Report from Dagstuhl Seminar 17201

Formal Synthesis of Cyber-Physical Systems

Edited by

Calin A. Belta[1], Rupak Majumdar[2], Majid Zamani[3], and Matthias Rungger[4]

1 **Boston University – Brookline, US,** `cbelta@bu.edu`
2 **MPI-SWS – Kaiserslautern, DE,** `rupak@mpi-sws.org`
3 **TU München, DE,** `zamani@tum.de`
4 **TU München, DE,** `matthias.rungger@tum.de`

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 17201 "Formal Synthesis of Cyber-Physical Systems." Formal synthesis is the application of algorithmic techniques based on automata and logic to the design of controllers for hybrid systems in which continuous components interact with discrete ones. The Dagstuhl seminar brought together researchers from control theory and from computer science to discuss the state-of-the-art and current challenges in the field.

Seminar May 14–19, 2017 – http://www.dagstuhl.de/17201
1998 ACM Subject Classification I.2.8 Problem Solving, Control Methods, and Search—Control theory; I.2.2 Automatic Programming; F.3.1 Specifying and Verifying and Reasoning about Programs
Keywords and phrases Cyber-physical systems, formal synthesis, reactive synthesis, discrete event systems, dynamical systems and control
Digital Object Identifier 10.4230/DagRep.7.5.84

1 Executive Summary

Calin A. Belta
Rupak Majumdar
Majid Zamani
Matthias Rungger

Cyber-Physical Systems (CPS) are complex systems resulting from intricate interaction of discrete computational devices with the continuous physical plants. Within CPS, embedded control software plays a significant role by monitoring and adjusting several physical variables, e.g. temperature, velocity, pressure, density, and so on, through feedback loops where physical processes interact with computational devices. Recent advances in computation, storage, and networking have made tremendous advances in hardware and system platforms for CPS. With this growing trend in computational devices, embedded control software is becoming more and more ubiquitous in many safety-critical applications including automotive, aerospace, transportation systems, critical infrastructure, energy, robotics, healthcare, and many other domains. Unfortunately, the design of embedded control software nowadays is still based on ad-hoc solutions resulting in brittle and error-prone software, and very high verification and validation costs. In order to detect and eliminate design flaws and inevitable software bugs, a large portion of the design budget is consumed with validation and verification efforts, which are often lengthy. On the other hand, by changing the emphasis from verification to synthesis, it is possible to synthesize correct-by-design embedded control software for CPS while providing formal guarantees of correctness and preventing the need for costly post facto verification.

In recent years, there has been a lot of progress in designing automatic and correct-by-construction techniques for controller synthesis for interacting discrete and continuous systems. These new techniques have combined techniques from continuous control theory as well as from computer science. The focus of this seminar was to provide a state-of-the-art of this nascent but important field, and to describe challenges and opportunities for synthesis techniques to transition to the real world. By the nature of the topic, the participation at the seminar was inter-disciplinary, and consisted of computer scientists and control theorists, both from academia and from industry. Instead of a sequence of presentations of individual research results, the seminar was organized as a sequence of open discussions on topics of common interest to the participants, such as techniques for scalable controller synthesis, identification of application domains and recent success stories, compositionality and system design, end-to-end arguments about systems, as well as education and outreach.

This seminar benefitted the control as well as computer science communities by bridging the gap between many complementary concepts studied in each community. A more detailed survey on the topics of the seminar is in preparation.

Outcomes of the seminar

The seminar focused on the challenges in the application of formal synthesis techniques for automatic, correct-by-construction synthesis of CPS. The seminar had a total of 45 participants with a mix of computer scientists and control theorists.

Sessions

The seminar was organized as a sequence of open discussions led by one or two moderators. Each session had a scribe to note down the discussion. The scribe notes were shared with all participants, who added their comments or filled in more information. The updated notes were used to prepare the session summaries (in the next Section).

