs. Such model is used by the intersection manager to simulate the vehicle trajectory and detect which parts of the intersection are required at which time. Thus, an intersection is modelled as a grid of squared tiles, whose size determines the intersection *granularity*. When an intersection manager receives a request, it simulates the transit of the vehicle through the intersection and determines which tiles are required at each time-step. If at least one of the tiles, needed at a certain time-step t, has been already granted to another vehicle, the request is rejected, otherwise it is granted to the requester. Depending on the intersection granularity, a vehicle at a certain time-step may occupy one or more tiles. Figure 2.14 shows an intersection with 4 incoming links, each of them with 3 lanes. The granularity is set so that a vehicle occupies 15 tiles at each time-step. The arrows shows the allowed trajectories for a vehicle that wants to turn left (dark-grey arrow), go straight (mid-grey arrows) or turn right (light-grey arrow).

2.6 Discussion

The applications of MAS technology to road traffic analysed in this section is far from being complete. Nevertheless we can extract some general trends, according to the categories introduced at the beginning of the section.

As said before, the time perspective of the decision-making process of the application may be strategic, tactical or operational (i.e., long-, medium- or short-term). All but one the revised applications work at the operational level, since they are directly involved in the control strategy. This is a hint that multiagent system technology is not suited, or at least is not employed, for strategic and tactical decision-making, which is still a human activity. Multiagent-based microscopic simulation is probably the MAS application that is more closely related to strategic and tactical decision-making. Using simulators, traffic engineers may evaluate different strategies and have an estimation of the impact of a strategic decision.

Two main uses have been outlined: automation systems or decision-support systems. All but two of the revised works are automation systems, denoting a greater interest in this kind of application. One reason could be that operational decisions are easier to automate than tactical or strategical ones. In fact, decision-support systems deal with decisions on which human operators want to have the last say, and furthermore they need to "explain" the suggested actions [72].

Regarding the control, all but three works perform a distributed control. Distributed control seems quite necessary in the road traffic domain, for scalability reasons. Nevertheless, centralised traffic control systems like TRANSYT [80] or SCOOT [50] work reasonably well.

Regarding the communication topology, the majority of the revised works assume a static structure. This is a reasonable assumption because the control action is usually performed by agents of the infrastructure (such as traffic lights), which does not change so quickly.

In general, there is no "agreement" about the best mental attitude that the agents should have in order to perform effectively as a whole. The automation system is composed of agents that reside in the infrastructure, thus the designers of the system may organise them to work cooperatively, defining the individual agents' utility functions, goals and policies. Nevertheless, one may purposely design the agents so that they act selfishly, since optimising its own performance using only local information is a more tractable problem for an agent. Thus, to ensure that some desired global property emerges, the designer can shape the rules specifying the interactions between the system's agents, as occurs in mechanism design [30] or in self-organising systems [15].

Finally, all the applications of multiagent technology in the traffic domain that we

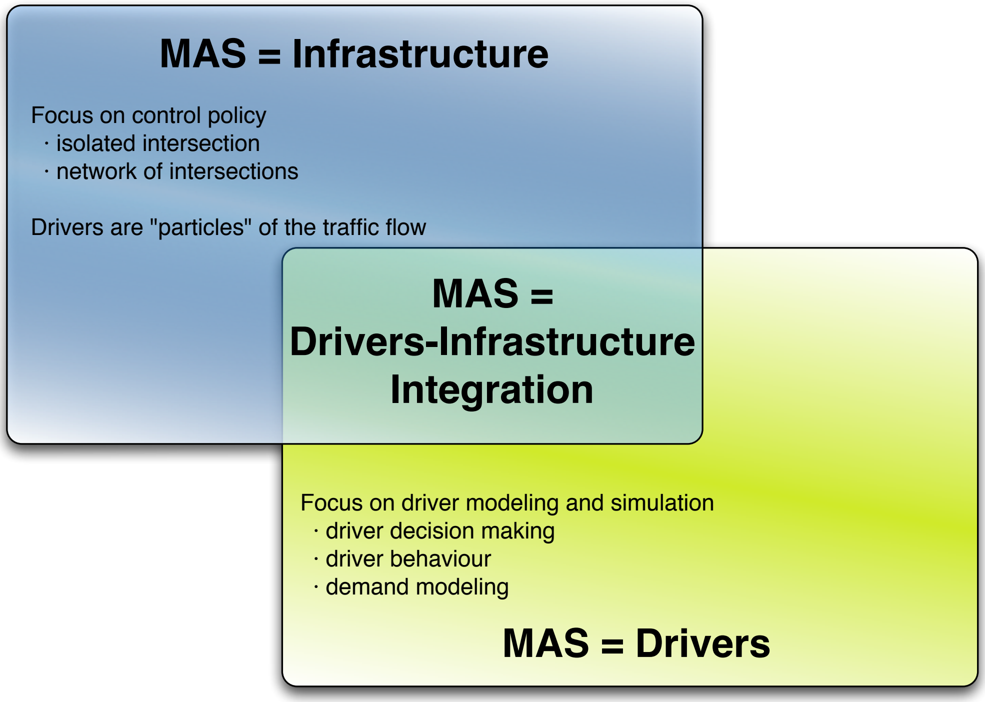


Figure 2.15: Agent-based approaches to ITS

have introduced in the previous section model the infrastructure and its components (traffic lights, signals, sensors, etc.) as a multiagent system, while the drivers are conceived as "particles" of a given traffic flow. Here the focus is on the control policies employed by the infrastructure agents, specially at intersections. Still, other agent-based applications conceive the drivers as the collective of agents whose behaviour is to be modelled, to simulate driver decision processes such as the departure time selection or the route choice [82].

Nevertheless, the continuous advances in software and hardware technologies will make a tighter integration between vehicles and infrastructure possible(figure 2.15), to take a step towards that future scenario described at the beginning of the introduction. For example, the Vehicle Infrastructure Integration (VII)[1] is an initiative that fosters research and development of applications for a series of technologies directly linking

[1] http://www.intellidriveusa.org

road vehicles to their physical surroundings. The first goal is improving road safety, for example reducing rear-end collisions by tracking obstructions in front or behind the vehicle and automatically applying brakes when needed. VII could also noticeably improve the operational efficiency of a transportation network. As vehicles will be linked together, the reaction times decrease and the gap between vehicles could be reduced so that there is less empty space on the road. VII may use real-time traffic data to provide accurate origin-destination studies to use in transportation forecasting and traffic operations. Tolling is another prospect for VII technology as it could enable roadways to be automatically tolled. Data could be collectively transmitted to road users for in-vehicle display, outlining the lowest cost, shortest distance, and/or fastest route to a destination on the basis of real-time conditions.

At the present time, cars can be equipped with features such as cruise control[52] and autonomous steering [60]. Furthermore, there exist small-scale systems of autonomous guided vehicles (AGV), for example in factory transport systems. If this trend holds, one day fully autonomous vehicles will populate our road networks. In this case, given that the system will have a variable (and possibly huge) number of vehicles and an open infrastructure, central control such as in today's AGV systems will be impossible. Thus, an infrastructure such as that proposed by Dresner and Stone [35] is more suitable to control and schedule the transit of AGVs. In their model, an intersection is regulated by an intelligent agent that assigns reservations of space slots inside the intersection to each autonomous vehicle intending to pass through the intersection. Such an approach has demonstrated, in a simulated environment, several advantages, because it may drastically reduce delays compared to traffic lights and it makes possible the use of fine grained, vehicle-centric, control policies.

Chapter 3

Single intersection

In theory, practice and theory are the same,
but in practice they differ.

Anonymous

In the intersection control mechanism proposed originally by Dresner and Stone (see section 2.5), the intersection manager processes the incoming requests with a *first-come-first-served* policy (FCFS). This means that if two vehicles send requests that require the same space-time in the intersection, the vehicle that sends the request first will obtain the reservation. This policy in extreme cases could result being quite inefficient. Consider the case of a set of n vehicles, v_1, v_2, ..., v_n, such that v_1's request has conflicts with every other vehicle, but that v_2, ..., v_n do not have conflicts with one another. If v_1 sends its request first, it will be granted and all other vehicles' requests will be rejected. On the other hand, if it sends its request last, the other $n - 1$ vehicles will have their requests confirmed, whilst only v_1 will have to wait.

A better request processing policy would be evaluating the whole set of n incoming requests, in order to confirm as many requests as possible. This problem can be formulated using the following graph-based formulation: be $\mathcal{G}(\mathcal{V}, \mathcal{E})$ an undirected graph, where $\mathcal{V}$ is the vertex set and $\mathcal{E}$ is the edge set. The vertex set $\mathcal{V}$ contains all

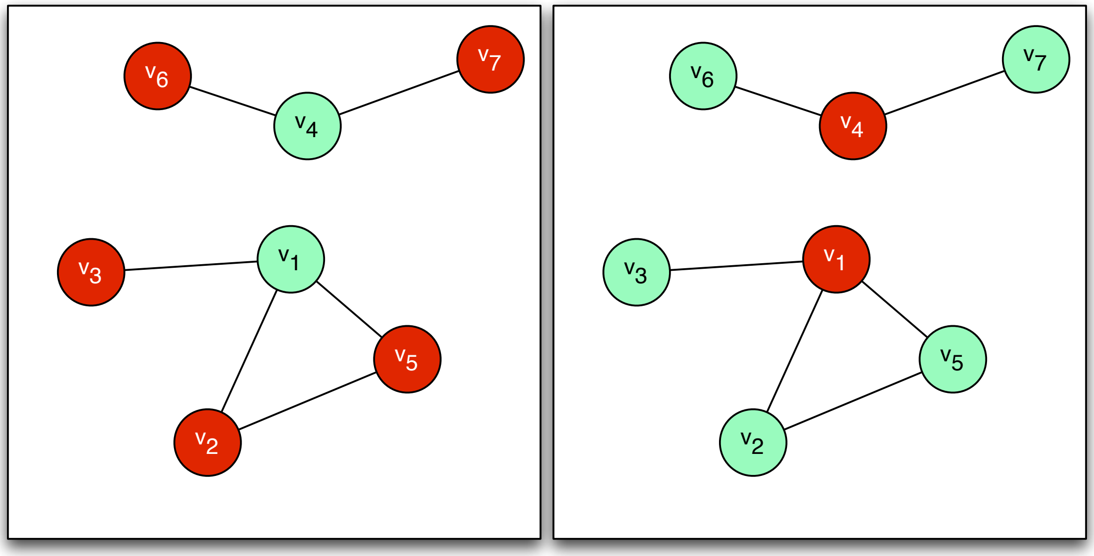


Figure 3.1: Confirmed requests (light circle) using the FCFS policy (left) with respect to the optimal solution (right)

the reservation requests to be processed. If two requests v_i and v_j conflict with one another, an edge connecting v_i and v_j is added to $\mathcal{E}$.

Figure 3.1 shows an example where 7 requests, labelled $v_1, \ldots, v_7$ (the subscript index represents the arrival order, the light color means that the request is confirmed, the dark color means that the request is rejected), are processed using the FCFS policy (left square). In this case, v_1 is confirmed, $v2$ and $v3$ are rejected due to conflicts with v_1, v_4 is confirmed, v_5 is rejected for conflicts with v_1, while v_6 and v_7 are rejected for conflicts with v_4. Thus, only 2 of 7 requests are confirmed. Nevertheless, considering all the 7 requests as a whole, 4 of 7 requests could have been confirmed, namely v_2, v_3, v_6 and v_7 (right square). With this formulation, finding the maximum number of non-conflicting requests is equivalent to solving the *maximum independent set problem* [93], which unfortunately is NP-hard.

In this chapter we analyse two different types of policies that an intersection manager may use to process the incoming requests. The policies of the first type are inspired by the research on adversarial queueing theory (AQT). The policies of the second type are based on the combinatorial auction (CA) theory. The chapter is structured as follows: in section 3.1 we briefly introduce the custom simulator of a single intersection that we used to perform the empirical evaluation; in section 3.2 we detail the AQT-based policies and their performance; in section 3.3 we analyse the auction-based policies; finally we discuss the experimental results in section 3.4.

3.1 Simulator

Traffic simulation tools are becoming fundamental in helping evaluate of new traffic control strategies or assess the impact of infrastructure improvements. Traffic simulation is a very active field, with many projects that are developed both in academia as well as in commercial firms. Academic traffic simulation tools are usually offered to the research community as open source software, although using them may be quite complicated: proprietary file formats, scarce documentation and "programming tricks" only known to the developers do not facilitate the use and the extension of those software. Among them, MITSIMLab [1], developed at MIT, is probably the best academic traffic simulator.

On the other hand, commercial traffic simulators come as full-fledged products, with many features and functionalities, such as 3D visualisation, network modelling editors, programming API and analysing tools. Still, apart from their price (that is often around the ten thousands euros), they are hardly extensible to pursue specific researching goals. Among them, the most famous and slicker are AIMSUN[2], Paramics[3], VISSIM[4], TransModeler[5] and Trafficware[6].

Since we need a simple traffic simulator that emulates the traffic at a single inter-

[1]http://mit.edu/its/mitsimlab.html

[2]http://www.aimsun.com

[3]http://www.paramics-online.com

[4]http://www.ptvag.com

[5]http://www.caliper.com/TransModeler/default.htm

[6]http://www.trafficware.com

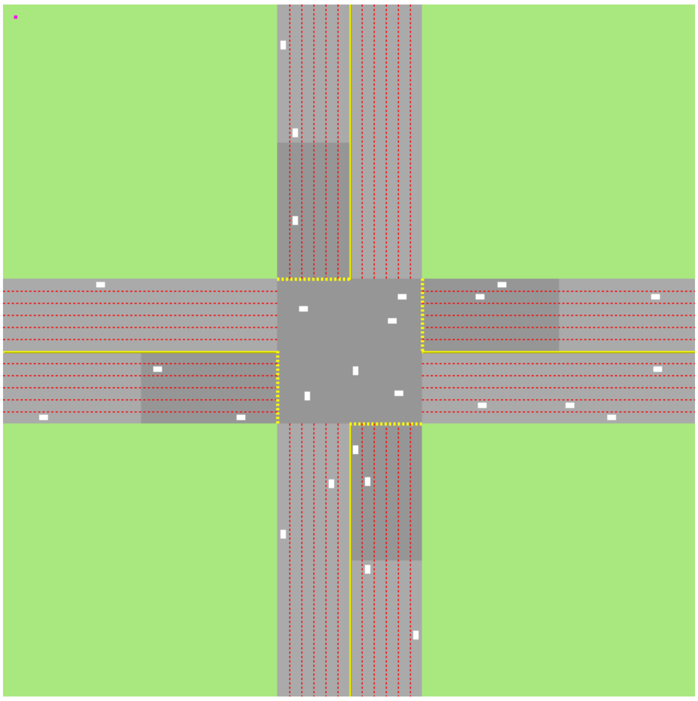


Figure 3.2: Simulator of a single intersection

section, we decided to develop our own custom, microscopic, traffic simulator.

Microscopic model

The simulator is a microscopic, time-and-space-discrete, simulator, with simple rules for acceleration and deceleration. The simulated area is modelled as a grid, and subdivided in lanes. Each lane is $3m$ wide, and subdivided in 12 squared tiles of $0.25m$ each. Each vehicle is modelled as a rectangle of 8×16 tiles, or equivalently, as a rectangle of $2m \times 4m$.

Each vehicle has a preferred speed $\in [30, 50]km/h$, which is assigned when the vehicle is spawned inside the simulation using a normal distribution with mean $40km/h$ and variance $5km/h$.

The vehicle dynamic is regulated by the IDM car-following model [98]. The decision of any driver agent to accelerate or to brake depends only on its own speed, and on the speed of the vehicle immediately ahead of it. Specifically, the acceleration

dv/dt of a given vehicle depends on its speed v, on the distance s to the front vehicle, and on the velocity difference Δv (positive when approaching)

$$\frac{dv}{dt} = a \cdot \left[1 - \left(\frac{v}{v_p} \right) - \left(\frac{s^*}{s} \right)^2 \right] \tag{3.1}$$

where

$$s^* = s_0 + \left(v \cdot T + \frac{v \cdot \Delta v}{2 \cdot \sqrt{a \cdot b}} \right) \tag{3.2}$$

and a is the acceleration, b is the deceleration, v is the actual speed, v_p is the preferred speed, s_0 is the minimum gap, T is the time headway.

The acceleration is divided into an acceleration towards the preferred speed on a free road, and braking decelerations induced by the front vehicle. The acceleration on a free road decreases from the initial acceleration a to 0 when approaching the preferred speed v_p.

The braking term is based on a comparison between the "preferred distance" s^*, and the actual gap s with respect to the front vehicle. If the actual gap is approximately equal to s^*, then the braking deceleration essentially compensates the free acceleration part, so the resulting acceleration is nearly zero. This means that s^* corresponds to the gap when following other vehicles in steady traffic conditions. In addition, s^* increases dynamically when approaching slower vehicles and decreases when the front vehicle is faster. As a consequence, the imposed deceleration increases with decreasing distance to the front vehicle, increasing its own speed, and increasing speed difference to the front vehicle. The aforementioned parameters were set to $v_p = 50km/h$, $T = 1.5s$, $s_0 = 2m$, $a = 0.3m/s^2$, $b = 3m/s^2$.

The speed of a vehicle is updated every second, and its position, since the space is discrete, is updated to the tile closest to the "real" new position in the continuous space.

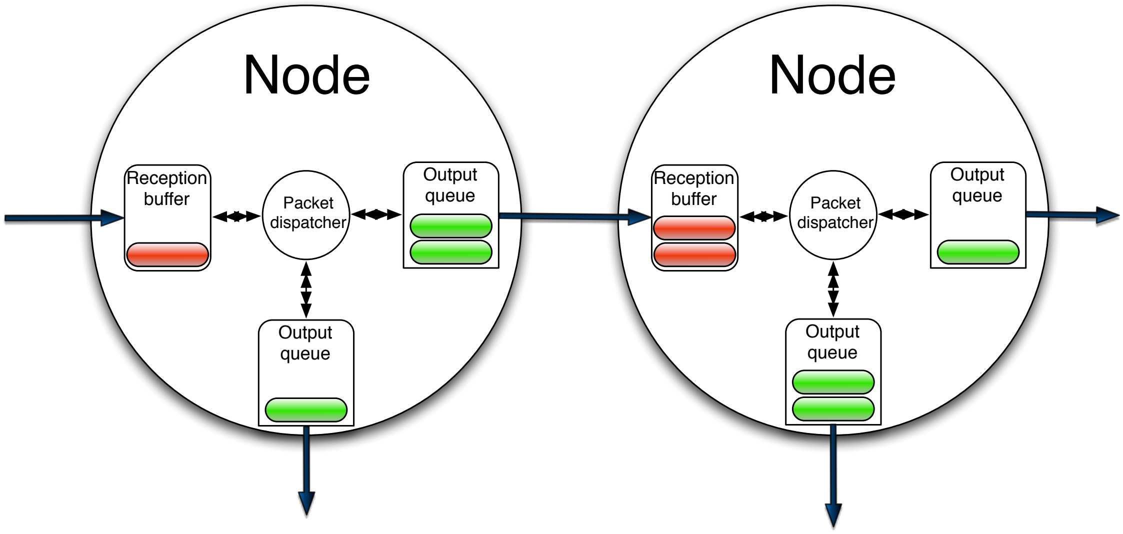


Figure 3.3: Elements of the network in the AQT model

3.2 AQT-inspired policies

In the original version of the work by Dresner and Stone, the intersection manager uses a simple FCFS policy to process the incoming requests. Still, other requests processing policies, inspired by the research on adversarial queueing theory (AQT), can be employed. The AQT [16] model has been used in recent years to study the stability and performance of packet-switched networks. In this model, the arrival of packets to the network (i.e., the traffic pattern) is controlled by an adversary that defines, for each packet, the place and time in which the packet joins the system. Each node in the network has a reception buffer for every incoming edge, an output queue for every outgoing edge, and a packet dispatcher that dispatches each incoming packet into the corresponding output queue (or removed, if this is the final node of the packet), using a specific policy (see figure 3.3). Under these assumptions, the stability of the network system is studied, where stability is the property that at any time the maximum number of packets present in the system is bounded by a constant that may depend on system parameters.

The packet processing of a packet dispatcher in the AQT model and the request processing of the intersection manager in the reservation-based control mechanism share some similarities. In the same way a packet dispatcher decides which packet from the reception buffer will be dispatched to the corresponding output queue, an intersection manager may decide in which order a set of reservation requests must be processed, assigning priorities to requests according to its scheduling policy. Taking inspiration from the AQT model, we compared the FCFS policy with 4 universally stable policies, namely *longest-in-system* (LIS), *shortest-in-system* (SIS), *farthest-to-go* (FTG) and *nearest-to-source* (NTS). The LIS policy gives priority to the request of the vehicle which joined the system earliest. The SIS policy gives priority to the request of the vehicle which joined the system latest. The FTG policy gives priority to the request of the vehicle which still has to traverse the longest path until reaching its destination. The NTS policy gives priority to the request of the vehicle which is closest to its origin, i.e., which has traversed the least of its whole route.

In order to implement these 4 policies, we need that a reservation request contain the necessary additional information: the time stamp when the vehicle joined the system, i.e., when it started to travel, an identifier of the origin location, and an identifier of the destination location.

Experimental results In this scenario we simulate a single intersection with 4 incoming links of 3 lanes each (see figure 3.2). We simulate different traffic demands by varying the expected number of vehicles (λ) that, for every O-D pair, are spawned in an interval of 60 seconds. We spawned vehicles for a total time of 10 minutes. Table 3.1 summarises the overall traffic demand for different values of λ. As a baseline, we used an intersection regulated by traffic lights with 4 phases (one per incoming link) of 30 seconds each (TL in the following figures and tables).

The metrics we used to evaluate the performance of the different policies were the following:

Average delay (sec.) The average delay measures the increase in travel time due to the presence of the intersection (be it either reservation-based or regulated by traffic lights). It is measured running two types of simulation: in the first one,

	Expected vehicles per minute per O-D pair (λ)						
	1	5	10	15	20	25	30
# of vehicles	29	136	285	438	633	716	832

Table 3.1: Traffic demands for a single intersection

the intersection is regulated by the control mechanism under evaluation and the vehicles must obey the norms that the control mechanism imposes; in the second one, the vehicles travel as if they could pass through the intersection unhindered. The difference between the two average travel times gives us the average delay. Formally:

$$\frac{\sum_{i\in\mathcal{V}}(t_f^i - t_0^i) \;-\; \sum_{i\in\mathcal{V}}(\widehat{t}_f^i - \widehat{t}_0^i)}{N}$$

where $\mathcal{V}$ is the set of vehicles, N is the number of vehicles, t_f^i and t_0^i are respectively the time when vehicle i arrives at its destination and when it leaves its origin in the simulation with the intersection regulated by a control mechanism, while $\widehat{t}_f^i$ and $\widehat{t}_0^i$ are respectively the time when vehicle i arrives at its destination and when it leaves its origin if we make the vehicles cross the intersection unhindered.

Average queue time (sec.) The average queue time is the time spent by the vehicles at the intersection queue of its corresponding lane. When the vehicle reaches the front of the queue, it enters the intersection and passes through it. Note that the queue time could be zero, if the queue is empty and the vehicle enters the intersection directly. Formally:

$$\frac{\sum_{i\in\mathcal{V}}(t_{q_f}^i - t_{q_0}^i)}{N}$$

where $\mathcal{V}$ is the set of vehicles, N is the number of vehicles, $t_{q_f}^i$ is the time when vehicle i leaves the queue of the intersection and $t_{q_0}^i$ is the time when it enters the queue of the intersection.

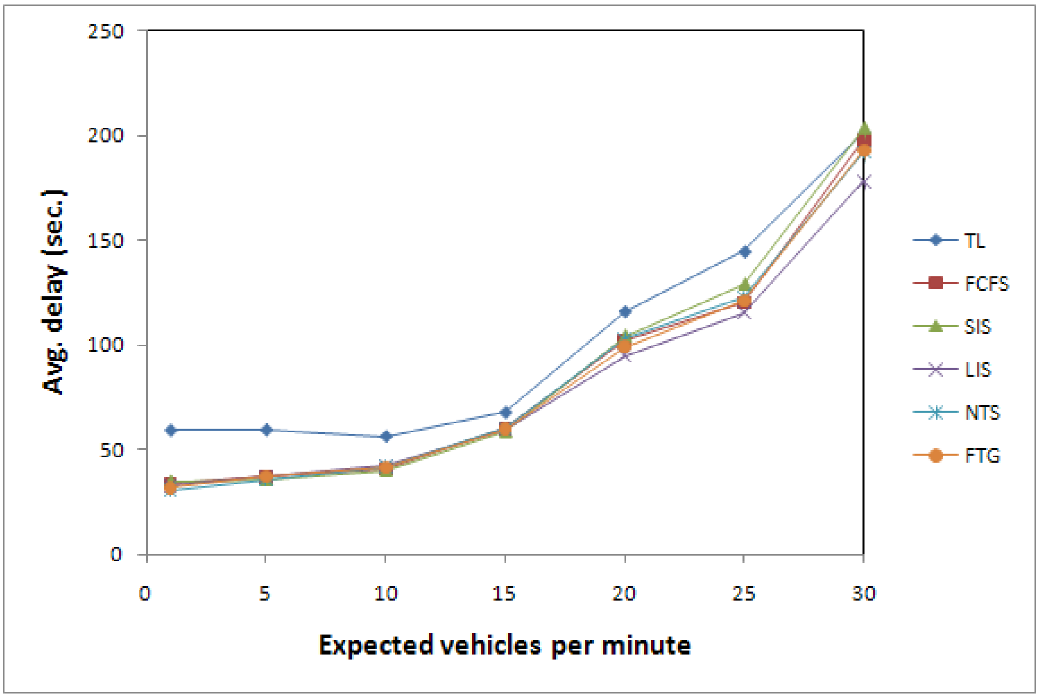


Figure 3.4: Average delay

Average rejected requests (% of sent requests) The average rejected requests is a metric that applies only to the reservation-based policies, and is measured as the ratio between the rejected requests and the sent requests. Formally:

$$\frac{\sum_{i \in \mathcal{V}} r^i / s^i}{N}$$

where $\mathcal{V}$ is the set of vehicles, N is the number of vehicles, r^i is the number of rejected requests of vehicle i and s^i is the number of requests sent by vehicle i.

Figure 3.4 plots the average delay for different traffic demands ($\lambda \in [1, 30]$). In general, all the reservation-based policies (FCFS, LIS, SIS, FTG, NTS) tend to behave in the same manner, reducing the average delay by about 30% for low traffic demand ($\lambda \in [1, 15]$) and 10% for greater traffic demand ($\lambda \in [15, 30]$). Still, when the traffic demand reaches extreme values (around $\lambda = 30$), the performance of reservation-based intersection converges to the performance of an intersection controlled by traffic lights.

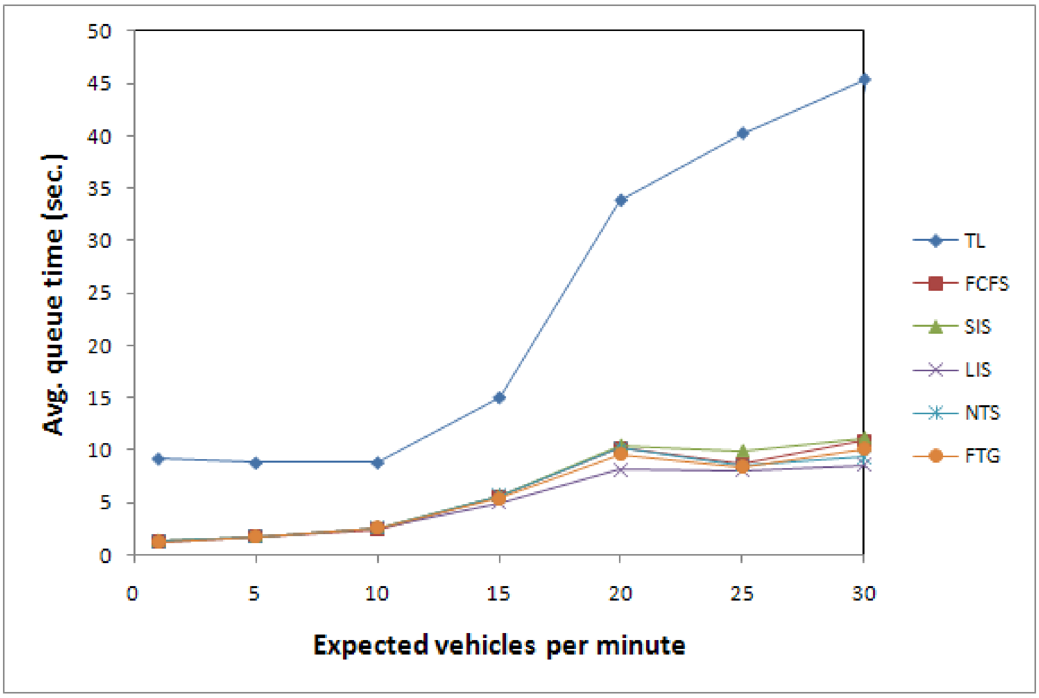


Figure 3.5: Average time in queue

In this extreme case, LIS is the best performing policy, with a reduction of 12% of the average delay, compared to 2% of FCFS, 5% of NTS, 4% of FTG and −1% of SIS (i.e., SIS actually increases the average delay with respect to TL).

Generally speaking, we can conclude that the reservation-based intersection outperforms traffic lights particularly when the traffic demand is below a certain threshold, because few requests are rejected and the majority of vehicles can pass through the intersection without waiting for the corresponding green phase, as in intersections controlled by traffic lights. Nonetheless, when the traffic density reaches the critical value of $\lambda = 30$, the reservation-based intersection tends to show the same performance of traffic lights, because the correct arrival time becomes harder to estimate, so that many requests are cancelled and resubmitted. Although the experiments were performed with a custom simulator that is different to that used in the original work by Dresner and Stone [35], the above results seem consistent with the results that we can find in that work. In fact, in [35] the reservation-based intersection with FCFS

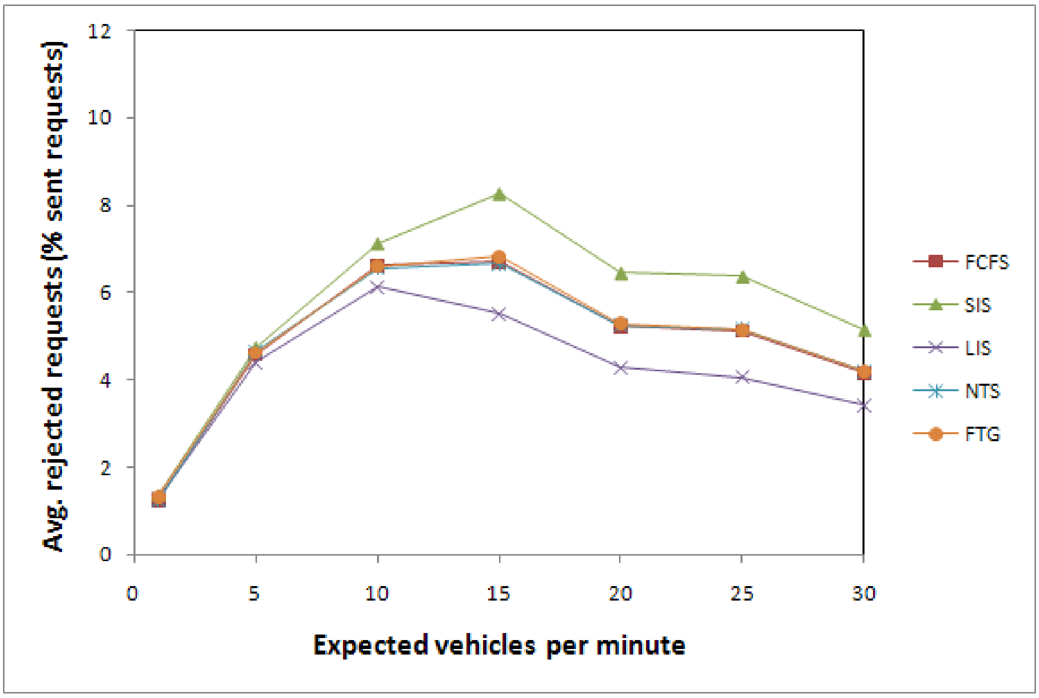


Figure 3.6: Average rejected requests

policy outperforms traffic lights when the traffic density is between 0 and 1 vehicle per second, while the authors did not give any results for higher traffic demands. In our experiments, the performance of a reservation-based intersection converges to the performance of traffic lights when the expected number of vehicles (λ) is 30, which corresponds to 1.38 vehicles per second, beyond the maximum value evaluated in [35].

Figure 3.5 plots the average time spent at the intersection queue. Two very distinct dynamics can be seen here. With traffic lights, the time spent by the vehicles at the intersection queue grows linearly with the traffic demand. On the other hand, with a reservation-based intersection, the queue time settles around about 7 seconds, whatever the policy in use.

This plot gives us an idea of the vehicle's behaviour when approaching the two different types of intersection. If the intersection is regulated by traffic lights, the vehicle proceeds at the speed permitted by the traffic conditions and, once it reaches the intersection proximity, if the traffic light is red it enters the intersection queue. In this way, the more the vehicles approaching the intersection, the longer the waiting time at the intersection queue. With a reservation-based intersection the dynamic of the vehicle approaching the intersection is different. If the vehicle holds a valid reservation, it maintains its speed because a safe transit is guaranteed. On the other hand, if it does not have such reservation, it reduces its speed and tries again. Thus the collective behaviour is a slower, smoother, traffic flow through the intersection, with little time spent at the intersection queue. Finally, we evaluated the reservation-based policies (FCFS, LIS, SIS, FTG, NTS) in terms of average rejected requests (as a percentage of the sent requests). Here, with rejected request, we refer to a request that cannot be granted due to *conflicts* with the already confirmed reservations. Figure 3.6 plots the results for the different traffic demands under evaluation. With low traffic demand, all the policies perform quite similarly, and the percentage of rejected requests increases linearly with the number of vehicles approaching the intersection. When the traffic demand reaches a critical point (around $\lambda = 15$), the percentage of rejected requests tends to decrease with the traffic demand. The reason for this counterintuitive trend is the effect of the *reservation distance*. As said in section 2.5, the reservation distance is the maximum distance at which a driver agent is allowed to request a reservation. Whenever a driver agent cannot get a reservation due to conflicts with the existing ones, the reservation distance is updated to the distance at which the driver agent requested the reservation. As the number of rejected requests increases, the reservation distance tends to become smaller. With high traffic demand, the effect of the reservation distance becomes predominant, filtering out the majority of the reservation requests and processing only those of the nearest vehicles. Since less requests are processed, less conflicts are detected, so that the percentage of rejected requests decreases.

3.3 Auction-based policies

In the reservation-based mechanism, evaluating the incoming requests to grant the associated reservations can be considered as the process of assigning resources to agents that request them. This process can be driven not only by the principle of *maximisation of the number of assigned resources*, but also by the principle of *rewarding the agents that value the disputed resources the most*. From this perspective, all the policies introduced in the previous section are inspired by the first principle, because there is no notion of a value of a reservation, while the second principle is often one of the objectives of any auction mechanism. A similar objective underlies the work by Schepperle and Bohm [88]. In their work it is the intersection manager that initiates a Vickrey auction, offering the earliest time slot to the first vehicles that are approaching the intersection on each lane. They assume that the intersection manager is able to detect if a vehicle has other vehicles in front of it, in which case the vehicle is excluded to participate in the auction. Furthermore, since the intersection manager assigns the earliest time-slot with a non-combinatorial auction, only one bidder will get a specific time-slot. However, it is possible for two reservation requests to share the same time-slot and be non-conflicting at the same time, thus potentially reducing the intersection throughput. For example, consider a 4-links-1-lane intersection. A vehicle that will enter the intersection from the top link at time t, to exit from the its right-hand link, and another vehicle that will enter the intersection from the bottom link at time t, to exit from the its right-hand link, will share the time-slot t, but they are not conflicting, since their trajectories do not intersect at all.

In this section, we introduce an auction-based policy to process the incoming requests, formally specifying the auction design space (resources, bidding rules, clearing policy etc.) and how the original protocol is modified.

3.3.1 Auctioned resources

The first step for the design of any auction is the definition of the resources (or items) that are allocated through the auction itself. The nature of items determines which type of auction can be employed to allocate them. For instance, if the items are N single indivisible goods, then the auctioneer may allocate them with N consecutive English open-outcry auctions [59].

In our scenario, the auctioned good is *the use of the space inside the intersection at a given time*. According to the Dresner and Stone's model, an intersection is

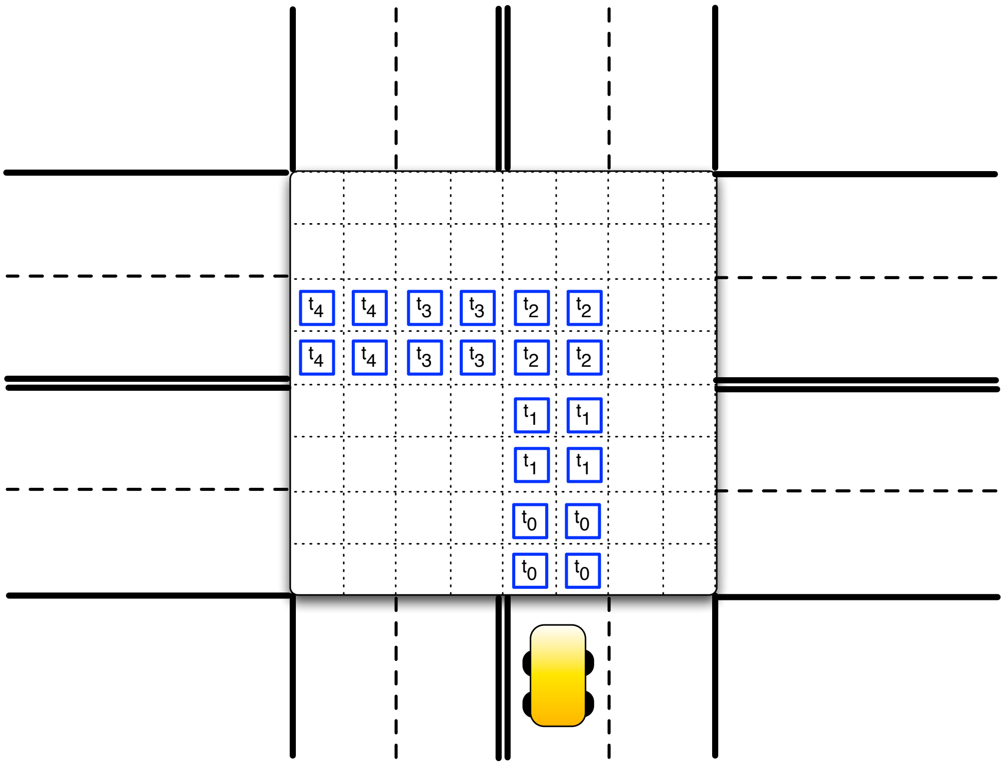


Figure 3.7: Bundle of items defined by a reservation request.

modelled as a discrete matrix of space slots. $\mathcal{S}$ is the set of the intersection space slots, $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$, t_{now} is the actual time, and $\mathcal{T}(t_{now}) = \{t_{now} + \tau, \forall \tau \in \mathbb{N}\}$ is the set of (future) time-steps. The set of items that a bidder can bid for is the set $\mathcal{I} = \mathcal{S} \times \mathcal{T}(t_{now})$.

Due to the nature of the items, a bidder is only interested in *bundles of items* over the set $\mathcal{I}$. In fact, a reservation request implicitly defines which space slots at which time the driver agent needs in order to pass through the intersection (see figure 3.7). Thus, the items must necessarily be allocated by a *combinatorial auction.* Combinatorial auctions present many new challenges (computational and economic) as compared to traditional auctions. The main computational problem is the winner determination problem (WDP), that is, how to efficiently determine the allocation once the bids have been submitted.

```
(request reservation
            :sender  D-4888
          :receiver  IM-05402
          :content(
                     :arrival_time   18:03:15
                     :arrival_speed  33km/h
                     :lane           1
                     :type_of_turn   STRAIGHT
                     :bid            1.45 €
                    )
 )
```

Figure 3.8: Example of a bid contained in a REQUEST message

3.3.2 Bidding rules

The bidding rules define the form of a valid bid accepted by the auction [116]. In our scenario, a bid over a bundle of items is implicitly defined by the reservation request. Given the parameters arrival time, arrival speed, lane and type of turn, the auctioneer (i.e., the intersection manager) is able to determine which space slots at which time are needed. Thus, the additional parameter that a driver agent must include in its reservation request is the value of its bid, i.e., the amount of money that it is willing to pay for the requested reservation (see figure 3.8).

A bidder is allowed to withdraw its bid and submit a new one. In general some dominance rules must hold between previous bids by the same bidder. For instance, in the English open outcry auction, a new bid must beat the highest so far (perhaps by a specified increment). In our scenario, the new bid must be greater than or equal to the old one by the same bidder, guaranteeing a certain degree of incentive compatibility. In fact, a bidder cannot acquire a reservation with a high-valued bid, and then iteratively try to resubmit lower bids and gain the same reservation at a lower price.

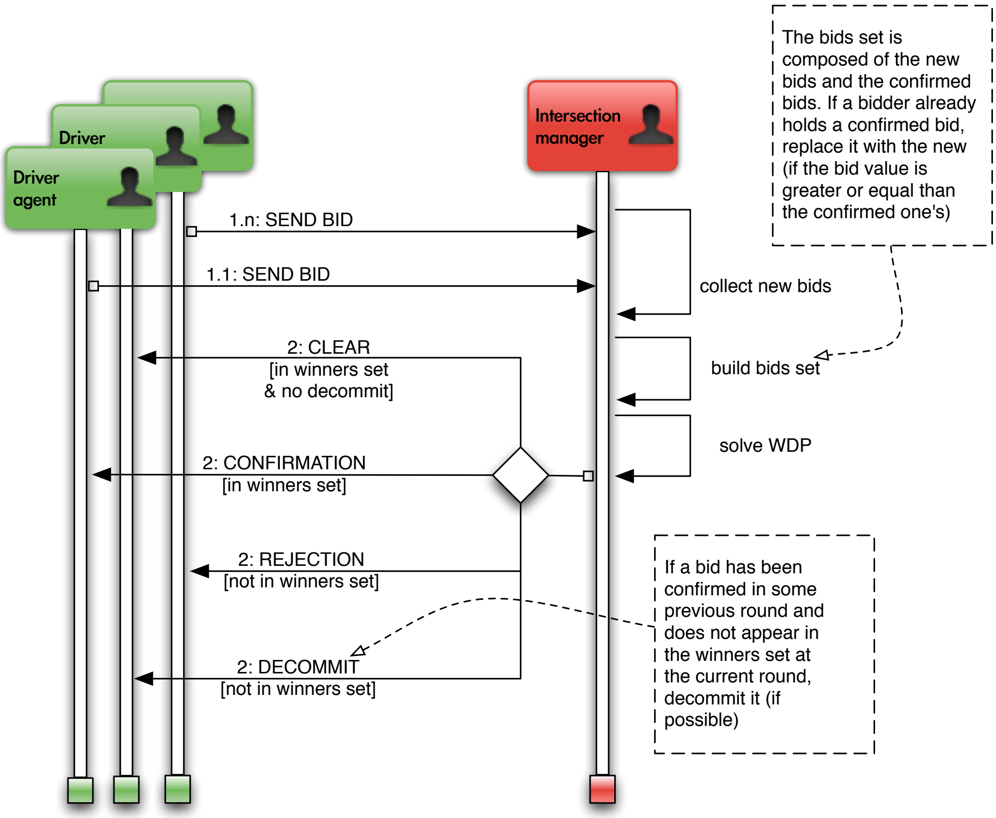


Figure 3.9: Auction protocol

3.3.3 Auction protocol

The auction proceeds as a continuous alternation of two phases: *bids collection* and *winner determination*. The protocol (figure 3.9) starts with the auctioneer waiting for bids for a certain amount of time. The incoming bids form the *bids set*. Then the auctioneer executes the winner determination algorithm, and the *winners set* is built. The auctioneer sends a CONFIRMATION message to all the bidders that submitted the bids contained in the winners set, while a REJECTION message is sent to the bidders that submitted the remaining bids. Then a new round begins, and the auctioneer collects new incoming bids for a certain amount of time. Once the new bids are collected, the bid set is built, as the union of the new bids and the previously confirmed (i.e., winning) bids[7]. Then the auctioneer executes the winner determination algorithm and builds the winners set. For all the bids in the winners set, the auctioneer sends a CONFIRMATION message to the respective bidder, unless such confirmation has been already sent in a previous round. For all the other bids, the auctioneer sends either a REJECTION message or a DECOMMIT message. The DECOMMIT message is sent to the bidders whose bids have been confirmed in a previous round, but at the present round they do not appear in the winners set. This mechanism avoids the situation where some low-valued bids, in the winners set at round N, impede the allocation of disputed reservations to some high-valued bid, submitted at round $N + K$.

At the end of any round, the auctioneer sends a CLEAR message to the bidders whose bids are in the winners set and cannot be decommitted (the condition that a bid must satisfy in order to not be decommitted is introduced in the section below). The bidders that receive a CLEAR message can take for granted its reservation. Notice that, in general, for driver agents approaching an intersection it is rational to treat their provisionally accepted bids as if they were cleared, as they can safely decelerate in case of a DECOMMIT.

3.3.4 Winner determination algorithm

Since the auction must be performed in real-time, both the bid collection and the winner determination phase must be time bounded, that is, they must occur within a specific time window. This implies that optimal and complete algorithms for the WDP, as those proposed by Leyton-Brown et. al. in [63] and by Sandholm in [86], are not suited for this kind of auction. An algorithm with *anytime* properties is needed, such as the stochastic local search proposed by Hoos et al. [49] that we have adapted to our scenario in order to manage the decommitment of bids. The anytime property make the solution improve with time, so that the more the time available

[7]Note that even a bidder that submitted a winning bid is allowed to resubmit a new bid, which will replace the old one. This is because a driver agent might want to change its confirmed reservation because, for instance, it realised that it is not able to actually use the confirmed reservation, due to changing traffic conditions

for execution, the better the solution it finds.

