

Computer-Assisted Engineering for Robotics and Autonomous Systems

Edited by

Erika Abraham¹, Hadas Kress-Gazit², Lorenzo Natale³, and Armando Tacchella⁴

1 RWTH Aachen, DE, abraham@cs.rwth-aachen.de

2 Cornell University – Ithaca, US, hadaskg@cornell.edu

3 Italian Institute of Technology – Genova, IT, lorenzo.natale@iit.it

4 University of Genova, IT, armando.tacchella@unige.it

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 17071 “Computer-Assisted Engineering for Robotics and Autonomous Systems”. This seminar brought together researchers from three distinct communities – Robotics, Model-driven Software Engineering, and Formal Methods – to discuss the path towards creating safe and verifiable autonomous systems.

Seminar February 12–17, 2017 – <http://www.dagstuhl.de/17071>

1998 ACM Subject Classification D.2.4 Software/Program Verification, I.2.9 Robotics

Keywords and phrases analysis, artificial intelligence, autonomous systems, computer-aided software development, domain-specific languages, formal methods, model-driven software engineering, robotics, safety, synthesis, testing, verification

Digital Object Identifier 10.4230/DagRep.7.2.48

Edited in cooperation with Rüdiger Ehlers

1 Executive Summary

Hadas Kress-Gazit

Erika Abraham

Lorenzo Natale

Armando Tacchella

License © Creative Commons BY 3.0 Unported license

© Hadas Kress-Gazit, Erika Abraham, Lorenzo Natale, and Armando Tacchella

This seminar focused on *autonomous systems*, and more specifically robots, that operate without, or with little, external supervision. For these systems to be integrated into society, it is highly important to make sure that they are functionally safe. *Formal Methods* are techniques adopted in engineering for the verification of software and hardware systems. As models are a basic requirement for the formal analysis of systems, *Model-driven Software Engineering* plays an important role to enable the application of *Formal Methods*. Though autonomous systems are increasingly involved in our everyday life, both exact formalizations of safe functionality (standards, what we want to be confident in) and methods to achieve confidence (methodologies, how we get confident in the properties we want to assure) are still scarce.

This seminar brought together experts in *Artificial Intelligence* and *Robotics*, *Model-driven Software Engineering*, and *Formal Methods*. It included researchers from academia as well



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Computer-Assisted Engineering for Robotics and Autonomous Systems, *Dagstuhl Reports*, Vol. 7, Issue 2, pp. 48–63

Editors: Erika Abraham, Hadas Kress-Gazit, Lorenzo Natale, and Armando Tacchella



DAGSTUHL
REPORTS Dagstuhl Reports

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

as from industry. The following list summarizes high-level themes that emerged from the seminar:

- Dealing with highly complex systems, it is difficult to verify or even model all aspects of the system, therefore focusing effort on efficient falsification rather than costly verification can be highly impactful for industrial applications.
- The community can and should leverage results and systems built for different robotic competitions to reason about possible requirements and techniques to verify/falsify them. These competitions include the DARPA robotics challenge, the Amazon picking challenge, different leagues in Robocup, etc. Creating benchmarks based on these competitions will enable progress in verification of autonomous systems.
- Creating small interdisciplinary teams that include people from formal methods, robotics and model based design that tackle small yet realistic problems, possibly inspired by industrial applications, will help formalize the language of requirements, models and verification techniques that will have an impact on autonomous systems.

2 Table of Contents

Executive Summary

Hadas Kress-Gazit, Erika Abraham, Lorenzo Natale, and Armando Tacchella 48

Overview of Talks

The Power of Satisfiability Checking	52
Model-Driven Control Software / System Design for Robotic Systems	52
Safety Cases. Arguing the Safety of Autonomous Systems	53
Computer-Assisted Engineering for Robotics and Autonomous Systems: Verification Techniques That (May) Work in Practice	53
Towards Best-Effort Autonomy	54
Provably Safe Collision Avoidance in Dynamic Environments	55
Heteronomous Systems They are, Let's Face it.	56
GenoM3 Templates: from Middleware Independence to Formal Models Synthesis	56
Synthesis of Shared Control Protocols with Provable Safety and Performance Guarantees	57
A storm is Coming: A Modern Probabilistic Model Checker	57
High-Level Verifiable Robotics	58
(Learning to) Learn to Control	59
Optimizing the Performance of Robots in Production Logistics Scenarios	59
Artificial Intelligence Planning and Robotics and Autonomous Systems	60
Human-Robot Collaboration – Industrial Applications and Open Challenges	60
A Competition on Formal Methods for Robotics	61
Specification: the Biggest Bottleneck in Formal Methods and Autonomy	61
Development and Adoption of Model-Based Tools in Robotics	62
How Safe is Your Autonomous Robot? (A Tale of Courage, Passion, and Perspiration)	62
Participants	63

3 Overview of Talks

3.1 The Power of Satisfiability Checking

Erika Abraham (RWTH Aachen, DE)

License © Creative Commons BY 3.0 Unported license
© Erika Abraham

Joint work of Erika Abraham and Gereon Kremer

Main reference Erika Abraham, Gereon Kremer, “Satisfiability Checking: Theory and Applications”, in Proc. of the 14th Int’l Conf. on Software Engineering and Formal Methods (SEFM 2016), LNCS, Vol. 9763, pp. 9–23, Springer International Publishing, 2016.

