



HAL
open science

Multi-Agent Approaches for Dynamic Transportation Problems

Mahdi Zargayouna

► **To cite this version:**

Mahdi Zargayouna. Multi-Agent Approaches for Dynamic Transportation Problems. Multiagent Systems [cs.MA]. Université Paris Dauphine, 2019. tel-02465720

HAL Id: tel-02465720

<https://hal.science/tel-02465720v1>

Submitted on 17 Feb 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



ECOLE DOCTORALE DE DAUPHINE

Multi-Agent Approaches for Dynamic Transportation Problems

Habilitation Thesis
(*Computer Science*)

by

Mahdi Zargayouna

JURY

10/01/2019

- Coordinator: Mr Jamal Atif
Professor at Université Paris Dauphine
- Evaluators: Mrs Franziska Klugl
Professor at Orebro University
Mr Abderrafiâa Koukam
Professor at Universié Belfort-Montbéliard
Mr René Mandiau
Professor at Université de Valenciennes
- Examiners: Mr Flavien Balbo
Professor at Mines Saint-Etienne
Mrs Marie-Pierre Gleizes
Professor at Université Toulouse III
Mr Omer Rana
Professor at Cardiff University
Mr Gérard Scemama
Research Director Emeritus at IFSTTAR
Mr Danny Weyns
Professor at Katholieke Universiteit Leuven

A Maya & Sofia

Contents

1	Introduction	1
1.1	Context and Motivation	1
1.2	Research Domain	2
1.3	Problems, Objectives and Methodology	3
1.4	Research Projects	5
1.5	Research Supervision	5
1.6	Design Choices	6
1.7	Multi-Agent Modeling Applied to Transportation Applications	7
1.8	Multi-Agent Simulation for Dynamic Transportation Applications	10
1.9	Multi-Agent Optimization for Dynamic Transportation Problems	11
2	Multi-Agent Simulation for Dynamic Transportation	13
2.1	Introduction	14
2.1.1	Multimodal Traffic Simulation Platform	14
2.1.2	Impact of Real-Time Information	15
2.1.3	Distributed Traffic Simulation	15
2.2	The SM4T Simulator	16
2.2.1	Data and Parameters	20
2.2.2	Multi-Agent System	22
2.2.3	Application: Impact of Passenger Real-Time Information	23
2.2.4	Temporal model of the simulation	28
2.2.5	Experiments and Results	29
2.3	Simulations Distribution	36
2.3.1	Generic Agent-Based Traffic Simulations	37
2.3.2	Distribution Methods	39
2.3.3	Diffusive Load Balancing	43

2.3.4	Experiments and Results	44
2.3.5	Discussion	50
2.4	Conclusion and Perspectives	50
3	Multi-Agent Optimization for Dynamic Transportation	53
3.1	Introduction	53
3.2	Online Localized Resource Allocation Problems	55
3.2.1	Resources and Consumers	56
3.2.2	Allocation Modeling	58
3.2.3	Solution Constraints	59
3.2.4	Objectives	60
3.3	Multi-Agent Approach for Urban Parking Management	60
3.3.1	An OLRA Modeling of Urban Parking	61
3.3.2	Multi-Agent Solution for Urban Parking	62
3.3.3	Experiments	69
3.4	Multi-Agent Approach for Dynamic VRPTW	74
3.4.1	An OLRA Modeling of the Dynamic VRPTW	75
3.4.2	Multi-Agent Solution for the Dynamic VRPTW	75
3.4.3	Experiments	80
3.5	Multi-Agent Configuration for the Dial A Ride Problem	82
3.5.1	Multi-Agent System for the Multi-Company Dial A Ride	84
3.5.2	Experiments	86
3.6	Conclusion and Perspectives	91
4	Conclusion and Perspectives	93
4.1	Summary of the Research Work	93
4.2	Perspectives	94
4.2.1	Integrating Autonomous Transportation	95
4.2.2	Designing Smart Systems	95

Contents	vii
4.2.3 Optimizing New Mobility Services	95
4.2.4 Merging Modeling, Simulation and Optimization	96
4.2.5 Distributing Multi-Agent Environments	96
4.2.6 Modeling Multi-Scale Mobility	97
4.2.7 Coupling Simulation and Regulation	97
Bibliography	99

List of Figures

1.1	LACIOS architecture	8
1.2	General schema	9
2.1	Multimodal travel platform public interface	17
2.2	Workflow of the simulation	19
2.3	Fundamental diagram	20
2.4	Speed in function of density	21
2.5	Multi-agent system model	25
2.6	Spatiotemporal graph	26
2.7	Travel time improvements (1,000 passengers)	31
2.8	Travel time improvements (5,000 passengers)	32
2.9	Travel time improvements (10,000 passengers)	32
2.10	Travel time improvements (20,000 passengers)	33
2.11	Travel time improvements (30,000 passengers)	33
2.12	Travel time improvements (connected passengers vs. non-connected passengers) with 1,000 passengers	34
2.13	Travel time improvements (connected passengers vs. non-connected passengers) with 30,000 passengers	35
2.14	Average execution times	35
2.15	Average memory usage	36
2.16	Steps of a simulation	38
2.17	Agent-based distribution	40
2.18	Environment distribution	41
2.19	Diffusive load balancing	44
2.20	Speedup for the agent-based distribution (Paris-Saclay network)	46
2.21	Speedup for the environment-based distribution (Paris-Saclay network)	46

2.22	Speedup for the agent-based distribution (Barabasi network)	47
2.23	Speedup for the environment-based distribution (Barabasi network)	48
2.24	Impact of the number of hosts (agent-based distribution)	48
2.25	Impact of the number of hosts (environment-based distribution)	49
2.26	Load imbalance	50
2.27	Speedup for the different methods	51
2.28	Execution time in function of the number of processes	51
3.1	An OLRA Modeling of Urban Parking	63
3.2	Information dissemination in the community	64
3.3	Assistant agent state diagram	64
3.4	Assistant agent internal data flows	68
3.5	Cooperative model (with $\theta = 15$)	71
3.6	Coopetitive model (with $\theta = 15$)	71
3.7	Density of the agents searching for a spot (300 agents and $\theta = 15$)	72
3.8	Impact of information lifetime, with $\theta = \{5, 10, 15, 20\}$	72
3.9	Impact of the variation of r	73
3.10	Number of messages (centralized vs. cooperative model)	74
3.11	An OLRA Modeling of the Dynamic VRPTW	76
3.12	Spatiotemporel graph	78
3.13	Initial action zone of a vehicle agent	79
3.14	Action zone after the insertion of a traveler	79
3.15	Computation time of the static problem instances	88
3.16	QoS versus total costs for a single and a multiple company setting	89
3.17	Average total costs for instances with a minimal QoS	90
3.18	QoS for instances with a minimal QoS	91
3.19	QoS versus total costs for a single and a multiple company setting with a minimal QoS	92

List of Tables

2.1	Implementation of the space-time graph in the simulator	27
2.2	Information level in each scenario	30
2.3	The bus capacities in the simulator	30
2.4	Synthesis of travel time improvement (Average (Standard deviation))	31
2.5	Load balancing: computational times (in seconds)	49
3.1	Results summary (criterion: fleet size)	81
3.2	Results summary (criterion: total traveled distance)	82

Introduction

“It should be known that the secret and spirit of speech – that is, expression and address – lie in conveying ideas. If no effort is made to convey ideas, speech is like dead land which does not count.”

Ibn Khaldun, The Muqaddimah (1377)

Contents

1.1	Context and Motivation	1
1.2	Research Domain	2
1.3	Problems, Objectives and Methodology	3
1.4	Research Projects	5
1.5	Research Supervision	5
1.6	Design Choices	6
1.7	Multi-Agent Modeling Applied to Transportation Applications	7
1.8	Multi-Agent Simulation for Dynamic Transportation Applications	10
1.9	Multi-Agent Optimization for Dynamic Transportation Problems	11

1.1 Context and Motivation

This document presents a synthesis of my research work, which started in 2003 as PhD student at University of Paris Dauphine and has continued as lecturer at University of Paris Nanterre (2006-2008), postdoc at TU-Delft (2008-2009) and since 2009 researcher at the French institute of science and technology for transport, spatial planning, development and networks (Ifsttar).

In the computer science domain, my work is positioned in the field of artificial intelligence. My main interests are in the multi-agent systems paradigm, in which systems are designed in the form of distributed, autonomous and intelligent interacting entities. My theoretical contributions concern formal models for the systems modeling and methods for simulation and optimization. In this context, I have applied theories from process algebra, multi-agent systems and simulation models. In 15 years, I have conducted a set of research projects, developing them directly, supervising their development or in collaboration with partners. This work has a unifying theme: dynamic transportation problems. It has been tackled with a unique approach: the computational modeling following the multi-agent paradigm.

My research is supported by a set of activities (collaborations, scientific animation, supervision and teaching) aiming at giving it a more solid foundation and a higher influence. I have also acted as member of the scientific committee or reviewer for more than 40 journals, conferences and workshops and I have worked as expert in the evaluation of several research institutions and projects proposals.

1.2 Research Domain

The transportation sector is an important generator of activities (5% of the European GVA¹ in 2015 [EC, 2016]). Mobility and technological development growth are accelerating and this sector is facing increasingly important issues. The activity growth in this domain is accompanied with a structural and organizational evolution of the mobility supply and demand.

On the one side, the transportation supply is changing, traditional transportation services, with fixed timetables, are now in competition with new mobility services, which are gaining more and more markets and are becoming an important generator of traffic. Ride sharing, dial a ride and adaptable public transportation are attracting more and more travelers. Autonomous transportation is also on the rise with the technological and legal obstacles being progressively tackled. Autonomous transportation could be responsible of a great part of the traffic in the future. European public transportation markets are being opened to competition, and all current transportation actors are obliged to adapt to this new reality if they desire to safeguard and to develop their historic dominance over the market. Ecologic conscience is growing all over the world, and measures like road space rationing and pollution taxes are encouraging the transportation supply to be more efficiently managed and more environment-friendly.

On the other side, demand is also changing. Indeed, with the widespread use of smartphones and high speed Internet, travelers are more connected today and are less reluctant to using new mobility services. They are becoming more demanding with the quality of service of the transportation supply. They expect a quality of service that goes beyond punctuality, and desire the satisfaction of multiple criteria in their trips. They also expect to have ubiquitous services, that would work in all major cities of the world. On the other hand, they increasingly accept to be tracked and to provide personal information, if this would improve the quality of the transportation service that is offered to them. They expect to have an accurate, personalized and real-time information service. With the widespread of social networks, they are now capable to challenge the operators traveler information services. With ride sharing, they are even participating in the transportation supply.

This evolution requires a rapid transformation of the existent services and the development of new ones to answer these needs. The methods that the transportation actors are using to manage their networks are increasingly confronted with an operational and organizational reality that is different from their design time reality. The new operational reality notably includes the widespread use of smartphones and onboard units. This means that, contrarily to the last decades, we now have the possibility to track individual movements of travelers and transportation means. To take full advantage of these new mobile data sources, the modeling paradigm has to allow the individual representation of transporta-

¹Gross Value Added, Postal and courier activities included.

tion system components. For this reason, the models, algorithms and architectures that we propose are based on the multi-agent systems (MAS) paradigm.

In [Bazzan and Klügl, 2014], the authors state multiple reasons to use multi-agent systems in transportation applications. Among other arguments, the authors indicate that the solving of several transportation problems with multi-agent systems is natural and intuitive. Autonomous agents are also able to model heterogeneous systems and to capture complex constraints that connect all the solving phases. The dynamic transportation applications that we consider are particularly suitable for an agent-based design. Indeed, the objective in this kind of applications is to take into account intelligent behaviors, interacting in an open, dynamic and complex environment [Bessghaier et al., 2012]. In most of these applications, transportation actors (e.g. travelers or vehicles) perceive individual information, and make individual decisions, while being situated in and interacting with an environment (e.g. the transportation network or the information sources) on which they have partial and incomplete information. This configuration obeys the general definitions of agents as entities that: i) are situated in some environment, ii) are capable of autonomous actions on it [Wooldridge and Jennings, 1995], iii) can perceive this environment and iv) have a partial and incomplete perception of it [Ferber, 1999].

The research work presented in this document is positioned in this context. We are interested in the development of multi-agent systems to support multimodal² transportation actors in solving the complex problems induced by the changes in the supply and demand of transportation services. We propose a number of models, algorithms and simulations supporting the actors in observing and estimating the future states of the transportation networks. Our work contributes to a better understanding of the problems raised by the new mobility services (dial a ride, ride sharing, urban parking search systems, etc.) and to a better design of operating systems and traveler information systems, integrating these new services.

1.3 Problems, Objectives and Methodology

Our work has the objective of improving the modeling, the simulation and the complex problem solving for decision support in the domain of dynamic transportation of persons. A transportation problem is classically defined as the transfer of entities (goods, persons, vehicles, etc.) between geographically separate locations at a minimum cost [Steenbrink, 1974]. The collective movement of these entities is called traffic. For this movement to take place, a transportation infrastructure is used. The solving of transportation problems, such as the traffic assignment problem, the traveling salesman problem, the vehicle routing problem and the toll pricing problem has attracted a huge amount of research and was a necessary step to understanding the problems complexity and the suitable methods to solve them. The methods to solve these transportation problems are still useful today for planning or supply dimensioning purposes. However, since they consider supply, demand and infrastructure as being static, the methods developed to solve this kind of transportation problems can rarely be used as is in operational settings, in traffic regulation or in online ride sharing problems for instance.

A dynamic transportation problem can be defined as “a transportation problem over

²Multimodality in transportation refers to the use of different transportation modes.

time” [Bookbinder and Sethi, 1980]. Today, this term refers to a wide range of transportation problems where the problem data are not available *a priori* [Barbucha and Jdrzejowicz, 2009]. The missing or incomplete information might concern the transportation demand (e.g. goods, travelers or customers), the transportation supply (e.g. the drivers or vehicles) or the transportation environment (e.g. the transportation policy, the trafficable network or the traffic status). Online transportation problems are a subset of dynamic transportation problems where the missing information concerns the transportation demand, which is discovered while the system is running. In online transportation problems, the system response time to the online demand is key. Time and space are key concepts in dynamic transportation problems. These transportation problems are interesting to study for their high spatiotemporal complexity.

The methods to solve nowadays dynamic and online transportation problems have to evolve, following today’s configurations and practices. Indeed, operational transportation systems are now rarely grounded exclusively on one decision center that would concentrate all the information and all the decisions of the system. Indeed, several system components (vehicles, drivers, control entities, etc.) are located in the transportation environment and are equipped with computational power that could allow them to react to events in that environment. With the scale change of the supply and the demand of mobility services, systems with exclusive centralized decision systems would be eventually obliged to evolve toward more distribution and parallelism. A system solving dynamic transportation problems should also be able to take into account the missing or incomplete information as it becomes available, and provide short response time. For these reasons, we believe that the multi-agent systems paradigm is a candidate of choice to address these problems.

The founding principle of multi-agent systems is to allow the coexistence and the interaction of autonomous intelligent entities, called agents, evolving in an environment, with the objective of having a satisfactory or realistic overall behavior of the whole system. This paradigm could be a relevant modeling taking into account the distribution of resources, the personalization of behaviors and the dynamics of the environment. It is also a suitable paradigm for the consideration and the representation of human behaviors, and could relevantly integrate models from the human and social science domain. We have chosen to use this paradigm in the problems that we face, and we have been interested more particularly in the interaction and coordination between agents in the system.

Our contributions in the multi-agent systems domain are linked to the transportation applications that support them. Indeed, all our proposals in the multi-agent systems domain have led to transportation systems development, and conversely all the transportation problems that we have addressed are modeled in the form of multi-agent systems. We have defined a specification model to design and implement open multi-agent systems that we apply to dial a ride systems and travelers information. We have designed and implemented a multi-agent simulation platform that we have applied to multimodal traffic representation, and distribution models that we have applied to multi-agent traffic simulations. We also have defined a multi-agent model solving the urban parking problem. We have designed a multi-agent configuration to solve the multi-company dial a ride problem and we have specified a multi-agent space-time model that we have applied to dynamic vehicle routing problems.

This research work has been made possible using funded projects and with the supervision of students and researchers. These activities are briefly described in the following.

1.4 Research Projects

We have participated in two European research networks, one European research project and two National Projects. All of them address new technologies and intelligent transportation systems. Firstly, we were partners of the Nearctis (Network for Advanced Road Cooperative Traffic management in the Information Society, 2008-2013)³ Network of Excellence (NoE), which addresses cooperative systems (Vehicle-to-Vehicle and Vehicle-to-Infrastructure). In the network, I participated as reports writer and as invited researcher (six months) with a partner of the NoE. Secondly, we were partners and I was member of the management committee of the European COST action TU1102 ARTS (Towards Autonomic Road Transport Support Systems, 2011-2015)⁴, which addresses autonomous traffic support systems, at the crossroad between multi-agent systems and traffic modeling. In this context, we organized a summer school at École des Ponts Paristech (France) and we launched a scientific collaboration with Cardiff University on the subject of multi-agent traffic simulation and cloud computing. Thirdly, we participated in the European project Instant Mobility (2011-2013)⁵ as tasks leaders. Our objective was the design of a multimodal guidance platform and the design and implementation of a demonstration for the project (ITS World Congress 2012). Fourthly, we are participating as package leaders in the French project LasDim (Large Scale Data Infrastructure for Mobility, 2016-2020)⁶, which objective is to provide a referential for mobility in the Île-de-France Region. We are finally participating in the French project MSM (Modeling of Mobility Solutions, 2016-2020)⁷, with the SystemX research Institute, which focuses on mobility representation at a neighborhood scale.

1.5 Research Supervision

This research work is supported with teaching and supervision. I have given more than 1000 hours of lectures in six different universities and engineering schools in Paris (university of Paris Dauphine, university of Paris Est, university of Paris Nanterre, École des ponts ParisTech, Esiee-Paris and Epita Engineering schools). My teachings allow me to have a wider vision of my research domains and provide opportunities to hire students for Masters and PhD projects. My teachings cover all the domains that I address: my disciplinary domain with multi-agent systems lectures, my application domain with intelligent transportation systems and mobility simulation lectures and my technological environment with distributed computing and Web services lectures.

My work has been supported by three PhD theses, five Masters theses, and two postdocs that I co-supervised on key subjects in the transportation domain. Nesrine Bessghaier's PhD project (co-directed with Flavien Balbo and Suzanne Pinson) tackled the problem of resource allocation in transportation with an application on urban parking management. Amine Othman's PhD thesis (co-directed with Gérard Scemama) addressed the impact of real-time travelers information on transit networks. Matthieu Mastio's PhD thesis (co-directed with Gérard Scemama and Omer Rana) dealt with distributed multi-agent traffic simulations. I have also supervised two postdocs. Funded by the European Commission, the

³<http://nearctis.ifsttar.fr/en/welcome/>, last visited 8 Oct. 2018.

⁴http://www.cost.eu/COST_Actions/tud/TU1102, last visited 8 Oct. 2018.

⁵http://cordis.europa.eu/project/rcn/100100_en.html, last visited 8 Oct. 2018.

⁶<http://lasdim.net/>, last visited 8 Oct. 2018.

⁷<http://www.irt-systemx.fr/project/msm/>, last visited 8 Oct. 2018.

first dealt with multi-agent simulation platforms (Besma Zeddini). The second was funded by VeDeCom research Institute and dealt with the data specification for multi-agent traffic simulations (Feirouz Ksontini). Finally, I have co-supervised the work of five M.Sc thesis. I have notably co-supervised Ferdi Groetenboers (with Matthijs de Weerd) who worked on a multi-company dial a ride problem and Khadim Ndiaye (with Flavien Balbo) who worked on the community-based urban parking management. I am currently co-supervising two more PhD theses and one more postdoc. Xavier Boulet's PhD project (co-directed with Gérard Scemama and Fabien Leurent) deals with the scales interaction (regional and neighborhood scales) in mobility simulations. Negin Alisoltani's PhD project (co-directed with Ludovic Leclerc) tackles the interaction between traffic modeling and new mobility services. Finally, I am currently co-supervising a postdoc on quality of service of public transportation services (Muhammad Naeem), funded by VeDeCom Research Institute.

I am now responsible of the "Modeling & Multimodality" research group of the Grettia (Engineering of Surface Transportation networks and Advanced Computing Laboratory) lab in the Components and Systems (Cosys) department of Ifsttar (18 members), which brings together researchers and engineers working on multimodal modeling in transport, including computer scientists and applied mathematicians. I am also co-animator of a multidisciplinary scientific pillar of the Cosys department (about a hundred members, the third of the staff of the Cosys department - Ifsttar), gathering researchers and engineers working on the modeling of multi-scale traffic and the regulation of transport systems. My tasks include the collaborative definition of the research project, the animation of meetings and seminars, the preparation of programming documents and annual indicators as well as the compilation of reports for evaluation. In all these tasks, I am the intermediary between the management staff and the colleagues researchers, with the human and social management that implies.

1.6 Design Choices

We have made some structuring design choices in the work presented here, which can be summarized as follows. First, as for all agent-based systems, the entities representation is always individual. That means that we do not represent a collective behavior of the system entities in the form of equations or flows dynamics. Instead, we associate properties and a behavior with each entity, the collective behavior is made of the interaction of the individual entities. Representing entities individually in the form of agents does not necessarily mean that each agent represents a single individual in the real world. Indeed, each agent could represent a group of individuals in the real world, which would have the same properties and the same behavior. This individual representation has several benefits. It is possible to have heterogeneous properties and different or conflicting behaviors in the same system. We could also have systems that mix human actors and artificial ones in the same application. Having individual representations also allows to retrace the system dynamics from an individual point of view, and to explain what has happened exactly.

Secondly, the agents in all our proposal are rational and adopt a behavior that optimizes some criteria. In the transportation applications that we consider, these criteria are more or less directly linked to their travel times. In the dynamic vehicle routing problem for instance (cf. chapter 3), both vehicles and travelers aim at minimizing their detour w.r.t a direct itinerary. In a multi-company settings of the dial a ride problem (cf. chapter 3), the

travelers choose the company offering the best quality of service, which is based on a ratio between a direct itinerary travel time and the proposed travel time. In the proposed traffic simulations (cf. chapter 2), the traveler agents are interested in the fastest itineraries.

Thirdly, there is always an explicit representation of the environment in all of our proposals. The environment represents at a minimum the transportation network, which is always represented in the form of a graph. That means that we do not reason at a road, crossroad or station scale, but always at a network scale, representing a neighborhood, city or region scale. That also means that we do not represent the very details of the edges and the nodes, in the form of a continuous or a 3D space for instance. In some of our proposals, the environment also integrates a temporal dimension, in addition to the space dimension. The consideration of the only space dimension means that the agents only reason about the present and the current situation. Considering space-time graphs allows the agents to reason about the future and to use the environment for planning purpose. The environment does not necessarily represent the transportation network. In some of our proposals, the environment is used for communication and coordination purposes. In this case, all the agents interact with the environment and do not have to maintain knowledge about the other agents of the system.

Finally, the time is always represented as discrete events. In our simulation proposals, time progression happens at discrete moments, and all the activities between each progression are supposed to happen simultaneously. The space-time network mentioned earlier is designed based on a discretization of time and a duplication of the spatial graph multiplied by the number of resulting discrete times.

Our contributions can be classified in three categories, which shape the structure of the remainder of this habilitation thesis. First, the multi-agent modeling applied to dynamic transportation applications, which will be briefly summarized hereinafter. Second, the multi-agent simulation of dynamic transportation applications which is the subject of chapter 2. Finally, the multi-agent optimization of dynamic transportation problems (chapter 3).

1.7 Multi-Agent Modeling Applied to Transportation Applications

In this work, in continuation of Flavien Balbo's PhD thesis [Balbo, 2000], we represent the multi-agent environment explicitly for the modeling of multi-agent systems. This consideration of the multi-agent environment as an explicit entity is a structuring choice for my research work. Our proposal was a formal coordination language called LACIOS that extends data-oriented coordination models. Data-oriented coordination languages, with the pioneer language Linda [Gelernter, 1985] and its extensions, propose solutions for the coordination of sequential processes via a shared data space, made of data tuples. The main advantages of these models, and that are of interest for the design of multi-agent systems, are the anonymous interaction and the decoupled communication in space and time. We adopt this shared space-based model, and we extend its expressiveness and functionalities to propose a communication environment for the agents of the system. The resulting model coordinates agents instead of processes. In LACIOS, the agents have a complex behavior (sequential, conditional, parallel, interaction with an external system) and a state that is

observable from the shared space. The agents use this state to condition their interaction in the system. LACIOS uses a data structure that allows a representation of the exchanged data that is richer than state-of-the-art tuples. It also allows for a matching mechanism that is more expressive than the state-of-the-art templates, and which allows the agents to express a complex interaction need. The interaction mechanism allows an agent to condition his interaction with the state of several entities of the environment, allowing for what we call a “contextual interaction”. We also propose a specification language based on process algebra to allow the specification of multi-agent systems adhering to the model, a security mechanism that allows the agents to secure their interactions and a programming language allowing to execute the multi-agent system written in the specification language.

One of the advantages of the model is that it allows for the specification of open systems. A MAS written in LACIOS is an open system in two ways. On the one side, agents in LACIOS can join and leave the system freely. On the other side, external - non modeled - systems and users can interact with the MAS. Users (e.g. travelers) interact with the MAS by instantiating the values of certain variables in the code of the agents that represent them. External systems (e.g. trains) can interact with the MAS by instantiating variables values as well. They can also execute agents that interact with the MAS Environment directly. The figure 1.1 illustrates the MAS architecture. The modeled MAS executes on a host, where (local) agents add, read and take objects to/from the MAS environment. Every agent is either independent (like agent 1), or representing a non-modeled system or user in the MAS. This external interaction raises security threats in the MAS, and LACIOS was defined to tackle them.

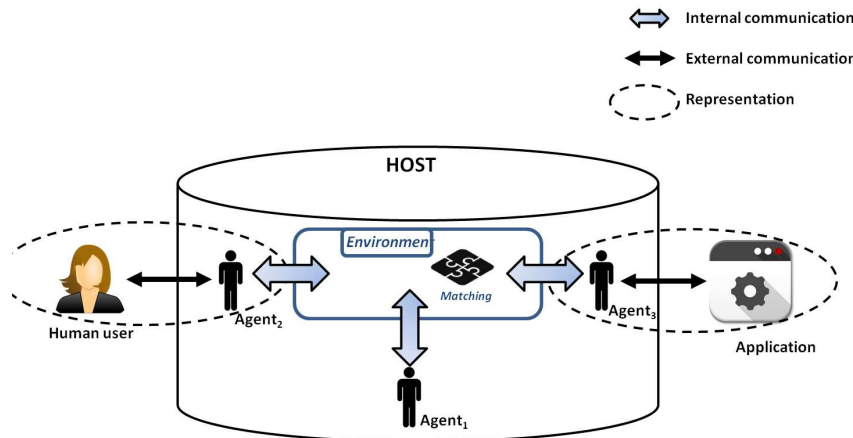


Figure 1.1: LACIOS architecture

Agents can specify *who* can access the object that they add to the environment. To do so, agents have to have a state defining who they are. This is the first modification we perform to the original model: the consideration of an abstraction of agents’ states in the form of data at language level. These states are defined as a set of *property*←*value* pairs (e.g. $\{identifier \leftarrow 10, position \leftarrow node_1\}$). Agents’ states in LACIOS are data representing the state of the agents that are accessed by the environment only for matching and security

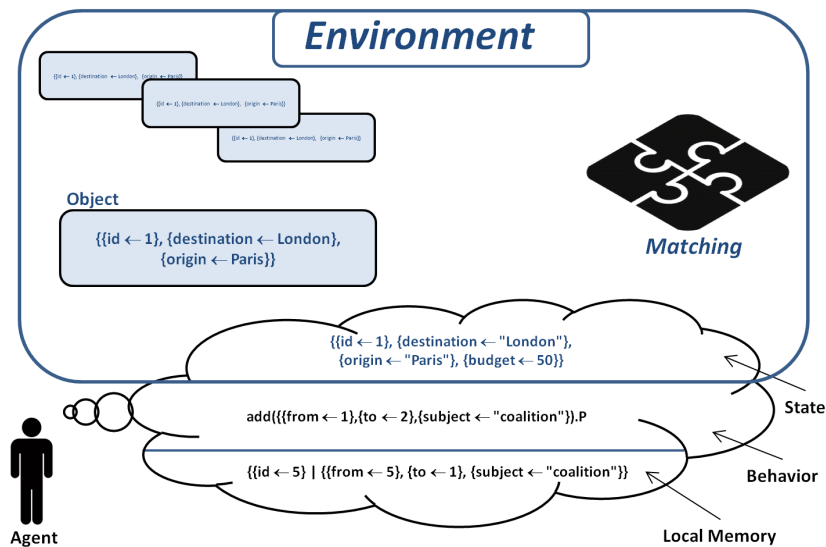


Figure 1.2: General schema

purposes (they are not directly accessible by the other agents). To be observable to the others, an agent has to add an object representing himself to the environment explicitly (as in Fig. 1.2, where the agent decides not to publish a part of his state). Having data representing agents in the environment allows the agents of the system to discover each others by simply interrogating the environment *à la Linda*.

Both global control and local control are possible in a MAS written in LACIOS. Global control is defined by the designer of the system, who knows the conditions under which certain additions of objects are fraudulent and we provide him or her with a global control of agents additions of objects. A threat to authenticity (when an agent tries to forge a message for example) is an example of such fraudulent additions. A set of rules are defined by the designer specifying the conditions that have to be met by an object to be added to the environment (for instance, the *from* property of a message has to be equal to the *id* property of the agent adding the message). Local control on the other side is defined by the agents of the system, who know best the conditions under which the perception or retrieval of an object they add is fraudulent, and we provide them with local control to manage the observability of their own objects. A confidentiality threat (e.g. the interception by an agent of another's confidential information or message), or a threat to availability (e.g. the deletion of the agent's information or message by another agent) are examples of such fraudulent access. LACIOS allows agents to define the observability rules of the objects that they add, which they associate with them, and the multi-agent environment verifies these conditions before allowing objects perception or reception.

The model and the language that we have proposed are virtually applicable to any multi-agent system. They are particularly useful for the open systems in which agents join and leave the system freely, and where the agents interaction needs are complex. Dynamic transportation applications fit perfectly with such a description. Traveler information systems are an example of such systems. Indeed, travelers join the system when they need an information. Transportation operators and transportation information services and sources

may also join and leave the system freely. The interaction needs of the travelers are complex, since they are related to their itineraries, and are contextual since they can be related to several sources at the same time (an example of needs expression: “I desire to take the bus only if it is not raining”). Dial a ride systems also fit with the description of the systems that might take particularly advantage of the model and the language. Indeed, both travelers and vehicles join and leave the system freely. The vehicles interaction needs are also complex and contextual, since they are related to the scheduling of the travelers in their route. We have proposed two systems written in LACIOS to set up these two applications. We have also defined a programming language that translates a LACIOS script to a Java program.

1.8 Multi-Agent Simulation for Dynamic Transportation Applications

Chapter 2 presents our contributions to the multi-agent simulation of dynamic transportation applications. A first research on simulation started with Flavien Balbo and Fabien Badeig [Badeig, 2010], in continuation with the work on environment-based modeling. The idea was to use the same environment-centered principle to set up multi-agent simulations. The environment would activate and schedule the agents activities in a contextual manner. The resulting simulations are general-purpose, which could of course be as well applied to transportation applications. Following this work, which is not in the scope of this document, we were interested in simulations that would be dedicated to traffic and dynamic transportation applications.

Modeling and simulation play an important role in transportation networks analysis [Gruer et al., 2001]. With the widespread of personalized real-time information systems, the behavior of the simulation depends heavily on individual travelers reactions to the received information. As a consequence, it is relevant for the simulation model to be individual-centered, and multi-agent simulation is one of the the most promising paradigm in this context. This is the subject of Amine Othman’s PhD thesis [Othman, 2016] and Matthieu Mastio’s PhD thesis [Mastio, 2017]. The first contribution of this chapter is the SM4T⁸ multi-agent simulation platform, which represents multimodal networks, and the travelers and vehicles movements on them. We also present a travelers information application implemented in SM4T, in which we model personal real-time information as well as local information. Results show that real-time personalized information may have an increasingly positive impact on overall travel times following the increasing ratio of connected passengers. However, there is a ratio threshold after which the effect of real-time information becomes less positive.

The second contribution of this chapter concerns the scalability of this kind of simulations. Indeed, representing the movements of realistic numbers of travelers within reasonable execution times requires significant computational resources. It also requires relevant methods, architectures and algorithms that respect the characteristics of transportation networks. We define two generic multi-agent simulation models representing the existing sequential multi-agent traffic simulations. The first model is fundamental diagram-based model (e.g. SM4T), in which travelers do not interact directly and use a fundamental

⁸Simulation of Multi-agent MultiModal Mobility of Travelers

diagram of traffic flow to continuously compute their speeds. The second model is car following-based, in which travelers interact with their neighbors to adapt their speeds to their surrounding environment. Then we define patterns to distribute these simulations in a high-performance environment. The first is agent-based and distributes agents equally between available computation units. The second pattern is environment-based and partitions the environment over the different units. The results show that agent-based distribution is more efficient with fundamental diagram-based model simulations while environment-based distribution is more efficient with car following-based simulations.

1.9 Multi-Agent Optimization for Dynamic Transportation Problems

Chapter 3 presents our contributions to the optimization of dynamic transportation problems. The chapter first proposes a generic definition of dynamic transportation problems. A generic definition of dynamic transportation problems is a difficult task, provided the wide variety of operational objectives and constraints. We made the choice of considering the subpart of problems that can be defined as a resource allocation problem. Resource allocation problems remain a general framework, since even the general dynamic transportation problem can be defined as a resource allocation problem, where the nodes and edges of the network are the resources and where the goods or persons to transport are the consumers. However, the original resource allocation problem is too general for dynamic transportation applications, and we propose a more adequate definition of this problem, called online localized resource allocation problem, and dedicated to transportation problems, taking systematically into account the space and time dimensions.

Three dynamic transportation applications are then defined following this definition, and multi-agent solutions are proposed to them. The first application is urban parking. The solving of parking spots search is an important issue, because of its economic and ecologic fallouts. We have proposed a multi-agent system aiming at decreasing, for the drivers of private vehicles, the search time for a spot. In this system, a community of drivers share information about spots availability. The decrease in search time is obtained with the communication and the collaboration of the system's agents. The communication between agents is fulfilled via an intervehicular network, avoiding a costly infrastructure. The cooperation model does not need any initial information and insure the scalability of the proposed system. The solution was tested with different cooperation models and the results show the improvement of the spots search time with the use of the proposed community system. This contribution is based on Nesrine Bessghaier's PhD project and on Khadim Ndiaye's M.Sc thesis.

The next application is dynamic vehicle routing problems. These are complex optimization problems with many operational applications. In this context, the exclusive and direct consideration of the traditional optimization criteria (basically the number of mobilized vehicles and the total traveled distance) has adverse effects by creating empty space-time zones in the network. We use a space-time representation of the multi-agent environment to introduce a new criteria that the vehicles optimize (their "space-time action zone") instead of the traditional criteria. The experimental results show the superiority of the new measure over the traditional measures. This contribution is based on Besma Zeddini's postdoc, in continuation with my PhD thesis [Zargayouna, 2007].

The third application is the dial a ride problem, which provides a service half-way between individual taxis and public transportation. The traveler receives a door-to-door taxi service that is less expensive than individual taxis. In return, the traveler accepts a detour w.r.t a direct trip and to share his ride with other travelers. In several implemented systems, a drop in the quality of service of dial a ride systems is observed, in terms of travel times that are too long for the travelers. Generally, a single company is responsible of providing the service for a certain region. In this work, we verify the effect on costs and quality of service if several companies were to compete on each traveler's request for service. The traveler bases his company choice on the offered quality of service. To perform the tests, we put in place a multi-agent framework to simulate the assignment of the requests to companies following a reversed auction on quality of service, and the insertion of travelers in the vehicles routes is performed following online optimization techniques. Results show that the multi-company configuration improves the service that is offered to the travelers, with a moderate increase in the costs for the companies. This contribution is based on Ferdi Grootenboers M.Sc thesis.

