Synthesising Monitors for Autonomous Traffic (Research Abstract)*

Christopher Bischopink

Department of Computing Science, University of Oldenburg, Germany
bischopink@informatik.uni-oldenburg.de

Abstract. In this research abstract, I cover an approach to synthesise monitors from specifications for cars in autonomous traffic. These monitors can be used to restrict the behaviour of the cars, so that a system consisting of cars equipped with the monitors satisfies certain properties. As the specification language, an extended version of Extended Multi-Lane Spatial Logic (EMLSL), a two-dimensional logic to reason about (highway) traffic is used.

Keywords: Multi-Lane Spatial Logic · Monitor Synthesis · Distributed Car Controllers · Cooperative Cars

1 Introduction

In recent years, the topic of autonomous driving got more and more popular to a wide audience. Indications for this are e.g. the DARPA Urban Challenge from 2007 or the European Project SARTRE [3]. As (partially) autonomous vehicles begin to enter the market, one might want them to behave provable correct in every situation. A step in this direction was e.g. done in [5], where Hilscher et al. introduced a Logic for reasoning about highway traffic. Another popular work is [7], where Loos et al. verified an adaptive cruise control for highway traffic.

Later work (extending [5], most remarkable [4], [6] and [9]) aimed at providing safe controllers for autonomous cars for country road and urban traffic.

Most work mentioned by now however aimed at constructing a controller for such vehicles and afterwards proving its correctness. Additionally, only safety properties were almost always considered. My aim differs from this, as I want to provide an approach that enables cars to always behave according to their own and the specifications of the overall system, which may include more than safety specifications. For this, I proceed as follows: First, there is need for a specification language that enables one to specify properties of a car's and the overall system's behaviour. Second, I synthesise monitors for these specifications. Third, we need interaction between monitors and the controllers of the cars, so that controllers may ask monitors if they are allowed to execute an action without violating some car's properties. This way, all individual and local actions participate in the satisfaction of the specifications of the whole system.

* This research was supported by the German Research Council (DFG) in the Research Training Group DFG-GRK 1765: "System Correctness under Adverse Conditions".

Another approach to the problem of safe control of systems was proposed in [10] and uses learning techniques to build a Timed Automaton [2] as a controller that acts correct for (at least) the training set. In my approach, it is planed that the controllers act correct for every reality quantified by the specifications, not only for some of them. It is therefore more similar to [1], which – additionally to learning – uses a monitor (*"shield"*) that allows only certain actions.

2 Recent and Ongoing Work

The formal traffic model my work is based on was introduced in [5] and is called a *traffic snapshot*. The graphical representation of an exemplary traffic snapshot TS is depicted in Fig. 1. In a traffic snapshot, the reservations, claim, position, speed and acceleration of all cars are stored. In the excerpt of TS depicted in Fig. 1, there are two cars, A and B. Car A is driving on lane 0 at position 2.2, car B is on the same lane some space behind. Car B also has a claim on lane 1 (achieved by formerly executing the action $c(B, 1)$), indicating that it is preparing to change to this lane. The next step for the car would be to change the claim to a reservation by executing $r(B)$. The resulting traffic snapshot would be similar to the actual one with the only difference that car B now has two reservation, one on each lane. Formulae of EMLSL can be evaluated on traffic snapshots, on TS e.g. the formula $\langle {}^{cl(B)}_{re(B)} \rangle$ holds, stating that somewhere within the traffic snapshot there is a reservation on one lane and a claim (both of car B) on the lane above. Restricting the actions between traffic snapshots to fulfil certain properties for the whole system is the major goal of my research.

Recently, the aim was to develop a suitable approach to guarantee different kind of properties given in some specification language for all cars. For an easy-to-understand but powerful specification language, I decided to extend EMLSL [6] towards a timed version called *Timed* MLSL (TMLSL) that shares basic concepts with Timed Linear Temporal Logic (TLTL) [8]. The following formula illustrates the new component of this logic:

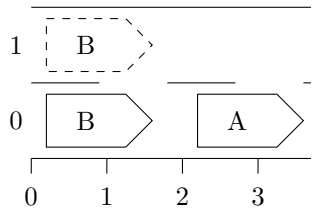

Fig. 1. Excerpt of a traffic snapshot TS.

$$blive \equiv \forall\gamma(\langle cl(\gamma)\rangle \rightarrow \rhd_{\langle {}^{re(\gamma)}_{re(\gamma)} \rangle} \in [0, t))$$

This formula states that if there is somewhere a claim of some car γ, the formula $\langle {}^{re(\gamma)}_{re(\gamma)} \rangle$ has to hold at some point in the future in the interval $[0, t)$. Analogously to $\rhd_\varphi \in \mathcal{I}$ stating that the formula φ holds somewhere in the future, the logic also contains $\lhd_\varphi \in \mathcal{I}$ stating that a formula has to hold somewhere in the past in $\mathcal{I}$. For specifications like this, I plan to synthesise monitors that observe the evolution of traffic snapshots. If a controller then wants to execute an action, the monitor checks whether the action will violate some car's specification or not. Different solutions are possible to achieve such a behaviour. One solution could be that the car itself has a monitor for each specification of each other

car. This way, no communication between different cars would be needed. On the other hand, this solution might be unnecessary computational expensive, as all properties are monitored by all cars, which adds a lot of redundancy. Another solution would be that a car only monitors its own specifications and communication is used between the cars to evaluate whether an action is possible.

