

# The Futures of Reactive Synthesis

Nathanaël Fijalkow<sup>\*1</sup>, Bernd Finkbeiner<sup>\*2</sup>, Guillermo A. Pérez<sup>\*3</sup>,  
Elizabeth Polgreen<sup>\*4</sup>, and Rémi Morvan<sup>†5</sup>

1 CNRS – Talence, FR. [nathanael.fijalkow@gmail.com](mailto:nathanael.fijalkow@gmail.com)

2 CISA – Saarbrücken, DE. [finkbeiner@react.uni-saarland.de](mailto:finkbeiner@react.uni-saarland.de)

3 University of Antwerp, BE. [guillermoalberto.perez@uantwerpen.be](mailto:guillermoalberto.perez@uantwerpen.be)

4 University of Edinburgh, GB. [elizabeth.polgreen@ed.ac.uk](mailto:elizabeth.polgreen@ed.ac.uk)

5 University of Bordeaux, FR. [remi.morvan@u-bordeaux.fr](mailto:remi.morvan@u-bordeaux.fr)

---

## Abstract

The Dagstuhl Seminar 23391 “The Futures of Reactive Synthesis” held in September 2023 was meant to gather neighbouring communities on a joint goal: Reactive Synthesis. We identified five trends: neural-symbolic computation, template-based solving for constraint programming, symbolic algorithms, syntax-guided synthesis, and model learning; and the objective was to discuss the potential futures of the field.

**Seminar** September 24–29, 2023 – <https://www.dagstuhl.de/23391>

**2012 ACM Subject Classification** Computing methodologies → Artificial intelligence; Theory of computation → Formal languages and automata theory; Computing methodologies → Parallel programming languages

**Keywords and phrases** program synthesis, program verification, reactive synthesis, temporal synthesis

**Digital Object Identifier** 10.4230/DagRep.13.9.166

## 1 Executive Summary

*Nathanaël Fijalkow*

*Bernd Finkbeiner*

*Guillermo A. Pérez*

*Elizabeth Polgreen*

**License**  Creative Commons BY 4.0 International license

© Nathanaël Fijalkow, Bernd Finkbeiner, Guillermo A. Pérez, and Elizabeth Polgreen

This report documents the program and the outcomes of Dagstuhl Seminar 23391 “The Futures of Reactive Synthesis”.

The seminar was meant to gather neighbouring communities on a joint goal: Reactive Synthesis. We identified five trends: neural-symbolic computation, template-based solving for constraint programming, symbolic algorithms, syntax-guided synthesis, and model learning. They were represented by different participants, and in particular by four invited speakers. We had three female invited speakers and one male invited speaker; all delivered very insightful and forward-thinking talks:

- Anne-Kathrin Schmuck
- Armando Solar-Lezama
- Ruzica Piskac
- Dana Fisman

---

\* Editor / Organizer

† Editorial Assistant / Collector



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 4.0 International license

The Futures of Reactive Synthesis, *Dagstuhl Reports*, Vol. 13, Issue 9, pp. 166–184

Editors: Nathanaël Fijalkow, Bernd Finkbeiner, Guillermo A. Pérez, and Elizabeth Polgreen



Dagstuhl Reports

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

We introduced a number of mechanisms to encourage discussions and the exchange of ideas: an open problem session, long Q&A sessions after each invited talk, and most importantly working sessions. The working sessions were proposed by participants (who volunteered in advance, after a call by email to all participants). The proposer would have a few minutes to introduce the topic they would like to discuss. Each session included 3 or 4 different topics, discussed in parallel in smaller groups. In each case, we had (by some miracle!) a fair division of all participants into the 3 or 4 topics, and we had very good feedback that many working sessions resulted in very fruitful and insightful discussions. We had “progress report sessions” where the leaders of the working sessions gave a 5 or 10-min summary of the discussions.

We also had 9 contributed talks from participants, responding to an open call. They were 20 minutes each, and greatly contributed to getting all participants involved and for representing all trends and recent advances in the field.

We as organizers had very good feedback about the organization of the week: the rather light schedule gave enough time for people to discuss, and the different talks and organized sessions gave enough ways to get to know new people and topics. The seminar included a number of junior participants, who got to meet experts in the field. The mix of tools and theory topics covered during the seminar gives us hope that it will yield results both in the short and long term.

