Using TraSMAPI for the Assessment of Multi-Agent Traffic Management Solutions

Ivo J. P. M. Timóteo, Miguel R. Araújo, Rosaldo J. F. Rossetti, Eugénio C. Oliveira

Department of Informatics Engineering, Artificial Intelligence and Computer Science Lab, Faculty of Engineering, University of Porto, Rua Dr Roberto Frias S/N, 4200–465 Porto, Portugal

{ivo.timoteo, miguel.araujo, rossetti, eco}@fe.up.pt

Intelligent Traffic Management is undoubtedly a promising solution to tackle modern cities' problems related to the growth of the urban traffic volume as it is a non-invasive approach when compared to interventions to the road network structure. Among possible solutions aiming at Intelligent Traffic Management, we believe that Multi-Agent Systems (MAS) are the most appropriate metaphor to deal with complex domains such as road networks and traffic management and control systems. However, we feel that traffic management and control, particularly intelligent traffic control, is an issue that has not yet been addressed to its full potential. Therefore, we propose the use of TraSMAPI's (Traffic Simulation Management API) multi-agent framework for multi-agent simulations over multiple microscopic simulators, as a basis for the development of intelligent policies for traffic management. We present a case study in which the advantages of cross-validation using two simulators are highlighted.

Keywords: multi-agent systems, agents in traffic and transportation, traffic simulation, intelligent traffic control and management, calibration methodologies.

INTRODUCTION

Modern cities are experiencing an incredible growth in terms of both population and financial power of each individual. This financial independence allows more and more citizens to have private vehicles and to take the ease of mobility and communication for granted. In fact, the flow of goods and people within cities directly related to the road network is now reaching previously unthinkable levels. This makes road networks one of the most important assets of any city as the quality of that flow directly affects the city's economic dynamics, health levels related to air pollution and the well-being of citizens related to the satisfaction resulting from the freedom of movements. However, urban road networks are not entirely prepared to face the new problems that arise with the incredibly raising volume of urban traffic in large cities.

These problems are more profound than they first appeared to be, as the bottleneck for further improvements on the traffic flow and the ability to deal with larger volumes of traffic seems to be in the appropriate management and control of the traffic flow, especially at intersections, rather than on infrastructure. In fact, the typical solutions to traffic congestion problems, such as building new roads, increasing the number of lanes, setting up speed limits as well as dedicated lanes, are not only usually insignificant in urban areas but can even lead to negative feedback worsening congestion in road networks. This phenomenon has been demonstrated by the Braess's paradox [1, 2] and confirmed by several experiments and real-life situations [3, 4 and 5]. Also supporting the effective traffic management approach is the fact that structural intervention to the road network in urban areas is generally impossible or unviable.

Effective Traffic Management can be approached through various methods from scheduling the utilization of certain lanes, changing the traffic lights program according to the time of the day or day of the week, having police agents controlling traffic in special occasions, and so on. However, we feel that one of the major flaws in traffic control systems today is that they are very "static" in their approach using little to no information from their environment and having previously collected data as a foundation for their solutions. In modern cities with great traffic volumes, traffic behaves rather chaotically [6] and past data concerning the typical behavior starts to lose validity. This fact demands future traffic management systems to be adaptive according to their surrounding environment and possibly possess learning abilities to better respond to future problems. This adaptive behavior and/or learning ability can be considered intelligent and will be referred to in this paper as Intelligent Traffic Management.

Intelligent Traffic Management is the main focus of our work and our ultimate goal. When analyzing our domain of study, road networks, its actuators (i.e. traffic lights, variable speed signs, GPS, and so forth), its sensors (i.e. induction loops, video cameras, radars, and so on), as well as

its users (i.e. travelers in general terms), we can perceive it, quite intuitively, as a swarm of interacting agents. Users are definitely intelligent and selfish, looking for the best solution for themselves without any concern with the global repercussions their actions may have. The actuators, however, tend to be, as pointed out before, static and non-adaptive, resulting in trivial solutions which could be greatly improved. Improvements to these solutions, as mentioned before, should be achieved by means of Intelligent Traffic Management through the creation of intelligent agents responsible for the actuators decision. We might have sought for centralized solutions to this problem but given the dimension and complexity of such a domain, it raised many problems both in terms of processing power and error recovery. Also, simply adding more controlling agents would not be efficient if it had to be done in the main system every time.

