Controller Synthesis and Verification for Multi-Agent Systems

Rong Gu
Mälardalen University, Västerås, Sweden
Email: (first.last)@mdh.se

***Abstract*—Controller synthesis and verification are crucial in the design of Multi-Agent Systems (MAS), as the controller serves as the brain of the autonomous systems, which are often safety- and mission-critical. In this study, we propose a two-layer framework for formal modeling and verification of MAS. The static layer of the framework focuses on mission planning that involves path planning and task scheduling, whereas the dynamic layer of the framework focuses on the mission execution, where the continuous motion of the agents, the uncertain occurrence of moving obstacles, and the design details of the embedded control systems are considered. Specifically, the framework adopts timed automata and reinforcement learning for mission planning, and hybrid automata, stochastic timed automata, and statistical model checking for mission execution and collision avoidance. The method of scalable mission-plan synthesis is implemented as a tool called TAMAA, which also provides GUI for mission and environment configuration. This approach and tool are evaluated in an industrial use case: autonomous quarry that is provided by VOLVO CE, Sweden.**

I. Introduction

Multi-Agent Systems (MAS) are consisted of multiple agents that move and function autonomously. The design of controllers of these systems involves path planning and task scheduling. The former requires the ability of calculating collision-free paths that lead to the destination, whereas the latter requires the agents to schedule their tasks effectively so that the synthesized plans satisfy various requirements, e.g., task execution order, timing constraints, etc.

In order to solve this problem and verify the synthesized controller in a realistic environment model containing uncertainties, we propose a novel approach that utilizes formal methods, advanced path-planning and collision-avoidance algorithms, as well as reinforcement learning. We implement the approach in a tool that provides a user-friendly GUI for mission management and environment configuration, and a calculation core for calling the aforementioned algorithms, generating formal models and temporal logic queries, as well as synthesizing controllers. To verify the synthesized controllers, we use hybrid automata (HA) as the modeling language to mimic the continuous motion of agents and stochastic timed automata to model the probabilistic events in the environment. By simulating and statistically verifying the HA model that is equipped with the synthesized controller, we obtain qualitative and quantitative analysis of the MAS.

As the controller synthesis and verification concern two aspects: mission planning and execution, which focus on the discrete and continuous features of the systems respectively,

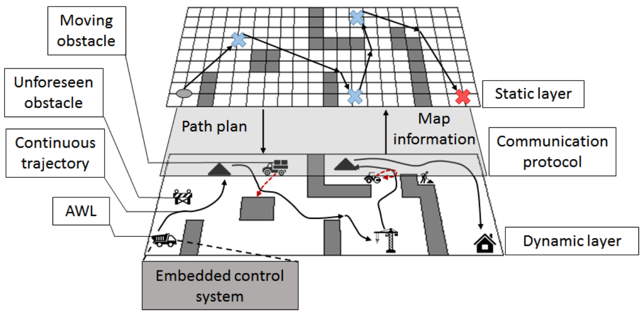


Fig. 1. A two-layer framework

we propose a two-layer framework to provide a separation-of-concerns when designing MAS. Last but not least, when the number of agents in MAS increases, the complexity of the problem grows exponentially. Hence, we propose a method combining model checking with reinforcement learning to alleviate the state-space-explosion problem. Evaluation of the methods are conducted on an industrial use case: an autonomous quarry.

II. A Walk Through the Methods and Tool Chain

Modeling and Verification Framework. There are two main functionalities of MAS that need to be realized: mission planning and execution. The former concerns path planning and task scheduling, which focuses on the discrete feature of the system, whereas the latter concerns path following, task execution, and collision avoidance, which takes into account the continuous motions of the system. The inherent difference motivates us to design a two-layer framework for modeling and verification in order to provide a separation-of-concerns when designing MAS. In paper [1], we propose an initial design of the framework consisting of a *static layer* and a *dynamic layer*, which is depicted in Figure 1. The communication protocol supports data exchange between the layers. The *static layer* is responsible for path and mission planning, based on the information of the environment detected by the *dynamic layer*. The static layer includes static obstacles and milestones where the tasks should be carried out. Moving obstacles that are unforeseen by the autonomous vehicles are considered in the *dynamic layer*, which is designed to simulate and verify the systems to guarantee that they follow the reference path generated by the *static layer* and avoid dynamic obstacles. These two layers support the modeling of mission planning

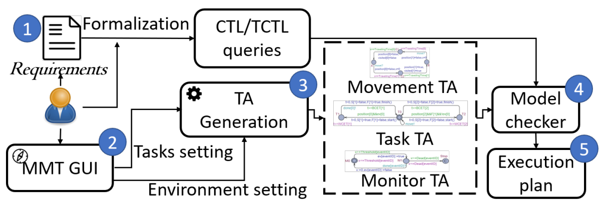


Fig. 2. The Process of TAMAA

and continuous behavior separately, such that the desired decoupling is achieved.

