S. Götz, L. Linsbauer, I. Schaefer, A. Wortmann (Hrsg.): Software Engineering 2021 Satellite Events,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2021 1

Formal Verification of Intelligent Cyber-Physical Systems with the Interactive Theorem Prover KeYmaera X

Paula Herber[1] Julius Adelt[1] Timm Liebrenz[1]

Abstract: Cyber-physical systems are increasingly used in dynamic environments, and have to make safety-critical decisions autonomously. Machine learning is a powerful technique to make such decisions. For example, reinforcement learning can be used to learn controllers that outperform manually designed controllers in highly dynamic or uncertain environments. However, the learning process and the resulting controllers often impede system verification, as their behavior is hard to predict. To formally guarantee safety properties, a formal description is required, which is often not available in industrial design processes and hard to obtain for unpredictable machine learning components. In this paper, we briefly discuss our current work on the semi-automatic deductive verification of intelligent cyber-physical systems. To support industrial design processes, we target the widely used modeling language Simulink together with its Reinforcement Learning Toolbox. To support deductive formal verification, we propose to use and extend our existing framework for the service-oriented verification of hybrid systems that are modeled in Simulink. The framework enables the automated transformation of Simulink models or services into differential dynamic logic and formal verification with the interactive theorem prover KeYmaera X.

Keywords: Formal Verification; Reinforcement Learning; Cyber-Physical Systems

1 Introduction

Cyber-physical systems (CPS) are increasingly used in dynamic environments, for example, in smart factories or autonomous driving. With machine learning, CPS can adapt their behavior to highly dynamic and uncertain environments. The use of machine learning, however, is typically based on a statistical learning process, which makes the decisions hard to predict. If a CPS is used in a safety-critical application, where a failure may result in enormous cost or even endanger human lives, this poses a serious problem. To verify that the system behaves correctly for all possible input scenarios, a precise description of the possible system behavior is needed, i.e., a formal model. In industrial design processes, such formal models are often not available. Existing approaches for safe learning typically assume that a formal model, e.g., a Markov decision process, is provided by the designer. In this extended abstract, we sketch an approach for the formal verification of intelligent CPS using transformations from industrial design languages into formal languages together with contracts and abstractions for intelligent components. Furthermore, we briefly discuss our current work on the semi-automatic deductive verification of intelligent CPS with the

[1] University of Münster, Computer Science Department, Embedded Systems Group, Einsteinstr. 62, 48149 Münster, Germany {paula.herber,julius.adelt,timm.liebrenz}@uni-muenster.de

interactive theorem prover KeYmaera X. To support industrial design processes, we target the widely used modeling language Simulink together with its Reinforcement Learning (RL) Toolbox [Th21]. To support deductive formal verification, we propose to use and extend our existing framework for the service-oriented verification of hybrid systems that are modeled in Simulink [LHG18, LHG19] with KeYmaera X [Fu15].

2 Related Work

There exist some approaches to provide formal semantics to industrial modeling languages, e.g. [HRB13, AIER14, MF16, He18]. Most of them are based on an automated transformation from a system description with informally defined semantics (e.g., a SystemC design [IE11] or a Simulink model [Th21]) into a formally well-defined language. This enables computer-aided formal verification. However, existing approaches do not support intelligent systems. In [FP18], the authors ensure the safety of an RL controller via verified runtime monitors that are based on a dL model. In [Al17], a shield is used to substitute unsafe for safe actions. The shield is synthesized from a safety automaton and an abstraction of the environment. These methods, however, require a formal model, which is typically not available in industrial design processes and hard to obtain for machine learning algorithms.

3 Formalization and Verification of Intelligent CPS

The key idea of our approach for the formal verification of intelligent CPS is to augment existing approaches that provide transformations from industrially used design languages into formal models with support for learning components. The general idea is depicted in Fig. 1. In transformation-based approaches, the informally defined execution semantics of industrially used design languages is automatically translated into an existing formally well-defined language. By using existing formal languages, we gain access to existing verification tools, which we can use to verify given requirements, for example, safety, liveness, or timing properties. If the requirements are not satisfied, we typically get a counter-example, which can be used for debugging. In previous work, for example, we

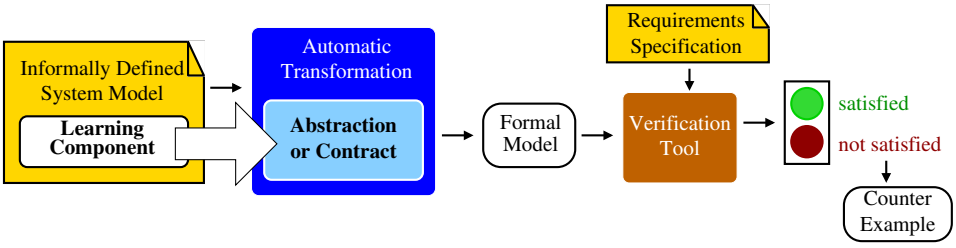

Fig. 1: Formalization of Intelligent CPS

have presented a transformation from SystemC into timed automata, which enables the fully-automatic verification of HW/SW co-designs with the UPPAAL model checker [HPG15], and an automated transformation from Simulink into dL (Simulink2dL) [LHG18], which enables deductive verification of hybrid systems with KeYmaera X. To leverage transformation-based formalizations to intelligent CPS, we assume that the informal system model contains learning components, for example RL agents. RL is a class of machine learning methods for learning in a trial and error approach by interacting with an environment through actions. Their behavior is hard to describe. To overcome this problem, we propose to capture embedded RL components by contracts or abstractions, which provide an over-approximation of their possible safe behavior, for example, as a concise description of safe actions in given system states. With that, we can then verify safety properties of the overall system under the assumption that the learning component only performs safe actions in each system state. To ensure that the RL component adheres to its contract or abstraction at runtime, we propose to use a safe RL approach as, for example, proposed in [FP18, Al17].