The following sessions were organized:

1. Application domains, success stories, and obstacles to adoption
2. Fundamental algorithmic and scalability challenges in formal synthesis
3. Tools and infrastructure
4. Education and outreach
5. Data-driven and search-driven approaches to synthesis
6. Compositionality in synthesis
7. Optimality and completeness
8. Synthesis of distributed protocols from scenarios and requirements.
9. Specification languages
10. Robustness and resiliency
11. Explainability and user interaction
12. End-to-end correctness
13. Formal synthesis challenges in robotics
14. Cyber-Security of CPS

2 Table of Contents

3 Summary of Discussions

3.1 Application Domains I: Robotics and Automotive Control

Calin Belta and Jyotirmoy Deshmukh

The session was scribed by Paulo Tabuada.

Applications are important as they identify the theoretical, scalability, and computational challenges. Thus, we started the seminar with an identification of domains where formal synthesis could be profitably applied, or has had some preliminary success. The following application domains were discussed:

- Automotive
- Robotics
- Power Networks, Power Systems, Smart Grids
- Transportation (Smart Cities, Traffic Networks)
- Avionics
- Medical Devices
- Biology
- Manufacturing
- Smart buildings

Based on the expertise present at the seminar, we focused on the automotive, robotics, traffic network, and power systems. As a general feature, it was pointed out that an advantage of formal methods is that it is easy to change specifications and get new controllers, whereas a manual method may need the designer to start from scratch.

In the robotics domain, we discussed how formal approaches can be reconciled with standard practice of "build, test, iterate." We discussed synthesis problems specific the robotics infrastructure such as the use of higher-level programming abstractions and declarative specifications, and problems arising out of software upgrades. We discussed the abstraction level at which synthesis should be carried out. For "simple" dynamics, it is common to abstract the continuous dynamics and present the synthesis problem as planning in a grid world. However, for more complicated tasks such as grasping or bipedal walking, control for the continuous dynamics is important. A related question is how to encode background knowledge into the synthesis problem definition? For example, in many robotics applications we may need a large set of predicates, but LTL synthesis algorithms work better when we have a small number of predicates. Existing logics may not model reactive systems.

In the *automotive control domain*, we discussed where formal synthesis would fit in the existing industrial design process. Model-based Controller Design using Simulink is a success story since as this entails a refinement-based design methodology. Skepticism towards formal control design in the industry could be attributed to lack of knowledge on using formal methods properly. For instance, it is often not clear at what level of abstraction should formal synthesis be applied. Formal methods can already provide solutions for some of the small problems that are solved during the design process. Formal methods help in writing contracts between components and guide the solution. Modest sub-problems can be handled with formal methods.

In the automotive context there are several hardware platforms and protocols for communication between distributed electronic control units. These protocols are well-defined and there are many tools for timing analysis at the hardware level. However, they are decoupled from Simulink and other tools that can analyze the desired global behavior; e.g., existing timing analysis tools do not care about stability and we may be solving a problem harder than needed. There seems to be a gap, and there may be a need for tools that act as glue between existing tools at various levels of abstraction. To make problems harder, these tools are used by different teams, potentially in different organizations.

The NSF expeditions program on Computer-Aided Program Engineering (ExCAPE) indicates that solving the verification problem can help you solve the synthesis problem using counter-example guided synthesis. An optimistic view is that a similar approach could be used for CPS. A challenge for CPS is that specifications are not articulated well, so solving the specification problem is a prerequisite for solving the verification problem!

It would be beneficial to create a road-map for the use of formal methods in industry. Benchmarks are crucial to understand application scenarios for formal synthesis. Synthesis competitions for CPS may be a good exercise, following existing competitions that cover reachability analysis and falsification.

3.2 Application Domains II: Traffic Networks and Power Networks

Murat Arcak and Agung Julius

This session was scribed by Paulo Tabuada.

Recently, formal synthesis techniques have been applied to congestion control problems in traffic networks. Discrete-time piecewise affine (PWA) systems are widely used as models for traffic networks. There exist works on verification and synthesis for such systems. By exploiting the sparsity and the structure inherent in PWA models of traffic networks, abstractions can be constructed for fairly large systems (almost 60 dimensions). Congestion can be characterized as polyhedral sets and included as predicates in temporal logic formulas used as specifications.