The algorithm starts initialising the set $\mathcal{B}$ containing all the bids (new ones and confirmed ones). The winners set $\mathcal{W}$ is initialised to the empty set, while the set $\mathcal{C}$ is initialised with all the confirmed bids that cannot be decommitted. A bid can be decommitted if the driver agent that submitted the bid can safely decelerate and reach zero speed before the arrival time at the intersection (which is contained in the bid, see figure 3.8). This condition is determined as follows: v_a the arrival speed, b a deceleration factor, and t_a the arrival time at the intersection. The deceleration equation is defined by $v(t) = v_a - b \cdot t = 0$, thus, the vehicle can safely reach zero speed before reaching the intersection if $t = v_a/b < t_a$.

Once the initialisation has been concluded, the algorithm executes the main loop for 1 sec. Within the main loop, a stochastic search is performed for a number of steps equal to the number of bids $\in \mathcal{B}$. The set $\mathcal{A}$, which at every step contains the candidate bids for the winners set, is initialised with the bids that cannot be decommitted, $\mathcal{C}$. Then, with probability wp (walk probability), a random bid is selected from the set of bids that are not actually in the candidate winners set $(\mathcal{B} \setminus \mathcal{A})$, while, with probability $1 - wp$, the highest and the second highest bids are evaluated. The highest bid is selected if its age (i.e., the number of steps since a bid was last selected to be added to a candidate solution) is greater than or equal to the age of the second highest. Otherwise, with probability np (novelty probability) the second highest is selected, and with probability $1 - np$ the highest is selected. Once the bid b to be added to the candidate solution has been selected, the neighbourhood of b, $\mathcal{N}(b)$, is evaluated. The neighbourhood of a bid b is defined by the set of bids over bundles that share with b at least one item. If the neighbourhood $\mathcal{N}(b)$ does not contain any bids that cannot be decommitted, the bid b is added to the candidate solution $\mathcal{A}$ and all the neighbours of b are removed from $\mathcal{A}$. Finally, if the value of $\mathcal{A}$ (i.e., the sum of the bids $\in \mathcal{A}$) is greater then the value of the best-so-far winners set, $\mathcal{W}$, the best solution found so far is updated.

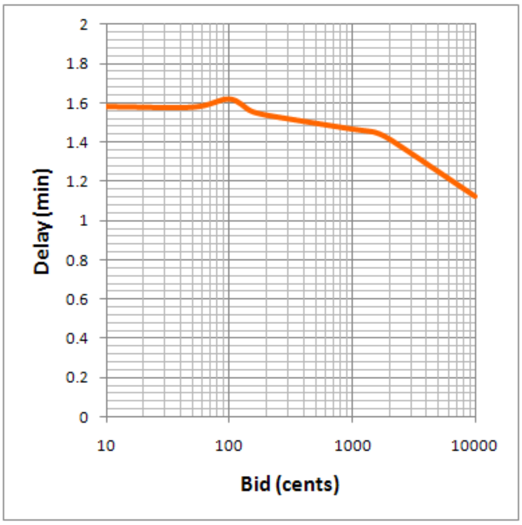
Algorithm 1 Winner determination algorithm

```
B ← allBids
W ← ∅
C ← notDecommitBids
start ← currentTime
while currentTime − start < 1 sec do
  A ← C
  for step = 1 to |B| do
    step ← step + 1
    random ← drawUniformDistribution[0 − 1]
    if random < wp then
      b ← selectRandomlyFromB \ A
    else
      highest ← selectHighestFromB \ A
      secondHighest ← selectSecondHighestFromB \ A
      if highest.age ≥ secondHighest.age then
        b ← highest
      else
        random ← drawUniformDistribution[0 − 1]
        if random < np then
          b ← secondHighest
        else
          b ← highest
        end if
      end if
    end if
    if N(b) ⋂ C = ∅ then
      A ← A ⋃ {b} \ N(b)
      if A.value > W.value then
        W ← A
      end if
    end if
  end for
end while
```

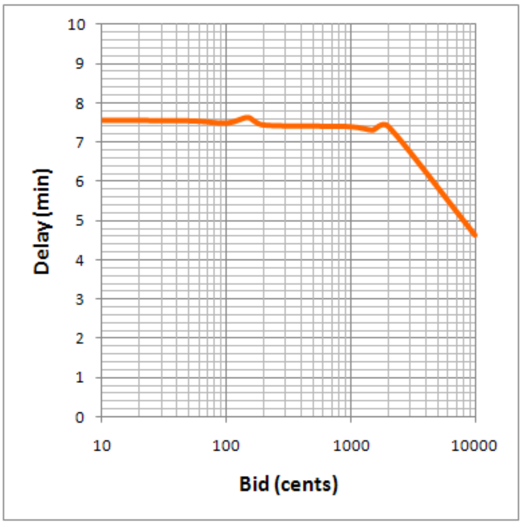
Experimental results. To evaluate the auction-based policy, we simulated again a single intersection with 4 incoming links of 3 lanes each (see figure 3.2). We generated the same traffic demands of the experiments of section 3.2 (see table 3.1). As a baseline, we used the reservation-based intersection with the FCFS policy.

The main goal of this set of experiments was to test whether or not the auction-based policy enforces an inverse relation between money spent by the bidders and delay. In other words, a bidder that is willing to bid high to acquire a reservation should experience a lower travel time. A crucial aspect of these experiments is the generation of the initial endowments of the driver agents. We performed two kinds of experiment: in the first one, we generated an artificial population of bidders whose initial endowment is uniformly distributed between $(0, 2000]$ cents; in the second experiment, we used a normal distribution with mean 100 cents and variance 25 cents to draw the initial endowment. In this population, we inserted a set of driver agents, which we used as *floating cars* to evaluate their delay, endowed with 10, 50, 100, 150, 200, 1000, 1500, 2000 and 10000 cents. Finally, we also evaluated the auction-based policy with respect to the "usual" metrics, that is, the average delay and the average rejected requests. For this last experiment, we used a population of driver agents with normally distributed endowment (mean 100 and variance 25 cents). In all the experiments, we set the walk probability wp and the novelty probability of the winner determination algorithm to 0.15 and 0.5 respectively. These values have been selected because they gave the best results in the original paper of Hoos et. al. [49] for the same kind of auction (number of bidders, expected size of bundles) that we expected in our simulated scenario.

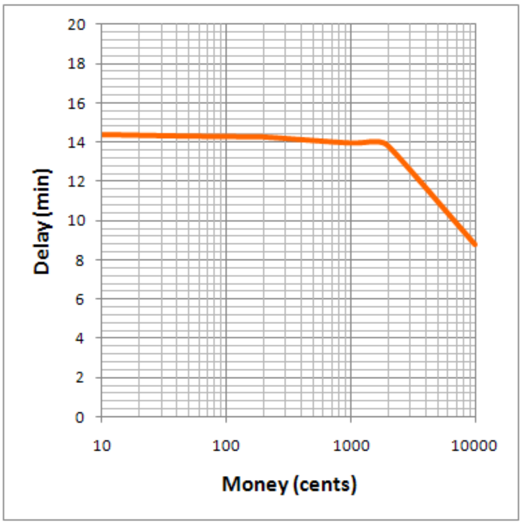
Figure 3.10 plots (in logarithmic scale) the relation between travel time and bid value for different values of λ, using initial endowments uniformly distributed between $(0, 2000]$. The general trend is that only the driver agents that bid more than the average bid (i.e., more than 1000 cents) experience a significant reduction of the delay. This reduction goes from a 29% for low traffic demand ($\lambda = 10$) up to a 39% for high traffic demand ($\lambda = 30$). On the other hand, the "average bidders" (i.e., those bidders whose bid falls in the $(0, 1000]$ interval) do not experience any significant delay reduction.



(1) $\lambda = 10$



(2) $\lambda = 20$

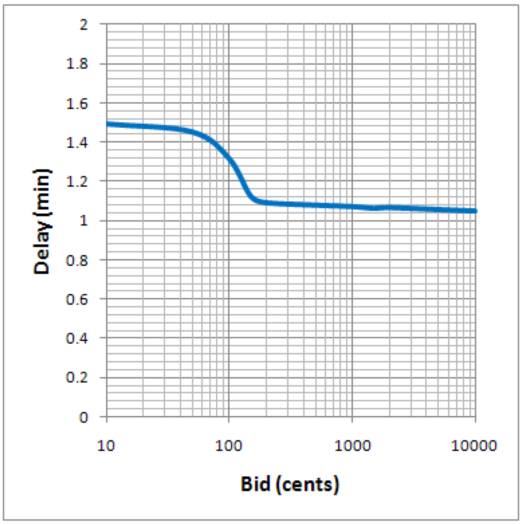


(3) $\lambda = 30$

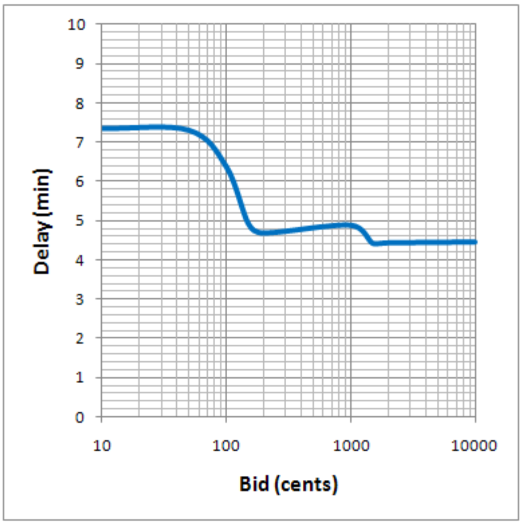
Figure 3.10: Bid-delay relation for various values of λ and uniformly distributed endowments

The uniform distribution we used in the aforementioned experiments is quite questionable. In fact, some studies noticed that the willingness to pay is usually normally (or log-normally) distributed [47]. Figure 3.11 plots (in logarithmic scale) the relation between travel time and bid value for different values of λ, using initial endowments normally distributed with mean 100 and variance 25. The results are somewhat similar to the former experiment. In fact, there is a sensible decrease of the delay experienced by the driver agents which bid from 100 to 150 cents, that is, the 49.8% of driver agents whose bid is greater than the mean bid. Still, such delay reduction tends to settle for driver agents that bid more than 150 cents. Thus, having a potentially infinite amount of money cannot guarantee zero delay, and if a driver agent wants to minimise the delay, it must to make sure to belong to the "right" side of the normal distribution of the driver agents' endowment.

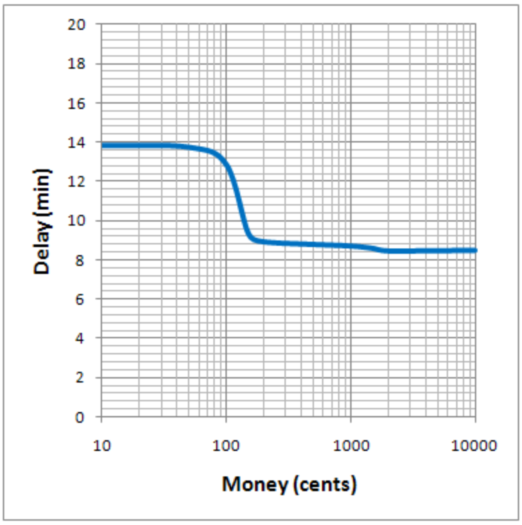
Figure 3.12 plots the average delay for different traffic demands ($\lambda \in [1, 30]$). When traffic demand falls between 1 and 15 expected vehicles per minute ($\lambda \in [1, 15]$), the performance of the combinatorial auction policy (CA) and the first-come-first-served policy (FCFS) is approximately the same. Still, when traffic demand increases (around $\lambda \geq 20$), the CA policy performs worse than the FCFS, with a noticeable increase of the average delay. This was somewhat expected, because the CA policy aims to grant a reservation to the driver agent that values it the most, rather than maximising the number of granted requests. Thus, a bid b, whose value is greater than n bids that share some items with b, is likely to be selected in the winners set. If so, only 1 vehicle will be allowed to transit, while n other vehicles will have to slow down and try again. This fact is highlighted also by the average rejected requests (figure 3.13). Since all the non-winning bids are rejected, the number of rejected requests with the CA policy is up to four times greater than with the FCFS policy.



(1) $\lambda = 10$



(2) $\lambda = 20$



(3) $\lambda = 30$

Figure 3.11: Bid-delay relation for various values of λ and normally distributed endowments

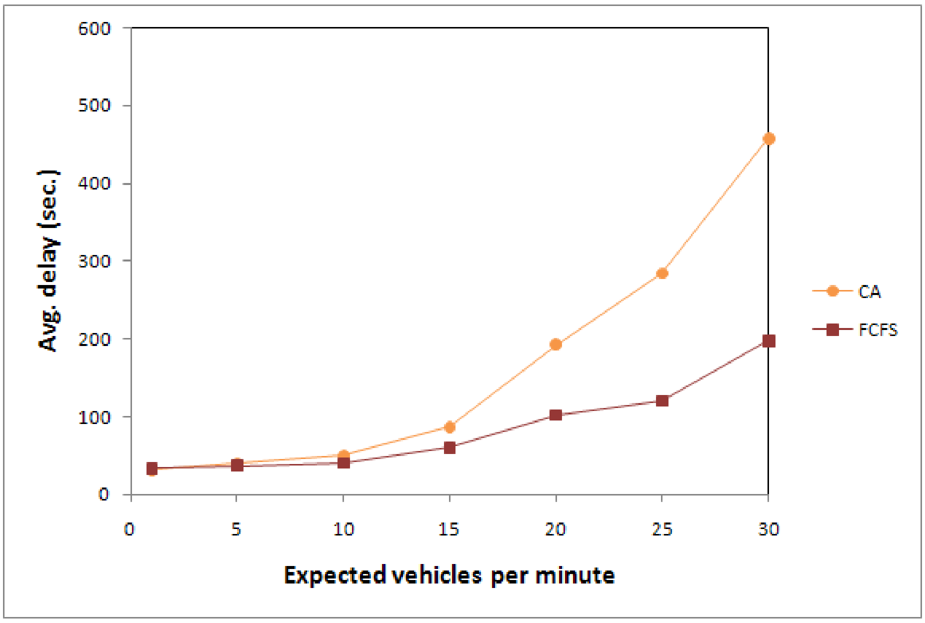


Figure 3.12: Average delay

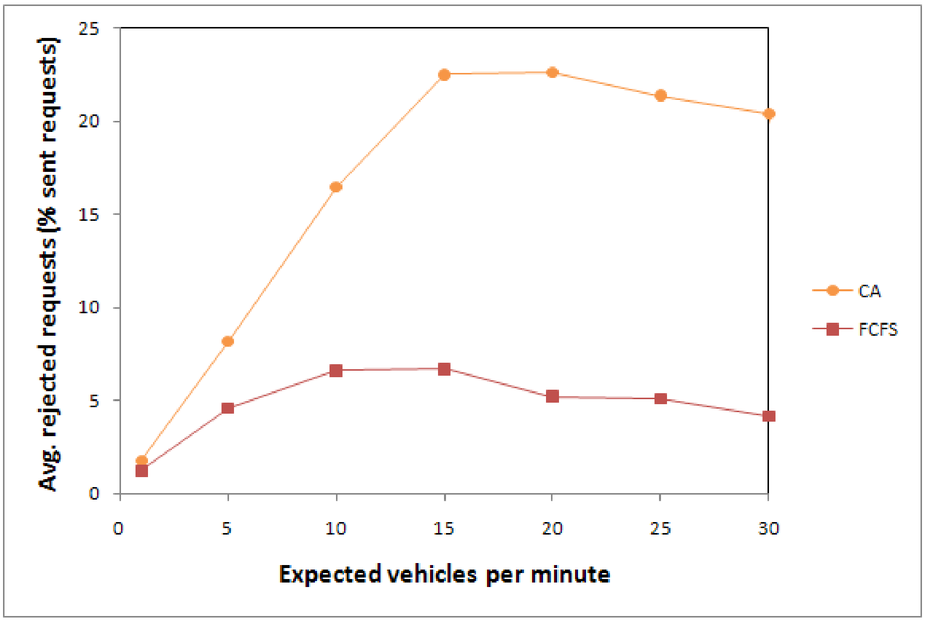


Figure 3.13: Average rejected requests

3.4 Discussion

As seen in the experimental results of section 3.2, the reservation-based intersection drastically reduces the average delay when the traffic demand is low: the vehicles are able to cross the intersection unhindered, the time spent at the intersection queue is almost constant, and the allocation capacity of the intersection is maximised.

Still, when the demand increases, the performance of reservation-based intersection converges to the performance of an intersection controlled by traffic lights. This is because a reservation-based intersection is less robust than traffic lights and its performance is very sensitive to traffic demand. When many vehicles are approaching the intersection, the correct arrival time at the intersection becomes harder to estimate and more sensitive to traffic variations, therefore many confirmed requests are withdrawn by the vehicles, thus reducing the intersection throughput.

The reservation-based intersection also has an impact on the traffic flow pattern. Although the average delay increases with the traffic density, the reservation-based intersection drastically reduces the time spent by the vehicles at the intersection queue, especially in worst case situations: the queue time with high traffic demand ($\lambda \in [20, 30]$) is four times lower with the reservation-based intersection. These two metrics suggest that a reservation-based intersection produces a slower, smoother, flow through the intersection, where the vehicles spend less time stuck at the intersection.

Assessing if this kind of travelling is preferable for the drivers is not an easy task, since each valuation of the quality of a journey is very private and hard to model in simulation: some people are exclusively concerned with the travel time, others (including the author of this thesis) prefer slightly longer routes if the journey is smooth, with no queues, less acceleration/deceleration. Surely this type of travelling is preferable from the environmental point of view, since travelling at a constant pace helps to save fuel and reduce pollution.

Regarding the different reservation-based policies, the main result is that in spite of its simple behaviour, the FCFS performs rather well in all the situations. Although in theory FCFS could be quite inefficient in some extreme cases (such as that introduced at the beginning of this chapter), in practice such extreme cases rarely occur. Thus, if we focus on improving the intersection efficiency, FCFS can be considered the best choice, since it needs less information with respect to other policies: SIS and LIS need to know when the vehicle joined the system, while NTS and FTG need the information about where the vehicle is coming from and where is going to.

In this respect, it is interesting to analyse if the additional information that the AQT-based policies need can be manipulated by malicious driver agents. Two policies, FTG and NTS, need *spatial* information, i.e., the identifier of the destination (FTG) and the origin (NTS) locations. Such information can be provided exclusively by the driver agent, because it is the only one that knows where it wants to go and where it comes from. Consequently, malicious agents could exploit the policy, providing every intersection they contact with the farthest location from the intersection they are approaching, in case of FTG, or the closest location to the intersection they are approaching, in case of NTS. The other two policies, LIS and SIS, need *temporal* information, i.e., the time stamp when the vehicle joined the system. In this case is it possible to set up a mechanism to avoid manipulation by malicious agents. For example, when the driver agent starts up, it has no time stamp, so the very first reservation that it will request won't have any time stamp. The intersection manager, detecting that the request has no time stamp, could manage the request with a default time stamp (e.g., the current time) and then it could "stamp" the driver agent with this time stamp so that, for the rest of its journey, it will provide a good approximation of the time when its joined the system. This fact makes the LIS policy very interesting, because it can be implemented with not much more effort compared to FCFS, it does not need extra information that can be manipulated by the driver agent and, as shown in the above experiments, it performs better than the FCFS in reducing delays and queue time, specially in high-load situations.

The principle of optimising the use of the available resources is not the unique guiding principle of a traffic controller. In the real world, depending on the context and their personal situation, drivers value the importance of travel times and delays quite differently. Thus, it makes sense to elaborate control policies that are aware of these different valuations and that reward the driver agents that value the disputed resources the most. In this respect, we evaluated a control policy for reservation-

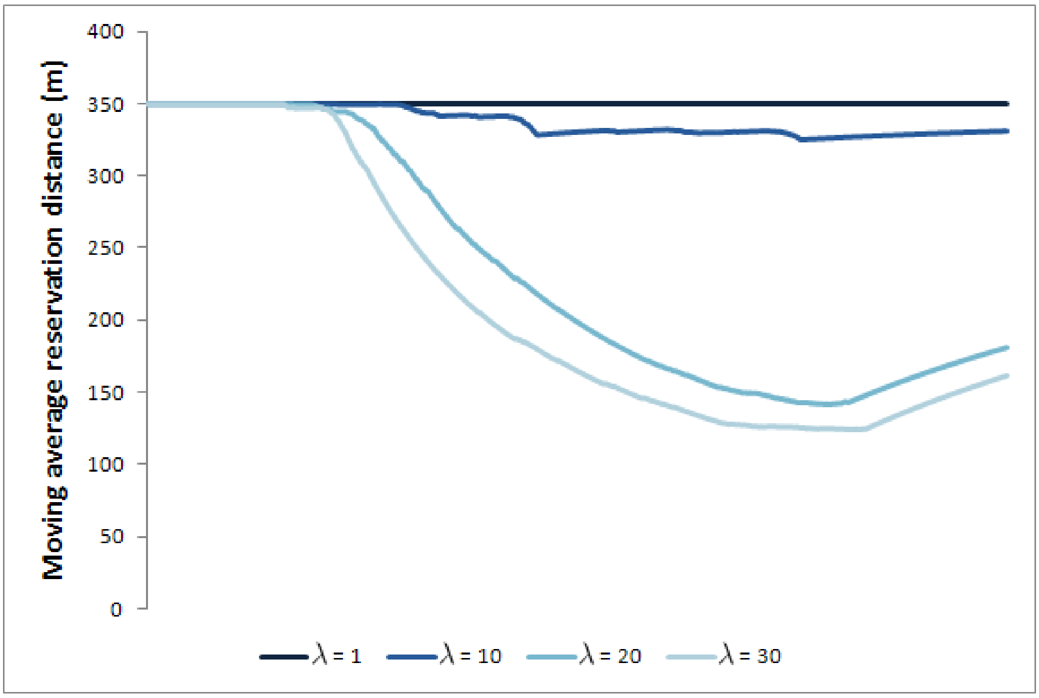


Figure 3.14: Average reservation distance

based intersections that relies on an auction mechanism. With such a policy, driver agents that bid high usually experience a great delay reduction (about 30% less with respect to driver agents that submit low bids). However, since the objective of this policy is not maximising the number of granted reservations, it pays a social costs, in the form of greater average travel times.

It is also worth noting how it is possible that a driver agent, even with a theoretically infinite amount of money, cannot experience zero delay when approaching an intersection. This is because a realistic traffic scenario is quite different from "synthetic" auctions that have been set-up for benchmarking purposes [49]. The auctions that arise in the traffic scenario are affected by the high level of dynamism, uncertainty and noise intrinsic to the domain. For example, in high load situations, the reservation distance plays an important role, since it filters out many potentially winner bids coming from a greater distance (figure 3.14 plots the average reservation distance variation over time for different traffic demands). These wealthy driver

	λ			
	1	10	20	30
# of resubmitted bids	0.2	0.66	1.4	2.32

Table 3.2: Average resubmitted bids

agents cannot even participate in the auction and acquire a reservation, so that they have no other choice other than to reach the reservation distance (thus increasing their travel time).

The estimation of the arrival time also affects a lot the performance of the auction. In fact, in high load situations, such estimation is much more noisy and uncertain, and it is likely that a driver agent must resubmit a reservation request with the updated arrival time (table 3.2 shows the average number of bids that each driver agent, which holds a winner bid, resubmits as a result of a new estimation of the arrival time). In this way, it is possible that an agent wins an auction at time t and then, due to a new estimation of the arrival time, must resubmit its bid at time $t + \Delta t$. The bidders that participate in the auction at time $t + \Delta t$ are obviously different from those that participate at time t, so there is no guarantee that the agent might win the auction again.

Finally, we remark that our specific scenario limits the auction design space. For example, open-outcry auctions (such as the English auction) are not time-bounded (an auction ends when all the bidders stop raising their bids), which makes them hardly applicable in a highly dynamic and safety-critical scenario such as intersection crossing. Furthermore, we addressed only the winner determination problem of the combinatorial auction, while for the payments calculation we did not adopt any sophisticated method. i.e., a winner pays a price that is exactly the bid that he submitted. This, as any first-price payment mechanism, could in principle leads to malicious behaviours, with driver agents that try to acquire reservations by submitting bids that are lower than the real valuations they have. In single item auctions it is computationally easy to set up an incentive compatible payment mechanism, such as the second-price (or Vickrey) mechanism. In this kind of auction, the winner (which is simply the bidder who placed the highest bid) pays a price equal to the second highest bid. In this way a bidder has no incentives to lie and to not submit his real valuation of the good. In fact, if the bidder who has the highest valuation submits a bid that is lower than his real valuation, but greater than the second highest, he still wins the auction but pays the same amount (i.e., the second highest bid) that he would have paid if he had submitted the real valuation. On the contrary, if the bid that he submits is too low, his bid becomes the second highest and he looses the auction. Unfortunately, it is not straightforward to extend these mechanisms to the combinatorial auctions, since they are computationally harder the single item ones. As we have seen in section 3.3, even the winner determination is a NP-hard problem, and so is any payment mechanism that is equivalent, from the incentive compatibility point of view, to the Vickrey mechanism. In fact, the generalisation of the Vickrey mechanism in the combinatorial world is the Vickrey-Clarke-Groves (VCG) payment mechanism [23][45][102]. This mechanism charges each bidder the harm they cause to the rest of the world, that is, a price equal to the total amount better off everyone else would be if this bidder would have not participate in the auction. Although VCG mechanisms enforce dominant truth-revealing strategies, they have many serious practical problems (non-existence of dominant strategy equilibria, NP completeness, revenue deficiency, etc.) which make VCG mechanisms of limited practical value [84].

Therefore, although a driver agent could potentially acquire a reservation by submitting a bid $\widehat{b}$ that is lower than its real valuation b, from a practical point of view this affects exclusively the revenues that the auctioneer should gain if every bidder was truth-telling, which is not our primary concern. Another possible weakness is the fact that a bidder could start bidding lower than his real valuation and than raising his bid if is not able to acquire it, thus leading to a communication overhead between bidders and auctioneer. Still, only the bidders in the proximity of an intersection (and within the reservation distance) are able to submit a bid, thus the number of bids that the intersection manager may receive simultaneously is necessarily bounded.

Chapter 4

Network of intersections

Never underestimate the bandwidth
of a station wagon full of tapes
hurtling down the highway.

Andrew S. Tanenbaum

In the previous chapter we analysed the performance of different policies that can be applied to the management of a *single intersection*. If we focus on a *network of intersections*, the scope of the management policy increases as well as the size and the complexity of the management task. Thus, an integrated strategy is needed, which acts not only on the *traffic control*, but also on the *traffic assignment*. Traffic assignment refers to the problem of the distribution of traffic in a network, considering demands between several locations, and the capacity of the network. Traffic assignment strategies aim at making easier the task of the traffic controllers, by means of a better distribution of the traffic demand.

In this chapter, we first introduce the simulator that we will use to empirically evaluate our models (section 4.1), as well as the model of driver agent that we use in our simulations (section 4.2). In section 4.3, we will analyse how the policies based on the adversarial queueing theory, which have been described in section 3.2, perform at network level. Then, we will study two computational markets for the traffic assignment (section 4.4): in the first model, the intersection managers act as cooperative learning agents that learn which pricing policy optimises a global profit function and, indirectly, the average travel time of the population of driver agents; in the second model, the intersection managers compete with each other for the supply of the resources that are traded, i.e., the reservations at the intersections, dynamically adapting the price of the reservations in order to match the supply with the current demand. Finally, in section 4.5 we will propose an integrated market model that combines the competitive mechanism of section 4.4, which deals with the traffic assignment, with the auction-based policy of section 3.3, which acts on the traffic control at intersection level.

4.1 $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ - Multiagent Intelligent Transportation Environment

4.1.1 Desiderata

In developing a traffic simulator to pursue our researching goals, we aim to satisfy the following list of properties.

Accuracy. Traffic is an emergent phenomenon, the result of the individual decisions of drivers, traffic controllers and all the actors that are part of the system. Agent-based modelling helps building simulated models with detailed, rich behaviours for individual entities. For our purposes, each agent of the system must have a certain degree of autonomy (especially the driver agent) and it must be possible to program their behaviour and decision making.

Integration with control systems. Borrowing the terminology of the theory of control, the simulator must clearly separate the "plant" (i.e., the vehicles, the road links, the intersections . . .) from the control system, to enable experimentation with different control mechanisms.

Large-scale simulation. In our research we focus on management mechanisms for (possibly) very large systems, with thousands of vehicles travelling through the road network and dozens of infrastructure agents that aim to control the system. The simulator must be able to simulate a great number of entities with high accuracy and good computational performance.

Realistic dynamics. The simulator must employ validated traffic models to simulate the dynamics of the vehicles. Developing a new model of traffic is outside the scope of this thesis (indeed, it could be a thesis topic), thus we will adopt a validated model that we can find in literature.

Typically, traffic simulation models can be divided into two classes, depending on the granularity of the modelling of the traffic phenomena: *macroscopic* and *microscopic*. Macroscopic models [29][107][119] represent the traffic with macroscopic flow/density/speed functions, without taking into account individual decision making. The macro models are usually derived from fluid dynamics, and they involve aggregate parameters such as traffic volume and average speed on road links. Simulations based on macroscopic models have the advantage that run-time can be fairly short, as the computation is based on aggregate, abstract parameters. Macroscopic simulation can provide adequate results for applications that do not require a high degree of accuracy in the results. Nevertheless, more precise information about queues, flows, speeds and densities than those that can be provided by a macroscopic simulation model is normally required.

Microscopic models treat the traffic flow as the result of the interactions between agents, representing the individual vehicle behaviour. A microscopic model usually includes a car-following model [68][98][112], a lane-changing model [97] and other models for yielding and merging. A discretisation of time is normally used, and at each time-step, behavioural choices are calculated and vehicles' positions updated as necessary. Microscopic simulation models are theoretically able to provide such required level of accuracy, nevertheless they face several significant obstacles. The first obstacle in microscopic traffic simulation is the complexity of the models describing the drivers decisions. Driver's behaviour is complex to model, and the models must be simplified in order to reduce the computational burden. This shortage of accurate models is probably the most significant drawback of microscopic simulation. The increased complexity of modelling is not compensated for by equally accurate results. Furthermore, due to the level of detail inherent with microscopic simulation models, the computational resources necessary to simulate a large urban area consisting of numerous streets and roads is very high. Network geometry, topology, socio-demographic data, trip data, historical data, etc. consume very large amounts of memory, as the number of modelled network elements increases. Furthermore, the number of concurrent vehicles that are in a network can be very large, and the memory and CPU power required to be able to move all those vehicles is highly significant.

Mesoscopic traffic simulation is valid as intermediate solution between microscopic and macroscopic models. Mesoscopic simulations can model the network and the vehicle movements at the same level of detail of microscopic simulations. However, because the driver behaviour is highly simplified and the vehicle dynamic is determined by macroscopic calculations, it is possible to model larger areas and move more vehicles than it would be possible with a microscopic simulation. With mesoscopic traffic simulation, it is possible to provide results at a level of significance close to those available with microscopic simulation, while gaining in simulation speeds, reducing resource constraints, and simplifying the modelling work.

4.1.2 Simulator

In order to empirically evaluate the different management mechanisms, we developed a custom traffic simulator called $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$, which stands for Multiagent Intelligent Transportation Environment. $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ is a time-discrete hybrid mesoscopic-microscopic simulator, based on two different models of traffic flow: DYNEMO, the mesoscopic model by Thomas Schwerdtfeger [89], and the microscopic model by Kai Nagel and Michael Schreckenberg [68]. The mesoscopic model simulates the traffic flow *on the road links*, while the microscopic model simulates the traffic flow *inside* *the intersections.* We adopted this solution because the mesoscopic model allows us to simulate large-scale systems, with thousands of vehicles moving in the road network, while the microscopic model permits the implementation of fine-grained control policies inside the intersections.

Mesoscopic model

DYNEMO [89] is a member of the Payne-Cremer [78] family of traffic flow models, which had previously been demonstrated to be suitable for the assessment of ITS strategies [27]. The main characteristic is that the "atomic unit" of traffic flow is the individual vehicle rather than the temporal and spatial aggregates used in macroscopic models. Still, the vehicle dynamic is governed by the average traffic density on the link they traverse rather than the behaviour of other vehicles in the immediate neighbourhood as in microscopic models.

Network representation. The network is divided into *stretches* with similar characteristics (number of lanes, speed limits, etc.), which in turn are divided into *sections* of typically 500 meters length, for which constant traffic conditions are assumed. The traffic condition on a section S_i is defined by its traffic density ρ_i (expressed in $vehicles/km$). A section is characterised by a relationship between density and mean speed, $U_i : \rho \rightarrow \mathbb{R}$, and a distribution of speeds at free flow, V_i. U_i is called the *speed-density* function and V_i is called the *desired speed distribution.*

Vehicle representation. A vehicle is described by its position within the section where it is driving, $x_j \in [0, length(S_i)]$, its speed, v_j, and a parameter that defines the proportion of vehicles with lower free flow speed, $\delta_j \in [0, 1]$. The parameter δ_j is used to determine the desired speed of the vehicle, v_{δ_j}, given the section minimum and maximum desired speed, u_i^{min} and u_i^{max} respectively.

Vehicle dynamic. A basic assumption of the model is that the individual speed of vehicles in section S_i, at a given traffic condition ρ_i, varies within the interval defined by $[\underline{u}_i, \overline{u}_i]$, where $\underline{u}_i$ and $\overline{u}_i$ are respectively the lower bound and the upper bound of the section mean speed u_i. The individual speed of the vehicle in this interval depends on its desired speed v_{δ_j}, e.g. if $v_{\delta_j} = u_i^{min}$, the vehicle will set its speed to $\underline{u}_i$, whilst if $v_{\delta_j} = u_i^{max}$, the vehicle will set its speed to $\overline{u}_i$.

Under this assumption, the *reference speed* of section S_i for vehicle j is calculated as:

$$\widehat{u}_i = \underline{u}_i + \frac{v_{\delta_j} - u_i^{min}}{u_i^{max} - u_i^{min}} \cdot (\overline{u}_i - \underline{u}_i) \tag{4.1}$$

Given the reference speed of the section where vehicle j is driving, S_i, and the reference speed of the next section on the vehicle's route, S_{i+1}, a vehicle *target speed* is computed as:

$$\widehat{v}_j = (1 - \frac{x_j}{length(S_i)}) \cdot \widehat{u}_i + \frac{x_j}{length(S_i)} \cdot \widehat{u}_{i+1} \tag{4.2}$$

In other words, the influence of the next section's reference speed on the new speed of the vehicle grows as the vehicle approaches the end of its current section. In the limit, the new target speed depends only on the average speed downstream when the vehicle reaches the section boundary.

To compute $\widehat{u}_i$, it is necessary to define the lower and upper bounds of u_i, namely $\underline{u}_i$ and $\overline{u}_i$. The lower bound is computed as:

$$\underline{u}_i = \begin{cases} u_i^{min} - \dfrac{u_i^{min} - u_i^{opt}}{\rho_i^{opt}} \cdot \rho_i & \text{if } \rho_i < \rho_i^{opt} \\ u_i & \text{if } \rho_i \geq \rho_i^{opt} \end{cases} \tag{4.3}$$

where ρ_i^{opt} is the density that maximises the traffic flow (ϕ_i^{max}), and u_i^{opt} is the corresponding mean speed.

The upper bound is computed as:

$$\overline{u}_i = \begin{cases} \underline{u}_i - \dfrac{u_i^{max} - u_i^{min}}{\mathbb{E}[V_i] - u_i^{min}} \cdot (u_i - \underline{u}_i) & \text{if } \rho_i < \rho_i^{opt} \\ u_i & \text{if } \rho_i \geq \rho_i^{opt} \end{cases} \tag{4.4}$$

where $\mathbb{E}[V_i]$ is the expected value of the desired speed distribution V_i.

Once the *target speed* has been computed, the vehicle speed at time $t + \Delta t$ is updated using the following formula:

$$v_j^{t+\Delta t} = \begin{cases} min\left(v_{\delta_j}\ ,\ v_j^t + b_{max} \cdot \dfrac{\widehat{v}_j - v_j^t}{u_i^{max}} \cdot \Delta t\right) & \text{if } \widehat{v}_j > v_j^t \\ max\left(\widehat{v}_j\ ,\ v_j^t + b_{min} \cdot \dfrac{v_j^t - \widehat{v}_j}{u_i^{max}} \cdot \Delta t\right) & \text{if } \widehat{v}_j \leq v_j^t \end{cases} \tag{4.5}$$

where b_{min} and b_{max} are the minimum and maximum possible acceleration factors of a vehicle (set to $-7\ m/s^2$ and $4\ m/s^2$ respectively).

Finally, given the new speed at time $t + \Delta t$, the vehicle's position is updated to:

$$x_j^{t+\Delta t} = x_j^t + \frac{1}{2} \cdot \left(v_j^t + v_j^{t+\Delta t}\right) \cdot \Delta t \tag{4.6}$$

If $x_j^{t+\Delta t} > length(S_i)$, the vehicle enters the next section and the position is reset to $x_j^{t+\Delta t} - length(S_i)$.

Microscopic model

In 1992, Nagel and Schreckenberg proposed a stochastic traffic cellular automata (TCA) model that was able to reproduce several characteristics of real-life traffic flows, e.g., the spontaneous emergence of traffic jams. TCA models arise from the discipline of statistical mechanics, having the goal of reproducing the correct macroscopic behaviour based on a minimal description of microscopic interactions. The main advantage of TCAs is that they are efficient and fast performing when used in computer simulations, due to their rather low accuracy on a microscopic scale.

The Nagel-Schreckenberg TCA model uses a rectangular lattice, with a cell size of 7.5 meters for freeway traffic and 5 meters for urban traffic, and it comprises the following three rules for the update of the vehicle position and speed:

R1: acceleration and braking

$$v_i(t) \leftarrow min\left\{v_i(t-1) + 1, g_{s_i}(t-1), v_{max}\right\}$$

where $v_i(t)$ is the speed at time t, $v_i(t-1)$ is the speed at time $t-1$, $g_{s_i}(t-1)$ is the spatial gap in front of the vehicle, and v_{max} is the maximum speed.

R2: randomisation

$$\xi(t) < p \Rightarrow v_i(t) \leftarrow max\{0, v_i(t) - 1\}$$

where $\xi(t) \in [0, 1)$ is a random number uniformly distributed and $p \in [0, 1]$ is the slowdown probability.

R3: vehicle movement

$$x_i(t) \leftarrow x_i(t - 1) + v_i(t)$$

where $x_i(t)$ and $x_i(t-1)$ are the vehicle positions at time t and $t-1$ respectively.

The TCA contains a rule for increasing the speed of a vehicle and braking to avoid collisions (rule R1), as well as a rule for the the actual vehicle movement (rule R3). However, the TCA also contains an additional rule R2, which introduces stochasticity in the system. At each time-step t, a random number $\xi(t)$ is drawn from a uniform distribution. This number is then compared with a stochastic noise parameter p (the slowdown probability); as a result, there is a probability of p that a vehicle will slow down to $v_i(t) - 1$ cells/time-step. The Nagel-Schreckenberg TCA model is called a minimal model, in the sense that all these rules are necessary for mimicking the basic features of real-life traffic flows.

Since we use the microscopic model exclusively for simulating the transit of vehicles through a rather small area (e.g., an intersection with 4 incoming links of 3 lanes each results into a 6×6 lattice), we further simplified the model assuming that the vehicles cross the intersection *at constant speed*. Thus, we don't apply either rule R1 or rule R2, but we simply update the vehicle position inside the intersection using rule R3.

Network structure

The road network in $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ is represented by a directed multi-graph that consists of vertices and edges. There are two different types of vertices:

Section vertex. A section vertex is used to split the stretches into section of ~ 500 meters length.

Stretch vertex. A stretch vertex is a point where where multiple traffic streams join or diverge, such as intersections, as well as origins or destinations of traffic.

The edges represent the lanes of the links between such vertices, and are unidirectional. This means that a street that links vertex A with vertex B, with three lanes for each direction, is represented by 6 edges, 3 from A to B and 3 from B to A.

Each stretch vertex has a queue for each incoming edge, which is used as a buffer area for the vehicles that must switch from the mesoscopic model to the microscopic model. When the updated vehicle position $x_j^{t+\Delta t}$ becomes greater than the length of the section where the vehicle is driving and the end of the section is connected to a stretch vertex (see equation 4.6), the vehicle is put at the back of the queue. It remains in the queue until it reaches the front, then it is placed in the first cell of the lattice of the microscopic model. From then on its dynamic is simulated using the Nagel-Schreckenberg model described in section 4.1.2. Figure 4.1 shows an example of a network, with all the elements composing it.

Speed-density functions

A central element of the mesoscopic model is the *speed-density* function, i.e., the relationship between density and mean speed of a section. This speed-density function can take different forms. The oldest known formulation of a speed-density function is by Greenshields [44]:

$$U_i(\rho) = u_i^{free} \cdot \left(1 - \frac{\rho_i}{\rho_i^{max}}\right) \tag{4.7}$$

where ρ_i is the density of section S_i, u_i^{free} is the speed at free flow on that section, and ρ_i^{max} is the maximum density of that section. Since u_i^{free} and ρ_i^{max} are constants, this leads to a linear relationship between speed and density.

Other important speed-density functions are by Greenberg [43]:

$$U_i(\rho) = u_i^{opt} \cdot ln\left(\frac{\rho_i^{max}}{\rho_i}\right) \tag{4.8}$$

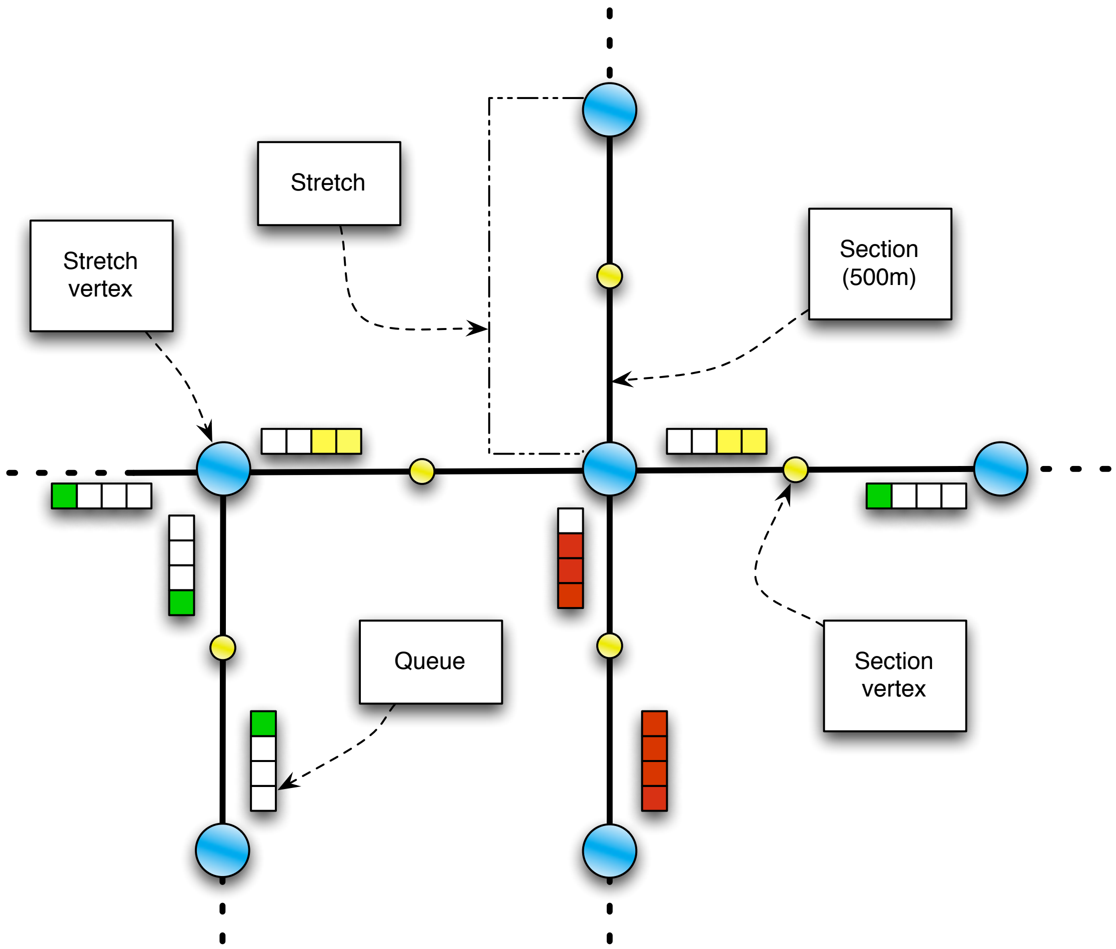


Figure 4.1: Network structure

where u_i^{opt} the speed that maximises the traffic flow (vehicles per time unit that leaves the section).

Underwood [100]:

$$U_i(\rho) = u_i^{free} \cdot e^{\left(-\frac{\rho_i}{\rho_i^{opt}}\right)} \tag{4.9}$$

where ρ_i^{opt} is the density that maximises the traffic flow.

and Drake [34]:

$$U_i(\rho) = u_i^{free} \cdot e^{\left(-\frac{1}{2}\cdot\left(\frac{\rho_i}{\rho_i^{max}}\right)^2\right)} \tag{4.10}$$

In $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ we used the same speed-density function used in DYNEMO[89]:

$$U_i(\rho_i) = \begin{cases} u_i^{free} & \text{if } \rho_i = 0 \\ u_i^{free} - \left(\dfrac{u_i^{free} - u_i^{opt}}{\rho_i^{opt}}\right) \cdot \rho_i & \text{if } 0 < \rho_i < \rho_i^{opt} \\ u_i^{opt} & \text{if } \rho_i = \rho_i^{opt} \\ \dfrac{\phi_i^{max} \cdot \rho_i^{opt}}{\rho_i^2} & \text{if } \rho_i > \rho_i^{opt} \end{cases} \tag{4.11}$$

where u_i^{free} is the speed at free flow on section S_i, ρ_i^{opt} is the density that maximises the traffic flow, ϕ_i^{max}, and u_i^{opt} is the corresponding mean speed.