TAMAA. Mission planning includes path planning and task scheduling. Classic path-planning algorithms provide a means of calculating static paths between two positions in the environment. However, when the requirement includes temporal logic constraints, such as repetitively executing tasks B and C after task A is done, path-planning algorithms are not enough. Hence, we propose a method called Timed-Automata-based mission planner for Multiple Autonomous Agents (TAMAA) and implement it as a tool that is connected with a graphic mission management tool called MMT [2].

Overall, the approach is composed of the steps shown in Figure 2: i) Step 1 - formalizing the requirements into CTL/TCTL queries, ii) Step 2 - configuring the information of the environment and tasks in MMT, iii) Step 3 - automatically generating the UPPAAL TA of movement, tasks, and monitors, iv) Step 4 - verifying models generated in Step 3 in UPPAAL against the queries of Step 1, and generating execution traces that satisfy or violate the queries, and v) Step 5 - using the traces to obtain the mission plans in cases when the requirements are met, or counter-examples when no mission plan exists in the environment configuration. Since this is an automatic approach, users are only involved in the first two steps in the configuration phase of Figure 2.

MCRL. Experimental results show that TAMAA is capable of synthesizing mission plans satisfying various requirements, e.g., reachability, safety, timing ones [2]. However, when the number of agents increases, the method fails to generate any result, as it uses exhaustive model checking, which leads to the notorious state-space-explosion problem when the complexity of the model grows. Therefore, we propose a novel approach for synthesizing mission plans by using formal methods while still keeping its ability of exhaustive verification, namely MCRL [3]. The method combines model checking with reinforcement learning so that instead of exhaustively exploring the state space of the model, the method uses random simulation and a reinforcement learning algorithm, i.e., Q-learning, to visit the states of the model and stores the possible actions and their rewards at each state. Thereafter, a Q-table is populated and injected back to the model of the MAS. The new model contains a conductor for each of the agents, which reads the Q-table and always select the action that owns the highest reward at the current state and does not conflict with

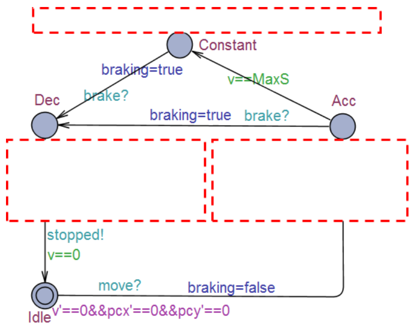


Fig. 3. The skeleton of the pattern

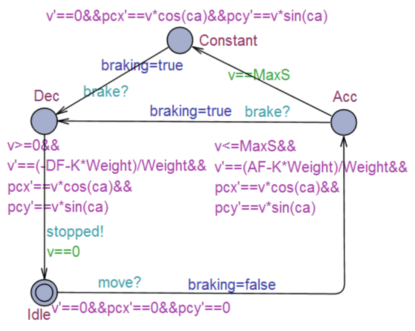


Fig. 4. The hybrid automaton of the pattern

other agents. In this way, the behavior of the MAS model is restricted by the Q-table so that when one performs exhaustive verification on the model, the state space is much constrained. Paper [3] presents the experiment of a comparison among TAMAA, MCRL, and UPPAAL STRATEGO. The result shows that the computation time of MCRL increases linearly as the number of agents grows, whereas the other two methods raise exponentially.

Pattern-based Modeling and Statistical Verification of the Dynamic Layer. As the dynamic layer concerns the continuous motions of the agents, we adopt hybrid automata as the modeling language and UPPAAL SMC as the model checker to conduct statistical model checking. For example, the linear motion of the agents are modeled as a hybrid automaton as it is depicted in Figure 4. The changing rates of the positions and velocity are described by ordinary differential equations (ODE) based on Newtonian laws of motion (see Figure 4). As replaceable modules of the model, the ODE are replaced by blank boxes in Figure 3 that presents the skeleton of this pattern. Beside the linear motion component, the hybrid-automata model of the agents include rotation component, vision component and controlling component. The vision component captures the information of the environment and sends it to the controlling component that runs A* [4] or Theta* algorithm [5] for path planning and a collision-avoidance algorithm based on dipole flow field [6] to make decisions of moving speed and directions. The environment model contains moving obstacles that are predefined before the verification starts, and unforeseen obstacles that are generated during the verification, which simulates the scenario where moving objects suddenly appear in front of the agents.

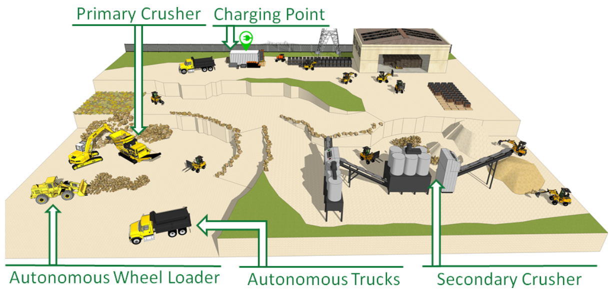
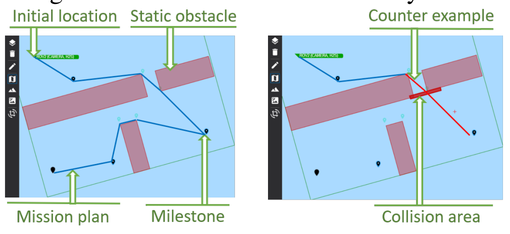


Fig. 5. An example of an autonomous quarry

Statistical model checking is conducted on the model and the result shows that the agents can manage to avoid all obstacles and reach the obstacles in most of the cases, but fails when the moving obstacles rush recklessly towards the agents [1].