Some concrete challenges in this domain were identified:

1. The connection between microscopic models (VSim) and the PWA density based models is not understood. There are other models at different layers. not clear what the formal connection is between these models.
2. Existing models do not explicitly capture V2V, V2I.
3. There is a need for probabilistic approaches to traffic networks. This would not improve scalability, but it will be more realistic.
4. Not clear how to integrate existing approaches focused on traffic light and meter control policies with autonomous vehicles.
5. Not clear how to integrate other control inputs, e.g., speed limits?
6. Existing approaches are not real time. If a road in closed or there is an accident, it is not clear how to adapt the controller

As with the previous application domains, there are several opportunities for formal synthesis in power networks, e.g., avoiding cascading failures in a power grid (Susuki et al, IEICE Trans. Fundamentals 2009). We discussed an approach based on nonlinear dynamics and circuit breakers. Changing the operating conditions of some generators might prevent the cascading failure. Metric temporal logic (MTL) was used as specification to prevent current peaks. This problems is a transient regime problem. It was identified that abstractions in this domain should capture the transient behavior.

These networks have many time-scales dynamics ranging from secs to minutes to hours-intervals. For formal verification there are different sources of adversaries. They are also very large. One way to address scalability would be to exploit possible symmetries in the network.

3.3 Fundamental algorithmic and scalability challenges in formal synthesis

Paulo Tabuada and Rupak Majumdar

This session was scribed by Calin Belta.

The synthesis problem has a long history both in computer science as well as in control theory. At an abstract level, the synthesis problem can be stated as follows: given a system and a specification, synthesize a controller that restricts the behavior of the system so as to satisfy the specification. When the system is given as a finite-state automaton and the specification given as a linear-time temporal logic formula, the problem can be solved algorithmically with time complexity that is doubly exponential in the length of the formula and polynomial in the size of the system. These results can also be used when the system is described by a differential equation. In this case, we first construct a finite-state abstraction of the differential equation and then use the results for finite-state automata. Unfortunately, existing abstraction techniques result in finite-state automata with a size that is exponential in the number of variables appearing in the differential equation. Given the high complexity of existing solutions for the synthesis problem, we discussed in this session several promising research directions to manage the high computational complexity, including incremental synthesis algorithms, how to leverage interactions with the user, robust strategies, and the use of machine learning techniques.

3.4 Tools and infrastructure

Ruediger Ehlers and Matthias Rungger

This session was scribed by Anne-Kathrin Schmuck

Different dimensions of existing tools for synthesizing controllers were discussed, ranging from the type of models and specifications they support, the type of guarantees they provide to formal correctness of the implementation. There was a general consensus that a set of exemplary instances of synthesis problems serving as benchmark problems are missing in the context of CPS. Additionally, benchmark examples should be advertised and made publicly available, e.g. at www.cps-vo.org. In order to foster the adaptation of formal methods tools in industry, using Simulink would be suitable candidate, as there are already automated techniques for translating Simulink to formal models. Finally, it was observed that software tools play an important role in education.

3.5 Teaching and Outreach

Calin Belta and Stephane Lafortune

This session was scribed by Manuel Mazo.

The participants discussed who to incorporate CPS concepts and synthesis techniques in undergraduate and graduate curricula in engineering and computer science. Existing courses and recently-published textbooks were identified. It was suggested to maintain a list of such courses and their syllabi on the CPS-VO. Industry needs should be considered in the development of such courses. An important resource on CPS education is the recent US NAE report titled "A 21st Century Cyber-Physical Systems Education". The participants also discussed outreach to industry. The availability of software tools that can be used by engineers in industry was identified as key to allow transfer of the technologies developed in the CPS synthesis community.

3.6 Data-driven synthesis and/or search-driven synthesis

Georgios Fainekos and Agung Julius

Model based search methods have proven their value in several practical control applications as well as in verification and falsification of CPS. For example, search based methods can be used to compute paths for autonomous vehicles, or to compute insulin infusion schedules, and to detect errors in Model Predictive Controllers (MPC) used in closed-loop artificial pancreas systems. System verification methods can also benefit by the utilization of machine learning techniques. In addition, there is a great opportunity to combine synthesis methods with machine learning techniques in order to control systems with unknown or uncertain models. Success has already been demonstrated in robotics applications where robots learn automata modeling the behavior of the environment and, then, using this information to plan and complete their missions. Along these lines, it may be possible to use data to learn system requirements, or to classify data, and then design controllers which satisfy the learned requirements. Another, approach would be to directly use already classified desirable and/or undesirable system behaviors along formal requirements for control synthesis.