URL http://dx.doi.org/10.1007/978-3-319-41591-8_2

Satisfiability checking aims to develop algorithms and tools for checking the satisfiability of existentially quantified logical formulas. For propositional logic, in the late '90s impressive progress was made towards practically applicable solutions, resulting in powerful SAT solvers. Driven by this success, a new line of research started to enrich propositional SAT solving with solver modules for different theories. Nowadays, sophisticated SAT-modulo-theories (SMT) solvers are available for, e.g., equality logic with uninterpreted functions, bit-vector arithmetic, array theory, floating point arithmetic, and real and integer arithmetic. SAT and SMT solvers are now at the heart of many techniques for the analysis of programs and probabilistic, timed, hybrid and cyber-physical systems, for test-case generation, for solving large combinatorial problems and complex scheduling tasks, for product design optimisation, planning and controller synthesis, just to mention a few well-known areas.

In this talk we gave a short introduction to the theoretical foundations of satisfiability checking, mentioned some of the most popular tools, and discussed the successful embedding of SMT solvers in different technologies.

3.2 Model-Driven Control Software / System Design for Robotic Systems

Jan Broenink (University of Twente, NL)

License © Creative Commons BY 3.0 Unported license
© Jan Broenink

In dealing with system architectures for robotic and automation systems, it is crucial to consider the total system (machine, control, software and I/O), because the dynamics of the machine influences the robot software. Therefore, we use appropriate Models of Computation and tools, namely bond graphs for the machine part, dataflow diagrams for the algorithm / software parts. Via meta-models, these formalisms are related. This allows for a structured approach for designing the architecture of the robotic system. The design work is done as a stepwise refinement process, whereby each step is verified via simulation, yielding shorter design time, and a better quality product. The tools use templates and pass model-specific information between each other via parameterised tokens in the generated, high-level code, to get a better separation of design steps. This allows for better quality of the models and more reuse, thus enhancing the efficiency of model-driven design for the (industrial) end user. This approach is illustrated with two case studies: the control stack for a mobile robot, manipulating blocks, and on incorporating safety layers in the embedded control system architecture.

3.3 Safety Cases. Arguing the Safety of Autonomous Systems

Simon Burton (Robert Bosch GmbH – Stuttgart, DE)

License  Creative Commons BY 3.0 Unported license
© Simon Burton

This talk introduced the topic of safety cases for arguing the safety of autonomous systems. Examples are given for where existing standards do not provide sufficient guidance to demonstrate certain properties of autonomous systems and therefore require a justification from “first principles”. The Goal Structuring Notation is described as a means of formulating and communication such argumentation structures. A roadmap for how to extend these concepts in combination with model-based Systems Engineering and formal methods is presented to motivate future research and encourage collaboration between these domains.

3.4 Computer-Assisted Engineering for Robotics and Autonomous Systems: Verification Techniques That (May) Work in Practice

Kerstin I. Eder (University of Bristol, GB)

License  Creative Commons BY 3.0 Unported license
© Kerstin I. Eder

Joint work of Dejanira Araiza-Illan, David Western, Pjotr Trojanek, Anthony G. Pipe, Arthur Richards, Kerstin I. Eder

This presentation is focused on practical techniques for the verification of autonomous systems. Because no single technique is adequate to cover a whole system in practice, the use of a variety of techniques is proposed, including formal and state-of-the-art simulation-based, to address verification needs in autonomous system design.

At the code level, re-implementing three well-known robot navigation algorithms in SPARK enables formal verification to establish freedom from run-time errors without performance penalties when compared to implementations in C/C++ [1]. This shows that selecting a programming language designed for software-reliability leads to significant advantages when it comes to establishing code correctness.

At the design level, an assertion-based approach is proposed to verify control system designs with respect to high-level requirements, such as stability, combining simulation-based techniques with automatic theorem proving [2]. Requirements are first formalized as properties over the signals in the Simulink model using Simulink blocks that then become part of the Simulink model. The so extended Simulink model is then automatically translated into Why3 theories and proof goals for formal verification using SMT-based theorem provers. A case study that illustrates how stability can be decomposed from a single high-level requirement into a set of sub-requirements to be implemented as assertions in Simulink is discussed [3], together with the advantages of combining assertion-checks performed during simulation with automatic theorem proving performed at system design time.

