Towards Simulating Heterogeneous Drivers with Cognitive Agents

Arman Noroozian, Koen V. Hindriks and Catholijn M. Jonker
{a.noroozian,k.v.hindriks,c.m.jonker}@tudelft.nl

Interactive Intelligence, Delft University of Technology, The Netherlands

Keywords: Simulation, Heterogeneity, Mixed Traffic, Cognitive Driver Agent, Scalability.

Abstract: Every driver behaves differently in traffic. However, when it comes to micro-simulation of drivers with a high level of detail no framework manages to model the complexities of various driving styles as well as scale up to larger simulations. We propose a framework of micro-simulation combined with cognitive agents to facilitate such simulation tasks. Our goal is to (i) model individual drivers, and (ii) use this framework for the purpose of simulating realistic highway traffic with heterogeneous driving styles. The challenge is therefore to create a framework that facilitates such complex modeling and supports large scale simulations. We evaluate the framework from two perspectives. First, the ability to represent, model and simulate dissimilar drivers in addition to study and compare emerging behavior. Second, the scalability of the framework. We report on our experiences with the framework, outline several challenges and identify future areas for development.

1 Introduction

Drivers behave differently as recent studies confirm (Ossen and Hoogendoorn, 2011; Hoogendoorn et al., 2010). While traffic micro-simulation allows us to study driving behavior in recreated or hypothetical scenarios in a cost effective way, currently no traffic simulation framework manages to model such variations in behavior as well as scale. The ability to model such varying behavior is of importance because the particular models that are used impact the explanation/prediction strength of the simulation tools as demonstrated by Ossen and Hoogendoorn.

In micro-simulation driving behavior is described in terms of leading vehicle behavior with longitudinal (*car-following*) models. Some established car-following models include *IDM* (Treiber et al., 2000) and *Gipps* (Barceló, 2010, Chap. 8) which respectively describe driver acceleration and speed choices as a function of attributes relating to the leading vehicle (i.e. distance, relative speed) and a number of driver related attributes (i.e. desired speed and gap). *Lane-change* models extend car-following by describing behavior in the lateral direction (i.e. changing lanes) based on the impact on vehicles affected by the movements. Examples include *LMRS* (Schakel et al., 2012) and *MOBIL* (Kesting et al., 2007). Such *micro-models* are characterized by parameters that require calibration against ground truth data since drivers behave differently in different contexts (*i.e.* per country, time of day).

Calibration is a complex and time consuming process which requires minimization of the measured error between simulated results and ground truth data through search algorithms like gradient descent, simulated annealing, genetic algorithms or probabilistic methods like Kalman filters (see Treiber and Kesting (2013, Chap. 16) for an overview of the various methods). The assumption here is that calibrated parameters apply to *all drivers*. That is, the heterogeneity of drivers is not considered in this process. Heterogeneity is introduced by adding randomness a posteriori. This leads to limitations in explanation/prediction power and the generalization of models to new traffic contexts (i.e. different highway) or may lead to observations of unrealistic behavior. For instance in one empirical study Ossen and Hoogendoorn (2011) demonstrate that with calibration a model like IDM is able to model the behavior of only 19% of drivers due to heterogeneity.

Our goal is to model and study heterogeneous driving styles and measure the effects of heterogeneity in a structured and controllable manner. In particular we are interested in studying traffic scenarios with a mix of autonomous vehicles and human drivers. For this purpose we need a framework that goes beyond the current models that allow for large numbers of homogenous drivers and at the same time extends on classic cognitive models (e.g. ACT-R) that model *single* drivers with high detail. We require cognitive modeling abilities because we want to study the impact of cognitive phenomena (i.e. cognitive workload). Current simulation frameworks fall short here because of their tight integration with specific car-following and lane change models. This limitation not only applies over the driver population but also over time as driving style can also vary with time. Modeling such heterogeneity is not supported in current approaches and cognitive frameworks like ACT-R that have been used so far for micro-level modeling of individual drivers do not scale enough for our purposes.

To address this issue we have built a simulation framework that combines *cognitive agent* modeling with micro-simulation of traffic. Agents developed by *Agent Programming Languages* (APLs) are assigned to control simulation entities thereby removing tight integration between model and simulator. The idea is to provide an API similar to ones used for programming AI in computer games (see Bartish and Thevathayan (2002); Hindriks et al. (2011)). The ability to communicate and reason based on various percepts such as brake lights, turn indicators, horns and the ability to predict ahead of time are added benefits of using such a framework and desirable for modeling complex driver behavior (Treiber and Kesting, 2013, Chap. 12).