## 2 Table of Contents

### Executive Summary

*Nathanaël Fijalkow, Bernd Finkbeiner, Guillermo A. Pérez, and Elizabeth Polgreen* 166

### Overview of Talks

Solving Infinite-State Games via Acceleration	
<i>Rayna Dimitrova</i> . . . . .	170
Fixpoint Equations for Synthesis – Towards a Renewed Interest	
<i>Rüdiger Ehlers</i> . . . . .	170
Synthesis from LTL specifications and examples	
<i>Emmanuel Filiot</i> . . . . .	171
A primer on reactive synthesis	
<i>Bernd Finkbeiner</i> . . . . .	172
$\omega$ -Automata Learning	
<i>Dana Fisman</i> . . . . .	172
Compositional Synthesis with Hyperproperties	
<i>Niklas Metzger</i> . . . . .	172
Ups and downs of distributed synthesis	
<i>Anca Muscholl</i> . . . . .	173
Making New Friends in Software Synthesis	
<i>Ruzica Piskac</i> . . . . .	173
Synthesis Modulo Oracles	
<i>Elizabeth Polgreen</i> . . . . .	174
A primer on SYNTCOMP	
<i>Guillermo A. Pérez</i> . . . . .	175
Reactive Synthesis as a Programming Language Paradigm	
<i>Mark Santolucito</i> . . . . .	175
Deep Learning for Reactive Synthesis	
<i>Frederik Schmitt</i> . . . . .	176
The power of feedback	
<i>Anne-Kathrin Schmuck</i> . . . . .	176
Constraint-based synthesis	
<i>Armando Solar-Lezama</i> . . . . .	177
Synthesizing Pareto-optimal Interpretations for Black-box Models	
<i>Hazem Torfah</i> . . . . .	177

### Working groups

Quantitative Specification	
<i>Shaul Almagor</i> . . . . .	178
A programmatic approach for reactive synthesis	
<i>Nathanaël Fijalkow</i> . . . . .	178

Minimization of deterministic parity automata	
<i>Antonio Casares</i> . . . . .	178
Minimization of deterministic (co)Büchi automata	
<i>Rémi Morvan</i> . . . . .	179
Positionality and memory	
<i>Pierre Ohlmann</i> . . . . .	179
Graph neural networks and reactive synthesis	
<i>Guillermo A. Pérez</i> . . . . .	179
SYNTCOMP benchmarking	
<i>Guillermo A. Pérez</i> . . . . .	180
IPASIR-UP: User Propagators for CDCL	
<i>Andre Schidler</i> . . . . .	181
Reactive Synthesis Beyond the Booleans	
<i>César Sánchez</i> . . . . .	182
Reactive synthesis of Linear Temporal Logic on finite traces	
<i>Shufang Zhu</i> . . . . .	182
<b>Participants</b> . . . . .	184

### 3 Overview of Talks

#### 3.1 Solving Infinite-State Games via Acceleration

*Rayna Dimitrova (CISPA – Saarbrücken, DE)*

**License** © Creative Commons BY 4.0 International license  
© Rayna Dimitrova

**Joint work of** Philippe Heim, Rayna Dimitrova

**Main reference** Philippe Heim, Rayna Dimitrova: “Solving Infinite-State Games via Acceleration”, CoRR, Vol. abs/2305.16118, 2023.

**URL** <https://doi.org/10.48550/ARXIV.2305.16118>

Two-player graph games have found numerous applications, most notably in the synthesis of reactive systems from temporal specifications, but also in verification. The relevance of infinite-state systems in these areas has lead to significant attention towards developing techniques for solving infinite-state games. In this talk I will present novel symbolic semi-algorithms for solving infinite-state games with omega-regular winning conditions. The novelty of our approach lies in the introduction of an acceleration technique that enhances fixpoint-based game-solving methods and helps to avoid divergence. Classical fixpoint-based algorithms, when applied to infinite-state games, are bound to diverge in many cases, since they iteratively compute the set of states from which one player has a winning strategy. Our proposed approach can lead to convergence in cases where existing algorithms require an infinite number of iterations. This is achieved by acceleration: computing an infinite set of states from which a simpler sub-strategy can be iterated an unbounded number of times in order to win the game. Ours is the first method for solving infinite-state games to employ acceleration. Thanks to this, it is able to outperform state-of-the-art techniques on a range of benchmarks, as evidenced by our evaluation of a prototype implementation.