Coverage-Driven Verification (CDV) is as a systematic, goal directed simulation-based verification method that is capable of exploring systems of realistic detail under a broad range of environment conditions, providing a high degree of automation. I will illustrate the benefits of CDV, functional and situation coverage [4] together with model-based [5] as well as intelligent, agent-based test generation techniques [6] on the example of code used in robots that directly interact with humans.

I conclude my presentation with a brief discussion of the challenges in this area: specification, automation, combination of techniques and using AI for verification and validation.

Acknowledgement. The research presented is based on collaborations within the EPSRC funded projects “Robust Integrated Verification of Autonomous Systems” (EP/J01205X/1) and “Trustworthy Robotic Assistants” (EP/K006320/1).

References

- 1 Piotr Trojanek and Kerstin Eder. Verification and Testing of Mobile Robot Navigation Algorithms: A Case Study in SPARK. In *Proc. International Conference on Intelligent Robots and Systems (IROS)*, 2014.
- 2 Dejanira Araiza-Illan, Kerstin Eder, and Arthur Richards. Formal verification of control systems’ properties with theorem proving. In *2014 UKACC International Conference on Control (CONTROL)*, Loughborough, UK, July 2014.
- 3 Dejanira Araiza-Illan, Kerstin Eder, and Arthur Richards. Verification of Control Systems Implemented in Simulink with Assertion Checks and Theorem Proving: A Case Study. In *European Control Conference (ECC)*, Linz, Austria, 2015.
- 4 Dejanira Araiza-Illan, David Western, Anthony Pipe, and Kerstin Eder. Coverage-Driven Verification: An Approach to Verify Code for Robots that Directly Interact with Humans. In *Haifa Verification Conference*. Haifa, Israel, 2015. DOI: 10.1007/978-3-319-26287-1_5.
- 5 Dejanira Araiza-Illan, David Western, Anthony G. Pipe, and Kerstin Eder. Systematic and Realistic Testing in Simulation of Control Code for Robots in Collaborative Human-Robot Interactions. In *Towards Autonomous Robotic Systems (TAROS)*. June 2016.
- 6 Dejanira Araiza-Illan, Anthony G. Pipe, and Kerstin Eder. Intelligent agent-based stimulation for testing robotic software in human-robot interactions. In *Proceedings of the 3rd Workshop on Model-Driven Robot Software Engineering, MORSE’16*, pages 9–16, New York, NY, USA, 2016. ACM.

3.5 Towards Best-Effort Autonomy

Rüdiger Ehlers (*Universität Bremen, DE*)

License © Creative Commons BY 3.0 Unported license
© Rüdiger Ehlers

Joint work of Rüdiger Ehlers, Salar Moarref, Ufuk Topcu

Main reference Rüdiger Ehlers, Salar Moarref, Ufuk Topcu, “Risk-averse control of Markov decision processes with ω -regular objectives”, in *Proc. of the 55th IEEE Conference on Decision and Control (CDC 2016)*, pp. 426–433, IEEE, 2016.

URL <http://dx.doi.org/10.1109/CDC.2016.7798306>

URL <http://progirep.github.io/ramps/>

Highly autonomous systems degrade in performance over time, need to work correctly in off-nominal conditions, and need to adapt without the help of a human operator. We do not always know in advance of the system’s deployment how they are degrading in the long run, and not all possible degradation scenarios can be covered in a systematic system engineering process. To counter this problem, we could synthesize adapted control strategies at runtime, using action failure probabilities inferred from observed data. However, classical policy synthesis techniques for ω -regular specifications yield no policy in case of inevitable eventual violation of the specification. We present an approach to mitigate this problem for omega-regular specifications and environments that can be modelled as Markov decision processes.

3.6 Provably Safe Collision Avoidance in Dynamic Environments

Christian Heinzemann (Robert Bosch GmbH – Stuttgart, DE)

License  Creative Commons BY 3.0 Unported license
© Christian Heinzemann

For many applications of autonomous robots in intralogistics and mobile service robotics, it is an absolute must to guarantee that the robot will not cause harm to its environment. This particularly includes that the robot must not cause collisions with moving obstacles such as humans or animals. Guaranteeing collision-free motion of autonomous systems is increasingly hardened by the fact that these systems increasingly operate in shared, open-context environments. In such environments, the robot operates in the same space as the humans and we as the developers do not know all contexts in which the system will have to operate during its runtime. In particular, we will often not know how the environment looks like and which kinds of obstacles the system will face. To this end, an approach for guaranteeing provably safe motion of mobile robots is necessary. The main safety concept being adopted therefore is passive safety [1], requiring that the robot is not moving when a collision with an obstacle happens. The existing approaches either make the optimistic assumption of knowing the future behavior of any obstacle [2, 3], which is unrealistic for humans, or they make rather conservative assumptions about obstacles [4, 5, 6, 7] that significantly decrease the robot’s performance. The latter is true particularly in cases where many obstacles are in the robot’s environment and where these obstacles are relatively near to the robot, for example, when moving through an area populated by humans in a city center, airport, or train station. Probabilistic approaches to collision avoidance [8, 9] improve the performance but cannot give the necessary safety guarantees that we need for heavy robots used, for example, in intralogistics.