Traffic generation

The demand is represented by an Origin/Destination matrix, which specifies for each time the trips (i.e., a pair of IDs of the origin and destination stretch vertices) that must be generated and, for each trip, the driver agent that operates the vehicle. The driver agent defines the behaviour of the vehicle with respect to the decision it must face, such as the route choice. The Origin/Destination matrix is stored as a xml file and loaded by the simulation engine before the simulation starts (see section 4.1.3).

Route choice

The route choice in $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ is performed by the driver agent that operates the vehicle. We can distinguish two phases: *pre-trip* route choice and *route-replanning*. The pre-trip route choice is based on a pre-calculated set of known routes, with their respective travel time at free flow and distance. The driver agent freely selects its route, according to its utility function (see section 4.2 for more details about utility functions and choice models of driver agents).

Once a driver agent has selected the initial route, it starts its journey, following that particular route. Then it may reconsider the route it has selected in the pre-trip phase and eventually it may divert from its current path. This would happen for example if information about an incident is received, or if the driver experiences an excessive delay compared to the expected travel time. When and how the driver agent applies the route-replanning depends on the driver agent itself and its behaviour.

Operational issues

$\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ is a discrete-time simulator, i.e., the state of the traffic is computed at fixed time intervals, whose duration is a parameter of the simulator. From each time-step to the next, the changes in the traffic state are calculated, i.e., the vehicles' speed and position are updated, as well as the density and mean speed of the sections. The space is *continuous*, in the mesoscopic part of the simulator, and *discrete*, in the microscopic part of the simulator. That is, in a section S_i, a vehicle can occupy every position $\in [0, length(S_i)]$, while inside the stretch vertex lattice a vehicle occupies a single cell.

Simulation inputs. $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ needs the following inputs:

- Network description. This file contains a description of the topology of the network, i.e., the vertices and the edges of the multi-graph.

  Each vertex contains the following information:

  - *vertex ID*
  - *longitude*
  - *latitude*
  - *class*
  - *[intersection topology]*

  The *class* parameter refers to the Java class that implements the vertex behaviour, since a vertex may be, for example, an intersection stretch vertex regulated by traffic lights that needs an agent that governs its behaviour (implementing a signal plan). The *intersection topology* is optional and needed only if the vertex is an intersection stretch vertex. It defines the lattice topology of the cellular automaton microscopic model, as well as how it must be integrated into the road network.

Each edge contains the following information:

- *edge ID*
- *source vertex ID*
- *target vertex ID*
- *class*
- *optimal flow*
- *optimal density*
- *maximum desired speed*
- *minimum desired speed*
- *road*

The *class* parameter refers to the Java class that implements the speed-density function (given the optimal flow, the optimal density and the maximum and minimum desired speed).

- Origin/Destination file. This file contains the trips that must be generated for the simulation.
- Simulation engine properties. This file contains some parameters needed by the simulation engine, such as the time-step of the simulation, the refresh rate of the visualisation, the initial time of the simulation, the number of shortest routes that must be calculated before the simulation starts and the message transmission delay of the vehicle-to-infrastructure communication.

  The simulation engine calculates, for every origin-destination pair, two types of shortest routes, using the free-flow travel time on the links (shortest travel time route) and the distance of the path between origin and destination (shortest distance route).

```xml
-<trial>
   <road_network_file value="./experiments/exp001/exp001-madrid-FTG.map"/>
   <od_matrix_file value="./experiments/exp001/exp001-high-traffic-R.od"/>
   <simulation_engine_file value="./experiments/exp001/exp001.properties"
   class="es.urjc.cetinia.mite.simulation.impl.DefaultSimulationEngine"/>
 </trial>
```

Figure 4.2: Master file

The references of the necessary files can be passed to the simulator using a master file to speed up the execution. Figure 4.2 shows the xml master file, containing the network specification, the Origin/Destination file, and the simulation engine properties file.

Simulation outputs. $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ generates different text files that can be used for offline analysis.

- Vehicle metrics. This file contains the metrics of the simulated vehicles. This file contains the following information for each vehicle trip:
  - vehicle ID
  - origin
  - destination
  - trip start time
  - trip end time
  - travel time
  - covered distance
  - average speed
  - queue time

  Before the simulation execution ends, to make the offline analysis easier, the simulator calculates average and standard deviation of travel time, covered distance, average speed and queue time, grouped by origin-destination pair.

- Network metrics. This file contains information related to the network, such as density, traffic flow and average speed of sections over time. It can be used to give different snapshots of the network during the simulated time.

4.1.3 Implementation

$\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ has been implemented in Java, thus ensuring that the simulator can be executed on different platform (MS Windows, Linux and MacOS) without modifications. The program code come to more than 30 thousands lines, it has been designed with the tool MagicDraw UML[1], and developed with the Eclipse editor[2]. There are no limits to the size of the networks that can be simulated, nor to the amount of vehicles that can be simulated concurrently. A network with 300+ vertices and 1600+ edges, with 10000+ vehicles can be simulated quite a lot faster than real time on a Pentium 1.7 GHz with 1 Gigabyte RAM under Linux.

Figure A.1 (see appendix A) shows the relations between the classes *Mite*, *SimulationEngine* and *RoadNetworkGui*. *Mite* is the main class used to launch the simulator, and serves as listener of the events generated by the *SimulationEngine* and the *RoadNetworkGui*.

The *SimulationEngine* generates two types of event, one to notify that a simulation time-step has been performed and one to notify that the simulation run has terminated. Both event are processed by the *Mite* class: the former is used to refresh the graphical user interface (GUI), updating vehicle positions, traffic densities etc., the latter is used to reset the GUI and make it ready for a new simulation.

The *RoadNetworkGui* generates 4 events, triggered when the user starts, pauses, resumes or stops the simulation using the GUI buttons.

Figure A.2 (see appendix A) shows the components of the *SimulationEngine*. A *SimulationEngine* uses a *ODMatrix* to store and retrieve the *Trips* that must be generated, and a *DataCollector* to gather the data generated by the *VehicleAgents*, which drive the simulated vehicles, and by the *SensorAgents*, which are deployed throughout

[1]http://www.magicdraw.com

[2]http://www.eclipse.org

the network to collect measurements related to the sections (density, traffic flow, average speed, etc.). The *SimulationEngine* triggers all the *SimulationItems* that must be simulated and processes all the events that are generated by *SimulationItems*, such as *VehicleEvents*, *MessageEvents* or *SensorEvents*.

The main loop of the *SimulationEngine* is shown in figure A.3 (see appendix A). First, the *SimulationEngine* checks if the simulation run has finished. If so, the *SimulationEngine* is disposed. Otherwise, the *SimulationEngine* generates the scheduled trips for the actual time, creating a vehicle for each trip, driven by a *VehicleAgent*. Then, the *SimulationEngine* triggers all the *SimulationItems* to be executed (*VehicleAgents*, *SensorAgents*, *TrafficLightAgents*, etc.). Once all the *SimulationItems* have executed their task, the *SimulationEngine* processes all the *SimulationItemEvents* that have been generated during the execution of the *SimulationItems*, be they *VehicleEvents*, *SensorEvents* or *MessageEvents*. Finally, when all the *SimulationItemEvents* have been consumed, the network state is updated and the clock is moved forward, adding the time-step to the actual time.

Figure A.4 (see appendix A) shows the components of the *RoadNetwork*. A *RoadNetwork* is composed of *SectionEdges* and *RoadVertices*. The latter can be *SectionVertices* or *StretchVertices*. A *StretchVertex* may implement an *Intersection*. In this case the *StretchVertex* must be regulated by a control facility, such as traffic lights (*TrafficLightStretchVertex*) or reservation-based intersection (*ReservationStretchVertex*).

A *RoadVertex* has installed zero or more *RoadVertexAgents*, which may define the behaviour of the control facility, in case of *TrafficLightStretchVertex* or *ReservationStretchVertex*, as well as take some measurements of the incoming sections connected with the *RoadVertex*.

The *BulletinBoard* is a *SimulationItem* that can be used by *RoadVertexAgents* to share information at global level as well as by the *VehicleAgents* to gather information about the status of the *RoadNetwork* (e.g., notification of accidents).

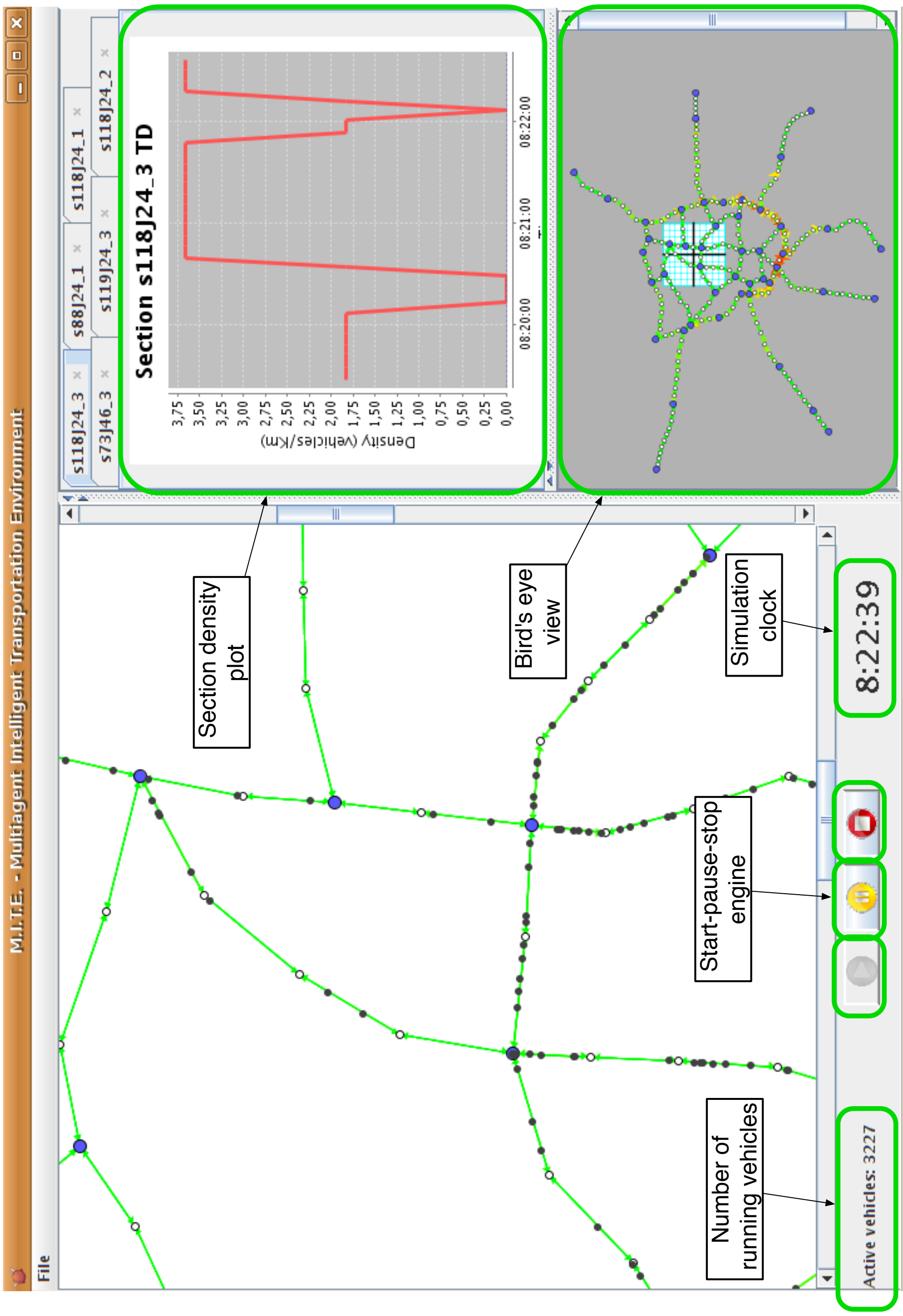


Figure 4.3: $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ graphical user interface

Graphical user interface Figure 4.3 shows the graphical user interface (GUI) of $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$. From the menu *File*, it is possible to load the files to run a simulation (network specification, Origin/Destination matrix and simulation engine properties separately or through a single master file).

In the bottom there is the control panel, with the buttons to start, pause, resume and stop the simulation. The control panel also shows the simulation clock as well as the number of vehicles that are actually travelling in the network.

In the centre, there is the network main view, which can be zoomed in/out using the mouse wheel or the keyboard shortcuts. In the bottom right-hand corner, there is the bird's eye view of the entire network. By moving the white square, it is possible to position the main view on the desired area. In the top right-hand corner, there is the tab panel where it is possible to show some plots related with the sections (density-flow relation and section density over time).

4.2 Driver agent model

The driver agent model defines the behaviour and the decision making of the agent in charge of operating a vehicle. It is one of the most important components of traffic simulation models, since the reliability of the simulation depends on the realism of the underlying behavioural model. The specification of the model, and its estimation using real-world data is a challenging task, which requires the application of behavioural principles (e.g., aggressiveness, mental state, risk aversion) and econometric techniques. In general, a driver model must take into consideration *pre-trip* decision making as well as *en-route* behaviour. Pre-trip decision making refers to all the choices that a human driver makes before starting the trip, such as the departure time selection, the route choice and the transportation mode (i.e., public transport, private vehicle, etc.). The en-route behaviour is related to the microscopic dynamic of the driver, and it encompasses the car-following behaviour, the lane-changing behaviour and all the other aspects related to driving (safety distance, aggressiveness, risk aversion, speed limits, etc.).

Defining a full-fledged driver model is outside the scope of this thesis, however, we need a reasonably realistic driver model to run our simulations. The simulation model in use is mesoscopic, therefore we do not focus on modelling a precise en-route behaviour. Furthermore, we generate the traffic demand with pre-assigned departure time, in order to focus exclusively on the most important aspect of the pre-trip decision making, the route choice. The route choice refers to the selection of the most preferred path between origin and destination, and it is a central aspect of the discrete choice models [14]. Discrete choice models have played an important role in transportation modelling over the last few years. In order to develop models that capture how individuals are making choices, four modelling aspects must be considered, namely the *decision-maker*, the *alternatives*, the *attributes* and the *decision rules*.

Decision maker

The decision-maker is the individual that makes decisions, according to his/her characteristics. The model may take into consideration several characteristics, or attributes, of the individual, such as age, gender, income, and even eye color or social security number. In our model, the decision makers are the driver agents, and we assume that they are *homogeneous*, in the sense that we do not group them by any characteristic or attribute.

Alternatives

The alternatives determine what the possible options of the decision-maker are. Analysing the choice of an individual requires the knowledge of what has been chosen, but also of what has not been chosen. The set containing these alternatives, called the choice-set, must be characterised. The choice-set may be *continuous* or *discrete*. In a continuous choice-set, the alternatives are defined by some constraints and cannot be enumerated. A discrete choice-set contains a finite number of alternatives that can be explicitly listed. Two concepts of a discrete choice-set may be further considered: the *universal* choice-set and the *reduced* choice-set. The universal choice-set contains all the potential alternatives in the context of the application. The reduced choice-set is the subset of the universal choice-set considered by a particular individual. In our model, the reduced choice-set of a driver agent is composed of a certain number[3] of available routes that connect the origin with the destination of the driver agent. We refer to this reduced choice-set with $\mathcal{C} = \{\rho_1, \rho_2, \ldots, \rho_m\}$.

Attributes

The attributes identify the important characteristics of each potential alternative that the decision-maker is taking into account to make his/her decision. In our model, the alternatives are routes, which are potentially characterised by several attributes (travel time, travel speed, type of roads, fuel and tyre consumption, etc.). Furthermore, if some economic model is applied as a traffic control mechanism (as in section 3.3), a route ρ_i could be characterised also by an additional monetary cost factor, which models the fact that a driver agent must acquire a resource that is no more freely assigned (i.e., the reservations at intersections).

Decision rule

The decision rule describes the process used by the decision-maker to reach his/her choice. Two main theories on decision rules are considered here, namely the neoclassical economic theory and the multinomial logit model.

- Neoclassical economic theory. The neoclassical economic theory assumes that each decision-maker is able to compare two alternatives a and b in the choice-set $\mathcal{C}$ using a preference-indifference operator $\succeq$. If $a \succeq b$, the decision-maker either prefers a to b, or is indifferent. Since the choice-set $\mathcal{C}$ is finite and preference-indifference operator has the properties of *reflexivity*, *transitivity* and *comparability*, the existence of an alternative, a^*, which is preferred to all of them is guaranteed. Because of the three properties listed above, there exists a function

$$U : \mathcal{C} \to \mathbb{R} \tag{4.12}$$

  such that

[3]The number of available options is set as a simulation parameter, as explained in section 4.1.2

$$a \succeq b \Leftrightarrow U(a) \geq U(b) \quad \forall a, b \in \mathcal{C} \tag{4.13}$$

is guaranteed. Therefore, the alternative a^* that is preferred to all of the alternatives in $\mathcal{C}$ is

$$a^* = argmax_{a \in \mathcal{C}} U(a) \tag{4.14}$$

In summary, making a choice is equivalent to assigning a value, called *utility*, to each alternative and selecting the alternative a^* associated with the highest utility. The concept of utility associated with the alternatives plays an important role in the context of discrete choice models, although the neoclassical economic theory has the limitations that does not consider some level of uncertainty.

- Multinomial logit model. Multinomial logit model is a random utility model that assumes, as neoclassical economic theory, that the decision-maker has a perfect discrimination capability. However, the analyst of the system is supposed to have incomplete information and, therefore, uncertainty must be taken into account. Thus, making a choice is equivalent to assigning an utility value to each alternative and a probability that an alternative is chosen. Using the multinomial logit model, the probability that an individual chooses alternative a within the choice-set $\mathcal{C}$ is $P(a) = e^{U(a)} / \sum_{k \in \mathcal{C}} e^{U(k)}$.

Utility of alternatives

Both decision rules described above, the neoclassical economic theory and the multinomial logit model, are based on the concept of utility of the alternatives. In general the alternatives are characterised by multiple attributes, so that the utility of an alternative is the output of a *multi-attribute utility function*. Multi-attribute Utility Theory(MAUT) [55] studies the aggregation of different criteria into a single utility function.

Be $\mathcal{C} = \{a_1, a_2, \ldots, a_m\}$ the set of alternatives, $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ the set of attributes, and $\mathcal{W} = \{w_1, w_2, \ldots, w_n\}$ the set of the weights of each attribute. The weight w_i reflects the relative importance of attribute x_i and is assumed to be positive. The weights of the attributes are usually determined on subjective basis, representing the opinion of the single decision maker.

Keeney and Raiffa [55] demonstrated that, if the attributes are mutually utility independent, then the multi-attribute utility function U can be expressed as follows:

$$U = \sum_{i=1}^{n} w_i u_i + w \sum_{\substack{i=1 \\ j>i}}^{n} w_i w_j u_i u_j + w^2 \sum_{\substack{i=1 \\ j>i \\ k>j}}^{n} w_i w_j w_k u_i u_j u_k + \ldots + w^{n-1} w_1 w_2 \cdots w_n u_1 u_2 \cdots u_n \tag{4.15}$$

where $u_i = u_i(a_j) \in [0, 1]$ represents the utility value of alternative a_j with respect to the attribute x_i and w is a scaling constant that must satisfy the normalising constraint $1 + w = \prod_{i=1}^{n}(1 + w w_i)$.

If $\sum_{i=1}^{n} w_i = 1$, then $w = 0$ and the Keeney and Raiffa function collapses to the following linear form:

$$U = \sum_{i=1}^{n} w_i \cdot u_i \tag{4.16}$$

In all the experiments of this chapter, we rely on the neoclassical economic theory to model the decision making process of the driver agent, which always selects the route with the highest utility value.

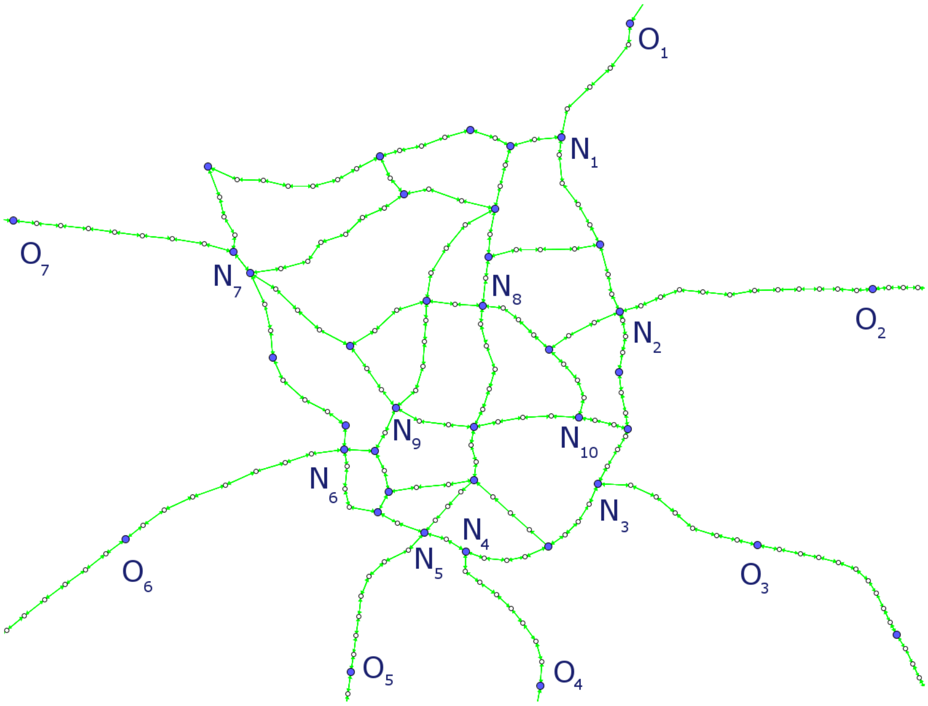


Figure 4.4: Network of intersections

4.3 AQT-inspired policies

In section 3.2, we analysed, in a scenario with a single intersection, the performance of different policies for the processing of the incoming reservation requests, inspired by the adversarial queueing theory (AQT). In this section, we recreate a scenario of an entire network of intersections (figure 4.4). We defined several locations that serve as origins and destinations for the traffic demand. The vehicles that commute from/to locations $\in \mathcal{O} = \{O_1, O_2, O_3, O_4, O_5, O_6, O_7\}$ form the traffic under evaluation. The vehicles that commute from/to locations $\in \mathcal{N} = \{N_1, N_2, N_3, N_4, N_5, N_6, N_7, N_8, N_9, N_{10}\}$ serve to add "noise" and to populate the network more realistically. We aimed at recreating a typical morning peak, with 2 different traffic demands, namely *low* and *high* (the total number of vehicles for the 2 traffic demands are summarised in table 4.1 and figure 4.5).

The metrics we used to evaluate the performance of the different policies for a

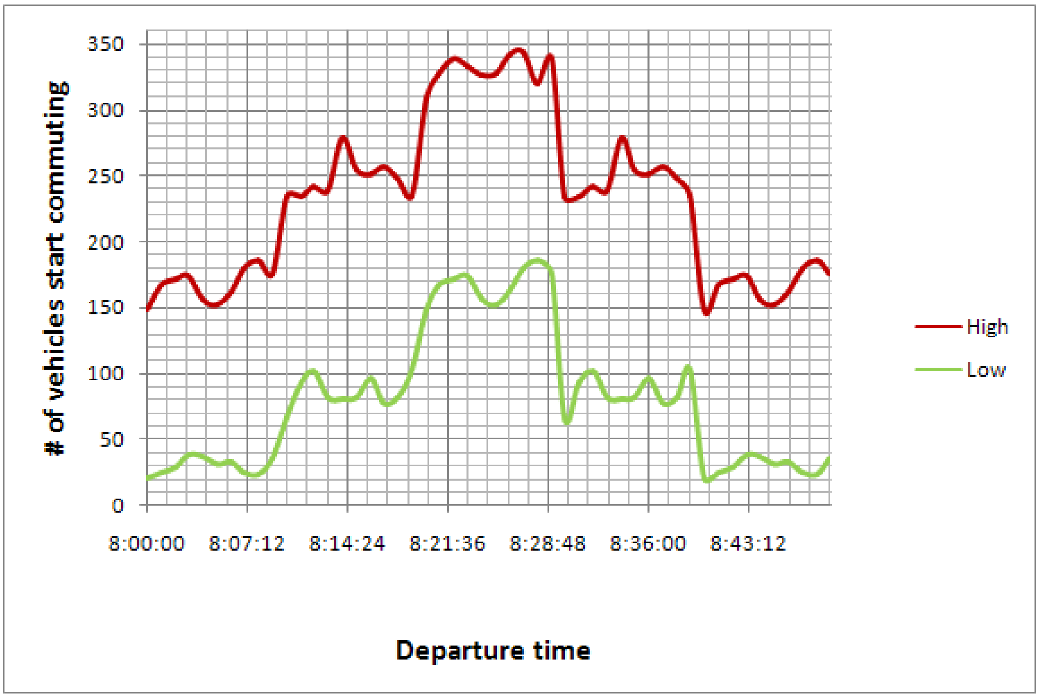


Figure 4.5: Low and high traffic demand

given O-D pair (o, d), with $o, d \in \mathcal{O}$, was the *average delay* (min.). Formally:

$$\frac{\sum_{i=1}^{N}(t_f^i - t_0^i) \;-\; \sum_{i=1}^{N}(\widehat{t_f^i} - \widehat{t_0^i})}{N}$$

where N is the number of vehicles, t_f^i and t_0^i are respectively the time when vehicle i arrives at its destination and when it leaves its origin in the simulation with the intersection regulated by a control mechanism, while $\widehat{t_f^i}$ and $\widehat{t_0^i}$ are respectively the time when vehicle i arrives at its destination and when it leaves its origin if we make the vehicles pass through the intersection unhindered.

To have a more global view of the performance of a single policy, we also calculated the average delay per covered km. Formally:

	Traffic density	
	low	*high*
# of vehicles	3986	11601

Table 4.1: Traffic demands for scenario 2

$$\frac{\sum_{(o,d)\in\mathcal{OD}} avgDelay(o,d)/||(o,d)||}{||\mathcal{OD}||}$$

where $\mathcal{OD}$ is the set of all the O-D pairs (o,d), $avgDelay(o,d)$ is the average delay of the given O-D pair, $||(o,d)||$ is the length of the route from o to d, and $||\mathcal{OD}||$ is the number of O-D pairs in the set $\mathcal{OD}$.

We are also interested in calculating the average delay per crossed intersection. Formally:

$$\frac{\sum_{(o,d)\in\mathcal{OD}} avgDelay(o,d)/intersections(o,d)}{||\mathcal{OD}||}$$

where $intersections(o,d)$ is the number of intersections of the route from o to d.

Experimental results

Low traffic demand. We first evaluated the reservation-based policies (first-come-first-served, FCFS, longest-in-system, LIS, shortest-in-system, SIS, farthest-to-go, FTG, nearest-to-source, NTS) and the intersection regulated by traffic lights (TL) under conditions of low traffic demand, with almost 4 thousands vehicles commuting through the road network. The average delays for each O-D pair are summarised in table 4.4 and in figure 4.6. For each O-D pair, we highlighted the lowest delay, to have a snapshot of which policy performs the best for a specific O-D pair.

At first glance, it seems confirmed that, as in the case of a single intersection, TL is the policy that introduces more delay for low traffic demand. This confirms the results of the single intersection scenario: the reservation-based intersection with

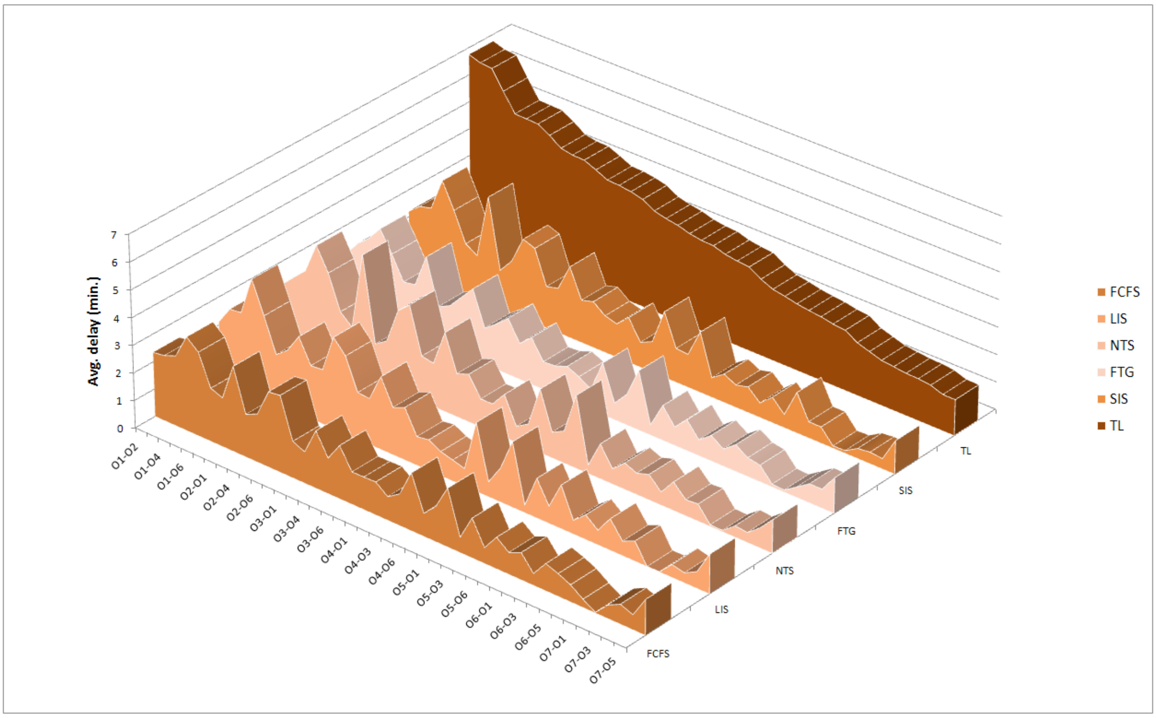


Figure 4.6: Average delay for low traffic demand

a good policy takes advantage of low traffic demands, drastically reducing the delay compared to traffic lights. With traffic lights it is possible for a vehicle to be stopped by a red light even if there are no vehicles on the link which has the green phase. Whereas, with a reservation-based intersection, few vehicles mean few reservations that are rejected, so that the transit speeds up.

To assess the overall performance of each policy, we rely on the average delay per covered km and the average delay per crossed intersection. Table 4.2 shows the average delay per covered km and the relative delay (where 100 is the best policy). When the traffic demand is low, the FTG policy performs the best, with 5.02 seconds of delay per covered km. Also FCFS performs quite well, with 5.27 seconds of delay per covered km, while LIS, SIS and NTS perform similarly and slightly worse than FTG. It is evident that with 12.54 seconds of delay per covered km, the intersection regulated by traffic lights performs the worst with low traffic demand.

Regarding the average delay per crossed intersection (table 4.3), the results are

Table 4.2: Average delay per covered km (low traffic demand)

	Average delay (sec/km)	**stdev.**	**Relative delay**
TL	12.54	4.21	249.75
FCFS	5.27	2.70	104.96
LIS	5.54	3.37	110.29
SIS	5.44	2.91	108.23
FTG	5.02	2.62	100.00
NTS	5.51	3.42	109.78

Table 4.3: Average delay per crossed intersection (low traffic demand)

	Average delay (sec/intersection)	**stdev.**	**Relative delay**
TL	41.26	10.05	254.92
FCFS	17.00	7.23	105.05
LIS	17.67	9.84	109.18
SIS	17.37	7.66	107.29
FTG	16.19	7.17	100.00
NTS	17.47	9.01	107.94

again similar. The FTG policy is the policy that causes least delay, 16.19 seconds per intersection, less than half that of TL. With traffic lights, a vehicle is delayed by 41.26 seconds when it passes through each intersection that makes up its route, with respect to the $16 - 17$ seconds of a reservation-based intersection.

Table 4.4: Average delay with low traffic demand (min.)

		Destination						
Origin		O_1	O_2	O_3	O_4	O_5	O_6	O_7
O_1	TL	-	1.47	3.11	3.97	6.56	3.31	1.64
	FCFS	-	0.71	0.84	**1.30**	**2.58**	1.15	0.61
	LIS	-	0.36	**0.55**	1.74	2.80	**0.64**	0.72
	SIS	-	0.42	1.25	1.57	2.68	1.13	**0.59**
	FTG	-	0.69	0.61	1.33	2.87	0.75	0.61
	NTS	-	**0.36**	0.94	1.58	3.00	0.87	0.62
O_2	TL	1.78	-	2.32	3.54	5.19	6.75	3.61
	FCFS	0.82	-	0.79	1.19	1.79	2.35	1.18
	LIS	**0.52**	-	**0.49**	1.12	2.16	2.10	1.05
	SIS	0.60	-	0.53	1.24	**1.61**	2.21	1.36
	FTG	0.68	-	0.58	1.05	1.87	**1.95**	**1.02**
	NTS	0.58	-	0.78	**1.03**	1.64	2.16	1.03
O_3	TL	2.61	1.53	-	1.31	3.72	4.64	6.60
	FCFS	0.56	0.02	-	0.54	1.95	1.57	**2.40**
	LIS	**0.28**	**0.01**	-	0.61	1.92	1.96	2.75
	SIS	0.72	0.10	-	**0.37**	**1.69**	1.43	2.57
	FTG	0.38	0.08	-	0.69	1.73	1.41	2.50
	NTS	0.31	0.05	-	0.73	1.75	**1.38**	2.59

	TL	4.54	3.18	1.31	-	2.37	4.15	5.21
	FCFS	2.75	2.10	1.28	-	1.14	**1.34**	1.96
	LIS	3.36	2.51	1.42	-	1.58	1.72	1.81
O_4								
	SIS	2.93	2.41	1.22	-	**0.93**	1.54	1.88
	FTG	**2.11**	**1.89**	1.29	-	1.12	1.35	**1.78**
	NTS	2.97	2.32	**1.21**	-	0.96	1.68	2.03
	TL	5.90	4.07	2.80	1.52	-	2.43	4.13
	FCFS	3.46	2.10	**2.12**	0.48	-	0.57	**1.14**
	LIS	4.21	2.66	2.37	0.47	-	0.63	1.69
O_5								
	SIS	3.87	2.62	2.21	**0.36**	-	0.82	1.65
	FTG	**3.39**	**2.07**	2.17	0.39	-	0.68	1.68
	NTS	4.10	2.90	2.65	0.41	-	**0.55**	1.60
	TL	2.53	4.94	4.36	3.07	2.19	-	2.24
	FCFS	1.44	3.12	2.82	1.28	**0.42**	-	0.98
	LIS	1.44	**2.83**	2.87	1.24	0.53	-	**0.89**
O_6								
	SIS	**1.04**	3.96	2.77	1.21	0.49	-	0.93
	FTG	1.27	3.07	**2.60**	**1.11**	0.62	-	0.93
	NTS	1.27	4.46	3.36	1.17	0.48	-	0.92
	TL	1.92	3.41	5.23	4.56	3.34	1.59	-
	FCFS	0.98	1.20	3.09	1.77	0.86	0.30	-
	LIS	1.14	**0.88**	3.05	2.00	0.77	**0**	-
O_7								
	SIS	1.48	1.65	2.95	1.94	0.92	0.08	-
	FTG	**0.96**	1.22	**2.63**	**1.63**	1.02	0.10	-
	NTS	1.03	1.57	3.27	1.69	**0.59**	0.06	-

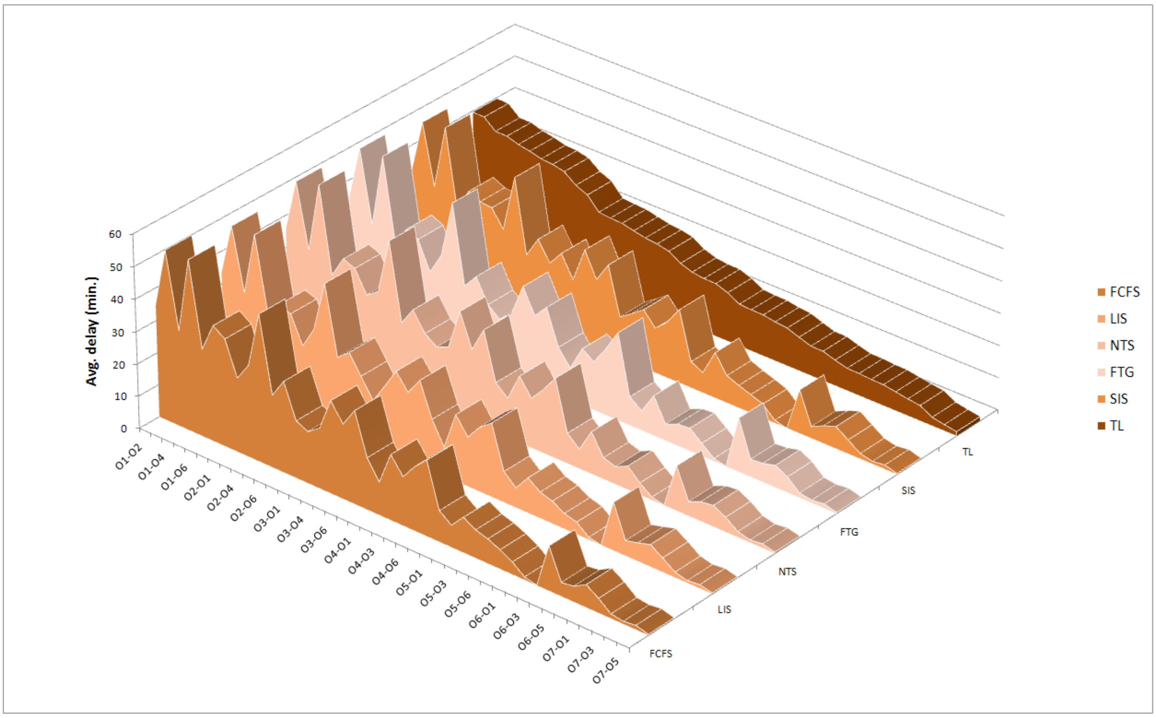


Figure 4.7: Average delay for high traffic demand

High traffic demand. Finally, we evaluated the reservation-based policies in a situation of high traffic demand, with more than 11 thousands vehicles travelling through the road network. The average delays for each O-D pair are summarised in table 4.7 and in figure 4.7. With high traffic demand, it is evident how the performance of the two types of intersection becomes similar.

To assess which policy is the best at reducing delays, we rely again on the average delay per covered km (table 4.5) and average delay per crossed intersection (table 4.6). When the traffic demand is high, the performance of the reservation-based intersection converges to that of an intersection controlled by traffic lights. LIS is the best policy, with 62.15 seconds of delay per covered km. The other reservation-based policies (FTG, SIS, NTS, FCFS) perform slightly better than traffic lights, with about $63-66$ seconds of delay per covered km. It is interesting to notice how TL has the lowest standard deviation. This is a hint that as demand increases, the performance of a reservation-based intersection becomes more volatile: in some parts of the network

Table 4.5: Average delay per covered km (high traffic demand)

	Average delay (sec/km)	**stdev.**	**Relative delay**
TL	66.39	35.84	106.83
FCFS	64.88	46.33	104.39
LIS	62.15	43.33	100.00
SIS	63.34	43.25	101.92
FTG	63.86	45.40	102.75
NTS	66.17	46.34	106.46

the transit can be sped up, while in other parts it may slow down the transit even more than traffic lights, whose behaviour is more predictable and stable. If we look at the average delay per crossed intersection, the results are similar. All the reservation-based policies perform similarly and slightly better than traffic lights. LIS is again the best one, with 195.28 seconds of delay per crossed intersection, 17.87 seconds less than traffic lights. All the other reservation-based intersections are 4% to 5% worse than LIS.

Similarly to the experimental results described in section 3.2, the reservation-based intersection considerably speeds-up the traffic flow if the arrival rate of vehicles does not exceed its management capability. Thus, keeping the arrival rate at the intersection low is crucial for the performance of a reservation-based policy. In fact, in these cases it is on average 2 times more efficient than an intersection controlled by traffic lights. Furthermore, if we look at the different AQT-based policies that have been tested, we notice that they show an improvement on the FCFS policy, even if this improvement is not so significant from a practical point of view. For example, for high traffic demand, the FCFS average delay per intersection is 8 seconds greater than the LIS one, which is hardly a noticeable improvement from the perspective of the driver. If improving the throughput of a reservation-based intersection with more sophisticated policies is hard, we must explore other ways to make the reservation-based intersections more efficient. For this reason, we claim that the distribution of vehicles in the network is the crucial problem that must be tackled.

Table 4.6: Average delay per crossed intersection (high traffic demand)

	Average delay (sec/intersection)	**stdev.**	**Relative delay**
TL	213.15	101.29	109.15
FCFS	203.65	132.87	104.29
LIS	195.28	124.15	100.00
SIS	198.51	122.84	101.65
FTG	200.75	130.06	102.80
NTS	206.85	131.82	105.92

Table 4.7: Average delay with high traffic demand (min.)

		Destination						
Origin		O_1	O_2	O_3	O_4	O_5	O_6	O_7
O_1	TL	-	6.62	22.69	35.20	39.66	12.88	1.36
	FCFS	-	4.30	18.89	27.39	34.69	6.84	0.32
	LIS	-	4.69	19.92	27.32	33.42	6.13	0.32
	SIS	-	**3.66**	**18.54**	**25.22**	**31.99**	6.33	**0.29**
	FTG	-	5.47	19.26	26.50	34.67	6.11	0.33
	NTS	-	4.42	20.46	27.45	35.76	**6.06**	0.43
O_2	TL	2.67	-	16.62	28.07	32.47	36.44	4.78
	FCFS	0.89	-	15.16	22.62	28.68	30.01	**1.43**
	LIS	**0.76**	-	15.38	22.52	28.23	30.34	1.67
	SIS	0.86	-	**14.00**	**20.32**	**25.35**	29.92	1.70
	FTG	0.92	-	15.14	21.86	28.41	**29.13**	1.47
	NTS	0.86	-	15.50	21.87	28.56	32.33	1.80
O_3	TL	8.02	6.98	-	12.96	18.45	23.71	33.29
	FCFS	0.96	0.06	-	9.70	16.16	**17.89**	23.13
	LIS	**0.76**	0.02	-	9.59	16.05	19.92	**21.46**
	SIS	0.89	**0.0**	-	**8.22**	**12.91**	20.70	30.64
	FTG	0.85	0.06	-	8.93	14.95	19.04	21.68
	NTS	0.94	0.07	-	9.46	15.62	21.51	25.78

	TL	**16.68**	**15.27**	**6.91**	-	**5.89**	12.25	23.56
	FCFS	20.38	19.93	13.81	-	7.03	10.93	16.16
	LIS	20.71	19.96	14.79	-	6.80	**10.88**	**13.90**
O_4								
	SIS	18.31	17.52	13.56	-	6.90	14.13	25.28
	FTG	17.59	20.76	16.64	-	6.81	12.42	15.97
	NTS	20.78	19.76	13.86	-	6.54	13.08	18.84
	TL	**20.75**	**21.92**	**13.21**	5.26	-	6.53	17.18
	FCFS	28.99	28.89	23.49	4.50	-	**4.81**	10.32
	LIS	26.48	26.11	22.89	4.39	-	5.66	**9.02**
O_5								
	SIS	27.71	29.39	22.94	**4.30**	-	5.51	15.56
	FTG	25.86	29.27	27.28	4.87	-	5.82	10.03
	NTS	30.35	33.00	24.85	4.57	-	6.06	12.33
	TL	11.85	**36.39**	**29.66**	**24.08**	21.54	-	10.51
	FCFS	8.75	53.34	45.93	28.43	23.21	-	7.62
	LIS	8.11	49.71	43.82	25.66	**21.15**	-	6.97
O_6								
	SIS	8.56	49.82	43.10	26.68	21.64	-	7.04
	FTG	**5.12**	51.80	46.19	26.26	21.77	-	6.77
	NTS	6.69	53.96	45.61	27.36	22.84	-	**5.48**
	TL	1.97	9.26	**39.55**	34.55	33.57	9.20	-
	FCFS	1.35	5.96	53.24	36.18	33.62	3.88	-
	LIS	**1.20**	**4.81**	49.37	**31.08**	**29.87**	3.77	-
O_7								
	SIS	1.36	5.74	48.68	32.82	31.04	4.37	-
	FTG	1.36	7.07	51.34	32.01	30.35	4.02	-
	NTS	1.30	7.31	52.05	33.97	32.58	**3.70**	-

4.4 Computational economies for traffic assignment

There is great interest in developing models to efficiently allocate space in a urban road network and thus relieve congestion. Most of the work in the literature uses static transportation models for analysis, which significantly underestimate network congestion levels in traffic networks. Dynamic traffic assignment models have attracted recent attention, due to their ability to account for time-varying properties of traffic flow [53], although these formulations generally lead to extremely complicated solution procedures. Nevertheless, progress has been made using techniques such as simulation for solving large networks [66], with the great advantage that agent-based, adaptive, pricing models can be applied.

In the following sections, we will present two different computational economies that can be applied to solve the traffic assignment problem. The first one, $\mathcal{ECO}^+$, is a cooperative economy that tackles the problem from the *equilibrium* perspective. In fact, we applied multiagent reinforcement learning techniques to dynamically coordinate the intersection managers' pricing policies with the aim of converging to the optimal joint policy.

The second one, $\mathcal{ECO}^-$, is a competitive economy that tackles the problem from the *adaptation* perspective. In this economy, the intersection managers act as competitors that strive to sell the resources they supply (i.e., the reservations), thus dynamically adapting their prices in response to the current demand.

4.4.1 $\mathcal{ECO}^+$: a cooperative economy for for traffic assignment

In this section, we present $\mathcal{ECO}^+$, a cooperative economy for networks of reservation-based intersections. We started from Dresner and Stone's work [35] and assume the existence of an advanced traffic management infrastructure that allows for reservation-based intersection control. In $\mathcal{ECO}^+$, driver agents trade with the intersection managers in a virtual marketplace, purchasing reservations to cross intersections when travelling through the urban road network. We apply multiagent reinforcement learning techniques to dynamically coordinate the intersection managers' pricing policies with the aim of optimising a common objective.