Another important aspect is that we believe traffic management can be summarized as finding local solutions, accounting for information from nearby agents through communication and expecting some kind of swarm behavior. In other words, a traffic light agent that keeps an intersection unobstructed is working towards global improvements in the flow of the network unless it is creating congestion in one of its neighbors. If that is the case, then the neighbor will launch an alert and the traffic light agent will adapt its solution, to a certain extent, in order to help the neighbor agent thus working, again, towards the global improvement in the flow of the network. It is expected that such a behavior will spread its influence throughout the entire network, then affecting the system as whole.

These reasons lead towards a distributed solution where every agent is autonomous, locally aware of its environment, adaptive to its neighboring environment and able to communicate with other agents in order to exchange important information about the system. Having this in mind, we believe that a Multi-Agent System approach is appropriate since it can be easily distributed and that a set of autonomous agents is a good metaphor for distributed and complex domains such as control systems in traffic management. In fact, such a solution has been widely used by the community working on traffic analysis as suggested by Schleiffer [7].

When dealing with real-life systems that have to be fully functional without interruptions, real-life experiments can only be made in the late phases of development. This implies that the development of such solutions needs to be supported by studies in simulators. In this work we present the TraSMAPI MAS platform, which provides real-time interaction with microscopic traffic simulators, collects metrics and statistics and offers an integrated framework to develop multi-agent system solutions in this domain. Using TraSMAPI we can focus solely on the creation of agents for intelligent traffic management without having to implement any interaction with the simulator. TraSMAPI allows the user to devise and implement agents irrespectively of the simulator to be used, allowing the same solution to be tested in different simulators. For the case-study in this paper, we have used TraSMAPI and the microscopic simulators SUMO [8] and ITSUMO [9].

This paper starts by focusing on the use of multi-agent systems in the traffic control and management field. In Section 3, we present TraSMAPI as well as a deeper look into TraSMAPI's multi-agent system framework architecture so that we can present a typical architecture of a multi-agent solution using our approach. Then we shall discuss a simple case study in Section 4, focusing on the implementation of the multi-agent system solutions using TraSMAPI. Finally, we draw some conclusions and speculate about future work.

MAS IN TRAFFIC AND TRANSPORT

Before continuing to the development of multi-agent system solutions to traffic management, it is necessary to discuss on key concepts related to autonomous agents, their relationships and how suited they are to the field of traffic and transport. Multi-agent systems are under the umbrella of the Distributed Artificial Intelligence and have inspired increasing interest among scientists from different knowledge fields. The rapid evolution in computational resources, both in terms of hardware and in software, has contributed greatly to its development as it has become relatively easy to create distributed systems with various commodity machines under a reasonable price.

Basically, there are two major ways in which agent-based solutions have been proposed and effectively applied. Firstly, real agents are playing an important role in contemporary society. Not only robotics has profited from such a technology but also the Internet environment experiences the presence of software agents that are frequently interacting with human users. Secondly, agent-based models become a natural metaphor to represent domains where a number of intelligent and autonomous entities interact with each other and with the environment, being ideal to represent

and simulate social interactions in a vast range of domains.

These models are being increasingly used within analysis frameworks as an effective tool to aid the understanding of complex and stochastic phenomena. Traffic and transportation systems have profited from the adoption of the multi-agent metaphor and have also stimulated much research on and development of agent-based technologies as they are, in fact, complex environments filled with intelligent agents. Indeed, as we shall see in the following discussion, most applications of multi-agent systems to traffic and transport focus on traffic control and management, although other forms are also addressed by the scientific community as well as by practitioners.

Multi-agent systems' main premise is to interpret real world in terms of agents that exhibit intelligence, autonomy, and some degree of interaction with other agents and with its environment. Other typical characteristics of agents include, for example, reactivity, adaptability, pro-activity, the ability to learn, and the ability to communicate and to behave socially. An agent can be described, in terms of its architecture, as being a set of sensors through which it can gather information from the environment, and effectors through which it can act upon that environment and behave according to its objectives [10]. This structure can feature both reactive and cognitive abilities, and a mixture of both, to mimic human behavior in a wide range of applications. Steels [11] suggests that each single agent albeit possibly having a very simple structure can contribute to a more complex and efficient behavior of the system as a whole demonstrating more advanced and efficient solutions can be achieved through the use of multi-agent systems. If the behavior of such a single agent can be backtracked, then this can be used to aid the understanding of the more complex behaviors at aggregate level, such as the social phenomena for instance.