#### 3.2 Fixpoint Equations for Synthesis – Towards a Renewed Interest

*Rüdiger Ehlers (TU Clausthal, DE)*

**License** © Creative Commons BY 4.0 International license  
© Rüdiger Ehlers

**Joint work of** Ayrat Khalimov, Rüdiger Ehlers

Reactive synthesis is traditionally reduced to solving a game between two players, where the game graph and the winning condition for one of the players in the game encodes the specification and the known information about the environment of the system to be synthesized. In this context, it is customary to encode all the available information into the game graph itself, so that a simple parity, Rabin, or Streett winning condition (among others) remains to be applied to the game graph. This allows to use game solving algorithms optimized for the respective winning condition off-the-shelf to solve the synthesis games.

Combining complex game graphs and relatively simple winning conditions is however not the only way to approach game-based reactive synthesis. We can alternatively distil only a part of the available information (such as the known information about the environment of the system to be synthesized and some simple specification parts) into the game graph, and encode the more complicated specification parts into the winning condition. In practice, this means computing a fixpoint equation that is evaluated over the game graph, where the result of evaluating the equation is the set of game positions from which the specification is realizable. This approach is followed in the Generalized Reactivity(1) Synthesis algorithm by

Piterman, Pnueli, and Sa’ar, which exploits the fact that for the specifications supported for it, there is a simple fixpoint formula template that can be instantiated for any specification of the supported specification class. In this way, the fixpoint equation can be evaluated symbolically if the game graph is easy to encode symbolically, which helped with scaling Generalized Reactivity(1) synthesis to a good number of applications in robotics and control.

In this talk, we discuss one commonly known, one recent, and one new result on computing fixpoint equations encoding complex specifications that go beyond Generalized Reactivity(1) synthesis. All the discussed results are applicable to symbolically represented game graphs. Apart from reviewing how to build such fixpoint formulas from deterministic parity automata for a given specification to be enforced in a game graph, we discuss a recent result by Hausmann, Lehaut, and Piterman and give a summary of our own results on translating a polynomial-time minimizable chain-of-co-Büchi-automata representation of a given omega-regular specification to a fixpoint equation. We provide some experimental results and employ them to argue for establishing a branch of reactive synthesis research that aims at computing efficient to evaluate fixpoint equations over symbolic game graphs. Focusing on such fixpoint equations has three advantages: Firstly, even at the current early state of research, the first approaches are already faster than previous full-LTL synthesis tools on specifications that decompose quite naturally into a game graph and a complex specification. Then, a compilation of a reactive synthesis problem to a game graph plus a fixpoint formula is a concise starting point for performing symbolic reasoning beyond the use of BDDs. Finally, fixpoint equations encoding complex specifications for synthesis would be useful for tackling the synthesis problem in implicitly represented infinite state spaces of games, which may be interesting for control and robotics applications.

This talk led to a working session.

### 3.3 Synthesis from LTL specifications and examples

*Emmanuel Filiot (UL – Brussels, BE)*

**License** © Creative Commons BY 4.0 International license

© Emmanuel Filiot

**Joint work of** Emmanuel Filiot, Mrudula Balachander, Jean-François Raskin

We study a variant of the problem of synthesizing Mealy machines that enforce LTL specifications against all possible behaviours of the environment including hostile ones. In the variant studied here, the user provides the high level LTL specification  $S$  of the system to design, and a set  $E$  of examples of executions that the solution must produce. The examples are used to guide the synthesis procedure, and are generalized as much as possible, while preserving realizability of the specification. This talk presents some approach to this problem based on a combination of RPNI automata learning and antichain-based LTL synthesis methods.

#### References

- 1 Mrudula Balachander, Emmanuel Filiot and Jean-François Raskin. *LTL Reactive Synthesis with a Few Hints*. Tools and Algorithms for the Construction and Analysis of Systems (TACAS), 2023.

### 3.4 A primer on reactive synthesis


Bernd Finkbeiner (CISPA – Saarbrücken, DE)

License  Creative Commons BY 4.0 International license  
© Bernd Finkbeiner

The synthesis of reactive systems has been actively investigated since the inception of the problem by Alonzo Church more than sixty years ago. This talk gives an overview of the main results of the area and an outlook on potential future directions to be discussed in the seminar, including neural-symbolic computation and, more generally, machine learning techniques, template-based solving in the context of constraint programming, active learning algorithms, and connections to program synthesis and in particular Syntax Guided Synthesis.