We study and illustrate how heterogeneity in *longitudinal* driving style affects macroscopic and microscopic characteristics of traffic using this framework. Note that we limit our models to longitudinal because we are mainly interested in evaluating our framework rather than the particular driver models that we use in doing so. This limitation will not affect our evaluation and we plan to extend this study with lane-change models in the future. We evaluate our framework from two perspectives. First, the ability to model and compare dissimilar driving styles. Our hypothesis here is that with sufficient heterogeneity in longitudinal driving behavior, small effects on macroscopic characteristics of traffic (i.e. flow and speed) are observable; at the same time significant effects on microscopic characteristics (i.e. gap distribution) should be observed. Second, we evaluate scalability in terms of required resources for an increasing number of agents. The expectation here is to see a linear loss of performance with an increasing number of agents. We require the ability to study small scale traffic scenarios such as merging at on-ramps, therefore we need hundreds of simulated vehicles but not thousands.

The remainder of this paper is structured as follows: In Section 2 we present an overview of the related work. In Section 3 we present our proposed cognitive framework and its evaluation in Section 4. In Section 5 we outline some of the challenges and discuss future work. We conclude in Section 6.

2 Related Work

High fidelity traffic simulation tools are based on incorporating some form of *agent-based* simulation with micro-simulation. Some instances are based on ad-hoc *reactive* agents that are computationally light weight but lack communication (Smith et al., 1995; Ehlert and Rothkrantz, 2001). Because of their simplicity these scale to larger simulations, however, their driver models do not consider heterogeneity of driving style. This is likely due to a trade-off between scalability and realism. Some instances use structured cognitive agents capable of reasoning however, the focus has largely been on the strategic aspects of commuting, i.e. planning activities, when/how to travel, which routes to take and which transportation modes to use (Rossetti et al., 2002; Rindsfüser, 2005) rather than the detailed control of a vehicle. Other instances use cognitive agents to make maneuvering decisions like the decision to overtake other vehicles or merge (Hidas, 2002; Sukthankar et al., 1998). In (Hidas, 2002) the focus is on developing a driver model for lane changing behavior, however heterogeneity is not considered. With respect to (Sukthankar et al., 1998) the focus is on learning to drive and scalability is not a main concern. For a survey on applications of agent-based simulation in the traffic domain we refer the reader to (Cheng, 2010).

Multiple traffic micro-simulation tools have been developed that provide various levels of programming control over entities in the simulation. Examples include DRACULA (Barceló, 2010, Chap. 8) , TRANSIMS (Nagel and Rickert, 2001), SUMO (Krajzewicz and Hertkorn, 2002) and MOTUS (Schakel et al., 2012) [1]. Such tools use hard-coded car-following models and provide application programming interfaces (APIs) that are geared towards studying different aspects of traffic (i.e. supply and demand or infrastructure control). The micro-models are an integral part of these tools and modifying them involves serious restructuring of the code. A common factor in these tools is that each transportation mode (i.e. passenger car, truck) is assumed to behave according to a single calibrated model of behavior. That is, a single car-following model is used per mode of transport and its calibrated parameters represent the entire population of drivers. This tight integration makes it difficult to mix and compare the various driver models that have been proposed and studied using a single framework.

[1] For a more comprehensive list of simulators refer to (Barceló, 2010).

Our approach uses *BDI* (Belief-Desire-Intention) agents to model individual drivers and their driving styles. BDI agents are based on *intentional stance theory* which simplifies the causes for actions to desires (Pokahr et al., 2005). They provide a useful framework for modeling driving at all levels of decision making required for the task and can potentially incorporate additional ranges of factors influencing driver behavior (Bazzan et al., 1999). By decoupling the driver models from the simulator we are able to provide a flexible modeling approach that supports more complex encoding of behavior and facilitates the use and comparison of dissimilar driving styles.

Agent-based simulations inherently involve a trade-off between the encoded detail in the agents and number of agents that can be simulated (Navarro et al., 2011). One of the main challenges here is that the scalability of the approach diminishes as the level of detail encoded in the models increases from the strategic to control level. To the best of our knowledge, limited attention has been payed to the scalability of BDI agents. For example Wolfe et al. (2008) investigate the use of BDI agents in air-traffic flow management and conclude that an overuse of the BDI paradigm results in scalability limitations and draw guidelines for the use of the BDI agents.