To the best of our knowledge, some former attempts to apply agent-based techniques to address transportation issues date back to the 90's. For instance, Haugeneder and Steiner [12] proposed a co-operative agent-based architecture as a means to improve traffic management and control. Not surprisingly, that was also in a moment when much research and controversies were going on so as to define the actual scope of agency [13]. For instance, many people from different fields in Computer Science and even in Artificial Intelligence (AI) were trying to realize whether agents were different from objects, either from an object-oriented perspective or from autonomous processes, operating systems and network point of views. If in the beginning people from the AI community benefited from the complex and dynamic nature intrinsic to transportation systems to devise and support agent theory, transportation engineers and practitioners have now started to recognize the natural ability of the multi-agent metaphor to model traffic phenomena. Owing to their characteristics and concepts, multi-agent systems have a natural aptitude to cope with a wide range of issues in contemporary traffic and transportation scenarios [7].

Not amazingly, most works report on applying agent-based techniques to control systems and traffic management to make those systems more autonomous and responsive to recurrent traffic demand (e.g. [14]). The analysis of Intelligent Traffic Systems through this approximation has also been investigated (e.g. [15, 16]), and some other works report on applications to freight transport and optimization of resource use (e.g. [17]). Another work has been reported in the literature, which provides a fairly good survey on the application of agent-based approaches to transport logistic [18]. Nonetheless, the challenging issue of modeling more realistically the decision-making process underlying travelers' behavior has encouraged an increasing use of agents for such a purpose. For example, drivers are endowed with cognitive abilities to plan a trip accounting for a mental model of the world and an expectation of the utility their choices would bring about (e.g. [19, 20, 21 and 22]). In this same direction, agent concepts have also proved to be very useful in fostering the improvement of the activity-based analysis of travel demand and of advanced travelers' information systems [23]. More recently, a thorough review has been presented on the applications of agent-based solutions to the specific domain of traffic and transportation [24].

In this work we propose a multi-agent architecture using TraSMAPI to underlie the implementation of a tool to analyze and to test with different traffic control strategies and management policies effectively. The multi-agent system architecture herein proposed is conceived in a way it allows agents to be generic for any kind of solution and that can interact both autonomously and cooperatively through a specific interaction protocol that fosters short-term tactic as well as long-term strategic control and management decisions.

THE DEVELOPMENT OF MAS USING TraSMAPI

Simulators try to emulate the real world. However, it is impossible to consider every detail of the environment which originates imprecision in the simulation. Imprecision might generate and accumulate invalid results. By evaluating solutions in different simulators, sturdier results are to be expected as individual faults are minimized. Solutions developed this way will also be much more resistant to simulator-bias: cases in which one's solution unconsciously exploits the shortcomings of the simulator being used.

Comparing and contrasting simulators is key to increase the relationship between existing simulators and reality. This is a widely discussed topic in the literature: in [25] and [26], different phases of the simulation process are compared among simulators, the car following behavior and phase transitions; in [27], authors discuss on the problem of combining mesoscopic and microscopic simulators in an integrated environment.

Nevertheless, the development of agent-based traffic solutions is usually drawn back by the great effort the developer has to spend in designing interactions with simulators and a proper communication protocol between the agents. Therefore, TraSMAPI is herein presented as a tool that offers complete integration between the multi-agent system framework and the simulator interaction module allowing the developer to focus solely on the creation of the solution agents. Furthermore, TraSMAPI is independent from any simulator, which allows developers to cross information obtained in different simulators without changing their solution code. This very characteristic allows us to test with different simulators profiting from the previously mentioned advantages.

This section will focus on the architecture of TraSMAPI multi-agent system framework, comparing it to other multi-agent system frameworks that could be used and finally describing the typical architecture of an agent-based solution.

The TraSMAPI Framework

The general architecture of TraSMAPI is based on modules each with a well-defined function in the whole system. (Figure 1 is useful for the reader to have a correct understanding of the current section).

The researcher is responsible for selecting the simulator to be used (given it is supported by the API) and for implementing the desired agents. We try neither to limit the applicability of the framework by not restricting what information each agent has access to, nor checking whether the processing time is bigger than the time step of the simulator by leaving such decisions to the researcher designing the solution.

TraSMAPI uses Java objects in order to hide several layers implicitly used by the agent. Ideally, this higher abstraction should make the codification of the agent's behavior completely independent from the chosen simulator allowing total code reuse. This is guaranteed by our framework as long as the simulator to be used implements all functionalities that are expected by the agent. For instance, a particular simulator might implement variable speed signs whereas another might not.