In this talk, I give briefly characterize the problem of collision avoidance to be solved for mobile robots and discuss in more detail why the problem is not solved sufficiently by existing approaches. A possible trail for future works could be online verification approaches based on reachability analysis [10] that use models to overapproximate the space that an obstacle will occupy at the end of a planning period of the reactive obstacle avoidance algorithms. I conclude by summarizing the key challenges that need to be solved for the approaches.

References

- 1 K. Macek, D. Vasquez, T. Fraichard, R. Siegwart: Towards Safe Vehicle Navigation in Dynamic Urban Scenarios. In: *Automatica* 50, Issue 3–4, pp. 184–194, 2009.
- 2 L. Martinez-Gomez and T. Fraichard: Collision avoidance in dynamic environments: An ICS-based solution and its comparative evaluation. In: *Proceedings of ICRA 2009*.
- 3 F. Large, C. Laugier, and Z. Shiller: Navigation Among Moving Obstacles Using the NLVO: Principles and Applications to Intelligent Vehicles. In: *Autonomous Robots* 19(2), 2005.
- 4 S. Bouraine, T. Fraichard, O. Azouaoui, H. Salhi: Passively safe partial motion planning for mobile robots with limited field-of-views in unknown dynamic environments. In: *Proceedings of ICRA 2014*, pp. 3576–3582, 2014.
- 5 S. Mitsch, K. Ghorbal, and A. Platzer: On Provably Safe Obstacle Avoidance for Autonomous Robotic Ground Vehicles. *Proceedings of Robotics: Science and Systems*, 2013.
- 6 M. Zhang, X. Zhang: Formally verifying navigation safety for ground robots. 2016 IEEE International Conference on Mechatronics and Automation.
- 7 C. Dabadie, S. Kaynama, C.J. Tomlin: A practical reachability-based collision avoidance algorithm for sampled-data systems: Application to ground robots. *Proceedings of IROS 2014*, pp. 4161–4168, 2014.

- 8 A. Bautin, L. Martinez-Gomez, T. Fraichard: Inevitable Collision States: A probabilistic perspective. In: Proceedings of ICRA 2010.
- 9 D. Althoff, M. Althoff, D. Wollherr, M. Buss: Probabilistic collision state checker for crowded environments. In: Proceedings of ICRA 2010.
- 10 M. Althoff, J.M. Dolan: Online Verification of Automated Road Vehicles Using Reachability Analysis. *IEEE Transactions on Robotics* (30), pp. 903–918, 2014.

3.7 Heteronomous Systems They are, Let's Face it.

Holger Hermanns (Universität des Saarlandes, DE)

License © Creative Commons BY 3.0 Unported license
© Holger Hermanns

Joint work of Raimund Dachselt, Holger Hermanns

Heteronomy refers to actions that are influenced by forces outside the individual. Autonomy is the opposite. For good reason, cars were originally called automobiles (and in some languages they still are). They give autonomy to people. So, what is an autonomous automobile?

In this talk I will argue that the currently acclaimed vision of fully autonomous systems is nothing but a trend towards heteronomy. This puts computer-assistance for heteronomous system design into a different perspective. I will elaborate on this perspective, and will discuss research challenges directly resulting from this.

3.8 GenoM3 Templates: from Middleware Independence to Formal Models Synthesis

Felix Ingrand (LAAS – Toulouse, FR)

License © Creative Commons BY 3.0 Unported license
© Felix Ingrand

Joint work of Mohammed Foughali, Félix Ingrand, Anthony Mallet

Main reference Mohammed Foughali, Félix Ingrand, Anthony Mallet, “GenoM3 Templates: from Middleware Independence to Formal Models Synthesis”, Rapport LAAS no. 17022. 2017.

URL <https://hal.laas.fr/hal-01457881>

GenoM is an approach to develop robotic software components, which can be controlled, and assembled to build complex applications. Its latest version, GenoM3, provides a template mechanism which is versatile enough to deploy components for different middleware without any change in the specification and user code. But this same template mechanism also enables us to automatically synthesize formal models (for two Validation and Verification frameworks) of the final components. We present and illustrate our approach on a real deployed example of a drone flight controller for which we prove offline real-time properties, and an outdoor robot for which we synthesize a controller to perform runtime verification.

This work was supported in part by the EU CPSE Labs project funded by the H2020 program under grant agreement No. 644400.