A general future direction that came out of this workshop is to investigate what are the links between synthesis and learning, and how to combine the two. There are at least two ways one can think of combining the two:

1. Learning techniques for existing verification/synthesis problems: Here the emphasis is on using, adapting, or getting inspiration from, techniques from the broad machine learning field, in order to solve existing ("traditional") synthesis (and perhaps also verification?) problems.
2. New verification/synthesis problems, that adopt concepts from machine learning: Here, the idea is to come up with new and interesting problems (and then of course with solution methods for these problems). One example of a perhaps new such problem is the SSE problem (synthesis from spec and examples). Another example could be new variants of the Learning problem mentioned in the same section. New variants could be created by revisiting the notion of what it means for a learned system to "generalize well."

Two major challenges/open problems were identified during the session discussion:

- What are the right problems to solve? Learning in verification vs verification of learning.
- Where is the boundary of solvable and unsolvable computational problems when simultaneous learning and synthesis are concerned? For example, can we simultaneously learn the controller and the objective function from observations? How do we combine synthesis methods from high level specifications with reinforcement learning methods?

3.7 Compositionality in synthesis

Majid Zamani and Murat Arcak and Stavros Tripakis

This session was scribed by Matthias Rungger.

The main aim of this session is to investigate potential directions on compositional synthesis of controllers enforcing some global high-level specifications over interconnected cyber-physical systems.

In general, distributed reactive synthesis and supervisory control are undecidable but several approaches solve the problem for special cases. We discussed promising approaches to compositional synthesis of controllers, as well as their current limitations:

- There is some initial work on specification decomposition but it does not take into system composition structure. Specifically, can one exploit the structure and the sparsity of interconnection topology?
- There are promising results on discrete event systems on control design by abstracting components, composing abstractions, and abstracting them again, and then solving the resulting synthesis problem monolithically. It was suggested that the benchmarks from DES should be used for reactive synthesis.
- Synthesis from component libraries has been proposed for hardware and software; control theory needs to investigate this direction more.
- An important open direction is the unification of compositional rules in control theory (e.g. small-gain type reasoning) and computer science (e.g. assume-guarantee reasoning).
- An input-output interconnection may not be appropriate in those scenarios in which the dynamics of each component change due to interconnection; What is a good contract formalism for such systems?

3.8 Optimality and Correctness

Gunther Reissig and Antoine Girard

The first part of the session was dedicated to the synthesis of controllers optimizing quantitative objectives while enforcing some qualitative properties. Such optimal control problems appear naturally when a cost (e.g. energy consumption) is naturally associated with trajectories. Sometimes, it is possible to encode the qualitative requirements inside the quantitative objectives (as in linear quadratic regulation where stability is a consequence of finiteness of the quantitative objective). Synthesis of robust controllers can also be tackled through optimal control, e.g., by maximizing bounds on admissible disturbances or by optimizing the quantitative value of a temporal logic formula given by its quantitative semantics.

There exists a variety of possible cost functions for optimal control problems: discounted costs are often used in control theory, while mean payoff objectives (i.e., long term average cost) have been considered in computer science. These costs functions are actually complementary. Mean payoff objectives only optimize the asymptotic behavior of a system, while discounted costs focus mainly on the transient dynamics. While optimal control with discounted costs can be solved using classical dynamic programming, there are no known polynomial algorithms to solve mean payoff games. Actually, optimal strategies in mean payoff games may require infinite memory. Other types of costs functions may include positive and negative costs as in consumption or energy games.

The second part of the session focused on the problem of completeness. Asymptotic completeness in abstraction-based control holds when it is possible to approximate the true solution (e.g., maximal controller, value function) arbitrarily closely by refining sufficiently the abstraction. Some completeness results can be given for systems admitting bisimilar abstractions (e.g., controllable linear systems). For other systems, asymptotic completeness does not hold for all problems and is related to robustness and continuity of the value function with respect to the problem data. These conditions are hard to check a priori so a posteriori estimates of performance gaps and convergence rates may be useful. A promising direction is the development of synthesis approach that are anytime correct / asymptotically complete, allowing to obtain quickly a correct solution, which can be refined iteratively if needed and, which approaches the maximal / optimal solution asymptotically.