Other related work investigates the scalability of the more general concept of *Multi-Agent Systems* (MAS). Lee et al. (1998) provide definitions for the performance and scalability in MAS. Important performance indicators include throughput, response time, number of concurrent agents and communication overhead. Scalability is defined as the degree to which performance degrades as a result of expanding the number of agents. In addition the topology of the MAS and the rate of agent arrival/departure (churn) is also identified as an important scalability factor (Lee et al., 1998; Turner and Jennings, 2001). Navarro et al. (2011) propose a hybrid approach for dynamic adjustment of the level of detail in the simulation and conclude that the impact on dissimilarity between a full scale simulation and a simulation that smartly reduces the level of detail depending on the context is minimal.

Alternative cognitive modeling approaches include frameworks such as ACT-R (Salvucci, 2006) and Soar (Langley et al., 2009). In (Salvucci, 2006) ACT-R has been used to simulate the maneuvering of a single driver. Soar has been used for simulating a fighter pilot in (Langley et al., 2009). One of the main differences between BDI and cognitive frameworks is that the latter take biological human constraints into account to create more plausible human models (i.e. short/long term memory, learning abilities). The integration of these constraints is motivated by modeling the *human* reasoning process, its sources of knowledge and the way they are used to reason. Our approach is focused on general reasoning that does not necessarily reproduce the exact human reasoning process. In terms of scalability cognitive frameworks are not well suited for large scale simulation. In fact the scalability of such cognitive frameworks is still an open problem (Langley et al., 2009). Scalability is also a concern with BDI agents however to a lesser extent. BDI agents do not have to deal with complex characteristics such as short/long term memory which improves their execution time and hence the scalability of the framework as a whole.

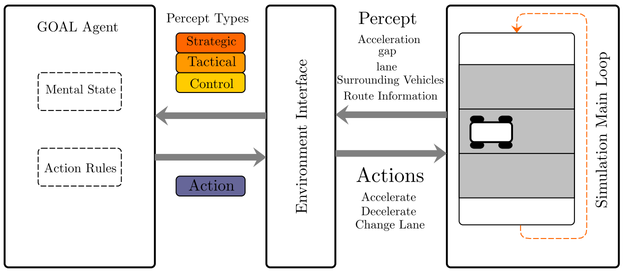

Figure 1: Framework architecture.

3 Traffic Simulation Framework

Our review of the related work motivates the need for a traffic simulation framework based on a flexible and scalable formalism to model driver heterogeneity. In this section we present our proposed framework and its components: *jSim*, the GOAL *agent programming framework* and the *Environment Interface* connecting the former two. Figure 1 demonstrates the high level architecture of our framework.

Our microscopic traffic simulator jSim is a time-discrete, space-continuous, open-source Java-based simulator. It is an extended version of *MOTUS* (Schakel et al., 2012) which has been modified to provide necessary percepts and actions to agents for encoding driver behavior.

jSim operates with a main loop where each iteration represents the passage of time by one *time step* (virtual time). During each time step agents that control entities within the simulation get the opportunity to perceive changes since the previous time step and perform actions within the current time step. The simulation loop consists of 4 phases: (***i***) updating the state of the world, (***ii***) waiting for agents to act, (***iii***) executing the agent actions and (***iv***) gathering statistics. Note that the rendezvous point in phase ***ii*** is a design decision in order to ensure fairness; that all agents get the chance to perform an action in each cycle. This results in a pseudo-parallel update step in which the updated world state is independent of the order in which agent actions are executed.

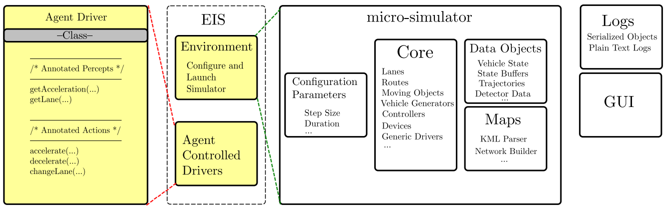

Figure 2: jSim overview and components.

Some simulation entities are controlled by agents while others may be passive. For example *detectors* simply count the number of vehicles passing over a specific point of the road network. Controllable entities include *drivers* controlling the vehicles, *on board units* and *road side units*. The agents controlling them go through a observe-decide-act cycle. Observation involves perceiving relevant information from the environment while decision making involves executing the agent's code which in turn will execute some actions within the environment. Note that due to space limitation we only give an overview of the driver specific functionality relevant to longitudinal behavior. A driver agent can perceive time, speed, acceleration and current lane. It can also perceive surrounding vehicles within a boxed area, the gap and speed difference with a leading vehicle (if any). Relevant actions available to a driver are adjusting the vehicle acceleration/deceleration. Vehicle states are updated according to kinematic equations of motion once. Figure 2 gives and overview of the components within jSim. We have integrated components for importing road networks.