This document is composed of four chapters. Some parts of the document are new material, but large parts of it are based on published papers, with the necessary harmonization effort between them. The first part of chapter 2 is based on a combination of [Zargayouna et al., 2014] and [Zargayouna et al., 2018], while the second part is based on [Mastio et al., 2018], augmented with new unpublished results. The first two sections of chapter 3 are based on [Zargayouna et al., 2016], the third section is based on a part of [Zargayouna and Zeddini, 2012] while the fourth section is a simplified version of [Grootenboers et al., 2010]. This document ends with a concluding chapter 4, which summarizes the contributions presented in this document and provides perspectives to my scientific project.

Multi-Agent Simulation for Dynamic Transportation Applications

“Given for one instant an intelligence which could comprehend all the forces by which nature is animated and the respective situation of the beings who compose it – an intelligence sufficiently vast to submit these data to analysis – it would embrace in the same formula the movements of the greatest bodies of the universe and those of the lightest atom; for it, nothing would be uncertain and the future, as the past, would be present to its eyes.”

Pierre-Simon Laplace, A Philosophical Essay on Probabilities (1825)

Contents

2.1	Introduction	14
2.1.1	Multimodal Traffic Simulation Platform	14
2.1.2	Impact of Real-Time Information	15
2.1.3	Distributed Traffic Simulation	15
2.2	The SM4T Simulator	16
2.2.1	Data and Parameters	20
2.2.2	Multi-Agent System	22
2.2.3	Application: Impact of Passenger Real-Time Information	23
2.2.4	Temporal model of the simulation	28
2.2.5	Experiments and Results	29
2.3	Simulations Distribution	36
2.3.1	Generic Agent-Based Traffic Simulations	37
2.3.2	Distribution Methods	39
2.3.3	Diffusive Load Balancing	43
2.3.4	Experiments and Results	44
2.3.5	Discussion	50
2.4	Conclusion and Perspectives	50

2.1 Introduction

To define mobility policies, the decision makers need support systems to assist them. In this context, simulation is one of the important tools that allow them to test strategies and multiple scenarios without impacting the real traffic [Abadi et al., 2015, Hu et al., 2012, Mcardle et al., 2014, Zheng et al., 2014]. However, transportation systems are becoming progressively more complex since they are increasingly composed of connected entities (mobile devices, connected vehicles, etc). It becomes critical that simulation tools take into account this fact. Indeed, with the generalization of real-time traveler information, the behavior of modern transportation networks becomes harder¹ to analyze and to predict. For these reasons, multi-agent simulation, which adopts an individual-centered approach, is one of the most relevant paradigms to design and implement such applications. The design and development of multi-agent traffic simulations are relevant in several contexts and in pursuit of various objectives. The simulation can be used to validate the impact of the use of cooperative systems [Gueriau et al., 2015], to test changes in behavior after the introduction of new mobility services, such as carpooling, dial a ride, etc. A multi-agent traffic simulation platform simulates the behavior of travelers interacting in a complex, dynamic and open environment, on which they have a partial perception [Badeig et al., 2008]. Each agent tries to find the most efficient route to reach his destination in a network that is evolving dynamically. In some applications, an agent can potentially be informed of the status of the network and use this information to modify his route.

Static transportation studies and traffic assignment usually have the objective to study the dimensioning of the transportation supply, and define the final dispatching of the demand on the available supply, in an equilibrium state. In the contrary, the simulations described in this chapter represent the dynamic movements of travelers all along their trip, step by step (a few seconds between each two steps). The final outcome of these simulations can also be used for supply dimensioning, but they are more relevant for the study of dynamic regulation policies and real-time operational measures.

2.1.1 Multimodal Traffic Simulation Platform

In the context of the EC-funded project *Instant Mobility*¹, we have designed and implemented a multimodal travel platform [Zargayouna et al., 2012] that guides the traveler to the needed transit stops and stations or directs the car driver along the best route. The platform supports the provision of the essential subset of these services. To test the platform, we needed to continuously feed it with thousands of individual online transportation service requests and dynamic positions. Since it is not feasible to equip real travelers with tens of thousands of smartphones, we chose to simulate their behaviors. This context was the original motivation that oriented our work toward multimodal traffic simulations. In the first part of this chapter, we present the multimodal travel simulator SM4T (Simulator for Multi-agent MultiModal Mobility of Travelers) that interacts, in its original version, with the platform on behalf of travelers and transportation means. The platform was later made standalone, and now works independently from any guidance platform. It enables for the rapid prototyping and execution of simulations for several kinds of online applications. The application simulates the movements of travelers on the different transportation modes and networks while taking into account the changes in travel times and the status

¹http://cordis.europa.eu/project/rcn/100100_en.html, last visited 8 Oct. 2018.

of the networks. Since it assumes the continuous localization of travelers, SM4T can notably simulate and evaluate the impact of a wide range of community transportation apps, such as user-submitted travel times and route details, community-based driver assistance, community parking, etc. The first part of this chapter presents the main building blocks of the simulator, which we believe provide design principles that are reproducible in other transportation applications pursuing similar objectives.

2.1.2 Impact of Real-Time Information

To give an example of an actual dynamic application that could profit from the simulation, we consider a real-time passenger information scenario. Indeed, it is now possible to provide passengers with optimal itineraries and also to update these itineraries in real-time, following the dynamic status of the networks (congestions, breakdowns, accidents, etc.). Providing passengers with traffic information has generally a positive effect on them and improves traffic by alleviating congestion and reducing waiting and travel times. However, the broadcast of traffic information and the individual guidance of passengers might have adverse effects and impact negatively the traffic status. These possible side effects have been listed in [Pereira et al., 2015]. The authors state that three phenomena could be observed as a side effect of the use of advanced traveler information systems. First, when travelers receive too much information, they tend to ignore it and try to find an itinerary on their own (oversaturation). Second, if the same itineraries are provided to travelers who have the same transportation preferences, congestion might be created (concentration). Finally, if the same alternative is provided to too many passengers, the original congestion could be simply moved to another location (overreaction). We evaluate the advantages and the limits of the provision to passengers of real-time public transportation traffic, notably in disturbed conditions, using SM4T. These effects are measured by simulating several scenarios according to the ratio of connected passengers to a real-time information system. The information provided to the connected passengers is based on a space-time representation of the transportation networks. Results show that real-time personalized information may have an increasingly positive impact on overall travel times following the increasing ratio of connected passengers. However, there is a ratio threshold after which the effect of real-time information becomes less positive.

2.1.3 Distributed Traffic Simulation

When simulating dynamic transportation applications, it is sometimes important to model and simulate a realistic number of travelers to correctly observe the effects of individual decisions. In the aforementioned project for instance, the objective was to supply the multimodal platform with a realistic individual and multimodal travel queries and dynamic positions of travelers and vehicles. The simulation of an actual number of passengers in a big city (up to millions of travelers) requires both considerable computing power and an architecture allowing the distribution of computations on many hosts. The majority of current multi-agent traffic simulators do not provide such distribution. This induces limitations on the number of simulated travelers, means of transportation and the size of the considered networks. Our main objective in the second part of this chapter is to present reproducible generic distribution patterns that could be used by existing and future implementations of multi-agent traffic simulations. We propose to study

distribution methods for multi-agent traffic simulations. We define two generic multi-agent simulation models, representing the main types of multi-agent simulations of the literature. The first model is called macroscopic model, in the sense that travelers do not interact directly but use a fundamental diagram of traffic flow to continuously compute their speeds. This is the choice performed for instance by [Zargayouna et al., 2014, Meignan et al., 2007] and [Cajias et al., 2011]. The second model is called microscopic, in which travelers interact with their neighbors to adapt their speeds to their surrounding environment. This is the most common choice performed in the literature for multi-agent simulations (e.g. [Behrisch et al., 2011, Maciejewski and Nagel, 2012]). We study two distribution patterns (agent-based and environment-based) applied to these two simulation models. The results show that agent-based distribution is more efficient with macroscopic model simulations while environment-based distribution is more efficient with microscopic model simulations.

Here follow the main design choices that we have made in this chapter. In the simulations presented here, the travelers and vehicles are always represented individually. This does not necessarily mean that each agent represents a single individual in the real world. Indeed, each agent could represent a group of individuals with the same properties and the same behavior. Having individual representations allows, among other benefits, to retrace the system dynamics from an individual point of view, and to explain what has happened exactly. Second, the transportation network is represented in the form of a graph. We do not represent the very details of the edges and the nodes, in the form of continuous or 3D space for instance. Third, in our simulations, time progression happens at discrete moments, and all the activities between each progression are supposed to happen simultaneously. Finally, the agents in this chapter always optimize their travel times.

The remainder of this chapter is structured as follows. In section 2.2, we present the simulation platform, its parameters and data, the behaviors of the agents in the system, etc. We also present the application based on the simulator: the simulation of the impact of real-time information in transit networks. Section 2.3 defines the distribution methods that we propose. It presents a generic simulator for the execution of both macroscopic and car microscopic multi-agent traffic simulations, the two distribution patterns (agent-based and environment-based) and their application to the two simulation models, and the experimental results. Section 2.4 concludes the chapter.

2.2 The SM4T Simulator

Figure 2.1 presents a view of the multimodal travel platform from an external point of view. It specifies the requirements for a specific region to set an Instant Mobility service and the different exchanges that should take place with the platform. The platform interacts with three types of actors: the public transportation operators, the road transportation operators and the travelers (passengers and drivers).

Each public transportation operator has to provide the platform a description of their network and theoretical timetables. As depicted in the figure, we do not require the public transportation operators to have a common database that integrate all the public transportation means and networks. Each transportation mode operator might have its own database, and the platform has to integrate them and manage them simultaneously. Each

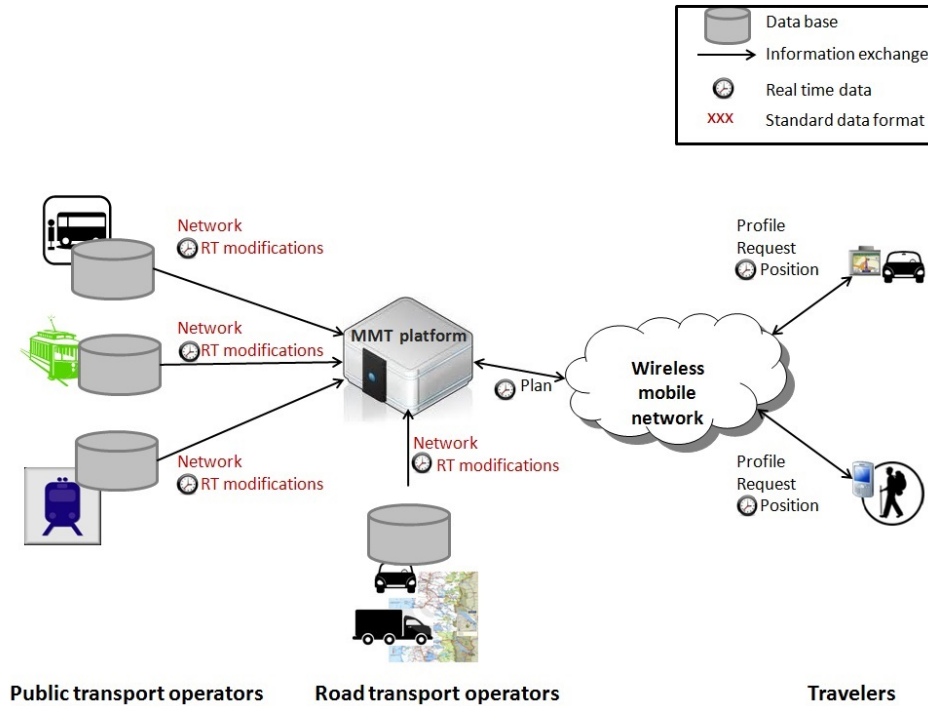


Figure 2.1: Multimodal travel platform public interface

public transportation operator has to provide three types of data:

- The description of the transportation supply
- The position, the advance/delay of all its fleet
- The events that cause transportation services disruptions

As for public transportation operators, road transportation operators have to provide the platform with a description of their network, together with all static information related to it. They also have to provide three types of data:

- The description of the road network
- The speeds, densities, occupancy rates and states
- The events that impact the transportation supply

Each traveler provides the system with his profile, which includes detailed information regarding his properties and preferences. After receiving a plan from the platform, the mobile device of the traveler continuously sends his current position to the platform. If the difference between the actual position of the traveler and the planned position (i.e. the expected position following the plan that was proposed to the traveler) is big enough,

following the traveler's preferences, the traveler receives a new plan taking into account his new context.

To function properly, the platform has to interact with the outside world (cf. the public interface described earlier). Since it is very difficult to test this scenario under real conditions, with notably a significant number of multimodal travelers equipped with smartphones and an Instant Mobility app, we design and implement the SM4T simulator. Its original purpose was to completely replace the environment of the platform and provide it with all the needed data: static and dynamic, concerning the transportation operators and the travelers all along their journey.

Since we desire to model individual and heterogeneous behaviors of passengers, we choose the multi-agent paradigm for the system modeling. Several multi-agent simulators for passenger mobility exist in the literature. MATSim [Maciejewski and Nagel, 2012] is a platform for micro-simulation of traffic that is widely known and used. AgentPolis [Jakob et al., 2012] is an agent-based platform for multimodal traffic. Miro [Chipeaux et al., 2011] simulates the urban dynamics of a population and proposes a multi-agent simulation to test planning scenarios, while Transims [Nagel and Rickert, 2001] represents travelers movements in multimodal networks and assesses the effects of the policy changes in traffic. However, none of these proposals assume the continuous localization of travelers and means of transport. In the simulation platform presented in this section, travels are fully multimodal: they concern private cars, all the public transport modes as well as pedestrians. Carpooling, car and bicycle sharing services are easily integrable in the simulation. Travelers routes are continuously monitored and alternatives are proposed to them if something wrong happens with their itinerary.

As a result of this study on existing multi-agent simulators for passengers mobility, we decide to design and implement a new simulator. To design and implement a multi-agent simulator, there are mainly two possibilities. It is either possible to directly use a programming language, or to use a multi-agent simulation platform. We choose to use an existing agent-based platform because the implementation is faster and more efficient. Two criteria have guided our platform choice. First, we desire to be able to deploy the simulations on many hosts, which is one of our ongoing works (cf. next section). The second capacity that we are looking for is the ability of the simulation platform to create geospatial models, i.e. its ability to handle geographic information. Based on these criteria, we have studied multiple agent-based platforms such as Jade [Bellifemine et al., 2007], Mason [Luke et al., 2004], Gama [Taillandier et al., 2012] and Repast Symphony [Tatara and Ozik, 2009]. We believe that Repast Symphony is the platform that meets most our criteria. The platform integrates a GIS library (Geotools), and provides advanced geographic services (graph modeling, shortest paths, visualization of 2D and 3D data, etc.). It also offers an easy integration of distribution platforms such as Terracotta and Gridgain. This is why the simulator is based on this platform.

The original SM4T simulator represents travelers (drivers and passengers) and transportation means (public transportation vehicles and private cars) individually and simulates their dynamic movements and their interaction with the guidance platform (tracking, planning requests, plans update, etc.). To this end, it needs input data that are similar to those needed by the platform, and some additional parameters.

The simulator can now be used in isolation from the platform, in which case the exchanges with the platform are turned off. To this end, the itinerary planning and travel

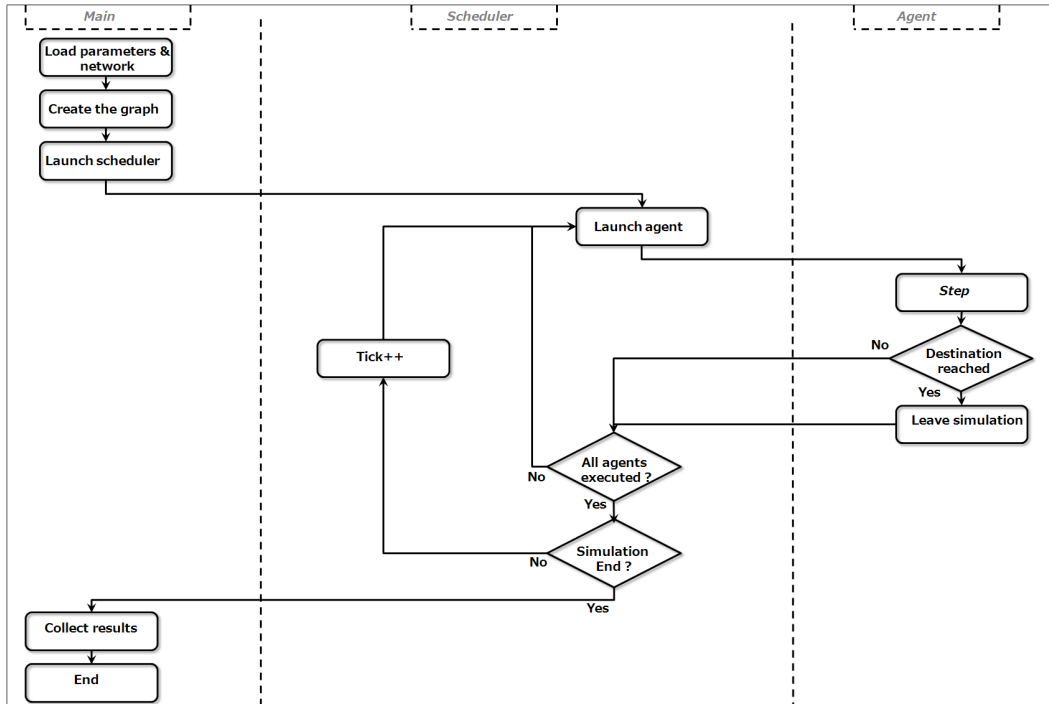


Figure 2.2: Workflow of the simulation

monitoring features of the platform are integrated in the simulator.

The workflow of Figure 2.2 details one simulation execution and structures the remainder of the presentation. A simulation first loads the parameters (the duration of a simulation, the number of agents in the simulation, etc.). The graphs representing the transportation networks are then created (described in section 2.2.2) and the scheduler is launched. The scheduler is a central component in the simulation since it synchronizes the execution of the agents. It schedules the execution of the agents for each tick of simulated time. The actions executed in one simulated tick of time are considered as being simultaneous. At each tick of simulation, the agents are executed in parallel threads (to take advantage of multicore architectures), while the scheduler waits for them to finish the execution of the actions planned for the current tick, before incrementing the tick and starting over again. When launched, each agent executes a *step* method, in which he executes the behavior described later (compute an itinerary, move on the network, get available information, wait for a vehicle, etc.). Hence there is a main program with a scheduler that controls the simulation, launches the agents and synchronizes their actions, while between each two time ticks, agents are executed in parallel. When the simulation ends, the results are collected and the simulation run ends. The collected results concern all the travelers itineraries and some indicators, such as the travel times and waiting times for the travelers.

To execute a simulation, input data and some additional parameters are needed.

2.2.1 Data and Parameters

2.2.1.1 Data

The input data (xml files) of the simulator are:

- the road network,
- the pedestrian network,
- the public transportation network,
- the transfer mapping,
- the timetables of the public transportation vehicles,
- the travel patterns,
- the travelers profiles.

The road network is a description of the roads, crossroads and driving directions. Each road has, among other information, a corresponding minimum and maximum speeds. It also has a mapping between traffic flows (vehicles/hour), the traffic density and speeds. The speed of a private car on a road is computed following the number of other agents traveling on the same road. To this end, a fundamental diagram of traffic flow is used. The diagram defines a relation between the flow (vehicles/hour) and the density (vehicles/km) (cf. Figure 2.3) on a road or a part of a road to calculate the speed of the agents at each time. The fundamental diagram suggests that if we exceed a critical density of vehicles, the more vehicles are on a road, the slower they will move (cf. Figure 2.4). This choice of speeds computation was originally motivated by the fact that we needed realistic movements of vehicles, but not necessarily very detailed ones. To have an even more realistic representation of vehicles movements on a road, a car-following model has to be used. This choice for speeds computation is of great importance when distributing the simulation over several hosts, as we will see it in the second part of this chapter.

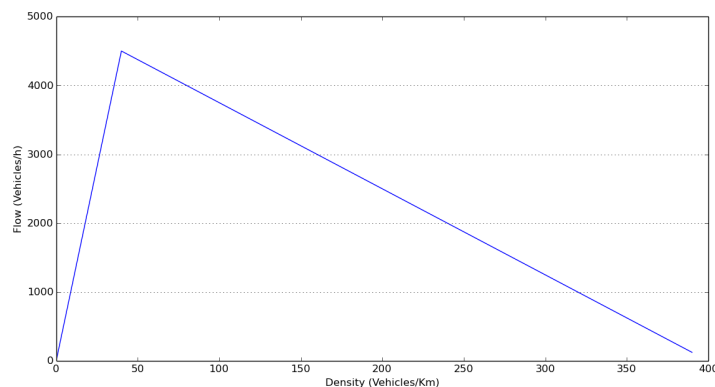


Figure 2.3: Fundamental diagram

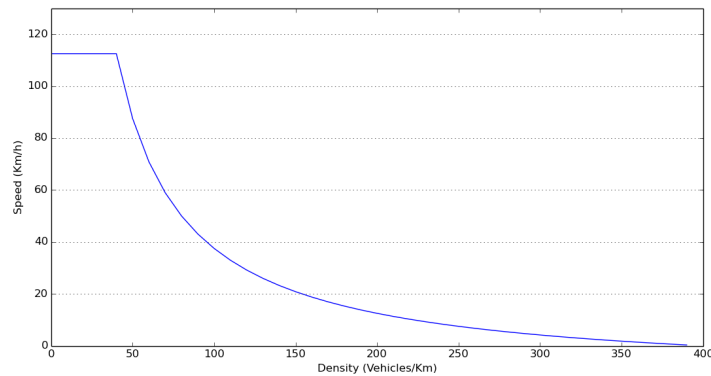


Figure 2.4: Speed in function of density

The pedestrian network is a subset of the road network in which pedestrians can move, but which is undirected (pedestrians don't have to obey to the one-way limitations). A public transportation network is composed of transportation lines, each of them composed of a set of itineraries. An itinerary is composed of a sequence of edges. Each edge has a tracing in the form of a sequence of pairs $\langle longitude, latitude \rangle$, and is composed of an origin node and a destination node. Finally, every node is defined by its name and coordinates. The transfer mapping is a table informing about the stops of the network for which a transfer by foot is possible and the road transportation nodes that are reachable from the stops. The three networks and the transfer mapping form a connected hypergraph, made of different edge types corresponding to each transportation mode, and form the spatial environment of the system.

The timetables of the vehicles are composed of a set of missions. Each mission corresponds to a specific itinerary and describes the path of a vehicle and the corresponding visit times. Each timetable is then a sequence of pairs $\langle stop, time \rangle$. A travel pattern, when existing, clusters the considered geographic region in zones and describes the number of persons asking to leave or to join each region. The travelers profiles, when existing, define the properties and preferences of the travelers. The preferences of the travelers also define the accepted time gap between their computed itinerary and their real situation, before asking for a new up-to-date itinerary.

2.2.1.2 Parameters

Simulation duration: Two values define the duration of a simulation run. The first is an interval defining the first and last date that are simulated. In the absence of these parameters, we use respectively the values associated to the maximum date and minimum date in the timetables of the public transportation vehicles. The second value that has to be defined is the number of discrete ticks of time that the simulation will execute before terminating. At each tick, all the agents are activated for a particular action defined by their behavior.

Agents speeds: The speeds of the agents are defined in four ways:

- Public transportation vehicles: based on the timetables (input data), their speeds are inferred from the $\langle stop, time \rangle$ pairs: the speed is equal to the distance between two successive stops divided by the difference between the visit time of the destination stop and the departure time of the origin stop.
- Private cars: their speeds are inferred from the fundamental diagram of the road they are currently traversing.
- Pedestrians: they are defined in the traveler profile.
- Default: the simulator user defines speeds for pedestrians, cars and public transportation vehicles as parameters. The user-defined pedestrian speed is used if no traveler profile is present. The user-defined mean car speed is used if the speed data is missing from the current road of the car. The user-defined public transportation speed is used if two successor pairs $\langle stop, time \rangle$ give an inconsistent speed due to errors in the data².

Units transformation: All the data that are expressed in function of time (e.g. the visit times of the public transportation vehicles) have to be expressed in terms of simulation ticks. Since the original time data are expressed in terms of date, they have to be transformed. In addition, all the speeds are originally defined in terms of Km/h. They have to be transformed into meters/tick. The transformation is quite straightforward.

2.2.2 Multi-Agent System

The multi-agent system is composed of a planning service and three types of agents: traveler agents, car agents and public transportation vehicle agents.

2.2.2.1 Planning service

The planning service has the responsibility of calculating the best road itinerary for the car agents and the best multimodal itinerary for the traveler agents. The service always considers the newest known status of the networks. An itinerary is made of consecutive edges (either in the road network, pedestrian network or the public transportation network) together with their corresponding visit times.

2.2.2.2 Agents movements

Each agent knows the coordinates that he has to sequentially move to, following his computed path. Depending on the type of agent and his corresponding speed, he can move for only a certain distance at each simulated tick of time. At each tick, the agent iteratively checks if he can move from his current coordinate to the next in his list. If not, he calculates the intermediate coordinate that corresponds with the remaining distance that he is

²E.g. infinite speed due to two identical visit times of two successive stops.

allowed to travel. In the next tick, the agent can travel the remaining distance to the next coordinate.

2.2.2.3 Car agents

When created, a car agent has an origin and a destination. If travel patterns exist for the considered network, the origin and destination of each car agent are chosen such as the origins and destinations of all agents are proportional to the pattern. If no travel pattern exists, the origin and destination of the car agent are chosen randomly. The agent then asks the planning service for the best itinerary between his origin and his destination. The car agent follows the received itinerary as explained above. At each simulation tick, the car agent checks if he has reached his destination. If so, he leaves the simulation.

2.2.2.4 Public transportation vehicle agents

The vehicle agents represent public collective transportation vehicles that have a defined itinerary and a timetable. Vehicles infer their paths from the timetable input data. Each vehicle agent moves at each simulation time tick with the allowed distance following his current speed. At each visited stop, the onboard passengers who have a transfer at this stop leave the vehicle and the passengers waiting for this vehicle at the stop take the vehicle. While moving, the vehicle agent moves his onboard passengers to the same coordinates at the same time. At each simulation tick, the vehicle agent checks if he has reached his destination (the last stop in his itinerary). If so, he is removed from the simulation.

2.2.2.5 Traveler agents

As for car agents, the origin and destination of the traveler agent are either inferred from travel patterns if they exist, or are chosen randomly. When they are not walking, traveler agents do not travel on their own, but take public transportation vehicles, which are responsible of their movements. The traveler agent alternates between walking, waiting for a public transportation vehicle and being onboard a vehicle.

These are the main building blocks of the SM4T simulation platform. In the following, we describe an application that demonstrates the use of the simulation on an interesting and highly topical application: the impact of real-time information on transit networks.

2.2.3 Application: Impact of Passenger Real-Time Information

The prediction of the impact of traveler information on transit networks is of great importance to operators. With the widespread use of personalized real-time information sources, the status of the networks depends heavily on individual travelers reactions to the received information. The effects of information on passengers travel and transportation networks have been generally investigated in the literature using either surveys or simulations. Works that use surveys investigate the passenger's experience feedback (e.g. [Brakewood et al., 2014], [Dzikan and Kottenhoff, 2007] and [Schweiger, 2003]) and generally concern small scales

(in terms of number of passengers and network size). This is due to the difficulty and the high cost of studies with a big number of passengers. The findings of these studies are more reliable because they are based on actual observations, but they are not easily generalizable because they are limited in time and space and concern a small number of observations. For these reasons, a large number of works have chosen simulation as an evaluation tool. Simulation is able to extend the experimentation field and makes it possible to test a big number of scenarios. Scenarios can integrate different levels of information and different network conditions. In the following, we present the simulation proposals for evaluating the effects of passenger information on transit networks.

In [Coppola and Rosati, 2009], the authors simulate the passenger’s decision process after the reception of real-time information. This process is in charge of the computation and the choice of the passenger’s itinerary and its execution. In [Cats et al., 2011], the passenger decision model is divided into two sub-models: the generator of the set of choices and the itinerary choice model. In [Hickman and Wilson, 1995], the authors also use a dynamic itinerary choice model to manage the passenger decision process. In [Fonzone and Schmöcker, 2014], the authors propose a passengers information model that manages the information provision to passengers in the simulation model. In these approaches, the personalized information resulting from the use of smartphones is not directly considered. The equipped passengers are represented as entities with a “high access level” to real-time information. Indeed, different levels of information are generally used in the experimentations as in [Estrada et al., 2015] where 6 access levels to real-time information are studied. Results reported in [Coppola and Rosati, 2009] and in [Cats et al., 2011] show that providing more comprehensive real-time information may lead to path choice shifts and time savings. They recommend to provide real-time information at the station level to enable more informed decisions and gains in travel time.

Individual real-time information on smartphones is not explicitly considered in the literature. Indeed, the real-time information evaluated in the studied works concern only localized information (screen displays and voice messages in the stops and stations). This kind of information differs from personalized information, accessible via smartphones, that assist passengers during their movements by providing specific information about their itinerary. This kind of information represents an increasingly important element in the advanced travelers information systems and plays a significant role in modern transportation systems. In the work presented in this section, we consider the equipped passengers as a distinct category, with a specific behavior. We analyze the results according to the two types of passengers (equipped and non-equipped).

Results reported in the literature also concern small networks or only some couples of origins-destinations. The results in the work presented in this section concern the data of a big network with thousands of edges. We also consider thousands of passengers representing up to 25% of the real population of the considered network.

With SM4T, it is possible to integrate traveler information and to test the impact of real-time information on the status of the network. In the following, we describe how this integration is achieved.

2.2.3.1 Multi-agent system

Figure 2.5 presents the new definition of the multi-agent system for this application. It describes the agents that compose it, the environment and the interaction between agents. We have removed the car agents and transformed the traveler agents to two types of passenger agents. The passenger agents and the (public transportation) vehicle agents are the agents that move on the network. There are two kind of passenger agents: connected passenger agents (to a real-time information source) and non-connected passenger agents. Non-connected passenger agents rely on a personal and static view of the network. The local information agents are responsible for sharing disturbances information in the network stops. The simulation platform is responsible for the planning and the monitoring of the connected passengers trips based on a spatiotemporal network, presented below.

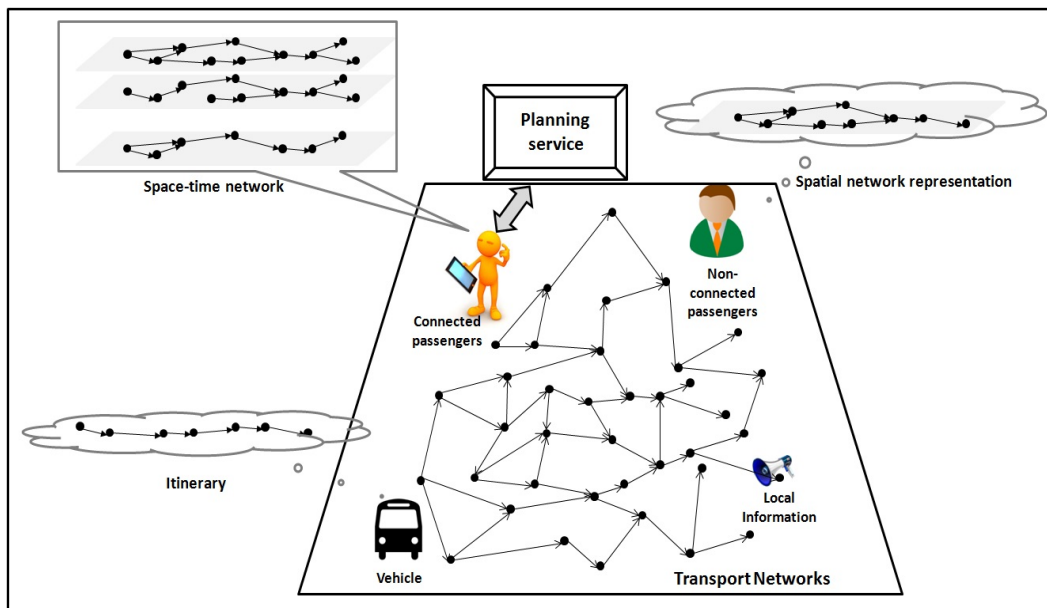


Figure 2.5: Multi-agent system model

2.2.3.2 Space-time representation of the environment

The notification about disturbances in public transportation networks and the replanning of routes of passengers is a complex task. Indeed, the changes in the itineraries and travel times are exogenous to the agents and these latter do not know *a priori* where and when they may occur. On the one hand, we have the real-time information providers that produce dynamic information. In the other hand, we have passenger agents who are potentially interested in this information. The broadcast of dynamic information to all passenger agents is a simple and intuitive method for the matching of passengers and information providers, but this method is expensive because it generates unnecessary processing and uses unnecessary bandwidth.

Several approaches exist in the literature for matching agents who don't know each other *a priori* (e.g. middle-agents [Wong and Sycara, 2000] and recommendation systems

[Vercouter, 2001]). We adopt an environment-centered approach, which focuses on the shared data and allows for the selection of relevant information by the agents without having to know or to maintain knowledge about the emitters of these data. Our environment-based approach is based on a representation of the environment in the form of a space-time network. This representation will also be used in the dynamic vehicle routing problem described in chapter 3. In the current application, the multi-agent space-time environment represents the state of the public transportation network through time. This environment is the main interlocutor for connected passenger agents and is active, in the sense that it stores information and triggers reaction when some events occur.

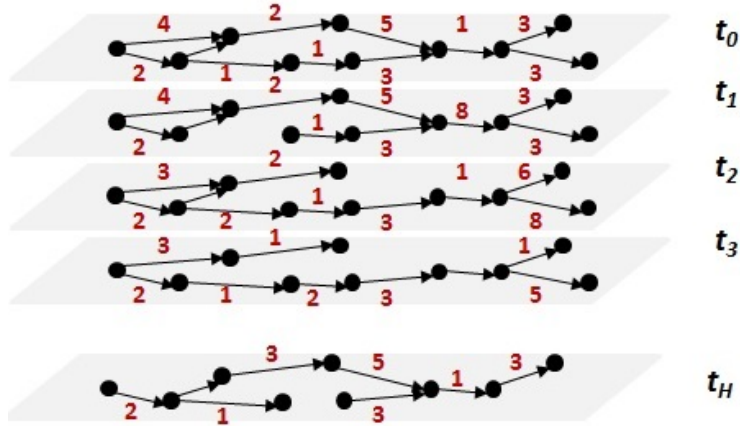


Figure 2.6: Spatiotemporal graph

Consider the transportation network $G = (V, E)$, with a node set $V = \{(v_i)\}$, $i = \{0, \dots, N\}$ and an edge set $E = \{(v_i, v_j) | v_i \in V, v_j \in V, v_i \neq v_j\}$. Let D and T be two matrices of costs, $D = \{(d_{ij})\}$ and $T = \{(t_{ij})\}$, of dimensions $N \times N$ (the link (v_i, v_j) has d_{ij} as distance and t_{ij} as travel time). The space-time representation of the multi-agent environment is made of H copies of G , where H depends on the considered timeframe for the application³: $G(t) = (V(t), E(t))$, with $V(t)$ the set of nodes at time t and $E(t)$ the set of directed links at time t with $0 \leq t \leq H$ (cf. Figure 2.6). The cost matrices become $D(t)$ and $T(t)$ to be time-related as well. At each progression of time, the t_0 version of the space-time network is deleted, t_1 becomes t_0 , $\forall n \in \{1, \dots, H\}$ t_n becomes t_{n-1} and t_H is created.