3.9 Synthesis of Shared Control Protocols with Provable Safety and Performance Guarantees

Nils Jansen (Univ. of Texas at Austin, US)

License © Creative Commons BY 3.0 Unported license
© Nils Jansen

Joint work of Nils Jansen, Murat Cubuktepe, Ufuk Topcu

Main reference Nils Jansen, Murat Cubuktepe, Ufuk Topcu, “Synthesis of Shared Control Protocols with Provable Safety and Performance Guarantees”, in Proc. of the 2017 American Control Conference (ACC’17), preprint available at arXiv:1610.08500v1 [cs.RO], 2017.

URL <https://arxiv.org/abs/1610.08500v1>

We formalize synthesis of shared control protocols with correctness guarantees for temporal logic specifications. More specifically, we introduce a modeling formalism in which both a human and an autonomy protocol can issue commands to a robot towards performing a certain task. These commands are blended into a joint input to the robot. The autonomy protocol is synthesized using an abstraction of possible human commands accounting for randomness in decisions caused by factors such as fatigue or incomprehensibility of the problem at hand. The synthesis is designed to ensure that the resulting robot behavior satisfies given safety and performance specifications, e.g., in temporal logic. Our solution is based on nonlinear programming and we address the inherent scalability issue by presenting alternative methods. We assess the feasibility and the scalability of the approach by an experimental evaluation.

3.10 A storm is Coming: A Modern Probabilistic Model Checker

Joost-Pieter Katoen (RWTH Aachen, DE)

License © Creative Commons BY 3.0 Unported license
© Joost-Pieter Katoen

Joint work of Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, Matthias Volk

Main reference Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, Matthias Volk, “A storm is Coming: A Modern Probabilistic Model Checker”, in Proc. of the 29th Int’l Conf. on Computer Aided Verification (CAV’17), LNCS, Vol. 10427, pp. 592-600, Springer, 2017.

URL http://dx.doi.org/10.1007/978-3-319-63390-9_31

URL <https://moves-rwth.github.io/storm/>

In the last five years, we have developed our in-house probabilistic model checker with the aim to have an easy-to-use platform for experimenting with new verification algorithms, richer probabilistic models, algorithmic improvements, different modeling formalism, various new features, and so forth. Although open-source probabilistic model checkers do exist, most are not flexible and modular enough to easily support this. Our efforts have led to a toolkit with mature building bricks with simple interfaces for possible extensions, and a modular set-up. It comprises about 100,000 lines of C++ code. The time has come to make this toolkit available to a wider audience: this paper presents storm.

Like its main competitors PRISM, MRMC, and iscasMC, storm relies on numerical and symbolic computations. It does not support discrete-event simulation, known as statistical model checking. The main characteristic features of storm are:

- it supports *various native input formats*: the PRISM input format, generalized stochastic Petri nets, dynamic fault trees, and conditioned probabilistic programs. This is not just providing another parser; state-space reduction and generation techniques as well as analysis algorithms are partly tailored to these modeling formalisms;

- in addition to Markov chains and MDPs, it supports *Markov automata*, a model containing probabilistic branching, non-determinism, and exponentially distributed delays;
- it can do *explicit state* and *fully symbolic* (BDD-based) model checking as well as a *mixture* of these modes;
- it has a *modular* set-up, enabling the easy exchange of different solvers and distinct decision diagram packages; its current release supports about 15 solvers, and the BDD packages CUDD [1] and multi-threaded Sylvan [2];
- it provides a *Python API* facilitating easy and rapid prototyping of other tools using the engines and algorithms in storm;
- it provides the following functionalities under one roof: the synthesis of counterexamples and permissive schedulers (both MILP- and SMT-based), game-based abstraction of infinite-state MDPs, efficient algorithms for conditional probabilities and rewards, and long-run averages on MDPs;
- its performance in terms of verification speed and memory footprint on the PRISM benchmark suite is mostly better compared to PRISM

Although many functionalities of PRISM are covered by storm, there are significant differences. storm does not support LTL model checking and does not support the PRISM features: probabilistic timed automata, multi-objective model checking, and an equivalent of PRISM's "hybrid" engine (a crossover between full MTBDD and storm's "hybrid" engine), a fully symbolic engine for continuous-time models, statistical model checking, and the analysis of stochastic games as in PRISM-GAMES.

References

- 1 Fabio Somenzi: CUDD: CU Decision Diagram package release 3.0.0, <http://vlsi.colorado.edu/~fabio/CUDD/> (2016)
- 2 Tom van Dijk and Jaco van de Pol: Sylvan: Multi-Core Decision Diagrams. *Tools and Algorithms for the Construction and Analysis of Systems – 21st International Conference (TACAS 2015)*. pp. 677–691, 2014.

3.11 High-Level Verifiable Robotics

Hadas Kress-Gazit (Cornell University – Ithaca, US)

License © Creative Commons BY 3.0 Unported license
© Hadas Kress-Gazit

Joint work of Gangyuan Jing, Tarik Tosun, Mark Yim, Scott Hamill, Jon DeCastro, Kai Weng Wong,
Hadas Kress-Gazit

URL <http://verifiablerobotics.com/>

In this talk I gave a quick overview of different projects in my lab in which we have used LTL synthesis and verification techniques to automatically create provably-correct robot controllers. I finished the talk with a provocative question on what is the role of formal verification and synthesis in the era of learning-based robotics.