The second component of our framework is the GOAL agent programming platform. GOAL agents encode driver behavior using the GOAL programming language (Hindriks et al., 2001). GOAL is a platform for implementing *Belief-Desire-Intention* (BDI) agents. They have the ability to perceive their environment, reason about its state, adopt or drop goals and decide on what actions to perform.

GOAL is a *rule-based logic programming* language that uses Prolog for representing the beliefs and goals of an agent. Rules consist of a condition on the agent's mental state and are used for deciding what to do next. Using these rules, an agent derives its choice of action from its beliefs and goals. There is a one-to-one mapping between GOAL agents and driver entities in the simulator.

Our motivation for using BDI agents and GOAL in particular is threefold. First, encoding existing driver models as BDI agents is natural as such models already use cognitive notions such as desired speed/gap which naturally fit the BDI paradigm. Second, BDI agents support cognitive modeling and allow complex driver models which use reasoning as part of their deliberation. Third, GOAL agents do not have to deal with the complexities of short/long term memory that other cognitive frameworks like ACT-R do and therefore could potentially produce more scalable agents. As an independent motivation we also find the traffic domain challenging for improving GOAL and BDI agents in general.

Note that GOAL supports programming reactive as well as proactive, cognitive agents. Existing driver models that consist of a single formula can be implemented as agents that simply react to changes in the environment. Others can be implemented as cognitive agents with rule based behavior that integrates declarative goals into the deliberation process. This type of flexibility has been demonstrated in real time (Hindriks et al., 2011) and non-real time environments (Dekker et al., 2012) and facilitates an array of complex driving styles to be simulated.

The final component of our framework is the *Environment Interface*. This interface is based on the *Environment Interface Standard* (EIS) (Behrens et al., 2010). It acts as a middle-ware between the environment and the agent. This middle-ware exposes required functionality for implementing agents and is similar to the API exposed in computer games for implementing character specific AI. In addition this interface also allows other APLs to be used in combination with jSim. Other platforms that provide support for EIS include JIAC and AGENT FACTORY.

EIS is Java based and provides functionality for encapsulating and launching virtual environments in addition to perceiving changes and executing actions therein. It also facilitates the coupling/decoupling of entities and their controlling agents. This is achieved through methods that are used to dynamically couple and decouple agents with entities. This dynamism not only facilitates the use of mixed agent types for controlling drivers but also to change and adapt a single driver's behavior during a simulation.

4 Framework Evaluation

In this section we are interested in evaluating our framework from two perspectives. First, the ability to implement, mix and compare different driver models as a method for modeling heterogeneity. Second, the scalability of the framework in terms of the number of agents in the simulation.

4.1 Mixing Driver Models

Our framework provides a controllable level of heterogeneity with the various types of driving styles that can be used in combination with the frequency by which they occur in the population of our simulated drivers. We showcase several experiments and highlight how our framework facilitates the implementation, use and comparison of mixed driver models and

```
1  main module{
2    knowledge {
3      getting_too_close :- time_gap( TG ), min_timegap( MTG ), TG =< MTG.
4      min_timegap( 2.0 ).
5    }
6    program [order=linear] {
7      if a-goal( speed( GS ) ), bel( speed( CS ), GS > CS, max_acceleration( MA ), A is MA
           * (1 - (CS/GS)**5)) then accelerate( A ).
8      if a-goal( speed( GS ) ), bel( speed( CS ), GS < CS, max_deceleration(MD), D is MD
           * (1 - (CS/GS)**5)) then decelerate( D ).
9      if bel( true ) then skip.
10 } }
11 event module {
12   program {
13     % Update Goals based on new beliefs (Drop and Adopt).
14     %--------------------------------------
15     % Always reconsider current speed.
16     if goal( speed( Y ) ) then drop( speed( Y ) ).
17     % Reasons for dropping speed synchronization goal:
18     % 1st: Large gap in front ; I am aggressive, close the gap.
19     if bel( time_gap( TG ), min_timegap( MTG ), TG >= 2 * MTG) then drop( same_speed ).
20     % 2nd: no car in front anymore.
21     if bel( not( blocked(me) ) ) then drop( same_speed ).
22     % Reconsider to see if we need to adopt new goals
23     % If getting too close we maintain same speed as car in front.
24     if bel( getting_too_close ) then adopt( same_speed ).
25     if true then setSpeedGoal.
26 } }
27 module setSpeedGoal {
28   program [order=linear] {
29     % either there is a gap to fill or not. set speed goal to lane speed limit.
30     if not( a-goal( same_speed ) ), bel( lane_speed_limit( SL ) ) then adopt( speed( SL )
           ).
31     % we're getting too close, adjust and compute lower speed.
32     if a-goal( same_speed ), bel( speed( S ), speed_delta( DS ), X is S - DS) then adopt(
           speed( X ) ).
33 } }
```

Figure 3: Cognitive driver agent code.