TraSMAPI's most important components are the Communication and the passive Statistics modules. The Communication module provides the abstraction layer responsible for the interaction with the various microscopic simulators. The interaction is based on queries, to gain knowledge on the environment state and agents as well, possibly changing the state of the environment and affecting the state of other agents. The passive Statistics module is responsible for recording all the information transmitted by the simulator. This information, both past and present, is thus made available to the agents via a simple query interface. This module can be very helpful in the creation of heuristics to be used in decision procedures and in learning algorithms.

The TraSMAPI MAS Architecture

TraSMAPI offers an additional Multi-Agent System module which supports agent-to-agent as well as broadcast communication (see Figure 1). It manages the flow of activity among agents to guarantee synchronism. Additionally, it connects directly to a statistics module so that the performance of any deployed multi-agent system solution can be properly assessed. Such a MAS framework allows the creation of new agents by following a common interface which implements

the basic MAS<->Agent interactions that allow the MAS framework to control the flow of activity and ask for agents' action in each time step.

Agents are able to reference objects inside the simulation environment and are able to query and change the environment autonomously. Concurrency problems, which are unlikely to exist in a robust solution, are coped with by the framework.

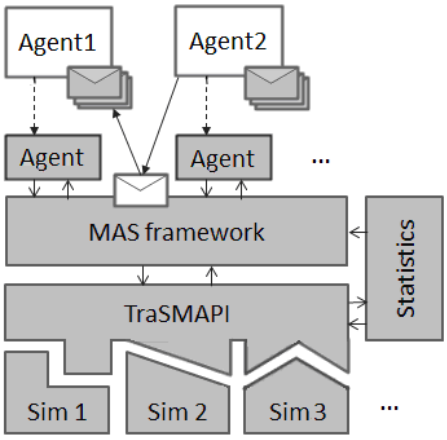

Figure 1 – A MAS solution using the TraSMAPI framework

Figure 1 presents a view of the TraSMAPI MAS solution general architecture. Dark gray blocks represent what is already available while light gray modules represent what must be implemented by the developer as a means to create customized solutions. As one can see, the only development task is to create the agents according to the objectives of the experiment to be carried out.

The communication in the Multi-Agent System Framework is based on an asynchronous message system. Agents have their own message queue in which other agents, and the MAS framework itself, might leave messages. These messages are variable and highly dependent on the implementation and goals of the application (e.g. messages can be information requests, answers to requests previously made, advice, general messages sent by the moderator to control the flow of the negotiation, and more). All the messages should be created using a generic *Message* interface so that the MAS framework can manage the communication properly.

The MAS framework is also responsible for the advancement of time steps in the simulator. It tracks the activity of each agent by defining it as either being ready to receive information from the new simulation step or as an *in process*, which means that the agent is still processing.

Agents are run in different processes; the MAS framework is responsible for managing their overall state, saving processor time when possible and "awaking" them when the simulation advances a new step or a new message has arrived in this message queue. If analyzed carefully, it can be seen that this opens the possibility of an infinite exchange of messages between agents without letting the simulation state evolve. Therefore, a maximum value of negotiation rounds was added to prevent these negotiation cycles. However, once again, an elegant solution should prevent this without such an explicit specification. The interested reader is referred to [28], where a more detailed explanation on the implementation of TraSMAPI can be consulted.

TraSMAPI MAS Framework vs. Other MAS Frameworks

Questions might arise on the motives that led us to the creation of the TraSMAPI MAS framework and why we use it instead of other widely distributed MAS frameworks. The basic motive is that TraSMAPI MAS framework was developed to be completely integrated with TraSMAPI which enables their users to simply call TraSMAPI methods and Statistic module methods when coding their agents in a fully transparent way. This might seem trivial but the gain in productivity is evident. If we were using more generic frameworks we would have to expend a fair amount of effort to the configuration of the interaction with TraSMAPI and on the definition of the communication protocol.

Also, the TraSMAPI MAS framework is intended for simulation and not real world implementation. This allows a certain freedom that would not be possible if we were creating a tool for real world use. When simulating, at first, we are looking for abstract solutions and fast prototyping in order to prove our concept and test with different scenarios. We do not want to be

concerned with communication delays, disturbances in communication, implementation of real-world communication protocols, fail-safe measures, and so on. The TraSMAPI MAS framework allows the developer to start coding his solution agents and to test with them immediately. This makes it almost the ideal brainstorming tool. Naturally, if the developer intends to implement the solution in a real world scenario, then it should be developed and tested using a more powerful generic MAS framework. Nevertheless, a general MAS framework can still use TraSMAPI as the communication layer, taking advantage of the code reuse among simulators. As suggesting an appropriate general MAS framework is out of the scope of this work, we leave to the reader deciding among a myriad of available options.