3.9 Robustness and resilience

Hadas Kress-Gazit and Pavithra Prabhakar

Robust synthesis corresponds to the synthesis of a controller that along with the model satisfies a specification, even in the presence of perturbations in the model, controller and specification. Such perturbations capture uncertainty/temporary environment violations, measurement/actuation errors, and robust semantics of properties described in temporal lgoics such as LTL/STL. The discussion consisted of different versions of robust synthesis problem that captured different relations between perturbations in the model, controller, and specification. In addition, metrics were identified as important in formulating the robust synthesis problem, and different metrics were discussed.

3.10 Explainability

Hadas Kress-Gazit

The usual outcome of formal synthesis is a synthesized controller. However, there are other artifacts that can be automatically generated that will make impact on the user. In this session, we discussed the different artifacts that can be generated to debug specifications, reason about controllers and facilitate the composition of the controller in a system, focusing on:

- Can formal synthesis be leveraged for explainability?
- What artifacts can be generated through the synthesis process to aid the user in understanding/debugging/validating the controller and the specification?

What can a tool return when the specification is unrealizable? One can return counterstrategies to present the reasons for unrealizability. This can happen through understandable presentation of the counterstrategy or suggestions of assumptions that would make the specification realizable or example executions for the environment that make the system fail. Tools can also return minimal specification revisions to make it realizable. Dually, tools can identify vacuous parts of specifications. There is work in the formal methods community on detecting vacuous specifications (in hardware verification) that can be leveraged to identify vacuity in specifications used to synthesize controllers. Additional possibilities are sensitivity analysis of counterexamples for identifying fragile areas in the control, grouping together counterexamples, monitors for runtime assumption violation from the specification,

We discussed how to produce understandable witnesses that "prove" the black-box controller obtained as the output of the synthesis tool. Witnesses could be "incimplete but understandable": for example, through decision trees, skeletons, simulations, or other "understandable" representations of the controller.

3.11 End-to-end correctness

Samarjit Chakraborty and Sanjoy Mitter

The goal of this session is to discuss how to integrate different layers of the design stack in embedded control systems in order to facilitate end-to-end validation and synthesis. Current workflows are based on independent design and analysis of these layers, often involving assumptions at the higher modeling and algorithms development layers that do not hold true at the lower implementation layers. This results in significant ex post facto integration, testing and debugging efforts and pose major roadblocks in the validation, certification and automated synthesis of such systems. Partitioning the entire design problem into independent layers was necessary for conquering design complexity and has led to significant progress in general-purpose computing, where no assumptions on the applications were anyway possible. Hence, the focus —at all fronts such as compilers, operating systems and computer architecture— has always been on optimizing a general set of performance metrics such as (average case) execution time, memory footprint, communication speed and power consumption. However, when it comes to control algorithms, first, these secondary metrics do not serve the purpose adequately since they do not necessarily optimize the primary metrics such as stability, settling time or peak overshoot. Second, there is a rich variety of design methods and proof techniques available within control theory which are now restricted to the modeling and algorithms design layer and do not permeate into the implementation layers below. This brings up the following broad research questions/directions:

- How to develop a cross-layer framework for end-to-end design and validation of embedded control systems?

- How to incorporate implementation level details directly in the control algorithms design phase so that different implementation-level resource tradeoffs can be accounted for during controller design?
- Control/architecture co-synthesis? As a starting point, the "architecture" could be a scheduler or a protocol.
- Synthesis on the fly: Control algorithm (or at least its parameters), along with its implementation level parameters to change on the fly, in response to either (a) changes in the plant, or (b) changes in resource availability.
- Proof transformation techniques: Example, develop compilation techniques that will transform the proofs along with the models, so as to automatically generate a certificate of correctness along with the implementation.
- Develop tool chains.