3.12 (Learning to) Learn to Control

Jan Kretinsky (TU München, DE)

License © Creative Commons BY 3.0 Unported license
© Jan Kretinsky

Joint work of P. Ashok, T. Brazdil, K. Chatterjee, M. Chmelik, P. Daca, A. Fellner, V. Forejt, T. Henzinger, J. Kretinsky, M. Kwiatkowska, T. Meggendorfer, D. Parker, T. Petrov, V. Toman, M. Ujma

Main reference Tomás Brázdil, Krishnendu Chatterjee, Martin Chmelik, Vojtech Forejt, Jan Křetínský, Marta Z. Kwiatkowska, David Parker, Mateusz Ujma, “Verification of Markov Decision Processes Using Learning Algorithms”, in Proc. of the 12th International Symposium on Automated Technology for Verification and Analysis (ATVA 2014), LNCS, Vol. 8837, pp. 98–114, Springer, 2014.

URL http://dx.doi.org/10.1007/978-3-319-11936-6_8

On the one hand, formal verification methods provide hard guarantees on analysis results, but do not scale well and are often hard to use. On the other hand, machine learning comes with weak or no guarantees, but scales well and can provide more understandable solutions. In this talk, we show several examples how these approaches can be combined and the best of the two worlds achieved. We demonstrate this on controller synthesis [1,2] and controller representation [3] in the setting of Markov decision processes and comment on extensions to games [4].

References

- 1 Tomás Brázdil, Krishnendu Chatterjee, Martin Chmelik, Vojtech Forejt, Jan Křetínský, Marta Z. Kwiatkowska, David Parker, and Mateusz Ujma. *Verification of Markov decision processes using learning algorithms*. In *ATVA*, pages 98–114. Springer, 2014.
- 2 P. Ashok, K. Chatterjee, P. Daca, J. Křetínský, T. Meggendorfer. *Value Iteration for Long-run Average Reward in Markov Decision Processes*. Submitted.
- 3 T. Brazdil, K. Chatterjee, M. Chmelik, A. Fellner, J. Křetínský. *Counterexample Explanation by Learning Small Strategies in Markov Decision Processes*. In *CAV (1)*, pages 158–177. Springer, 2015.
- 4 T. Brazdil, K. Chatterjee, J. Křetínský, V. Toman. *Strategy Representation by Decision Trees in Reactive Synthesis*. Submitted.

3.13 Optimizing the Performance of Robots in Production Logistics Scenarios

Gerhard Lakemeyer (RWTH Aachen, DE)

License © Creative Commons BY 3.0 Unported license
© Gerhard Lakemeyer

Joint work of Tim Niemueller, Erika Abraham, Gerhard Lakemeyer

Main reference Frederik Zwilling, Tim Niemueller, Gerhard Lakemeyer, “Simulation for the RoboCup Logistics League with Real-World Environment Agency and Multi-level Abstraction”, in RoboCup 2014: RoboCup 2014: Robot World Cup XVIII, LNCS, Vol. 8992, pp. 220–232, Springer, 2015.

URL http://dx.doi.org/10.1007/978-3-319-18615-3_18

URL <https://www.fawkesrobotics.org/media/publications/llsf-sim-rc2014.pdf>

We consider the problem of optimizing the decision making of mobile robots managing the supply chain in a semi-structured factory setting. To keep things manageable and comprehensible we focus on a game-like environment provided by the Robocup Logistics League (RCLL). While the RCLL has been around for a number of years, there has been little progress in optimizing the performance of the robots. In order to make progress in a more principled way we recently joined forces with Erika Abraham’s group with the aim of applying SMT techniques to this problem. In this talk I will mainly focus on describing the

problems and challenges the RCLL raises and advertise the simulation-based variant of the RCLL as a possible benchmark to develop and test formal methods in robotics. I will also briefly outline our approach and the first steps we have taken to address the problem using SMT.

3.14 Artificial Intelligence Planning and Robotics and Autonomous Systems

Daniele Magazzeni (King's College London, GB)

License © Creative Commons BY 3.0 Unported license

© Daniele Magazzeni

Joint work of Daniele Magazzeni, Michael Cashmore, Maria Fox, Derek Long

Main reference Michael Cashmore, Maria Fox, Derek Long, Daniele Magazzeni, “A Compilation of the Full PDDL+ Language into SMT”, in Proc. of the 26th Int’l Conf. on Automated Planning and Scheduling (ICAPS 2016), pp. 79–87, AAAI Press, 2016.