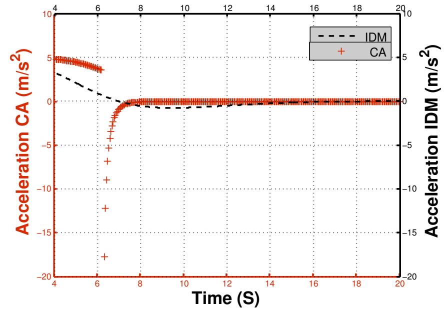

Figure 4: Agent behavior for gap filling and speed synchronization for scenario in Figure 5.

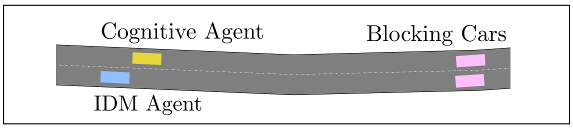

Figure 5: Blocked drivers scenario. Initial time gap of 4 seconds between agents and blocking cars

their emerging behavior.

Our experiments involve two longitudinal driver models: (***i***) IDM (Treiber et al., 2000) and (***ii***) a simple *cognitive agent* (CA) implemented by ourselves. Note that the framework facilitates the use of any number/type of model to create heterogeneity and is not limited to the two types of agents which we demonstrate here.

We use the calibrated values from (Schakel et al., 2012) for our IDM agents. The agent implementation contains a single action rule that uses the model formula to generate acceleration/deceleration actions. We can simply replace this single action rule with a different formula to implement any other longitudinal driver model without having to rebuild our tool.

Our cognitive agent (CA) is more involved than the IDM agent. Parts of its implementation are presented in Figure 3. In contrast to the IDM agent, the CA agent makes use of cognitive constructs (i.e. goals, beliefs) to encode driving behavior. Note the use of such constructs in the agent's action rules (Line 7-8). The *modules* specified on lines 11 and 27 constitute the agent's deliberation process. The CA agent implements an aggressive style of driving. That is to say that it will constantly try to keep a time gap of 2 seconds with the leading vehicle (if any) and that it uses a considerable amount of acceleration/deceleration in doing so. This constitutes the only parameter of our CA agent (2 second headway). Figure 3 demonstrates the use of the BDI paradigm in encoding a particular driving style. BDI provides a useful and natural formalism for modeling drivers (Bazzan et al., 1999). While we do not make use of all available perceptual input to implement our CA agent, more complex agents can be implemented using the perceptual information (see Section 3 for more information on the available perceptual data). Differences between the CA and IDM agent's longitudinal driving style are demonstrated in Figure 4. The plotted acceleration values have been gathered during the simulation scenario depicted in Figure 5. The CA agent aggressively fills the gaps while it avoids getting dangerously close to its leading vehicle. A discontinuity in the acceleration of the CA agent occurs at the point where the 2 second time gap is violated and the agent decelerates to stop further violation. The IDM agent chooses its accelerations smoothly based on the IDM formula (Schakel et al., 2012).

Given these agents we study the effects of heterogeneity on macroscopic and microscopic characteristics of traffic. The following three scenarios are simulated in our experiments: (***i***) All drivers use the IDM model, (***ii***) all use the CA model (***iii***) divers use the IDM or CA model with 50% chance.

We simulate a hypothetical 500 meter highway section. The density of vehicles on each lane varies. Density is controlled by the gap at which vehicles are inserted on the lane. Vehicles enter with a speed that gives them enough time to come to a full stop if the leading vehicle breaks at maximum deceleration. Note that vehicles are restricted by and do not change lane since we are comparing longitudinal driver behavior.

Our first experiment demonstrates driver behavior in traffic flow situations before and leading to a congested traffic state. For this experiment we compare the emerging macro level flow, density and speed

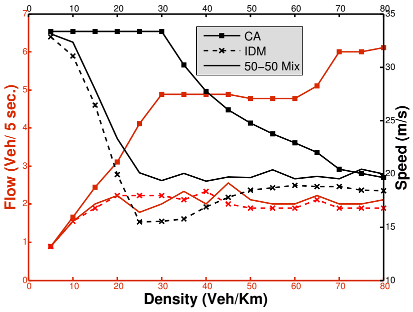

Figure 6: Comparison of Fundamental Diagrams for Cognitive Agent (CA), IDM and 50-50% mix.