This structure is generic and may have a different semantics depending on the considered application. The temporal copies of G are generally not identical. Indeed, we can have different travel times between two copies to reflect the traffic state. Edges and nodes can be present in a copy and absent in another copy to reflect the expansion of a crisis for instance. Edges can be absent to reflect vehicles's timetables and disturbance situations, as in the application of this section.

An agent who wish to be informed by the only changes occurring on a node v of the network during a period ranging from t_1 to t_2 has to subscribe to the space-time nodes $\{(v, t_1), \dots, (v, t_2)\}$. As we show in the following, with this representation of the environ-

³For instance, if we consider a timeframe of two hours, and a temporal discretization into one minute, $H = 120$.

ment, every dynamic modification of the transportation supply is directly reported to the only concerned agents, which avoids massive dissemination of information to all the system agents.

In the following, we describe the use of this structure to inject disturbances and disseminate general and personalized information to both connected and non-connected passengers.



Edge	Time ↑	$\langle \textit{Vehicle}, \langle \textit{agents}, \dots \rangle \rangle$
	t_1	$\langle \textit{Vehicle}_1, \langle \textit{agent}_1, \textit{agent}_3, \dots \rangle \rangle$
	t_1	$\langle \textit{Vehicle}_2, \langle \textit{agent}_4, \textit{agent}_8, \dots \rangle \rangle$
	t_2	...


Table 2.1: Implementation of the space-time graph in the simulator

In the simulator, we have implemented the space-time network in the form of a map (cf. Table. 2.1). The keys to the map are the edges of the spatial graph. The corresponding values are sorted lists of times related to start times of vehicles from the origin node of the edge. To each time value we associate a pair $\langle \textit{vehicle}, \langle \textit{agents} \rangle \rangle$, corresponding to the concerned vehicle and all the agents who have subscribed to the corresponding space-time edge. When a vehicle's timetable changes, the concerned edges are identified. Then the map is requested with these edges as keys, and the $\langle \textit{vehicle}, \langle \textit{agents}, \dots \rangle \rangle$ pairs corresponding to the old visit times are retrieved from the map. Finally, all the agents in the lists are notified about the changes. The agents that subscribe to space-time edges in the present application are local information agents and connected passenger agents, as described in the following.

2.2.3.3 Local information agents

Local information agents represent devices that broadcast traffic information on screens or through voice announcements at the stops. The information broadcasted by the local information agents concern events generally occurring elsewhere in the network. To get this information, local information agents subscribe to the space-time edges of the public transportation lines that they are interested in. In our experiments, the local information agents subscribe to the lines that pass by their stop. Indeed, we make the realistic choice to broadcast on stops the only information that concerns the lines passing by them. Otherwise, the amount of information in the stops would be too difficult to display on the screens or to enounce through vocal messages. It would also lead to the oversaturation adverse effect described in the literature. When dealing with different networks, the choice of the subscription strategy can be adapted in consequence. Only passenger agents present in a stop can perceive the locally broadcasted information.

2.2.3.4 Connected passengers behavior

Passengers that are connected to a real-time information source receive their calculated path from the planning service. When they receive the computed path, connected passengers interact with the space-time network. Indeed, to be aware of the only traffic events that

concern him, a passenger agent subscribes to the edges of the space-time that form his itinerary (that he received from the planning service). When the path of the passenger is impacted by a disturbance, the information is received from the space-time network. The planning process is executed again, and the planning service will base his calculation on the new configuration and the new status of the network.

2.2.3.5 Non-connected passengers behavior

Passengers that are not connected to a real-time information source ground their path calculation on a personal view of the network. This view is made of the network topology and theoretical timetables. They calculate their path based on this view. They follow this path and wait at the programmed stops. They deviate from their path in two cases. Either they are actually caught in a disturbance; or they are informed about a disturbance, via announcements in the stop. If they receive such an alert, they integrate the modifications in their mental view of the transportation network, and calculate a new path using that view.

2.2.3.6 Injecting disturbances

To verify the effects of real-time information on passengers, both with personalized information and with local information, we need to inject disturbances. Indeed, information is most important in case of disturbances, when passengers need to find new routes and need to be oriented on the network.

The space-time network status is modified to integrate disturbances. To inject a delay, we erase the space-time edge corresponding to the old visit time and create a new space-time edge with the up-to-date visit time. To model breakdowns and disconnections, we delete all the concerned space-time edges. Hence, when the timetable of a vehicle is modified, all the concerned agents are notified of the change in their plan. These agents are unsubscribed of these space-time edges, and the planning service calculates new itineraries for them, which leads to new subscriptions.

2.2.4 Temporal model of the simulation

The concurrent execution of agents, especially with a parallel scheduler, necessitates the synchronization of their access to the transportation environment, to avoid incoherent states of the network due to simultaneous modification. Thus, the calls to the movement methods on the network are synchronized: passenger and vehicle agents, once they take their movement decision, move sequentially (while in the same time tick). Other considerations concerning the execution order of the agents have also to be taken into account. Indeed, the main interactions between agents are performed between:

- passenger agents and vehicle agents at the stops;
- passenger agents and local information agents at the stops;
- connected passenger agents and the planning service;

- connected passenger agents and the space-time networks.

The order in which agents are activated by the scheduler might alter the coherent outcome of these interactions. On the one side, since the vehicles arrive at the stops and look for passenger agents who plan to take them, there is a risk that a passenger agent who wants to take a given vehicle is not yet at the stop, not because he is late, but because the scheduler has not activated him yet for the current simulation time tick. For this reason, passenger agents are activated by the scheduler first, before the vehicle agents. On the other side, incoherences in passengers reactions to disturbances might also occur. For instance, we could have passengers who react immediately to disturbances, while others only react at the next time tick, because the disturbance injection is performed in the middle of agents execution for a simulation time tick. To avoid this situation, disturbances and their impact on the space-time network and on the local information agents are performed in priority before activating passenger and vehicle agents.

2.2.5 Experiments and Results

2.2.5.1 Setup

The experiments are executed with the data of the city of Toulouse in France. We choose this French city because we have detailed data about its network and a description of the travel patterns of the region [Zargayouna et al., 2014]. The public transportation network of Toulouse is composed of 80 lines, 359 itineraries and 3,887 edges. The multi-agent system is made of 18,180 vehicles and from 5,000 to 30,000 passengers. We define the number of ticks per simulation to 5,000 for an operation day from 6 am to 2 am. Every simulated tick corresponds then to approximately 14 seconds. The origins and destinations of passengers are chosen coherently with travel patterns of the region (our origins-destinations generation method is described in [Ksontini et al., 2016]).

Previous works have shown that traveler information has little impact in case of small disturbances. For this reason, we decide to use serious disturbances instead in the form of complete disconnection of edges. In every simulation, we have generated 5 random edge disconnections on the network during the whole simulation (one disconnection every 233 real time minutes approximately). Every disconnection lasts 250 simulated ticks (slightly less than one hour in real time). Due to edge disconnections, some passengers cannot find an itinerary to their destination anymore, because edges disconnection impacts network connectivity. In the following results, these passengers are not considered⁴. Disturbances are random but concern only a certain number of edges which we have considered as significant to disconnect: the edges through which pass at least 5 different itineraries. The 5 randomly disconnected edges are chosen between 21 candidate edges that satisfy this requirement.

2.2.5.2 Scenarios

We consider six different information level scenarios (cf. table 2.2) and each one is executed 25 times. The first scenario is “the reference configuration” (to which we compare all the

⁴The ratio of passengers without itinerary is stable, around 5% in all the simulations.

Scenarios	Local Information	Personal Information
Reference	No	No
1	Yes	No
2	Yes	20%
3	Yes	50%
4	Yes	80%
5	Yes	100%

Table 2.2: Information level in each scenario

others) where no up-to-date information are provided to the passengers, neither local nor personalized. They only have the static description of the network and timetables. In the second scenario, only local information are given. The new travel times are available for the only passengers that are present in the considered stop. We do not consider any connected passenger in this scenario. In the remaining scenarios (3, 4, 5 and 6), local information are provided to passengers at the stops, and personal information is only available for the connected passengers. We consider 20%, 50%, 80% then 100% of connected passengers respectively in these scenarios. We report the average travel times for the passengers.

Every scenario is executed 25 times. Indeed, for each information level scenario, we consider increasing number of travelers : 1,000, 5,000, 10,000, 20,000 and 30,000 passengers, and every configuration (scenario and number of travelers) is executed 5 times to verify that the simulations are unbiased and results are reliable. The increasing number of simulated travelers (1000, 5000, 10000, 20000 and 30000) can be seen as an increasing precision for the same simulation. That means that, with 1,000 simulated travelers for instance, each traveler agent represents around 100 human travelers, with 5,000 simulated travelers, each traveler agent represents 20 human travelers, and so on⁵. We report the average results and the observed standard deviations.

In case of disturbances, vehicle capacity limitation is one of the main causes of passengers delays. Thus, vehicle capacities have to be chosen carefully to avoid simulation bias. Indeed, if vehicle capacity is too big, the effect of disturbances might be underestimated. On the contrary, if vehicles capacity is too small, it can artificially amplify the impact of disturbances by introducing a big waiting time in the stops. Since we simulate a fraction of the real number of passengers, we adapt the bus capacity proportionally to this fraction. We base our calculation on the 2010 annual data of the bus network of Tisseo [Tisseo, 2011]. Our goal is to have an equivalent ratio between the daily passengers number, the number of available places in the vehicles (seating and standing) and the daily number of passengers. The result of this adaptation is reported in the table 2.3.

Passengers number	Vehicles capacity
1,000	1
5,000	4
10,000	8
20,000	17
30,000	25

Table 2.3: The bus capacities in the simulator

⁵The average number of travelers in the considered region is 100,000 human travelers

2.2.5.3 Results

	1,000	5,000	10,000	20,000	30,000
0%	0.45% (0.03%)	1.45% (0.1%)	7.56% (0.25%)	5.43% (0.52%)	3.44% (0.33%)
20%	3.56% (0.23%)	8.25% (0.57%)	7.97% (0.26%)	8.43% (0.8%)	10.57% (1%)
50%	5.15% (0.33%)	12.21% (0.84%)	15.64% (0.52%)	11.62% (1.1%)	14.40% (1.36%)
80%	8.99% (0.58%)	11.63% (0.8%)	16.3% (0.54%)	10.43% (0.99%)	12.35% (1.17%)
100%	4.95% (0.32%)	10.24% (0.71%)	12.99% (0.43%)	8.2% (0.79%)	8.82% (0.84%)

Table 2.4: Synthesis of travel time improvement (Average (Standard deviation))

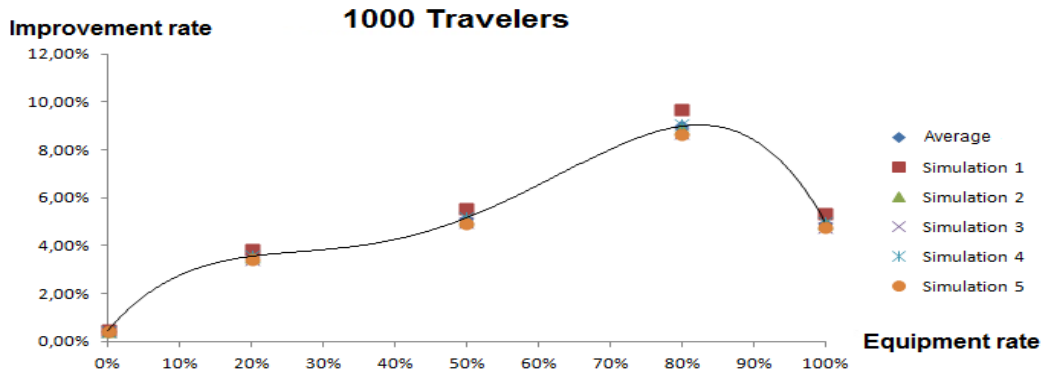


Figure 2.7: Travel time improvements (1,000 passengers)

The results are reported in Table 2.4 in which we provide the average travel times improvements and the observed standard deviations. The same results are reported in the figures 2.7, 2.8, 2.9, 2.10 and 2.11. The curves represent the improvement rate in travel times in comparison with the reference configuration (with no information). It means that every point in the curve represents the improvement rate of the concerned simulation w.r.t the scenario with no information at all. Recall that the scenario with 0% connected passengers is the scenario where only local information is provided, which is different from the reference configuration in which there is no information at all. As we said earlier, we execute several simulations of every (equipment rate, number of passengers) pair to verify that the curve shape is not due to the origin-destination choices or from the difference in the injected disturbances, which are stochastic. We add to the figures a trend curve associated to the average values to facilitate the interpretation. As we see it in the table, the standard deviations are very low and do not question the trends of the curves.

Compared to the scenario without information, all scenarios improve the travel times. The lowest improvement (0.45%) is related to the scenario with 1,000 passengers and 0% of connected passengers and the highest one (16.3%) is related to the scenario with 10,000 passengers and 80% of connected passengers. It is clear that the reference configuration (without information) provides the worst average travel time, since all the improvement

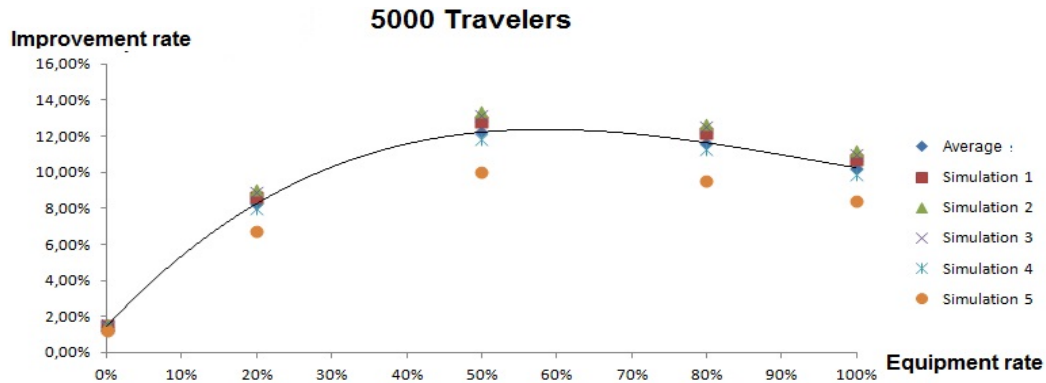


Figure 2.8: Travel time improvements (5,000 passengers)

values are positive. Indeed, in this scenario passengers have the information about a disturbance when they are already blocked somewhere in the network. As a consequence, they look for alternative paths and use concurrently the rest of the transportation network. By doing so, they might miss vehicles because of their capacity constraints. They also could get stuck in another disturbance. The scenario 2, with local information and no personalized information, provides better results than the reference configuration. In this scenario, the passenger does not have necessarily to be present at the stops impacted by the disturbance to know that something is wrong. The passenger has the information when he visits a stop where local information is provided. However, this information is provided to him with a certain delay. Furthermore, since the path of the passenger is grounded on his partial vision of the network, he can head towards another disturbance.

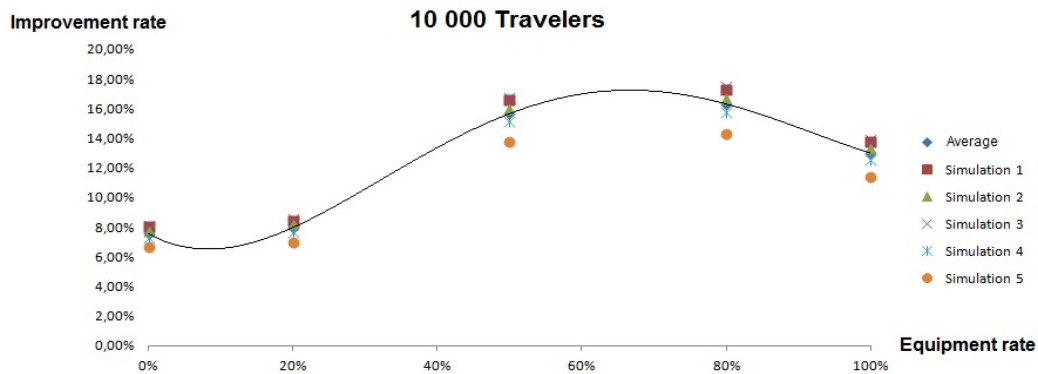


Figure 2.9: Travel time improvements (10,000 passengers)

The following scenarios, which integrate gradually more and more connected travelers, provide better results until 50% of connected passengers for scenarios with 5,000, 20,000 and 30,000 passengers and until 80% for scenarios with 1,000 and 10,000 passengers. In these scenarios, the alternative paths are immediately computed and provided to the passengers. These paths are also based on the latest status of the network. As a consequence, the passengers avoid the disturbed zone early and do not risk to head towards another disturbance. The connected passenger does not have to be present at a stop where local information is provided, he receives a notification and a new plan immediately. However, this

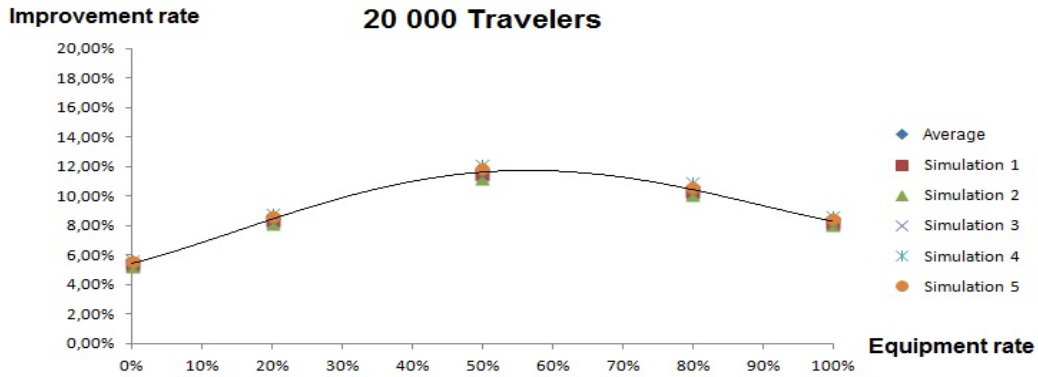


Figure 2.10: Travel time improvements (20,000 passengers)

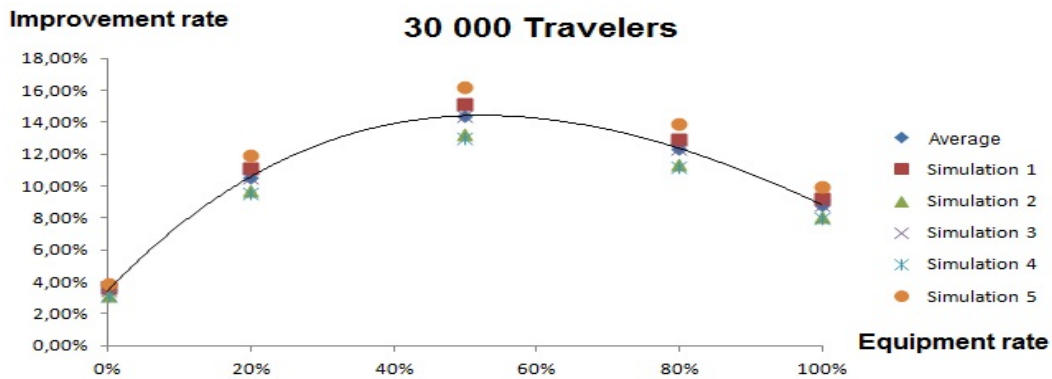


Figure 2.11: Travel time improvements (30,000 passengers)

improvement is maximal at 50% of connected passengers for some scenarios and at 80% for others. After these thresholds, the improvement rate becomes lower with more connected passengers. In the scenarios where the majority of passengers is connected, in case of disturbances, the concerned passengers receive simultaneously new up-to-date plans. In this case, when they apply their new plans, they are faced with vehicles capacity constraints and will see their average travel times increase.

Starting from scenario 4, the results start to become worse. Indeed, with 80% of connected passengers for scenario 5 and 100% of connected passengers for scenario 5, most of the passengers get personal real-time information about disturbances in real-time and new up-to-date plans are generated simultaneously. The consequence is that the passengers will face capacity constraints of the vehicles and will see their mean travel times increase. The results are 15% worse for scenario 4 compared to scenario 3, and 23% worse for scenario 5 compared to scenario 4.

Normally, the provision of real-time personalized information is supposed to have a positive impact, especially for connected passengers. To verify this impact on the only connected passengers, we provide the same results, but this time we differentiate improvement rates according to the passenger agent type (connected versus non-connected). The two example results are reported in the figures 2.12 and 2.13. All the results follow the same

trend. We report in these figures the observed average values.

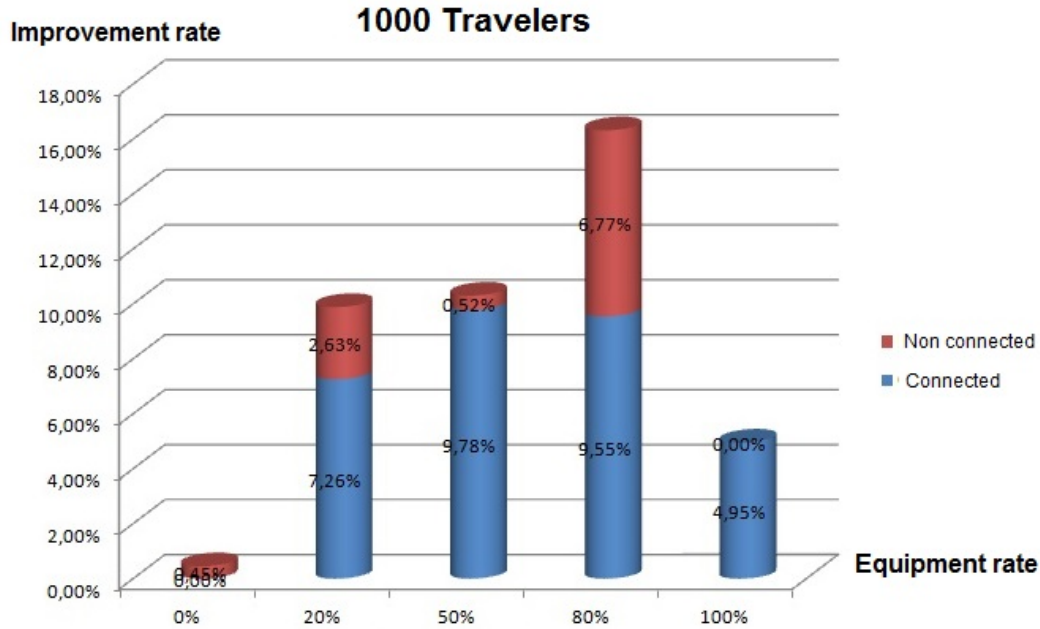


Figure 2.12: Travel time improvements (connected passengers vs. non-connected passengers) with 1,000 passengers

Every figure represents the improvements in travel times for a concerned number of passengers. The red part of the vertical bars of each figure represent the improvements for the non-connected passengers and the blue part represents the improvements for the connected passengers. Thus, for the scenarios with 0% of connected passengers, the improvement is ensured exclusively by the non-connected passengers and for the scenarios with 100% of connected passengers, the improvement is totally ensured by the connected passengers.

Results show that the improvement in travel times for the connected passengers follow the same trend that the curves in the figures 2.7 to 2.11, i.e. a growing improvement until a certain threshold before becoming lower. However, this time, the maximum improvement is reached systematically at 50% of connected passengers. The improvement oversupply for the scenarios with 1,000 and 10,000 passengers and 80% of connected passengers is actually provided by the non-connected passengers.

The improvement rate for the non-connected passengers is quite stable between the scenarios with the same number of passengers. Thus, it is between 0.45% and 6.8% for 1,000 passengers, between 1.4% and 5.7% for 5,000 passengers, between 5.5% and 7.6% for 10,000 passengers, between 0.1% and 5.9% for 20,000 passengers, and between 2.2% and 8% for 30,000 passengers. Their curves don't follow a particular trend because their behavior is not directly impacted with the increase of the information level of other agents. Probably, the oscillation in improvement rates comes from the processing (nondeterministic) order of the passenger agents when vehicles arrive at stops. If the number of passengers in stops exceeds the vehicle capacity, no type of passengers is privileged to board. That is probably what gives this variations of the non-connected passengers' travel times improvements.

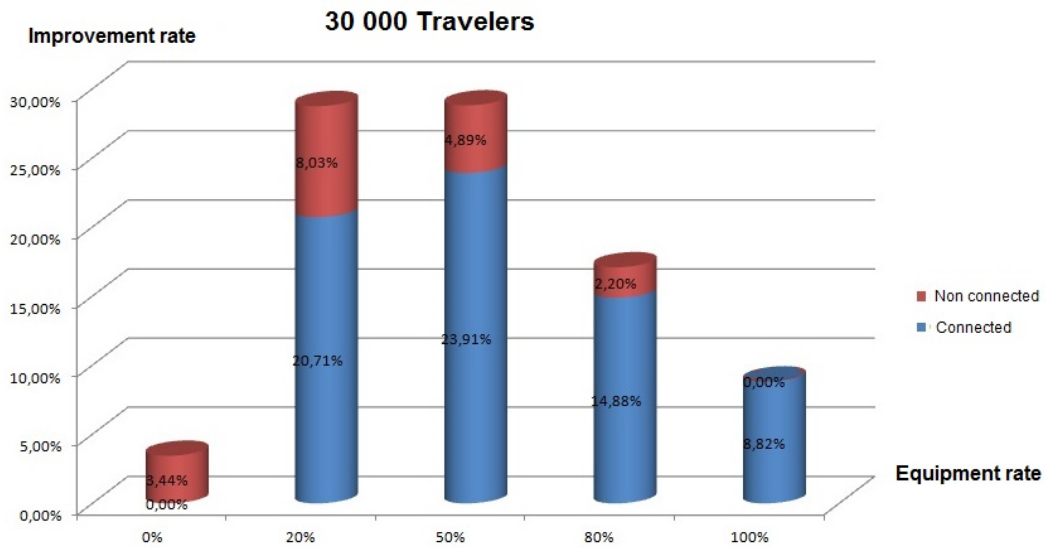


Figure 2.13: Travel time improvements (connected passengers vs. non-connected passengers) with 30,000 passengers

2.2.5.4 Computational complexity

The average execution times of our simulations are reported in Figure. 2.14. and the average memory usage are reported in Figure. 2.15. Both Computation times and memory usage are correlated with the number of simulated travelers. Indeed, we have executed several simulations with vehicles only, and they were always very fast and use little memory. Based on these results, we believe that we should further optimize the traveler agents structure and behavior, notably regarding the shortest paths computation.

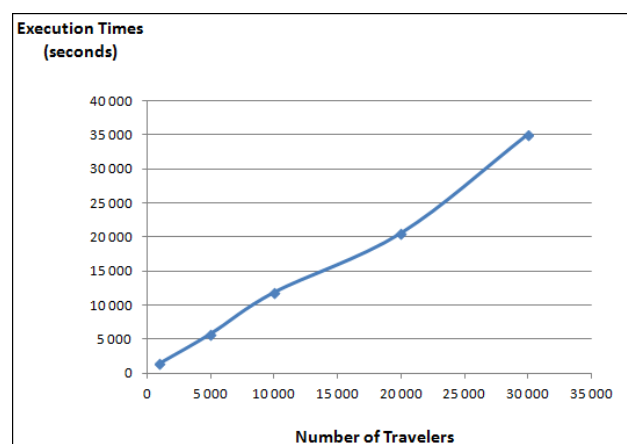


Figure 2.14: Average execution times

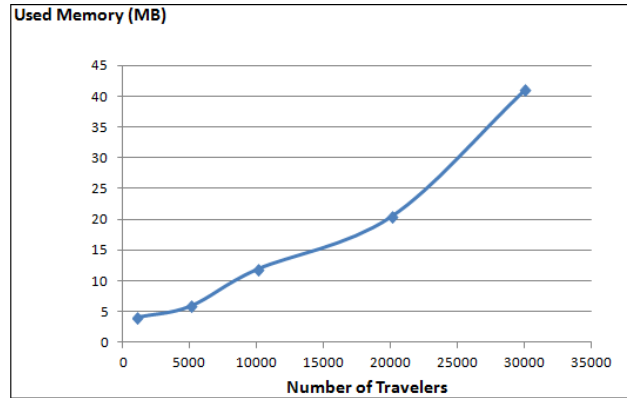


Figure 2.15: Average memory usage

2.2.5.5 Discussion

The most important result reported in this section concerns the lower improvement in the passengers travel times starting from a certain ratio of connected passengers. This result may seem paradoxical, since it is intuitively expected that the average travel times continuously improve when the ratio of connected passengers increases. In reality, this result is coherent with previous theoretical studies performed by [Pereira et al., 2015] for road networks. Indeed, the phenomenon observed in this section could be seen as a materialization of concentration and overreaction. Concentration is the consequence of a uniform perception of the network status by the passengers. The more connected passengers we have, the more uniform their perception will be. In our system, this uniform perception is stored in the space-time network, and indeed at 100% of equipment rate, all passengers have the exact same view of the network status. Overreaction is similar to concentration but is related to received traffic information, which make a fraction of the passengers transfer the congestion from one region to another. The more connected passengers in the system, the more substantial is the fraction that would follow the same paths and transfer the congestion. Congestion is mainly materialized by passengers who cannot take a vehicle because of capacity constraints. However, this relatively negative impact of information provision starts at very high threshold of connected passengers. Below that threshold, as it is the case in nowadays transportation systems, providing real-time personalized information does have an absolutely positive impact on the network status.

2.3 Simulations Distribution

Even if SM4T uses a parallel scheduler, we faced a scale limit for our simulation, around 130,000 simulated agents (of all type), beyond which the simulations became too slow. Our objective was to go beyond these numbers and to be able to execute simulations on higher scales and to simulate, say, millions of passengers and drivers on all available modes, including new mobility services. We desired to provide solutions to all agent-based traffic simulations, and not only to our own platform. To do so, we have defined a model that represents the existing simulations, and then proposed two methods to distribute

them. The objective is to provide guidelines to improve the scalability of agent-based traffic simulations (see [Rana and Stout, 2000] for scalability definition in multi-agent systems). This contribution is presented in the following sections.

2.3.1 Generic Agent-Based Traffic Simulations

In the following, we propose to study distribution methods for agent-based traffic simulations. We define two generic agent-based simulation models, representing the main types of agent-based simulations of the literature. The first model is called “macroscopic”, in the sense that travelers do not interact directly but use a fundamental diagram of traffic flow to continuously compute their speeds. This is the choice performed for instance in SM4T, but also in these works [Meignan et al., 2007, Cajias et al., 2011]. The second model is called “microscopic”, in which travelers interact with their neighbors to adapt their speeds to their surrounding environment. This is the most common choice performed in the literature for agent-based simulations (e.g. [Behrisch et al., 2011, Maciejewski and Nagel, 2012, Champion et al., 1999]). We study two distribution patterns (agent-based and environment-based) applied to these two simulation models. The results show that agent-based distribution is more efficient with macroscopic simulations while environment-based distribution is more efficient with microscopic simulations. We also propose a load-balancing mechanism for the environment-based distribution and show that, with the right parameters definition, it has a positive impact on the distribution method.

2.3.1.1 The multi-agent system

A common base is shared by both microscopic and macroscopic simulations. Both systems are composed of a dynamic set of agents representing travelers, interacting with a transportation network environment. We model the transportation network in which the travelers evolve with a graph $G(V, E)$, where $E = \{e_1, \dots, e_n\}$ is a set of edges representing the roads and $V = \{v_1, \dots, v_n\}$ is a set of vertices representing the intersections. The agents, representing the travelers, move on this network from their origins to their destinations, trying to minimize their travel costs. Figure 2.16 describes the steps followed by a traffic simulation. First, the simulation platform loads the parameters (simulation duration, number of generated agents, etc.) and the description of the network. Then, it creates the logical graph from the network representation, to enable shortest paths calculation and agents movements. The scheduler, which is responsible of agents execution, ranges over the agents and asks them to execute one step of simulation (either to compute a shortest path or to move from one position to another). When an agent reaches his destination, he comes back to his origin point (to keep a constant number of agents in the simulation). When all agents have executed their step instructions for one tick of simulation time, the scheduler increments the simulation tick counter (*step++*), and the process starts again. When the simulation duration is reached, the simulation stops and the results are collected.

The agents execute a step method each time they are activated by the scheduler. When created, an agent has an origin node o and a destination node d . The first action that he executes when created and activated by the scheduler is to compute an A^* shortest path algorithm between o and d . The shortest path is performed on the graph G , which edges costs are dynamic, depending on the current traffic. When he has a current path, the agent

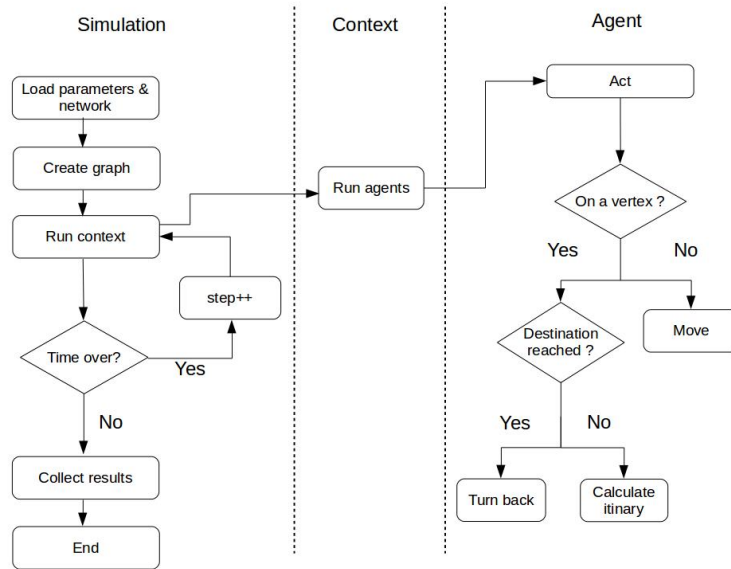


Figure 2.16: Steps of a simulation

moves according to it. At each tick, he moves the allowed distance following his current speed. The speed of the agent is computed following the simulation model (microscopic or macroscopic), described in the following sections. Each time he reaches a node, the agent recomputes a shortest path, to check if the current traffic conditions have evolved and if a new shortest path has become available⁶.

These are the main components of the model that are common to both types of simulations. In the following sections, we present the specific methods for the two simulations, namely macroscopic and microscopic.

2.3.1.2 Macroscopic simulation model

In the macroscopic simulation model, the speed of an agent on an edge is computed following the number of other agents traveling on the same edge. To this end, the fundamental diagram of traffic flow presented in section 2.2 is used. With the distribution objective that we have, the locations of the agents and their interaction patterns are the most important. In the macroscopic model, the agents do not interact directly. The speed of the agent is computed with an interaction between the agent and the edge. The latter knows the number of agents currently using it and provides the right speed of the agent, based on the fundamental diagram.

⁶The graph being directed, turnarounds are only possible at nodes and there is no need for the agent to execute a shortest path while traveling on an edge.

2.3.1.3 Microscopic simulation model

In the microscopic simulation model, the speed of an agent on an edge is computed following the position and speed of the vehicles surrounding him. In this model, the information available to each agent is local. The agents perceive a part of their environment, delimited by their area of interest and then calculate their next move given the perceived information. This implies many local communications between the agents, because their actions will be conditioned by the actions of the other agents present in their area of interest [Mandiau et al., 2008, Doniec et al., 2008]. This model is generally based on a car-following model. Following this model, each vehicle computes, at each simulation tick, his speed in function of the relative position of the vehicle preceding him, their current speed, acceleration, distance and his reaction time. If there is no vehicle preceding the agent, he accelerates until he reaches the speed limit of its edge.

In this model, we cannot rely on the fundamental diagram of traffic to continuously have the current travel times on the edges. Instead, each agent registers his experienced travel time when he reaches the end of the edge, as proposed by the authors in [Wahle et al., 2002]. The shortest path calculation is based on the graph where the travel times costs are fed by the agents following this procedure.