A Typical MAS Solution Using TraSMAPI MAS framework

The development of solutions using TraSMAPI can be simply interpreted as the creation of agents according to our objectives. In fact, the development of MAS solutions using TraSMAPI is fairly simple and follows a well-defined process.

The first step is to initialize the TraSMAPI framework, which is done by calling a couple of methods. Agents are created using the *Agent* interface. They are able to use TraSMAPI objects which will act as proxies for the objects in the simulation (e.g. a specific Traffic Light object acts as a proxy for the traffic light in the simulation environment). Proxy objects are abstractions to make the manipulation and extraction of information transparent to the developer and there is such a generic proxy object for every element in the simulation with which the agent can interact. They are independent from the simulator used since all the differences are managed internally.

In order to allow the MAS framework to control the flow of the simulation and the interaction between agents and between agents and the simulator, agents have to implement two methods. One is the *action* method, called at every simulation step which is simply the MAS framework telling the agent to act. The action method is used to evaluate the environment, choose the behavior and actuate upon the environment (using the proxy object). The other method is *newMessage* which is called every time a new message is sent to the agent. Agents can send messages during the execution of both methods.

Since TraSMAPI is openly distributed, it is possible to add new modules and to expand existing ones. It is common to add new statistical functions to the statistics module that fit more appropriately the solution being developed. It is also common to code new behaviors in the MAS framework (e.g. to execute scheduled tasks, to send warnings, and so on). Ideally, all these new features would be made available to the TraSMAPI community so that it becomes an ever evolving tool.

TRAFFIC LIGHTS CONTROL IN A GRID ROAD NETWORK

The creation of a simple multi-agent solution using TraSMAPI and its main features is illustrated hereafter. We are not interested in obtaining ground-breaking results in the field of intelligent traffic management, but rather show how a single solution can be used in two different simulators and empirically assess the validity of the TraSMAPI MAS framework. In this particular case study, our objective is to create a multi-agent solution in a grid network scenario in order to verify that the outputs from both simulators are comparable and that, by using both simulators, we have a sturdier evaluation of the overall quality of our solution.

Simulation Environment

This simulation will be performed using SUMO [8] and ITSUMO [9] microscopic traffic simulators, which integrate simple traffic light control for traffic management. The simulation environment is a grid formed by three horizontal roads and three vertical roads, with traffic lights in every intersection (see Figures 2 and 3). The road sections (the length of each section is 250 meters) start with one lane but when approaching the intersection they split into three lanes, each allowing turning left, going forward and turning right, respectively, where the probability of each maneuver is equal. The allowed traffic light states are green on the horizontal lanes and red on the vertical lanes (and red on the horizontal lanes and green on the vertical lanes, to complete the cycle).

Although the two road networks need to be individually specified for each of the simulators, we parameterized them equally. More specifically, we considered the same max speed for each lane and vehicle accelerations.

We would also like to stress that both simulations will be performed using exactly the same agent code.

The proposed approach

We aim at the creation of a multi-agent solution in which each agent is responsible for controlling one traffic light. The agents have an adaptive behavior considering the stopped vehicles in each lane in order to minimize the waiting time of incoming vehicles. Furthermore, they will communicate with their neighbors in order to improve the flow of the network as a whole and not only locally. The communication should be informative, as an advice, or asking for help. In the beginning of the simulation they will broadcast messages and receive direct answers to get to know which agents are their neighbors (and what is their position), making full use of the MAS framework features. We will then compare the results obtained by running the agent described below in both simulators, comparing the results obtained.

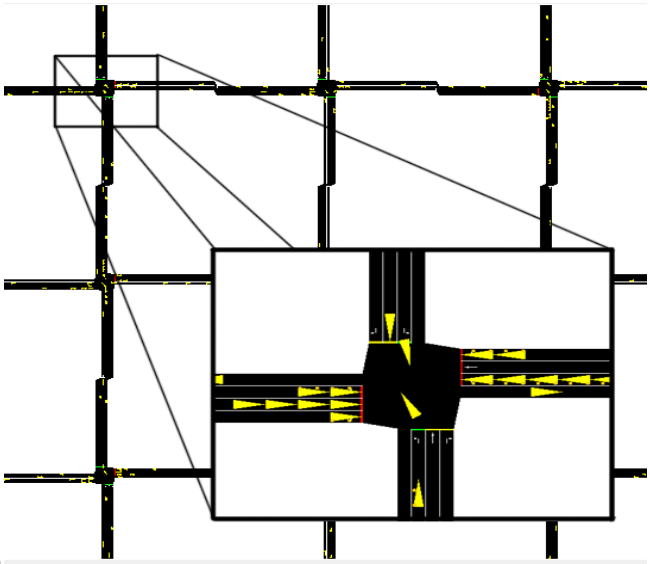

Figure 2 – Simple grid network for the simulation of a multi-agent solution using SUMO