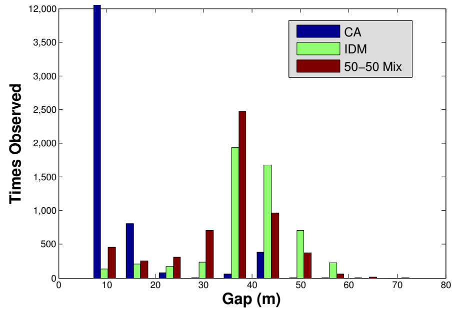

Figure 7: Histogram of observed vehicle gaps for cognitive agent (CA), IDM and 50-50% mix of CA/IDM.

properties of the traffic for the 3 scenarios outlined above. The relations between these properties are macroscopic features and are referred to as the fundamental diagrams (i.e. flow and speed in relation to density) (Treiber and Kesting, 2013, Chap. 4). Figure 6 illustrates the emerging fundamental diagrams. A detector positioned at 10% of the segment length was used to measure the flow and speed of vehicles. These measurements show the effects of heterogeneity on macroscopic features of the traffic.

Due to their aggressive driving style CA agents achieve the highest flow rates (Fig. 6). Aggressiveness does not have the same effect on flow when heterogeneous traffic is considered. It is the result of CA agents being restricted by IDM agents. Effectively, CA agents have a limited maneuvering space between the lead and following vehicle. Note however, that the effect of heterogeneous drivers (mix) is more visible in the speed measurements. This is due to the CA agents using more acceleration (and higher speeds) to fill in gaps thereby lifting the average. Note that this effect is more pronounced in mid range densities where the differences in driving style are more likely to play a significant role.

Our second experiment demonstrates driver behavior in congested traffic. Here we repeat the previous experiment with the addition of one slow moving vehicle per lane to block the movement of all following vehicles and produce artificially congested traffic. For this experiment we demonstrate the effect of heterogeneity on the distribution of gaps between the vehicles in the bottom denser lanes. This is a microscopic characteristic of the traffic which is plotted in Figure 7 as histograms. The figure demonstrates that CA agents maintain shorter gaps from their leading vehicles. For the IDM agents the gaps are generally larger. With heterogeneous driving styles the gaps become relatively smaller in comparison to IDM and the distribution is skewed towards smaller gaps because of the CA agent driving style.

These two experiments confirm our hypothesis with respect to the small effect of heterogeneity on macroscopic and a larger effect on microscopic characteristics of traffic (see Section 1) and demonstrate how our framework facilitates the implementation, use and mixing of multiple driver models as a method for modeling heterogeneity.

4.2 Scalability

In addition to the use, mixing and comparison of the diver models we have also evaluated the scalability of our framework. Our scalability requirements are derived from our goal to study driver behavior in small sections of highways. Such small sections may contain drivers in the order of 100s. Note that all figures and results relating to the scalability of the framework have been obtained on an Intel i7 processor chip with 4GBs of available RAM.

Figure 8 plots the average execution time of an iteration of the simulation loop. We measure two quantities. The first is the agent's *reaction window* which is the amount of time the simulation waits for all agents to perform an action. The second is the duration of a single iteration of the main loop. Note that the execution of our agent codes is the main bottleneck in the framework. This is not surprising since we have executed agent codes sequentially and not in parallel. We observe a linearly increasing trend in the execution time and therefore a linearly decreasing trend in the framework scalability as expected.

The step size in the measurements depicted in this figure is equal to 25 milliseconds. Given this time step a simulation of 10 seconds with 100 agents, requires approximately 156 seconds to execute. Doubling the step size to 50 *ms* results in a simulation of the same duration to require 78 seconds. Using the linear approximation trend line depicted in the figure we can draw the following relation: $d_r = \frac{d_s}{\Delta t} \times (3.3a + 64.35) \times 10^{-3}$, where d_r is the real time duration of the simulation in seconds, d_s is the duration of simulation that we want to run in seconds, Δt is the simulation step size and a is the number of agents. This property is known as the *real time ratio* of the simulation. This quantity is of practical importance because it indicates the time required to run a simulation which also impacts the calibration process.