In contrast with the macroscopic model, the agents in the microscopic model do interact directly. The speed of an agent is computed with a direct interaction between him and the agents before him. This difference between the two models conditions the choice of the relevant distribution pattern for a considered simulation type. The distribution patterns are described in the following section.

2.3.2 Distribution Methods

We define two patterns to distribute traffic simulations. The patterns are the same than those identified by the authors in [Rihawi et al., 2014] for general-purpose situated multi-agent simulations, and we believe that they present two representative distribution patterns for this kind of simulations. The first pattern (called agent-based distribution) is the distribution model used by [Barceló et al., 1998]. It consists in the duplication of the transportation environment on all processing units, and the equal dispatching of the agents on each one. As a consequence, agents stay on the same unit during all the simulation. The second pattern is the mostly used pattern in the literature. It consists in partitioning the transportation environment and dispatching each subpart of the environment - and all the agents in it - on each processing unit. In this pattern, agents might have to move from one unit to another if their itinerary crosses several subparts of the transportation environment.

2.3.2.1 Agent-based distribution

The first distribution pattern is agent-based, since it clusters the set of agents in k equal parts (with k the number of available processing units), and distributes each subset on a unit and executes the simulation (cf. Figure 2.17). The transportation network is duplicated on each unit. This method ensures that each unit has the same amount of work at any time of the simulation. In the following, we describe the use of this pattern for both simulation

models that we have defined.

Macroscopic simulation with agent-based distribution: In a macroscopic simulation, when it is distributed following the agent-based distribution pattern, every unit continuously (at each simulation tick) informs the other units of its network state. This is because they do not have a complete view of the network state, since only a part of the agents evolves in the unit. Thus, they send the list of edges together with the number of agents currently on them. Each unit is then able to compute the shortest paths and the relevant speed of the agents (using the fundamental diagram of traffic).

Microscopic simulation with agent-based distribution: When distributed following the agent-based distribution pattern, the agents in a microscopic simulation do not use a fundamental diagram of traffic to compute their speeds. Instead, they need to know the state of the agents preceding them. To do so, they interrogate the edges in the other units to know if there are agents preceding them, and if it is the case, to know their states. Moreover, the units exchange the current travel times (provided by the agents as explained earlier), to compute the shortest paths of the agents.

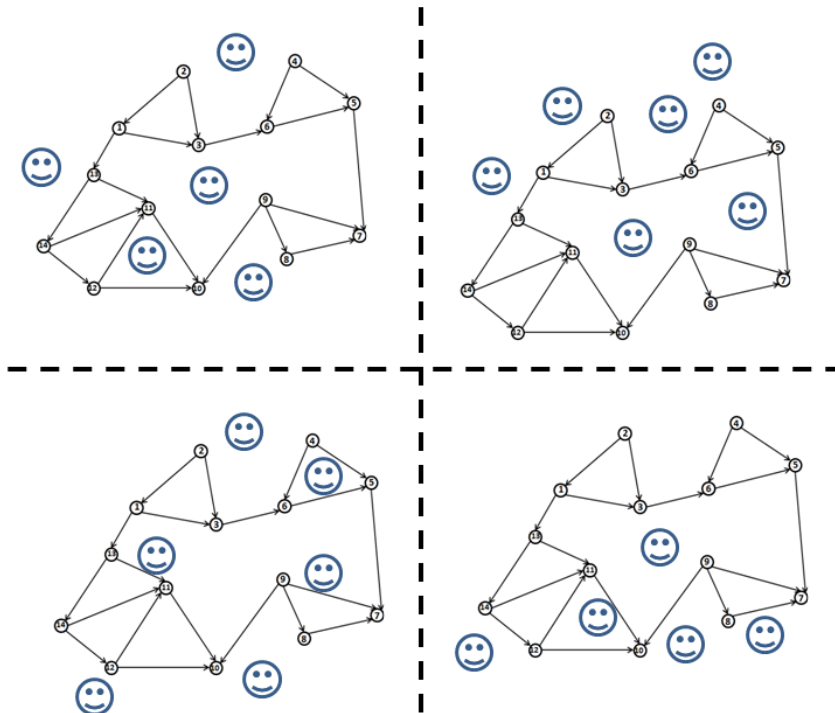


Figure 2.17: Agent-based distribution

2.3.2.2 Environment-based distribution

The second approach to distribute traffic simulations is environment-based. It tries to keep on the same unit the agents who are geographically close in the transportation network (cf.

Figure 2.18). To this end, the network is split in k parts (with k the number of available processing units), and distributed on the different units. Each unit is only aware of what is happening on the part of the graph that it is managing, and the agents that are in the same area are now likely to be on the same unit. If an agent reaches a part of the network that is not managed by his current unit, he moves to the proper unit. For the environment distribution method to be effective, each unit has to manage approximately the same number of agents and the number of edges connecting the partitions has also to be minimized (to reduce the number of agents being transferred between units).

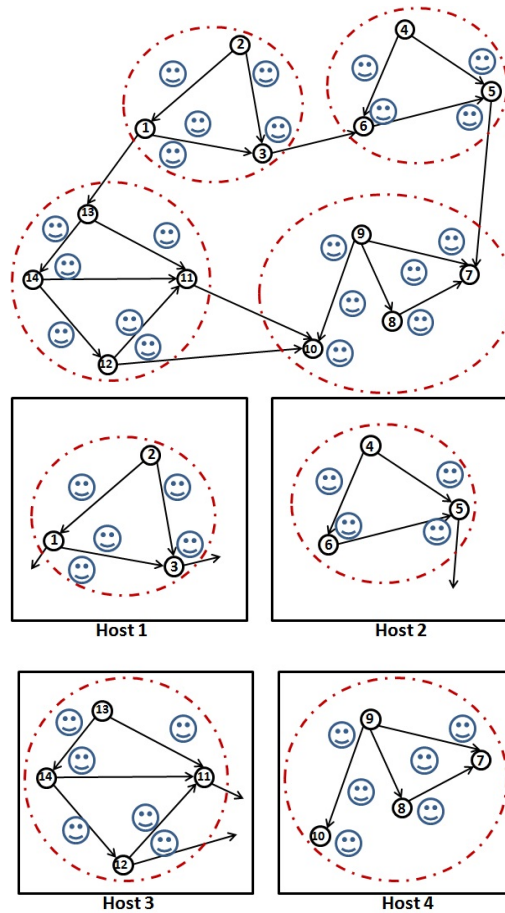


Figure 2.18: Environment distribution

The problem of graph partitioning has been widely studied in the scientific literature. We propose a method derived from the Differential Greedy algorithm [Fiduccia and Mattheyses, 1982] (Algorithm 1). For edges partitioning, we make the same choice as [Cetin et al., 2003] by not cutting edges in the middle. We associate each edge with the partition of its origin node.

The algorithm starts by creating a minimal partition with only one node each (instructions 1 to 6). Then, while there are nodes to be associated to partitions, the algorithm:

- chooses the lightest partition P_p , in terms of agents present in it (instruction 9),

Algorithm 1 Differential Greedy**Require:** Graph $G = (V, E)$ **Require:** Number k of partitions**Ensure:** Partition P

```

1:  $P \leftarrow P_0, \dots, P_{k-1}$ 
2:  $V' \leftarrow V$ 
3: for  $p \in [0, k - 1]$  do
4:    $v \leftarrow$  random vertex of  $V'$ 
5:    $P_p \leftarrow \{v\}$ 
6:    $V' \leftarrow V' \setminus \{v\}$ 
7: end for
8: while  $|V'| > 0$  do
9:    $p \leftarrow$  index of the lightest partition
10:   $m = \min_{v \in V'} (1 + \varepsilon)(\text{number of } v\text{'s neighbors} \in P_p) - (\text{number of } v\text{'s neighbors} \notin P_p)$ 
11:   $mv =$  random vertex  $v \in V' | (1 + \varepsilon)(\text{number } v\text{'s neighbors} \in P_p) - (\text{number of } v\text{'s neighbors} \notin P_p) = m$ 
12:   $P_p \leftarrow P_p \cup \{mv\}$ 
13:   $V' \leftarrow V' \setminus \{mv\}$ 
14: end while
15: Return  $P$ 

```

- finds the nodes that are the most connected with the nodes already in P_p and that are the least connected with the nodes that are not in P_p . The parameter ε gives more or less importance to the nodes that are close to the partition (instruction 10),
- chooses one of these nodes, adds it to the partition and removes it from the nodes to process (instructions 11 to 13).

This algorithm is fast and intuitive. Our modification of the original differential greedy algorithm concerns the choice of the current partition to treat. The “lightest” partition in the original algorithm concerns the number of nodes in the partition, while in our algorithm, it concerns the number of agents in the partition. The second difference concerns the use of ε , which encourages the connectivity of the sub-partitions. Indeed, having sub-partitions that are not connected could lead to agents going back and forth to the same host while staying in the same neighborhood.

Macroscopic simulation with environment-based distribution: When used with an environment-based distribution, the computation units in the macroscopic simulation exchange the current travel times on the transportation edges, to be able to compute the shortest paths for the agents. However, since all the agents on an edge are present on the same unit, they do not need to exchange the number of agents per edge. The speeds of the agents can indeed be computed locally.

Microscopic simulation with environment-based distribution: When distributed following the environment-based distribution pattern, the agents in a microscopic simulation need to know the state of the agents preceding them. In contrast with the agent-based distribution model, the agents preceding them are by definition present on the same com-

putation unit. The interrogation of the edges is then local to the concerned computation unit. The units keep on exchanging the current travel times (provided by the agents) to compute the shortest paths for the agents.

2.3.3 Diffusive Load Balancing

With the environment-based distribution, the graph partitioning is executed once, based on the initial positions of the agents and the network structure. However, if the network structure is stable, agents positions are of course changing during the simulation, which could lead to load imbalance. Typically, travelers drive from their residential areas to work areas in the morning and drive back home in the evening. Certain parts of the network, and consequently their corresponding computation units, would have many more agents to handle than the others, and the whole simulation would slow down. Indeed, at the end of each time step of the simulation, all the units have to wait for each others to synchronize. The overall execution time of a simulation step is then equal to the execution time of the slowest unit. As the execution time of a given unit is directly linked to the number of agents executing on this unit, it is important in these conditions to keep the load balanced.

2.3.3.1 The algorithm

A straightforward way to balance the load dynamically would be to part the graph from scratch when one unit is overloaded. But in the traffic simulations we are targeting, we have to deal with big graphs and many agents: the time needed to part the graph and to move all the agents from one unit to another would be counterproductive and would slow down the simulation.

Algorithm 2 Diffusive Load Balancing

Require: P partition of a graph $G = (V, E)$

Require: P_i current partition

Require: n total number of agents

Require: k number of processing units

1: $threshold \leftarrow \alpha(n/k)$

2: **if** number of agents in $P_i > threshold$ **then**

3: $P_{min} \leftarrow$ partition connected to P_i with the minimum load

4: $v_{max} \leftarrow$ the heaviest vertex $v \in P_i$ connected to P_{min} **with** $|v| < 0.5(n/k)$

5: move v_{max} to P_{min}

6: **end if**

That is why we have developed a dynamic load balancing algorithm, able to diffuse incrementally the excessive workload of a unit on the units around. At the beginning of the simulation, we use the modified differential greedy algorithm to part the graph. Then, during the simulation, each unit maintains a list of boundary vertices of the traffic network. These vertices are the ones who have a common edge with a vertex managed by another unit. When the load of a processing unit (in terms of number of agents) exceeds a threshold, we trigger the load balancing mechanism (instructions 3, 4 and 5). The unit will request the load of the processing units around, and will transfer its most heavy vertex (in terms of number of agents on it) to his least loaded neighbor (cf. algorithm 2). However, when

there is a huge number of agents on a vertex, the latter will continually be sent between the units. To avoid this perpetual oscillation, we define a limit on the number of agents (here half of the perfect load, instruction 4) from which the vertex will not be moved. This algorithm avoids to part the graph from scratch, and allows a good load-balancing with a linear complexity: $O(n) + O(k)$.

For instance, in Figure 2.19, the W values represent the number of agents in each partition. The partition 1 is initially overloaded (a), compared to the others. The partition 2, which is the neighboring partition with the smallest load is selected for the transfer. At this point, both vertices 7 and 9 are candidates to be transferred, as they are in the boundary between the partition 1 and 2. The heaviest vertex (9) is selected, and sent to partition 2. This gives us a new graph partition, by load diffusion.

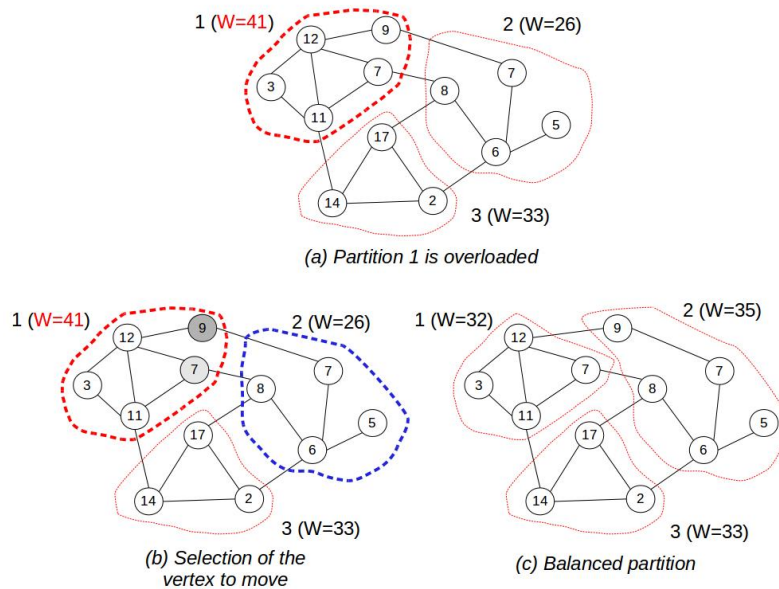


Figure 2.19: Diffusive load balancing

The choice of the coefficient $\alpha \geq 1$ is crucial here, because it will determine how often the mechanism will be triggered. Indeed, the closer α is of 1, the most often the procedure of load balancing will be triggered. Triggering it too often would lead to unstable partitions.

2.3.4 Experiments and Results

2.3.4.1 Implementation

A way to execute a distributed simulation is to define a distributed program where each computation unit, while executing the same program, owns only a part of the program data in its private memory, and all the processors are connected by a network. The advantage of this approach is its high scalability. Indeed, it can be implemented on most parallel architectures and we can deploy the same simulation on larger systems if we need more

computing power and memory. We use Python to develop our simulator for its efficiency in quick prototyping. Python is a mature portable language with many well tested scientific libraries and is along with C and Fortran one of the most used languages for high performance computing [Langtangen and Cai, 2008]. Here, we do not seek absolute performance, but we aim to study the relative efficiency of different distribution methods. Thus we believe that Python is a relevant choice. The inter-process communications are managed by MPI, which is the standard language for parallel computing with a huge community of users. MPI offers a simple communication model between the different processes in a program and has many efficient implementations that run on a variety of machines⁷.

We have executed the distributed simulations on an experimental cluster that we have set up. For our tests, we used two hosts under Linux Mint 17.2 Rafaela (kernel version 3.16.0-38-generic) each with an Intel Xeon processor CPU E7-4820 (32 cores at 2Ghz) with 250GB of memory. We ran the simulations on six configurations: the first is a sequential version of the program on a single core and the five others are distributed versions with 4, 8, 16, 32 and 64 cores. The behaviors in terms of traffic of the sequential and all the distributed versions are rigorously the same.

We have considered two networks. The first is a real network concerning the Paris-Saclay region, France, with 1895 nodes and 3831 edges. The number of daily travelers using this network is around 110,000. The origins-destinations of the travelers are based on real data travel patterns. We consider from 10,000 travelers to 500,000 travelers in our simulations. That means that we represent from around 10% to around 500% of the real number of travelers in our simulations. The second is a virtual network of 200 nodes power-law graph generated with the [Barabási and Albert, 1999] model with random origins and destinations.

2.3.4.2 Results

Distribution method \times simulation model: In this section, we compare the two methods of distribution (agent-based and environment-based distributions) with the different simulation models (microscopic and macroscopic) with increasing number of agents (from 10,000 to 500,000) on the Paris-Saclay network.

The speedup measures how many times the distributed simulation is faster compared to the corresponding sequential execution. The speedups for the two distributions methods applied with the different models are plotted in Figure 2.20 and Figure 2.21.

As we can see in Figure 2.20, the agent-based distribution is efficient for a macroscopic model (more than 5 times faster with 500,000 agents). There is no local interactions in this type of simulations and therefore this method allows to get a perfectly balanced load all along the simulation, while keeping the amount of inter-servers communications at the minimum. However, this method is particularly ineffective in the case of a microscopic model simulation. Indeed, the agents interact continuously with the other agents that are not situated in the same unit. This generates many communications between the servers, and the gain of the parallelization is annihilated by the time required by these communications. This method is even less efficient than the sequential execution for the microscopic model (speedup < 1).

⁷MPI4PY is an efficient interface that allows to use MPI with Python.

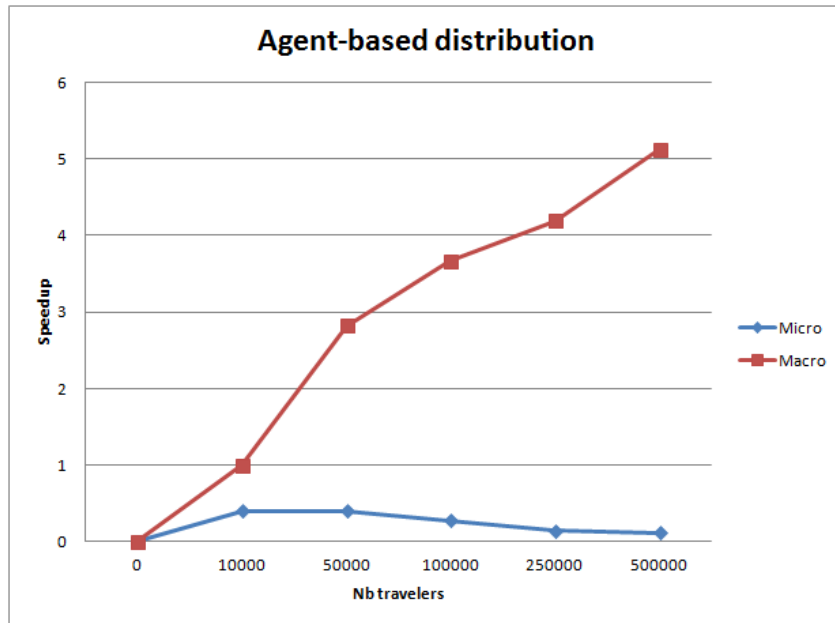


Figure 2.20: Speedup for the agent-based distribution (Paris-Saclay network)

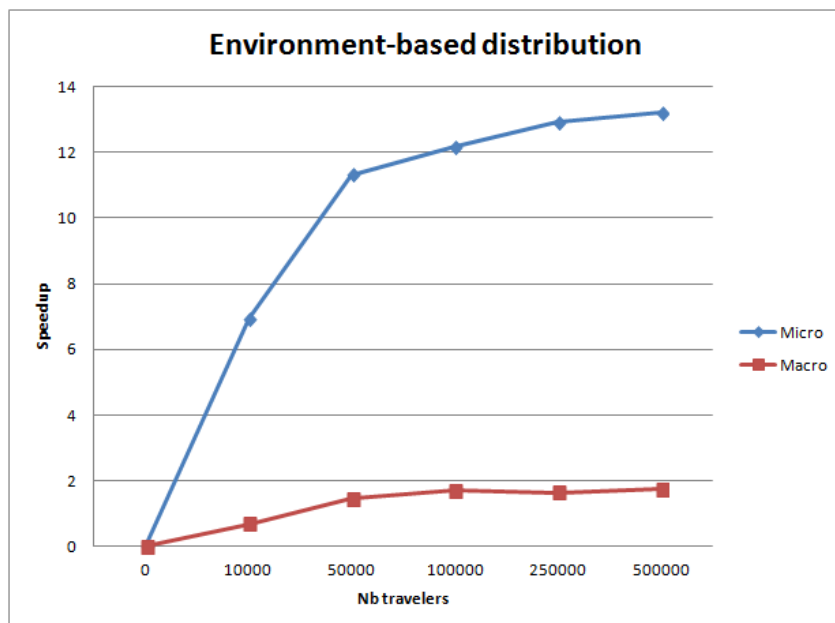


Figure 2.21: Speedup for the environment-based distribution (Paris-Saclay network)

As we can see it in Figure 2.21, the environment-based distribution is well adapted for a microscopic model simulation though. This method is up to 14 times faster than a sequential execution applied in a microscopic model simulation. This distribution pattern is less efficient than agent-based distribution for a macroscopic model simulation though. The explanation of the poor results of the environment-based distribution when applied to

a macroscopic model simulations is that the communication of edges costs between hosts, necessary for the computation of vehicles speeds, takes too much time and penalizes these simulations.

Impact of network type: To assess the impact of network type and size on the distribution, we have executed the same methods on a virtual network of 200 nodes power-law graph generated with the Barabasi-Albert model [Barabási and Albert, 1999]. Origins and destinations are this time generated randomly. The Figure 2.22 provides the results for the agent-based distribution and Figure 2.23 provides the results for the environment-based distribution. For agent-based distribution, the findings are the same: the macroscopic model simulations behave way better than microscopic model simulations. However, for environment-based distribution, the findings are different: the two simulations types profit of the distribution at the same level (macroscopic model simulations perform even slightly better). We explain this difference in the results by the network size. Indeed, Paris-Saclay network is ten times bigger than the considered virtual network. The communication of edges costs between hosts, necessary for the computation of vehicles speeds, is a lot less costly with the virtual network and does not penalize the macroscopic model simulations anymore.

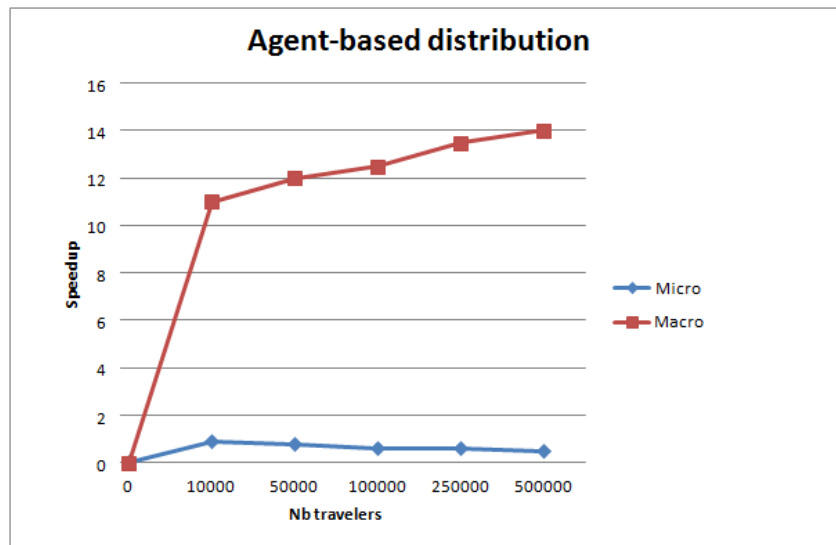


Figure 2.22: Speedup for the agent-based distribution (Barabasi network)

Increasing hosts numbers: To better assess the scalability of the different distribution methods, we execute the different simulations on the Paris-Saclay network with increasing number of available hosts: 4, 8, 16, 32 and 64. We consider 100,000 travelers in these simulations, which corresponds approximately to the real number of daily travelers. The results are reported in Figure 2.24 for agent-based distribution and Figure 2.25 for environment-based distribution. The results show that the speedup is increasing fast for environment-based distribution with microscopic model simulation and for agent-based distribution with macroscopic model simulation. The speedup is stable or increasing slowly for agent-based distribution with microscopic model simulation and for environment-based

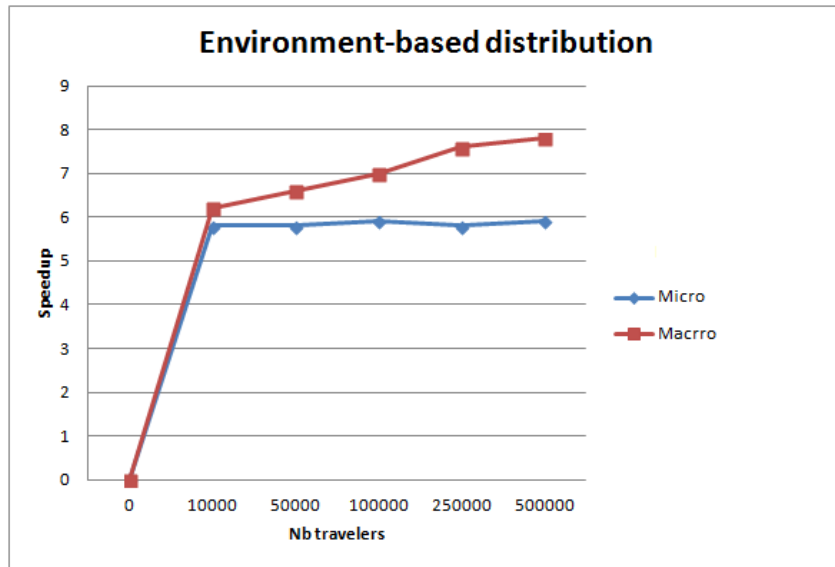


Figure 2.23: Speedup for the environment-based distribution (Barabasi network)

distribution with macroscopic model simulation. These results confirm the findings of the previous section: environment-based distribution is efficient with microscopic model simulation and agent-based distribution is efficient with macroscopic model simulation.

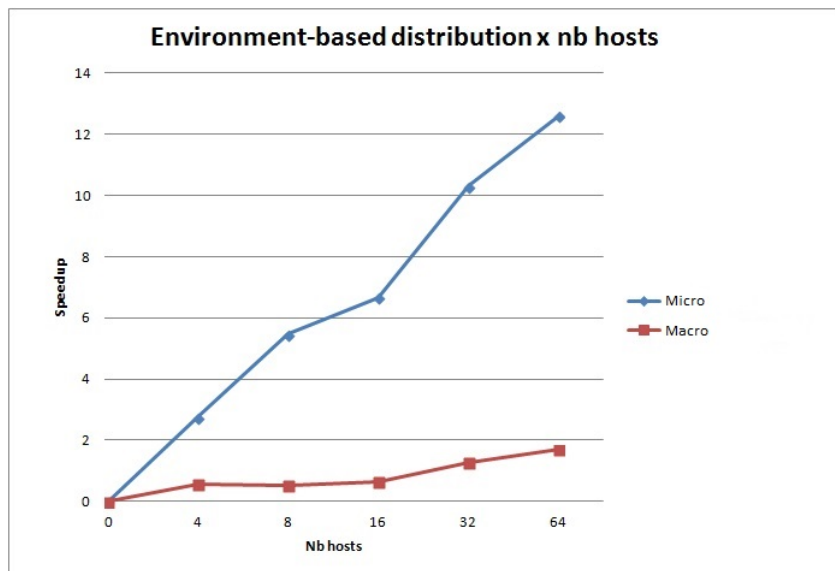


Figure 2.24: Impact of the number of hosts (agent-based distribution)

Impact of load-balancing: For the assessment of the load-balancing mechanism, we have to define the optimal value of α for the experiments. To do so, we execute three different types of simulations, each applied to the microscopic simulation type: the first,

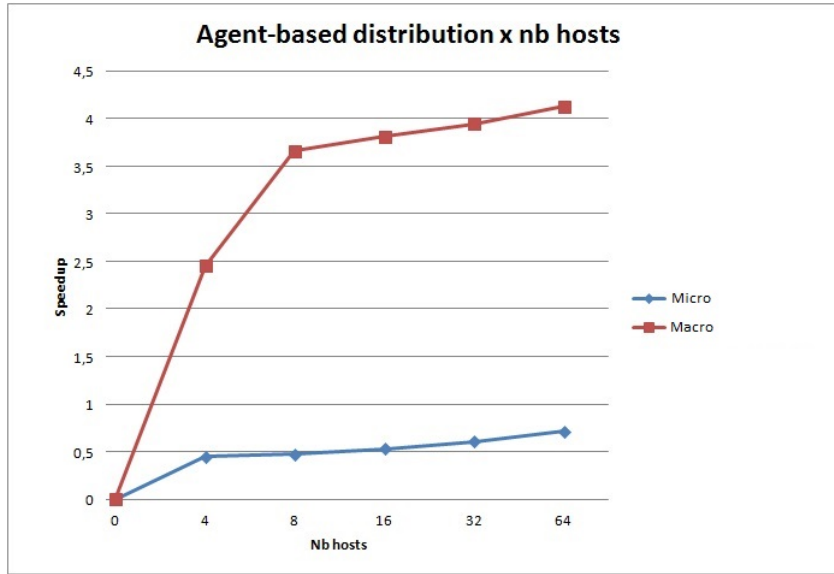


Figure 2.25: Impact of the number of hosts (environment-based distribution)

number of agents	10,000	50,000	100,000	250,000	500,000
Sequential (1 proc)	12,814	62,672	142,350	315,876	631,243
Static (64 cores)	463	2,136	3,902	9,636	18,929
Dynamic_1.3 (64 cores)	327	1,382	2,665	6,468	13,480

Table 2.5: Load balancing: computational times (in seconds)

called “static” is the environment-based distribution approach presented earlier. The second is the load-balancing approach with $\alpha = 1.2$ (called “dynamic_1.2”) and the third is the load-balancing approach with $\alpha = 1.3$ (called “dynamic_1.3”)⁸. The Figure 2.26 shows the results. Each point in the curves represents the difference between the optimal load (equal agents distribution between units) and the load on the most loaded process, for each time step. As we can see, with the “static” approach, the difference (the imbalance) is big. With the dynamic approach and $\alpha = 1.2$, the balance is better than with the static approach, but the load is unstable. Finally, with $\alpha = 1.3$ the oscillations cease and the load of the simulation is successfully balanced between the processes. Based on a series of experiments that we have executed, choosing a bigger α would lead to more imbalanced partitions, so we choose to keep $\alpha = 1.3$ for the rest of our experiments.

The Table 2.5 indicates the execution times for a simulation of 1,000 time steps with the sequential method and the two distribution methods (static and dynamic with $\alpha = 1.3$). The Figure 2.27 shows the speedups of the two methods in comparison to the sequential execution.

Finally, the Figure 2.28 exhibits the efficiency of the dynamic load balancing in function of the number of used processes. The simulation we ran here was for 100,000 agents and 1,000 time steps, with 64 processes. We can see that with our load balancing, the simulation

⁸We do not display the curve for $\alpha = 1.1$ because it was extremely unstable.

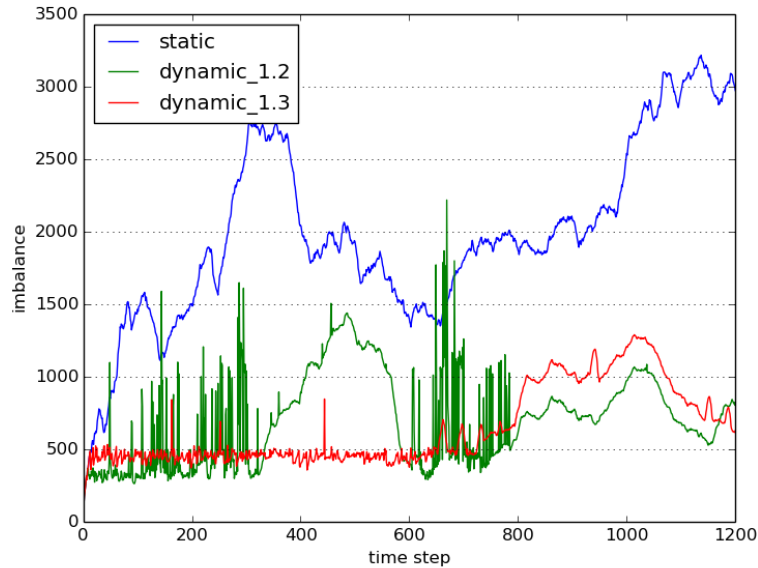


Figure 2.26: Load imbalance

scales very well with the number of process we have at our disposal. For 100,000 (which is approximately the load we can expect on the Paris-Saclay network) we reach a speedup of 54 with 64 processes.

2.3.5 Discussion

As we said in the introduction, the work on simulation distribution was motivated by the limitations of the simulation platform SM4T. We however desired to define methods that can be used for virtually all existing traffic simulations. To use the distribution methods presented in this section, we have to first classify the simulation platform at hand. For SM4T, the simulation is macroscopic because agents speeds are computed with a function associated with the edges that they are using. As a consequence, distributing SM4T should be performed following an agent-based distribution model. Of course, the actual implementation of the distributed version of the simulation platform has to be done. But we now know which distribution pattern has to be followed. This process can be applied with any simulation platform that one desires to distribute.

2.4 Conclusion and Perspectives

In this chapter, we have described our contributions in the multi-agent simulation for dynamic transportation applications. We defined the main building blocks of an agent-based simulator for multimodal travelers, which are easily reproducible in other transportation applications pursuing similar objectives. We have presented the main functionalities to-

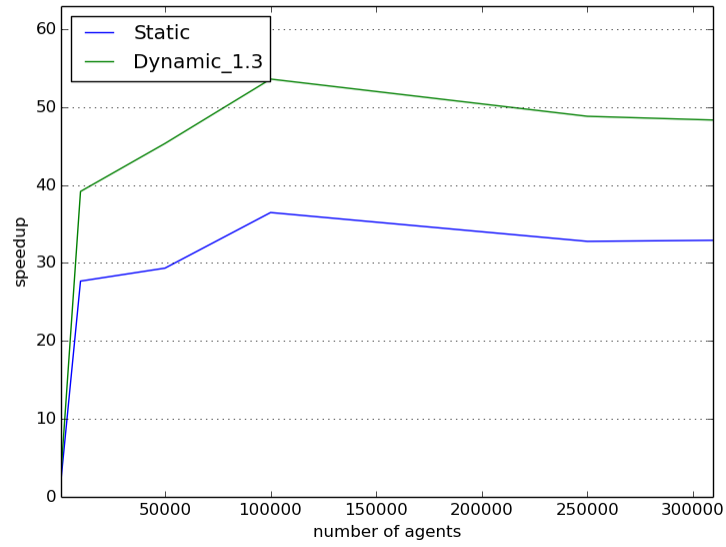


Figure 2.27: Speedup for the different methods

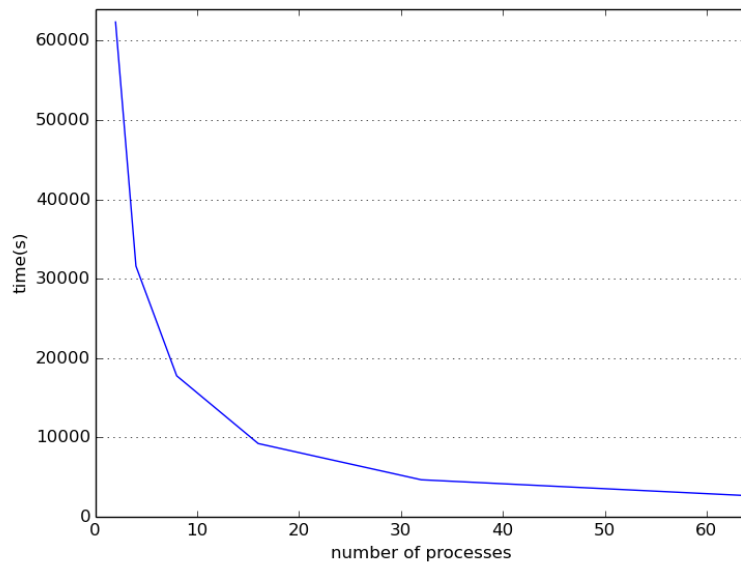


Figure 2.28: Execution time in function of the number of processes

gether with the data and parameters necessary for the application to work properly. We also presented a traveler information application implemented in the simulator, which objective is to evaluate the impact of personalized real-time information on the status of transit networks.