This section is organised as follows. We first define the infrastructure model that we take for granted, we then introduce the economic model of the market that is established between driver agents and intersection managers as well as the behavioural model of the intersection manager. Finally we perform an experimental evaluation of the model.

Infrastructure model

We assume that the agents in the future urban road management scenario that we envision will have the following capabilities:

- **Infrastructure-to-infrastructure communication.** Intersection managers are able to communicate with each other. This assumption is reasonable, because for example already existing fibre-optic connections along certain main urban roads could be used.

- **Vehicle-to-infrastructure communication.** Driver agents can communicate with the intersection managers in their proximity. Such proximity-based communication is already in use in different elements of today's traffic infrastructures. We assume that in general a driver agent is able to communicate with the forthcoming intersection on its route, and eventually also with the neighbours of such intersection.

- **Payment system.** A trusted payment system is available, allowing driver agents to securely transfer money to intersection managers when required. Such mechanisms are already in use in today's toll roads.

- **Price index board.** We assume that driver agents can be provided with the current prices of the intersections of the urban road network. For instance, each intersection manager can publish the price of the reservations it sells on a price index board, accessible to the driver agents. Also this assumption is reasonable and technically feasible. For instance, the New York Stock Exchange[4] indexes approximately 8500 stocks, whose price variations are spread worldwide in terms of seconds to guarantee the same information to all the operators. The city of Madrid has 1314 intersections regulated by traffic lights. We claim that setting up such price index board is already feasible with the current technologies.

Market model

In such a setting, in order to design the rules of the marketplace, it is essential to specify the regulations that govern the interactions between a driver agent and a single intersection manager. Such regulations need to specify how successful deals are made and what happens if something goes wrong (e.g., when a reservations needs to be withdrawn, or when a driver agent arrives at an intersection without a valid reservation).

- **Purchasing a reservation when approaching the intersection.**

  A driver agent is able to purchase reservations from the intersection managers in its proximity, while intersection managers apply the simple FCFS policy described in the previous chapter to honour reservation requests. To purchase a reservation when approaching the intersection, a driver agent "calls-ahead" the intersection manager and provides the necessary data to simulate its transit (see section 2.5 for more details about how the reservation-based system works). A driver agent is uniquely identified by a vehicle ID, so that at each intersection a driver agent may hold only one reservation. If the request cannot be satisfied, due to conflicts with the already confirmed reservations, the intersection manager refuses the reservation request. Otherwise, it sends a confirmation message to the driver agent, which includes the price of the reservation that will be charged when it makes use of the reservation, i.e., during the crossing. We assume that when a driver agent requests a reservation, it is aware of its price, consulting the price index board, still a driver agent is free to withdraw

[4] http://www.nyse.com

```
(request reservation
            :sender  D-3548
          :receiver  IM-05629
          :content(
                     :arrival_time   08:03:15
                     :arrival_speed  23km/h
                     :lane           2
                     :type_of_turn   LEFT
                     :queued         true
                  )
```

Figure 4.8: Augmented REQUEST message

a purchased reservation (see below). When the driver actually arrives at the intersection, the driver agent is charged with the contracted price and it safely crosses the intersection.

- **Receiving a reservation when queued at the intersection.**

  If a driver agent does not hold a valid reservation, when it reaches the edge of the intersection it must stop. In this case, it is entitled to purchase a reservation specifying that it is queued at the intersection. We augmented the REQUEST message adding a new boolean field, `:queued` (see figure 4.8). If true, the requester specifies that it is queued at the intersection, and in this case it is entitled to receive a reservation for free. We assume that the road infrastructure provides intersection managers with a way to actually confirm that a vehicle is stopped at the intersection, for example using cameras with plate recognition. Such systems are used nowadays to control the access by non-authorised users to restricted areas, such as historical city centres or bus/taxi lanes.

- **Withdrawing a reservation.**

  When a driver agent purchases a reservation, it tries to meet the reservation constraints, especially the arrival time. If it realises that these cannot be met, due for example to changing traffic conditions, it can withdraw the reservation. Giving the driver agents the possibility of freely withdrawing a previously acquired reservation is crucial for the efficiency of the reservation-based intersection control system. Otherwise, the driver agents would have incentives to purchase a reservation only when they are extremely sure of the arrival time, i.e., when it is close to the intersection, thus negatively affecting the intersection throughput.

Intersection manager model

As said in section 2.4, in a multiagent system the agents can be *cooperative* or *competitive*. In cooperative systems, the agents pursue a common goal. Such systems are characterised by the fact that the designers of the multiagent system are free to design the agents at will. The agents can be built with extensive knowledge of the system and they can expect benevolent intentions from other agents. Still, designing a cooperative multiagent system to have good emergent behaviour is not an easy task, due to issues to cope with, such as the credit assignment and the impact of co-adaptation. In contrast to cooperative multiagent systems, agents in a competitive setting have non-aligned goals, and individual agents seek only to maximise their own gains.

$\mathcal{ECO}^+$ is a *cooperative* economy, because the intersection managers, being part of the infrastructure, can be programmed to work as a team in a cooperative economy. Strictly speaking, a cooperative economy would be a cartel, where the intersection managers agree to set the highest possible price. Still, as market designer we take advantage of the market mechanisms as a *tool* to produce a good and fair traffic system (i.e., a system with less congestion, lower travel times, etc.). In this kind of economy, we model the intersection managers from the point of view of effective teamwork, aiming at i) discovering the effect of a specific price vector and ii) coordinating their prices in order to maximise the global profit.

In $\mathcal{ECO}^+$, the intersection managers trade with the driver agents the reservations of the intersections they manage. Thus, the action space of an intersection manager is formed by the prices that a reservation may be valued. Formally, for the generic intersection manager i, the actions space $\mathcal{A}_i$ is:

$$\mathcal{A}_i = \{p_i^0, p_i^1, \ldots, p_i^m\} \tag{4.17}$$

where $p_i^0 < p_i^1 < \cdots < p_i^m$, and p_i^0 is the minimum price and p_i^m is the maximum price of a reservation.

The central design issue in $\mathcal{ECO}^+$ is the profit function of the intersection managers, whose maximisation represents their goal. Such profit function is modelled with the aim of penalising *congested* as well as *unused* intersections. The profit function of intersection manager i is defined as the difference between the revenues, R_i, and the costs, C_i.

The revenues are:

$$R_i = p_i \cdot N \tag{4.18}$$

where p_i is the price of a reservation and N is the total number of sold reservations.

Regarding the costs, an intersection manager does not face real "production" costs, so that the cost factor should be 0. In this setting, the winning strategy for every intersection manager is to sell the reservations at the maximum price in order to maximise the global profit. Nevertheless, we are market designers that aim to model the market rules to enforce some desired properties of the underlying traffic system, such as less congestion and lower travel times.

For this reason, we model a cost term in the following way. When an intersection manager sells a reservation to a driver agent that is queued at the intersection, we enforce that the price charged to the driver agent is 0. This is equivalent to penalising the intersection manager with a cost penalty c_i, equal to the entire price of the reservation, p_i, for each reservation sold to a driver agent that is queued at the intersection. The cost function is so defined as:

$$C_i = c_i \cdot N_q = p_i \cdot N_q \tag{4.19}$$

where N_q is the number of reservations sold to driver agents that are queued at the intersection.

Given the revenues of equation 4.18 and the costs of equation 4.19, the profit is defined as:

$$P_i = R_i - C_i = p_i \cdot N - p_i \cdot N_q = p_i \cdot N_a \tag{4.20}$$

given that $N = N_a + N_q$, where N_a is the number of reservations sold to driver agents that are approaching the intersection.

Evaluating $\mathcal{ECO}^+$

The following section describes the multiagent reinforcement learning problem that we aim at solving, detailing the environment model, the reward functions and the learning method.

Environment. In a reinforcement learning problem, the learner interacts with the environment, which provides the learner with feedback information (reward and state transition), in response to the learner actions. In a multiagent reinforcement learning problem, the environment responds to the *joint action* of the learning agents. The simplest environment that we can model is a *stateless environment*, i.e., an environment that can be in one single state. In this setting, the environment is like a repeated single-stage game [37]. In these games, the learning agents are rewarded on the basis of their joint actions. In each round of the game, every agent chooses an action. These actions are executed simultaneously and the a global reward signal that corresponds to the joint action is broadcast to all agents.

In the following experiments, we model the environment as a repeated single-stage game. The intersection managers jointly select an action among those available. Such joint action produces a reward signal that is received by all the intersection managers, which use this information to update their policies. The goal is to enable the intersection managers to learn the optimal joint action, according to their goals.

Reward functions. As said in section 2.4.3, in multiagent cooperative reinforcement learning we can distinguish two levels of reward functions: the *global reward* and the *agent reward*. The global reward is a signal that rates the usefulness of a joint action with respect to the global goal that the collective of learning agents pursues. The agent reward is the signal that aims at rating the individual agent action, i.e., the contribution of the agent to the global reward. In $\mathcal{ECO}^+$ we use the following global reward:

$$G(\mathbf{p}) = \sum_i P_i(\mathbf{p}) \tag{4.21}$$

where $\mathbf{p}$ is the joint action (i.e., the price vector) and $P_i(\mathbf{p})$ is the profit of the intersection manager i (see equation 4.20).

We evaluate two different agent reward functions, namely the *Local Reward* (LR) and the *Expected Difference Reward* ($ExpDR$). The LR rewards the learning agent with its local term of the summatory of equation 4.21. Formally:

$$LR_i(\mathbf{p}) = P_i(\mathbf{p}) \tag{4.22}$$

The $ExpDR$ rewards the learning agent with the difference between the expected global reward and the expected global reward when the agent i's action, p_i, is set to a specific value $p_i^j \in \mathcal{A}_i$. Formally:

$$ExpDR_i(\mathbf{p}) = \mathbb{E}\left[G(\mathbf{p} \mid p_i = p_i^j \in \mathcal{A}_i)\right] - \mathbb{E}\left[G(\mathbf{p})\right] \tag{4.23}$$

where $\mathbb{E}\left[G(\mathbf{p})\right]$ is the expected global reward and $\mathbb{E}\left[G(\mathbf{p} \mid p_i = p_i^j \in \mathcal{A}_i)\right]$ is the expected global reward when the intersection manager i applies price p_i^j. Such expectation values can be calculated by averaging the global rewards that an agent observes over the learning episodes, so that they become more and more precise with time.

Learning method. To learn in a distributed and coordinated fashion which price vector leads to the best system performance we use independent Q-learning [24] with immediate rewards and ϵ-greedy action selection [108]. Each agent maintains a Q-value for each of its actions. The Q-value provides an estimate of the usefulness of performing this action in the next iteration of the game and these values are updated after each iteration of the game according to the reward received for the action. After having taken action p_i^j, the intersection manager agent updates its action-value function estimation as follows:

$$Q_i^{t+1}(p_i^j) = Q_i^t(p_i^j) + \alpha \cdot [r_t(\mathbf{p}) - Q_i^t(p_i^j)] \tag{4.24}$$

where $Q_i^t(p_i^j)$ is the estimation (at time t) of the usefulness of setting price p_i^j to the reservations sold by intersection manager i, $\alpha \in (0,1]$ is the learning rate, $\mathbf{p}$ is the joint action, i.e., the price vector $[p_1\ p_2\ \dots\ p_n]$, and $r_t(\mathbf{p})$ is the reward received by intersection manager i, which in general depends on the full joint action $\mathbf{p}$.

Each intersection manager selects a random action with probability $\epsilon \in (0,1)$, and the *greedy* action (i.e., the action with highest Q-value) with probability $1-\epsilon$.

Driver agent model. As said in section 4.2, the driver agent model must define a multi-attribute utility function that rates the available routes in the choice set $\mathcal{C}$ as well as a decision rule. Given that the traffic system is regulated by a market mechanism, the driver agent must take into consideration different aspects of a route to determine its utility value. In these experiments, we model the driver multi-attribute utility function as a 2-attributes function:

$$U(\rho_i) = w_{TT} \cdot u_{TT}(\rho_i) + w_K \cdot u_K(\rho_i) \tag{4.25}$$

where $U(\rho_i)$ is the utility of route ρ_i, $u_{TT}(\rho_i)$ is the normalised utility of route ρ_i against the estimated travel time attribute, $u_K(\rho_i)$ is the normalised utility of route ρ_i against the reservations cost attribute, w_{TT} is the weight of the estimated travel time attribute and w_K is the weight of the reservations cost attribute.

Basically, if $w_{TT} = 1$, the driver agent utility only considers the estimated travel time attribute (i.e., it prefers the *shortest route*, no matter the price of the reservations), if $w_K = 1$, the driver agent utility only considers the reservations cost attribute (i.e., it prefers the *cheapest route*, no matter the travel time), while for every other combination of the weights w_{TT} and w_k the driver agent considers the trade-off between estimated travel time and reservations cost.

Since the higher the utility values u_{TT} and u_K, the better the alternative ρ_i, these functions are defined as:

$$u_{TT}(\rho_i) = \frac{M_{TT} - TT(\rho_i)}{M_{TT} - m_{TT}} \tag{4.26}$$

$$u_K(\rho_i) = \frac{M_K - K(\rho_i)}{M_K - m_K} \tag{4.27}$$

where $TT(\rho_i)$ is the estimated travel time of route ρ_i, $K(\rho_i)$ is the reservations cost of route r_i, $M_{TT} = max_{\rho_i \in \mathcal{C}} TT(\rho_i)$, $m_{TT} = min_{\rho_i \in \mathcal{C}} TT(\rho_i)$, $M_K = max_{\rho_i \in \mathcal{C}} K(\rho_i)$ and $m_K = min_{\rho_i \in \mathcal{C}} K(\rho_i)$

Once the utility of the alternative routes $\rho_1, \ldots, \rho_k$ has been computed, the driver agent must choose one of the alternatives. As said in section 4.2, such choice may be probabilistic, according to the multinomial logit function, or deterministic. In the following experiments, we model the driver agent as a deterministic utility maximiser, that always selects the route with the highest utility value.

Experimental results. The first experiment simulates a simple network of 22 intersections (figure 4.9). We simulate a main traffic flow along the North-South axis, and a secondary traffic flow along the West-East axis. The traffic flow along the North-South axis is approximately 8 times more intense than that along the West-East axis. Table 4.8 summarises the traffic demand, 1051 vehicles generated in an interval of 20 minutes. A generation of the traffic demand corresponds to a single-stage of the game, and we call it *learning episode*. At the end of the learning episode (i.e., when all the vehicles have reached their destination), each agent computes the agent reward, updates its Q-table and selects the next action (i.e., the price of the reservations it sells) according to the ϵ-greedy action selection policy. Then the same traffic demand is generated again and the process continues for 100 learning episodes.

Single Q-learning works with discrete action spaces, so for each intersection manager we set the following action space:

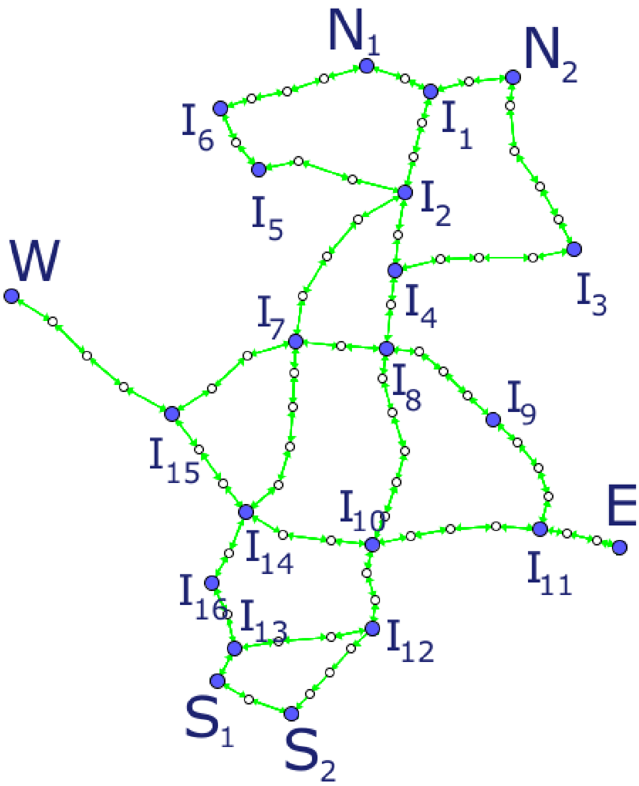


Figure 4.9: Network for the evaluation of $\mathcal{ECO}^+$

$$\mathcal{A}_i = \{p_i^0, p_i^0 + 10, p_i^0 + 20, \ldots, p_i^m - 10, p_i^m\} \tag{4.28}$$

The minimum price p_i^0 was set to 0, while the maximum price was set to 100. We assume that these quantities are cents of euros, so that the price range spans from 0 to 1 euro. Given that a single intersection manager has in total 11 actions, there are 21^{11} possible joint actions.

Regarding the update rule of the Q-values (equation 4.24), we set the learning rate α to 0.1, while the probability ϵ of the ϵ-greedy action selection policy is set to 0.5 and reduced geometrically every 10 learning episodes.

For the driver agent model, we set the parameter $w_{TT} = 1 - w_K$ and we sample w_{TT} using a normal distribution with mean 0.5 and variance 0.25. Since the prices of the reservations are changed at the end of the learning episode, i.e., when all the

	O-D pair					
	$N_1 - S_1$	$N_1 - S_2$	$N_2 - S_1$	$N_2 - S_2$	$E - W$	$W - E$
# of vehicles	213	211	207	227	96	97

Table 4.8: Traffic demands for the evaluation of $\mathcal{ECO}^+$

driver agents have reached their destination, a driver agent does not need to change "on-the-fly" the route it selected at the beginning of its trip.

The metric we used to evaluate the performance of the intersection managers in $\mathcal{ECO}^+$ was the *global reward* (equation 4.21) that they get during the learning. To assess the effect of this learning on the population of driver agents, we also measured the *average travel time* of the vehicles and how it varies during the learning episodes. We divided the driver agents into two groups: those travelling along the North-South axis (marked with N-S in the plots), and those travelling along the West-East axis (marked with W-E in the plots).

Figure 4.10 plots the global profit using the $ExpDR$ reward function. The intersection managers converge to a price vector that generates a global profit of about 3600 euros, which is approximately 21% of the theoretically highest global profit (16816 euros, obtained if all the 1051 driver agents passes through the 16 intersections that are selling reservations, and paying the highest possible price, i.e., 1 euro).

Nevertheless it is interesting to see the effect of this profit maximisation on the average travel time of the driver agents. If the intersection managers try to maximise the profit, they indirectly influence the driver agent decision making, thus affecting the selected route and consequently the travel time. For the vehicles that travel along the West-East axis, the travel time is basically the same in all the learning episodes, thus being insensitive to the learning activity of the intersection managers. Still, the average travel time of the vehicles travelling along the North-South axis is highly affected by the cooperative learning of the intersection managers, and falls from approximately 34 minutes to 23 minutes at the end of the learning. The cooperative economy provides the driver agents with incentives to explore alternatives to the shortest path, according to their utility function. This alleviates the congestion that

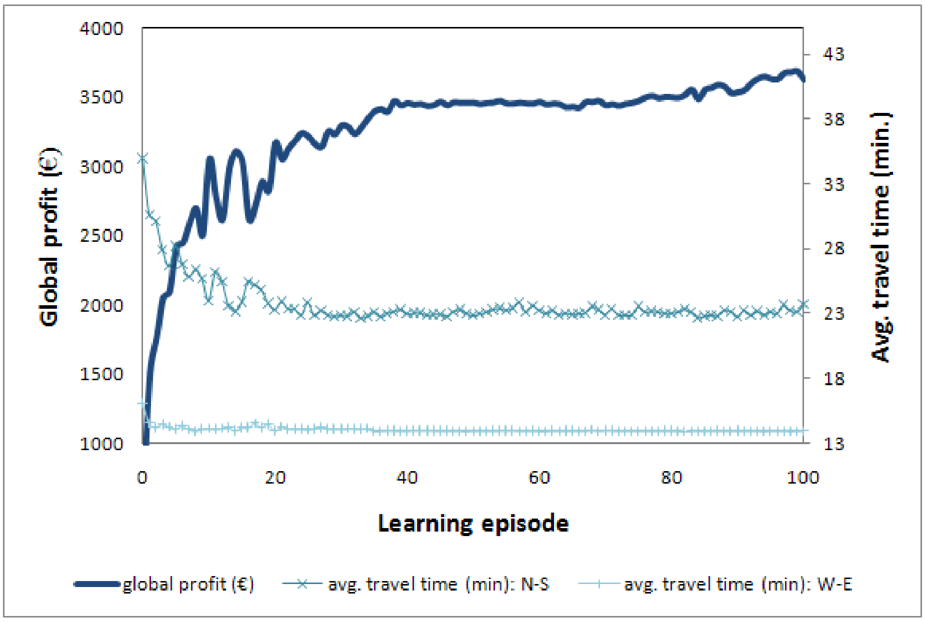


Figure 4.10: Global profit and average travel time using $ExpDR$

may arise and, consequently, decreases the travel times, due to a better allocation of the road network resource.

Figure 4.11 plots the global profit generated by the intersection managers using the LR reward function. The dynamic is quite different from that in figure 4.10. The collective of intersection managers converges to a price vector that generates a global profit of about 2800 euros, 22% less than when they use the $ExpDR$. Also the average travel time of the vehicles travelling along the North-South axis is quite a lot higher at the end of the learning, 30 minutes versus 23 minutes, while the average travel time of the vehicles travelling along the West-East axis is again insensitive to the learning performed by the intersection managers. Figure 4.12 compares the $ExpDR$ and the LR. The LR converges more quickly but settles around lower values of the global profit compared to the $ExpDR$. Furthermore, the average travel time at the end of the learning is 25% higher compared to the average travel time that is obtained when the intersection managers use $ExpDR$, 25 minutes versus 20 minutes. Nevertheless, the LR has the advantage that it needs less information to be computed, since each intersection manager is able to compute it locally, without communication with the

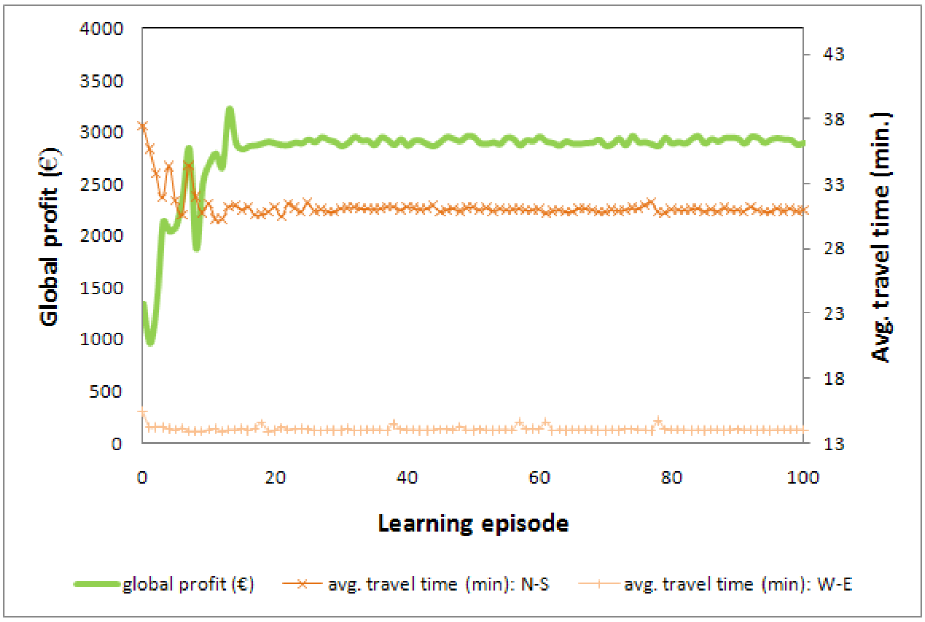


Figure 4.11: Global profit and average travel time using LR

other intersection managers.

A different profit function. In the experiments described above, the goal pursued by the team of intersection managers was the maximisation of the *global profit*, expressed as the sum of the local profits gained by each intersection manager. This sum is unbounded, because it depends on the number of vehicles that are travelling through the urban road network. Thus, this reward function does not fulfil the necessary condition that assures the convergence of the action-value function Q to the optimal one Q^* [108].

A different (and hopefully better) reward function is the *marginal profit*. The marginal profit, MP_i, is calculated as the difference between the *marginal revenue*, MR_i, and the *marginal cost*, MC_i.

The marginal revenue is defined as:

$$MR_i = \frac{dTR}{dN} \tag{4.29}$$

where TR is the total revenue and N is the total number of sold reservations.

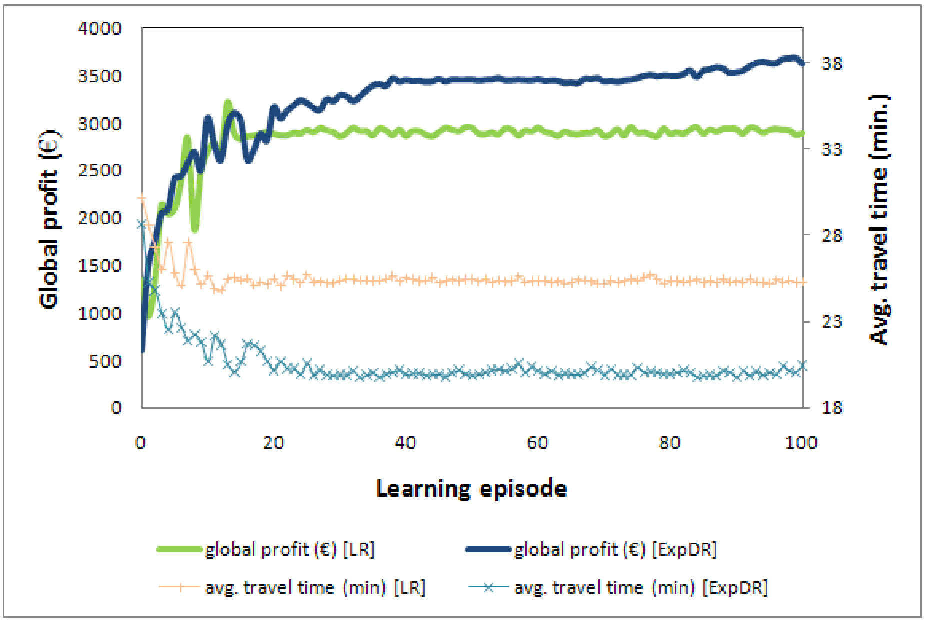


Figure 4.12: Comparison between *ExpDR* and *LR*

Basically the marginal revenue is the revenue increase that an intersection manager has for every reservation that it sells. Given that the total revenue is the product between the price of a reservation and the number of sold reservations, equation 4.29 becomes:

$$MR_i = \frac{d(p_i \cdot N)}{dN} = N \cdot \frac{dp_i}{dN} + p_i \cdot \frac{dN}{dN} \tag{4.30}$$

where p_i is the price of a reservation. Since in our modelling the price p_i does not change with the number of sold reservations N, $\frac{dp_i}{dN} = 0$, and equation 4.30 becomes:

$$MR_i = p_i \cdot \frac{dN}{dN} = p_i \tag{4.31}$$

As said before, we introduce a cost term to penalise congested intersections. This term, c_i, is applied to all the reservation sold to driver agents that are queued at the intersection. The marginal cost is thus defined as:

$$MC_i = \frac{dTC}{dN} = \frac{d(c_i \cdot N_q)}{dN} = c_i \cdot \frac{dN_q}{dN} + N_q \cdot \frac{dc_i}{dN} \tag{4.32}$$

where TC is the total cost, c_i is the cost factor, N_q is the number of reservations sold to driver agents that are queued at the intersection, and N is the total number of sold reservations.

Given that the cost factor c_i does not change with the number of sold reservations N, $\frac{dc_i}{dN} = 0$, and equation 4.32 becomes:

$$MC_i = c_i \cdot \frac{dN_q}{dN} = c_i \cdot \frac{N_q}{N} \tag{4.33}$$

Given the marginal revenue of equation 4.31 and the marginal cost of equation 4.33, the marginal profit is defined as:

$$MP_i = MR_i - MC_i = p_i - c_i \cdot \frac{N_q}{N} = \frac{N \cdot p_i - N_q \cdot p_i}{N} = p_i \cdot \frac{N_a}{(N_a + N_q)} = \tag{4.34}$$

if we set $c_i = p_i$ and given that $N = N_a + N_q$, where N_a is the number of reservations sold to driver agents that are approaching the intersection.

The evaluation is performed using the same setting described above, with exception of the global reward and the agent reward functions. Given the new (marginal) profit function, the global reward becomes:

$$G(\mathbf{p}) = \sum_i MP_i(\mathbf{p}) \tag{4.35}$$

where $\mathbf{p}$ is the joint action (i.e., the price vector) and $MP_i(\mathbf{p})$ is the marginal profit of the intersection manager i.

The two agent reward functions under evaluation, namely the *Local Reward* (LR) and the *Expected Difference Reward* ($ExpDR$), now becomes:

$$LR_i(\mathbf{p}) = MP_i(\mathbf{p}) \tag{4.36}$$

and

$$ExpDR_i(\mathbf{p}) = \mathbb{E}\left[G(\mathbf{p} \mid p_i = p_i^j \in \mathcal{A}_i)\right] - \mathbb{E}\left[G(\mathbf{p})\right] \tag{4.37}$$

Using the marginal profit to compute the reward function has the advantage that the reward will be bounded, so the necessary condition that assures the convergence of the action-value function Q to the optimal one Q^* is fulfilled. In fact, the marginal profit $MP_i(\mathbf{p})$ varies between 0 (when $N_a = 0$, that is, when all the driver agents purchase a reservation when they are queued at the intersection), and p_i (when $N_q = 0$, that is, when all the driver agents purchase a reservation when they are approaching the intersection).

Figure 4.13 plots the global marginal profit using the *ExpDR* reward function. The dynamic shows that the intersection managers converge to a price vector that generates a global marginal profit of about 8.6 euros, which is approximately the 53% of the theoretically highest global marginal profit (16 euros, i.e., 1 euro per intersection manager, excluding the intersections N_1, N_2, S_1, S_2, W, E).

It is again interesting to see the effect of this profit maximisation on the average travel time of the driver agents. If the intersection managers try to maximise the marginal profit, they indirectly influence the driver agent decision making, thus affecting the selected route and consequently the travel time. For the vehicles that travel along the West-East axis, the travel time is basically the same in all the learning episodes, thus being insensitive to the learning activity of the intersection managers. Still, the average travel time of the vehicles travelling along the North-South axis is highly affected by the cooperative learning of the intersection managers, and falls from approximately 30 minutes to 23 minutes at the end of the learning, which is a result somewhat similar to that of the previous experiments.

What is very different from the previous experiments is the performance of the local reward *LR*. Figure 4.14 plots the global marginal profit using the *LR* reward function. The dynamic is very similar to that in figure 4.13. The collective of intersection managers converges to a price vector that generates a global marginal profit of about 8.8 euros. This implies that the average travel time of the vehicles travelling along the North-South axis decreases to approximately 23 minutes at the end of the learning, while the average travel time of the vehicles travelling along the West-East axis is basically the same in all the learning episodes.

Figure 4.15 provides a comparison between the *ExpDR* and the *LR*. The *LR* converges more quickly and settles at the same global marginal profit with respect to the *ExpDR*, and both have the effect of reducing the average travel time. Since *LR*

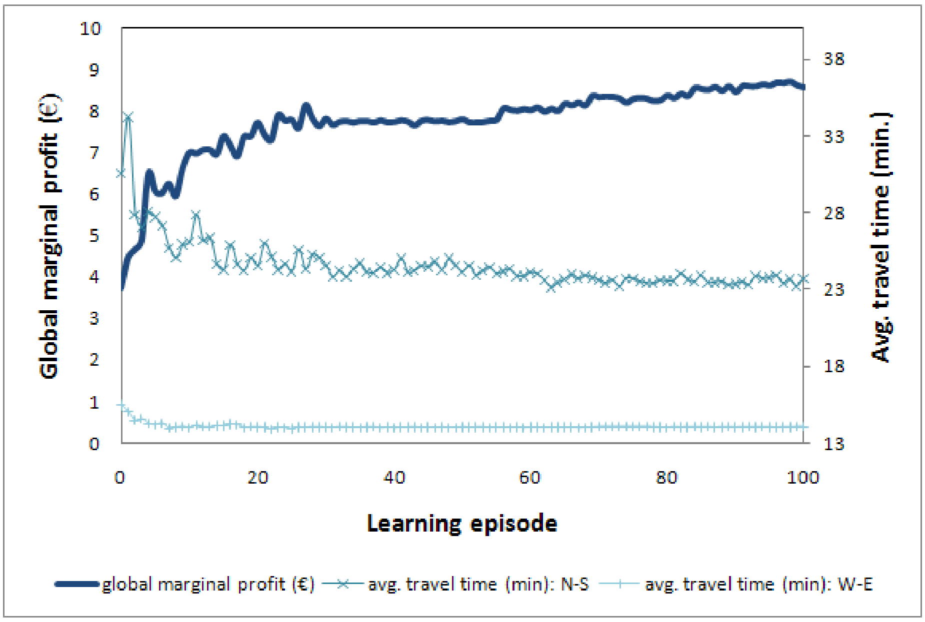


Figure 4.13: Global reward and average travel time using *ExpDR*

needs less information to be computed (each intersection manager is able to compute it locally, without communication with the other intersection managers), under the marginal profit modelling it is clearly the best option.

The above experiments are a clear clue the market is modelled in a way that it enforces an inverse relationship between the maximisation of the global marginal profit, pursued by the intersection managers, and average travel time, experienced by the driver agents. This relationship holds only in high-load situations, e.g., for the traffic flow along the North-South axis, whilst for low traffic demands the market does not beneficially affect the driver agents. This fact is very related to non functional aspects of the traffic system, i.e., the quality of service. It seems unfair to charge driver agents when the demand is low, for example during the night.

Another interesting aspect to be analysed is not only *how well* the intersection managers learn, but also *what* they learn, i.e, the prices applied by the intersection managers and the marginal profit of each of them (figure 4.16 and 4.17). Although the macroscopic results, using the two different reward functions, show the same dynamic, the learnt price vector is quite different. At first glance, it is evident that, using

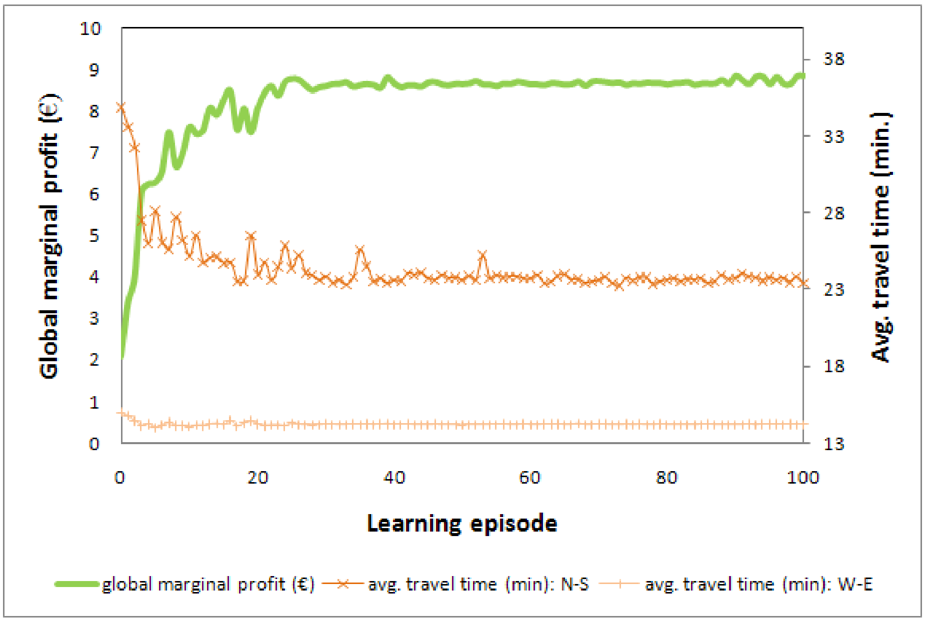


Figure 4.14: Global reward marginal and average travel time using LR

$ExpDR$, the prices are in general higher than when using LR, 0.74 euros when using $ExpDR$ versus 0.66 euros when using LR. If we look at the individual performance of each intersection manager, we notice that using $ExpDR$, some intersections have a marginal profit of 0 although they have a price greater than 0. On the contrary, using LR every intersection manager is able to generate a marginal profit, although small in some cases. This means that for the intersection managers that use $ExpDR$, given equation 4.34, either $N_a = 0$ or $N_q \gg N_a$. In other words, either they are unused and do not sell reservations ($N_a = 0$), or they sell reservations exclusively to driver agents queued at the intersection ($N_q \gg N_a$). The former is more likely since, for instance, intersections I5 and I6 have a marginal profit equal to 0 (using $ExpDR$) and since they do not lay on the shortest route is quite probable that they were not able to attract a quota of driver agents and divert them from the shortest route. In fact, the sub-path N1-I1-I2, which is part of the shortest route between N1 and S2, has the same price using the two reward functions, 1.4 euros. On the other hand, the alternative path N1-I6-I5-I2 is more expensive using $ExpDR$ (2.03 euros) and less expensive using LR (1.21 euros). With $ExpDR$ the intersection managers that govern

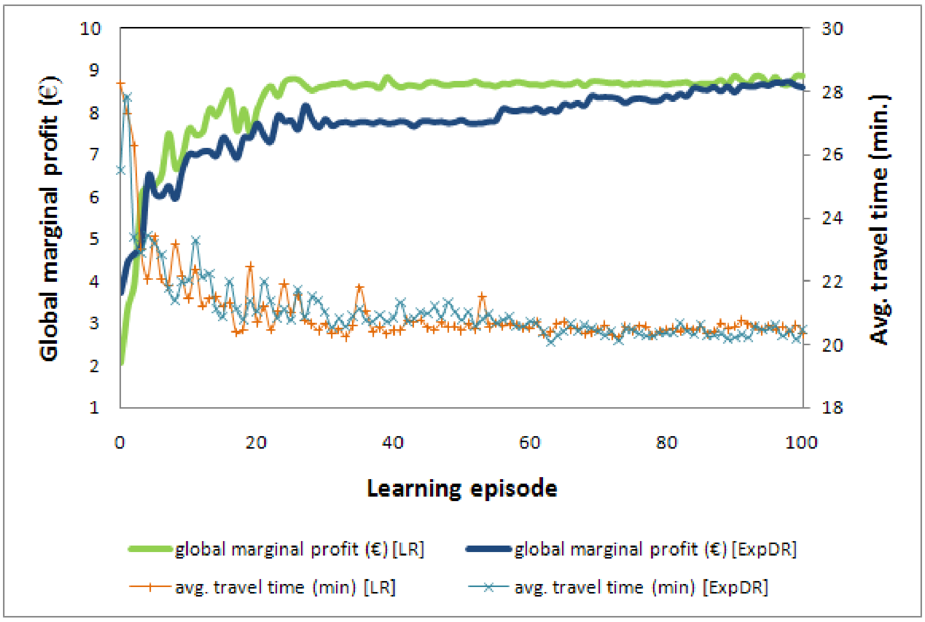


Figure 4.15: Comparison between *ExpDR* and *LR*

I5 and I6 learn a price that makes the longer sub-path N1-I6-I5-I2 more expensive than the shorter one, N1-I1-I2, so that no driver agents select that sub-path and a part of the road network remains unused. With *LR*, the intersection managers that govern I5 and I6, aiming to gain some marginal profit, attract the driver agents that are more concerned with low prices, and they create a sub-path that, albeit longer, is cheaper than the shortest one.

To pay or not to pay? In all the experiments described above, we modelled the driver agent as if it was always willing to pay a certain amount of money for its trip. Still, the possibility of travelling without paying for the reservations must be considered in the experimental scenario. In fact, a certain quota of driver agents could be interested in the possibility of travelling without any additional cost derived from the reservations. This kind of driver agent, which we call *non-payer* driver agents, has no incentives to divert from the route with the estimated shortest travel time, so that they consider only this attribute of a route ρ when they must choose the route to follow. To evaluate the impact of this kind of driver agent on the system functioning,

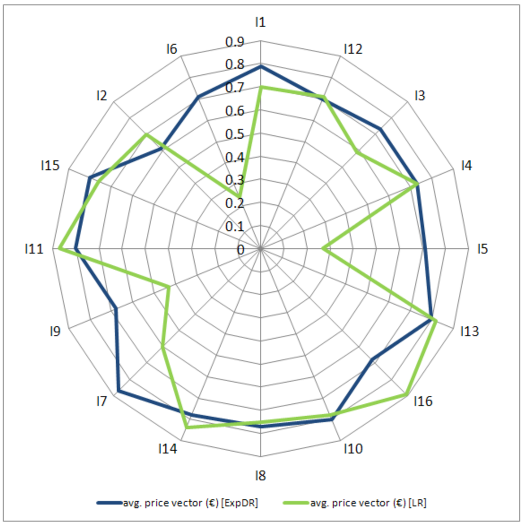


Figure 4.16: Average price vector

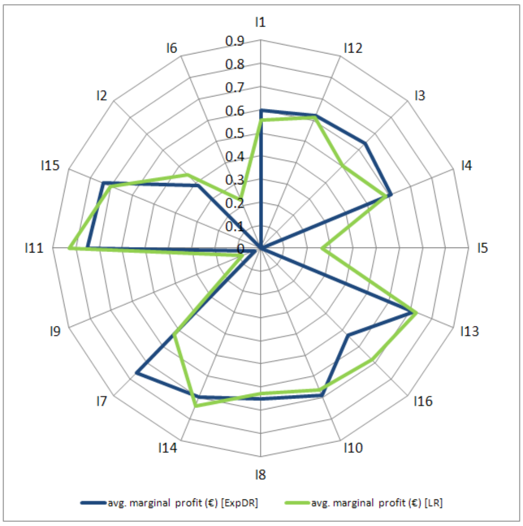


Figure 4.17: Average marginal profit

we set up a new experimental configuration. Similarly to the experimental setup described in section 4.4.1, the intersection managers use the LR to maximise the global marginal profit. We evaluated three different scenarios, with an increasing percentage of non-payer driver agents. The metric we used to evaluate the effect of the presence of non-payer driver agents is the *average travel time* (min.) of the vehicles travelling along the North-South axis. The average travel time is measured at the end of the learning performed by the intersection managers.

In general, the average travel time of the non-payer driver agent is higher than that of the payer driver agents (figure 4.18). This is because a non-payer driver agent, when approaching an intersection, reduces its speed until reaching the intersection queue, while a payer driver agent with a purchased reservation can maintain its speed. If the number of non-payer driver agents is 10% of the total, the average travel time of these settles at around 42 minutes to cover the distance between origin and destination, while the payer driver agents experience an average travel time of about 23 minutes. If the number of non-payer driver agents increases to 30% of the total, the average travel time grows by up to 65 minutes, while the average travel time of the payer driver agents is about 25 minutes. Finally, if the non-payer driver agents represent the 60% of the population of vehicles, the average travel time for this kind of driver

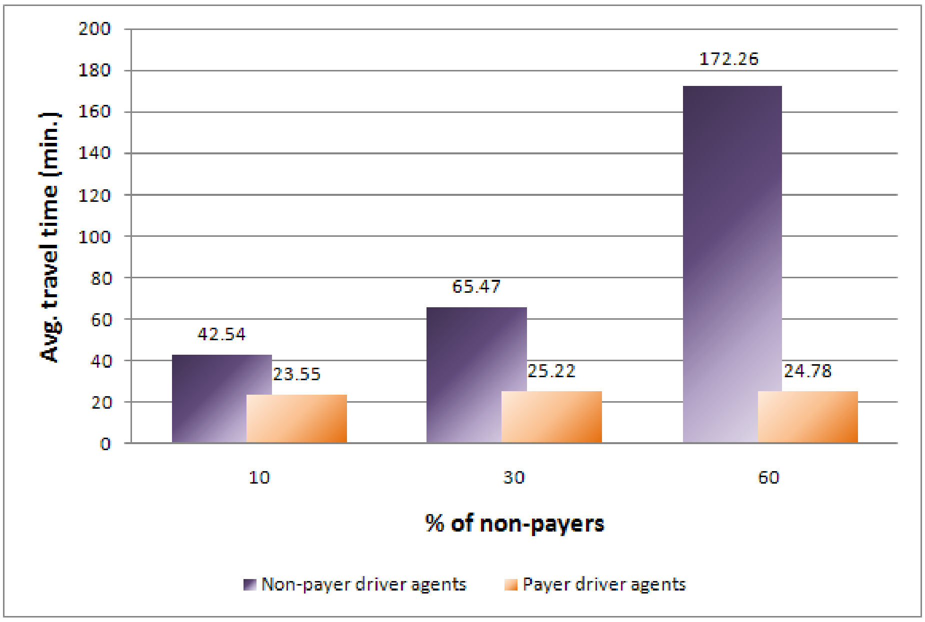


Figure 4.18: Average travel time for different percentages of non-payer driver agents

agents increases a lot and settles around 173 minutes, while the payer driver agents again basically experience the same quality of service, with an average travel time of about 23 minutes.

The intersection managers have some difficulties in maximising the marginal profit, due to the driver agents that slow down at the intersections in order to have a reservation for free. These driver agents are insensitive to the intersection manager actions, therefore these have a lot of costs that they cannot avoid. Figures 4.19 and 4.20 show the marginal profit obtained by the intersection managers at the end of the learning. As the number of non-payer driver agents increases, the global marginal profit decreases, falling from around 8 euros to around 3 euros. If we look at the individual performance of each intersection manager, we can see how the marginal profit of each intersection manager shrinks with the increase of non-payer driver agents.