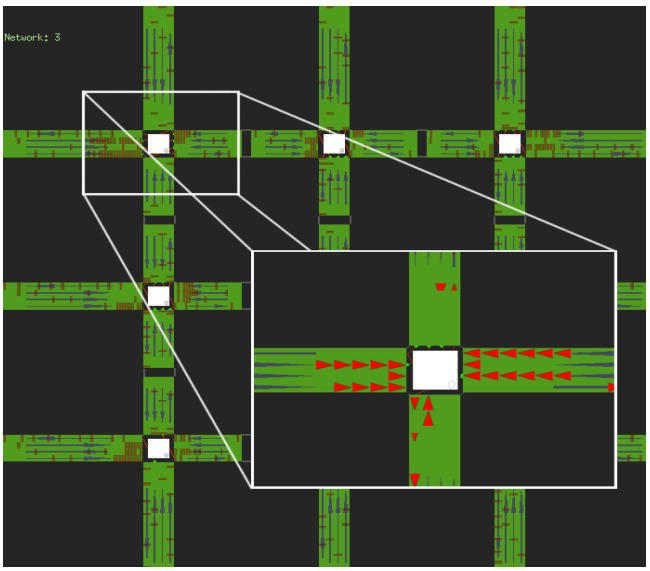

Figure 3 – Simple grid network for the simulation of a multi-agent solution using ITSUMO

The agent's decision module

The decision module decides between both traffic light states based on the result of an evaluation function. The lanes with the higher value need to be opened so that vehicles can flow.

The evaluation function used, F, is a linear combination of other evaluation functions:

$F = \sum_{i=0}^{n} k_i * E_i$, where k_i is the coefficient by which the evaluation function E_i is weighted.

The different evaluation functions, E_i, were directly derived from different metrics such as the sum of the number of vehicles waiting in each lane of each road, warning messages from neighbors, and the amount of seconds passed since the last state change. The coefficients k_i were tuned so as to yield the desired agent behavior.

Obtained results and discussion

In this study, we considered a single scenario of a uniform distribution, in which every incoming lane to the system contributes with an equal number of vehicles. Under these circumstances, we compared how the different densities impacted the overall stopped time of vehicles in each of the simulators.

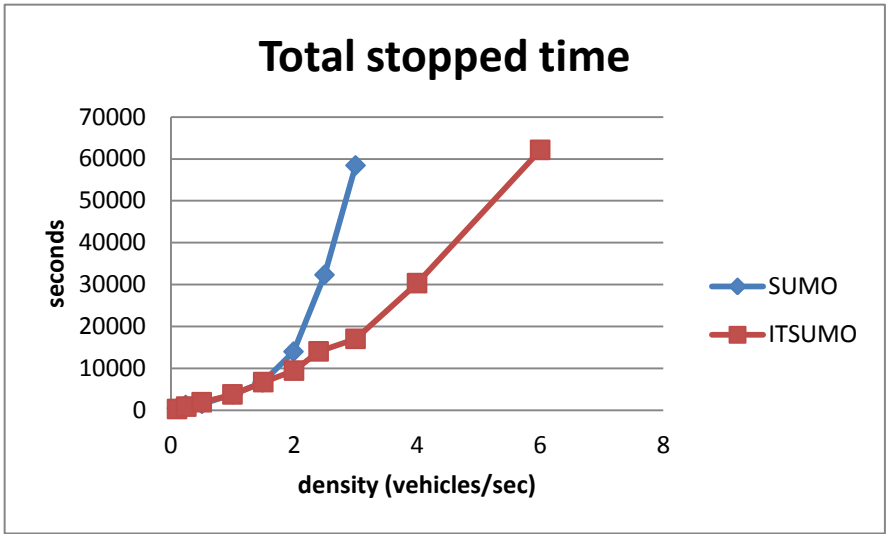

Figure 4 – Evolution of the total stopped time for each simulator.