In the second part, we described proposals motivated by the scalability limitations of existing simulation platforms, and specifically of our simulation platform. We presented two distribution methods applied to two types of multi-agent traffic simulators. The results show that agent-based distribution is well suited for macroscopic model simulators while environment-based distribution is well suited for microscopic model simulations, ideally with a dynamic load-balancing mechanism. These findings are useful for the distribution of the existing multi-agent traffic simulations.

We have multiple perspectives for our work on simulation. We are working on more powerful dynamic pathfinding algorithms like D*Lite [Koenig and Likhachev, 2002]. We also plan to integrate learning processes in the behavior of the traveler agents, which would also impact the itineraries calculation. The consideration of passengers preferences, other than the minimization of travel times (as in, e.g. [Adacher et al., 2014]) is also interesting and is currently under investigation.

For the traveler information application, the optimization of the passengers information strategies by the network operators is a main perspective. Indeed, from a certain threshold of connected passengers, the operators can decide to optimize the information provided to passengers for a best management of their assignment on the network. Two interesting problems arise in this context. First, assigning travelers dynamically on different itineraries, taking into account real-time disturbances is an interesting online optimization problem to tackle. Second, the compliance of passengers with the instructions, especially in this context (i.e. providing different itineraries for the same origin, destination and departure time) is interesting to verify. Surveys about this compliance level would be very useful in this context.

Concerning simulation scalability, We are working on the integration of information networks (such as social networks) and their impact on the distribution performance. Indeed, if travelers interact often, they should be preferably executed on the same units, or else they will generate too many communication and deteriorate the performance of the system.

Multi-Agent Optimization for Dynamic Transportation Problems

*“Boundedly rational agents experience limits
in formulating and solving complex problems
and in processing (...) information.”*

Herbert Simon, The quarterly journal of economics (1955)

Contents

3.1	Introduction	53
3.2	Online Localized Resource Allocation Problems	55
3.2.1	Resources and Consumers	56
3.2.2	Allocation Modeling	58
3.2.3	Solution Constraints	59
3.2.4	Objectives	60
3.3	Multi-Agent Approach for Urban Parking Management	60
3.3.1	An OLRA Modeling of Urban Parking	61
3.3.2	Multi-Agent Solution for Urban Parking	62
3.3.3	Experiments	69
3.4	Multi-Agent Approach for Dynamic VRPTW	74
3.4.1	An OLRA Modeling of the Dynamic VRPTW	75
3.4.2	Multi-Agent Solution for the Dynamic VRPTW	75
3.4.3	Experiments	80
3.5	Multi-Agent Configuration for the Dial A Ride Problem	82
3.5.1	Multi-Agent System for the Multi-Company Dial A Ride	84
3.5.2	Experiments	86
3.6	Conclusion and Perspectives	91

3.1 Introduction

Transportation problems are defined as the transfer of entities between geographically separate locations at a minimum cost [Steenbrink, 1974]. The solving of the static version of

these problems, such as the traveling salesman problem and the vehicle routing problem has attracted a great amount of research and is necessary, especially for supply planning or fleet dimensioning purposes. However, in operational settings, the supply, demand and infrastructure are not static. To consider dynamic supply or demand, dynamic transportation problems are defined. They refer to a wide range of transportation problems where the problem data are not available *a priori* [Barbucha and Jdrzejowicz, 2009]. The missing or incomplete information concern either the transportation demand (e.g. goods, travelers, etc.), the transportation supply (e.g. the drivers or vehicles) or the transportation environment (e.g. the transportation policy, the trafficable network or the traffic status). Online transportation problems are a subset of dynamic transportation problems where the missing information concerns the transportation demand, which is discovered while the system is running. In online transportation problems, the system response time to the online demand is very important. In today's operational settings, several system components (vehicles, drivers, control entities, etc.) are located in the transportation environment and are equipped with computational power that could allow them to react to events in that environment. This is one of the reasons advocating for the use of multi-agent systems to solve them.

The first part of this chapter addresses the formulation of dynamic transportation problems. Indeed, each considered dynamic transportation problem is usually formulated in a specific mathematical or constraint program. While this formulation is necessary to have an unambiguous specification of the problems, their constraints and the accepted solutions, they are generally very different depending on the considered problem. We chose to rely on the general framework of resource allocation, and to specialize it to dynamic transportation problems. The result is a generic framework to describe dynamic transportation applications. The first part of the chapter describes the general framework for "online localized resource allocation problems". This resource allocation formulation is particularly suited for dynamic transportation applications, because it systematically considers the space and time dimensions of the problem. The model allows to have the same formulation of the dynamic transportation problems that we have tackled in our work.

The second part of the chapter describes multi-agent solutions to the three dynamic transportation problems that we have addressed: the urban parking management problem, the dynamic vehicle routing problem with time windows and the dial a ride problem. The multi-agent solution to urban parking management is based on a community-based system, in which vehicles exchange information about free parking spots using intervehicular communication. The next application is the dynamic vehicle routing problem with time windows. In this problem, situated travelers in time and space appear nondeterministically and desire to be visited by a fleet of vehicles, each traveler specifying a time window inside which he or she wants to be visited. To solve the problem, we propose a new measure, in the context of insertion heuristics, that is based on space-time networks (which we have already used in chapter 2). The third problem is dial a ride. We propose a multi-agent configuration allowing to assess the impact the competition of multiple companies on each new traveler.

Here follow the main design choices that we have made in this chapter. First the entities representation is individual. This individual representation has several benefits. It is possible to have heterogeneous properties and different or conflicting behaviors in the same system, as for the company agents in the multi-company configuration of the dial a ride problem. Second, the agents are rational and adopt a behavior that optimizes some

criteria that are linked to travel times. In the dynamic vehicle routing problem for instance, both vehicles and travelers aim at minimizing their detour w.r.t a direct itinerary. In the multi-company settings of the dial a ride problem, the travelers choose the company offering the best quality of service, which is based on a ratio between a direct itinerary travel time and the proposed travel time. Third, there is always an explicit representation of the environment (the transportation network at a minimum) and the transportation network is always represented in the form of a graph. In the vehicle routing problem, the environment also integrates a temporal dimension, in addition to the space dimension. The consideration of the only space dimension means that the agents only reason about the present and the current situation. Considering space-time graphs allows the agents to reason about the future and to use the environment for planning purpose. Finally, the time representation is discrete. The space-time network is designed based on a discretization of time and a duplication of the spatial graph multiplied by the resulting discrete times.

The chapter is structured as follows. Section 3.2 presents the model for online localized resource allocation. Section 3.3 presents the a multi-agent solution to the urban parking management, defined as a resource allocation problem. Section 3.4 presents the dynamic vehicle routing problem with time windows, together with the multi-agent system to solve it. Section 3.5 presents the dial a ride problem, together with the multi-company configuration of the system and the impact of competition in this context. Section 3.6 concludes the chapter.

3.2 Online Localized Resource Allocation Problems

A great part of dynamic transportation problems can be seen as resource allocation problems, where the challenge is to find an optimal allocation of resources to consumers. These resource allocation problems are recurrent in transportation applications and we believe that they should have a generic problem formulation representing them. This formulation would identify the common concepts and constraints of these applications. One of the main characteristics of these problems is that they require the simultaneous consideration of time and space. Indeed, in a transportation application, there is always an explicit representation of the environment (i.e. the transportation network). The actors (drivers, travelers, etc.) are localized in this environment where they dynamically move. A generic formulation of resource allocation that is specific to these problems is the first objective pursued in this chapter.

In transportation applications, the time dimension has to be explicitly represented because the information about resources and/or consumers is not known at the beginning of the allocation. This kind of problem is generally modeled as an online resource allocation (ORA) problem [Tesauro, 2005]. The space dimension has to be explicitly modeled because resources and consumers are situated and because the distance between them generally conditions the allocation: resources and consumers have to be geographically collocated or close enough for the allocation to take place. This kind of problem can be modeled as a localized resource allocation (LRA) problem [Golkar and Sousa, 2011]. We present in this chapter a generic model for both ORA and LRA problems called OLRA (for Online Localized Resource Allocation) problem. One main contribution is the introduction of a systematic, explicit and dynamic representation of the physical environment in the problem definition. Different instantiations of the problem specify different transportation applica-

tions. Indeed, consumers might have only access to a sub-part of their physical environment at a certain time, resources might be volatile, especially in a shared environment, and can therefore be taken by any close consumer and resources might be uncontrollable because they are created and released in a nondeterministic way.

A lot of applications can be modeled as an OLRA problem. Actually, a resource allocation problem that involves moving entities (resources or consumers) can be seen as an OLRA problem. For instance, in the search of charging stations for electric cars [Acha et al., 2011], the consumers are mobile (the electric cars), while the charging stations are the resources. In the sharing of vehicles (car, bike, etc.) [Katzev, 2003], both the resources (the vehicles) and the consumers (the drivers or the passengers) are mobile. The scheduling of aircraft landings to multiple run-ways [Beasley et al., 2000] can also be seen as an OLRA problem, where the run-ways are the resources and the planes are the consumers.

The generic problem that we define in this section involves the assignment of resources to consumers, where both resources and consumers are situated in space and time and are not known in advance. A consumer starts searching for a resource at nondeterministic moments. These moments are not predefined and are discovered during the allocation process. On the other side, the resources are available starting from unknown moments and remain available during an unknown period of time. The compliance of a resource with a consumer's needs is conditioned, among others, with their spatial and temporal situation. As for all resource allocation problems, the compliance of the resource with the consumer is also conditioned by the latter's preferences, which concern the properties and the state of the resource. The preference of a consumer for resources is measured with an individual utility function. The local objective of the consumers is to maximize their own utility while the global objective of the allocation system is generally to minimize the total traveled distance and/or the total travel time.

In the following, we formulate the Online Localized Resource Allocation (OLRA) problem and we define its various components.

3.2.1 Resources and Consumers

An OLRA is defined as a tuple:

$$OLRA = \langle \mathcal{R}, \mathcal{C}, G, D \rangle$$

where:

- $\mathcal{R} = \{r\}$ is the set of resources.
- $\mathcal{C} = \{c\}$ is the set of consumers.
- $G = \langle V, E \rangle$ is a directed graph, with V the set of nodes indexed from 1 to N , and $E = \{e_{ij} | i, j \in V \text{ and } i \neq j\}$ the set of edges.
- $D = \{d_{ij} | i, j \in V \text{ and } i \neq j, d_{ij} \in \mathbb{R}_+\}$, d_{ij} is the distance between two successive nodes i and j .

Each node of the network can contain one or more resources of \mathcal{R} . Resources may represent, for instance, vehicle seats, parking spots, places to recharge electric cars, etc. The distances

between nodes are static, while the travel times (defined below) may vary according to the dynamics of the environment. If the graph represents a transportation network, the travel times would be impacted by traffic status and congestion.

To represent the non-deterministic availability of a consumer or a resource, we define the following function:

$$\text{availability} : (\mathcal{R} \cup \mathcal{C}) \times T \rightarrow \{0, 1\}$$

where T is the time horizon. This function returns 0 if the resource is not yet localized, or if the consumer is either not localized or is not interested in any resource.

The two following sets describe the different characteristics of the resources:

- $\mathcal{P} = \{p_1, p_2, \dots, p_m\}$: represents all the possible properties of resources, with m the number of properties.
- $\Delta = \{\Delta_1, \Delta_2, \dots, \Delta_n\}$: contains the description domains of the properties, with $n \leq m$ the number of description domains (two properties might have the same description domain).

A property $p_i \in \mathcal{P} : \mathcal{R} \cup \mathcal{C} \rightarrow \Delta_j$ is a function that gives the value of the property p_i in its description domain $\Delta_j \in \Delta$. The description domain can be quantitative, qualitative or a finite set of data. Each resource is characterized by a set of property-value pairs. The properties that are defined for a resource, together with the corresponding description domain, are given by this function:

$$\varrho : \mathcal{R} \rightarrow (\mathcal{P} \times \Delta)^q$$

where q is the number of properties that are defined for the resource. If \mathcal{R} is homogeneous, resources are defined by the same q properties. Otherwise, the problem considers different types of resources, represented by different properties.

The function $\text{compatibility}(c, r, t)$ defines the fact that a consumer c and a resource r are compatible at time t , meaning that the values of the properties correspond to the requirement of the consumer at that moment. For instance, if the property concerns the type of electricity plug, consumers and resources should have the same value for that property all the time.

$$\text{compatibility} : \mathcal{C} \times \mathcal{R} \times T \rightarrow \{0, 1\}$$

The two following functions define the dynamic costs and the dynamic positions of resources and consumers.

- $\tau : V \times V \times T \rightarrow \mathbb{R}_+$, $\tau(i, j, t)$ returns the travel time between i and j at time $t \in T$ (the time horizon).
- $\rho : \mathcal{R} \cup \mathcal{C} \times T \rightarrow V$, $\rho(r, t)$ or $\rho(c, t)$ returns the node where the resource r or the consumer c is located at t .

A resource or a consumer on the edge e_{ij} is considered to be positioned on i until he reaches j .

3.2.2 Allocation Modeling

The following functions specify the dynamics of the allocation process. The interest of a consumer for a resource varies over time, either following an internal process or following his context. This context may include his current position or his final destination. The usefulness of a resource to a consumer is given by the following utility function:

$$\mu : \mathcal{C} \times \mathcal{R} \times T \rightarrow \mathbb{R}_+$$

$\mu(c, r, t)$ returns the utility of the resource r for the consumer c at time t . However, the consumption of a resource by a consumer is conditioned by their co-location. This can be verified with the following function:

$$\mathbb{1}_{\mathcal{F}} : \mathcal{C} \times \mathcal{R} \times T \rightarrow \{0, 1\}$$

$\mathbb{1}_{\mathcal{F}}(c, r, t)$ is an indicator function that returns 1 if the consumer c could have the same position than the resource r at time t and if they are both available at that time. That means that $\mathbb{1}_{\mathcal{F}}$ returns 1 for all the tuples $\mathcal{F} = \{(c, r, t) \in \mathcal{C} \times \mathcal{R} \times T \mid \rho(c, t) = \rho(r, t) \wedge \text{availability}(c, t) = \text{availability}(r, t) = 1 \wedge \text{compatibility}(c, r, t) = \text{true}\}$. $\mathbb{1}_{\mathcal{F}}(c, r, t)$ returns 0 otherwise. Therefore, \mathcal{F} defines the set of potential space-time co-locations of resources and consumers under availability constraints. The availability constraint is necessary to filter the situations where a consumer and a resource are indeed colocated, but not available, for instance if the resource has been taken by another consumer.

The quality of a resource allocation in OLRA is generally related to the distance and the travel time of consumers. Their successive positions throughout the execution are specified with the three following functions.

$$\pi : C \rightarrow (\{1, \dots, |\cdot|\} \times T)^n, n \in \mathbb{N}$$

π defines the path of a consumer. Applied to a consumer c , π returns the nodes that the consumer has visited while moving towards a resource, together with the times corresponding to his visits. $\pi(c)[i, 1]$ allows to access the index of the i^{th} visited node, while $\pi(c)[i, 2]$ allows to access the corresponding visit time.

For instance, $\pi(c_1)[3, 1] = 10$ indicates that the third node visited by consumer c_1 is v_{10} , while $\pi(c_1)[3, 2] = t_3$ indicates that this visit occurs at time t_3 .

$$\delta(c) = \sum_{i=1 \dots |\pi(c)|-1} d_{\pi(c)[i,1], \pi(c)[i+1,1]}$$

δ determines the total distance traveled by c . The term $d_{\pi(c)[i,1], \pi(c)[i+1,1]}$ represents an element d_{xy} of the D matrix of distances, where $x = \pi(c)[i, 1]$ and $y = \pi(c)[i + 1, 1]$ are, respectively, the i^{th} and the $(i + 1)^{\text{th}}$ node indices returned by $\pi(c)$. $|\pi(c)|$ gives the total number of nodes visited by c .

For instance, if $\pi(c_1) = [(5, t_1), (10, t_3)]$, i.e. consumer c_1 visits node v_5 at time t_1 then node v_{10} at time t_3 ; if the distance $d_{5,10} = 6$, then $\delta(c_1) = 6$.

$$\varphi(c) = \pi(c)[|\pi(c)|, 2] - \pi(c)[1, 2]$$

φ gives the total travel time of a consumer c . The expressions $\pi(c)[|\pi(c)|, 2]$ and $\pi(c)[1, 2]$ are the instants of visits, respectively, of the last node and the first node visited by c . With the same above example, if $\pi(c_1) = [(5, t_1), (10, t_3)]$, then $\varphi(c) = t_3 - t_1$.

3.2.3 Solution Constraints

A solution to an OLRA instance is an allocation of resources to consumers. This solution is given by the function γ , which specifies that a consumer actually consumes a resource at a certain time:

$$\gamma : \mathcal{C} \times \mathcal{R} \times T \rightarrow \{0, 1\}$$

$\gamma(c, r, t)$ returns 1 if a consumer c takes the resource r at t and 0 if not.

A consumer cannot take a resource if they are not at the same position at the same time. Hence, $\gamma(c, r, t) = 1$ cannot be valid unless $\mathbb{1}_F(c, r, t) = 1$.

OLRA is not bound to specific resource properties. It can model problems where the resources are shareable or not, and where consumers can consume several resources at the same time or not. The considered variant of the problem is specified by two parameters κ and ζ . The solution to the considered problem has to comply with the two following constraints, which depend on κ and ζ .

$$\sum_{c \in \mathcal{C}} \gamma(c, r, t) \leq \kappa, \forall r \in \mathcal{R}, \forall t \in T \quad (3.1)$$

$$\sum_{r \in \mathcal{R}} \gamma(c, r, t) \leq \zeta, \forall c \in \mathcal{C}, \forall t \in T \quad (3.2)$$

The constraint (3.1) specifies that the resources can be shareable and be taken simultaneously by at most κ consumers ($\kappa \in \mathbb{N}$). If the resources are not shareable, κ is equal to 1. If several resources are colocated with a consumer, the problem definition may allow him to consume them simultaneously by a consumer (constraint (3.2)). The number of resources that can be taken simultaneously is a parameter $\zeta \in \mathbb{N}$. Again, if this is not allowed, ζ is set to 1. The values of κ and ζ are model parameters and enable to take into account different problem variants and therefore different application types.

3.2.4 Objectives

The social objective of the study of OLRA is generally to minimize the time and/or the distance spent in the search of resources. This social objective can be expressed as:

$$\min \sum_{c \in \mathcal{C}} [\lambda \delta(c) + \beta \varphi(c)]$$

where λ and β are positive numbers weighting the relative importance of time and space in the specific problem that is considered. We assume that λ and β integrate scaling factors in order for the function terms to be expressed in the same unit and to truly reflect the weights of time and space. Indeed, $\delta(c)$ is usually expressed in meters while $\varphi(c)$ is usually expressed in seconds. The scaling factor could be based, for instance, on the average speed *speed* (expressed in meters per second) of the consumers. In which case, λ would be equal to $a \times \frac{1}{speed}$ where a is the actual weight of space and $\frac{1}{speed}$ the scaling factor.

Besides, every consumer has the local objective of maximizing his own satisfaction by obtaining the resources that best satisfy his preferences and maximize his utility. This personal objective is defined as follow:

$$\max \sum_{r \in \mathcal{R}, t \in T} [\mu(c, r, t) \times \gamma(c, r, t)]$$

A system might behave well w.r.t the local objectives of the consumers while the social objective is not optimized. Or it may exhibit good results for the social objective, while the individual objectives are of poor quality. As usual in this kind of problems, there is a compromise between these two objectives that the proposed solutions to this problem have to find.

In the following, we present multi-agent solutions to three dynamic transportation problems. Before each solution proposal, we propose an OLRA modeling for the problem.

3.3 Multi-Agent Approach for Urban Parking Management

The management of urban traffic growth is an important issue, since the number of drivers that are cruising for parking can exceed 33% of all traffic in large crowded city centers [Shoup, 2017]. The situation is getting worse, because the usage of cars is increasing, the cost of energy is getting higher and parking spaces are getting scarcer and more coveted. Indeed, several studies such as [Bayless and Neelakantan, 2012] have identified the importance of better parking systems to improve the quality of life, and an increasing number of smart parking solutions are being proposed to help optimizing the search for parking spots.

There is a growing conscience that cities are unable to cope with the continuous increase in car traffic. Parking policies, if they are well designed, contribute to more efficient use of the transportation network, higher densities, lower emissions, and better urban design [Shoup, 2017]. If not, they can act in the opposite direction.

Three main objectives have been identified in this context [Marsden, 2006]:

- to use urban parking management as a means of regenerating specific parts of the urban area (e.g. for town centers: providing more parking attracts more business);
- to use parking controls to restrain traffic and improve environmental quality, or to encourage the use of sustainable transports;
- to secure sufficient income from the parking operation to cover costs or to fund other activities.

These are the reasons that lead us to consider this problem in particular. We first propose an OLRA modeling of the urban parking problem, then we propose a multi-agent solution to it.

3.3.1 An OLRA Modeling of Urban Parking

An OLRA model for urban parking is provided in Figure 3.1. In the urban parking application, the set of resources \mathcal{R} is homogeneous, and composed of the parking spots. At the start of execution, \mathcal{R} might be equal to \emptyset , and is enriched by the parking spots made available. The set of consumers \mathcal{C} is composed of the drivers. G is the transportation network of the considered town, region or neighborhood. The nodes of the network represent either a crossroad or a parking spot on an edge. The time horizon T is the considered timeframe for the execution, typically 24 hours. The function $availability(r, t)$ returns 1 from the moment when a spot r is free until the moment when it is occupied. Similarly, $availability(c, t)$ returns 1 from the moment when a driver c is seeking a parking spot.

The possible properties of a parking spot are its size in centimeters p_{size} ($\Delta_{size} = \mathbb{R}_+$), the rating of the neighborhood $p_{neighborhood}$ ($\Delta_{neighborhood} = [0, 1]$) and its safety p_{safety} ($\Delta_{safety} = [0, 1]$). A parking spot r_1 can have the following values: $size(r_1) = 200$, $neighborhood(r_1) = 0.9$ and $safety(r_1) = 0.9$. The *compatibility* function in the model allows to define the minimal conditions a driver has for a spot. For instance, a driver can be interested in the only safe spots ($safety \geq 0.7$), which size is longer than 2 meters and that are not further from his final destination than 500 meters. Among the resources matching these conditions, the drivers uses his utility μ to sort them following a preference criterion, for instance from the nearest to the furthest.

In this problem, the spots can be taken by anyone, but not more than one driver can take a spot. As a consequence, the parameter κ is equal to 1. In addition, not more than one spot can be colocated with a driver, and one driver cannot take more than one spot at the same time. The parameter ζ is then also equal to 1.

To assess our proposal with an objective decision criteria, we have used a utility function that takes into account the time to reach the resources: $\mu(c, r, t) = \frac{1}{\tau(\rho(r, t), \rho(c, t), t)}$. Following the value of ζ and κ , each spot will be taken by at most one driver. If the driver leaves the spot and starts looking for another, they both will be considered as a new consumer and a new resource. Each driver c is looking for one resource r and his problem is to find among the potential pairs $\{(r, t) \mid (c, r, t) \in \mathcal{F}\}$ the one maximizing $\mu(c, r, t)$.

3.3.2 Multi-Agent Solution for Urban Parking

In the following, we propose a solution to the problem of urban parking that is based on a distributed architecture. In our configuration, there is no central information system nor an infrastructure that would list the available spots. Each driver has to expand his knowledge about the available spots by locally interacting with the other drivers. That means that we don't have a single accurate and up-to-date representation of \mathcal{F} , but each driver must build his own approximation of \mathcal{F} based on the knowledge that he has, which continuously evolve over time. We propose a multi-agent model for a system assisting drivers in their search for parking spots in an urban agglomeration. This solution uses minimal information on shared, volatile and uncontrollable resources. The multi-agent system works without initial information, without infrastructure to collect information about spots availability, and without a central information system [Bessghaier et al., 2012].

3.3.2.1 Agents model

The multi-agent system proposed here is fully decentralized. Agents employ inter-vehicular communication to exchange information with the other vehicles, which have to belong to the same community of equipped vehicles. The choice of a distributed approach allows, among other advantages, to minimize the infrastructure needed to implement this solution and to limit investment.

Since communication between vehicles takes place very locally, and necessitates their colocation in space and time, the use of a generic shared environment for interaction *à la* LACIOS is not relevant. Indeed, vehicles do not have a global access (via Internet for instance) to a communication medium that would allow them to communicate anonymously. The multi-agent environment in this application is the road network on which they move and that conditions their local communication.

The proposed system for the search of spots in an urban area is made of a type of agent designated by *assistant* agent, embedded in the driver's vehicle that supports him during his spot search. The assistant agent passes through four states as indicated by the automata of Figure 3.3.

- state 0: the vehicle is parked, the assistant agent is stopped.
- state 1: the driver has left his origin and is moving toward his destination.
- state 2: the assistant agent is looking for a parking spot to propose to the driver.
- state 3: the driver moves toward the spot proposed by the assistant agent. The latter stays aware of possible alternatives which would be more suitable for the driver.

Starting from state 0, the assistant agent As goes to state 1 when the driver starts his trip (and possibly releases a parking spot) (arc (1) in Figure 3.3). When the driver is near his destination, As switches to state 2 (arc (2)). If As has no spot to propose, the driver keeps on driving and looking for a spot while As keeps on looking (i.e. he remains in state 2). In this case, if As cannot propose a spot before the driver manages to find one on his way, he returns to state 0 (arc (3)). However, if As proposes a spot to the driver

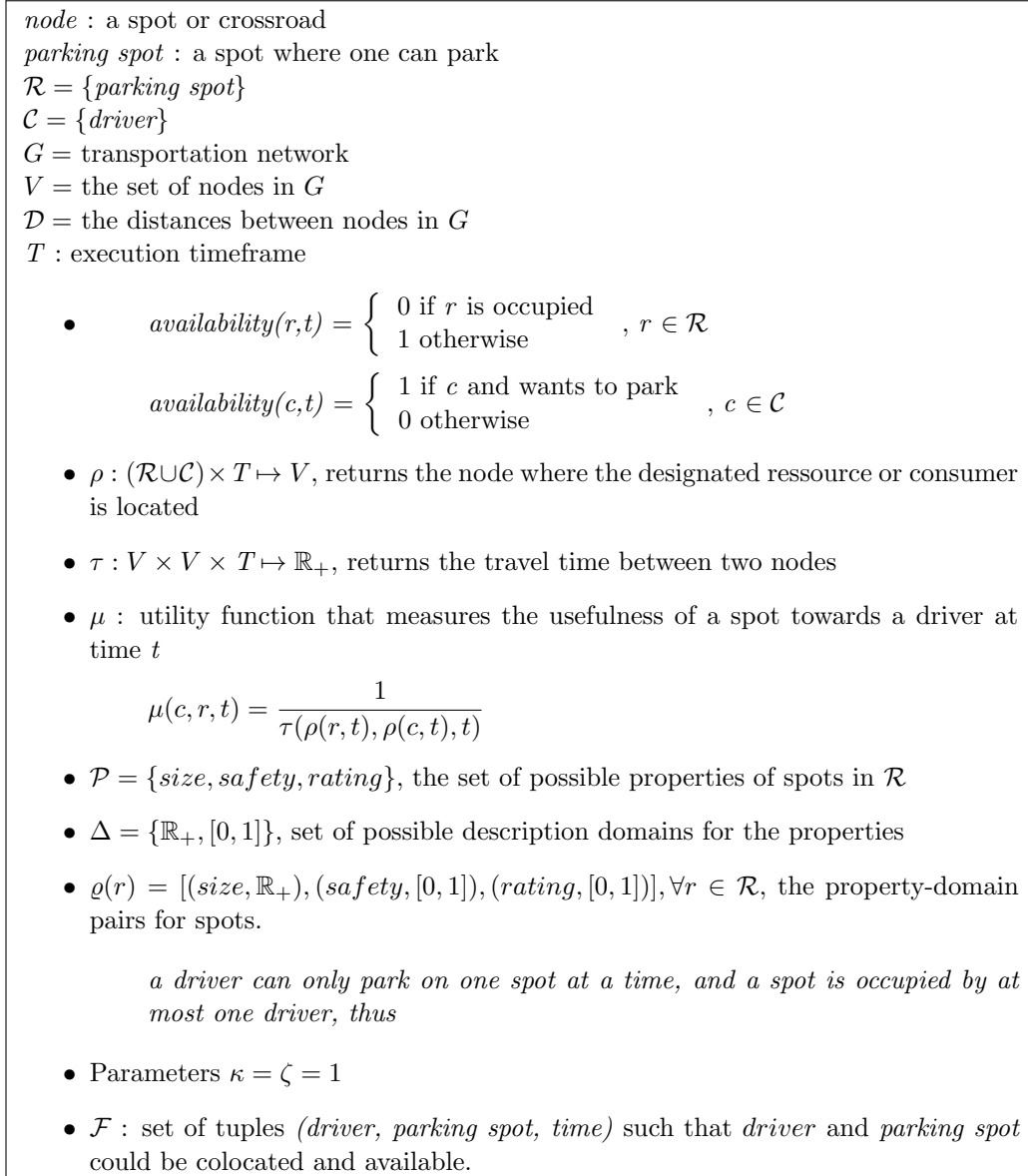


Figure 3.1: An OLRA Modeling of Urban Parking

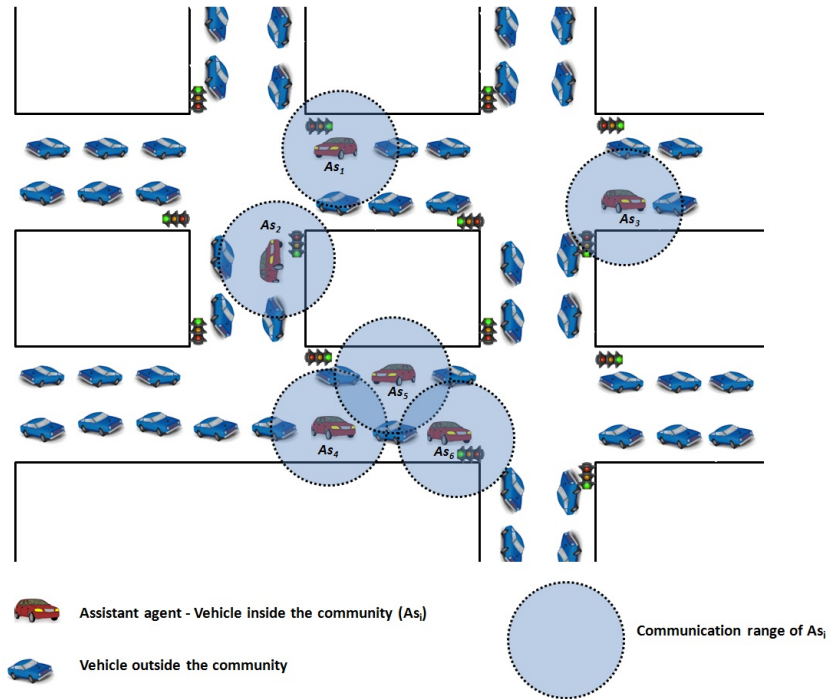


Figure 3.2: Information dissemination in the community

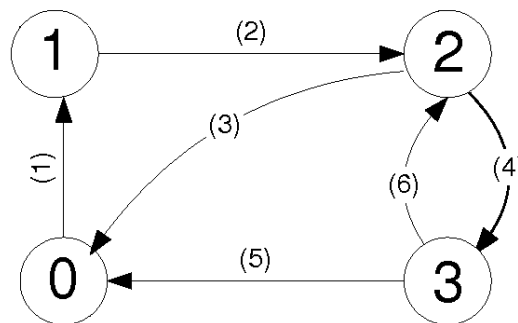


Figure 3.3: Assistant agent state diagram

together with his itinerary, he proceeds to state 3 (arc (4)) and the driver heads toward the chosen spot. Finally, from state 3, he goes to either:

- state 0 (arc (5)), if the driver finds a spot on his way that suits him better than the proposed one, or when he arrives at the chosen spot and it is free.
- state 2 (arc (6)), the search cycle starts again. This happens when the driver arrives at the spot and finds out that it is taken (for instance, if a driver from outside the community have found and taken the spot).

The internal architecture of the assistant agent is composed of three modules: an *Itinerary* module, a *Communication* module and a *Decision* module.

Itinerary module: The itinerary module calculates the route to the chosen parking spot starting from the driver's current position and then guides him to the spot. The choice of a parking spot is based on:

- the interaction with other assistant agents (*Communication module*)
- the choice of a spot maximizing μ and maximizing the chances of finding it available (*Decision module*)

Communication module: This module enables the agent to communicate with his neighbors, which have to belong to the community as well. This communication is based on messages and allows to exchange information about the availability of parking spots.

Our choice to make the agents communicate via an inter-vehicular network allows the information exchanged to move following two vectors. The first is specific to the communication. Indeed, the messages exchange takes place between each two neighboring vehicles in the community, and by transitivity agents can be informed of the availability of spots, however remote. For instance, in Figure 3.2, agents As_4 and As_6 share information via agent As_5 . The second vector concerns the movement of vehicles that mechanically move their information. For example, in Figure 3.2, agents As_1 and As_2 do not share information yet but will do so shortly following their movements.

However, the broadcast of information within the community can lead to a deterioration of the quality and the effectiveness of the system. There is quality degradation if an isolated agent cannot access or share his information. It is the case for agent As_3 in the figure. But also if many agents choose the same spot as it may be the case for agents As_4 and As_6 . The effectiveness of the system can also be challenged by a very large number of communications. Indeed, the information update is based on a restricted broadcast that depends on the vehicles location, but this communication is systematic. For instance, in Figure 3.2, communication between As_4 , As_5 and As_6 implies the exchange of four messages. On the scale of the entire transportation network, the number of messages at a time t is the sum of communication between all adjacent agents. Depending on the density of the network, this can represent a large number of messages. However, the communications take place very locally between vehicles and the total number of messages per agent should be less important than in a centralized architecture.

Decision module: Finally, the decision module takes care of two types of decision. With this module, the agent decides what information to share. This module also chooses which parking spots are the most relevant for the driver. The main issues of the information management about parking spots concern, on the one hand, the definition and the use of the information and on the other hand the update of this information. These two elements influence the quality of the knowledge of the assistant agent and therefore the quality of the decision process. The decision is based on information that is given by the members of the community via communication and is related to two events: 1) a parking spot is released by a member of the community; 2) a parking spot is occupied by a driver belonging or not to the community.

The choice of a parking spot for the driver must meet his criteria, which may concern for instance its distance from his final destination, the time since its release, or the safety of its location. In other words, the decision module of a driver c implements the utility function μ and computes his own approximation of \mathcal{F} (which defines the set of potential space-time co-locations of resources and consumers), both defined in the OLRA model. By abuse of notation, we use \mathcal{F} to denote the knowledge of the assistant agent about the space-time status of the available spots.

3.3.2.2 Cooperation model

The cooperation model depends on the nature of the information that is shared and how this information is used by the assistant agents. We propose to compare two cooperation models. In the first model, agents share the information about parking spots, while hiding the one that he has chosen. In the second model, the agents exchange all the information that they have about spots together with their intention about the parking spots. The shared information are:

- \mathcal{F} which contains information about the free spots and their associated release time.
- \mathcal{O} which contains the spots that were in \mathcal{F} but which turned out to be occupied with the moment when this information was known.

The minimal definition of a parking spot is a pair $\langle spot, time_{discovery} \rangle$: the geographic position of the spot and the moment when it was released. \mathcal{F} and \mathcal{O} contain each a set of such pairs that are exchanged between assistant agents. The combined use of the two sets provides a dynamic update of the system information. Indeed, one consequence of the volatility of information regarding the availability of spots is illustrated when an agent chooses a spot on his \mathcal{F} set assumed to be free but, once there, he finds it occupied. In this case, the \mathcal{F} sets contain incorrect information about this spot. The concerned spot is then moved from \mathcal{F} to \mathcal{O} and this information will spread over the community. Both sets are exchanged by the assistant agents and are updated gradually by the knowledge of each one, following two alternative cooperation models. The cooperation model defines which spots in \mathcal{F} to broadcast by the agent and which to hide. In the following paragraph, we describe the common behavior of the agents, regardless of the chosen cooperation model before to give the specific behavior of each model.