From these results, it is evident that the new market rules heavily affect the performance of the system: the driver agents now have incentives to save money, thus they slow down at the intersections and, depending on the number of agents who behave in the same way, may cause severe problems to the stability of the system. *A*

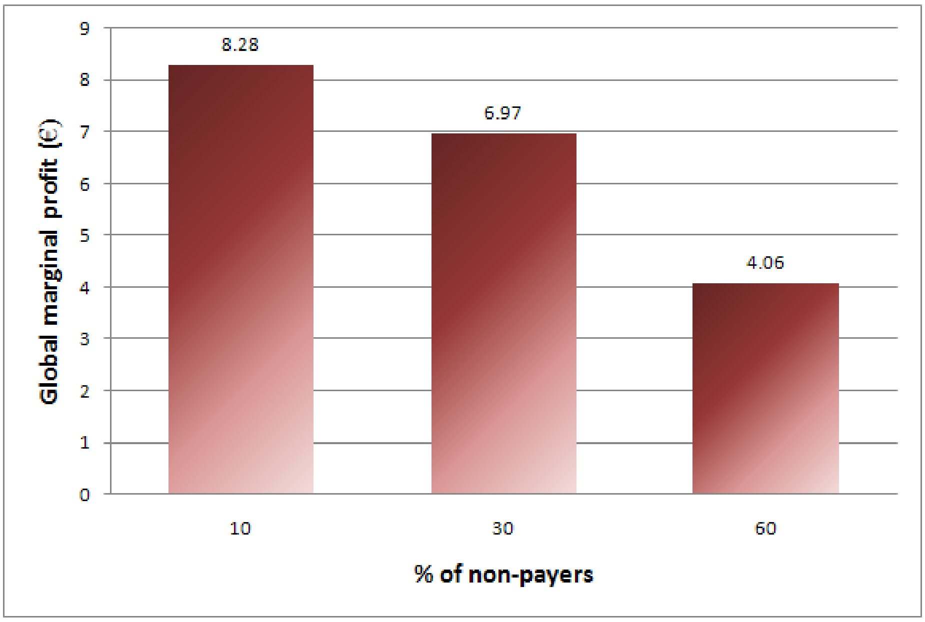


Figure 4.19: Global marginal profit for different percentages of non-payer driver agents

priori it is difficult to estimate the percentage of driver agents that are not willing to pay, so that it is difficult to predict how well the system may perform. For instance, if only a 10% of the driver agents do not pay, the system is still quite resilient, and also the non-payer driver agents experience a good enough quality of service. On the other hand, if more than 30% of the driver agents "boycott" the market, the performance of the system is severely affected.

The good news is that the payer driver agents always have a far better quality of service with respect to the non-payer driver agents, an important characteristic to foster the acceptance of the mechanism and give incentives to driver agents to join the market looking for a good deal.

4.4.2 $\mathcal{ECO}^{-}$: a competitive economy for traffic assignment

In this section, we present $\mathcal{ECO}^{-}$, a competitive economy for networks of reservation-based intersections. Again, we rely on Dresner and Stone's work [35] and we assume the availability of an advanced traffic management infrastructure that allows for reservation-based intersection control. As in the cooperative economy scenario, in

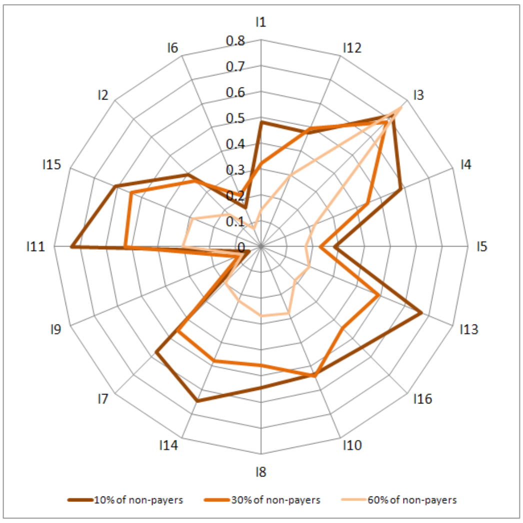


Figure 4.20: Average marginal profit for different percentages of non-payer driver agents

$\mathcal{ECO}^-$ driver agents trade with the intersection managers in a virtual marketplace, purchasing reservations to cross intersections when they travel through the urban road network. Still, in contrast to the cooperative economy scenario, we designed the intersection managers as competitors that strive to sell the resources they supply, that is, the reservations. Each intersection manager applies a pricing policy which is founded on the general equilibrium theory [105]. The general market equilibrium is a situation where the amount of resources sought by buyers is equal to the amount of resources produced by suppliers.

This section is organised as follows. We first introduce our assumptions about the infrastructure that we take for granted, and the economic model of the market that is established between driver agents and intersection managers. Then we define the structure and the behavioural model of the intersection managers. Finally we perform an evaluation of the model.

Infrastructure model

To implement the competitive economy, we rely on the same infrastructure model defined in $\mathcal{ECO}^+$. A communication medium that enables infrastructure-to-infrastructure communication and vehicle-to-infrastructure communication exists. We also assume that an electronic payment system is available, as well as a price index board where the prices of reservations are published.

Market model

The rules of the marketplace are essentially those defined for $\mathcal{ECO}^+$. The driver agent can purchase a reservation when it is approaching the intersection or it can receive a reservation when it is queued at the intersection. Additionally, if it holds a confirmed reservation but it realises that it cannot be at the intersection at the time defined in the reservation confirmation, it is authorised to withdraw the reservation.

Intersection manager model

In a competitive multiagent system the individual agents are assumed to behave selfishly, competing with each other to maximise their private utility. Nevertheless, this does not necessarily lead the system to perform poorly from a global point of view. Mechanism design [30], for instance, studies how to define the protocols that rule the interaction between agents in such a way that some desirable properties emerge, e.g., an efficient allocation of resource.

In $\mathcal{ECO}^-$, each intersection manager competes with all the others for the supply of the resource they trade, i.e., the reservations. Our goal as market designers is to reach the general market equilibrium [105], a situation where the amount of resources sought by buyers is equal to the amount of resources produced by suppliers. A condition for the existence of such equilibrium is that the utility functions of the buyers must satisfy the *gross substitutes* condition (GS). Informally, the GS condition states that when the price of one good goes up, demand for another good should not go down. More formally, let be Ω the set of items that compose the economy and $\mathbf{p} \in \mathbb{R}^\Omega$ a *price vector* that assigns a price for each item $\in \Omega$. Be $u : 2^\Omega \to \mathbb{R}$ a *utility function* on Ω, which assigns a value to each bundle $\mathcal{X} \subset \Omega$. With each utility function u we associate the *net utility function* $v : 2^\Omega \times \mathbb{R}^m \to \mathbb{R}$, which is defined by

$$v(\mathcal{X}, \mathbf{p}) = u(\mathcal{X}) - p(\mathcal{X})$$

where $p(\mathcal{X}) = \sum_{x \in \mathcal{X}} \mathbf{p}_x$.

For any utility function u, the *demand correspondence* $D : \mathbb{R}^{\Omega} \rightarrow 2^{\Omega}$ is defined by:

$$D(\mathbf{p}) = argmax_{\mathcal{X} \in \Omega} v(\mathcal{X}, \mathbf{p})$$

As introduced in Kelso and Crawford [56], u satisfies the GS condition if for any two price vectors $\mathbf{p}$ and $\mathbf{q}$ such that $\mathbf{q} \geq \mathbf{p}$, and any $\mathcal{X} \in D(\mathbf{p})$, exists $\mathcal{Y} \in D(\mathbf{q})$ such that $\{x \in \mathcal{X} \mid p_x = q_x\} \subset \mathcal{Y}$.

The price vector $\mathbf{p}^*$ that corresponds to the general market equilibrium is in general computed through a walrasian auction [21][26], which involves a set of buyers $\mathcal{B}$ and a set of suppliers $\mathcal{J}$. At time t, each buyer $b \in \mathcal{B}$ notifies to the suppliers the quantity of resources it is willing to buy, given the actual price vector $\mathbf{p}^t$. With this information, each supplier $j \in \mathcal{J}$ computes the difference between the demand for the resource it produces, d_j, and the supply of such resource, s_j. If there is excess supply ($s_j > d_j$), the prices are lowered, whilst if there is excess demand ($d_j > s_j$) the prices are raised. The new price vector $\mathbf{p}^{t+1}$ is communicated to the buyers that iteratively compute the new demand. No transactions take place at disequilibrium prices, and the process continues until the equilibrium price is reached, that is, when $d_j = s_j \ \forall j \in \mathcal{J}$. Only at that point the transactions take place and the resources are transferred from the suppliers to the buyers by means of money.

In $\mathcal{ECO}^-$, the buyers are the driver agents, the suppliers are the intersection managers, and the traded resources are the reservations. More precisely, an intersection manager sells reservations to the driver agents that want to cross the intersection j coming from one of the incoming links. Let be $\mathcal{L}_j$ the set of incoming links of intersection j. For each incoming link $l_h \in \mathcal{L}_j$, the intersection manager defines the following variables:

- Current price $p_j^t(l_h)$: is the price applied by the intersection manager j to the reservations sold to the driver agents that comes from the incoming link l_h.

- Total demand $d_j^t(l_h \mid p_j^t(l_h))$: represents the total demand for the reservations from the incoming link l_h that the intersection manager j observes at time t, given the current price $p_j^t(l_h)$. It is given by the number of vehicles that want to cross intersection j coming from link l_h at time t.

- Supply $s_j(l_h)$: defines the reservations supplied by the intersection manager j for the incoming link l_h. It is a constant and represents the number of vehicles crossing intersection j coming from link l_h at time t that intersection manager j is willing to serve.

- Excess demand $z_j^t(l_h \mid p_j^t(l_h))$: it is given by the difference between the total demand at time t and the supply, $z_j^t(l_h \mid p_j^t(l_h)) = d_j^t(l_h \mid p_j^t(l_h)) - s_j(l_h)$.

Given the set of all intersection managers that are operating in the market, $\mathcal{J}$, we define the price vector $\mathbf{p}$ as the vector of the prices applied by each intersection manager $j \in \mathcal{J}$ for the reservations sold to the driver agents that comes from the incoming link $l_h \in \mathcal{L}_j$. In compact notation:

$$\mathbf{p} = [\ p_j(l_h)\ ] \quad \forall j \in \mathcal{J},\ \forall l_h \in \mathcal{L}_j \tag{4.38}$$

The market in $\mathcal{ECO}^-$ is said to be in equilibrium at price vector $\mathbf{p}^*$ if $z_j^t(l_h \mid p_j^*(l_h)) = 0\ \forall j \in \mathcal{J},\ \forall l_h \in \mathcal{L}_j$, that is, if the demand and the supply are mapped by the price vector $\mathbf{p}^*$.

To implement the walrasian auction described at the beginning of this section, each buyer (i.e., driver agent) should communicate to the suppliers (i.e., intersection managers) the route that it is willing to choose, given the current price vector $\mathbf{p}^t$. With this information, each intersection manager j computes the demand $d_j^t(l_h \mid p_j^t(l_h))$ as well as the excess demand $z_j^t(l_h \mid p_j^t(l_h))$, $\forall l_h \in \mathcal{L}_j$. Then, each intersection manager j adjusts the prices $p_j^t(l_h)$ for all the incoming links $l_h \in \mathcal{L}_j$, lowering them if there is excess supply ($z_j^t(l_h \mid p_j^t(l_h)) < 0$) and raising them if there is excess demand ($z_j^t(l_h \mid p_j^t(l_h)) > 0$). The new price vector $\mathbf{p}^{t+1}$ is communicated to the driver agents that iteratively choose their new desired route, on the basis of the new price vector $\mathbf{p}^{t+1}$. Once the equilibrium price $\mathbf{p}^*$ is computed, the trading transactions take

Algorithm 2 Intersection manager price update

$t \leftarrow 0$
for all $l_h \in \mathcal{L}_j$ **do**
  $p_j^t(l_h) \leftarrow \epsilon$
  $s_j(l_h) \leftarrow initialValue$
end for
while true **do**
  for all $l_h \in \mathcal{L}_j$ **do**
    $d_j^t(l_h) \leftarrow evaluateDemand$
    $z_j^t(l_h) \leftarrow d_j^t(l_h) - s_j(l_h)$
    $p_j^t(l_h) \leftarrow p_j^t(l_h) + p_j^t(l_h) \cdot \dfrac{z_j^t(l_h)}{s_j(l_h)}$
  end for
  $t \leftarrow t + 1$
end while

place and each driver agent buys the required reservations at the intersections that lay on its route.

The walrasian auction relies on quite strong assumptions, which make a direct implementation in the traffic domain hard. For instance, the set of buyers is assumed to be fixed during the auction, which means for the traffic domain that new driver agents may not join an auction until it ends. Also the fact that no transactions can take place at disequilibrium prices is a strong assumption for the traffic domain: it is unreasonable for all the driver agents to wait to reach the equilibrium before choosing the desired route and starting to travel. Finally, given the infrastructure model described in section 4.4.1, a driver agent is actually able to transfer money to an intersection manager when it is spatially close to it, that is, when it is already travelling along its desired route.

Thus we implement a market that aims at reaching the general equilibrium such as the walrasian auction, but that works on a continuous basis, with driver agents that join and leave the market dynamically, and with transactions that take place at every moment. To reach the general equilibrium, each intersection manager applies the price update strategy sketched in algorithm 2. At time t, each intersection manager j computes independently the excess demand $z_j^t(l_h \mid p_j^t(l_h))$ and updates the price $p_j^t(l_h)$ using the formula [21][105]:

$$p_j^{t+1}(l_h) \leftarrow max\left[\epsilon,\ p_j^t(l_h) + p_j^t(l_h) \cdot \frac{z_j^t(l_h \mid p_j^t(l_h))}{s_j(l_h)}\right] \tag{4.39}$$

where ϵ is the minimum price and $s_j(l_h)$ is the supply of the intersection manager j for the incoming link l_h. The definition of ϵ and $s_j(l_h)$ is a design decision that may affect the functioning of the market: i) ϵ is the minimum price that an intersection manager charges for the reservations that it sells, and ii) $s_j(l_h)$ is the number of vehicles above which the intersection manager considers that there is an excess demand and it starts raising prices. In $\mathcal{ECO}^+$ we saw how the cooperative economy charged the driver agents of the minority flow, although these did not experience any increase in quality of service. Thus, we claim that driver agents that travel through road network links with low demand shall not incur in any costs. For this reason, we chose $\epsilon = 0$. To define the supply $s_j(l_h)$, we rely on the fundamental diagram of traffic flow [64]. Let ρ^{opt} be the density that maximises the traffic flow on link l_h (see figure 4.21). We chose $s_j(l_h) = 0.5 \cdot \rho^{opt} \cdot ||l_h||$, where $||l_h||$ is the length of link l_h. In other words, the intersection manager considers that there is an excess demand for capacity coming from link l_h when the density on that link reaches 50% of the optimal density.

Evaluating $\mathcal{ECO}^-$

Driver agent model. We use the same driver model that we used in $\mathcal{ECO}^+$. The driver multi-attribute utility function is defined as a 2-attribute function:

$$U(\rho_i) = w_{TT} \cdot u_{TT}(\rho_i) + w_K \cdot u_K(\rho_i) \tag{4.40}$$

where w_{TT} is the weight of the estimated travel time attribute, w_K is the weight of the reservations cost attribute, $u_{TT}(\rho_i)$ and $u_K(\rho_i)$ are the normalised utilities of route ρ_i against the estimated travel time attribute and the reservation cost attribute respectively. These utilities are defined by:

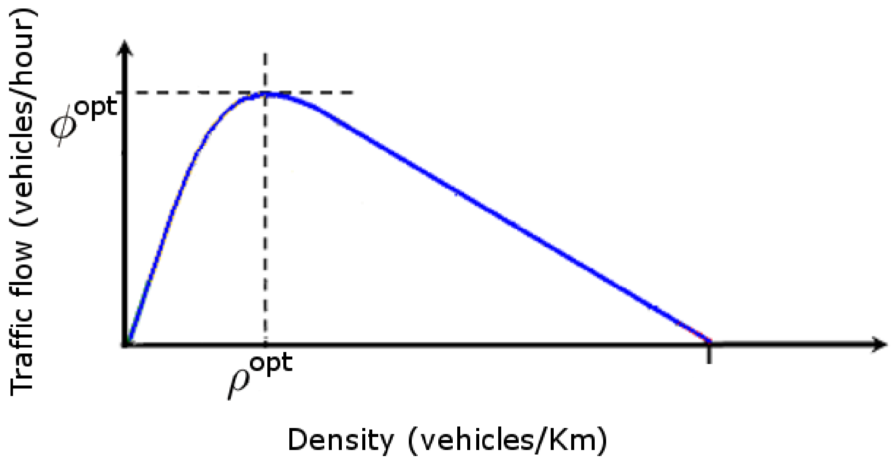


Figure 4.21: Fundamental diagram of traffic flow

$$u_{TT}(\rho_i) = \frac{M_{TT} - TT(\rho_i)}{M_{TT} - m_{TT}} \tag{4.41}$$

$$u_K(\rho_i) = \frac{M_K - K(\rho_i)}{M_K - m_K} \tag{4.42}$$

where $TT(\rho_i)$ is the estimated travel time of route ρ_i, $K(\rho_i)$ is the reservations cost of route ρ_i, $M_{TT} = max_{\rho_i \in \mathcal{C}} TT(\rho_i)$, $m_{TT} = min_{\rho_i \in \mathcal{C}} TT(\rho_i)$, $M_K = max_{\rho_i \in \mathcal{C}} K(\rho_i)$ and $m_K = min_{\rho_i \in \mathcal{C}} K(\rho_i)$. Once the utility of the alternative routes $\rho_1, \ldots, \rho_k$ has been computed, the driver agent always selects the route with the highest utility value.

This utility function satisfies the GS condition. As said before, the GS condition informally states that when the price of one good goes up, demand for another good should not go down. Consider the following simplified case, with two goods, i.e., two available routes, ρ_1 and ρ_2. Suppose that $TT(\rho_1) = 10$ and $TT(\rho_2) = 20$ and that $w_{TT} = 0.3$ and $w_K = 0.7$. The functional form of $U(\rho_1)$, for different values of $K(\rho_2)$, is depicted in figure 4.22. If the price of route ρ_2 increases, for example from 1 to 2, the $U(\rho_1)$ increases as well, so that the demand for route ρ_1 should not go down. In fact, more driver agents will prefer the route ρ_1, that becomes more attractive due to the price increase of route ρ_2.

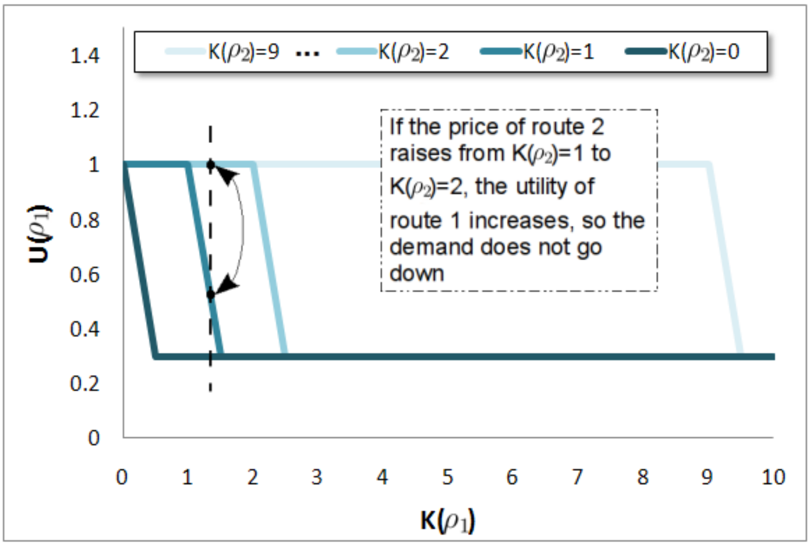


Figure 4.22: Driver agent utility function and the GS condition

Experimental results. Although our work does not depend on the underlying road network, we chose a (simplified) topology of the urban road network of the city of Madrid (see figure 4.23) for our empirical evaluation rather than an unrealistic, lattice-like, network. The network is characterised by several freeways that connect the downtown with the surroundings and a ring road. Each big dark vertex in figure 4.23, if it connects three or more links, is modelled as a reservation-based intersection. We aimed at recreating a typical morning peak scenario, with more than 11 thousands vehicles that depart within a time window of 50 minutes. The vehicles that travel from and to 7 destinations outside the city (marked with O_1 up to O_7 in figure 4.23) form the traffic under evaluation.

For the driver agent model, we set the parameter $w_{TT} = 1 - w_K$ and we sample w_{TT} using a normal distribution with mean 0.5 and variance 0.25. Since the prices of the reservations changes dynamically, a driver agent may change "on-the-fly" the route it selected at the beginning of its journey, therefore reacting to the market fluctuations.

To evaluate $\mathcal{ECO}^-$ we used two different types of metrics, one related to the vehicles and one related to the network. The network-related metric was the den-

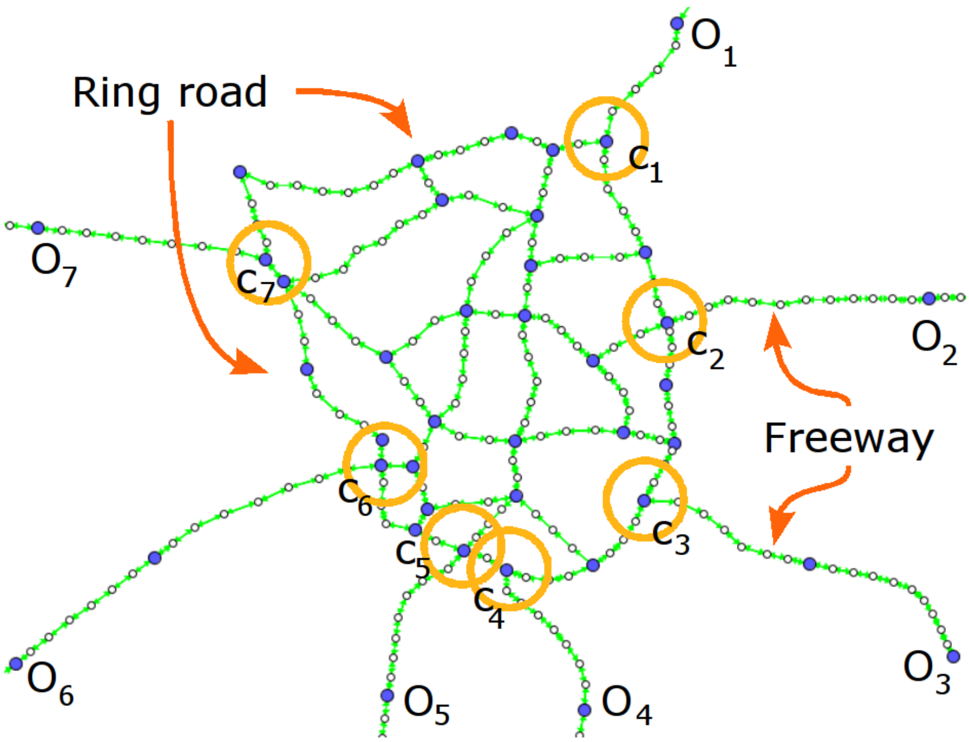


Figure 4.23: Network used to evaluate $\mathcal{ECO}^{-}$

sity variations in correspondence of 7 critical intersections (marked with $c_1 \dots c_7$ in figure 4.23), which connect the freeways going downtown with the ring road. The vehicle-related metrics are the average travel time and the moving average travel time, all grouped by the origin-destination (O-D) pair.

- Average travel time (for a given O-D pair):

$$\sum_i TT(\rho_i)/N$$

  where $TT(\rho_i)$ is the real travel time experienced by the driver agent i on its selected route ρ_i and N is the total number of driver agents for the given O-D pair.

- Moving average travel time (for a given O-D pair): this metric is intended to measure how the average travel time evolves during the simulation. This metric is initialised to 0 and calculated as follows: once a driver agent i concludes its trip, the travel time $TT(\rho_i)$ is computed and the moving average travel time TT^{avg} is updated with the formula

$$TT^{avg} = TT^{avg} + (TT(\rho_i) - TT^{avg})/(n+1)$$

where n is the number of driver agents that have completed their trips so far.

The evaluation is performed by running the simulator with two different configurations: in the first one, the intersections are governed by intersection managers that compete in the market for the supply of the reservations; in the second one, the intersections are governed by intersection managers that only regulate the transit with the usual FCFS policy. In the following tables and figures we refer to the two configurations with the abbreviations $\mathcal{ECO}^-$ and FCFS.

Network-related metrics. An important metric that is used to evaluate the effects of the trading activity between driver agents and intersection managers is the density variation over time in correspondence of the critical intersections c_1 to c_7.

The plots of the results are shown in figure 4.24. In general, the density tends to be lower in the competitive economy compared with the system regulated only by FCFS intersection managers. At the least demanded intersections c_1, c_2 and c_7 (where the density is below the optimal one) there is no substantial difference between $\mathcal{ECO}^-$ and FCFS. These critical intersections are less demanded due to the topology of the network. In fact, less origins (also far-between) are located in the northern part (O_1, O_2 and O_7).

At the critical intersections c_3, c_4 and c_6, the vehicles density when the intersection managers apply the price update strategy defined by $\mathcal{ECO}^-$ is always below the density when the network is regulated by simple FCFS intersection managers, especially in the case of intersections c_4 and c_6 where the density exceeds the optimal one to a small extent and for a limited period of time.

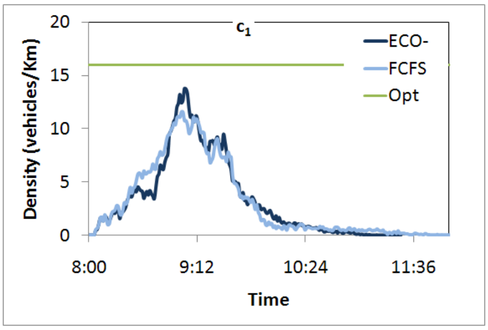
At intersection c_5, the density when $\mathcal{ECO}^-$ is running has a higher peak around $9:30$, but the density then later starts to exceed the optimal density and begins to fall below the optimal density earlier. To assess which of the two scenarios, $\mathcal{ECO}^-$ or FCFS, is preferable, we calculated the integral of the density curves, measured in the interval when the curve is above the optimal density, formally:

$$\int_{t_1}^{t_2} \delta_{\mathcal{ECO}^-}(t)dt \qquad \int_{t_1}^{t_2} \delta_{FCFS}(t)dt \tag{4.43}$$

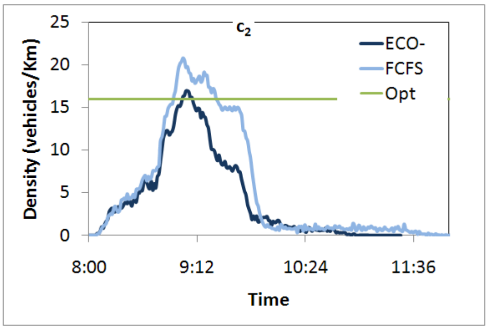
where $\delta_{\mathcal{ECO}^-}$ and δ_{FCFS} are the density functions, $t_1 = min(\ t \mid \delta_{\mathcal{ECO}^-}(t) > \rho_{opt},\ \ t \mid \delta_{FCFS}(t) > \rho_{opt}\ )$ and $t_2 = max(\ t \mid \delta_{\mathcal{ECO}^-}(t) < \rho_{opt},\ \ t \mid \delta_{FCFS}(t) > \rho_{opt}\ )$. This metric is lower when the reservations are allocated through the competitive economy (70.24 $vehicles \cdot h/km$ versus 105.07 $vehicles \cdot h/km$).

Thus, we can conclude that the competitive economy generates a better balanced network, since the price fluctuations force the demand to change towards less expensive intersections. Such fluctuations contribute to create a system in dynamic equilibrium by a matter of uneven development, where unused intersections become cheaper while congested ones become very expensive.

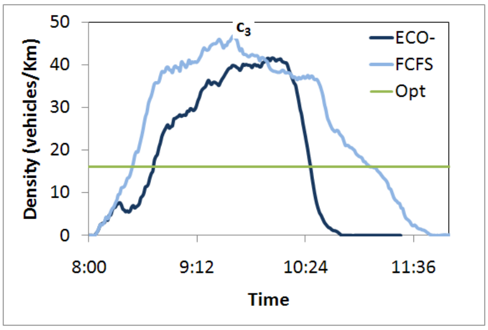
Vehicle-related metrics. Table 4.9 shows the average travel time of the driver agents, according to their origin-destination pairs, when the reservations are allocated through the competitive market ($\mathcal{ECO}^-$) and when they are granted with the usual FCFS policy. The competitive economy generates a net reduction of the average travel time for 30 of 42 origin-destination pairs. Such reduction is in general noteworthy for the busiest routes, such as those that connect O_6 and O_7 with O_3 and O_4. The plots of the moving average travel time are shown in figure B.1 (see appendix B). These plots show how the average travel time of the driver agents of a given origin-destination pair evolves during the simulation. It is noticeable that for the aforementioned 30 of 42 origin-destination pairs the moving average travel time of $\mathcal{ECO}^-$ is lower than that of FCFS, especially when the road network reaches saturation levels.



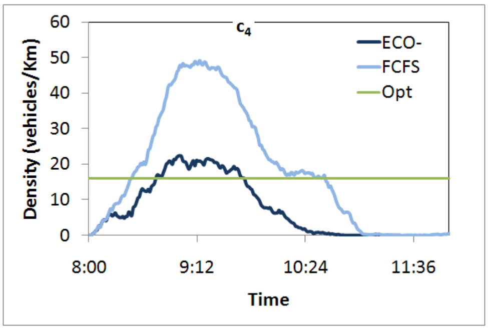
(1) Intersection c_1



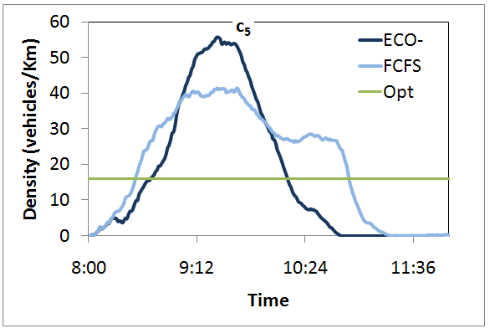
(2) Intersection c_2



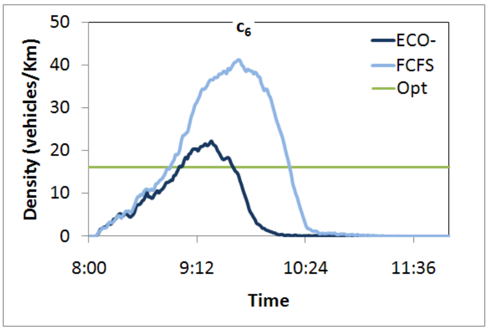
(3) Intersection c_3



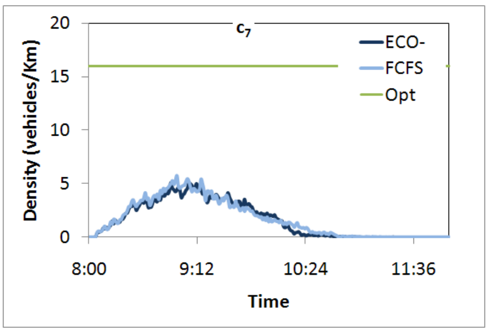
(4) Intersection c_4



(5) Intersection c_5



(6) Intersection c_6



(7) Intersection c_7

Figure 4.24: Density in the critical intersections under evaluation

Table 4.9: Average travel time (min): $\mathcal{ECO}^-$ vs. FCFS

Origin		Destination O_1	O_2	O_3	O_4	O_5	O_6	O_7
O_1	$\mathcal{ECO}^-$	-	12.08	**19.58**	**26.69**	**30.75**	**21.17**	14.13
	FCFS	-	**11.98**	22.88	35.13	43.56	21.35	**13.82**
O_2	$\mathcal{ECO}^-$	11.25	-	**14.16**	**19.01**	**23.72**	**24.00**	20.88
	FCFS	**10.14**	-	16.50	25.86	31.04	38.09	**19.50**
O_3	$\mathcal{ECO}^-$	15.57	10.79	-	**9.18**	**13.98**	**18.54**	**24.95**
	FCFS	**13.34**	**9.75**	-	12.21	17.63	23.68	31.73
O_4	$\mathcal{ECO}^-$	**24.79**	**20.39**	**11.61**	-	**8.20**	**14.34**	**21.66**
	FCFS	26.94	22.58	13.91	-	10.04	15.73	22.74
O_5	$\mathcal{ECO}^-$	**26.80**	**22.82**	**16.29**	**7.47**	-	11.10	19.46
	FCFS	32.16	30.61	21.53	8.83	-	**10.77**	**17.65**
O_6	$\mathcal{ECO}^-$	23.16	**27.30**	**25.30**	**16.39**	**12.12**	-	16.58
	FCFS	**22.51**	57.00	41.05	24.68	19.02	-	**13.73**
O_7	$\mathcal{ECO}^-$	15.04	23.51	**31.67**	**24.44**	**19.11**	**11.69**	-
	FCFS	**14.30**	**23.25**	56.42	34.99	31.23	11.99	-

4.5 $\mathcal{ECO}_{CA}^{-}$: an integrated computational economy for traffic assignment and control

In section 3.3, we introduced an auction-based policy for the control of a single intersection. The experimental results showed that this policy was quite effective in allocating the reservations to the driver agents that value them the most. Bidders who bid high usually experience a great reduction in delay (about 30% less). Still, this policy on its own showed a couple of drawbacks. First, it fosters the attainment of a user optimum rather than a global one, therefore it pays a social price, in the form of greater average delay for the entire population of driver agents. Furthermore, it is possible that even wealthy driver agents, in high-load situations, could not get a reservation, due to the decreasing reservation distance or due to driver agents in front of them that does not want to allocate a lot of money to acquire a reservation.

For this reason, if we focus on an urban road network, an integrated strategy is needed, which acts not only on the traffic control, but also on the traffic assignment. Traffic assignment strategies aim at making the task of the traffic controllers easier, by means of a better distribution of the traffic demand. In this section we introduce $\mathcal{ECO}_{CA}^{-}$, an integrated computational economy for traffic assignment and control, which combines the adaptive traffic assignment strategy $\mathcal{ECO}^{-}$ with the auction-based control policy described in section 3.3.

From the market perspective, the intersection manager is the supplier of the reservations that are allocated through the combinatorial auction. Thus, it may control the *reserve price* of the auctioned reservations. In auction terminology, the reserve price is the minimum price at which the intersection manager is willing to sell. Depending on the intersection usage, it may apply pricing strategies and raise (or lower) the reservation price. We model each intersection manager in such a way that they compete for the driver agents as in $\mathcal{ECO}^{-}$, raising the reserve price in case of increasing demand or lowering it in case of decreasing demand. The pricing strategy is founded on the general market equilibrium theory [21][26][105]. The adaptive and concurrent pricing strategy applied by the intersection managers are in charge of computing in a distributed way the price vector $\mathbf{p}^*$ that corresponds to the general market equilibrium.

The computation of $\mathbf{p}^*$ is performed as in $\mathcal{ECO}^-$. Let be $\mathcal{L}_j$ the set of incoming links of intersection j. For each incoming link $l_h \in \mathcal{L}_j$, the intersection manager defines the current reserve price $p_j^t(l_h)$, the total demand $d_j^t(l_h \mid p_j^t(l_h))$, the supply $s_j(l_h)$ and the excess demand $z_j^t(l_h \mid p_j^t(l_h))$. Each intersection manager applies the reserve price update strategy sketched in algorithm 2. At time t, each intersection manager j computes, independently from each other, the excess demand $z_j^t(l_h \mid p_j^t(l_h))$ and updates the price $p_j^t(l_h)$ using the formula [21][105]:

$$p_j^{t+1}(l_h) \leftarrow max\left[\epsilon,\ p_j^t(l_h) + p_j^t(l_h) \cdot \frac{z_j^t(l_h \mid p_j^t(l_h))}{s_j(l_h)}\right] \tag{4.44}$$

where ϵ is the minimum reserve price and $s_j(l_h)$ is the number of driver agents that the intersection manager j wants to participate in each auction. We set ϵ and $s_j(l_h)$ as in $\mathcal{ECO}^-$, that is, the minimum reserve price (ϵ) is set to 0 (driver agents that travel on network links with low demand shall incur the lowest costs as possible) and the number of vehicles above which the intersection manager considers that there is an excess demand ($s_j(l_h)$) is set to 50% of the optimal density.

Evaluating $\mathcal{ECO}^-_{CA}$

Driver agent model. The driver model that we used in the empirical evaluation is slightly different from that used in $\mathcal{ECO}^-$. In this model, each driver agent holds a private valuation of the bids that it is willing to submit to pass through the intersections of its chosen route, defined by the variable b_i. Given the monetary constraint, the driver agent selects the most preferred route ρ^*, taking into consideration the estimated travel time associated with the route. A route ρ is modelled as an ordered list of links, $\rho = [l_1 \ \ldots \ l_M]$, each of them characterised by two attributes, namely travel time at free flow

$$TT(l_k) = \frac{||l_k||}{v_{max}(l_k)} \tag{4.45}$$

and reserve price

$$K(l_k) = \begin{cases} p_j^t(l_k) & if\ l_k \in \mathcal{L}_j \\ 0 & otherwise \end{cases} \tag{4.46}$$

where $||l_k||$ is the length of link l_k, $v_{max}(l_k)$ is the maximum allowed speed on link l_k, and $p_j^t(l_k)$ is the reserve price set by the intersection manager j that governs the intersection to which the link l_k is connected. The summatory over all the links of ρ gives the travel time at free flow of the entire route ρ:

$$TT(\rho) = \sum_{k=1}^{M} TT(l_k) \tag{4.47}$$

Given b_i, the driver agent builds the choice-set $\mathcal{C}$, formed by those routes whose intersections have a reserve price lower than the bid b_i:

$$\mathcal{C} = \{\rho_1, \ldots, \rho_N \mid K(l_k) \leq b_i\ \forall l_k \in \rho_x\} \tag{4.48}$$

Once the choice-set is built, the driver agent selects the shortest route $\rho^* = argmin_{\rho \in \mathcal{C}} TT(\rho)$. Since the reserve prices change with time, the driver agents may react to the price fluctuations and rearrange its route on-the-fly.

Experimental results. To evaluate $\mathcal{ECO}_{CA}^{-}$ we again recreated a typical morning peak scenario, using the same network topology of figure 4.23. Each big dark vertex in figure 4.23, if it connects three or more links, is modelled again as a reservation-based intersection, whose intersection manager applies the reserve price update strategy detailed in this section and assigns reservations to the driver agents using the auction-based policy described in section 3.3. We are interested in two different types of properties. From one side we must evaluate whether or not the integrated management policy (traffic control+traffic assignment) guarantees lower delays to the driver agents that submit higher bids (user optimum). For this purpose, we calculate the average (percent) increase of the vehicles' travel times, defined as

$$\frac{TT_i^{real} - TT^{lower\ bound}}{TT^{lower\ bound}} \tag{4.49}$$

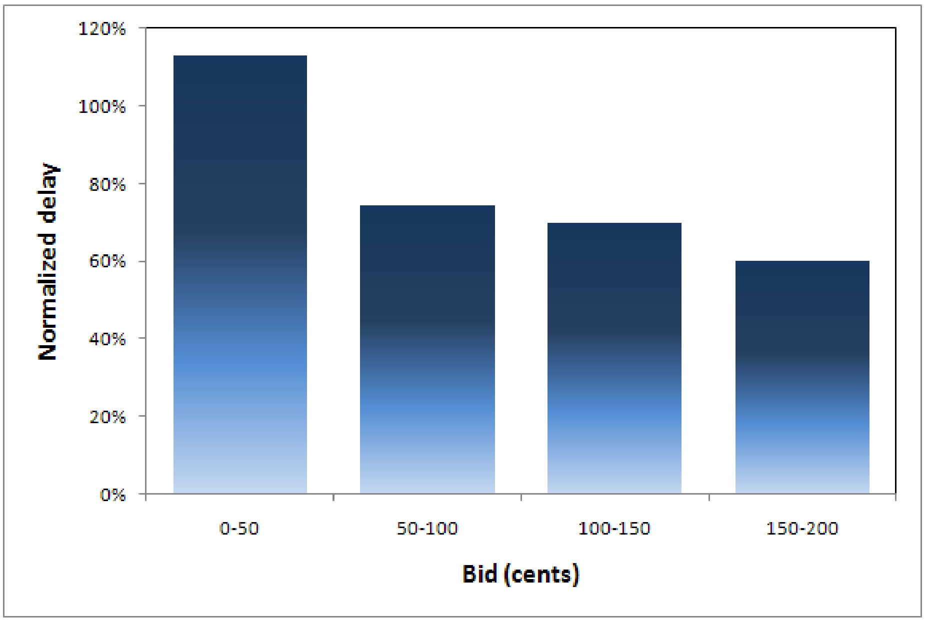


Figure 4.25: Relation between normalised delay and bid

with TT_i^{real} being the observed travel time for vehicle i from an origin to a destination, and $TT^{lower\ bound}$ the travel time from the same origin to the same destination if the vehicle could cross each intersection unhindered[5]. For simplicity, we refer to the percent increase of the travel time with the term normalised delay.

From the other side, we would set up a system that is fair to the entire population of driver agents, guaranteeing lower average delays (global optimum). Thus, we compared our integrated policy with networks of intersections governed by the first-come-first-served control policy without assignment[6]. The aim is to evaluate the global performance (in terms of average travel time) of the sophisticated $\mathcal{ECO}_{CA}^{-}$ policy compared to the rather simple FCFS policy, and to detect any potential social cost of $\mathcal{ECO}_{CA}^{-}$, similar to the one reported in section 3.3.

Figure 4.25 plots the relation between bid value and normalised delay of the

[5]This ratio enables us to aggregate the results of driver agents even though they have different origins and/or destinations.

[6]We assume that in this case the driver agents choose the shortest route from their origin to their destination.

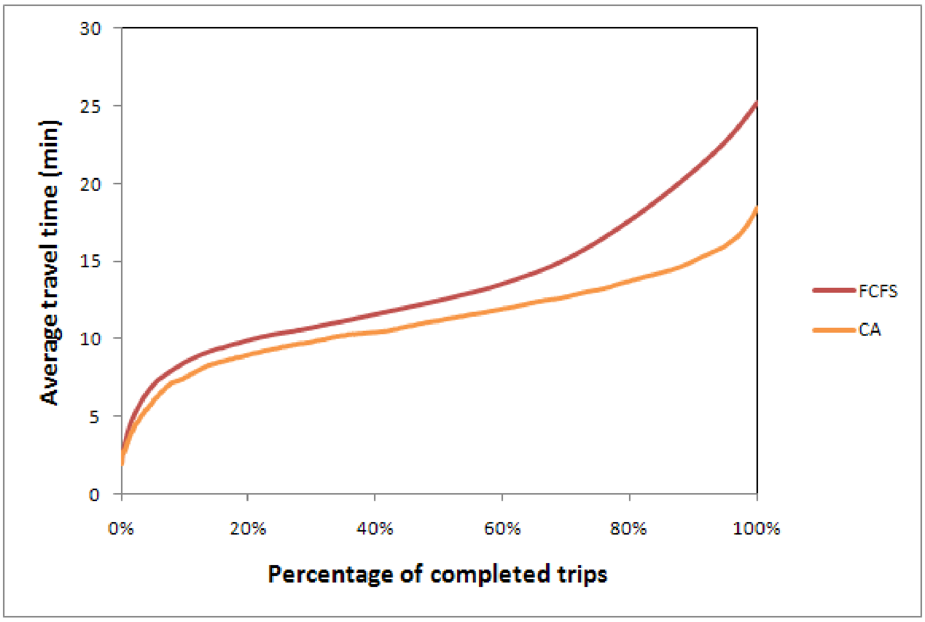


Figure 4.26: Overall average travel time

population of driver agents. As in the experiments of section 3.3, it is possible to appreciate an inverse relation between these two quantities: the driver agents that submit bids between 150 and 200 cents reduce the delay by about 50% compared to those which bid less than 50 cents.

Table 4.10 shows the average travel time of the driver agents, according to their origin-destination pairs, when the intersection managers use the $\mathcal{ECO}^{-}_{CA}$ policy and when the reservations are granted with the usual FCFS policy. With $\mathcal{ECO}^{-}_{CA}$, there is a net reduction of the average travel time for 29 of 42 origin-destination pairs. Such reduction is in general noteworthy for the busiest routes, such as those that connect O_6 and O_7 with O_3 and O_4. The plots of the moving average travel time are shown in figure C.1 (see appendix C).

Finally, figure 4.26 plots the evolution of the overall average travel time. This plot shows how the average travel time of the entire population of driver agents evolves during the simulation. This overall average travel time is calculated as in section 4.4.2. In the beginning, the average travel time is similar for both control policies, but as the number of driver agents that populate the network (i.e., its load) increases, it grows significantly faster with the FCFS than with the $\mathcal{ECO}^{-}_{CA}$ policy. This is due to the fact that the higher the network load, the more relevant the different reserve prices: with the $\mathcal{ECO}^{-}_{CA}$ policy, more drivers choose a route through less expensive intersections, thus leading to less demand at "bottleneck" intersections. There are two consequences: (1) in line with the results of section 3.3 (figure 3.12), lower demand leads to a lower "social cost" of the auction-based policy with respect to FCFS at intersection level; and (2) a more homogeneous distribution of vehicles over the network leads to a better use of network resources, and thus to lower average travel times. Our experiments show that, in general, the gains obtained by (2) outweigh the overhead introduced by (1) with respect to social welfare (i.e., average travel time). In summary, the fluctuations of the reserve prices act on the traffic assignment, forcing the demand to change towards less expensive intersections and generating a system in dynamic equilibrium where the unused intersections become cheaper while the high demanded ones become very expensive. At the same time, the traffic control policy rewards the driver agents that value the reservations the most, who experience lower delays.