URL <https://www.aaai.org/ocs/index.php/ICAPS/ICAPS16/paper/view/13101>

URL <http://kcl-planning.github.io/ROSPlan/>

AI Planning is about determining actions before doing them, anticipating the things that will need to be done and preparing for them. Planners use domain-independent heuristics to guide the search in huge state spaces, in order to find a plan that achieves the goal while satisfying numerical and temporal constraints and optimising a given metric. Planning for Robotics and Autonomous Systems requires rich models to capture complex dynamics as well as the uncertain and evolving environment, scalable planning techniques and robust methods of execution. PDDL+ is the formalism used in planning to describe hybrid systems, and allows the modelling of the differential equations governing the continuous behaviour of systems. This talk provides an overview of how PDDL+ can be used to model complex domains; presents a new PDDL+ planner based on SMT and the ROSPlan framework for planning with ROS; highlights some open challenges on the integration between task and motion planning.

3.15 Human-Robot Collaboration – Industrial Applications and Open Challenges

Björn Matthias (ABB AG Forschungszentrum – Ladenburg, DE)

License © Creative Commons BY 3.0 Unported license

© Björn Matthias

Main reference Björn Matthias, “Risk Assessment for Human-Robot Collaborative Applications”, Workshop on Physical Human-Robot Collaboration: Safety, Control, Learning and Applications at the IEEE/RSJ Int’l Conf. on Intelligent Robots and Systems (IROS 2015), 2015.

URL https://www.researchgate.net/publication/282778774_Risk_Assessment_for_Human-Robot_Collaborative_Applications

This contribution seeks to identify some important gaps in present methodology in the deployment of industrial robots in applications of human-robot collaboration (HRC). The drivers and enablers for deployment of industrial robots and of HRC in industrial practice are summarized. Safety of machinery, as called out for example in the European Machinery Directive, is introduced as a necessary boundary condition to fulfill in applications of industrial robots. A brief overview of the relevant standards to be followed is given. The basic types of collaborative operation of industrial robots are summarized, describing the specific protection

schemes for each. The challenges in planning and commissioning collaborative applications in industrial production are considered in more detail. This allows the identification of the present lack of methodology and tools to support the economical and safety-rated deployment of applications using HRC. The resulting research questions address these and other issues associated with the future of industrial robots and their applications.

3.16 A Competition on Formal Methods for Robotics

Vasumathi Raman (Zoox Inc., US)

License © Creative Commons BY 3.0 Unported license
© Vasumathi Raman

Joint work of Vasumathi Raman, Scott C. Livingston

Main reference Vasumathi Raman, “The 2016 Formal Methods for Robotics Challenge [Competitions]”, IEEE Robotics & Automation Magazine, Vol. 23(3), pp. 24–25, 2016.

URL <http://dx.doi.org/10.1109/MRA.2016.2587958>

URL <https://fmrchallenge.org/>

Formal methods refers broadly to techniques for the verification and automatic synthesis of transition systems that satisfy desirable properties exactly or within some statistical tolerance. Though historically developed for concurrent software, recent work has brought these methods to bear on motion planning in robotics. Challenges specific to robotics, such as uncertainty and real-time constraints, have motivated extensions to existing methods, as well as entirely novel treatments. However, when compared with other areas within robotics research, demonstrations of formal methods have been surprisingly small-scale. In this talk, I propose a robotics challenge that seeks to motivate advancement of the state of the art toward practical realization. The challenge is organized into three problem domains: arbitrary dimensional chains of integrators, traffic networks with Dubins cars, and factory cart clearing.

3.17 Specification: the Biggest Bottleneck in Formal Methods and Autonomy

Kristin Yvonne Rozier (Iowa State University, US)

License © Creative Commons BY 3.0 Unported license
© Kristin Yvonne Rozier

Main reference Kristin Yvonne Rozier, “Specification: The Biggest Bottleneck in Formal Methods and Autonomy”, in Proc. of the 8th Working Conf. on Verified Software: Theories, Tools, and Experiments (VSTTE’16), LNCS, Vol. 9971, pp. 1–19, Springer, 2016.

URL http://dx.doi.org/10.1007/978-3-319-48869-1_2

Advancement of autonomous systems stands on the shoulders of formal methods, which make possible the rigorous safety analysis autonomous systems require. An aircraft cannot operate autonomously unless it has design-time reasoning to ensure correct operation of the autopilot and runtime reasoning to ensure system health management, or the ability to detect and respond to off-nominal situations. Formal methods are highly dependent on the specifications over which they reason; there is no escaping the “garbage in, garbage out” reality. Specification is difficult, unglamorous, and arguably the biggest bottleneck facing verification and validation of autonomous systems.

We examine the outlook for formal specification, and highlight the on-going challenges of specification, from design-time to runtime. We exemplify these challenges for specifications in

Linear Temporal Logic (LTL) though the focus is not limited to that specification language. We pose challenge questions for specification that will shape both the future of formal methods, and our ability to more automatically verify and validate autonomous systems of greater variety and scale. We call for further research into LTL Genesis.