Generic behavior: The environment being dynamic, there is a *social update process* with information acquired in real time by communication, and also a *temporal update process* because the agent's knowledge evolves over time. The social update process begins in the communication module of the assistant agent where the sets \mathcal{F} and \mathcal{O} from each received message (denoted \mathcal{F}_B and \mathcal{O}_B respectively) are extracted and forwarded to the decision module. This corresponds respectively to edges (1) and (2) in Figure 3.4. The decision module updates both sets by aggregating the received \mathcal{F}_B and \mathcal{O}_B sets with his own \mathcal{F} and \mathcal{O} sets (denoted \mathcal{F}_A and \mathcal{O}_A respectively). The idea of the update process is to browse each received set (\mathcal{F}_B and \mathcal{O}_B) and update the local sets (\mathcal{F}_A and \mathcal{O}_A) in consequence, using the \oplus operator.

$$\begin{aligned}\mathcal{F}_A &\leftarrow \mathcal{F}_A \oplus \mathcal{F}_B \\ \mathcal{O}_A &\leftarrow \mathcal{O}_A \oplus \mathcal{O}_B\end{aligned}$$

The \oplus operator is defined as follows :

$$S_1 \oplus S_2 = S_1 \cup S_2 - \{\langle spot, time_{discovery} \rangle \in (S_1 \cup S_2) \mid \langle spot, time_{discovery2} \rangle \in (S_1 \cup S_2) \wedge time_{discovery} < time_{discovery2}\}$$

where S_1 and S_2 are sets of $\langle spot, time_{discovery} \rangle$ pairs. The \oplus operator merges the two sets ($S_1 \cup S_2$), and if there are two information about the same spot ($\langle spot, time_{discovery} \rangle$ and $\langle spot, time_{discovery2} \rangle$), only the one with the newest $time_{discovery}$ is kept. Note that $time_{discovery}$ associated with a spot in \mathcal{O} is the time when the occupancy of the spot has been discovered by a driver.

Then an update of \mathcal{F}_A or \mathcal{O}_A is launched, using the \ominus operator.

$$\begin{aligned}\mathcal{F}_A &\leftarrow \mathcal{F}_A \ominus \mathcal{O}_A \\ \mathcal{O}_A &\leftarrow \mathcal{O}_A \ominus \mathcal{F}_A\end{aligned}$$

The \ominus operator is defined as follows :

$$S_1 \ominus S_2 = S_1 - \{\langle spot, time_{discovery} \rangle \in S_1 \mid \langle spot, time_{discovery2} \rangle \in S_2 \wedge time_{discovery} < time_{discovery2}\}$$

$S_1 \ominus S_2$ subtracts from S_1 the $\langle spot, time_{discovery2} \rangle$ pairs $\in S_2$ with an information about a spot in S_1 ($\langle spot, time_{discovery} \rangle$ and $\langle spot, time_{discovery2} \rangle$) with a newer $time_{discovery}$. This way, the information in \mathcal{F} and \mathcal{O} continuously concern different spots with the newest information available.

The temporal update process of the agents is a filtering of outdated information after a time θ from \mathcal{F} and \mathcal{O} . The spots that were discovered θ time ago are deleted. The value of the parameter θ has to be chosen taking into account the transportation network activity. Thus, a low value reflects a high volatility (e.g. rush hours in downtown), while a high value keeps a longer sharing of information and reflects, for instance, the lower volatility in a residential area. This is done using the \odot operator.

$$\mathcal{F}_A \leftarrow \mathcal{F}_A \odot \theta$$

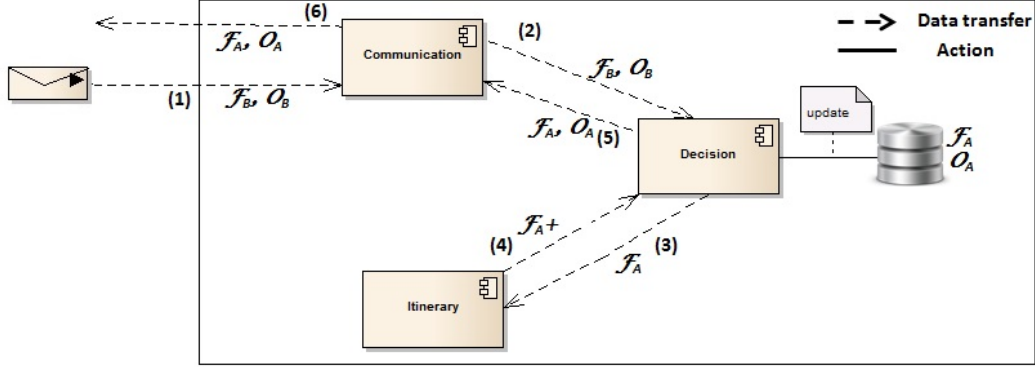


Figure 3.4: Assistant agent internal data flows

The \odot operator is defined as follows :

$S \odot \theta = S - \{\langle spot, time_{discovery} \rangle \in S \mid time_{discovery} < time_{current} - \theta\}$, with $time_{current}$ the current time.

The result of the application of $S \odot \theta$ will be the $(\langle spot, time_{discovery} \rangle)$ pairs in S , where all information older than θ is filtered (all the pairs where $time_{discovery} < time_{current} - \theta$ will be absent from $S \odot \theta$).

When the driver is looking for a parking spot, \mathcal{F}_A is sent to the itinerary module. This corresponds to the edge (3) in Figure 3.4. The itinerary module computes the routes for each spot on this set and forwards the result back to the decision module (edge (4) in Figure 3.4). Based on the utility function μ , the decision module proposes a spot that best meets the needs of the driver. Finally, he sends \mathcal{F}_A and \mathcal{O}_A to the communication module (edge (5) in Figure 3.4) which takes care of their distribution to the neighbors (edge (6) in Figure 3.4). Note that during the movement of the driver to the chosen spot, the assistant agent can receive new information about spots. That might make him suggest an alternative spot to the driver that would better meet his needs, following the generic behavior that we have just described.

Now, with the chosen spot and the lists \mathcal{F}_A and \mathcal{O}_A , the assistant agent has to choose which information to broadcast to the neighboring vehicles. We have identified two possibilities for this information broadcast, following two cooperation models: “cooperative” model and cooperative model, both described in the following paragraphs.

Every agent makes a decision individually, and based on his own knowledge, regarding the spot choice. If agents A and B cannot exchange their information, because their communication fields did not overlap directly or via other vehicles, they might head towards the same spots and there might be a conflict on the same spot. The cooperation models defined in the following sections try to limit the occurrences of such a situation. In our experiments, we verify if this situation happens, with respect to the cooperation model in the one hand and to defined parameters in the other.

The itinerary module calculates the travel times based on the latest known status of the network. The chosen spot is then the best possible for the driver when the assistant takes his decision. At each reception of information about available spots, the itinerary module

recalculates the shortest paths with the new status of the traffic. There is a continuous replanning for the drivers until they reach the chosen spot, that makes these choices the best possible, provided the knowledge of the agent and the absence of a central planner.

“Coopetition” model: The “Coopetition” model is a combination of cooperation and competition [Luo, 2007]. In our implementation of this model, agents are indeed cooperative, because they altruistically share information about spots. But they are also competitive, because they do not share an information about a spot if they are interested in it and are intending to take it. When the decision module chooses a spot, it deletes the information corresponding to the proposed spot from \mathcal{F} . The removal of the information about this spot reduces its spread within the community. Thus, the assistant agent increases the driver’s chances of finding the spot free.

Fully cooperative model: In the fully cooperative model, the agents exchange all the information about the spots, including the spot that they intend to take. But they also exchange their preferences about the parking spots as well. Indeed, they broadcast their intentions to the other agents. In this model, \mathcal{F} contains tuples $\{ \langle spot, time_{discovery}, time_{reach}, time_{intention} \rangle \}$. The $time_{intention}$ and $time_{reach}$ define the moment when the agent took a decision to head toward a spot and the time needed for him at that moment to reach the spot. This way, every agent would know if he can be at a spot before another has reached it. If not, he would choose another spot with more chances for him to get it. The fields $spot$ and $time_{discovery}$ together with the θ parameter are still used to filter the \mathcal{O} and \mathcal{F} sets with the \oplus , \ominus and \odot operators.

3.3.3 Experiments

To demonstrate the effectiveness and utility of our proposal, we have conducted many series of simulations. Our objective is to assess three hypothesis.

Hypothesis 1 *Inter-vehicular communication has positive impact on urban parking.*

To verify this claim, we compare simulations results with inter-vehicular communication and driving assistance, with simulations where drivers look for spots on their own.

Hypothesis 2 *The quality of the result depends on the accuracy of \mathcal{F} , thus declaring drivers intentions about spots is better for urban parking.*

To assess this hypothesis, we compare results with the cooperative model versus results with the cooperative model.

Hypothesis 3 *The number of messages per agent is less in a system with inter-vehicular communication than the number of messages treated by a central server guiding the drivers.*

To verify this claim, we compare results with the distributed model versus results with the a centralized model.

3.3.3.1 Configuration

In a dynamic and online problem such as the urban parking problem, evaluating a new proposal is an issue. Indeed, optimal solutions are for static problems where the problem data do not change during execution. Optimal solutions also assume the presence of a central system that optimizes every vehicle's route. To find optimal solutions for this problem, we would have to solve a static allocation problem with each modification in the problem data (traffic times, new driver, new available spot, etc.). Provided this difficulty, and since we propose fully distributed solutions, our objective in this experiments section is to compare the system behavior with the two cooperation models that we propose for the drivers (cooperative and competitive). The two models are compared with a default model, where the drivers are not informed at all (called the reference simulation).

We use the road network of the city of Saint Etienne, France. We place 124 spots on this network. We have 300 agents in all the simulations, and we vary the number of agents in the community from 100 to 300 with a step of 20 agents. As a consequence, we vary the number of agents outside the community from 200 to zero.

We define two system parameters. The first parameter is θ , the information lifetime in the agent's knowledge base. The second parameter is r , the radius of information broadcast around a vehicle. θ is expressed in simulation time cycles while a value of $r = 1$ is proportional to 4.26 meters. To tune the two system parameters θ and r , we define mean values and vary the two parameters around these values. For r , the mean value is chosen so that the communication range of 200 agents (the mean number of considered agents) cover the whole network if placed optimally. The computed value is 12. We vary the values of r in $\{5, 10, 15, 20\}$, then in $\{5, \dots, 60\}$ for specific tests. For θ , we run a first set of experiments to find a mean search time, which we use as a mean value for θ . The computed value is 14. We choose the values of θ in $\{5, 10, 15, 20\}$, then in $\{5, 10, \dots, 35\}$.

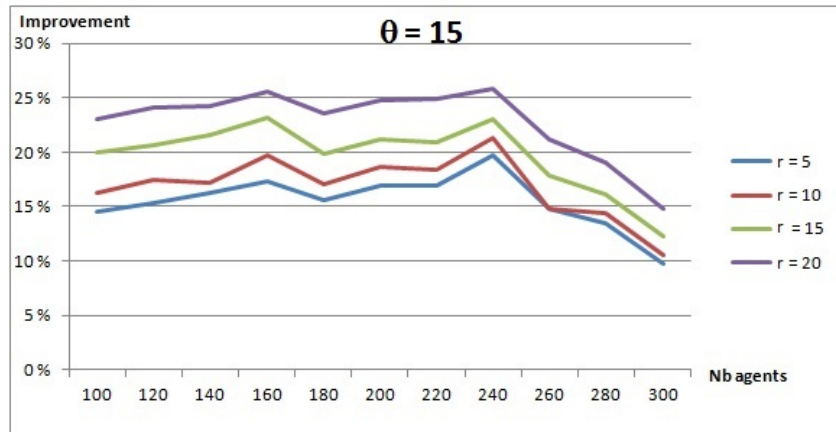
3.3.3.2 Results

In the following four paragraphs, we test our three hypothesis and we assess the impact of the parameters θ and r .

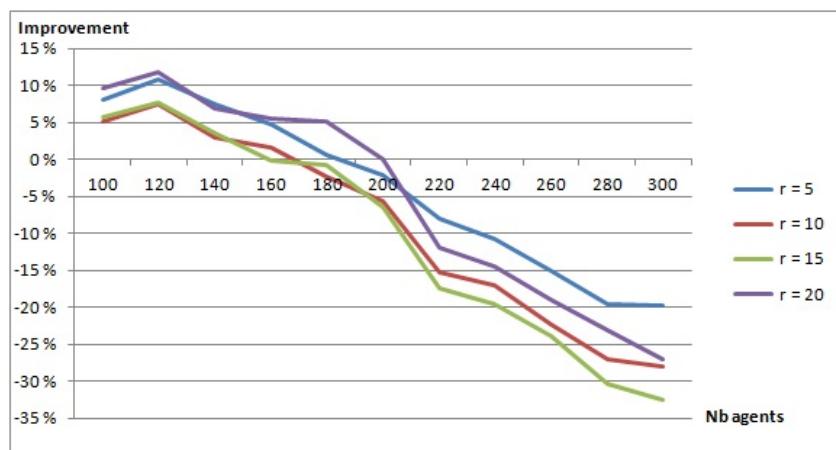
Cooperative Vs. competitive model: The first set of experiments is related to the comparison between three simulations:

1. a simulation with no intervehicular communication (the reference scenario)
2. a simulation with intervehicular communication and a competitive model
3. a simulation with intervehicular communication and a cooperative model

Figure 3.5 provides the results related to the cooperative model with $\theta = 15$. The results with $\theta \in \{5, 10, 20\}$ will be reported in the next subsection. The x-axis reports the number of considered agents in each simulation and the y-axis reports the improvements w.r.t the reference scenario. Each point in the different curves represents an average of 20 simulation runs. Results show that the cooperative model outperforms the reference

Figure 3.5: Cooperative model (with $\theta = 15$)

simulation. Starting from 240 agents, the difference between the two models becomes less important. This result validates hypothesis 1: using intervehicular communication and driving assistance is indeed beneficial for urban parking.

Figure 3.6: Coepetitive model (with $\theta = 15$)

An example of results, with $\theta = 15$, related to the coepetitive model this time, is reported in Figure 3.6. The results with $\theta \in \{5, 10, 20\}$ have the same trend, and suggest the same interpretation. The results show that the coepetitive model outperforms less and less the reference simulation when we consider from 100 to 180 agents. Starting from 200 agents, the reference simulation gives better results. That means that when the number of agents becomes widely greater than the number of available spots (3 agents for each 2 spots), it becomes useless to use urban parking assistance using a coepetitive model. This is due to the fact that hiding the chosen spot by the agent might lead several agents to choose the same spot, which is called “multiple-car-chasing-single-space” [Shi et al., 2004], especially when the number of available spots becomes limited. The figure 3.7 shows the effect of the chosen model on the concentration of vehicles around spots. We see clearly that the cooperative model enables to limit this concentration. Hypothesis 2 is then also valid.

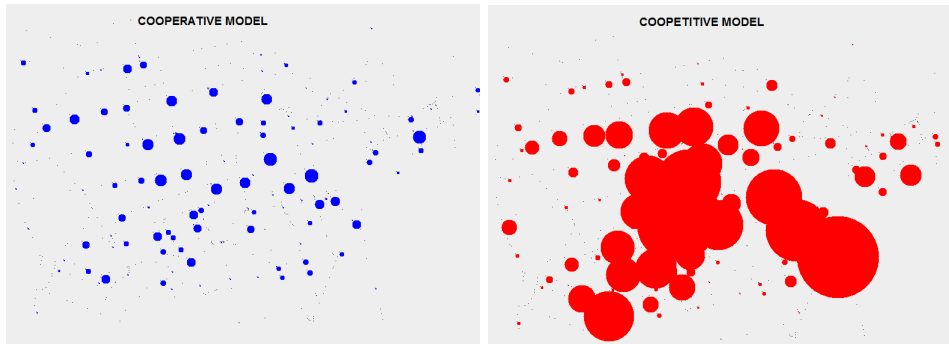


Figure 3.7: Density of the agents searching for a spot (300 agents and $\theta = 15$)

In the remainder of this experiments section, we use the simulation based on the cooperative model for investigating other aspects of the applications.

Impact of information lifetime: In the following series of simulations, we investigate the impact of the parameter θ , expressing the information lifetime, on the average search time. The results are reported in Figure 3.8. As we can see, a greater value of θ is beneficial for urban parking, since the improvements w.r.t the reference simulation are higher with higher values of θ . However, We observe a stagnation of the improvement beyond the mean value of $\theta = 15$. Indeed, the results are almost the same with $\theta = 15$ and $\theta = 20$. This result suggests that there is an optimum value of θ to be found, beyond which it is useless to keep information about spots. Indeed, high values of θ incur large data to be stored and exchanged between vehicles. If the marginal benefit of increasing θ becomes negligible, it is better not to increase it. This value has to be found and tuned for every considered region and each considered timeframe. Moreover, these results confirm that the use of θ reflects the volatility of the parking spots. Indeed, Figure 3.8 shows that the increase of θ influences mostly the result with a low volatility, i.e. when the number of agents in the community is less than 240.

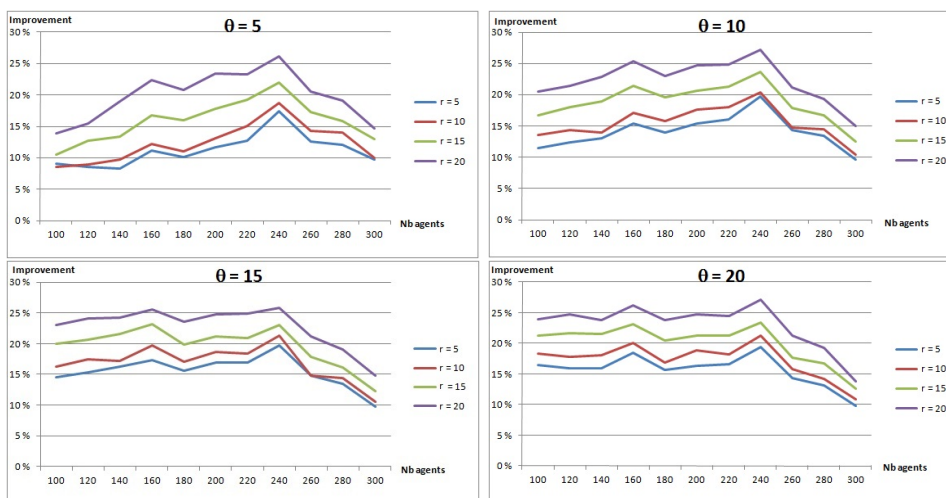


Figure 3.8: Impact of information lifetime, with $\theta = \{5, 10, 15, 20\}$

Impact of communication field: In the following series of simulations, we investigate the impact of the parameter r , expressing the communication field of the vehicles. In an urban area, vehicles communication fields can be limited due to obstacles, the objective of our investigation is to check whether lower communication field would significantly impact quality of service. Besides, it might be interesting to artificially limit that field since a higher value of r incurs higher number of exchanged messages and also more information to store and manage. Besides, the value of r cannot be increased indefinitely. Indeed, as the communication range r increases, it may cause communication interference and worsen the communication efficiency as well as the application built upon. In [Schmidt et al., 2011], it is specified that the degradation in VANETs starts to become significant starting from a range of 300 meters. In our simulation, the maximum value of $r = 60$ is equivalent to 255.6 meters and remains lower than the 300 meters threshold.

The results are reported in Figure 3.9. We have varied the value of r from 5 to 60 with a step of 5. As we can see it, a greater value of r is always beneficial, since the improvements w.r.t the reference simulation are higher with higher values of r . We also observe that the best results are with $\theta = 10$ and $\theta = 15$, higher values of θ provide worse results. However, whatever the value of θ , we observe less and less improvement starting from $r = 25$. There is a balance to find between the value of r and the number of incurred exchanged messages. This will be the object of the following paragraph.

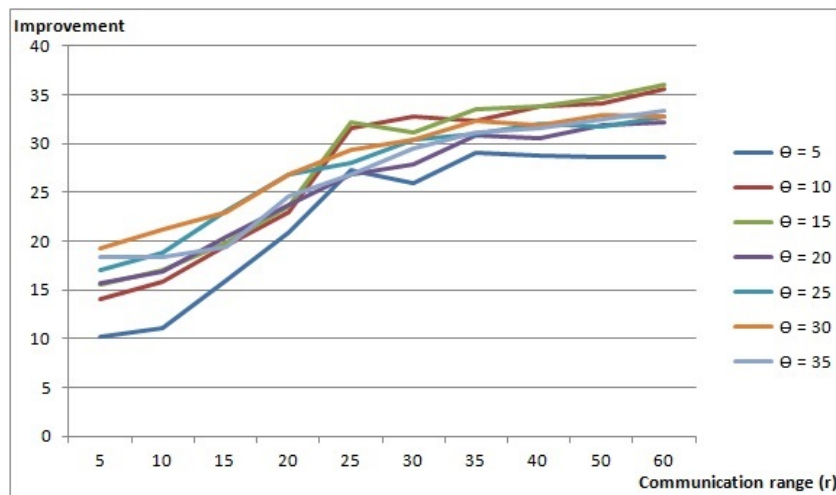


Figure 3.9: Impact of the variation of r

Number of messages: In the centralized version, each agent informs a central server when he releases a parking spot or if he finds a proposed spot occupied. The central server performs the choice of the spots, based on the vehicle knowledge and following the cooperative model. Thus, the central server mimics the same behavior of the cooperative model, and there is no difference in terms of travel times between the cooperative and central solutions. The objective is only to compare the number of messages per agent in the cooperative model with the number of messages manipulated by the central server.

We vary r from from 5 to 60 with a step of 5. The results are reported in Figure 3.10. Each point in the curve is the average of the simulation results with $\theta = \{0, 5, 10, \dots, 35\}$. As expected, a greater value of r incurs higher number of exchanged messages in the co-

operative model (r has obviously no impact on the centralized version since vehicles don't communicate directly). However, this number remains inferior to the number of messages handled by a central server until $r = 45$. Beyond this value, the cooperative model generates more messages per agent than the centralized version.

For this series of experiments, it seems than a value of $r = 25$ is a good compromise between search time optimization (around 27% better than the reference simulation) and number of exchanged messages (around 2000, which is 5 times less than the results with $r = 60$). These results validate hypothesis 3.

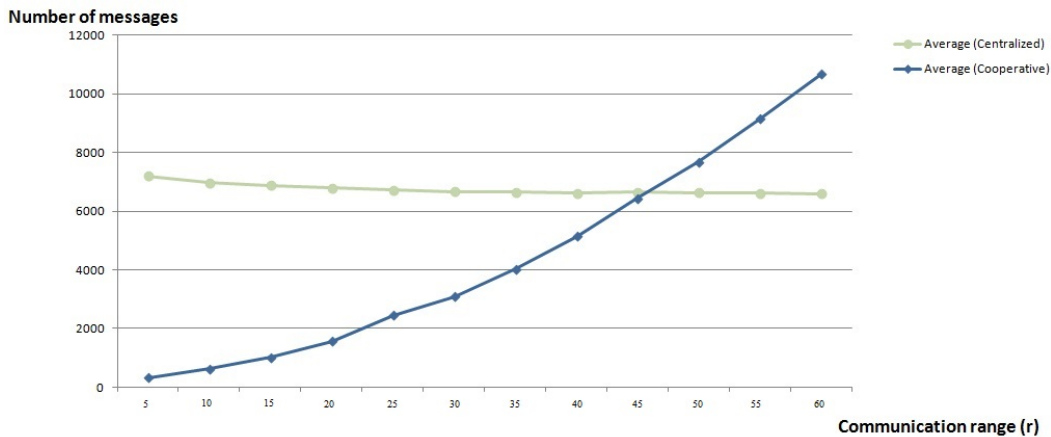


Figure 3.10: Number of messages (centralized vs. cooperative model)

3.4 Multi-Agent Approach for Dynamic VRPTW

Several operational distribution problems, such as the deliveries of goods to stores, the routing of school buses, the distribution of newspapers and mail etc. are instantiations of NP-Hard theoretical problems called the Vehicle Routing Problems (VRP). In its original version, a VRP is a multi-vehicle Traveling Salesman Problem: there exists a certain number of nodes to be visited once by a limited number of vehicles. The objective is to find a set of vehicles' routes that minimizes the total distance traveled. Besides their practical usefulness, the VRP and its extensions are challenging optimization problems with academic stimulating issues. One of the most widely studied variant of the problem is the time and capacity constrained version: the Vehicle Routing Problem with Time Windows (VRPTW henceforth) [Solomon, 1987]. Each traveler provides the following information: a node of the network, two temporal bounds between which he desires to be visited (his time window), a service time s , specifying the time needed for a vehicle to load the traveler before to depart for the next destination and a quantity q (number of goods to receive, number of persons to transport, etc.). Every vehicle has a limited capacity Q , which should not be exceeded by the sum of the quantities associated with the travelers he visits. The addition of time windows to the basic vehicle routing problem restrains considerably the space of valid solutions.

The performance criteria of VRPTW solutions are in general (following this order):

1. The number of mobilized vehicles,
2. the total distance or time traveled,

The VRP and the VRPTW can be divided in two categories: static problems and dynamic problems. The distinction between these two categories relies traditionally on the knowledge (static problem) or the ignorance (dynamic problem) before the start of the solving process of all the travelers that have to be visited. The operational problems are rarely fully static and we can reasonably say that today a static system cannot meet the mobility needs of the users. Indeed, in operational settings, and even if all the travelers are known in advance (before the execution starts), there always exists some element making the problem dynamic. These elements include breakdowns, delays, no-shows, etc. It is thus always useful to consider a problem that is not fully static.

3.4.1 An OLRA Modeling of the Dynamic VRPTW

An OLRA model of the Dynamic VRPTW is provided in Figure 3.11. In the dynamic VRPTW, the set of resources \mathcal{R} is homogeneous, and composed of the travelers. At the start of execution, \mathcal{R} might be equal to \emptyset , and is enriched by the new coming travelers. The set of consumers \mathcal{C} is composed of the vehicles. G is the transportation network of the considered town, region or neighborhood. The nodes of the network represent crossroads or pickup points. The time horizon T is the considered timeframe for the execution. The function $availability(r, t)$ returns 1 if t belongs to the time window of the traveler r and if the traveler is not yet assigned to a vehicle. $availability(c, t)$ returns 1 from the moment when a vehicle c is available until the end of his route.

The properties of a traveler are his node p_{node} ($\Delta_{node} = V$), his time window p_{tw} ($\Delta_{tw} = \mathbb{N}_+^2$), the service time p_s ($\Delta_s = \mathbb{N}_+$) and the quantity associated to his transportation request p_q ($\Delta_q = \{1, \dots, Q\}$). The vehicle has a property remaining capacity $p_{capacity}$ ($\Delta_{capacity} = \{1, \dots, Q\}$). The *compatibility* function checks if the traveler can be inserted in the vehicle's route, provided the already inserted travelers. Among the travelers matching these conditions, the vehicle uses his utility μ to sort them following a preference criterion, for instance from the nearest to the furthest.

In this problem, the travelers can be taken by anyone, but not more than one vehicle can take a traveler. As a consequence, the parameter κ is equal to 1. In addition, several travelers can be colocated with a vehicle (depending on his capacity), and one vehicle can take more than one traveler at the same time. The parameter ζ is then equal to Q (the vehicles capacity).

3.4.2 Multi-Agent Solution for the Dynamic VRPTW

3.4.2.1 Overview

The multi-agent system that we propose is composed of a dynamic set of traveler agents and vehicle agents which interact to solve the dynamic VRPTW. A solution consists of a series of vehicles routes. Each route contains a sequence of travelers with their associated

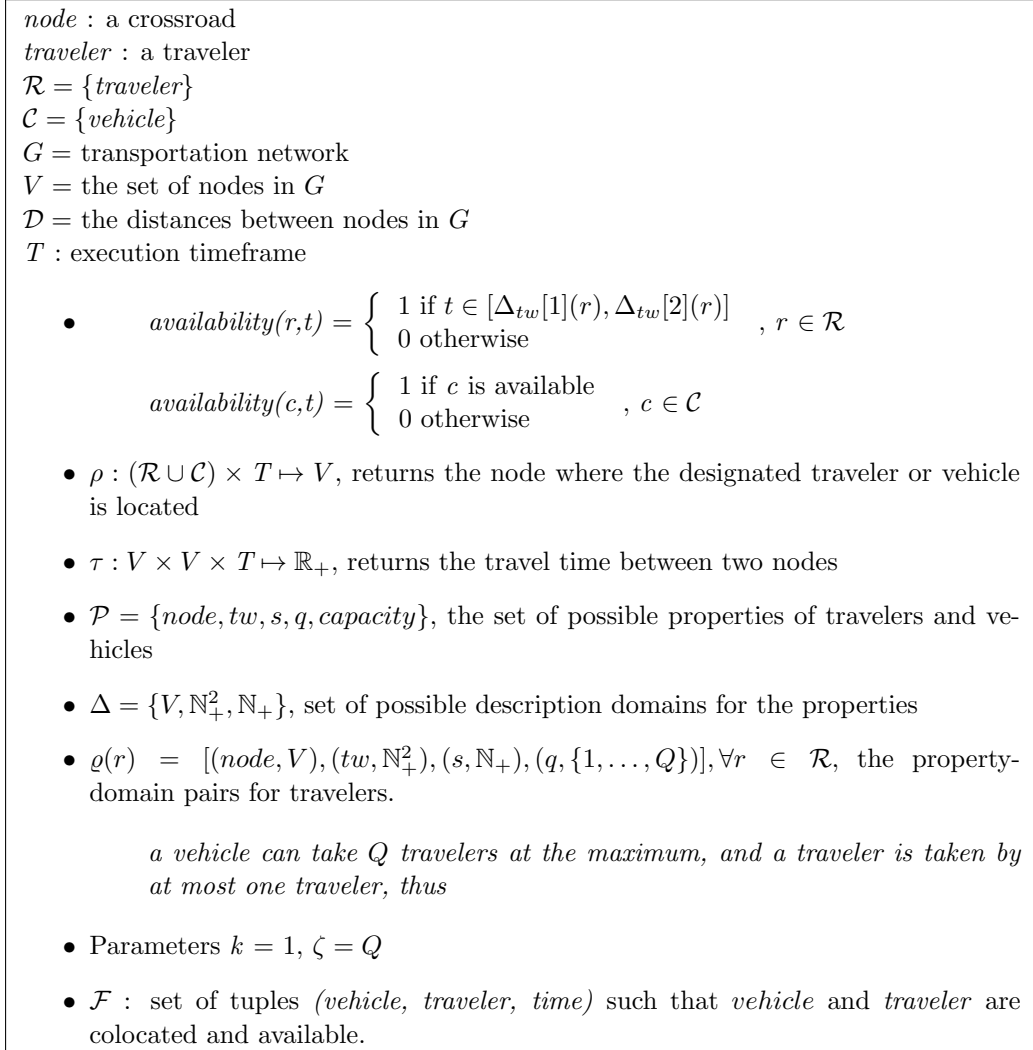


Figure 3.11: An OLRA Modeling of the Dynamic VRPTW

visit time. These routes are given by $\gamma(c, r, t)$ which indicates which vehicle visits which traveler and when.

When a user logs in the system, the provided data are verified (existing node, valid time windows, etc.) and, if the data are correct, a traveler agent representing him is created. The vehicle agents then send insertion proposals to the traveler agent, who collects bids chooses the one offering the best insertion price. If there is no vehicle agent that can insert the traveler, a new vehicle is created and will be the only bidder. Finally, the traveler agent informs the vehicles about his choice.

3.4.2.2 The Problem with insertion heuristics

The protocol that we just described is actually a distributed version of the so-called “insertion heuristics”. Insertion heuristics are methods that consist in inserting the travelers following their appearance order in the routes of the vehicles, with no further reconsideration. These heuristics are very promising in dynamic settings because they are the fastest. The question is now to define the criteria to choose the best vehicle candidate for the insertion of the new traveler, i.e. the insertion price that is proposed by the vehicles to the travelers.

The systems that are based on insertion heuristics use generally the measure of Solomon [Solomon, 1987] as an insertion price. This measure consists in inserting the traveler which has the minimal impact on the general cost of the vehicle (which is generally function of the vehicle’s incurred detour). In these proposals, the utility function $\mu(c, r, t)$ is based on the marginal cost of r in the route of c at time t . This measure is simple and the most intuitive but has a serious drawback, since inserting the current traveler might make lots of future travelers’ insertions infeasible, with the current number of vehicles. Its problem is that it generates vehicles’ plans that are very constrained in time and space, i.e. plans that offer a few possibilities of insertion between each pair of successive planned travelers. In this situation, the appearance of a new traveler might oblige the system to create a new vehicle to serve him.

Through the multi-agent environment modeling presented in the following section, we propose a new insertion price that should palliate this shortcoming.

3.4.2.3 Multi-agent environment

We model the MAS environment in the form of a space-time network (cf. Figure 3.12), inferred from the network graph. As for the traveler information application in chapter 2, each node of the space-time graph is a pair $\langle node, time \rangle$, which represents the state of the *node* in a discrete *time* period. This time, the temporal copies of the transportation network are the same.

This environment is a main interlocutor of the vehicle agents and traveler agents. Vehicle agents subscribe to the space-time nodes that they can visit, provided their current route. Conversely, each space-time node stores which vehicles can visit it. When a traveler appears, asking to be visited by a vehicle at node *node* in the time window $[time_1, time_2]$, he places his request in the corresponding space-time nodes: $(node, time_1), \dots, (node, time_2)$. Only

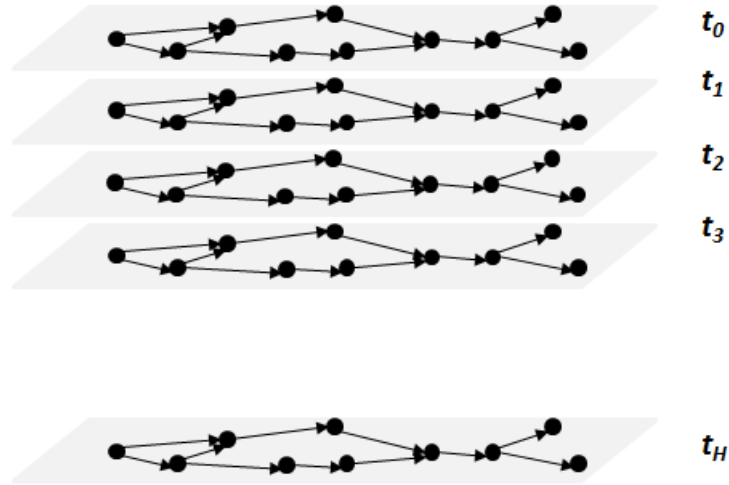


Figure 3.12: Spatiotemporel graph

the vehicles who have subscribed to these nodes will receive the traveler request. Indeed, it is useless for the other vehicle agents to perceive the new traveler, since their current route does not allow them to insert him. Hence, the first advantage of the use of the multi-agent environment in the form of a space-time graph is the same as chapter 2: to limit agents communication cost the most efficiently possible.

The second advantage of the use of the environment is related to the computation of insertion prices by the vehicles. Indeed, we propose a new way to compute the traveler's insertion price in the route of a vehicle, and a new choice criterion between vehicles for the insertion of a given traveler. The method allows the traveler to choose the vehicle agent who maintains the highest probability to participate in future insertions. The logic of our model is different from the traditional models, which focus on the increase of the cost, neglecting the impact of the current insertion decision on future insertion possibilities.

3.4.2.4 Space-time insertion price

Our proposal incites vehicle agents to cover a maximal space-time zone of the transportation network, avoiding the mobilization of a new vehicle if a new traveler appears in an uncovered zone. In the context of the dynamic VRPTW, maximizing the space-time coverage of vehicle agents results in giving the maximum chance to satisfy the demand of a future (unknown) traveler.