In order to put our scalability results in perspective we also report on performance results from the literature. Bartish and Thevathayan (2002) report on

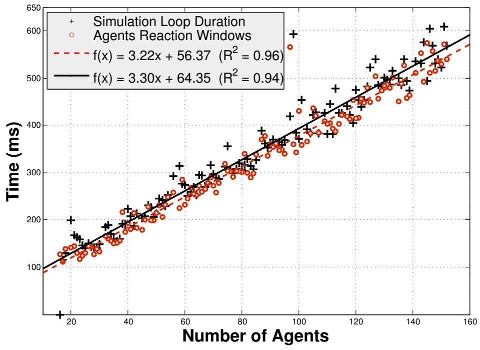

Figure 8: Scalability of Framework

using BDI agents for controlling game characters and arrive at comparable linear scalability results. Wolfe et al. (2008) report on a framework for air-traffic control using BDI agents. The authors identify the execution properties of the underlying BDI agent implementation as one of the important sources for the inefficiency of their framework. Salvucci (2006) report on simulating a single driver using the ACT-R cognitive framework. A comprehensive analysis of the real-time ratio of the parallel TRANSIMS traffic simulator is also reported in (Nagel and Rickert, 2001). Real-time ratios between 0.25 and 0.007 (faster than real-time) have been reported for simulating urban traffic of the city of Portland.

Whether agent based simulation frameworks are considered scalable depends on the requirements. In light of our scalability requirement we can conclude that the framework performance is acceptable. However, parallelism seems necessary when considering that calibration of driver models requires multiple rounds of simulation per parameter. We believe that the scalability of MAS and BDI agents in particular are important open questions. It is important to report trends rather than qualitative results to enable detailed analysis and comparison of frameworks.

5 Challenges and Future work

An important challenge that we have briefly touched upon is the calibration of heterogeneous models. Because of the larger number of parameters that need to be calibrated with heterogeneous models, the process is more time consuming than the methods described in Section 1. More efficient calibration methods are required. One promising area for further research is modeling driver behavior through learning from available data. Similar ideas have been explored in (Sukthankar et al., 1998) in which learning has been used for tactical level driving. The advantage here is that learning can be done offline to create archetypal driving styles (i.e. aggressive vs safe driver). However, it is not yet clear how realistic such models are in comparison to control theoretic models and how they can be used for studying real traffic.

Furthermore, scalability of cognitive agents still remains a challenge (Langley et al., 2009). An aspect of our future work is to use GOAL's ability to run distributed simulations to achieve better scaling. Another important aspect with respect to BDI agents is to improve the efficiency of the reasoning process of agents. Solutions like dynamic adjustment of the simulation fidelity proposed in (Navarro et al., 2011) also present promising research.

We are planning to extend this study with driver behavior in both longitudinal and lateral directions. We also plan to use the agent communication abilities to research distributed coordination and car platooning which impact road throughput and safety.

6 Conclusions

We have outlined our initial steps towards designing a cognitive framework for modeling heterogeneous driving styles using BDI agents and micro-simulation of traffic. The goal of the framework is to facilitate a *heterogeneous* modeling paradigm and to implement and compare different driver models in highway traffic. We have evaluated our framework from two perspectives: (***i***) Its ability to meet this goal, (***ii***) the scalability of the framework. Our evaluation demonstrates that the our framework is quite flexible in modeling complex heterogeneous driving behavior. The main advantage of our framework is its ability to mix multiple driver models rather than using a single model for an entire population of drivers as is done in current traffic simulation frameworks. A second advantage of our proposed framework is that in comparison with solutions like micro-simulation with reactive agents, the framework's use of BDI agents makes it better suited for modeling the cognitive complexities of driving behavior. While less scalable than reactive agents, our BDI approach easily scales to 100s of agents which makes them more scalable than other cognitive frameworks such as SOAR and ACT-R. In terms of scalability the framework shows promising results towards our requirements. The scalability of agents and their improvement remains an important avenue of research.

REFERENCES

Barceló, J., editor (2010). *Fundamentals of Traffic Simulation*, volume 145 of *Operations Research & Management Science*. Springer.

Bartish, A. and Thevathayan, C. (2002). BDI agents for game development. In *Proc. AAMAS '02, part 2*, page 668. ACM.

Bazzan, A. L. C., Wahle, J., and Klügl, F. (1999). Agents in traffic modelling - from reactive to social behaviour. *KI-99: Advances in Artificial Intelligence*, pages 303–306.

Behrens, T. M., Hindriks, K. V., and Dix, J. (2010). Towards an environment interface standard for agent platforms. *Annals of Mathematics and Artificial Intelligence*, 61(4):261–295.

Cheng, H. H. (2010). A Review of the Applications of Agent Technology in Traffic and Transportation Systems. *IEEE Transactions on Intelligent Transportation Systems*, 11(2):485–497.