3.12 Application Domains III: Formal synthesis challenges in robotics

Hadas Kress-Gazit

Since the discussion on application domains had not converged, there was a separate session on formal synthesis challenges in robotics. The main points discussed in this seesion were hierarchical approaches to formal synthesis, different abstraction formalisms (including open questions, such as an appropriate formalism for reconfigurable robots), and synthesis approaches to sensor/actuaror failures on-the-fly. It was suggested that synthesis can have an impact in robotics in providing explanable, robust, and formal plans which can be automatically integrated into the software infrastructure. We discussed the distinction between planning and synthesis, and the additional benefits synthesis can bring, e.g., by explicitly modeling reactivity and environment assumptions, by providing runtime monitors about environment assumptions, and guarantees w.r.t. a model.

3.13 Formal synthesis challenges in Cybersecurity

Paulo Tabuada and Ashutosh Trivedi

Security and privacy concerns are increasingly relevant in the CPS context, where a security attack can very easily have catastrophic consequences. Security attacks on control systems are no longer a theoretical possibility: there have been much publicized incidents in the recent past (e.g., Stuxnet). We discussed synthesis approaches and research challenges for cyber-security, including:

1. Synthesizing for side-channel (timing or space) vulnerabilities.
2. How do we measure tradeoff between security and the effort for mitigating the leakage-risk? In CPS settings, this can often be modeled as making the system unobservable. In Discrete Event Systems, security enforcement can be achieved through the notion of system (opacity), which involves hiding info about current state or initial state.

3. Security Enforcement method in control theory by checking if the system is invertible or by designing a controller to destroy surjectivity of the map.
4. Unique challenges for CPS security include the unacceptable overhead of standard cryptographic primitives, novel attack-surfaces (attacks on analog sensors, side channels in physical processes, algorithms interacting with physical environment, resource-usage info not in code)

Participants

- Erika Abraham
RWTH Aachen, DE
- Murat Arcak
University of California – Berkeley, US
- Calin A. Belta
Boston University – Brookline, US
- Ivana Cerná
Masaryk University – Brno, CZ
- Samarjit Chakraborty
TU München, DE
- Chih-Hong Cheng
fortiss GmbH – München, DE
- Jyotirmoy Deshmukh
Toyota Motors North America, US
- Rüdiger Ehlers
Universität Bremen, DE
- Martin Fabian
Chalmers UT – Göteborg, SE
- Georgios Fainekos
Arizona State University – Tempe, US
- Antoine Girard
CNRS – Gif sur Yvette, FR
- Anak Agung Julius
Rensselaer Polytechnic Institute, US
- Roozbeh Kianfar
Zenuity, SE
- Hadas Kress-Gazit
Cornell University – Ithaca, US
- Jan Kretinsky
TU München, DE
- Stéphane Lafortune
University of Michigan – Ann Arbor, US
- Jun Liu
University of Waterloo, CA
- Rupak Majumdar
MPI-SWS – Kaiserslautern, DE
- Manuel Mazo
TU Delft, NL
- Sanjoy K. Mitter
MIT – Cambridge, US
- Sahar Mohajerani
Chalmers UT – Göteborg, SE
- Necmiye Ozay
University of Michigan – Ann Arbor, US
- Ivan Papusha
Univ. of Texas at Austin, US
- Nir Piterman
University of Leicester, GB
- Pavithra Prabhakar
Kansas State University – Manhattan, US
- Gunther Reißig
Universität der Bundeswehr – München, DE
- Harald Ruess
fortiss GmbH – München, DE
- Matthias Rungger
TU München, DE
- Indranil Saha
Indian Institute of Technology Kanpur, IN
- Anne-Kathrin Schmuck
MPI-SWS – Kaiserslautern, DE
- Sadegh Soudjani
MPI-SWS – Kaiserslautern, DE
- Paulo Tabuada
University of California at Los Angeles, US
- Wolfgang Thomas
RWTH Aachen, DE
- Hazem Torfah
Universität des Saarlandes, DE
- Stavros Tripakis
University of California – Berkeley, US
- Ashutosh Trivedi
University of Colorado – Boulder, US
- Jana Tumova
KTH Royal Institute of Technology – Stockholm, SE
- Marcell Vazquez-Chanlatte
University of California – Berkeley, US
- Majid Zamani
TU München, DE