Consider a vehicle agent that has an empty route. Since he did not insert any traveler yet, this vehicle has no constraints and can go in any direction. Hence, he subscribes to all the space-time nodes illustrated by the triangular shadow¹ in the Figure 3.13, which forms his *action zone*. He could insert any traveler who asks to be visited in these space-time nodes. The vehicle notifies these nodes that they are reachable by him. At each notification from a vehicle agent, every node updates its list, containing the vehicle agents that are covering it. The action zone of the vehicle agent form an approximation of his \mathcal{F}

¹it is actually a conic shadow in a three-dimensional space.

defined in OLRA, assuming that a resource (a traveler) might appear in any of the nodes of the network.

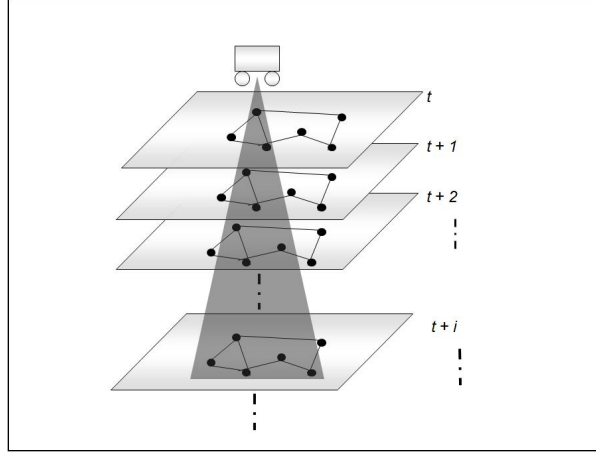


Figure 3.13: Initial action zone of a vehicle agent

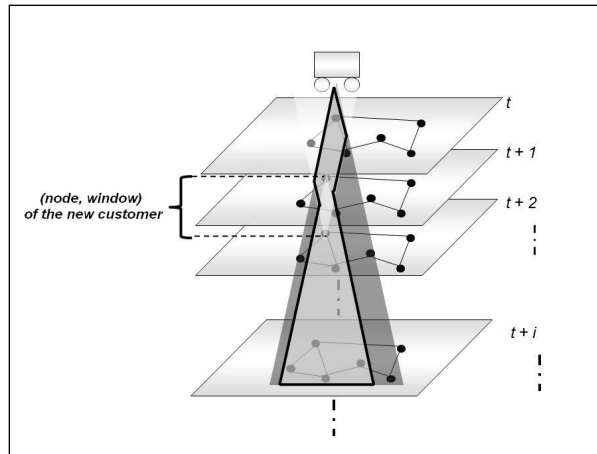


Figure 3.14: Action zone after the insertion of a traveler

When a vehicle agent inserts a traveler in his route, there are some parts of the space-time network that he cannot visit anymore, without violating the traveler's time window. Some $\langle node, time \rangle$ pairs become not valid because of this insertion. In Figure 3.14, a new traveler is inserted in the route of the vehicle. The space-time zone covered by the vehicle agent after inserting the traveler is represented by the interior of the contour of the bold lines, which represent the space-time nodes which remain accessible after the insertion of the traveler.

The vehicle agent interrogates each of the nodes that would be uncovered about the "price to pay" if he were not covering it anymore. This price is inversely proportional to the number of vehicles covering this node. More precisely, the price to pay for each space-time node $\langle node, time \rangle$ is equal to

$$\frac{1}{|vehicles_{\langle node, time \rangle}|} \quad (3.3)$$

with $vehicles_{\langle node, time \rangle}$ denoting the vehicle agents covering the space-time node $\langle node, time \rangle$ and $|vehicles_{\langle node, time \rangle}|$ the number of these vehicles.

The insertion price proposed by the vehicle agent to the traveler agent is equal to the sum of the prices associated with the space-time nodes that would not be covered anymore after the insertion. The idea is that the chosen vehicle for the insertion of a traveler is the one that loses the minimal chance to be candidate for the insertion of future travelers. The space-time network being the only entity knowing the action zones of all the vehicle agents (with the lists of vehicles associated with the nodes), it associates more or less penalty with the decisions of non-coverage of the network by the vehicles. Thus, the vehicle agents are incited to cover the whole space-time network in a coordinated way. Thus, the criterion that is maximized by the society of vehicle agents is the union of their action zones, i.e. the capacity of the MAS to react to the appearance of traveler agents, without mobilizing new vehicles.

3.4.3 Experiments

The main two hypotheses to be tested are related to the two main optimization criteria of the VRPTW: the fleet size and the total traveled distance.

Hypothesis 4 *A system using space-time insertion price mobilizes less vehicles than a system using Solomon insertion price.*

Hypothesis 5 *With space-time insertion price, the total distance traveled by the vehicles is less important than with Solomon insertion price.*

3.4.3.1 Setup

Marius M. Solomon [Solomon, 1987] has created a set of different static problems for the VRPTW. It is now admitted that these problems are challenging and diverse enough to compare with enough confidence the different proposed methods. In Solomon's benchmarks, six different sets of problems have been defined: C1, C2, R1, R2, RC1 and RC2. The travelers are geographically uniformly distributed in the problems of type R, clustered in the problems of type C, and a mix of travelers uniformly distributed and clustered is used in the problems of type RC. The problems of type 1 have narrow time windows (very few travelers can coexist in the same vehicle's route) and the problems of type 2 have wide time windows. Finally, a constant service time is associated with each traveler, which is equal to 10 in the problems of type R and RC, and to 90 in the problems of type C. Short service times would represent problems where the loading and unloading of the transported entities is fast (transportation of persons for instance). In every problem set, there are between 8 and 12 files containing 100 travelers each.

We choose to use Solomon benchmarks, while following the modification proposed by [Gendreau et al., 1999] to make the problem dynamic. To this end, let $[0, Sim_T]$ the simulation time. All the time related data (time windows, service times and travel times) are multiplied by $\frac{Sim_T}{time_{max} - time_{min}}$, with $[time_{min}, time_{max}]$ the minimum and maximum values of times windows of the problem. The authors divide the travelers set in two subsets, the first subset defines the travelers that are known in advance, and the second the travelers who appears during execution. We do not make this distinction, and we consider no

Problem	Distance	Space-Time
	Fleet	Fleet
R1 25 travelers	64	53
C1 25 travelers	34	31
R1 50 travelers	107	92
C1 50 travelers	60	53
R1 100 travelers	181	150
C1 100 travelers	121	108

Table 3.1: Results summary (criterion: fleet size)

travelers known in advance. For each traveler, an appearance time is associated, defining the moment when the traveler is known by the system. Given a traveler, the appearance time that is associated is generated randomly between $[0, \overline{time}]$, with:

$$\overline{time} = p_{tw}[1](traveler) \times \frac{Sim_T}{time_{max} - time_{min}} \quad (3.4)$$

It is known that the behavior of insertion heuristics is strongly sensitive to the appearance order of the travelers to the system. For this reason, we do not consider one appearance order only. We launch the process that we have just described ten times with every problem file, creating this way ten different versions of every problem file.

We have implemented two MAS with almost the same behavior, the only difference concerns the insertion price proposed by vehicle agents to traveler agents. The first MAS relies on the Solomon measure (noted Distance) and the second relies on the space-time model (noted Space-Time). We choose to run our experiments with the problems of class R and C, of type 1, which are the instances that are very constrained in time (narrow time windows).

3.4.3.2 Results

For each problem class and type, we have considered different travelers numbers in order to verify the behavior of our model w.r.t to the problem size. To this end, we have considered successively the 25 first travelers, the 50 first travelers, and finally all the 100 travelers contained in each problem file. Table 3.1 summarizes the results concerning the fleet size. Each cell contains the best obtained results with each problem class (the sum of all problem files). The results show, with the two classes of problems, that the use of the space-time model mobilizes less vehicles than the traditional measure, whatever the number of considered travelers. These results validate the intuition of the model, which consists of maximizing the future insertion possibilities for a vehicle agent. Hypothesis 4 is then valid.

The second hypothesis concerns the total distance traveled by all the vehicles. Table 3.2 summarizes the results². With respect to this criterion, the behavior of the space-time model is not always more efficient: it gives better results for the problems C1 with 25 travelers and R1 with 100 travelers, but is dominated by the traditional measure for the others. Hypothesis 5 is not valid.

²In Solomon's benchmarks, there is no unit associated with the distances.

Problem	<u>Distance</u>	<u>Space-Time</u>
	Distance	Distance
R1 25 travelers	6372	6561
C1 25 travelers	3167	3152
R1 50 travelers	12036	12089
C1 50 travelers	6712	7093
R1 100 travelers	17907	17348
C1 100 travelers	16011	16512

Table 3.2: Results summary (criterion: total traveled distance)

The fact remains that our results provide better results than the traditional heuristic, since the primary objective of the problem is to minimize the number of vehicles mobilized by the system..

We believe that the use of the multi-agent environment in the form of space-time networks is very promising in dynamic and online transportation problems. Indeed, the agents representing the transportation demand and the agents representing the transportation supply have a main interlocutor, which is the space-time network. If there is a new demand, it has simply to be properly placed in the environment, the relevant and interested supply will react accordingly. If there is a change in the supply or in the infrastructure, there is no need to inform each agent individually, and to reconsider previous decisions. The changes have simply to be implemented in the space-time network, and the agents will react accordingly.

3.5 Multi-Agent Configuration for the Dial A Ride Problem

Dial a ride is a transportation mode that is a compromise between public transportation and individual taxis. The principle of these systems is to define the itineraries and schedules of the vehicles based on the requests of the users. Travelers are thus provided with relatively cheap door-to-door transportation insofar as they accept to share their ride with others and tolerate a certain detour from their direct trip. A dial a ride problem is an extension of the dynamic VRPTW, defined in the previous section. The dial a ride problem is defined by a set of travelers and a fleet of vehicles with fixed capacity. Each traveler desires to be transported from an origin location to a destination. Travelers generally impose a time window which includes the earliest possible time and the latest possible time they can be either picked up or delivered. A quantity (number of travelers) and a service time is also associated to each traveler, specifying the time needed for a vehicle to load the traveler before to depart for the next destination. As we can see, the dial a ride problem is an extension of the dynamic VRPTW, where travelers define both an origin and a destination, instead of a single location, and two time-windows instead of one. The remaining capacity of the vehicle along his route is also not monotonically increasing as in the dynamic VRPTW, but increases when he takes travelers and decreases when they leave the vehicle. The OLRA modeling of the dial a ride problem is very similar to the dynamic VRPTW, the only difference concerns the presence of two nodes and two time-windows in the travelers properties. We do not provide it here for the sake of concision, and because the focus in

this part is not on the optimization of vehicles routes. Indeed, the problem we are tackling in this section concerns the impact of the competition of several companies for same set of travelers. For the optimization of vehicle routes by the companies, we use state-of-the-art online optimization techniques.

An important problem with current dial a ride services, as organized in the Netherlands for instance, is that the quality of service (QoS) cannot be guaranteed over long periods of time. A strong competition for the right to serve for a period of usually three years promises a reasonable quality at a low price, but has the effect that a company that is too optimistic in the contracting phase receives the assignment, but subsequently cannot meet the quality objectives without incurring serious losses. Heavily penalizing such a company for a low QoS will soon lead to bankruptcy, and therefore an even lower QoS until a new company has been found.

As we saw it with the dynamic VRPTW, QoS is usually not specifically addressed in the allocation of rides. The minimization of company's costs is treated as a primary objective, while imposing a minimal QoS [Cordeau, 2006]. The idea put forward in this work is to let companies compete on QoS on a day to day basis given a price per kilometer that is fixed in advance. Given known results that competition can reduce the total costs, the question is can we use it to improve the QoS instead, and at what costs?

We use auctions in this proposal, but we divert from research on using auctions and other price-based mechanisms for task allocation, because the company that receives the task is not the one proposing the lowest price, but the one that guarantees the highest QoS. Our main hypothesis is that this approach significantly increases the QoS without much additional costs.

To test our hypothesis, we implement the proposed approach as a multi-agent system, we simulate the appearance of travelers, we simulate the bidding and scheduling process of the companies, and we compute the resulting costs and QoS. We compare these results to a single-company setting where the company optimizes costs with and without a guaranteed QoS level. In the proposed multi-agent system, unlike the two previously presented transportation applications, the agents are the travelers and the companies (not the vehicles). The use of multi-agent systems in this application allows to represent the selfish behavior of the companies, the development of their own strategies as well as the autonomous behavior of the travelers who have the objective to maximize their received QoS.

In the multi-company configuration, each company tries to solve the dial a ride problem with a subset of travelers. In a technique called *on-line optimization* the optimal solution is searched for with exact algorithms, but only taking into account that part of the problem that is relevant for the moment [Mahr et al., 2010]. For instance, when searching for the best departure times for a request to insert into a current schedule, only that part of the current schedule that can be influenced by inserting the new request needs to be considered in the solution process. This results in smaller problems as input for exact algorithms, which implies less computation time. In our simulations of the multi-company environment, we apply this online optimization for the insertion of a ride into the schedule of one of the companies.

3.5.1 Multi-Agent System for the Multi-Company Dial A Ride

The multi-agent system is composed of traveler agents and company agents. A traveler submits a transportation demand to all known company agents. Once a company agent receives the demand, he checks whether it is possible to insert it into one of his vehicle schedules. If it is not possible to insert the traveler into one of the vehicle schedules, the company will not place a bid in the current auction. Otherwise, a bid value (an insertion price) is calculated. When the traveler has received all bids, he determines the best one (the highest QoS) and sets the conditions that have to be met by the winning company in serving his demand. The winning company is informed of the determined conditions, and all other companies are sent a message that they have not won the auction. The winning company then inserts the traveler into the schedule of the defined vehicle. We assume that there is always the possibility to have the request served by a taxi company outside the system at a (usually high) so-called reservation price. This is done when no bid is offered below this reservation price. The following sections detail the elements of this process, starting with the computation of the QoS.

3.5.1.1 Bidding service quality

As we explained it for the dynamic VRPTW, the additional costs needed to serve a traveler is usually used as a bid or insertion price [Mes, 2008], and to minimize overall costs, the request is assigned to the vehicle that has announced the bid with the lowest additional costs. In this work, we let the companies compete on the QoS instead. Therefore, the bid value in our setting contains the QoS that a company promises to provide. We define QoS as the ratio of the actual ride time to the direct ride time. For instance, when the time to travel from A to B directly (i.e. with no detours) is equal to 5 minutes, and the vehicle drives from A to B via C , in 7 minutes, then the QoS is $\frac{5}{7}$. For travelers, this measure emphasizes one of their biggest complaints, namely large detours. For companies, this ratio is a measure of how efficiently different rides are combined.

3.5.1.2 Auction on QoS and pre-determined payments

The mechanism that we propose is based on a *reversed sealed-bid second-price auction*, using QoS instead of prices. In such an auction, each bid is private to the company that submits it, and the winner of the auction has to meet the details of the second-highest bid value. The auction is *reversed*, because there are multiple sellers (the companies) and a single buyer (the traveler). This single buyer announces the details of his request, and then the companies can determine a bid value. The winning company is the company that announces the highest QoS, and if multiple companies announce the same highest bid, one of these companies is arbitrarily selected as winner. The request that has been auctioned is allocated to the winning company, which then has to serve the request with the amount promised by the second-highest bidder.

In our setting, the payment for the service is not defined by the auction, but must be set on forehand. We set the payment equal to a *price per kilometer* C_{km} multiplied by the direct distance between the origin and the destination of the traveler. The profit of a company is then defined as the total *income* a company receives from serving travelers

minus the total *costs* needed to serve these travelers. Clearly, C_{km} essentially determines the income of the companies.

Let for each request $(i, j) \in R$ the direct travel time t_{ij} be given. For this problem, we define solutions for two hypothetical situations with complete knowledge of the requests during the day (in advance). We let $OPT(R)$ denote the transportation costs when all rides are optimally combined (in hindsight), and $OPT^{1.0}(R)$ denote the transportation costs when each request is served with a QoS of 1.0.

Proposition 1 *If an auction on QoS is used for multiple companies, and the (fixed) price per kilometer C_{km} is below $\frac{OPT(R)}{\sum_{(i,j) \in R} t_{i,j}}$ few people will be transported. If C_{km} is above $\frac{OPT^{1.0}(R)}{\sum_{(i,j) \in R} t_{i,j}}$, everyone will be transported separately.*

Proof 1 *The lower bound for C_{km} is the minimal total costs needed to serve all requests divided by the total direct distance traveled by all travelers. When C_{km} is set below this value, companies have more costs than income, except when a ride largely overlaps with an existing ride (which is never the case when the schedule is still empty). Companies will thus not bid in the auction. Therefore few people will be transported. On the other hand, when C_{km} is set above the average cost of transporting everyone separately, every company makes a net profit for each traveler. Therefore, every company will bid a QoS of 1.0 for every request, because it then has the highest chance to win the auction. Since this holds for all companies, all requests have to be served with QoS 1.0.*

Therefore, C_{km} should be chosen between these two bounds.

3.5.1.3 Computations for the companies

The computations of the transportation companies are based on online optimization for the insertion of rides, and for bid determination use a look-ahead on possible future travelers via a Monte Carlo simulation in combination with an insertion heuristic. Every company solves a dial a ride problem to find the set of routes for their vehicles. The online optimization model can be found in [Grootenboers et al., 2010] and [Grootenboers, 2010].

Bid calculation: To check whether an incoming request can be inserted, we use the insertion heuristics developed by Jaw et al. [Jaw et al., 1986]. This is done before a bid value is calculated, and can save expensive computation time. If the request is feasible, the company can propose a bid to the traveler.

A company agent wants to maximize his profit, defined as income minus costs. The company agent can gain income by serving travelers (winning auctions) and he can decrease costs by combining travelers. Combining travelers often leads to a lower QoS (because there are less direct rides), which can lead to winning less auctions. Promising a higher QoS results in a higher probability to win the auction, but decreases the flexibility to insert future travelers.

There are different costs associated with the different QoS values that a company can bid. In general, a company can bid a low QoS for low internal costs, or a high QoS for higher internal costs. For a single-shot second-price auction, it can be shown that it is

optimal to bid the highest price possible. However, this is not the case in our (repeated) auction on QoS.

Proposition 2 *If each company bids the highest QoS possible, all rides will be transported at QoS of 1.*

Proof 2 *When a company bids the highest QoS possible for the first traveler, this bid will be a QoS of 1, since all its vehicles have empty routes so far. When all companies follow this behavior, the second “price” will also be a QoS of 1, so the ride is accepted at a QoS of 1. When a route contains only rides with a QoS of 1, a next ride cannot be combined, so the highest QoS possible will also be 1. With induction, all rides will be transported at a QoS of 1.*

To avoid this side effect of using QoS as a bid for the companies, we allow them to incorporate knowledge about future requests in their bid calculation. This way, they reason about the future possible combinations of the current request while bidding the promised QoS, instead of reasoning only about the current request. To incorporate the expected profit of future requests, the companies must have some knowledge about the distribution in time and space of future requests. From this distribution, they can calculate the expected profit for an incoming request, based on future requests that can give the companies possibilities to combine rides and lower costs. To this end, we use a Monte Carlo simulation in combination with an insertion heuristic.

Estimating expected profit: The idea is to estimate the *expected profit* that a company would make assuming that the current request is inserted into the schedule. To calculate the expected profit, a distribution has to be known on the arrival location and time of future travelers. With the help of these distributions, a set of possible future requests can be generated and inserted into the schedule. Once this is done, the total costs needed to insert these requests, and the total income gained by inserting them can be calculated. This expected profit is calculated by using an insertion heuristic based on the works of Jaw et al. [Jaw et al., 1986].

Monte carlo simulations: The specific set of generated future requests can have a big influence on the calculated expected profit. Therefore, a Monte Carlo simulation [Metropolis and Ulam., 1949] is performed. The above algorithm is repeated a number of times, and the final expected profit is taken as the average expected profit of these repetitions. To obtain the highest QoS level for the current request, taking into account future requests, Monte Carlo simulations are performed for different levels of QoS. The level for which the expected profit is closest to zero is taken as the bid value.

3.5.2 Experiments

The main hypothesis to be tested is the following.

Hypothesis 6 *When multiple companies compete on QoS, the average QoS is higher than in a situation with a single company which minimizes costs. Transportation costs are also higher.*

But also for a single company we expect that transportation costs are higher when the QoS is higher:

Hypothesis 7 *A higher required QoS is more expensive (for a single company).*

As discussed before, requiring a higher QoS from a single company fails in practice, because in general, a higher QoS is more expensive, and penalties are not sufficient to incentivize a company to meet the agreed QoS. Our main thesis is that competition can be used to realize a higher QoS, and we expect that this can be done at approximately the same additional cost as for a single company, leading to a third hypothesis.

Hypothesis 8 *When multiple companies compete on QoS, the costs are not significantly higher than in a situation with a single company which minimizes costs with the same average QoS.*

3.5.2.1 Setup

To test these hypotheses, we run the mechanism and algorithms proposed in the previous section on a set of benchmarks, that we have generated. The considered network is a continuous map, defined by a square area of 20 by 20 km with a node on every km. The size of these benchmark problems is chosen such that each one takes at most about 1 hour computation time to solve. Indeed, unlike the two previous dynamic transportation applications, there is a complex computation performed by the agents (the company agents), which is the online optimization part of their work. To decide how many travelers we should have in each instance, we solved static versions of the dial a ride problem (all the travelers known in advance), and verified the incurred execution times. Figure 3.15 reveals that with 16 travelers, execution exceeds one hour.

As a consequence, we decided to define 100 problem instances containing 16 travelers each. The origins, destinations and time windows of the travelers are distributed following a uniform distribution. We consider these instances over a planning period of 4 hours.

To make the problem instances dynamic, as for the dynamic VRPTW, we add to each traveler the moment at which he becomes available to the system. This is done by randomly choosing a number in the interval $[p_{tw}[1](traveler) - 90, p_{tw}[1](traveler) - 60]$ (i.e. between 90 and 60 minutes before the earliest pickup or delivery time). We choose these values because we want the instances as dynamic as possible. This means that travelers announce their requests quite late, but such that vehicles are able to respond to schedule changes. The maximum capacity for each vehicle is set to 3 passengers. All the experiments have been performed in Java and Java Agent DEvelopment Framework (JADE) [Bellifemine et al., 1999] on an Intel Xeon E5345 2.33GHz with 16 Gb RAM. Each company uses the MIP-solver SCIP to insert assigned requests. SCIP is one of the fastest non-commercial solvers [Achterberg, 2004].

To test the hypotheses, we run the system two times for each problem instance. The first time two companies have two vehicles each and compete on QoS. The second time, there is only a single company, having four vehicles and minimizing costs. In a single-company setting, the company does not have any incentive to bid high QoS, because it knows already that it contractually gets assigned all the requests.

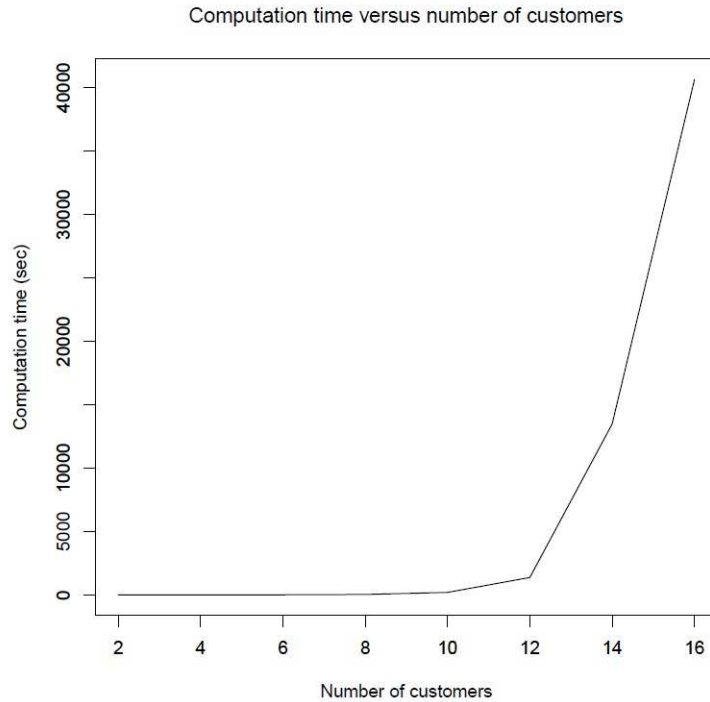


Figure 3.15: Computation time of the static problem instances

First experiment: In Figure 3.16, a plot is given for the comparison of QoS and total costs between a situation with a single company and a situation with multiple companies. Two clouds of points can be distinguished. The cloud of points of instances with multiple companies (\circ and $+$) is situated with relative high QoS and high total costs. The other cloud of points (Δ and \times) has less quality and less total costs and mainly contains instances with a single company. A paired t-test is performed for both average QoS and total costs. The mean difference for QoS is 0.097 with a higher quality in the multi-company setting. The confidence interval is $[0.071, 0.122]$ and the probability that these results are obtained assuming that there is no difference between the two settings is 5.98×10^{-10} . A t-test gives us a mean difference of 37.5 higher total costs for the multi-company setting with a confidence interval of $[25.3, 49.8]$, and a p-value of 1.25×10^{-7} .

In conclusion, we have discovered that total costs are about 13% higher in the multi-company setting than in the single-company setting. The fact that companies in a multi-company setting have the incentive to bid higher service quality instead of minimizing costs, results in a higher average service quality. This follows from the results of the paired t-test showing that the QoS of the multi-company setting is 12% higher. Both differences are (very) significant, confirming Hypothesis 6.

Second experiment: In the previous experiment, the single company did not take any required level of QoS into account. However, to make a fair comparison of costs (to establish the hypothesis 7), we would like to have the same average QoS for the single company as the multiple companies obtain by competition. To arrive at a certain average QoS, we ensure that each ride of the single company has a certain minimal QoS. To determine how to set the required QoS to arrive at such a desired *average* QoS, we first run the experiments

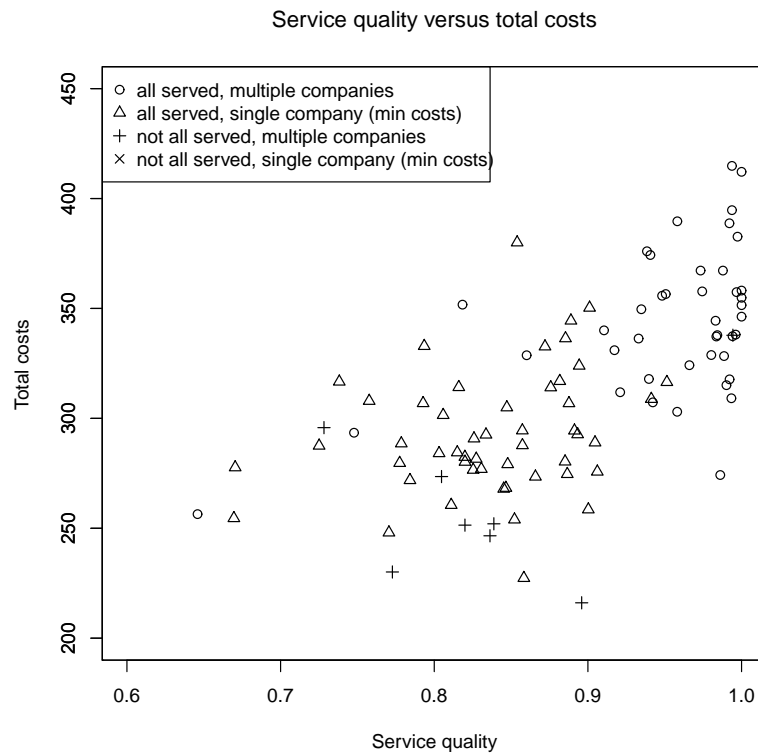


Figure 3.16: QoS versus total costs for a single and a multiple company setting

for a single company for a required QoS of $\{0.00, 0.05, \dots, 1.00\}$. We then investigate the relation between these required QoS levels and the average QoS, and at the same time the relation between the required QoS levels and the total costs, to test Hypothesis 7.

In Figure 3.17, it is shown that the average total costs are increasing as from a QoS level of 0.4, with a slight decrease at level 1.0. This last decrease of average total costs can be interpreted by the fact that if the minimal QoS level is 1.0, less requests can be served, which leads to lower total costs. The non-increasing part of the figure (from level 0.0 to 0.4) can be explained by the fact that these QoS levels do not influence the outcome, because even when a single company minimizes costs, it still serves requests with an average QoS of about 0.66. This is also shown in Figure 3.18, in which we see no increase in QoS at this interval.

Besides these two exceptions, overall there is a strong correlation between total costs and minimal QoS. The correlation coefficient over the complete range 0.93, confirming Hypothesis 7. When we take only the interval from 0.40 to 0.95 into account, the correlation coefficient is even 0.99. The minimal QoS level that is needed in order to let the company in the single-company setting serve requests with an equal average QoS as in the multi-company setting is derived by searching in Figure 3.18 for the corresponding level. The average QoS over all instances in the multi-company setting is 0.93 and when we search for the corresponding minimal QoS level we find a value of 0.77.

Subsequently, we run the first experiment again, but now requiring a QoS of 0.77 for the single company. In Figure 3.19 a scatter plot is shown in which QoS is plotted against total costs, for the multi-company and single-company setting. From this we observe that the

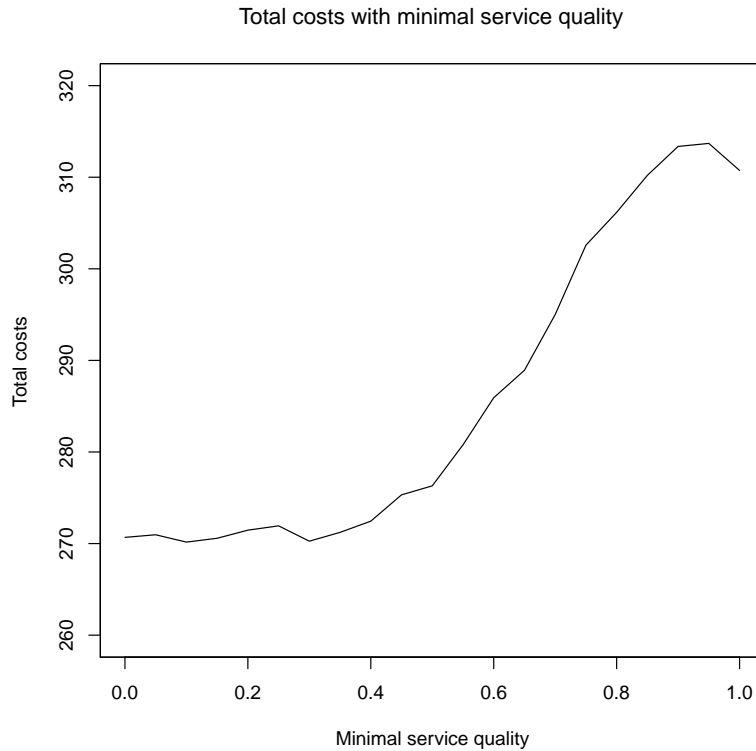


Figure 3.17: Average total costs for instances with a minimal QoS

costs are somewhat higher in the multi-company setting, but the points in the plot are too close to each other to give a proper judgment about this measure. The paired t-test gives us a mean difference of 23.5 total costs, a 95%-confidence interval of [11.9, 35.0], and a p-value of 1.71×10^{-4} , with higher total costs in the multi-company setting. Another paired t-test is performed to verify that the difference in average QoS between the two settings is not significant. The results of this test confirm this with a mean difference of 0.0013 (higher in the multi-company setting), and a p-value of 0.92.

We thus conclude that compared to the multi-company setting, the total costs in a single-company setting are less, even if this single company provides equal QoS, rejecting Hypothesis 8.

As a conclusion, we conclude that if the price per kilometer is fixed within a reasonable range (Proposition 1), and expectations about the future are somehow taken into account (see also Proposition 2), it is indeed possible to obtain a higher QoS in door-to-door transportation by letting multiple companies compete on QoS (Hypothesis 6). However, in our experiments, the costs are about 7% higher than in the idealistic case where a single company always meets a required QoS while minimizing costs (which is not verified in nowadays systems).

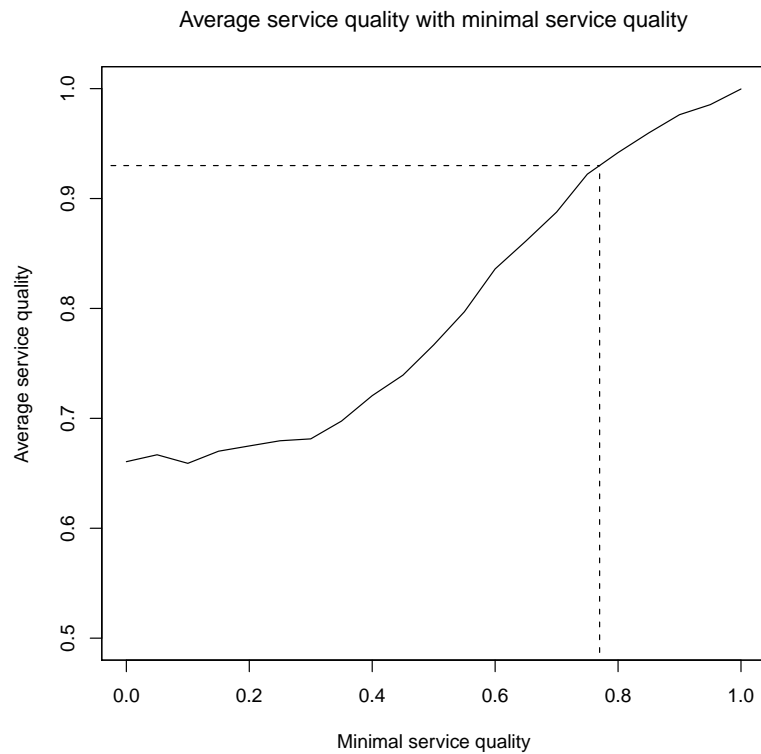


Figure 3.18: QoS for instances with a minimal QoS

3.6 Conclusion and Perspectives

In this chapter, we have proposed a modeling for the resources allocation problem taking into account simultaneously the location and the moment when the resources are available. This modeling is well adapted to the transportation domain where many applications are characterized by the difficulty to take into account their space-time dimension. Our modeling is able to take into account several kinds of constraints: i) space constraints: the resources or consumers have static or dynamic positions?; ii) time constraints: do the availability of the resources and needs of consumers change over time?; iii) space-time constraints: do the resources and the consumers have to be at the same location?

We have used this modeling to specify three dynamic transportation applications: the management of parking spots in an urban area, the dial a ride problem and the dynamic VRPTW. To each of these applications, we have proposed a multi-agent system to solve the problem. For urban parking, the system is based on a community of drivers that interact to keep up-to-date information regarding the availability of parking spots. For the dynamic VRPTW, the multi-agent system is composed of vehicles inserting travelers as they come using a new measure based on a space-time representation of the environment. For the dial a ride problem, the system is composed of multiple companies competing to serve the travelers, and companies use online optimization to insert the travelers.

We have two main perspectives for this work. On the one side, we plan to use the space-time multi-agent environment with many other dynamic transportation applications.