Table 4.10: Average travel time (min): $\mathcal{ECO}^{-}_{CA}$ vs. FCFS

Origin		Destination O_1	O_2	O_3	O_4	O_5	O_6	O_7
O_1	$\mathcal{ECO}^{-}_{CA}$	-	12.10	**13.76**	**24.00**	**27.27**	22.61	**13.82**
	FCFS	-	**11.98**	22.88	35.13	43.56	**21.35**	13.82
O_2	$\mathcal{ECO}^{-}_{CA}$	11.06	-	**10.90**	**19.01**	**23.97**	**25.87**	21.00
	FCFS	**10.14**	-	16.50	25.86	31.04	38.09	**19.50**
O_3	$\mathcal{ECO}^{-}_{CA}$	14.95	12.85	-	**9.18**	**13.53**	**19.03**	**27.90**
	FCFS	**13.34**	**9.75**	-	12.21	17.63	23.68	31.73
O_4	$\mathcal{ECO}^{-}_{CA}$	**19.73**	**18.29**	**10.01**	-	**7.13**	**13.11**	23.30
	FCFS	26.94	22.58	13.91	-	10.04	15.73	**22.74**
O_5	$\mathcal{ECO}^{-}_{CA}$	**25.04**	**20.74**	**12.07**	**7.47**	-	**10.05**	21.36
	FCFS	32.16	30.61	21.53	8.83	-	10.77	**17.65**
O_6	$\mathcal{ECO}^{-}_{CA}$	24.46	**27.24**	**16.33**	**16.39**	**10.35**	-	14.16
	FCFS	**22.51**	57.00	41.05	24.68	19.02	-	**13.73**
O_7	$\mathcal{ECO}^{-}_{CA}$	14.82	23.93	**20.09**	**27.53**	**16.92**	12.37	-
	FCFS	**14.30**	**23.25**	56.42	34.99	31.23	**11.99**	-

4.6 Discussion

In this chapter we extended the scenario evaluated in chapter 3 from a single intersection to a network of intersections. In this new setting, the complexity that a management policy must deal with increases considerably, since the problems that are raised by the underlying system are distributed, characterised by a great level of uncertainty, and intrinsically dynamic.

The first approach was evaluating how the policies inspired by the adversarial queueing theory, formulated in section 3.2, perform at the network level. Looking at the different AQT-based policies that have been tested, we noticed that although they show an improvement on the FCFS policy, they do not improve (from a practical point of view) the intersection's throughput so significantly. This was a hint that, if the scope of the problem moves from a single intersection to a network of intersections, we probably cannot expect great improvements even from more sophisticated policies. This is because in a network, an intersection manager is not uncoupled anymore, and it has influence on/it is influenced by the other intersection managers in the network. For instance, it is possible for several intersection managers to be very effective in increasing the throughput of their own intersections, causing serious problems for another intersection manager downstream, which must deal with a great traffic demand that is superior to its management capability.

For this reason, we focused on defining mechanisms that act on the traffic assignment, i.e., the distribution of the vehicles in the network. Traffic assignment mechanisms aim at making the task of the traffic controllers easier, by means of a better distribution of the traffic demand and an improved allocation of the network capacity. We tackled this problem from a market-based perspective, using markets as a way to design an efficient traffic assignment strategy. The advantages of such an approach are many. In fact, the market dynamic provides the driver agents with incentives to explore different alternatives for the route choice. Furthermore, the intersection managers, participating in and (eventually) ruling the market, have more power to influence the driver agents behaviour.

We studied two different computational economies. In the first one $\mathcal{ECO}^+$, we tackled the problem from the *profit maximisation* perspective. We applied multiagent reinforcement learning techniques to make the intersection managers dynamically coordinate their pricing policies with the aim of converging on the optimal joint policy. The experimental evaluation showed that the intersection managers, striving to maximise a profit function, indirectly influence the driver agent decision making and "unintentionally" minimise the average travel time. We compared two different agent reward functions, one with a global scope ($ExpDR$) and one with a local scope (LR). The LR performed worse than $ExpDR$ in maximising the global profit, whereas it considerably improved its performance in maximising the global marginal profit. The fact that a purely local reward behaves as well as a global one could appear surprising. Still, in a cooperative multiagent learning problem, the most important aspect of an agent reward function is not the scope (local or global) but the *quantity of information* it carries. One may think that a global signal is intrinsically more informative than a local signal, but it is not always the case, because global information carries a lot of noise as well. For example, the gross national product of a nation (global scope) is a signal that does not provide useful information to firms that want to know how well they performed in the market, while the previous year balance sheet (local scope) clearly carries much more information. A noisy agent reward function that does not rate the true contribution of an agent could make it act badly, since the signal it receives adds an uninformative bias. Differently to the profit P_i, the marginal profit MP_i is bounded between 0 and p_i, so that all marginal profit that an agent receives at the end of an episode is highly informative: if it is close to 0, the feedback is clearly negative, while if it is close to p_i, the feedback is clearly positive, because p_i is the highest reward the agent can receive.

Furthermore, a global feedback signal could incentivise "lazy" agent behaviours. In the experiments we noticed how with $ExpDR$ several agents do not provide any contribution to the global marginal profit, while with LR all the agents were motivated to contribute to the global marginal profit. This happened due to the nature of the two reward functions: with $ExpDR$, each intersection manager is rewarded with a team-wide information, so that although a particular intersection manager does not perform well, if globally the entire collective has a good performance, it is rewarded positively; using LR, the reward is strictly local, therefore each intersection manager strives to increase its own marginal profit, being in this way more proactive in respect to the collective task.

Another interesting issue that arose was the effect of giving the possibility to the driver agent to travel for free through the urban road network, if it stops at every intersection it encounters on its route. The experiments showed that the non-payer driver agent does not see its travel time dramatically increased if only a small portion of the population of driver agents behaves in the same way. This fact represents an interesting decision making problem for an individually rational agent. In fact, it is a sort of Minority Game [4]: not paying becomes interesting if *few agents* (the minority) decide to behave in the same way, while if many agents do the same, the agent utility is quickly reduced. The idea behind the minority game is the study of how many agents may reach a collective solution to a problem by adapting to expectations about the future of all the other agents. In such a problem, a complete steady state in the collective of agents is impossible, since agents keep changing and adapting their strategies in a quest for a non-existing equilibrium. Furthermore, the collective of agents is heterogeneous, since they have different ways to tackle available information about the game and convert it into expectations about future. Let's consider a hypothetical scenario where a set of driver agents must decide to travel either as a non-payer or as a payer. We take the perspective of the generic driver agent a. We can expect that a, like the majority of the driver agents, without any extra information has incentives for saving money and so behaving as a non-payer. The collective decision create congestion, due to mutual interactions, so that a observes high travel times. After some iterations, a may make an "exploratory" move and become a payer, so experiencing low travel times. Driver agent a keeps being a payer, and observing the behaviour of the other agents it may infer that the majority of the driver agents did the same and switched their behaviours to payer driver agent. But at this point, if the majority of the driver agents are payers, being again a non-payer becomes attractive, because now there are few non-payer agents. It is evident that a steady state or equilibrium in such a game is quite impossible to reach.

In the second computational economy, $\mathcal{ECO}^-$, we tackled the traffic assignment problem from the *adaptation* perspective. We modelled the intersection managers as competitors that strive to sell the reservations they supply, thus dynamically adapting their prices in response to the current demand. The experimental evaluation showed how this competitive economy generated a more balanced network, since the price fluctuations forced the demand to change towards less expensive intersections. The main advantage of $\mathcal{ECO}^-$ compared to $\mathcal{ECO}^+$ is that it overcomes the necessary assumptions and the limitations that all learning-based approaches suffer, such as the stationarity of the environment or the reduction of both action and state spaces to make the problem tractable. Although learning-based approaches can be able to cope with long-term effects of actions, they usually need many interactions with the environment to discover such long-term effects. On the other hand, adaptive and reactive strategies seem more suitable when the problem at hand is highly dynamic and uncertain.

For this reason, we chose $\mathcal{ECO}^-$ as the underlying traffic assignment mechanism for our integrated strategy, which we called $\mathcal{ECO}^-_{CA}$. This strategy combined an adaptive reserve-price update policy, which dealt with the traffic assignment, with the auction-based control policy that had been designed for the single intersection scenario, which acted on traffic control at intersection level. The experiments showed that the adaptive policy for traffic assignment made the auction-based traffic control policy perform better than FCFS for even high traffic demand, overcoming the main drawback of this policy, i.e., greater average travel times. This was due to the fact that the higher the network load, the more relevant the different reserve prices. In fact, the fluctuations of the reserve prices generated a system in dynamic equilibrium, where unused intersections became cheaper while the high demanded ones became very expensive. If the demand at particularly disputed intersections is lowered by the reserve price fluctuations, also the "social cost" of the auction-based control policy is lowered (at intersection level). Thus, a more homogeneous distribution of vehicles over the network leads to a better use of network resources, and thus to lower average travel times. The experiments showed that, in general, the gains obtained by a better distribution of demand outweigh the overhead introduced by the auction-based control policy with respect to social welfare (i.e., average travel time). In this way, the entire population of driver agents was rewarded with lower average travel times and, at the same time, the traffic control policy enforced an inverse relation between bid value and delay, rewarding the driver agents that valued the reservations the most with lower delays.

Chapter 5

Conclusions

I do not fear death.
I had been dead for billions
and billions of years before I was born
and I have not suffered
the slightest inconvenience from it.

Mark Twain

In this thesis we studied distributed mechanisms for the control and management of a (future) urban road network, where intelligent autonomous vehicles, controlled by driver agents, interact with the infrastructure in order to travel on the links of the network. In this last chapter we summarise and discuss the main contributions, and we propose some future lines of work.

5.1 Contributions

The first objective, pursued in chapter 3, was the analysis of the reservation-based intersection control system proposed by Dresner and Stone in [35]. The aim of this analysis was understanding the functioning of this control facility, and discovering potential niches to improve the intersection throughput. We analysed the performance of policies, grounded on the adversarial queueing theory (AQT), which can be used to assign reservations to driver agents. From the experimental results, we noticed that they show a statistically significant improvement on the original policy proposed by Dresner and Stone (first-come-first-served, FCFS). Nevertheless, this improvement was not so significant from a practical point of view. Although in theory FCFS could be very inefficient in some extreme cases, in practice such extreme cases are evidently rare. For this reason, we focused on studying a policy that might enforce different and new properties of the reservation-based intersection control system. We relied on the theory of combinatorial auctions (CA) [59] to model the mechanism that regulates the assignment of reservations to the driver agents. From the empirical experimentation, we discovered that the combinatorial auction-based policy guarantees lower delay to those driver agents that value their time the most, i.e., those that submit higher bids. On the other hand, this new policy showed that it paid a social cost, in term of greater average delays, specially when the traffic demand is high. This was somewhat expected, because the auction-based policy aims at granting a reservation to the driver agent that values it the most, rather than maximising the number of granted request.

The second objective of this thesis was to go beyond the single intersection setting, and extending the reservation-based model to a network of intersections. The new setting opened many new interesting questions, apart from considerably complicating the scenario. We realised that a traffic assignment strategy could make the task of a traffic control policy easier, by better distributing the traffic flow in the network. We studied two different traffic assignment strategies, relying on market-based methods as a solution method to solve the problem. We developed two different economic models. The first one, $\mathcal{ECO}^+$, was a cooperative economy that tackled the problem from the *profit maximisation* perspective. The intersection managers have been modelled as cooperative learning agents that collaboratively learn which price vector optimises a global profit function. Modelling the profit functions in the proper way, we discovered that the intersection managers, maximising their profit, indirectly optimise also the average travel time of the population of driver agents.

The second economic model, $\mathcal{ECO}^-$, was a competitive economy that tackled the problem from the *adaptation* perspective. In this economy, the intersection behaved selfishly, competing with all the others for the supply of the reservations at the intersections. In this case, our objective as system designers was to reach the market equilibrium, that is, a situation where the amount of resources sought by buyers is equal to the amount of resources provided by suppliers. The experimental evaluation showed that in this way the available resources are efficiently allocated to the driver agents, generating a better balanced network, since the price fluctuations forced the demand to change towards less expensive intersections. Furthermore, $\mathcal{ECO}^-$ overcame some limitations that all learning-based approaches suffer, including $\mathcal{ECO}^+$, such as the stationarity of the environment or the need of reducing action and state spaces to make the problem tractable.

For this reason we chose the competitive model as traffic assignment mechanism to be combined with the auction-based policy for traffic control, in order to develop an adaptive, integrated, strategy for full-fledged traffic management, which we called $\mathcal{ECO}^-_{CA}$. The demand-response pricing policy acts on the traffic assignment, adapting the reserve price (i.e., the minimum price at which the intersection manager is willing to sell) and generating a system in dynamic equilibrium, where unused intersections become cheaper while highly demanded ones become very expensive. If the demand at particularly disputed intersections is lowered by the reserve price fluctuations, also the social cost of the auction-based control policy is lowered (at intersection level). Thus, a more homogeneous distribution of vehicles over the network leads to a better use of network resources, and thus to lower average travel times. In this way, the entire population of driver agents is rewarded by lower average travel times and, at the same time, the traffic control policy enforces an inverse relation between bid value and delay, rewarding the driver agents that value the reservations the most with lower delays.

In order to evaluate all the proposed models, we developed a simulation tool, called $\mathcal{M.I.T.E.}$. We analysed the *pros* and *cons* of the different traffic flow models that we found in literature, and we opted to implement a hybrid mesoscopic-microscopic simulator. The simulation tool integrates the mesoscopic model by Thomas Schwerdtfeger [89] to model the traffic flow along the road links, and the microscopic model by Kai Nagel and Michael Schreckenberg [68] to model the traffic flow inside the intersections. We adopted this solution because the mesoscopic model allowed us to simulate large-scale systems, with thousands of vehicles moving within the road network, while the microscopic model permitted the implementation of fine-grained control policies inside the intersections.

5.2 Future work

This doctoral thesis is not exempt from limitations, such as the assumptions we relied on, or the type of experimental analysis that we did. From one side, it is true that our model, to be applicable, needs a very advanced, agent-based, urban road traffic infrastructure, populated by autonomous vehicles that interact with the infrastructure. Although this is not the reality we live in, we claim that it is not so far-fetched as one may think. Many people worldwide are working to make a scenario of autonomous vehicles integrated with the infrastructure a reality, such as those involved in the DARPA Urban Challenge and the Vehicle Infrastructure Integration. Furthermore, there is an increasing interest in developing control and management systems that, taking advantage of the "agentification" of vehicles and infrastructure, act on *individual* vehicles rather than on *flows*, such as the work by Dresner and Stone in [35] or the work by Schepperle and Bohm [88].

Furthermore, we remark that as researchers on distributed artificial intelligence (and not traffic engineers), we put forward a *user-centric analysis* of the proposed mechanisms, having in mind as final goal the quality of service perceived by the drivers (travel times, delays, spent money, etc.), and not focusing on other issues such as sizing of transportation facilities or saturation flows analysis of intersections.

Starting from the work developed in this doctoral thesis, we identified possible future lines of research that we can follow.

Economic models. In this work, two economic models have been implemented and evaluated, that is, trading and one-to-many (combinatorial) auctions. Nevertheless, other economic models are viable and implementable in our scenario. For example, the market could be regulated by a continuous double auctions, where many sellers (i.e., the intersection managers), place their sell-bids, and many buyers (i.e., the driver agents) submit their buy-bids, and the market continuously clears when a match between sell-bids and buy-bids is found. Also, bargaining could be easily implemented in our scenario, with driver agents and intersection managers negotiating and agreeing on a price that is acceptable to both parties.

Driver models. This thesis did not focus on developing a sophisticated driver agent model, thus we implemented a rather simple driver agent model. The only decision it had to make was about the route choice. The route choice is modelled as a utility maximisation problem: the agent evaluates the possible alternatives in its choice set and selects the one that maximises its utility measure.

In order to be more realistic and capture the inherent complexity of urban traffic systems, it is important to extend and enrich the driver behavioural model. For example, the driver agent could be implemented as a two layer decision maker as in [83], where a reactive, rule-based, layer make short-term decisions about car-following and lane-changing, and a cognitive, BDI-style, layer is in charge of making the more complex decisions such as route choice and departure time selection. Furthermore, another characteristic of human drivers that need to be implemented is learning from experience. Human drivers implicitly use historical information and past experiences to update the likelihood of selecting a specific route at a certain time. For example in [57], the driver agents make probabilistic decisions about route choice, and periodically update the probability of selecting a specific route according to the rewards (i.e., the inverse of the travel time) it has obtained so far selecting that route.

Vehicle-to-vehicle communication. In all the scenarios that have been evaluated in this thesis, only interactions between the vehicles and the infrastructure take place. Thus, no collaboration at all is possible between vehicles. Nevertheless, vehicle-to-vehicle communication is receiving great attention from the scientific and engineering community [117]. In the context of this work, vehicle-to-vehicle communication could be used to enrich the action space of a driver agent. For example, consider this situation: a wealthy driver agent is in a hurry but it is travelling behind a vehicle that does not want to allocate a lot of money to acquire a reservation. In this case, the willingness of the wealthy driver agent to bid a lot of money is not effective, because it will be impossible to actually make use of the reservation that it can potentially gain in the auction. If vehicle-to-vehicle communication is enabled, the wealthy driver agent could help the driver agent in front of it to get a reservation, subsidising it with a quantity of money that could be enough to get a reservation, as in [88], or forming a coalition with it to submit a joint bid. Research on platoon formation could also be relevant to this respect [46]. In fact, synchronising vehicle movements by communicating speed variations and lane changing could dramatically increase roads capacity and reduce the size and impact of traffic congestion[1].

[1]http://www.path.berkeley.edu/PATH/Research/demos

Appendix A

$\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ UML diagrams

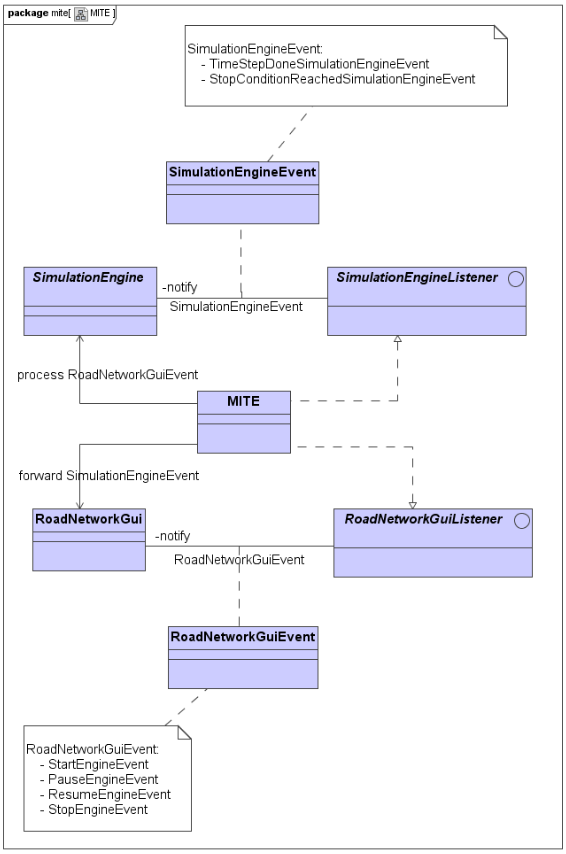


Figure A.1: *MITE* class diagram

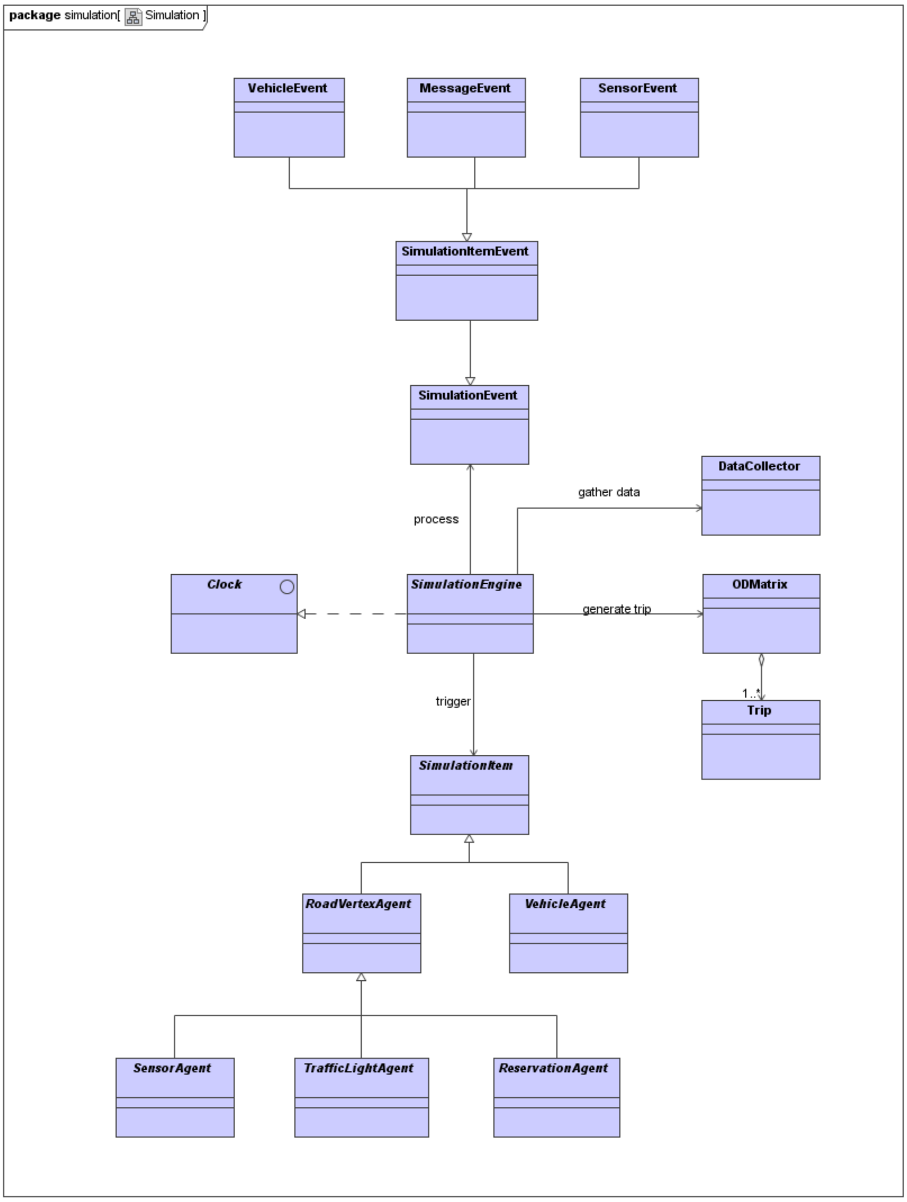


Figure A.2: *SimulationEngine* class diagram

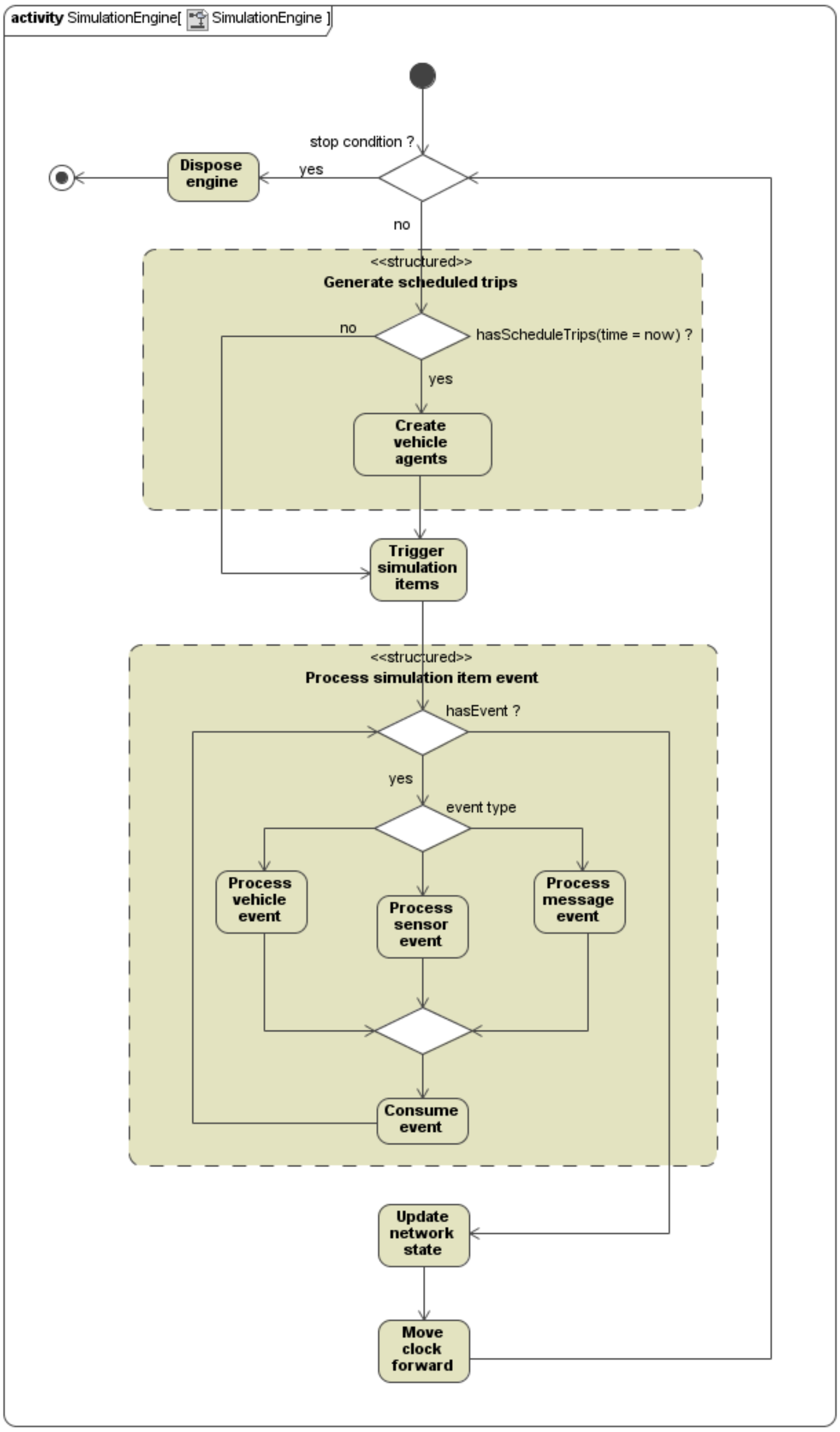


Figure A.3: *SimulationEngine* activity diagram

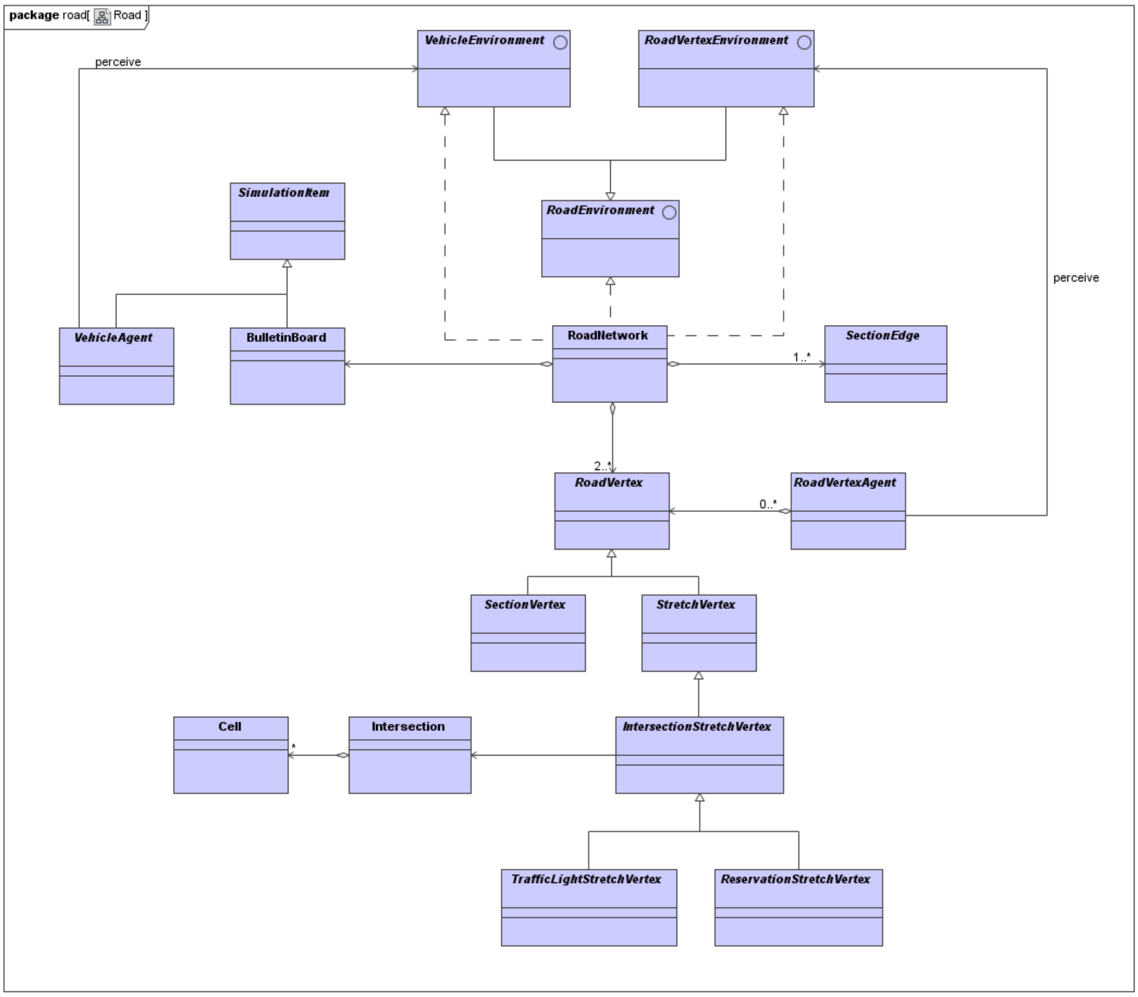
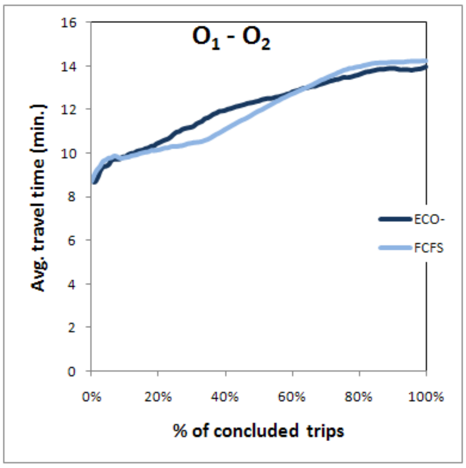