III. Evaluation

The evaluation of the methods proposed by this study is based on an industrial use case: an autonomous quarry that is provided by our industrial partner VOLVO CE. As an example, in Figure 5 we show the case of an autonomous quarry that contains several autonomous vehicles such as trucks and wheel loaders. According to requirements from VOLVO CE, an autonomous wheel loader digs a given stone pile and loads them into autonomous trucks, which carry an amount of stones to a primary crusher that crushes the stones at given fractions, after which the trucks continue to transfer the stones to the secondary crusher and finish their one-round job. During this process, the vehicles must go to the charging point when their battery-level is low. In addition, the vehicles must finish carrying all the stones within a time limit to guarantee a certain level of productivity. To solve the mission planning and verification problem in this use case, we configure the environment and vehicles in MMT and automatically generate the formal model of agents by using TAMAA, then synthesize mission plans by MCRL and depict the result in MMT as it is shown in Figure 6. To evaluate the scalability of the method,



(a) A mission plan (b) A counter example

Fig. 6. Two screenshots of the MMT user interface

we experimented several scenarios with different numbers of tasks, milestones, and agents. The result shows that MCRL is able to handle more agents than TAMAA that uses exhaustive model checking and UPPAAL STRATEGO and the explored states and time consumption of MCRL are much less than the other two methods [3].

IV. Conclusion and Future Work

In this thesis, we propose a two-layer framework for the controller synthesis and verification of MAS. The main contribution of this framework is to decouple the discrete mission planning from the verification of concrete execution and collision avoidance in continuous environments. To facilitate mission planning, we implement a tool called TAMAA, which contains the implementation of our model-generation algorithms and connects to UPPAAL and a graphic user interface for mission management, i.e., MMT. To improve the ability of handling multiple agents for TAMAA, we combine model checking technique with reinforcement learning, and conduct a series of experiments to compare the new approach with the original TAMAA and UPPAAL STRATEGO. For the dynamic layer of the framework, we propose a model in the format of hybrid automata, to describe the discrete state transition of the systems as well as dynamics and kinematics of autonomous vehicles and unforeseen obstacles. As the embedded control software is complex, we propose a pattern-based modeling method to facilitate the modeling process and enable reuse. This approach has demonstrated to be correct by building models in UPPAAL SMC and conducting a series of statistical analysis of a real-world industrial system: the autonomous quarry use case, provided by VOLVO CE.

The future work has several possible directions. One is to integrate the two layers of the framework so that they communicate in a real-time manner and the mission planning and verification are both optimized in this way. Another direction is about improving the MCRL approach by embedding the reinforcement learning into the state-space exploration of the model when running verification. This can be achieved by leveraging the function of calling external functions in UPPAAL STRATEGO, or implementing a customized model checker by leveraging existing libraries and frameworks, such as Plasma [7], and Storm [8].

References

[1] R. Gu, R. Marinescu, C. Seceleanu, and K. Lundqvist, "Towards a two-layer framework for verifying autonomous vehicles," in *NASA Formal Methods Symposium*. Springer, 2019, pp. 186–203.

[2] R. Gu, E. P. Enoiu, and C. Seceleanu, "Tamaa: Uppaal-based mission planning for autonomous agents," in *The 35th ACM/SIGAPP Symposium On Applied Computing*, April 2020. [Online]. Available: http://www.es.mdh.se/publications/5685-

[3] R. Gu, E. P. Enoiu, C. Seceleanu, and K. Lundqvist, "Combining model checking and reinforcement learning for scalable mission planning of autonomous agents." Mälardalen Real-Time Research Centre, Mälardalen University. [Online]. Available: http://www.es.mdh.se/publications/5782-

[4] S. Rabin, "Game programming gems, chapter a* aesthetic optimizations," *Charles River Media*, 2000.

[5] K. Daniel, A. Nash, S. Koenig, and A. Felner, "Theta*: Any-angle path planning on grids," *Journal of Artificial Intelligence Research*, vol. 39, pp. 533–579, 2010.

[6] L. A. Trinh, M. Ekström, and B. Cürüklü, "Toward shared working space of human and robotic agents through dipole flow field for dependable path planning," *Frontiers in neurorobotics*, vol. 12, 2018.

[7] A. Legay, S. Sedwards, and L.-M. Traonouez, "Plasma lab: a modular statistical model checking platform," in *International Symposium on Leveraging Applications of Formal Methods*. Springer, 2016, pp. 77–93.

[8] C. Dehnert, S. Junges, J.-P. Katoen, and M. Volk, "A storm is coming: A modern probabilistic model checker," in *Computer Aided Verification*, R. Majumdar and V. Kunčak, Eds. Cham: Springer International Publishing, 2017, pp. 592–600.