4 Verification of Intelligent CPS with KeYmaera X

To put our general approach into practice, we are currently working on a contract-based verification approach for intelligent CPS that are modeled with Simulink together with the RL Toolbox. The key idea is to use *hybrid contracts* in dL [LHG19] to formally describe RL agents, and to safely embed these contracts into the Simulink2dL transformation [LHG18]. We use a case study of an autonomous ground robot that moves around in a factory setting with moving obstacles. We have formalized the prerequisites the intelligent controller of the robot has to fulfill in order to avoid collisions, i.e., minimal safety distances, in a hybrid contract. By translating the system with Simulink2dL and embedding the hybrid contract of the RL agent, we are able to verify collision avoidance of the overall system with deductive verification techniques using KeYmaera X. Preliminary results indicate that this approach will enable us to precisely capture the safety-relevant behavior of learning components within complex CPS, formally verify crucial safety properties of the overall system, and automatically generate runtime monitors as proposed by [FP18] to ensure the safety-compliant behavior of the learning component at runtime.

5 Conclusion and Future Work

We have sketched an approach for the formal verification of intelligent CPS that are modeled in industrially used system design languages such as Simulink and the RL Toolbox. We believe that existing approaches for the formal verification of CPS urgently need to be leveraged to learning components, as the demand to cope with highly dynamic, uncertain environments is on the rise. The statistical, trial-and-error approach of such components leads to inherently hard to predict behavior. We propose to use contracts and abstractions to overcome this problem. However, how to define good contracts and abstractions, which

provide enough abstraction to enable scalable verification and still capture safety-relevant behavior of learning components precisely enough such that the component is not restricted too much at runtime, is an open challenge for future work. We also plan to extend our SystemC to timed automata engine to cope with intelligent CPS. As the UPPAAL model checker does not support contracts, we plan to base this work on abstractions.

Bibliography

[AIER14] Araiza-Illan, Dejanira; Eder, Kerstin; Richards, Arthur: Formal verification of control systems' properties with theorem proving. In: UKACC Int. Conference on Control (CONTROL). IEEE, pp. 244–249, 2014.

[Al17] Alshiekh, Mohammed; Bloem, Roderick; Ehlers, Rüdiger; Könighofer, Bettina; Niekum, Scott; Topcu, Ufuk: Safe Reinforcement Learning via Shielding. arXiv preprint arXiv:1708.08611, 2017.

[FP18] Fulton, Nathan; Platzer, André: Safe Reinforcement Learning via Formal Methods: Toward Safe Control Through Proof and Learning. The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18), 32(1):6485–6492, 02 2018.

[Fu15] Fulton, Nathan; Mitsch, Stefan; Quesel, Jan-David; Völp, Marcus; Platzer, André: KeYmaera X: An Axiomatic Tactical Theorem Prover for Hybrid Systems. In: International Conference on Automated Deduction. Springer, pp. 527–538, 2015.

[He18] Herdt, Vladimir; Le, Hoang M; Grosse, Daniel; Drechsler, Rolf: Verifying SystemC using intermediate verification language and stateful symbolic simulation. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 38(7):1359–1372, 2018.

[HPG15] Herber, Paula; Pockrandt, Marcel; Glesner, Sabine: STATE – a SystemC to Timed Automata Transformation Engine. In: ICESS. IEEE, 2015.

[HRB13] Herber, Paula; Reicherdt, Robert; Bittner, Patrick: Bit-precise formal verification of discrete-time MATLAB/Simulink models using SMT solving. In: Int. Conference on Embedded Software (EMSOFT). IEEE, pp. 1–10, 2013.

[IE11] IEEE Standards Association: , IEEE Std. 1666–2011, Open SystemC Language Reference Manual. IEEE Press, 2011.

[LHG18] Liebrenz, Timm; Herber, Paula; Glesner, Sabine: Deductive Verification of Hybrid Control Systems Modeled in Simulink with KeYmaera X. In: International Conference on Formal Engineering Methods. Springer, pp. 89–105, 2018.

[LHG19] Liebrenz, Timm; Herber, Paula; Glesner, Sabine: A Service-Oriented Approach for Decomposing and Verifying Hybrid System Models. In: International Conference on Formal Aspects of Component Software. Springer, pp. 127–146, 2019.

[MF16] Minopoli, Stefano; Frehse, Goran: SL2SX translator: from Simulink to SpaceEx models. In: Int. Conf. on Hybrid Systems: Computation and Control. ACM, pp. 93–98, 2016.

[Th21] The MathWorks: , Simulink. `www.mathworks.com/products/simulink.html`, 2021.