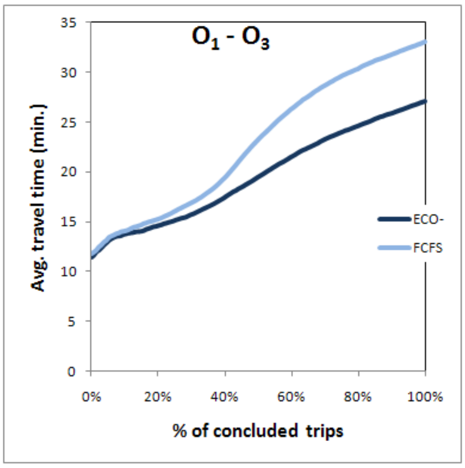
Figure A.4: *RoadNetwork* class diagram

Appendix B

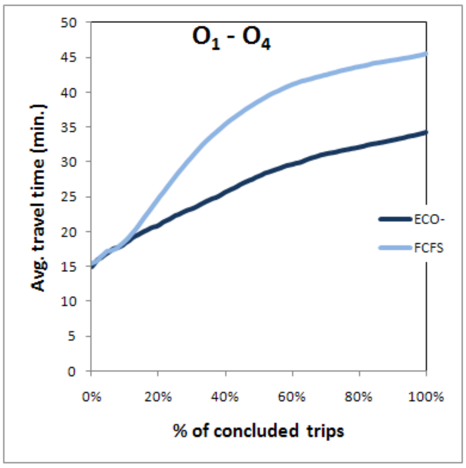
$\mathcal{ECO}^-$: vehicle metrics



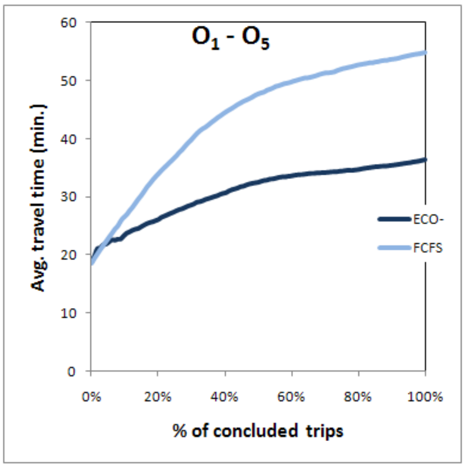
(1) Moving average travel time O_1-O_2



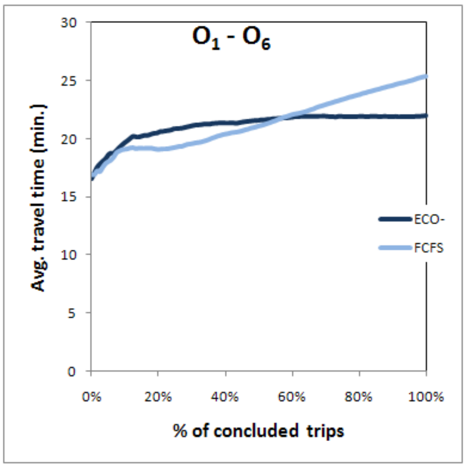
(2) Moving average travel time O_1-O_3



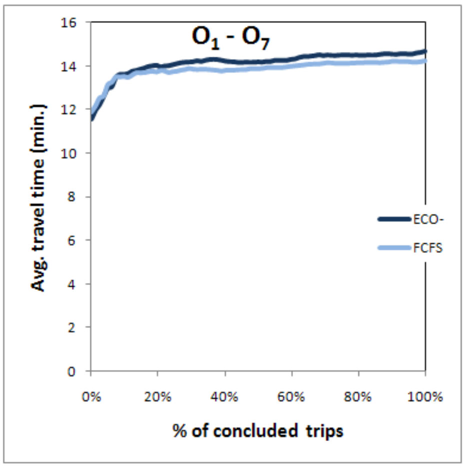
(3) Moving average travel time O_1-O_4



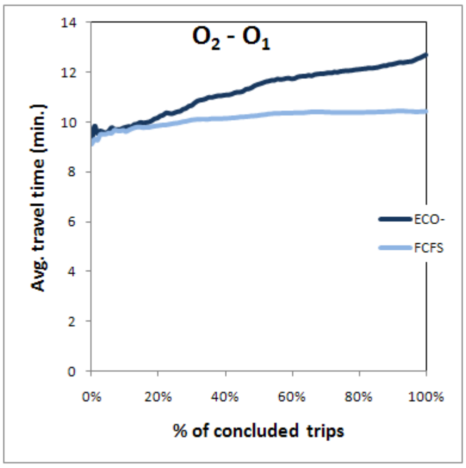
(4) Moving average travel time O_1-O_5



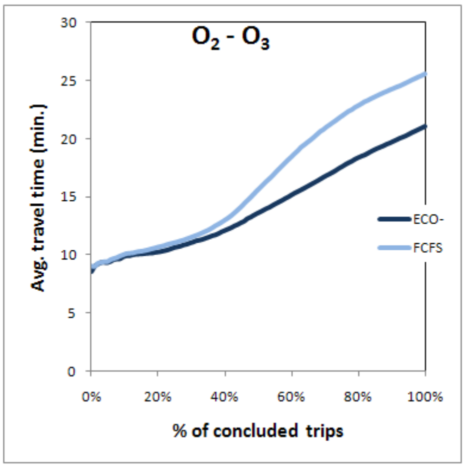
(5) Moving average travel time O_1-O_6



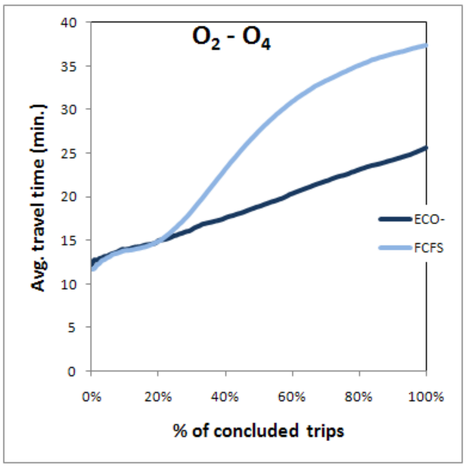
(6) Moving average travel time O_1-O_7



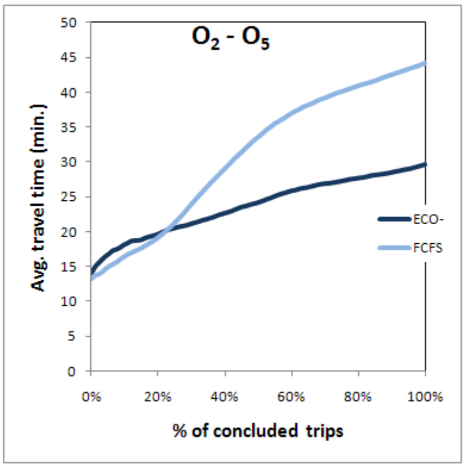
(7) Moving average travel time O_2-O_1



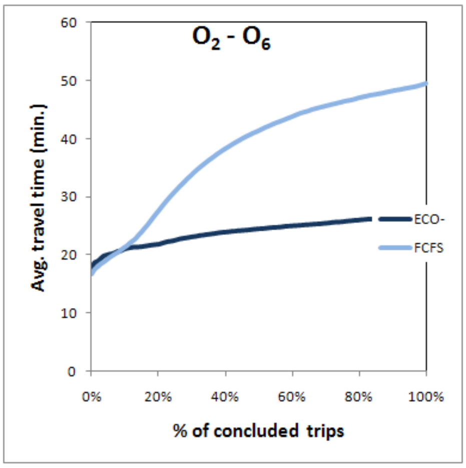
(8) Moving average travel time O_2-O_3



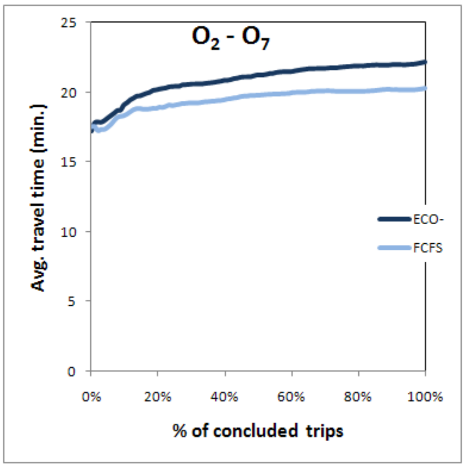
(9) Moving average travel time O_2-O_4



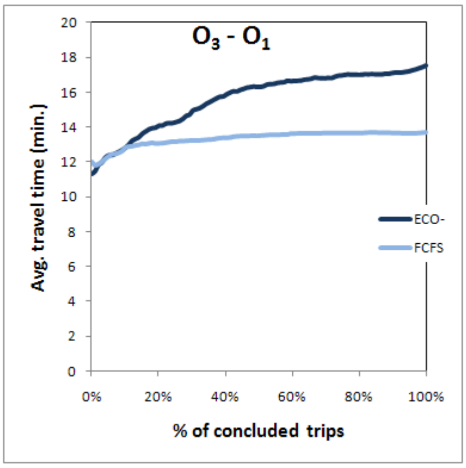
(10) Moving average travel time O_2-O_5



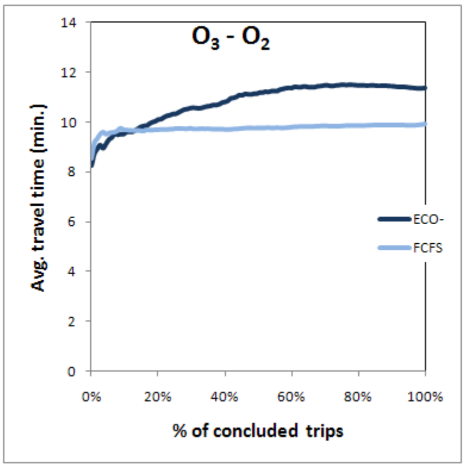
(11) Moving average travel time O_2-O_6



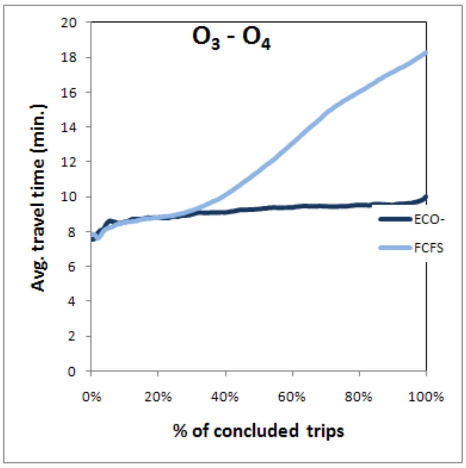
(12) Moving average travel time O_2-O_7



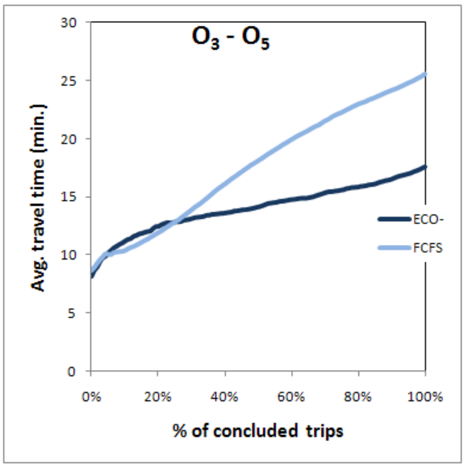
(13) Moving average travel time O_3-O_1



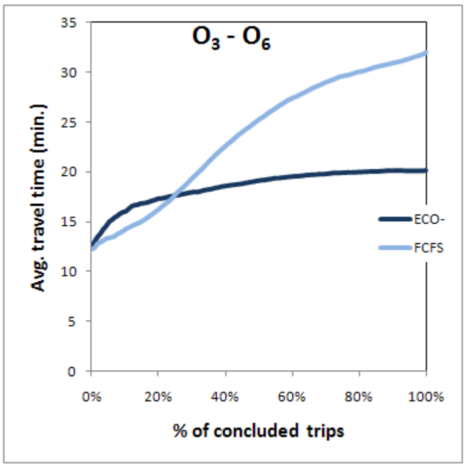
(14) Moving average travel time O_3-O_2



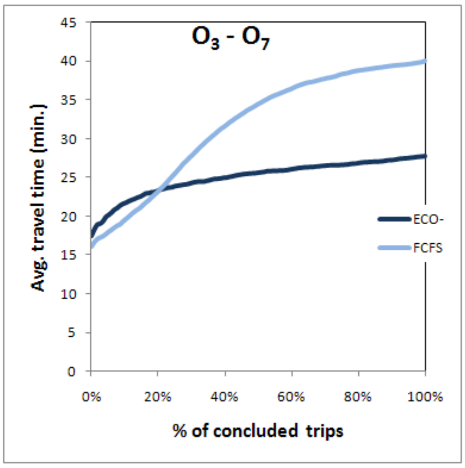
(15) Moving average travel time O_3-O_4



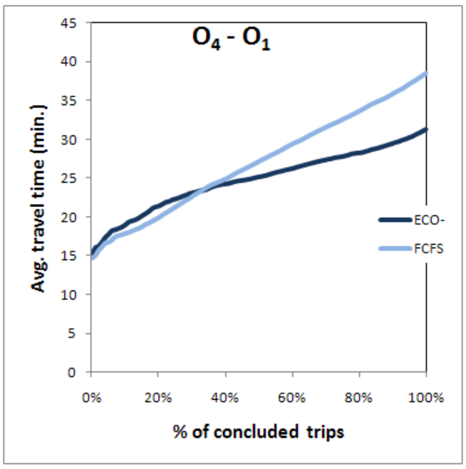
(16) Moving average travel time O_3-O_5



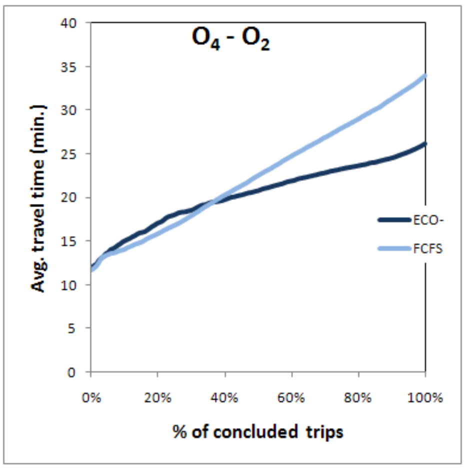
(17) Moving average travel time O_3-O_6



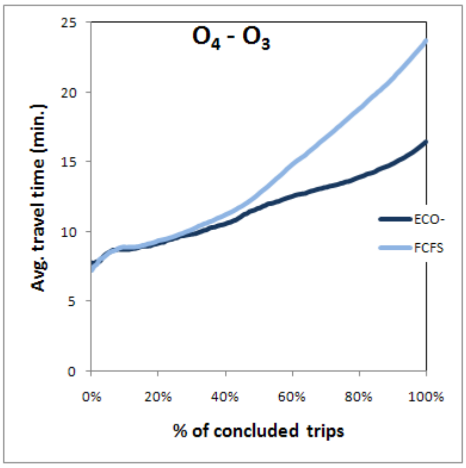
(18) Moving average travel time O_3-O_7



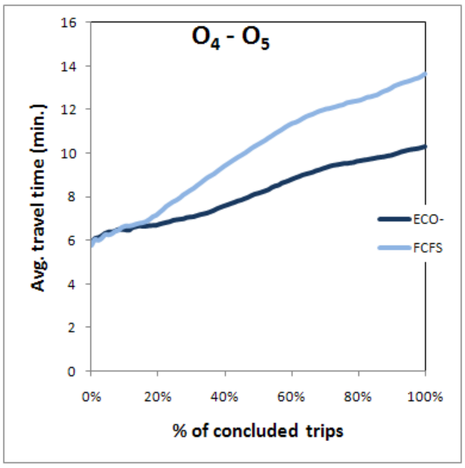
(19) Moving average travel time O_4-O_1



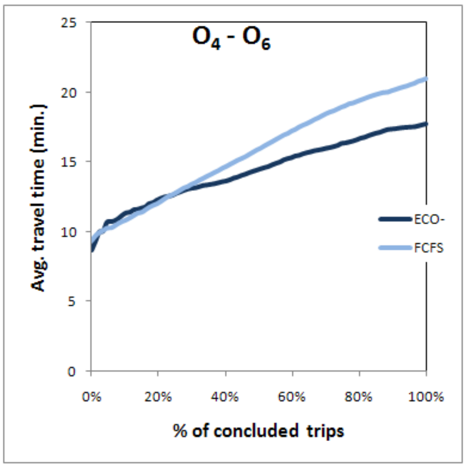
(20) Moving average travel time O_4-O_2



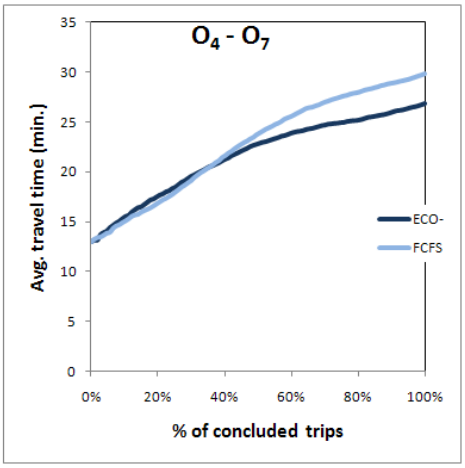
(21) Moving average travel time O_4-O_3



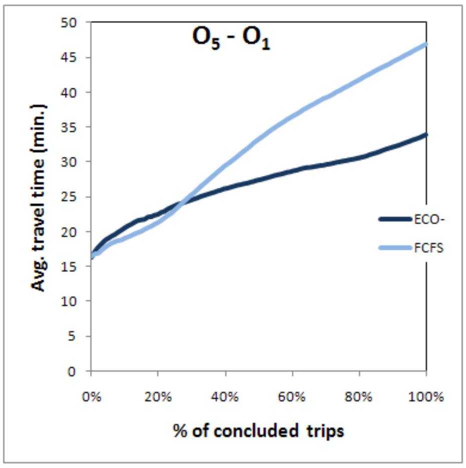
(22) Moving average travel time O_4-O_5



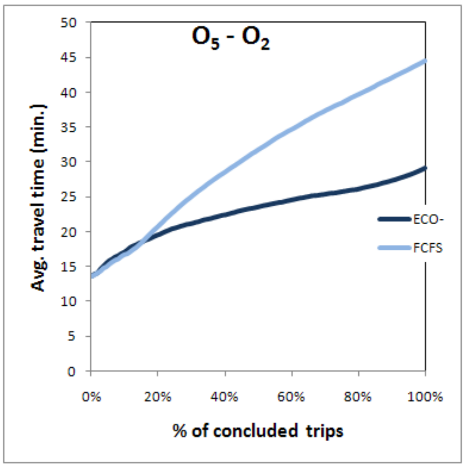
(23) Moving average travel time O_4-O_6



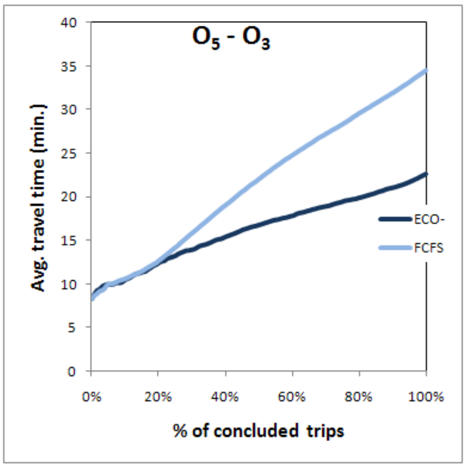
(24) Moving average travel time O_4-O_7



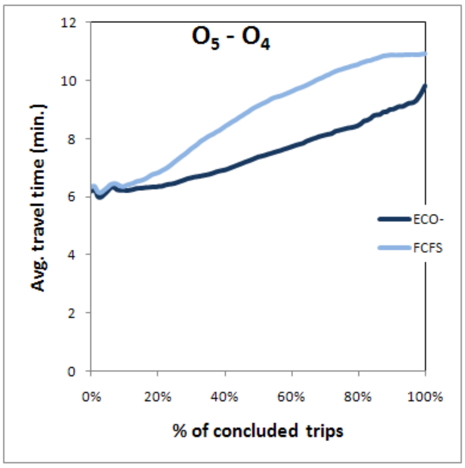
(25) Moving average travel time O_5-O_1



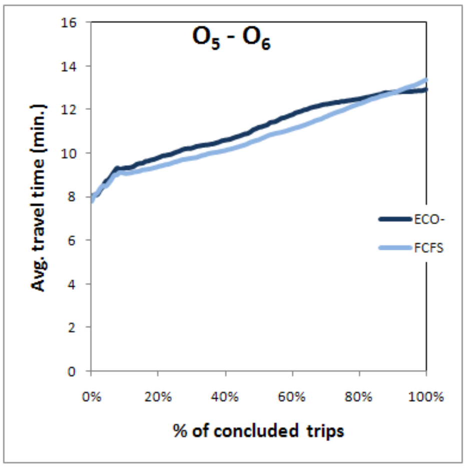
(26) Moving average travel time O_5-O_2



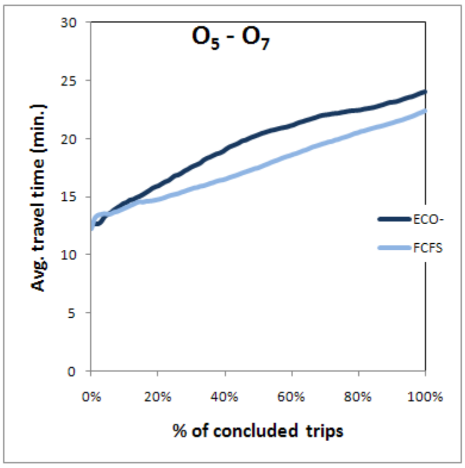
(27) Moving average travel time O_5-O_3



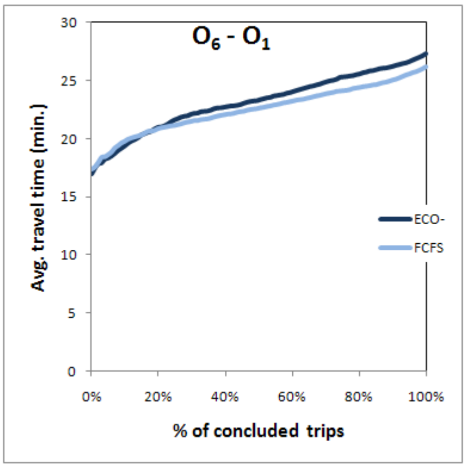
(28) Moving average travel time O_5-O_4



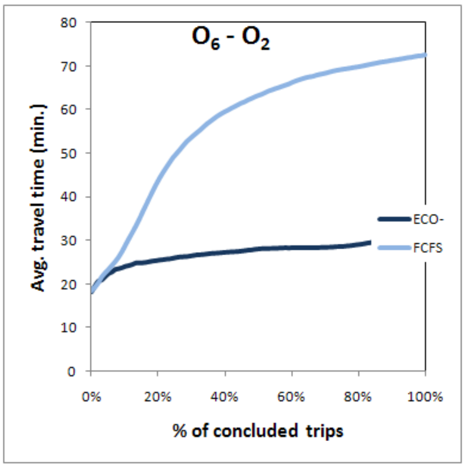
(29) Moving average travel time O_5-O_6



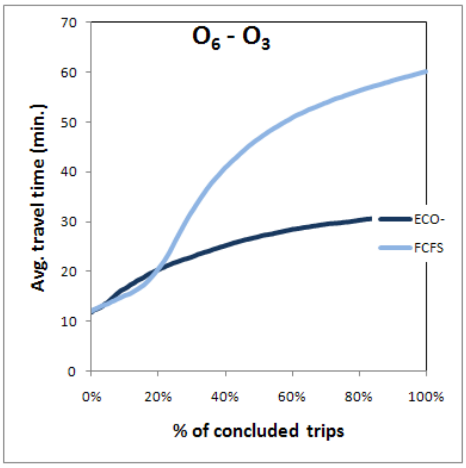
(30) Moving average travel time O_5-O_7



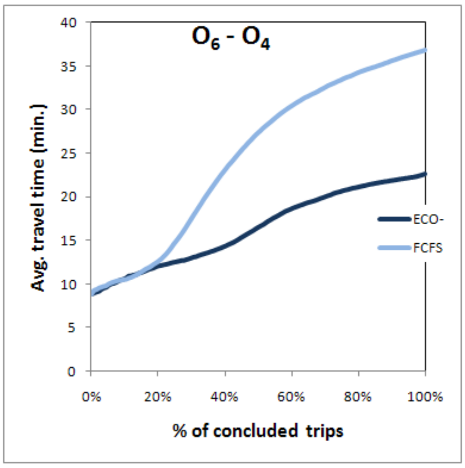
(31) Moving average travel time O_6-O_1



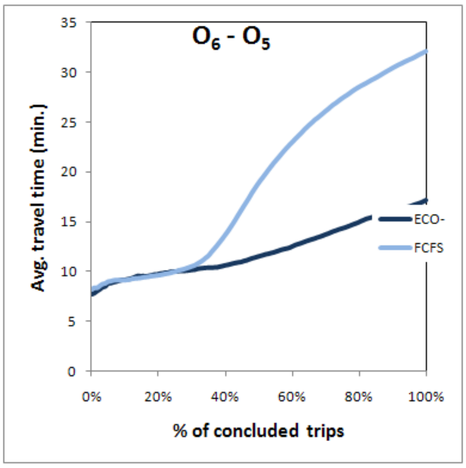
(32) Moving average travel time O_6-O_2



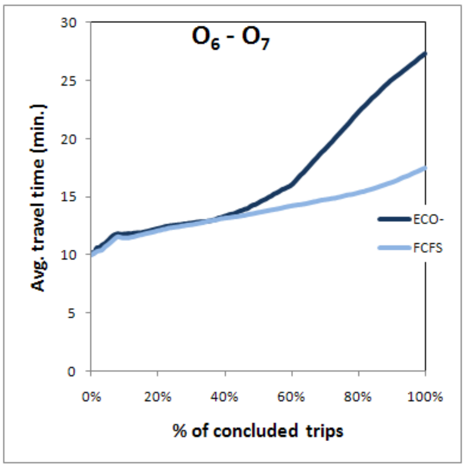
(33) Moving average travel time O_6-O_3



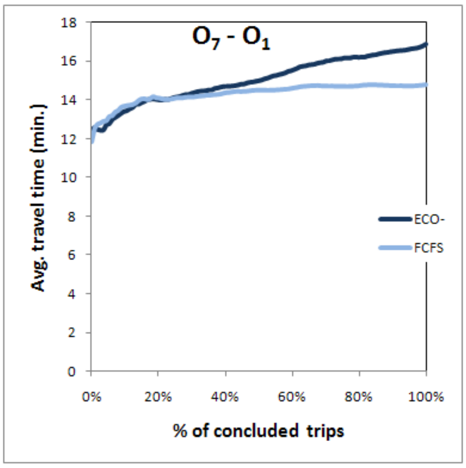
(34) Moving average travel time O_6-O_4



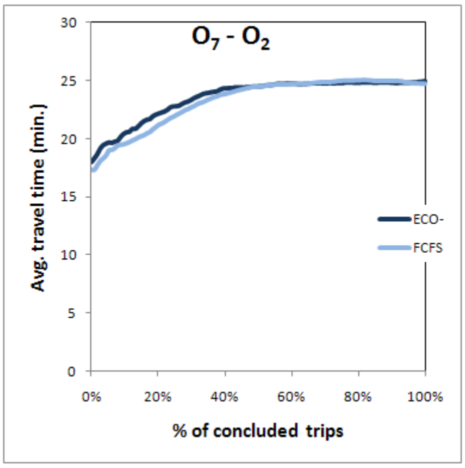
(35) Moving average travel time O_6-O_5



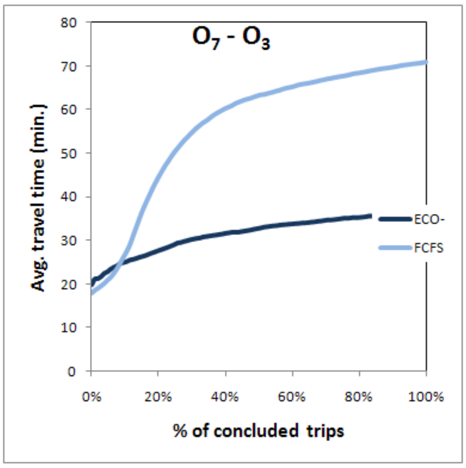
(36) Moving average travel time O_6-O_7



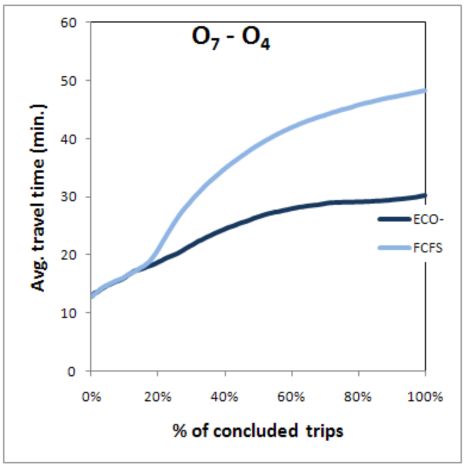
(37) Moving average travel time O_7-O_1



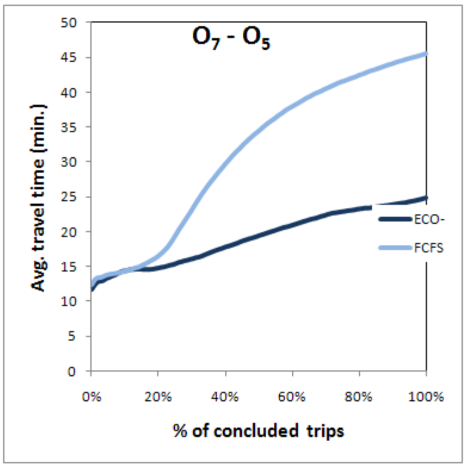
(38) Moving average travel time O_7-O_2



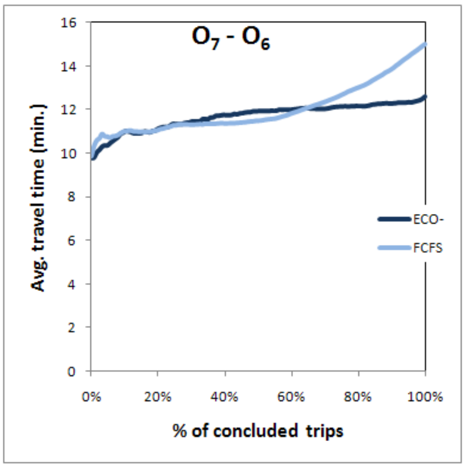
(39) Moving average travel time O_7-O_3



(40) Moving average travel time O_7-O_4



(41) Moving average travel time O_7-O_5

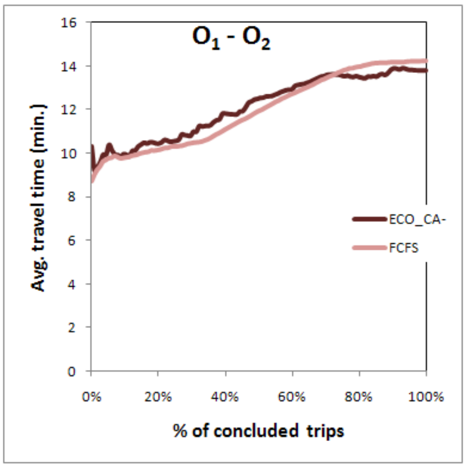


(42) Moving average travel time O_7-O_6

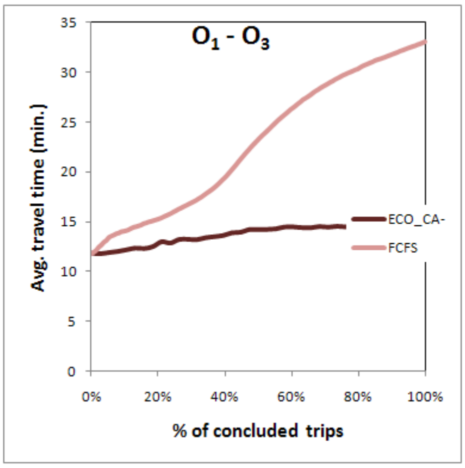
Figure B.1: Moving average travel time, grouped by origin-destination, for $\mathcal{ECO}^-$

Appendix C

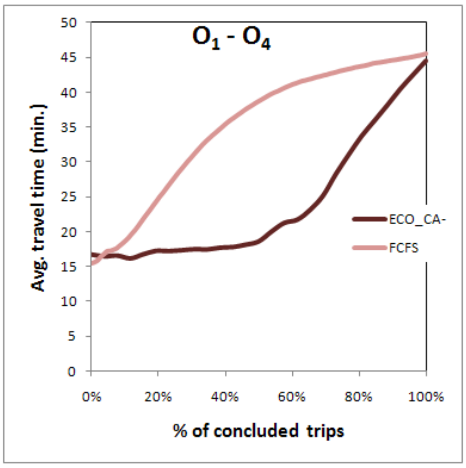
$\mathcal{ECO}^{-}_{CA}$: vehicle metrics



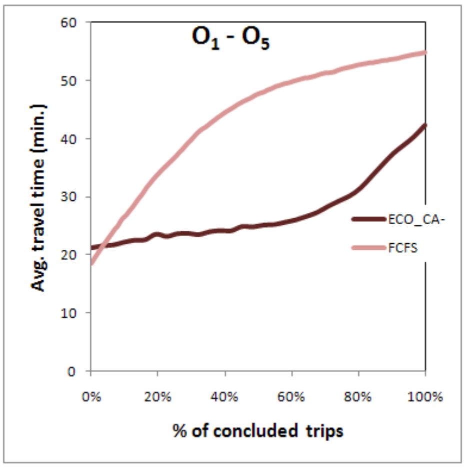
(1) Moving average travel time O_1-O_2



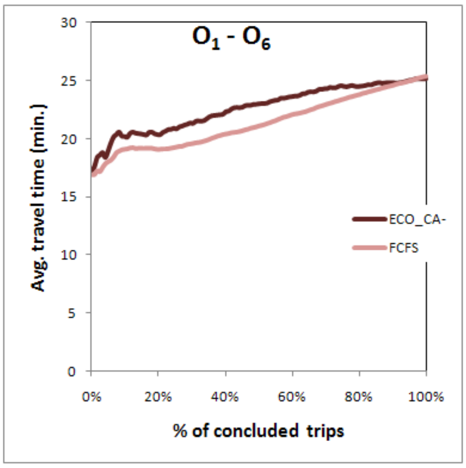
(2) Moving average travel time O_1-O_3



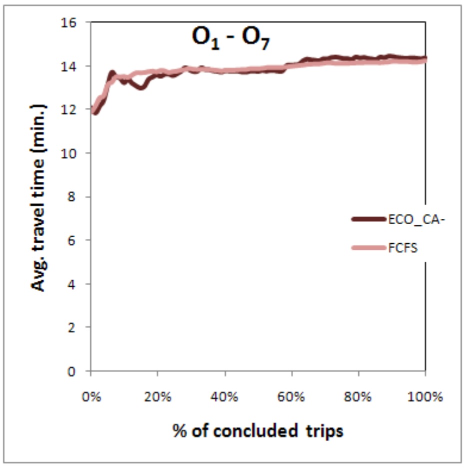
(3) Moving average travel time O_1-O_4



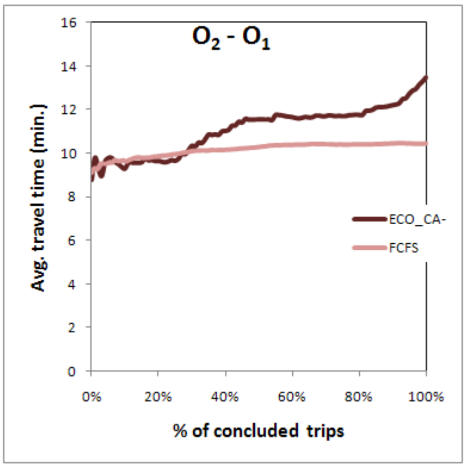
(4) Moving average travel time O_1-O_5



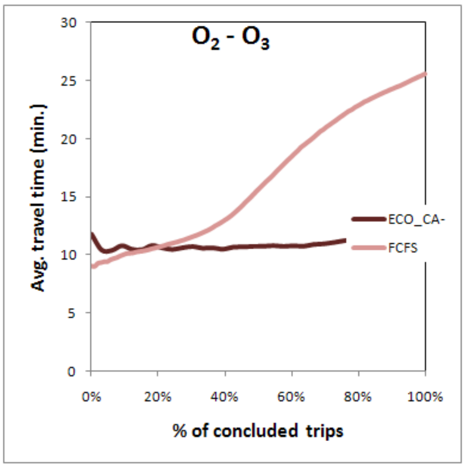
(5) Moving average travel time O_1-O_6



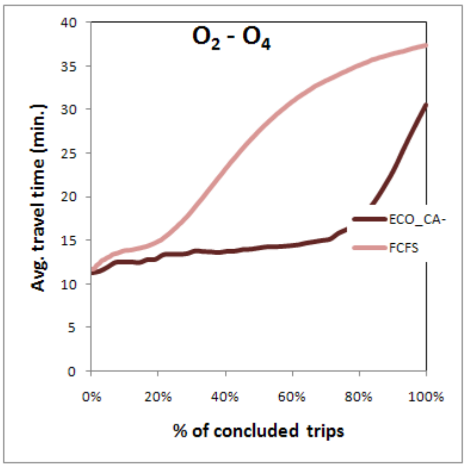
(6) Moving average travel time O_1-O_7



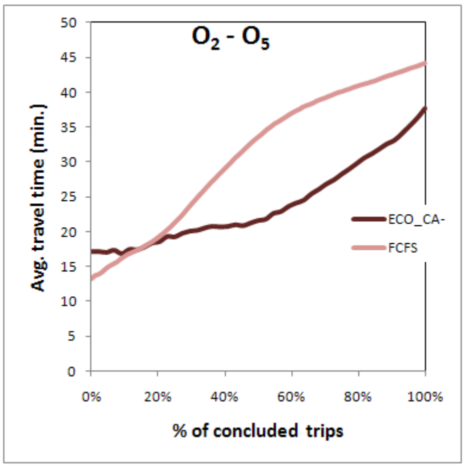
(7) Moving average travel time O_2-O_1



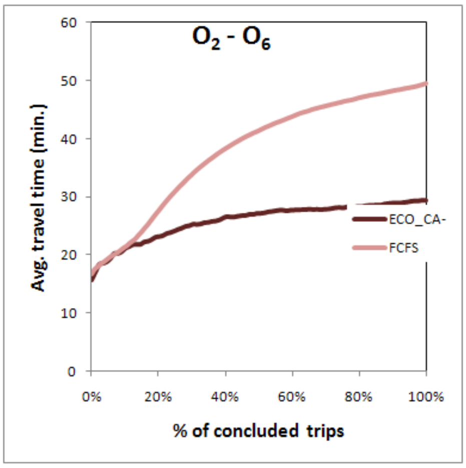
(8) Moving average travel time O_2-O_3



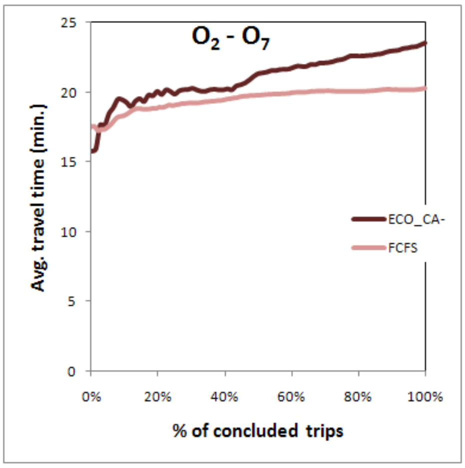
(9) Moving average travel time O_2-O_4



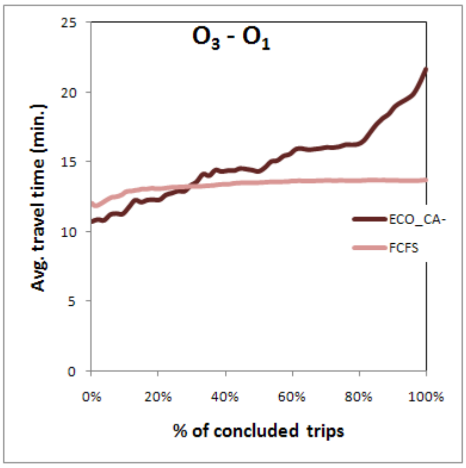
(10) Moving average travel time O_2-O_5



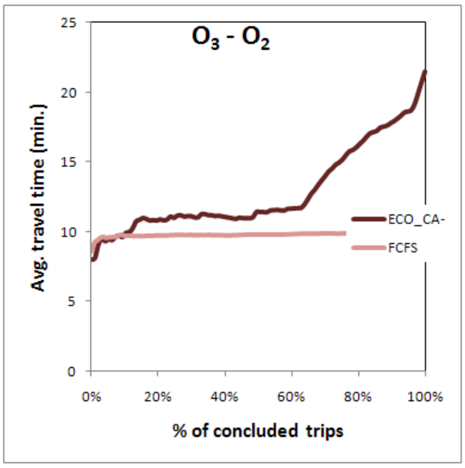
(11) Moving average travel time O_2-O_6



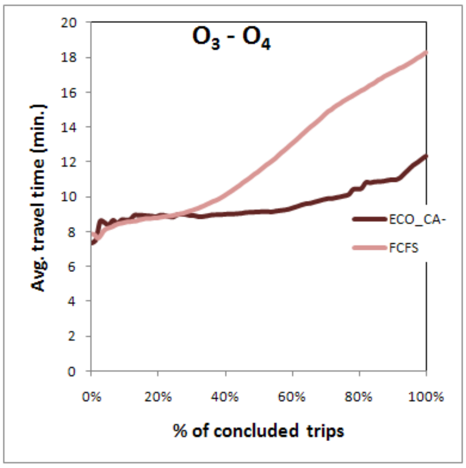
(12) Moving average travel time O_2-O_7



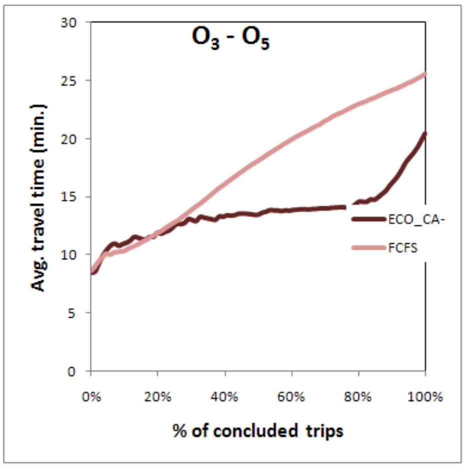
(13) Moving average travel time O_3-O_1



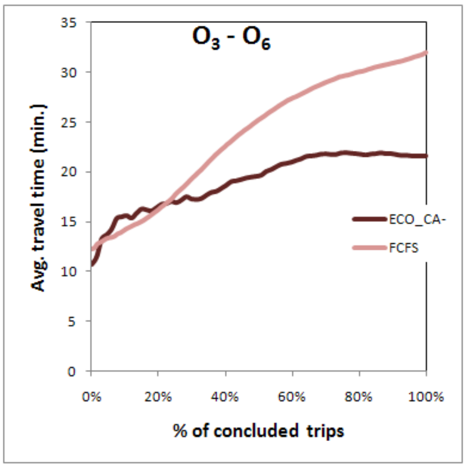
(14) Moving average travel time O_3-O_2



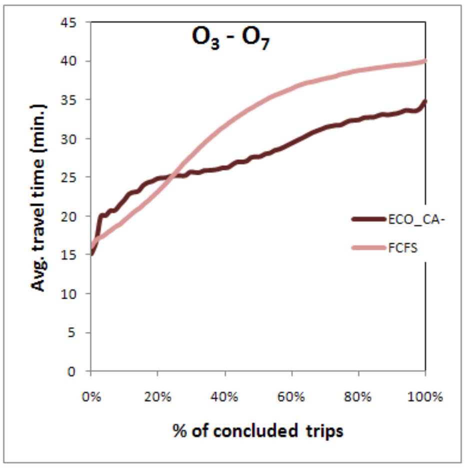
(15) Moving average travel time O_3-O_4



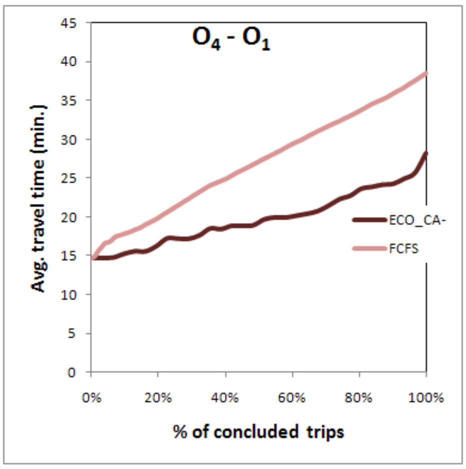
(16) Moving average travel time O_3-O_5



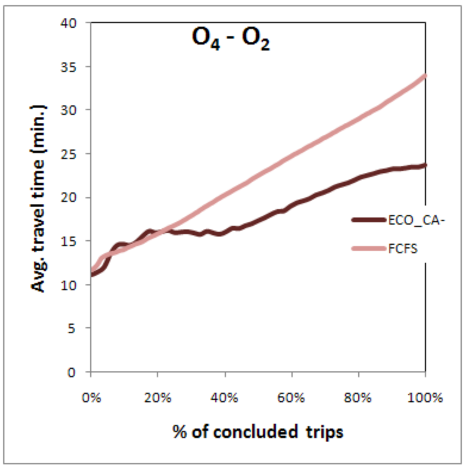
(17) Moving average travel time O_3-O_6



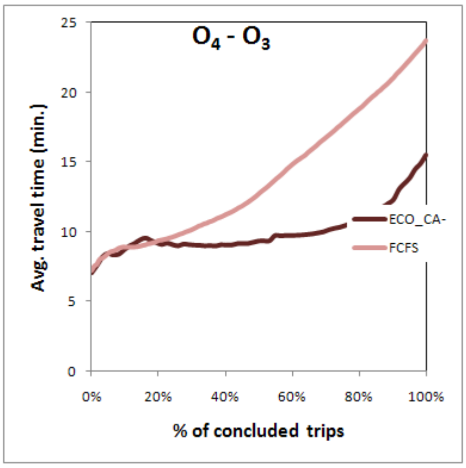
(18) Moving average travel time O_3-O_7



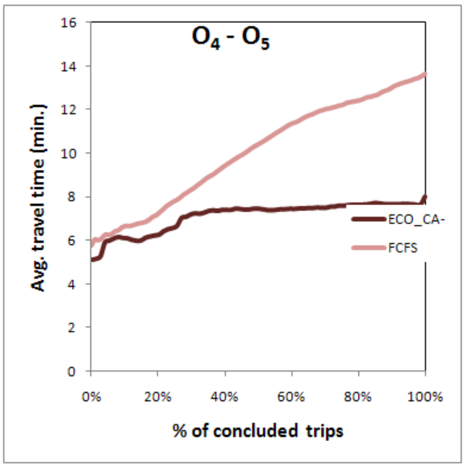
(19) Moving average travel time O_4-O_1



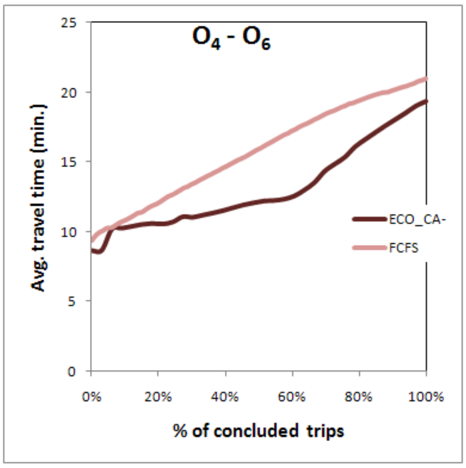
(20) Moving average travel time O_4-O_2



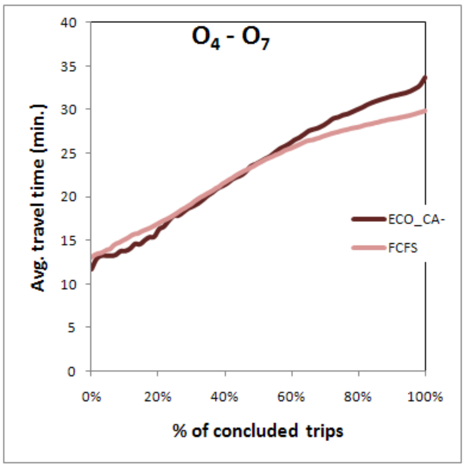
(21) Moving average travel time O_4-O_3



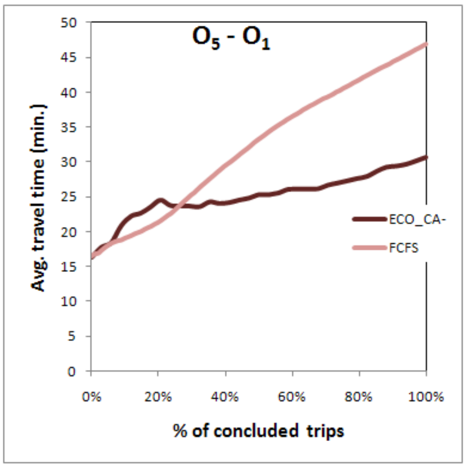
(22) Moving average travel time O_4-O_5



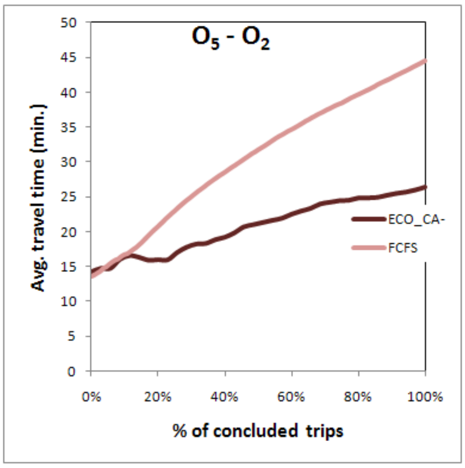
(23) Moving average travel time O_4-O_6



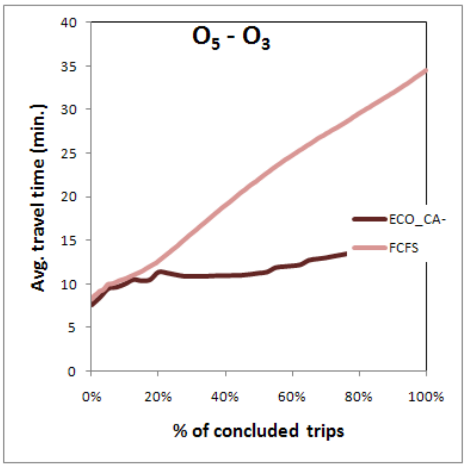
(24) Moving average travel time O_4-O_7



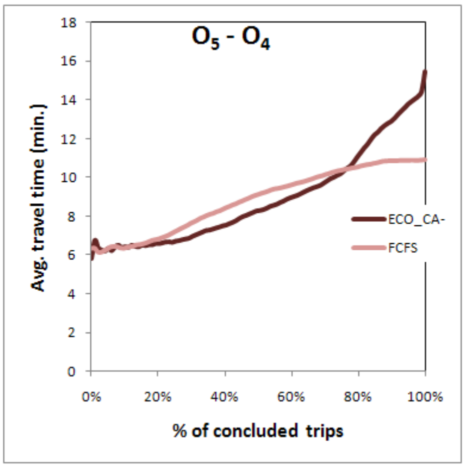
(25) Moving average travel time O_5-O_1



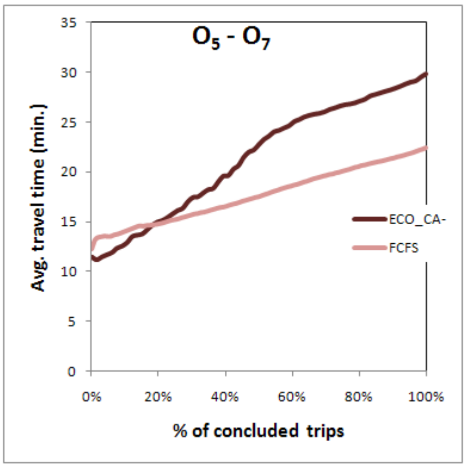
(26) Moving average travel time O_5-O_2



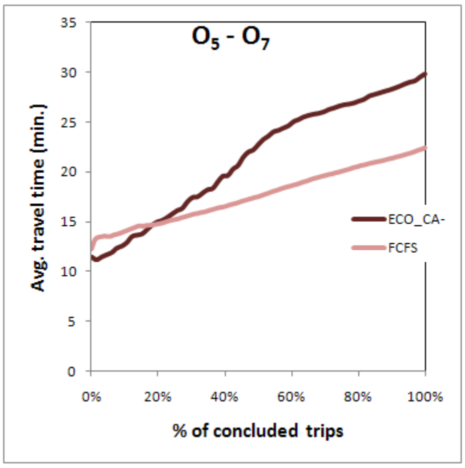
(27) Moving average travel time O_5-O_3



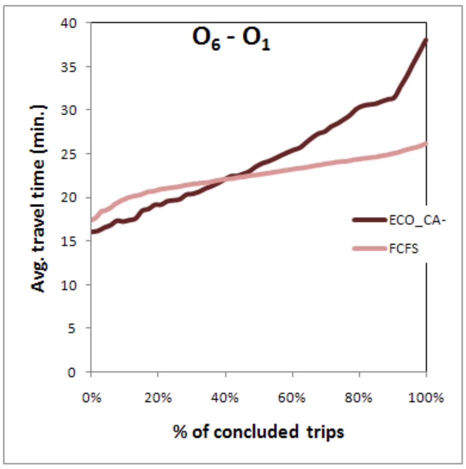
(28) Moving average travel time O_5-O_4



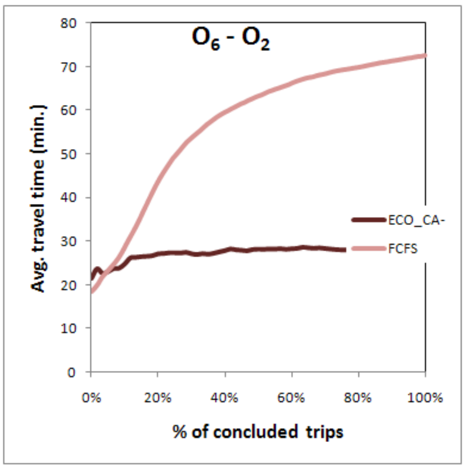
(29) Moving average travel time O_5-O_6



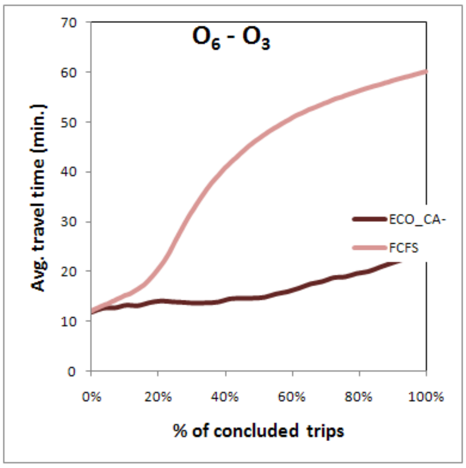
(30) Moving average travel time O_5-O_7



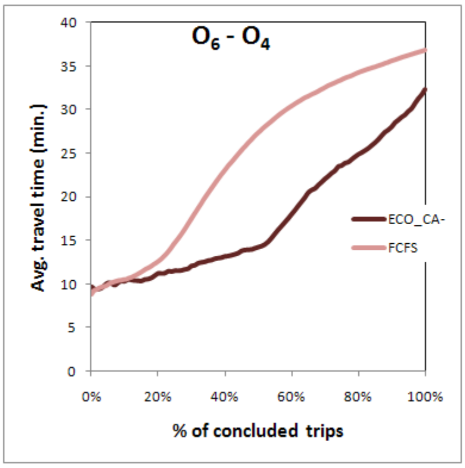
(31) Moving average travel time O_6-O_1



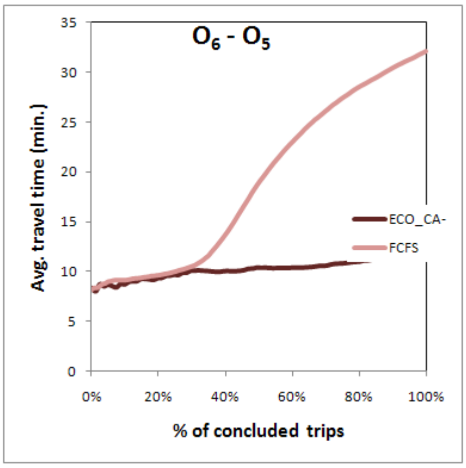
(32) Moving average travel time O_6-O_2



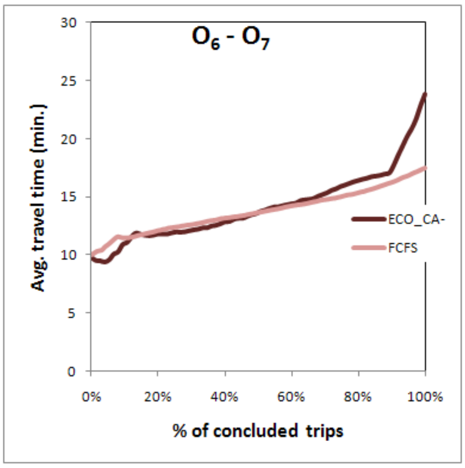
(33) Moving average travel time O_6-O_3



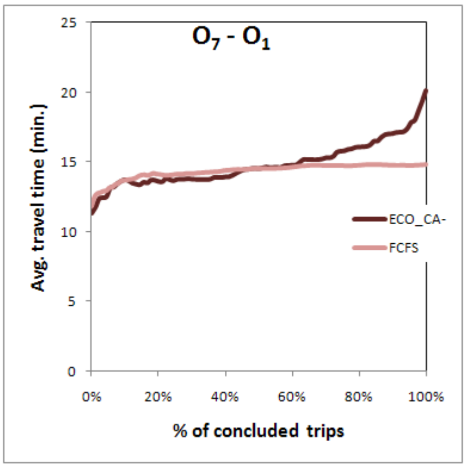
(34) Moving average travel time O_6-O_4



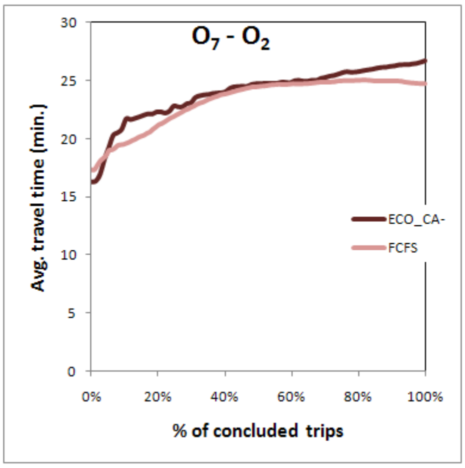
(35) Moving average travel time O_6-O_5



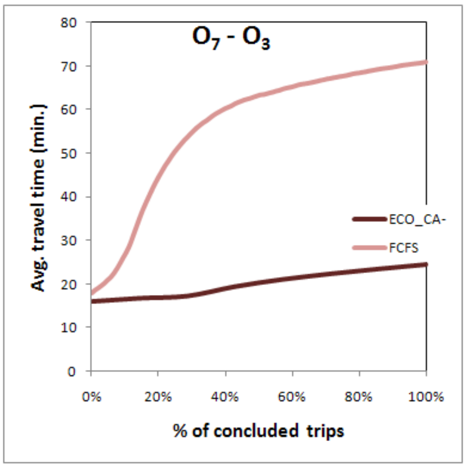
(36) Moving average travel time O_6-O_7



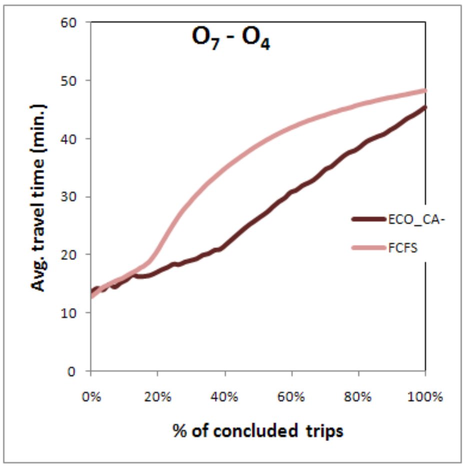
(37) Moving average travel time O_7-O_1



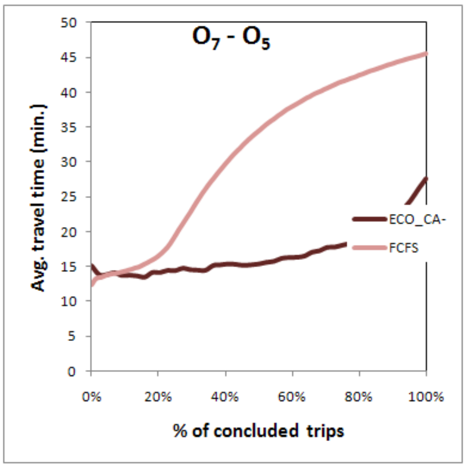
(38) Moving average travel time O_7-O_2



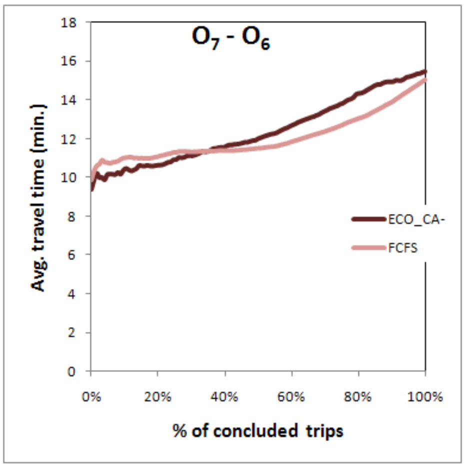
(39) Moving average travel time O_7-O_3



(40) Moving average travel time O_7-O_4



(41) Moving average travel time O_7-O_5



(42) Moving average travel time O_7-O_6

Figure C.1: Moving average travel time, grouped by origin-destination, for $\mathcal{ECO}_{CA}^{-}$

Appendix D

Publications derived from the thesis

If knowledge can create problems
it is not through ignorance
that we can solve them.