A possible monitor we would like to synthesise is depicted in Fig. 2 in the shape of some extended version of Timed Automata. At this stage, the monitor is entirely passive and is only a handy way of representing the necessary knowledge about the system in a closed manner. The benefit of this is that there is no need for storing the actual evolution of traffic snapshots, which would otherwise be needed to evaluate a TMLSL formula. Based on the location and clock values, the actions a controller plans to execute can be evaluated.

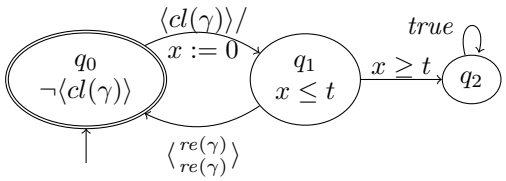

Fig. 2. Monitor for live lane change.

Starting with a most permissive controller only checking the basic application conditions of the actions (e.g. for reservations and claims) shown in Fig. 3, one adds more and more constraints to each action, based on the number of specifications.

If it is e.g. desired that a system controlled by such controllers satisfies the property *blive*, only the monitor depicted in Fig. 2 and a new synchronisation expression between the controller and the monitor(s) need to be added. This includes an expression in the controller asking for each action if it is valid to execute it and an additional transition for every location of the monitor that checks if the action is valid with respect to that monitor. While the concrete approach for the synchronisation with the monitors is not yet fixed, it is planned that the expressions are synthesised too.

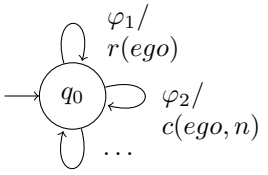

Fig. 3. A Controller without any restrictions from specifications.

The benefit of handling specifications this way is that controllers are easily extensible to new specifications. For a new specification, one only need to synthesise a monitor and add another expression to all transitions of the controller that check with the help of the monitors if the action is valid. It is expected that the synthesis of the monitors is less computationally expensive as e.g. model checking a newly constructed controller satisfying all formerly specifications and the new specification. This applies especially if the constructed controller was incorrect and multiple iterations of constructing a controller are needed.

3 Future Work

For the moment, all work mentioned was sketched for highway traffic only. In the future, we would like to extend it to country roads (including opposing cars)

and urban traffic (including intersections). Indeed, EMLSL and its model were extended to other traffic scenarios too, so our approach will in a later version include these extensions. In doing so, it seems interesting to investigate other properties such as timed-rechability of a certain location in a traffic snapshot.

Another planned extension is to focus on imperfect knowledge about the traffic participants. For this reason, we would like to take *Views* [5] into account. Views restrict the available information about cars to such cars that are nearby the own car. We expect that this will lead to a violation of e.g. the liveness property formerly guaranteed by the monitors if the parameters of the system remains unconstrained. By finding and restricting the parameters (such as maximum speed, minimal breaking forces etc.) to certain values, it is expected that the specified properties can be guaranteed again. Additionally to the investigation of the impact of these parameters, other properties of the model itself, such as maximum throughput and speed limitations are examined.

For all the cases mentioned in both ongoing and future work, tool support is desirable. It is planned to implement such tools and later test the efficiency of the approach to compare it with other solutions. This means the efficiency of the synthesis as well as the computational effort at runtime and the performance of a system containing only cars equipped with this kind of controllers and monitors.

References

1. Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., Topcu, U.: Safe reinforcement learning via shielding. CoRR **abs/1708.08611** (2017)
2. Alur, R., Dill, D.L.: A theory of timed automata. Theor. Comput. Sci. **126**(2)
3. Chan, E., Ekfjorden, A., Jootel, P., Gidney, J., Dávila, A., Brännström, M., Skarin, D., Wahlström, L.: SAfe Road TRains for the Environment (SARTRE): Project final report. Tech. rep. (2012), www.sartre-project.eu/en/publications/Documents/SARTRE_Final-Report.pdf
4. Hilscher, M., Linker, S., Olderog, E.R.: Proving safety of traffic manoeuvres on country roads. In: Liu, Z., Woodcock, J., Zhu, H. (eds.) Theories of Programming and Formal Methods. LNCS, vol. 8051, pp. 196–212. Springer (2013)
5. Hilscher, M., Linker, S., Olderog, E.R., Ravn, A.: An abstract model for proving safety of multi-lane traffic manoeuvres. In: Qin, S., Qiu, Z. (eds.) Int'l Conf. on Formal Engineering Methods (ICFEM) (2011)
6. Linker, S.: Proofs for traffic safety - combining diagrams and logic. Ph.D. thesis, Universität Oldenburg (2015), http://oops.uni-oldenburg.de/2337/
7. Loos, S.M., Platzer, A., Nistor, L.: Adaptive cruise control: Hybrid, distributed, and now formally verified. In: Butler, M., Schulte, W. (eds.) FM 2011: Formal Methods. Springer (2011)
8. Raskin, J.F., Schobbens, P.Y.: The logic of event clocks: Decidability, complexity and expressiveness. Automatica **34**(3), 247–282 (Mar 1998)
9. Schwammberger, M.: An abstract model for proving safety of autonomous urban traffic. Theoretical Computer Science **744**, 143–169 (2018)
10. Tappler, M., Aichernig, B.K., Larsen, K.G., Lorber, F.: Time to learn – learning timed automata from tests. In: André, É., Stoelinga, M. (eds.) Formal Modeling and Analysis of Timed Systems. Springer (2019)