Dekker, M., Hameete, P., and Hegemans, M. (2012). HactarV2: an agent team strategy based on implicit coordination. *Programming Multi-Agent Systems , LNCS*, 7217:173–184.

Ehlert, P. and Rothkrantz, L. (2001). Microscopic traffic simulation with reactive driving agents. In *ITSC. 2001 IEEE Intelligent Transportation Systems.*, pages 860–865. IEEE.

Hidas, P. (2002). Modelling lane changing and merging in microscopic traffic simulation. *Transportation Research Part C: Emerging Technologies*, 10(5-6):351–371.

Hindriks, K. V., de Boer, F. S., van der Hoek, W., and Meyer, J.-J. C. (2001). Agent programming with declarative goals. *Intelligent Agents VII Agent Theories Architectures and Languages, LNCS*, 1986:228–243.

Hindriks, K. V., van Riemsdijk, B., Behrens, T., Korstanje, R., Kraayenbrink, N., Pasman, W., and de Rijk, L. (2011). Unreal goal bots. *Agents for Games and Simulations II, LNCS*, 6525:1–18.

Hoogendoorn, R., Hoogendoorn, S. P., Brookhuis, K., and Daamen, W. (2010). Mental Workload, Longitudinal Driving Behavior, and Adequacy of Car-Following Models ... *Transportation Research Record*, 2188:64–73.

Kesting, A., Treiber, M., and Helbing, D. (2007). General Lane-Changing Model MOBIL for Car-Following Models. *Transportation Research Record*, 1999(1):86–94.

Krajzewicz, D. and Hertkorn, G. (2002). Sumo (simulation of urban mobility). *Proc. of the 4th Middle East Symposium on Simulation and Modelling*, pages 183—-187.

Langley, P., Laird, J. E., and Rogers, S. (2009). Cognitive architectures: Research issues and challenges. *Cognitive Systems Research*, 10(2):141–160.

Lee, L., Nwana, H., Ndumu, D., and Wilde, P. D. (1998). The stability, scalability and performance of multi-agent systems. *BT Technology Journal*, 16(3):94–103.

Nagel, K. and Rickert, M. (2001). Parallel implementation of the TRANSIMS micro-simulation. *Parallel Computing*, 27(12):1611–1639.

Navarro, L., Flacher, F., and Corruble, V. (2011). Dynamic level of detail for large scale agent-based urban simulations. *Proc. AAMAS*, 2:701–708.

Ossen, S. and Hoogendoorn, S. P. (2011). Heterogeneity in car-following behavior: Theory and empirics. *Transportation Research Part C: Emerging Technologies*, 19(2):182–195.

Pokahr, A., Braubach, L., and Lamersdorf, W. (2005). Jadex: A BDI reasoning engine. *Multi-Agent Programming*, pages 149–174.

Rindsfüser, G. (2005). Multi Agent System Simulation for the Generation of Individual Activity Programs. *Applications of Agent Technology in Traffic and Transportation*, pages 165–180.

Rossetti, R. J., Bordini, R. H., Bazzan, A. L., Bampi, S., Liu, R., and Vliet, D. V. (2002). Using BDI agents to improve driver modelling in a commuter scenario. *Transportation Research Part C: Emerging Technologies*, 10(5-6):373–398.

Salvucci, D. D. (2006). Modeling Driver Behavior in a Cognitive Architecture. *Human Factors and Ergonomics Society*, 48(2):362–380.

Schakel, W. J., Knoop, V. L., and van Arem, B. (2012). Integrated Lane Change Model with Relaxation and Synchronization. *Transportation Research Record*, 2316:47–57.

Smith, L., Beckman, R., and Baggerly, K. (1995). TRANSIMS: Transportation analysis and simulation system. Technical report, Los Alamos National Lab (United States).

Sukthankar, R., Baluja, S., and Hancock, J. (1998). Multiple Adaptive Agents for Tactical Driving. *Applied Intelligence*, 9(1):7–23.

Treiber, M., Hennecke, A., and Helbing, D. (2000). Congested Traffic States in Empirical Observations and Microscopic Simulations. *Physical Review E*, pages 1805–1824.

Treiber, M. and Kesting, A. (2013). *Traffic Flow Dynamics - Data, Models and Simulation*. Springer.

Turner, P. and Jennings, N. (2001). Improving the scalability of multi-agent systems. *Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems, LNCS*, 1887:246–262.

Wolfe, S. R., Sierhuis, M., and Jarvis, P. A. (2008). To BDI, or not to BDI: design choices in an agent-based traffic flow management simulation. In *Proc. SpringSim '08*, pages 63–70.