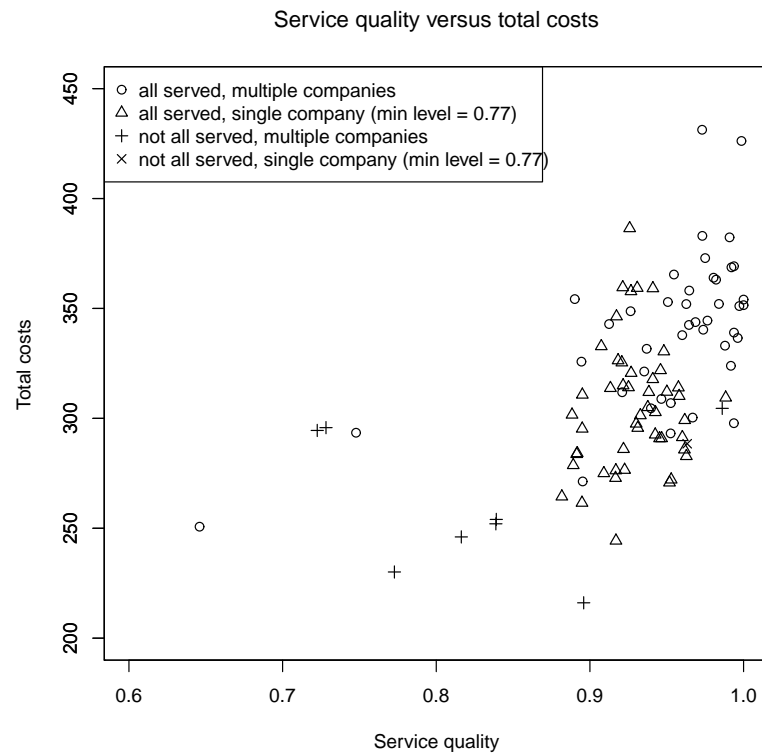


Figure 3.19: QoS versus total costs for a single and a multiple company setting with a minimal QoS

One of these applications concerns the online evacuation management in crisis situations. In this application, a crisis situation impacts a region and its transportation network, and the population in some zones has to be evacuated to safe zones. The zones and the people to evacuate are known progressively as the predictions about the risk become more and more accurate. The use of the multi-agent environment in this context is promising, since it allows for the consideration of new people to evacuate and the parts of the network that become unsafe, without losing the optimization effort already engaged.

On the other side, we plan to consider the second important source of dynamism in transportation applications, which is the traffic status and congestions. Indeed, in the considered dynamic VRPTW and in the dial a ride problem, the dynamism comes from the unknown travelers but the network traffic state is supposed stable over time. Traffic dynamics was the purpose of the previous chapter, and we plan to couple our proposals in simulation with the approaches presented in this chapter, to consider traffic dynamics in the optimization.

Conclusion and Perspectives

“What we term history does not represent the sum-total of all conceivable things that have been done in space and time; history comprises those small illuminated sections of world happenings which have had thrown upon them the light of poetical or scientific description.”

Stefan Zweig, The Story of Magellan (1938)

Contents

4.1 Summary of the Research Work	93
4.2 Perspectives	94
4.2.1 Integrating Autonomous Transportation	95
4.2.2 Designing Smart Systems	95
4.2.3 Optimizing New Mobility Services	95
4.2.4 Merging Modeling, Simulation and Optimization	96
4.2.5 Distributing Multi-Agent Environments	96
4.2.6 Modeling Multi-Scale Mobility	97
4.2.7 Coupling Simulation and Regulation	97

4.1 Summary of the Research Work

Our work is motivated by the development of models, simulations and algorithms to support the multimodal transportation actors solving nowadays and future complex and dynamic problems. Our work contributes to a better understanding of the problems raised by the new mobility services (dial a ride, ride sharing, urban parking search systems, etc.) and to a better design of traveler information systems, integrating these new modes. This work has a unifying theme: dynamic transportation problems. It has been conducted with a unique approach: the computer systems modeling following the multi-agent paradigm.

The solving of static transportation problems, such as the static traffic assignment problem, the static traveling salesman problem and the static vehicle routing problem is very important in the planning and the dimensioning of transportation supply (infrastructure and/or vehicles fleets). It is however less efficient in operational settings and for dynamic regulation purposes. Dynamic transportation problems tackle these kinds of problems, and refer to a wide range of transportation problems where the problem data are not available

a priori. The missing or incomplete information concern the supply, the demand or the transportation infrastructure and become progressively available during execution.

In modern operational settings, transportation systems are increasingly composed of several decision centers, and some system components that are located in the transportation environment (vehicles, drivers, control entities, etc.) are equipped with computational power that could allow them to react to events in that environment. For these reasons, we have chosen the multi-agent systems paradigm to address these problems.

All our proposals in the multi-agent systems domain have been applied to dynamic transportation applications, and all the tackled dynamic transportation problems have been modeled with multi-agent systems. We have defined a specification model to design and implement open multi-agent systems that we apply to dial a ride systems and travelers information (cf. my PhD thesis [Zargayouna, 2007]). We have designed and implemented a multi-agent simulation platform that we have applied to multimodal traffic representation, and distribution models that we have applied to multi-agent traffic simulations (chapter 2). We also have defined a multi-agent model solving the urban parking problem. We have designed a multi-agent configuration to solve the multi-company dial a ride problem and we have specified a multi-agent space-time organization model that we have applied to dynamic vehicle routing problems (chapter 3).

One main design choice has greatly influenced the work presented in this document. It concerns the explicit representation of the multi-agent environment. Our first proposal in this context was the use of the environment to support the interaction and to coordinate the agents of the multi-agent system. This proposal was general-purpose and applicable to any open multi-agent system. When dealing with dynamic transportation applications, we have adapted this environment-centric principle and applied it to the transportation network. In the traveler information application (cf. chapter 2) and in the dynamic VRPTW (cf. chapter 3), a shared space-time representation of the transportation network is used for information and dynamic planning purposes. Some applications are not suitable for this kind of environment representation. For instance, in the community-based urban parking management (cf. chapter 3), vehicles interaction is very local, and vehicles can only communicate with others that are co-located in space and time with them. In these conditions, using a central communication environment or a space-time environment is not useful. In this kind of applications, and when designing general-purpose traffic simulations, the multi-agent environment is simply made of the transportation network, and is used as a geographical reference for the agents.

4.2 Perspectives

In this section, we draw the most structuring perspectives for our research work. Three main tendencies seem to shape the next decades in transportation systems, and our work plan is greatly influenced by them: autonomous transportation, Internet of Things and new mobility services. We are also planning for an evolution of our proposals toward more integration and more coupling with the dynamic regulation of transportation systems.

4.2.1 Integrating Autonomous Transportation

Autonomous transportation is mobilizing a continuously increasing number of researches and industrial projects. Autonomous transportation could have a dramatic (but gradual) effect on multimodal traffic. Some think that autonomous transportation would make congestion a “thing of the past”¹. However, some studies [Smith, 2012] predict that these systems would be subject to a transportation demand that still exceeds the capacity. That means that simulating and optimizing these new transportation systems would still be necessary. In addition, several decades are needed to envision a transportation system exclusively composed of autonomous vehicles. In the meantime, it will be necessary to model, to simulate and to optimize hybrid systems, where autonomous vehicles coexist with connected and non connected vehicles. The agent-based modeling, with its individual-based representation, is a candidate of choice to model these heterogeneous systems. As a first step in this direction, we are planning to extend SM4T to consider autonomous vehicles (both private vehicles and public transportation vehicles) along with connected and non-connected vehicles and travelers.

4.2.2 Designing Smart Systems

Mobile devices and Internet of Things are getting more and more omnipresent, especially in urban areas. In the Internet of things paradigm, an enormous number of small objects will be accessible on the network in one way or another. This results in the creation of very large quantities of data to be stored, processed and presented in an efficient form. The presence of these new sensors offers new possibilities for transportation applications. We could think of, for instance, a guidance system that guides a disabled traveler through the available adapted facilities, taking into account their real-time status. The internet of things is also likely to replace the existing sensor networks of vehicle detectors such as inductive loops [Gubbi et al., 2013] and provide a wider coverage of the networks. The presence of great number of individual information sources advocates for the use of a relevant paradigm, and multi-agent systems are one of these candidate systems. Our work in this direction starts with the definition of a mobility simulation that considers all the available data from local sensors (crowd detection, elevators and escalators status, etc.) and represents travelers that are guided using preferences about these sensors.

4.2.3 Optimizing New Mobility Services

New mobility services are reshaping both public transportation and private vehicles use. Indeed, the growing demand pressure on urban transportation is producing a shift toward shared mobility services that would increase the transportation system efficiency. The concept of “Mobility as a Service” offers door-to-door transportation and decreases the need for private vehicle ownership. The term mobility as a service stands for buying mobility services that are based on travelers needs instead of buying the means of mobility [Kamargianni et al., 2016]. With the increased market penetration of the new mobility services, their impact on traffic would increasingly be significant. The optimization of these

¹<http://foreignpolicy.com/2011/08/15/micromultinationals-will-run-the-world/>, last visited 8 Oct. 2018.

services would be an important part of multimodal traffic management. Our first objective in this direction is to continue working on new mobility services optimization with the multi-agent paradigm (cf. chapter 3), and to merge it with the multimodal traffic simulation (cf. chapter 2).

4.2.4 Merging Modeling, Simulation and Optimization

In the near future, we are planning to work on the convergence of the different proposals of this document in modeling, simulation and optimization. First, we are planning to implement the urban smart parking application presented in chapter 3 in SM4T. That will allow for a large scale simulation and evaluation of the algorithms that we proposed. Second, we are planning to implement libraries allowing to use the multi-agent environment-based model of LACIOS (cf. chapter 1) in the simulations. This will allow the agents in SM4T to use one of the four following interaction modes:

- direct interaction with addressed messages
- indirect interaction via the spatial environment
- indirect interaction via the space-time environment
- indirect interaction via the generic environment *à la* LACIOS.

We also plan to use SM4T in the context of the dynamic vehicle routing problems presented in chapter 3. Indeed, dynamic traffic is generally not considered in vehicle routing problems. However, the traffic status might impact the respect of travelers time-windows, and might call for replanning of vehicles routes. The simulation in this context can provide the traffic dynamics and be used in conjunction with the routes planning for execution monitoring and routes correction.

4.2.5 Distributing Multi-Agent Environments

The design of the multi-agent environment as an explicit entity is often criticized because it introduces centrality in systems that are supposed to be completely distributed. Following these arguments, centrality could lead to communication bottlenecks, to weak fault tolerance and to poor scalability [Billhardt et al., 2016]. However, as we can see it in the models and applications presented in this document, this architecture has several benefits, and we believe that there is a compromise between the two visions (explicit environment vs. completely distributed systems). In our ongoing work, we develop the idea that we still can benefit from an explicit representation of the multi-agent environment without losing the benefits of distribution, namely fault tolerance and scalability. This is done by splitting the design process in two phases. In the first phase, the environment is designed as a central entity, as in several applications presented in this document. In a second phase, at the implementation level, the environment is automatically split over different computation units. To do so, we will base the distribution on a “natural” clustering of the environment and the agents (that we already used to speed up the matching process of JAVA-LACIOS, cf. [Zargayouna, 2007]) using Galois Lattices.

4.2.6 Modeling Multi-Scale Mobility

The studies on mobility generally concern wide geographic zones and consider extended and multimodal transportation networks. However, very local mobility issues, in the scale of a neighborhood, need particular modeling that allows for the understanding of mobility at this scale. This modeling must be coherent with a dynamic traffic modeling of the region to which the neighborhood belongs. To model and simulate mobility at a smaller scale, we have to ensure that the “zoom” from region to neighborhood scale takes place without information loss and that the resulting simulation remain realistic w.r.t. the models of the concerned region. Our objective in this context is to propose new methods for the modeling and simulation of travels at the scale of a neighborhood.

Understanding how region and city scales articulate with the neighborhood scale will be one of the challenges considered in this work. Two types of simulations are targeted. A simulation representing and managing wide regions mobility: it will play the role of a dynamic environment for the second simulation. The latter, interacting and synchronized with the first simulator has to be capable of representing realistically the movements in the neighborhood (between stations, crossroads, parking, activity zones, etc.). Several types of actors will be represented in these simulations: passengers, pedestrians, drivers, private cars, public transportation vehicles, bikes, etc. The contributions of this work will be based on multi-agent modeling and simulation on the one side, and on traffic modeling and assignment on the other side. In this context, we will build a data model representing multimodal mobility on all the Île-de-France region (12 million residents), including the timetables of all the transportation operators, the networks description (16 metro lines, 10 tramway lines, 14 train lines and 1500 bus lines) and the demand on this region.

4.2.7 Coupling Simulation and Regulation

We are responsible of the Claire-Siti platform, which observes, analyzes and regulates both public transportation networks and road transportation networks. The system connects to the operating systems of transportation operators, which feed it with vehicles timetables and real-time positions, together with real-time traffic variables. The platform exhibits indicators about the status of the networks, allowing the regulators to know which parts of the transportation networks needs attention. Based on these indicators and on a rule-based decision support system defines the actions that have to be performed to improve the network status.

The platform is very useful and unique in its kind. Its development was made possible with the participation of our successive teams on several National and European projects for more than 25 years. However, the regulation platform is not connected with travelers data. At any moment, we know the status of the transportation supply, the vehicles and their positions, but we don't know who is in the vehicles, if they are empty or full. We don't know either who is in the stations of the network. Since the transportation service is made for the travelers, the quality of service of the operators should take into account their experience and their perceived service. Transportation operators nowadays do not have access to the real-time positions of their passengers. As a consequence, Claire-Siti cannot yet provide indicators integrating the travelers experience neither.

This is where our simulation platform SM4T comes into play. Since it represents both

vehicles and travelers mobility, it can feed Claire-Siti with passengers positions, and paves the way toward the definition of travelers-based indicators that would enrich the set of indicators already present in the platform. This way, Claire-Siti would observe and analyze both supply and demand in transportation networks. To this end, SM4T and Claire-Siti have to continuously interact and to be perfectly synchronized, both in time and space. The progression of a simulation has to be perfectly synchronized with the real time progression. The data about the region should be exactly the same in Claire-Siti and SM4T, together with all the events happening in the transportation networks.

Bibliography

- [Abadi et al., 2015] Abadi, A., Rajabioun, T., and Ioannou, P. A. (2015). Traffic flow prediction for road transportation networks with limited traffic data. *IEEE Transactions on Intelligent Transportation Systems*, 16(2):653–662. (Cited on page 14.)
- [Acha et al., 2011] Acha, S., Green, T. C., and Shah, N. (2011). Optimal charging strategies of electric vehicles in the UK power market. *Innovative Smart Grid Technologies (ISGT), 2011 IEEE PES*, pages 1 – 8. (Cited on page 56.)
- [Achterberg, 2004] Achterberg, T. (2004). SCIP - a framework to integrate constraint and mixed integer programming. Technical Report 4-19, Zuse Institute Berlin. (Cited on page 87.)
- [Adacher et al., 2014] Adacher, L., Oliva, G., and Pascucci, F. (2014). Decentralized route guidance architectures with user preferences in urban transportation networks. *Procedia-Social and Behavioral Sciences*, 111:1054–1062. (Cited on page 52.)
- [Badeig, 2010] Badeig, F. (2010). *Un environnement actif pour la simulation multi-agents. Application à la gestion de crise dans les transports*. PhD thesis, Université Paris Dauphine. In French. (Cited on page 10.)
- [Badeig et al., 2008] Badeig, F., Balbo, F., Scemama, G., and Zargayouna, M. (2008). Agent-based coordination model for designing transportation applications. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 402–407. IEEE. (Cited on page 14.)
- [Balbo, 2000] Balbo, F. (2000). *ESAC : Un modèle d’Interaction Multi-Agent utilisant l’Environnement comme Support Actif de Communication. Application à la gestion des transports urbains*. PhD thesis, Université Paris - Dauphine. In French. (Cited on page 7.)
- [Barabási and Albert, 1999] Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512. (Cited on pages 45 and 47.)
- [Barbucha and Jdrzejowicz, 2009] Barbucha, D. and Jdrzejowicz, P. (2009). Agent-based approach to the dynamic vehicle routing problem. In *7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS 2009)*, pages 169–178. Springer. (Cited on pages 4 and 54.)
- [Barceló et al., 1998] Barceló, J., Ferrer, J., García, D., Grau, R., Forian, M., Chabini, I., and Le Saux, E. (1998). Microscopic traffic simulation for ATT systems analysis. a parallel computing version. *Contribution to the 25th Anniversary of CRT*. (Cited on page 39.)
- [Bayless and Neelakantan, 2012] Bayless, S. H. and Neelakantan, R. (2012). Smart parking and the connected consumer: Opportunities for facility operators and municipalities. Technical report, ITS America, Washington, DC 20036 USA. (Cited on page 60.)
- [Bazzan and Klügl, 2014] Bazzan, A. L. and Klügl, F. (2014). A review on agent-based technology for traffic and transportation. *The Knowledge Engineering Review*, 29(03):375–403. (Cited on page 3.)

- [Beasley et al., 2000] Beasley, J. E., Krishnamoorthy, M., Sharaiha, Y. M., and Abramson, D. (2000). Scheduling aircraft landings the static case. *Transportation science*, 34(2):180–197. (Cited on page 56.)
- [Behrisch et al., 2011] Behrisch, M., Bieker, L., Erdmann, J., and Krajzewicz, D. (2011). SUMO - Simulation of Urban MObility - an overview. In *SIMUL 2011, The Third International Conference on Advances in System Simulation*, pages 55–60. (Cited on pages 16 and 37.)
- [Bellifemine et al., 1999] Bellifemine, F., Poggi, A., and Rimassa, G. (1999). JADE - a FIPA-compliant agent framework. In *Proceedings of the Practical Applications of Intelligent Agents*, pages 33–45. (Cited on page 87.)
- [Bellifemine et al., 2007] Bellifemine, F. L., Caire, G., and Greenwood, D. (2007). *Developing Multi-Agent Systems with JADE*. Wiley. (Cited on page 18.)
- [Bessghaier et al., 2012] Bessghaier, N., Zargayouna, M., and Balbo, F. (2012). Management of urban parking: an agent-based approach. In *Artificial Intelligence: Methodology, Systems, and Applications*, pages 276–285. Springer Berlin Heidelberg. (Cited on pages 3 and 62.)
- [Billhardt et al., 2016] Billhardt, H., Fernández, A., Lujak, M., Ossowski, S., Julián, V., De Paz, J. F., and Hernández, J. Z. (2016). Towards smart open dynamic fleets. In *13th European Conference on Multi-agent Systems (EUMAS 2015)*, pages 410–424, Cham. Springer International Publishing. (Cited on page 96.)
- [Bookbinder and Sethi, 1980] Bookbinder, J. H. and Sethi, S. P. (1980). The dynamic transportation problem: A survey. *Naval Research Logistics (NRL)*, 27(1):65–87. (Cited on page 4.)
- [Brakewood et al., 2014] Brakewood, C., Barbeau, S., and Watkins, K. (2014). An experiment evaluating the impacts of real-time transit information on bus riders in Tampa, Florida. *Transportation Research Part A*, 69:409–422. (Cited on page 23.)
- [Cajias et al., 2011] Cajias, R., Gonzalez-Pardo, A., and Camacho, D. (2011). A multi-agent traffic simulation framework for evaluating the impact of traffic lights. In *3rd International Conference on Agents and Artificial Intelligence (ICAART)*, pages 443–446. (Cited on pages 16 and 37.)
- [Cats et al., 2011] Cats, O., Koutsopoulos, H., Burghout, W., and Toledo, T. (2011). Effect of real-time transit information on dynamic path choice of passengers. *Transportation Research Record: Journal of the Transportation Research Board*, 2217:46–54. (Cited on page 24.)
- [Cetin et al., 2003] Cetin, N., Burri, A., and Nagel, K. (2003). A large-scale agent-based traffic microsimulation based on queue model. In *Proceedings of the Swiss Transport Research Conference (STRC), Monte Verita, CH*, pages 3–4272. (Cited on page 41.)
- [Champion et al., 1999] Champion, A., Mandiau, R., Kolski, C., Heidet, A., and Kemeny, A. (1999). Traffic generation with the scanner ii simulator: towards a multi-agent architecture. In *Driving Simulation Conference*, pages 311–324. (Cited on page 37.)
- [Chipeaux et al., 2011] Chipeaux, S., Bouquet, F., Lang, C., and Marilleau, N. (2011). Modelling of complex systems with AML as realized in MIRO project. In *LAFLang 2011, workshop of the Int. Conf. WI/IAT (Web Intelligence and Intelligent Agent Technology)*, pages 159–162, Lyon, France. IEEE Computer Society. (Cited on page 18.)

- [Coppola and Rosati, 2009] Coppola, P. and Rosati, L. (2009). Simulation-based evaluation of advanced public transportation information systems (aptis). In *Schedule-Based Modeling of Transportation Networks*, pages 1–21. Springer. (Cited on page 24.)
- [Cordeau, 2006] Cordeau, J.-F. (2006). A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586. (Cited on page 83.)
- [Doniec et al., 2008] Doniec, A., Mandiau, R., Piechowiak, S., and Espié, S. (2008). A behavioral multi-agent model for road traffic simulation. *Engineering Applications of Artificial Intelligence*, 21(8):1443–1454. (Cited on page 39.)
- [Dziekan and Kottenhoff, 2007] Dziekan, K. and Kottenhoff, K. (2007). Dynamic at-stop real-time information displays for public transport: effects on customers. *Transportation Research Part A: Policy and Practice*, 41(6):489–501. (Cited on page 23.)
- [EC, 2016] EC (2016). Eu transport in figures. Technical report, Publications Office of the European Union. (Cited on page 2.)
- [Estrada et al., 2015] Estrada, M., Giesen, R., Mauttone, A., Nacelle, E., and Segura, L. (2015). Experimental evaluation of real-time information services in transit systems from the perspective of users. In *Conference on Advanced Systems in Public Transport (CASPT 2015)*. (Cited on page 24.)
- [Ferber, 1999] Ferber, J. (1999). *Multi-Agent Systems : An introduction to Distributed Artificial Intelligence*. Addison-Wesley Longman Publishing Co., Inc. (Cited on page 3.)
- [Fiduccia and Mattheyses, 1982] Fiduccia, C. and Mattheyses, R. (1982). A linear-time heuristic for improving network partitions. In *19th Conference on Design Automation, 1982*, pages 175–181. (Cited on page 41.)
- [Fonzone and Schmöcker, 2014] Fonzone, A. and Schmöcker, J. (2014). Effects of transit real-time information usage strategies. *Transportation Research Board, Washington, D.C.* (Cited on page 24.)
- [Gelernter, 1985] Gelernter, D. (1985). Generative communication in linda. *ACM Transactions on Programming Languages and Systems*, 7(1):80–112. (Cited on page 7.)
- [Gendreau et al., 1999] Gendreau, M., Guertin, F., Potvin, J.-Y., and Taillard, E. D. (1999). Parallel tabu search for real-time vehicle routing and dispatching. *Transportation Science*, 33(4):381–390. (Cited on page 80.)
- [Golkar and Sousa, 2011] Golkar, B. and Sousa, E. (2011). Adaptive localized resource allocation with access point coordination in cellular networks. In *IEEE International Conference on Communications (ICC)*, pages 1–5. (Cited on page 55.)
- [Grootenboers, 2010] Grootenboers, F. (2010). The dynamic dial-a-ride problem with time windows in a competitive multi-company environment. Technical report, Delft University of Technology. (Cited on page 85.)
- [Grootenboers et al., 2010] Grootenboers, F., De Weerd, M., and Zargayouna, M. (2010). Impact of competition on quality of service in demand responsive transit. In *Lecture Notes in Computer Science*, volume 6251, pages 113–124. Springer Berlin Heidelberg. (Cited on pages 12 and 85.)

- [Gruer et al., 2001] Gruer, P., Hilaire, V., and Koukam, A. (2001). Multi-agent approach to modeling and simulation of urban transportation systems. In *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, volume 4, pages 2499–2504. IEEE. (Cited on page 10.)
- [Gubbi et al., 2013] Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660. (Cited on page 95.)
- [Gueriau et al., 2015] Gueriau, M., Billot, R., El Faouzi, N.-E., Hassas, S., and Armetta, F. (2015). Multi-Agent Dynamic Coupling for Cooperative Vehicles Modeling. In 30/01/2015, editor, *The Twenty-Ninth Conference on Artificial Intelligence AAI'2015 - (DEMO Track)*. (Cited on page 14.)
- [Hickman and Wilson, 1995] Hickman, M. D. and Wilson, N. H. (1995). Passenger travel time and path choice implications of real-time transit information. *Transportation Research Part C: Emerging Technologies*, 3(4):211–226. (Cited on page 24.)
- [Hu et al., 2012] Hu, T.-Y., Tong, C.-C., Liao, T.-Y., and Ho, W.-M. (2012). Simulation-assignment-based travel time prediction model for traffic corridors. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1277–1286. (Cited on page 14.)
- [Jakob et al., 2012] Jakob, M., Moler, Z., Komenda, A., Yin, Z., Jiang, A. X., Johnson, M. P., Pechoucek, M., and Tambe, M. (2012). Agentpolis: towards a platform for fully agent-based modeling of multi-modal transportation (demonstration). In *AAMAS 2012*, pages 1501–1502. (Cited on page 18.)
- [Jaw et al., 1986] Jaw, J.-J., Odoni, A. R., Psaraftis, H. N., and Wilson, N. H. M. (1986). A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20(3):243–257. (Cited on pages 85 and 86.)
- [Kamargianni et al., 2016] Kamargianni, M., Li, W., Matyas, M., and Schäfer, A. (2016). A critical review of new mobility services for urban transport. *Transportation Research Procedia*, 14:3294–3303. (Cited on page 95.)
- [Katzev, 2003] Katzev, R. (2003). Car sharing: A new approach to urban transportation problems. *Analyses of Social Issues and Public Policy*, 3(1):65–86. (Cited on page 56.)
- [Koenig and Likhachev, 2002] Koenig, S. and Likhachev, M. (2002). D* lite. In *AAAI/IAAI*, pages 476–483. (Cited on page 52.)
- [Ksontini et al., 2016] Ksontini, F., Zargayouna, M., Scemama, G., and Leroy, B. (2016). Building a realistic data environment for multiagent mobility simulation. *Smart Innovation, Systems and Technologies*, pages 57–67. (Cited on page 29.)
- [Langtangen and Cai, 2008] Langtangen, H. P. and Cai, X. (2008). On the efficiency of python for high-performance computing. In *Modeling, Simulation and Optimization of Complex Processes*, pages 337–357. Springer Berlin Heidelberg. (Cited on page 45.)
- [Luke et al., 2004] Luke, S., Cioffi-Revilla, C., Panait, L., and Sullivan, K. (2004). Mason: A new multi-agent simulation toolkit. In *SwarmFest Workshop*. (Cited on page 18.)
- [Luo, 2007] Luo, Y. (2007). A coepetition perspective of global competition. *Journal of World Business*, 42(2):129 – 144. (Cited on page 69.)

- [Maciejewski and Nagel, 2012] Maciejewski, M. and Nagel, K. (2012). Towards multi-agent simulation of the dynamic vehicle routing problem in matsim. In *Proceedings of the 9th International Conference on Parallel Processing and Applied Mathematics - Volume Part II*, PPAM'11, pages 551–560, Berlin, Heidelberg. Springer-Verlag. (Cited on pages 16, 18 and 37.)
- [Mahr et al., 2010] Mahr, T., Srouf, J., de Weerd, M. M., and Zuidwijk, R. (2010). Can agents measure up? a comparative study of an agent-based and on-line optimization approach for a drayage problem with uncertainty. *Transportation Research: Part C*, 18(1):99–119. (Cited on page 83.)
- [Mandiau et al., 2008] Mandiau, R., Champion, A., Auberlet, J.-M., Espié, S., and Kolski, C. (2008). Behaviour based on decision matrices for a coordination between agents in a urban traffic simulation. *Applied Intelligence*, 28(2):121–138. (Cited on page 39.)
- [Marsden, 2006] Marsden, G. (2006). The evidence base for parking policies—a review. *Transport Policy*, 13(6):447 – 457. Parking. (Cited on page 60.)
- [Mastio, 2017] Mastio, M. (2017). *Modèles de distribution pour la simulation de trafic multi-agent*. PhD dissertation, Université Paris-Est, Paris (France). In french. (Cited on page 10.)
- [Mastio et al., 2018] Mastio, M., Zargayouna, M., Scemama, G., and Rana, O. (2018). Distributed agent-based traffic simulations. *IEEE Intelligent Transportation Systems Magazine*, 10(1):145–156. (Cited on page 12.)
- [Mcardle et al., 2014] Mcardle, G., Furey, E., Lawlor, A., and Pozdnoukhov, A. (2014). Using digital footprints for a city-scale traffic simulation. *ACM Trans. Intell. Syst. Technol.*, 5(3):41:1–41:16. (Cited on page 14.)
- [Meignan et al., 2007] Meignan, D., Simonin, O., and Koukam, A. (2007). Simulation and evaluation of urban bus-networks using a multiagent approach. *Simulation Modelling Practice and Theory*, 15(6):659 – 671. (Cited on pages 16 and 37.)
- [Mes, 2008] Mes, M. (2008). *Sequential Auctions for Full Truckload Allocation*. PhD thesis, Universiteit Twente, Enschede, The Netherlands. (Cited on page 84.)
- [Metropolis and Ulam., 1949] Metropolis, N. and Ulam., S. (1949). The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341. (Cited on page 86.)
- [Nagel and Rickert, 2001] Nagel, K. and Rickert, M. (2001). Parallel implementation of the transims micro-simulation. *Parallel Computing*, 27(12):1611–1639. (Cited on page 18.)
- [Othman, 2016] Othman, A. (2016). *Simulation multi-agent de l'information des voyageurs dans les transports en commun*. PhD dissertation, Université Paris-Est, Paris (France). In french. (Cited on page 10.)
- [Pereira et al., 2015] Pereira, F. C., Rodrigues, F., Polisciuc, E., and Ben-Akiva, M. (2015). Why so many people? explaining nonhabitual transport overcrowding with internet data. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1370–1379. (Cited on pages 15 and 36.)
- [Rana and Stout, 2000] Rana, O. F. and Stout, K. (2000). What is scalability in multi-agent systems? In *Proceedings of the fourth international conference on Autonomous agents*, pages 56–63. ACM. (Cited on page 37.)

- [Rihawi et al., 2014] Rihawi, O., Secq, Y., and Mathieu, P. (2014). Effective distribution of large scale situated agent-based simulations. In *ICAART 2014 - Proceedings of the 6th International Conference on Agents and Artificial Intelligence, Volume 1, ESEO, Angers, Loire Valley, France, 6-8 March, 2014*, pages 312–319. (Cited on page 39.)
- [Schmidt et al., 2011] Schmidt, R. K., Kloiber, B., Schüttler, F., and Strang, T. (2011). Degradation of communication range in vanets caused by interference 2.0-real-world experiment. In *Communication Technologies for Vehicles*, pages 176–188. Springer. (Cited on page 73.)
- [Schweiger, 2003] Schweiger, C. L. (2003). Customer and media reactions to real-time bus arrival information systems. *Transportation Research Board*, 48. (Cited on page 23.)
- [Shi et al., 2004] Shi, Han, B., and Wang, J. (2004). Study of the mode of real-time and dynamic parking guidance and information systems based on fuzzy clustering analysis. In *Proceedings of 2004 International Conference on Machine Learning and Cybernetics, 2004*, pages 2790 – 2794. IEEE. (Cited on page 71.)
- [Shoup, 2017] Shoup, D. (2017). *The High Cost of Free Parking: Updated Edition*. Routledge. (Cited on page 60.)
- [Smith, 2012] Smith, B. W. (2012). Managing autonomous transportation demand. *Santa Clara L. Rev.*, 52(4):1401–1422. (Cited on page 95.)
- [Solomon, 1987] Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling with time window constraints. *Operations Research*, 15:254–265. (Cited on pages 74, 77 and 80.)
- [Steenbrink, 1974] Steenbrink, P. A. (1974). *Optimization of transport networks*. Wiley London ; New York. (Cited on pages 3 and 53.)
- [Taillandier et al., 2012] Taillandier, P., Vo, D.-A., Amouroux, E., and Drogoul, A. (2012). Gama: A simulation platform that integrates geographical information data, agent-based modeling and multi-scale control. In *PRIMA*, volume 7057 of *Lecture Notes in Computer Science*, pages 242–258. Springer. (Cited on page 18.)
- [Tatara and Ozik, 2009] Tatara, E. and Ozik, J. (2009). How to build an agent-based model iii – repast simphony. In *Applied Agent-based Modeling in Management Research, Academy of Management Annual Meeting, Chicago*. (Cited on page 18.)
- [Tesauro, 2005] Tesauro, G. (2005). Online resource allocation using decompositional reinforcement learning. In *Proceedings of the 20th national conference on Artificial intelligence*, volume 2, pages 886–891. AAAI Press. (Cited on page 55.)
- [Tisseo, 2011] Tisseo (2011). Tisseo, press kit, 24 février 2011. In French. (Cited on page 30.)
- [Vercouter, 2001] Vercouter, L. (2001). A distributed approach to design open multi-agent systems. In Omicini, A., Petta, P., and Tolksdorf, R., editors, *Engineering Societies in the Agents World II, Second International Workshop, ESAW 2001, Prague, Czech Republic, July 7, 2001, Revised Papers*, volume 2203 of *Lecture Notes in Computer Science*, pages 25–38. Springer. (Cited on page 26.)

- [Wahle et al., 2002] Wahle, J., Bazzan, A. L. C., Klügl, F., and Schreckenberg, M. (2002). The impact of real-time information in a two-route scenario using agent-based simulation. *Transportation Research Part C: Emerging Technologies*, 10(5):399–417. (Cited on page 39.)
- [Wong and Sycara, 2000] Wong, H. C. and Sycara, K. (2000). A taxonomy of middle-agents for the internet. In *Proceedings of the Fourth International Conference on MultiAgent Systems*, pages 465 – 466. (Cited on page 25.)
- [Wooldridge and Jennings, 1995] Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(02):115–152. (Cited on page 3.)
- [Zargayouna, 2007] Zargayouna, M. (2007). *Coordination model and language for open multiagent systems. Application to the Dial-A-Ride Problem*. PhD dissertation, University of Paris-Dauphine, Paris (France). In french. (Cited on pages 11, 94 and 96.)
- [Zargayouna et al., 2016] Zargayouna, M., Balbo, F., and Ndiaye, K. (2016). Generic model for resource allocation in transportation. application to urban parking management. *Transportation Research Part C Emerging Technologies*, 71:538–554. (Cited on page 12.)
- [Zargayouna et al., 2018] Zargayouna, M., Othman, A., Scemama, G., and Zeddini, B. (2018). Multiagent simulation of real-time passenger information on transit networks. *IEEE Intelligent Transportation Magazine*, v(n):pp–pp. To appear. (Cited on page 12.)
- [Zargayouna et al., 2012] Zargayouna, M., Scemama, G., Zeddini, B., Kompfner, P., Gatellier, P., Constant, P., and Beckman, D. (2012). Future internet for a personal travel companion service. In *19th ITS World Congress*, page 10 pages. (Cited on page 14.)
- [Zargayouna and Zeddini, 2012] Zargayouna, M. and Zeddini, B. (2012). Fleet organization models for online vehicle routing problems. In *Transactions on Computational Collective Intelligence VII*, pages 82–102. Springer Berlin Heidelberg. (Cited on page 12.)
- [Zargayouna et al., 2014] Zargayouna, M., Zeddini, B., Scemama, G., and Othman, A. (2014). Simulating the impact of future internet on multimodal mobility. In *11th IEEE/ACS International Conference on Computer Systems and Applications (AICCSA 2014)*, pages 230–237. IEEE Computer Society. (Cited on pages 12, 16 and 29.)
- [Zheng et al., 2014] Zheng, Y., Capra, L., Wolfson, O., and Yang, H. (2014). Urban computing: Concepts, methodologies, and applications. *ACM Trans. Intell. Syst. Technol.*, 5(3):38:1–38:55. (Cited on page 14.)

Multi-Agent Approaches for Dynamic Transportation Problems

Abstract: My research work deals with multi-agent systems and their application to dynamic transportation problems and applications. I am interested in the development of models, simulations and algorithms to support the multimodal transportation actors solving the complex problems induced by the continuous changes in the supply and demand of transportation services. I propose a number of models and simulations supporting the transportation actors in observing, estimating and optimizing the state of the transportation networks. My work contributes to a better understanding of the problems raised by the new mobility services (dial a ride, ride sharing, urban parking search systems, etc.) and to a better design of operating systems and traveler information systems, integrating these new services.

Keywords: Multi-agent systems, transportation science, simulation, vehicle routing problems, traveler information, traffic modeling