The plot depicted in Figure 4 shows how the overall stopped time evolves as the number of vehicles in the simulation changes. As mentioned before, we are neither interested in how this specific system fairs against a standard traffic light, nor how it could be improved. We are rather interested in how the two simulators behave in the same scenario.

Observing Figure 4 again, it is possible to identify a clear relationship in the overall stopped time in both simulators. After an almost linear beginning (in which the influence of each vehicle on others is almost negligible), the interactions add up to an apparently exponential growth. This similar behavior suggests that the framework is able to replicate results among different simulators.

Taking as an example the simulation with 3 vehicles/second entering the network, ITSUMO performs three times better than SUMO. Researchers using only one of the simulators could draw significantly different results, using the same implementation. This fact backs our understanding that cross-validation between different simulators is an essential step in any intelligent traffic management solution, and, as far as we are aware, TraSMAPI is the only framework which is able to reuse the same code implementation rather than just the same algorithm.

Comparing both simulators, or better yet, asserting which one is closer to reality, is out of the scope of this paper. However, the differences between both simulators are evident. We believe this

is due to calibration problems. Although they both try to model reality, as they implement different models (SUMO implements a continuous-space approach whereas ITSUMO is based on a cellular automata representation of space), their simulations are not equivalent. Albeit this is not the main goal of TraSMAPI, comparing different simulators with the objective of calibration is another possible use for this framework. In the future, we can expect two different simulators to provide similar results when presented with the same input as they are consistently calibrated.

CONCLUSIONS

Testing traffic control and management solutions in the real world is very hard given the implications to the road network stability and the importance it has to citizens' welfare, the city economy and air pollution levels. Given this reality, the ability to compare the same MAS solution in different simulators gains importance to assert the feasibility of one's solution through cross-validation. We believe that TraSMAPI and its MAS framework, acting as an interface with various microscopic simulators, allow quicker development of solutions and the possibility of exploring new possibilities more effortlessly.

This case study demonstrates how this framework can be used to find the shortcomings of a specific multi-agent system solution to a simple traffic management problem. It also shows how this framework can be used to compare the outputs of different simulators, allowing simulator developers to more accurately detect where their simulator differs from others. With this approach, researchers can also guarantee that their solution is not taking advantage of a particular characteristic of a specific simulator.

The very next steps in our work will include a deeper investigation on how TraSMAPI can be used to find differences between simulators and underlying models. We also wonder if it would be possible to include real world data to assess the performance of the simulators accurately. Scalability, security and safety aspects will also be studied and other performance measures shall be accounted for.

Acknowledgment

We are grateful to Prof. Ana Bazzan, from UFRGS, Brazil, for allowing us to use the ITSUMO simulator, and to José Luis Macedo and Manuel Guilherme Soares, for their invaluable assistance in setting up the simulation experiments. This work has been partially supported by Fundação para a Ciência e a Tecnologia (FCT), the Portuguese agency for R&D.

References

[1] Braess, D. 1969. Über ein Paradoxon aus der Verkehrsplanung. Unternehmensforschung 12 (Mars 1968) pp. 258–268.

[2] Braess, D.; Nagurney, A.; Wakolbinger, T. 2005. On a paradox of traffic planning. Journal of Transportation Science, vol. 39, 2005, pp. 446–450.

[3] Yang, H.; Bell, M. G. H. 1998. Models and algorithms for road network design: a review and some new developments. Transport Reviews, vol. 18, no. 3, pp. 257–278.

[4] Easley, D.; Kleinberg, J. 2008. Networks. Cornell Store Press.

[5] Knödel, W. 1969. Graphentheoretische Methoden und ihre Anwendungen. Springer-Verlag. pp. 57–59.

[6] Nagel, K.; Rasmussen, S. 1995. Traffic at the edge of chaos. arXiv:adap-org/9502005v1.

[7] Schleiffer, R. 2002, Guest Editorial, Transportation Research Part C. vol.10, no. 5–6, pp. 325–329.

[8] Krajzewicz, D.; Hertkorn, G.; Rössel, C.; Wagner, P. 2002. SUMO: Simulation of Urban Mobility. In Proceedings of the 4th Middle East Symposium on Simulation and Modelling, MESM2002. SCS European Publishing House, pp.183–187.

[9] Silva, B. C.; Junges, R.; Oliveira, D.; Bazzan, A. L. C. 2006. ITSUMO: an Intelligent Transportation System for Urban Mobility. In Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS. pp.1471–1472.