3.18 Development and Adoption of Model-Based Tools in Robotics

Christian Schlegel (Hochschule Ulm, DE)

License © Creative Commons BY 3.0 Unported license
© Christian Schlegel

Joint work of Christian Schlegel, Alex Lotz, Matthias Lutz, Dennis Stampfer

Main reference Dennis Stampfer, Alex Lotz, Matthias Lutz, Christian Schlegel, “The SmartMDSO Toolchain: An Integrated MDSO Workflow and Integrated Development Environment (IDE) for Robotics Software”, Special Issue on Domain-Specific Languages and Models in Robotics, *Journal of Software Engineering for Robotics (JOSER)*, 7(1), pp. 3–19, ISSN: 2035-3928, Open Journal Systems, 2016.

URL <https://joser.unibg.it/index.php?journal=joser&page=article&op=view&path%5B%5D=91>

We aim at making the development of better quality robot systems much less an effort by the means of model-driven tooling. This talk is about how to compose complex robotic software systems out of software building blocks and we advocate for moving from just source-code level integration towards model-driven composition with explicated properties. We consider the full stack from low level control over the task sequencing level up to the mission level. The challenge is to adhere to the principles of separation of concerns while at the same time, you need to package different concerns into structures such that these fit the views of e.g. component developers, system integrators and even the robots at run-time themselves. This talk underpins these ideas by the example of the matured model-driven SmartSoft/SmartMDSO approach and tooling. At various levels, there are hooks in the software engineering tools and in the run-time execution system where (formal) methods (e.g. for verification) could assist the different players in their different roles including the robot itself in better doing their jobs.

3.19 How Safe is Your Autonomous Robot? (A Tale of Courage, Passion, and Perspiration)

Armando Tacchella (University of Genova, IT)

License © Creative Commons BY 3.0 Unported license
© Armando Tacchella

In this work we consider the problem of checking safety in autonomous agents at the deliberative level. The interaction between the agent and the environment is modelled as a Markov decision process and it is assumed that control policies are learned using model-free approximate dynamic programming, best known as reinforcement learning (RL). Models and policies inferred during RL are combined to obtain discrete time Markov chains which can then be subject to verification and repair against probabilistic temporal logic properties. In particular, we consider repair both as an off-line strategy and an on-line technique to supplement execution monitoring with policy-mending capabilities. The approach is studied in the context of a standing-up task for a simple but nontrivial humanoid robot.

Participants

- Erika Abraham
RWTH Aachen, DE
- Jan Broenink
University of Twente, NL
- Simon Burton
Robert Bosch GmbH –
Stuttgart, DE
- Pablo Bustos
University of Extremadura –
Cáceres, ES
- Alessandro Cimatti
Bruno Kessler Foundation –
Trento, IT
- Manuel Alcino Cunha
INESC TEC and University of
Minho – Braga, PT
- Kerstin I. Eder
University of Bristol, GB
- Rüdiger Ehlers
Universität Bremen, DE
- Samira Farahani
MPI-SWS – Kaiserslautern, DE
- Stefano Ghidoni
University of Padova, IT
- Christoffer R. Heckman
University of Colorado –
Boulder, US
- Christian Heinzemann
Robert Bosch GmbH –
Stuttgart, DE
- Holger Hermanns
Universität des Saarlandes, DE
- Felix Ingrand
LAAS – Toulouse, FR
- Nils Jansen
Univ. of Texas at Austin, US
- Joost-Pieter Katoen
RWTH Aachen, DE
- Hadas Kress-Gazit
Cornell University – Ithaca, US
- Jan Kretinsky
TU München, DE
- Morteza Lahijanian
University of Oxford, GB
- Gerhard Lakemeyer
RWTH Aachen, DE
- Ratan Lal
Kansas State University –
Manhattan, US
- Martin Leucker
Universität Lübeck, DE
- Ingo Lütkebohle
Robert Bosch GmbH –
Stuttgart, DE
- Daniele Magazzeni
King's College London, GB
- Björn Matthias
ABB AG Forschungszentrum –
Ladenburg, DE
- Lorenzo Natale
Italian Institute of Technology –
Genova, IT
- Sandeep Neema
DARPA – Arlington, US
- Petter Nilsson
University of Michigan –
Ann Arbor, US
- Shashank Pathak
Technion – Haifa, IL
- Subramanian Ramamoorthy
University of Edinburgh, GB
- Vasumathi Raman
Zoox Inc., US
- Kristin Yvonne Rozier
Iowa State University, US
- Christian Schlegel
Hochschule Ulm, DE
- Maria Svorenova
University of Oxford, GB
- Armando Tacchella
University of Genova, IT
- Sebastian Wrede
Universität Bielefeld, DE