Isaac Asimov

- M. Vasirani and S. Ossowski, *Decentralized Coordination Strategies for the Vehicle Routing Problem*, Proceedings of the 23rd Annual ACM Symposium on Applied Computing (SAC'08), pages 130-131, ISBN 9781595937537, (CS Ranking: 0.85, 83rd of 169; CORE: B), 2008.

- M. Vasirani and S. Ossowski, *Towards reservation-based intersection coordination: an economic approach*, Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems, pages 1-6, ISBN 9781424421114, 2008.

- M. Vasirani and S. Ossowski, *Collective-based Multiagent Coordination: A Case Study*, Engineering Societies in the Agents World VIII, Lecture Notes in Artificial Intelligence, volume 4995, pages 240-253, ISBN 9783540876533, 2008.

- M. Vasirani and S. Ossowski, *Exploring the potential of multiagent learning for autonomous intersection control*, Multi-Agent Systems for Traffic and Transportation Engineering, pages 280-290, IGI Global, ISBN 9781605662268, 2009.

- M. Vasirani and S. Ossowski, *Market-based coordination for intersection control*, Proceedings of the 24th Annual ACM Symposium on Applied Computing (SAC'09), pages 747-751, ISBN 9781605581668, (CS Ranking: 0.85, 83rd of 169; CORE: B), 2009.

- M. Vasirani and S. Ossowski, *A market-inspired approach to reservation-based urban road traffic management*, Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'09), pages 617-624, ISBN 9780981738161, (CS Ranking: 0.76, 21st of 67; CORE: A+), 2009.

- M. Vasirani and S. Ossowski, *Evaluating Policies for Reservation-Based Intersection Control*, Proceedings of the 14th Portuguese Conference on Artificial Intelligence (EPIA'09), pages 39-50, ISBN 9789729689543, (CORE: B), 2009.

Appendix E

Resumen en castellano

Talk is cheap, show me the code.

Linus Torvalds

Introducción

"Juan se ha levantado muy temprano esta mañana. Hoy es su primer día de trabajo y no quiere llegar tarde. Un café rápido y ya está en el coche. Tan pronto como abre la puerta, la pantalla central se enciende y su driver agent arranca. Equipado con software de reconocimiento de voz, el agente está listo para recibir comandos acerca del sitio de destino. "Calle Ferrocarril 6", dice Juan a su driver agent. Basándose en el perfil de Juan, el driver agent selecciona una ruta y la muestra en la pantalla. Se tardará unos 45 minutos para llegar a destinación, por un precio muy barato, sólo 10 céntimos para cruzar un cruce que conecta los barrios periféricos con el centro de la ciudad. Normalmente Juan no tiene prisa, pero esta vez es diferente, quiere causar una buena impresión con su nuevo jefe. Usando la pantalla táctil, Juan cambia su perfil para el día de hoy y selecciona la modalidad <BusinessMode>, incrementando de 3 euros la cantidad de dinero que está dispuesto a pagar por su desplazamiento. El driver agent encuentra una nueva ruta mucho más rápida, que tarda

Figure E.1: Chevy Boss, el ganador del DARPA Urban Challenge 2007

sólo 25 minutos y cuesta 2.5 euros, visto que transita por un cruce muy demandado cerca del distrito financiero. Juan acepta la nueva ruta y el driver agent arranca el coche y cuidadosamente sale del garaje. Mientras que Juan ojea algunos documentos de trabajo, el driver agent autónomamente conduce el coche siguiendo la ruta seleccionada y pagando automáticamente los intersection managers que regulan las cruces que atraviesa. El driver agent continuamente consulta las fuentes de información que están diseminadas en la infraestructura y detecta que el precio de un cruce, anteriormente muy alto, ha bajado repentinamente. El driver agent replanifica la ruta: ahora se tardará 2 minutos más que antes, pero por 1.8 euros menos, un buen trato. El driver agent conduce el coche hasta el destino seleccionado, Calle Ferrocarril 6, y notifica a Juan con un mensaje de "Destino alcanzado".

La historia anterior parece una escena de una película de ciencia ficción, pero una escena de este tipo puede que esté más cerca de lo que podamos pensar. De hecho, remover el factor humano en la conducción, por medio de *vehículos autónomos* y de la integración de éstos con la *infraestructura inteligente*, se puede considerar como el objetivo final de aquel conjunto de sistemas agrupados bajo el nombre de Sistemas

Inteligentes de Transporte.

Los vehículos autónomos ya son una realidad. Hasta ahora, se han celebrado dos DARPA Grand Challenge y un DARPA Urban Challenge[1]. Los equipos que participan en este evento, organizado por el agencia de defensa americana (Defense Advanced Research Projects Agency), tienen que construir un vehículo autónomo capaz de conducir en el tráfico urbano, realizando complejas maniobras como adelantar y aparcar. En el DARPA Urban Challenge vehículos autónomos han interaccionado, por primera vez en la historia, con otros vehículos autónomos y también con vehículos conducidos por humanos en un entorno urbano. Un vehículo autónomo es un vehículo capaz de desplazarse autónomamente, sin intervención humana ni control remoto, utilizando varios sensores (sonar, láser) y sistemas de posicionamiento (GPS) para determinar las características de su entorno y actuar en consecuencia. Un ejemplo de este tipo de vehículo es Chevy Boss (figura E.1), desarrollado por la universidad Carnegie Mellon, ganador de la edición 2007 del DARPA Urban Challenge. Muchos productores de coches estiman que este tipo de tecnologías será económicamente más asequible y estéticamente más discreta en una década.

Otra iniciativa que fomenta la investigación para hacer realidad esta visión es la Vehicle Infrastructure Integration (VII)[2]. Esta iniciativa impulsa el desarrollo de tecnologías para integrar los vehículos con la infraestructura urbana, y así aumentar la seguridad y la eficiencia de las redes de tráfico urbanas. Por ejemplo, Dresner y Stone en [35] proponen una infraestructura basada en agentes para el control de los cruces. En su modelo, un cruce está gobernado por un agente inteligente (*intersection manager*) que asigna reservas de espacio dentro del cruce a los vehículos autónomos que se aproximan a él (véase figura E.2). Este enfoque ha demostrado, en un entorno simulado, varias ventajas, porque puede drásticamente reducir el tiempo de desplazamiento de los vehículos con respecto a un tradicional semáforo y permite la aplicación de póliticas de control centradas en los vehículos individuales en vez que en los flujos de vehículos.

---

[1]http://www.darpa.mil/grandchallenge/index.asp

[2]http://www.intellidriveusa.org

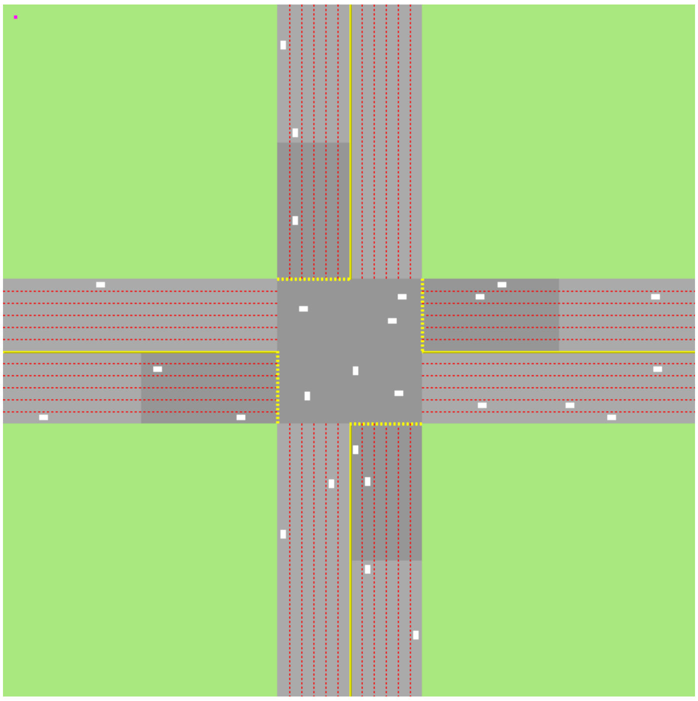


Figure E.2: Gestión de cruce *reservation-based*

Las aplicaciones que emergen con la introducción de estas tecnologías serán sistemas *abiertos*, *distribuidos*, *a gran escala* y compuestos por muchas *entidades autónomas*. La intrínseca distribución y el elevado grado de complejidad hacen que la natural descomposición de dichos sistemas sea en términos de *agentes autónomos* que interaccionan entre si [8][10][11]. En este tipo de sistemas podemos claramente distinguir dos categorías de agente: por un lado tenemos los agentes de la infraestructura, es decir, entidades que intentan controlar el sistema para mejorar su eficiencia, reducir las congestiones y agilizar el flujo de tráfico; por otro lado tenemos los agentes que representan los vehículos, es decir, entidades que conducen los vehículos de manera autónoma, toman decisiones por cuenta del usuario humano acerca de la ruta a seguir y la hora de salida, y aprenden de las experiencias pasadas. En general, el diseñador del sistema tiene control sobre los agentes de la infraestructura, y por lo tanto puede modelar sus espacios de acciones, objetivos, actitud mental (cooperativa o egoísta) etc. Por otro lado el diseñador del sistema no tiene control directo sobre los agentes que representan los vehículos. Por lo tanto la tarea a la cual se enfrentan los agentes de la infraestructura es muy compleja, porque intentan controlar un sistema que les proporciona observaciones parciales a través de actuadores que pueden condicionar sólo de manera indirecta el comportamiento y las acciones de los agentes que representan los vehículos.

Objetivos

En esta tesis doctoral se propone estudiar mecanismos distribuidos para la gestión y el control de una (futura) red de carreteras urbanas, donde vehículos inteligentes, gobernados por *driver agents*, interaccionan con la infraestructura para poder desplazarse. Este objetivo principal se puede descomponer en los siguientes sub-objetivos:

1. Esta tesis se basa en el sistema *reservation-based* para el control de los cruces propuesto por Dresner y Stone en [35]. Por lo tanto, el primer objetivo es un análisis detallado de dicho sistema, para evaluar el rendimiento y potenciales posibilidades de mejora en caso de *un único cruce*. Esta tarea se llevará a cabo desde dos perspectivas distintas: por un lado nos basaremos en la teoría de colas (*adversarial queueing theory*, AQT) [16] para elaborar diferentes políticas de gestión del cruce que maximicen la eficiencia del cruce; por otro lado nos basaremos en la teoría de las subastas combinatorias (*combinatorial auctions*, CA) [59] para modelar el mecanismo que regula la asignación de las reservas llevada a cabo por los *intersection managers*, de manera que los *driver agents* que más valoran las reservas sean recompensados con menores tiempos de desplazamiento.

2. Otro objetivo de esta tesis doctoral es extender el modelo propuesto por Dresner y Stone a una *red de cruces*. Este nuevo escenario pone nuevas e interesantes cuestiones, además de complicar sensiblemente el problema. Para tratar esta complejidad, descomponemos el problema de gestión del tráfico urbano en dos sub-problemas: *asignación de tráfico* y *control de tráfico*. La asignación de tráfico se puede ver como un problema de asignación de recursos y, dada la escala del sistema, necesita una formulación y un método de solución distribuido. Es

muy interesante notar como los mercados y las economías en general consiguen resolver el problema de asignación de recursos en entornos bastante complejos. De hecho, los mercados como método de solución de problemas de asignación de recursos ya han sido aplicados a muchos sistemas [25][104].

Por esta razón, se pueden aplicar mecanismos basados en los mercados para el diseño de un eficiente sistema de asignación de tráfico urbano. Las ventajas de este enfoque son múltiples:

(a) La dinámica del mercado proporciona a los *driver agents* incentivos para explorar alternativas a la hora de seleccionar la ruta a seguir.

(b) Los *intersection managers*, participando en el mercado y (eventualmente) regulándolo, tienen más poder para influenciar el comportamiento de los *driver agents*.

(c) Las políticas de precios tienen diferentes efectos sobre diferentes grupos de *driver agents*, por lo tanto es posible aplicar políticas de precios adaptadas a diferentes colectivos de *driver agents*.

(d) Alcanzar el equilibrio de mercado permite una asignación y una repartición eficiente de los recursos del sistema, es decir, la capacidad de la red, entre los *driver agents*.

(e) Las normas que regulan el mercado (y como se calculan los beneficios) se pueden diseñar de manera que los *intersection managers*, intentando maximizar sus beneficios, "involuntariamente" maximicen medidas de rendimiento del sistema subyacente, es decir, menos congestiones y menores tiempos de desplazamiento.

3. En este contexto, desarrollaremos un mercado computacional donde los *driver agents* tienen que adquirir las reservas necesarias para atravesar los cruces *reservation-based* que encuentran en sus rutas. Como dicho antes, en calidad de diseñadores del sistema, tenemos control sobre el comportamiento de los *intersection managers*, y también tenemos la libertad de diseñar los protocolos

de interacción entre éstos y los *driver agents*. De este modo, modelaremos los *intersection managers* como agentes cooperativos que aprenden qué política de precios optimiza una función de beneficio global. Modelando oportunamente cómo se calculan los beneficios, los *intersection managers* que maximizan la función de beneficio global indirectamente optimizan el tiempo de desplazamiento promedio del conjunto de *driver agents*.

4. Los mercados en los entornos reales suelen estar compuestos por agentes que se comportan de manera egoísta. Otro objetivo de esta tesis doctoral es estudiar un modelo de comportamiento competitivo de los *intersection managers*, basado en la teoría del equilibrio general. En este modelo, los *intersection managers* compiten en el mercado para proveer a los *driver agents* los recursos que "producen", es decir, las reservas para atravesar los cruces que controlan. En este caso, nuestro objetivo como diseñadores del sistema es alcanzar el equilibrio de mercado, es decir, una situación donde la demanda de recursos de los compradores (*driver agents*) es igual a la cantidad de recursos proporcionados por los proveedores (*intersection managers*).

5. Finalmente, otro objetivo de esta tesis doctoral es combinar las políticas de control de tráfico que se han estudiado en el caso de un único cruce con los modelos de mercado que se han estudiado en el caso de una red de cruces, con el objetivo de proporcionar una estrategia adaptativa e integrada para la gestión del tráfico urbano.

6. Paralelamente al diseño teórico de los mecanismos basados en el mercado, en esta tesis se desarrollará una herramienta para experimentar y evaluar los distintos mecanismos, mostrando la influencia de dichos mecanismos tanto sobre la utilidad de los *driver agents* (p.e., tiempo de desplazamiento) como sobre la utilidad global del sistema. Se analizarán las ventajas y desventajas de los modelos de flujo de tráfico propuestos en literatura, para luego desarrollar un simulador que se adapte a nuestras necesidades.

Estructura de la tesis

Esta tesis doctoral se estructura de la siguiente manera:

1. El capítulo 2 repasa el estado del arte de lo campos de investigación relacionados con esta tesis:

    (a) el conjunto de tecnologías hardware y software agrupadas bajo el nombre de Sistemas Inteligentes de Transporte [41].

    (b) los sistemas multiagente, con especial énfasis en los temas de coordinación, aprendizaje multiagente, diseño de mecanismos y la aplicación de los sistemas multiagente en el dominio del tráfico y transporte.

2. El capítulo 3 aborda el primer objetivo de esta tesis, analizando el mecanismo *reservation-based* para el control de cruces en el caso de un único cruce, evaluando diferentes políticas de gestión y asignación de las reservas, inspiradas en la teoría de colas y en la teoría de las subastas combinatorias.

3. El capítulo 4 aborda los otros objetivos de esta tesis, introduciendo el modelo cooperativo y el modelo competitivo para la asignación de tráfico. Además, se describe el modelo conceptual y la implementación de la herramienta de simulación desarrollada, llamada $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$ (Multiagent Intelligent Transportation Environment).

4. Finalmente, en el capítulo 5 se detallan las conclusiones principales de esta tesis, las publicaciones que han surgido a lo largo del desarrollo de este trabajo, así como algunas líneas de trabajo futuras.

Conclusiones

En esta tesis doctoral se han estudiado mecanismos distribuidos para la gestión y el control de una (futura) red de carreteras urbanas, donde vehículos inteligentes, gobernados por *driver agents*, interaccionan con la infraestructura para poder desplazarse. A continuación se resumen y comentan las principales aportaciones y se proponen algunas líneas de investigación futuras.

Contribuciones

El primer objetivo fue un análisis y extensión del sistema *reservation-based* de control de cruces propuesto por Dresner y Stone en [35]. Hemos analizado el rendimiento de varias políticas de asignación de las reservas basadas en la teoría de colas. Los resultados experimentales han demostrados una mejora estadísticamente significativa respecto a la política original *first-come-first-served* (FCFS) propuesta por Dresner y Stone, aunque esta mejora no sea probablemente tan significativa desde un punto de vista práctico. Aunque en teoría FCFS puede ser muy ineficiente en algunos casos extremos, en práctica dichos casos extremos son evidentemente raros. Por esta razón, hemos enfocado nuestro análisis en el estudio de políticas que puedan hacer surgir nuevas propiedades deseadas del sistema *reservation-based* de control de cruces. Basándonos en la teoría de las subastas combinatorias [59], hemos definido una nueva política que regula la asignación de las reservas a los *driver agents*. Los resultados de la evaluación empírica han demostrados que la política basada en las subastas combinatorias garantiza tiempos de desplazamiento menores para los *driver agents* que valoran más su tiempo, es decir, aquellos que someten las pujas más altas. Por otro lado, esta nueva política demostró que sufre un coste social, en términos de mayores tiempos de desplazamiento promedios, especialmente cuando la intensidad de tráfico es alta. Este efecto era de esperar, porque la política basada en las subastas combinatorias tiene como objetivo asignar una reserva al *driver agent* que paga más por ella, en vez que maximizar el número de reservas asignadas.

El segundo objetivo fue ir más allá del escenario con un único cruce, y extender el modelo *reservation-based* para una red de cruces. El nuevo escenario supuso nuevas e interesantes cuestiones, además de complicar sensiblemente el problema. Partiendo de la consideración que una estrategia de asignación de tráfico puede facilitar la tarea de una política de control, a través de una mejor distribución del flujo de tráfico en la red, hemos estudiados dos diferentes enfoques para la asignación de tráfico. Hemos modelado este problema como un problema de asignación de recursos, utilizando mecanismos basados en los mercados como método de solución. Hemos desarrollado dos distintos modelos económicos. El primero, $\mathcal{ECO}^+$, es un modelo cooperativo que aborda el problema desde la perspectiva de la *maximización de los beneficios*. Los agentes de la infraestructura (*intersection managers*) actúan como agentes cooperativos que aprenden conjuntamente qué vector de precios optimiza una función de beneficio global. Modelando dicha función de beneficio global de manera oportuna, hemos conseguido que los *intersection managers* indirectamente optimicen el tiempo de desplazamiento promedio del conjunto de *driver agents*.

El segundo modelo de mercado, $\mathcal{ECO}^-$, es un modelo competitivo que aborda el problema desde la perspectiva de la *adaptación*. En este modelo, los *intersection managers* actúan de manera egoísta, compitiendo el uno con el otro para el suministro de las reservas. En este caso nuestro objetivo como diseñadores del sistema fue alcanzar el equilibrio de mercado, es decir, una situación en la cual la cantidad de recursos demandadas por los compradores (*driver agents*) es igual a la cantidad de recursos proporcionados por los proveedores (*intersection managers*). La evaluación empírica demostró que a través de este mecanismo se consigue una asignación eficiente de los recursos disponibles, y que el sistema resultante se adapta dinámicamente a las variaciones de la demanda de recursos. Además, $\mathcal{ECO}^-$ supera las limitaciones que todo enfoque basado en aprendizaje tiene, incluido $\mathcal{ECO}^+$, tal como la estacionariedad del entorno o la necesidad de reducir espacio de estados y de acciones para que el problema sea tratable.

Por esta razón, elegimos el modelo competitivo como mecanismo base de asignación de tráfico, para luego combinarlo con la política de control basada en las subastas combinatorias y obtener una estrategia integrada y adaptativa para la gestión de tráfico, que hemos llamado $\mathcal{ECO}^-_{CA}$. La política de precios actúa en la distribución del tráfico en la red de carreteras, adaptando el *reserve price* (es decir, el precio mínimo al cual el *intersection manager* está dispuesto a subastar las reservas) y así generando un sistema en equilibrio dinámico, donde cruces sin usar se vuelven más baratos, mientras que los cruces más demandados se vuelven más caros. Si la demanda en correspondencia de un cruce muy concurrido baja por efecto de las fluctuaciones de los precios, también el coste social (en cada cruce) introducido por la política de control basada en subastas disminuye. De esta manera, una distribución más homogénea de los vehículos en la red conlleva un uso mejor de los recursos de la red, y por lo tanto el conjunto entero de *driver agents* es premiado con menores tiempos de desplazamiento. Al mismo tiempo, la política de control basada en subastas combinatorias premia los *driver agents* que valoran más las reservas, que por lo tanto experimentan menores tiempos de desplazamiento.

Para evaluar todos los modelos y mecanismos teóricos que hemos definido, se ha desarrollado una herramienta de simulación, llamada $\mathcal{M}.\mathcal{I}.\mathcal{T}.\mathcal{E}.$. Después de analizar las ventajas y desventajas de los diferentes modelos de flujo de tráfico que se pueden encontrar en literatura, hemos optado por implementar un simulador híbrido mesoscópico-microscópico. El simulador integra el modelo mesoscópico de Thomas Schwerdtfeger [89] con el modelo microscópico de Kai Nagel y Michael Schreckenberg [68]. El primero se ocupa de simular el flujo de tráfico en las carreteras, mientras que el segundo simula el flujo de tráfico dentro del cruce. Hemos optado por esta solución porque el modelo mesoscópico nos permite simular sistemas a gran escala, compuestos por miles de vehículos, mientras que el modelo microscópico nos permite simular con buena precisión y fiabilidad el flujo de tráfico dentro del cruce.

Línes de investigación futuras

Esta tesis doctoral no está exenta de posibles limitaciones, como las suposiciones que hemos hecho acerca de la infraestructura o el tipo de análisis experimental que hemos hecho. Por un lado, es verdad que nuestro modelo necesita una infraestructura basada en agentes muy avanzada, donde vehículos autónomos interaccionan con agentes que residen en la infraestructura. Aunque esta no sea la realidad actual, pensamos que este futuro no esté tan lejos como se podría pensar. Muchos investigadores en todo el mundo trabajan para hacer realidad un escenario donde vehículos autónomos interaccionan con otros vehículos autónomos y con la infraestructura, tal como aquellos involucrados en el DARPA Urban Challenge y en la Vehicle Infrastructure Integration Initiative. Además, hay un gran interés en desarrollar sistemas de control y gestión que, aprovechando de la "agentificación" de los vehículos y de la infraestructura, actúan sobre los vehículos *individualmente* en vez de *flujos* de vehículos, como el trabajo de Dresner y Stone en [35] o el trabajo de Schepperle y Bohm en [88].

Además, como investigadores en inteligencia artificial distribuida (y no ingenieros de tráfico), hemos analizado los mecanismos propuestos desde la perspectiva del usuario humano, teniendo como objetivo final la calidad de servicio percibida por el conductor (tiempos de desplazamiento, demoras, dinero gastado, etc.), y no nos hemos centrado en otras cuestiones como el dimensionamiento de la infraestructura de transporte o el análisis de los flujos de saturación en los cruces.

A partir del trabajo desarrollado en esta tesis doctoral ha surgido un conjunto de posibles líneas de investigación futuras en las que profundizar.

Modelos económicos. En este trabajo, dos modelos económicos han sido diseñados y evaluados, es decir, el *trading* y las subastas combinatorias *one-to-many*. Sin embargo, en nuestro escenario se podrían implementar otros modelos económicos. Por ejemplo, el mercado podría estar regulado por subastas dobles continuas (*continuous double auctions*), donde muchos vendedores (los *intersection managers*) someten pujas para vender, y muchos compradores (los *driver agents*) someten pujas para comprar, y el mercado se equilibra continuamente cuando se establece un *match* entre pujas para vender y pujas para comprar. También se podrían implementar mecanismos de regateo, donde los *driver agents* y los *intersection managers* negocian y acuerdan un precio aceptable para ambas partes.

Modelos del *driver agent*. Entre los objetivos de esta tesis no se encuentra el desarrollo de modelos sofisticados de *driver agent*, así que el modelo que se implementó fue bastante sencillo. La única decisión que el agente tiene que tomar se refiere a la selección de la ruta a seguir. Esta decisión se ha modelado como un problema de maximización de una función de utilidad: el agente evalúa las posibles alternativas y selecciona aquella que maximiza su utilidad (p.e., aquella con menor tiempo de desplazamiento estimado).

Para capturar la complejidad intrínseca de los sistemas de tráfico, es importante extender y enriquecer el modelo de comportamiento del *driver agent*. Por ejemplo, la toma de decisión del *driver agent* se podría modelar a través de dos capas [83], una reactiva para las decisiones a corto plazo (p.e., cambiar de carril), y una cognitiva para las decisiones más complejas como la selección de la ruta a seguir o la hora de salida. Además, otra característica de los conductores humanos que se debería implementar es el proceso de aprendizaje. Los conductores humanos implícitamente usan las informaciones históricas y las experiencias pasadas para actualizar la probabilidad de seleccionar una ruta especifica a una hora determinada. Por ejemplo, en [57] los *driver agents* toman decisiones probabilísticas acerca de la ruta a seguir, y periódicamente actualizan las probabilidades asociadas con las rutas disponibles de acuerdo con las recompensas (es decir, el inverso del tiempo de desplazamiento) que ha obtenido a lo largo de su vida.

Comunicación vehículo a vehículo. En todos los escenarios que han sido evaluados en esta tesis doctoral, sólo hay interacción entre los vehículos y la infraestructura. De este modo, no es posible para los vehículos colaborar y ponerse de acuerdo para intentar disminuir sus tiempos de desplazamiento. Sin embargo, la comunicación vehículo a vehículo está recibiendo gran atención por la comunidad científica e ingenieril [117]. En nuestro escenario, la comunicación vehículo a vehículo se podría emplear para enriquecer el espacio de acciones de los *driver agents*. Por ejemplo, vamos a considerar esta situación: el *driver agent* A tiene mucha prisa y estaría dispuesto a pagar un buen precio para adquirir una reserva, pero se encuentra detrás del *driver agent* B que no está dispuesto a pagar mucho para adquirir una reserva. En este caso, aunque el *driver agent* A consiga adquirir una reserva, sería imposible para él utilizarla, porque se encontraría bloqueado por el *driver agent* B. Si fuera posible para los vehículos comunicarse, el *driver agent* A podría ayudar el *driver agent* B a conseguir una reserva, subvencionándolo con una cantidad de dinero suficiente para conseguir una reserva, como se propone en el trabajo de Schepperle [88]. Además la comunicación vehículo a vehículo se podría utilizar para sincronizar los movimientos de los vehículos, comunicando variaciones de velocidad y cambios de carril, y así crear pelotones de vehículos, que potencialmente pueden incrementar mucho la capacidad de las carreteras y reducir las congestiones[3].

[3]http://www.path.berkeley.edu/PATH/Research/demos

References

[1] Annual urban mobility report. Technical report, Texas Transportation Institute, 2007.

[2] J. L. Adler, G. Satapathy, V. Manikonda, B. Bowles, and V. J. Blue. A multi-agent approach to cooperative traffic management and route guidance. *Transportation Research Part B - Methodological*, 39:297–318, 2005.

[3] Federal Highway Administration, editor. *Intelligent Transportation Systems. Benefits, Costs and Lessons Learned: 2005 Update*. United States Department of Transportation, 2005.

[4] W. B. Arthur. Inductive reasoning and bounded rationality. *American Economic Review*, 84:406–411, 1994.

[5] J. A. Bagnell and A. Y. Ng. On local rewards and the scalability of distributed reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS) 18*. MIT Press, 2006.

[6] G. Balan and S. Luke. History-based traffic control. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 616–621. ACM Press, 2006.

[7] T. Balch. *Behavioral Diversity in Learning Robot Teams*. PhD thesis, College of Computing, Georgia Institute of Technology, US, 1998.

[8] A. Bazzan. Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. *Autonomous Agents and Multiagent Systems*, 18(3):342–375, 2009.

[9] A. Bazzan, D. de Oliveira, F. Klügl, and K. Nagel. Adapt or not to adapt - consequences of adapting driver and traffic light agents. In *Adaptive agents and multi-agent systems III*, volume 4865 of *Lecture Notes in Artificial Intelligence*, pages 1–14. Springer-Verlag, 2008.

[10] A. Bazzan and F. Klügl. Reducing the effects of the braess paradox with information manipulation. In *Application of Agent Technology in Traffic and Transportation*, pages 85–97. Birkhäuser, 2005.

[11] A. Bazzan and F. Klügl, editors. *Multi-agent Architectures for Traffic and Transportation Engineering*. IGI-Global, 2008.

[12] R. Beckers, O. E. Holl, and J. L. Deneubourg. From local actions to global tasks: Stigmergy and collective robotics. In *Artificial Life IV*, pages 181–189. MIT Press, 1994.

[13] E. Bekiaris and Y. J. Nakanishi, editors. *Economic Impacts of Intelligent Transportation Systems*. Number 8 in Research in Transportation Economics. Elsevier, 2004.

[14] M. Ben-Akiva and M. Bierlaire. Discrete choice models with applications to departure time and route choice. In *Handbook of Transportation Science, 2nd edition*. Kluwer, 2003.

[15] B. Biskupski, J. Dowling, and J. Sacha. Properties and mechanisms of self-organizing manet and p2p systems. *ACM Trans. Auton. Adapt. Syst.*, 2(1):1–34, 2007.

[16] M. Blesa, D. Calzada, A. Fernández, L. López, A. Martínez, A. Santos, and M. Serna. Adversarial queueing model for continuous network dynamics. In *Proceedings of Mathematical Foundations of Computer Science*, volume 3618 of *Lecture Notes in Computer Science*, pages 144–155. Springer, 2005.

[17] G. Bottazzi, G. Devetag, and G. Dosi. Adaptive learning and emergent coordination in minority games. *Simulation Modelling Practice & Theory*, 10(5):321–347, 2002.

[18] C. Boutilier. Planning, learning and coordination in multiagent decision processes. In *Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*, pages 195–210. Morgan Kaufmann Publishers Inc., 1996.

[19] N. Busi, P. Ciancarini, R. Gorrieri, and G. Zavattaro. Coordination models - a guided tour. In *Coordination of Internet Agents: Models, Technologies, and Applications*, pages 6–24. Springer, 2001.

[20] Y. H. Chang, T. Ho, and L. Kaelbling. All learning is local: Multi-agent learning in global reward games. In *Proceedings of Neural Information Processing Systems (NIPS-03)*, 2003.

[21] J. Q. Cheng and M. P. Wellman. The walras algorithm: A convergent distributed implementation of general equilibrium outcomes. *Computational Economics*, 12(1):1–24, 1998.

[22] M. C. Choy, D. Srinivasan, and R. L. Cheu. Cooperative, hybrid agent architecture for real-time traffic control. *IEEE Transactions on Systems, Man, and Cybernetics - Part A*, 33(5):597–607, 2003.

[23] E. Clarke. Multipart pricing of public goods. *Public Choice*, 8:19–33, 1971.

[24] C. Claus and C. Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 746–752. AAAI Press, 1998.

[25] S. H. Clearwater. *Market-based control: a paradigm for distributed resource allocation*. World Scientific Publishing Co., Inc., 1996.

[26] B. Codenotti, S. Pemmaraju, and K. Varadarajan. The computation of market equilibria. *SIGACT News*, 35(4):23–37, 2004.

[27] M. Cremer, F. Meissner, and S. Schrieber. On predictive control schemes in dynamic rerouting strategies. In C. F. Daganzo, editor, *Proceedings of the 12th International Symposium on Transportation and Traffic Theory*, pages 407–426. Elsevier, 1993.

[28] B. C. d. Silva, E. W. Basso, A. Bazzan, and P. M. Engel. Dealing with nonstationary environments using context detection. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 217–224. ACM, 2006.

[29] C. F. Daganzo. Improving city mobility through gridlock control: an approach and some ideas. Technical report, U.C. Berkeley Center for Future Urban Transport, 2005.

[30] R. Dash, D. Parkes, and N. Jennings. Computational mechanism design: A call to arms. *IEEE Intelligent Systems*, 18:40–47, 2003.

[31] P. Davidsson, L. Henesey, L. Ramstedt, and J. Törnquist. Agent-based approaches to transport logistics. *Transportation Research Part C - Emerging Technologies*, 13:255–271, 2005.

[32] D. de Oliveira, P. R. Ferreira, A. Bazzan, and F. Klügl. A swarm-based approach for selection of signal plans in urban scenarios. In *Proceedings of the 4th International Workshop on Ant Colony Optimization and Swarm Intelligence*, volume 3172 of *Lecture Notes in Computer Science*, pages 416–417. Springer, 2004.

[33] E. Denti and A. Omicini. An architecture for tuple-based coordination of multi-agent systems. *Software: Practice and Experience*, 29(12):1103–1121, 1999.

[34] J. S. Drake, J. L. Schofer, and A. D. May. A statistical analysis of speed-density hypotheses. *Highway Research Record*, 156:53–87, 1967.

[35] K. Dresner and P. Stone. A multiagent approach to autonomous intersection management. *Journal of Artificial Intelligence Research*, 31:591–656, 2008.

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

[37] D. Fudenberg and D. K. Levine. *The Theory of Learning in Games*. MIT Press, 1998.

[38] W. A. Gamson. A theory of coalition formation. *American Sociological Review*, 26:373–382, 1961.

[39] D. Garg, Y. Narahari, and S. Gujar. Foundations of mechanism design: A tutorial part 1-key concepts and classical results. *Sadhana*, 33:83–130, 2008.

[40] C. Gershenson. Self-organizing traffic lights. *Complex Systems*, 16:29–53, 2005.

[41] S. Ghosh and T. Lee. *Intelligent Transportation Systems: New Principles and Architectures*. CRC, 2000.

[42] A. Gibbard. Manipulation of voting schemes. *Econometrica*, 41:587–601, 1973.

[43] H. Greenberg. An analysis of traffic flow. *Operations Research*, 7:79–85, 1959.

[44] B. D. Greenshields. A study in highway capacity. *Highway Research Board Proceedings*, 14:448–477, 1935.

[45] T. Groves. Incentives in teams. *Econometrica*, 41:617–631, 1973.

[46] S. Halle and B. Chaib-draa. Collaborative driving system based on multiagent modelling and simulations. *Transportation Research Part C - Emerging Technologies*, 13:591–656, 2005.

[47] D. A. Hensher and C. Sullivan. Willingness to pay for road curviness and road type. *Transportation Research Part D - Transport and Environment*, 8:139–155, 2003.

[48] J. Hernández, S. Ossowski, and A. García-Serrano. Multiagent architectures for intelligent traffic management systems. *Transportation Research Part C - Emerging Technologies*, 10(5):473–506, 2002.

[49] H. H. Hoos and C. Boutilier. Solving combinatorial auctions using stochastic local search. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pages 22–29. AAAI Press/MIT Press, 2000.

[50] P. B. Hunt, D. I. Robertson, R. D. Bretherton, and R. I. Winton. Scoot-a traffic responsive method of coordinating signals. Technical report, TRRL Lab. Report 1014, Transport and Road Research Laboratory, Berkshire, 1981.

[51] L. Hurwicz. Optimality and informational efficiency in resource allocation processes. *Mathematical Methods in the Social Sciences*, 1960.

[52] P. Ioannou and C. C. Chien. Autonomous intelligent cruise control. *IEEE Transactions on Vehicular Technology*, 42(4):657–672, 1993.

[53] D. Joksimovic, M. C. J. Bliemer, and P. H. L. Bovy. Optimal toll design problem in dynamic traffic networks with joint route and departure time choice. *Transportation Research Record: Journal of the Transportation Research Board*, 1923:61–72, 2005.

[54] R. Junges and A. Bazzan. Evaluating the performance of dcop algorithms in a real world, dynamic problem. In L. Padgham et al., editor, *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 599–606, 2008.

[55] R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. Wiley, 1976.

[56] A. S. Kelso and V. P. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50:1483–1504, 1982.

[57] F. Klügl and A. Bazzan. Route decision behaviour in a commuting scenario. *Journal of Artificial Societies and Social Simulation*, 7, 2004.

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

[59] V. Krishna. *Auction Theory*. Academic Press, 2002.

[60] B. Krogh and C. Thorpe. Integrated path planning and dynamic steering control for autonomous vehicles. *Proceedings of the IEEE International Conference on Robotics and Automation*, 3:1664–1669, Apr 1986.

[61] S. Lämmer and D. Helbing. Self-control of traffic lights and vehicle flows in urban road networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(04), 2008.

[62] M. Lauer and M. Riedmiller. An algorithm for distributed reinforcement learning in cooperative multi-agent systems. In *Proceedings of the 17th International Conference on Machine Learning*, pages 535–542, 2000.

[63] K. Leyton-Brown, Y. Shoham, and M. Tennenholtz. An algorithm for multi-unit combinatorial auctions. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pages 56–61. AAAI Press/MIT Press, 2000.

[64] H. Lieu. *Traffic-flow theory*. Federal Highway Administration, United States Department of Transportation, 1999.

[65] M. L. Littman. Value-function reinforcement learning in markov games. *Journal of Cognitive Systems Research*, 2(1):55–66, 2001.

[66] H. S. Mahmassani. Dynamic network traffic assignment and simulation methodology for advanced systems management applications. *Networks and Spatial Economics*, 1(3-4):267–292, 2001.

[67] A. MAs-Colell, M. D. Green, and J. R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[68] K. Nagel and M. Schreckenberg. A cellular automaton model for freeway traffic. *Journal de Physique I France 2*, pages 2221–2229, 1992.

[69] L. Nunes and E. Oliveira. Learning from multiple sources. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi Agent Systems*, pages 1106–1113, 2004.

[70] M. J. Osborne and A. Rubinstein. Bargaining and markets. Technical report, UCLA Department of Economics, 2005.

[71] S. Ossowski, J. Hernández, M. V. Belmonte, A. Fernández, A. García-Serrano, J. L. Pérez de la Cruz, J. M. Serrano, and F. Triguero. Decision support for traffic management based on organisational and communicative multiagent abstractions. *Transportation Research Part C - Emerging Technologies*, 13(4):271–298, 2005.

[72] S. Ossowski, J. Hernández, C. Iglesias, and A. Fernández. Engineering agent systems for decision support. In Tolksdorf Petta and Zambonelli, editors, *Engineering Societies in an Agent World III*, pages 184–198. Springer-Verlag, 2002.

[73] S. Ossowski and A. García Serrano. Social structure as a computational coordination mechanism in societies of autonomous problem-solving agents. In *Intelligent Agents V*, pages 133–148. Springer-Verlag, 1999.

[74] L. Panait and S. Luke. Selecting informative actions improves cooperative multiagent learning. In *Proceedings of the 5th international joint conference on Autonomous agents and multiagent systems*, pages 760–766. ACM, 2006.

[75] M. Papageorgiou. Traffic control. In *Handbook of transportation science*, pages 243–277. Kluwer Academic, 2003.

[76] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang. Review of road traffic control strategies. In *Proceedings of the IEEE*, volume 91, pages 2043–2067. IEEE, 2003.

[77] H. V. D. Parunak. Industrial and practical applications of dai. In *Multiagent Systems*. MIT Press, 1999.

[78] H. J. Payne. Models of freeway traffic and control. *Math. Models Publ. Sys., Simul. Council Proc*, 28(1):51–61, 1971.

[79] D. V. Pynadath and M. Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16:389–423, 2002.

[80] D. I. Robertson. Transyt: A traffic network study tool. Technical report, Rep. LR 253, Road Res. Lab., London., 1969.

[81] F. Rochner, H. Prothmann, J. Branke, C. Müller-Schloer, and H. Schmeck. An organic architecture for traffic light controllers. In *Proceedings of the Informatik 2006-Informatik für Menschen*, volume P-93 of *Lecture Notes in Informatics*, pages 120–127. Köllen Verlag, 2006.

[82] R. Rossetti, S. Bampi, R. Liu, D. Van Vliet, and H. Cybis. An agent-based framework for the assessment of drivers' decision making. In *Proceedings of the 3rd Annual IEEE Conference on Intelligent Transportation Systems*, pages 387–392, 2000.

[83] R. Rossetti and R. Liu. A dynamic network simulation model based on multi-agent systems. In *Applications of Agent Technology in Traffic and Transportation*, pages 181–192. Birkhäuser, 2005.

[84] M. H. Rothkopf. Thirteen reasons why the vickrey-clarke-groves process is not practical. *Operations Research*, 55(2):191–197, 2007.

[85] S. J. Russell and P. Norvig. *Artificial Intelligence: a Modern Approach*. Prentice Hall, 2003. 2nd edition.

[86] T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, pages 1–54, 2002.

[87] M. A. Satterthwaite. Strategy-proofness and arrow's conditions: Existence and correspondence theorem for voting procedure and social welfare functions. *J. Econ. Theory*, 10:187–217, 1975.

[88] H. Schepperle and K. Bohm. Agent-based traffic control using auctions. In *Cooperative Information Agents XI*, volume 4676 of *Lecture Notes in Computer Science*, pages 119–133. Springer, 2007.

[89] T. Schwerdtfeger. Dynemo: A model for the simulation of traffic flow in motorway networks. In *Proceedings of the 9th International Symposium on Transportation and Traffic Theory*, pages 65–87. VNU Science Press, 1984.

[90] S. Sen, M. Sekaran, and J. Hale. Learning to coordinate without sharing information. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 426–431. AAAI Press, 1994.

[91] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1-2):165–200, 1998.

[92] Y. Shoham, R. Powers, and T. Grenager. If multi-agent learning is the answer, what is the question? *Journal of Artificial Intelligence*, 171(7):365–377, 2007.

[93] S. Skiena. Maximum independent set. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, pages 218–219, 1990.

[94] M. Steingrover, R. Schouten, S. Peelen, E. Nijhuis, and B. Bakker. Reinforcement learning of traffic light controllers adapting to traffic congestion. In *Proceedings of the 17th Belgium-Netherlands Conference on Artificial Intelligence*, pages 216–223, 2005.

[95] K. Sycara. Multiagent systems. *AI Magazine*, 19:79–92, 1998.

[96] M. Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the 10th International Conference on Machine Learning*, pages 330–337. Morgan Kaufmann, 1993.

[97] M. Treiber and D. Helbing. Realistische mikrosimulation von straßenverkehr mit einem einfachen modell. In D. Tavangarian and R. Grützner, editors, *Proceedings of the Symposium "Simulationstechnik ASIM 200"*, pages 514–520, 2002.

[98] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, 2000.

[99] E Turban and J. Aronson. *Decision support systems and intelligent systems.* Prentice Hall, 1997.

[100] R. T. Underwood. Speed, volume and density relationships, quality and theory of traffic flow. *Yale Bureau of Highway Traffic, Newhaven, Connecticut*, 1961.

[101] R. T. van Katwijk, P. van Koningsbruggen, B. De Schutter, and J. Hellendoorn. A test bed for multi-agent control systems in road traffic management. In *Applications of Agent Technology in Traffic and Transportation*, pages 113–131. Birkhäuser, 2005.

[102] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *J. Finance*, 16:8–37, 1961.

[103] J. M. Vidal, P. Buhler, and H. Goradia. The past and future of multiagent systems. In *AAMAS Workshop on Teaching Multi-Agent Systems*, 2004.

[104] H. Voos and L. Litz. Market-based optimal control: a general introduction. In *Proceedings of the American Control Conference*, volume 5, pages 3398–3402, 2000.

[105] L. Walras. *Éléments d'économie politique pure, ou théorie de la richesse sociale.* 1874.

[106] J. G. Wardrop. Some theoretical aspects of road traffic research. In *Proceedings of the Institute of Civil Engineers*, volume 2, pages 325–378, 1952.

[107] J. G. Wardrop. Journey speed and flow in central urban areas. *Traffic Engineering and Control*, 9(11):528–532, 1968.

[108] C. Watkins. *Learning from Delayed Rewards.* PhD thesis, King's College, Cambridge, UK, 1989.

[109] C. Watkins and P. Dayan. Technical note: Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

[110] G. Weiss, editor. *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence.* MIT Press, 1999.

[111] G. Weiss and P. Dillenbourg. What is 'multi' in multi-agent learning? In *Collaborative-learning: Cognitive and Computational Approaches*, pages 64–80. Elsevier, 1999.

[112] R. Wiedemann. Simulation des straßenverkehrsflusses. Technical report, Schtiftenreihe des Instituts fur Verkehrswesen der Universitat Karlsruhe, 1974.

[113] M. Wiering. Multi-agent reinforcement learning for traffic light control. In *Proceedings of the 17th European Conference on Machine Learning*, pages 1151–1158, 2000.

[114] D. Wolpert and K. Tumer. Optimal payoff functions for members of collectives. *Advances in Complex Systems*, 4(2):265–279, 2001.

[115] M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley and Sons, 2002.

[116] P. R. Wurman, M. P. Wellman, and W. E. Walsh. A parametrization of the auction design space. *Games and Economic Behavior*, 35(1-2):304–338, 2001.

[117] X. Yang, J. Liu, F. Zhao, and N. H. Vaidya. A vehicle-to-vehicle communication protocol for cooperative collision warning. In *Proceedings of the 1st International Conference on Mobile and Ubiquitous Systems*, pages 114–123. IEEE Computer Society, 2004.

[118] M. Yokoo, E. H. Durfee, T. Ishida, and K. Kuwabara. The distributed constraint satisfaction problem: Formalization and algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):673–685, 1998.

[119] Y. Zahavi. Traffic performance evaluation of road networks by the α-relationship. *Traffic Engineering and Control*, 14(5):228–231, 1972.