[10] Russell, S. J. and P. Norvig. 1995. Artificial intelligence: a modern approach. Prentice-Hall, Englewood Cliffs.

[11] Steels, L. 1990. Cooperating between distributed agents through self-organisation. Decentralized A.I. (Y. Demazeau and J. P. Muller, eds.), pp. 175–196.

[12] Haugeneder, H.; Steiner, D. 1994. A multi-agent approach to cooperation in urban traffic. In Proceedings of the Workshop of the Special Interest Group on Cooperating Knowledge Based Systems, CKBS'93. pp. 83–99.

[13] Wooldridge, M.; Jennings, N. R. 1995. Intelligent agents: theory and practice. The Knowledge Engineering Review, vol. 10, pp. 115–152.

[14] Hernández, J. Z.; Ossowski, S.; Serrano, A. G. 2002. Multiagent architectures for intelligent traffic management systems. Transportation Research Part C, vol. 10, no. 5–6, pp. 473–503.

[15] Rickert, M.; Nagel. K. 1997. Experiences with a simplified microsimulation for the Dallas/Fort Worth area. International Journal of Modern Physics C, vol. 8, pp. 483–504.

[16] Wahle, J.; Bazzan, A. L. C.; Klügl, F.; Schreckenberg, M. 2002. The impact of real-time information in a two-route scenario using agent-based simulation. Transportation Research Part C, vol. 10, no. 5–6, pp. 399–417.

[17] Adler J. L. and V. J. Blue. 2002. A cooperative multi-agent management and route-guidance system. Transportation Research Part C, vol. 10, no. 5–6, pp. 433–454.

[18] Davidsson, P.; Henesey, L.; Ramstedt, L.; Törnquist, J.; Wernstedt, F. 2004. Agent-based approaches to transport logistics. In Proceedings of the 3rd Workshop on Agents in Traffic and Transportation, AAMAS, New York. pp. 14–24.

[19] Dia, H. 2002. An agent-based approach to modelling driver route choice behaviour under the influence of real-time information. Transportation Research Part C, vol. 10, no. 5–6, pp. 331–349.

[20] Nagel, K.; Marchal, F. 2002. Computational methods for multi-agent simulations of travel behaviour. 10th IATBR Conference.

[21] Rossetti R. J. F.; Liu, R.; Cybis, H. B. B.; Bampi, S. 2002. A multi-agent demand model. In Proceedings of the 13th Mini-Euro Conference and the 9th Meeting of the Euro Working Group Transportation. pp. 193–198.

[22] Rossetti R. J. F.; Bordini, R. H.; Bazzan, A. L. C.; Bampi, S.; Liu, R.; Van Vliet, D. 2002. Using BDI agents to improve driver modelling in a commuter scenario. Transportation Research Part C, vol. 10, no. 5–6, pp. 373–398.

[23] Dia, H.; Purchase, H. 1999. Modelling the impacts of advanced traveler information systems using intelligent agents. Road and Transport Research, vol. 8, p. 68–73.

[24] Chen, Bo; Cheng, H.H. 2010. A Review of the Applications of Agent Technology in Traffic and Transportation Systems. IEEE Transactions on Intelligent Transportation Systems, vol. 11, no. 2, p. 485–497.

[25] Panwai, S.; Dia, H. 2005. Comparative Evaluation of Microscopic Car-Following Behavior. IEEE Transactions on Intelligent Transportation Systems, vol. 6, no. 3, p. 314–325.

[26] Lu, L.; Yun, T.; Li, L.; Su, Y.; Yao, D. 2010. A Comparison of Phase Transitions Produced by PARAMICS, TransModeler, and Vissim. IEEE Intelligent Transportation Systems Magazine, vol. 2, no. 3, pp. 19–24.

[27] Vilaró, J. C.; Torday, A.; Gerodimos, A. 2010. Combining Mesoscopic and Microscopic Simulation in an Integrated Environment as a Hybrid Solution. IEEE Intelligent Transportation Systems Magazine, vol. 2, no. 3, pp. 25–33.

[28] Timoteo, I. J. P. M.; Araujo, M. R.; Rossetti, R. J. F.; Oliveira, E. C. 2010. TraSMAPI: An API Oriented Towards Multi-Agent Systems Real-Time Interaction with Multiple Traffic Simulators. In Proceedings of the 13th International IEEE Annual Conference on Intelligent Transportation Systems. pp. 1183–1188.