### 3.5 $\omega$ -Automata Learning

Dana Fisman (Ben Gurion University – Beer Sheva, IL)

License  Creative Commons BY 4.0 International license  
© Dana Fisman

Joint work of Dana Angluin, Timos Antonopoulos, Udi Boker, Dana Fisman, Nevin George, Yaara Shoval

This talk surveys the results on learning automata models for regular languages of infinite words. It discusses several positive and negative results across different learning paradigms. The positive results are mostly for automata models that are less common, in particular families of DFAs (FDFAs), strongly unambiguous Büchi automata (SUBAs) and mod-2 multiplicity automata (M2MA). These models have other good qualities, in particular the complexity of the boolean operations (intersection, union, complementation) and decision problems (emptiness, inclusion, equivalence) are good compared to the common omega-automata types. It is thus worth exploring whether they can also be usable for model checking and synthesis of reactive systems.

#### References

- 1 Dana Angluin, Dana Fisman *Learning regular omega languages*. Theor. Comput. Sci. 650: 57-72 (2016)
- 2 Dana Angluin, Udi Boker, Dana Fisman *Families of DFAs as Acceptors of  $\omega$ -Regular Languages*. Log. Methods Comput. Sci. 14(1) (2018)
- 3 Dana Angluin, Timos Antonopoulos, Dana Fisman *Strongly Unambiguous Büchi Automata Are Polynomially Predictable With Membership Queries*. CSL 2020: 8:1-8:17
- 4 Dana Angluin, Timos Antonopoulos, Dana Fisman, Nevin George *Representing Regular Languages of Infinite Words Using Mod 2 Multiplicity Automata*. FoSSaCS 2022: 1-20
- 5 *Constructing Concise Characteristic Samples for Acceptors of Omega Regular Languages*.

### 3.6 Compositional Synthesis with Hyperproperties

Niklas Metzger (CISPA – Saarbrücken, DE)

License  Creative Commons BY 4.0 International license  
© Niklas Metzger

The distributed synthesis problem is to translate a logical specification of a distributed system into an implementation that is guaranteed to satisfy the specification. What makes the synthesis of distributed systems far more challenging than standard reactive synthesis

is that each component only has partial knowledge of the global system state. Currently, there are no scalable algorithms for distributed synthesis. The challenge is to devise a compositional synthesis method, i.e., a method that constructs one component at a time. The fundamental difficulty is that the components often need to act upon information that is available only in another component. However, we do not know how that component encodes the information before we know its implementation; seemingly, it is impossible to build one component without knowing the implementation of the other. In this talk, I will present a compositional synthesis method based on the key idea of characterizing the necessary flow of information between the components as a hyperproperty. We introduce information flow assumptions, which are requirements that are necessary in order to realize a particular component. By formulating these assumptions as hyperproperties, we avoid referring to any particular encoding of the information. We develop methods that automatically derive information flow assumptions from the specification and a technique for the automatic synthesis of component implementations based on information flow assumptions. Together, these methods provide a compositional approach to the synthesis of distributed systems.

### 3.7 Ups and downs of distributed synthesis

*Anca Muscholl (University of Bordeaux, FR)*

**License** © Creative Commons BY 4.0 International license  
© Anca Muscholl

**Joint work of** Hugo Gimbert, Corto Mascle, Anca Muscholl, Igor Walukiewicz

**Main reference** Hugo Gimbert, Corto Mascle, Anca Muscholl, Igor Walukiewicz: “Distributed controller synthesis for deadlock avoidance”, CoRR, Vol. abs/2204.12409, 2022.

**URL** <https://doi.org/10.48550/ARXIV.2204.12409>

The talk gave an overview of several approaches to distributed reactive synthesis: Pnueli & Rosner model, controller synthesis for Zielonka automata and controller synthesis for lock-sharing systems. While partial information is a direct source of undecidability in the Pnueli & Rosner model, full causal information does not guarantee decidability either (cf. Gimbert 2022). Loose synchronization as in lock-sharing systems allows to recover decidability of controller synthesis at reasonable cost.

### 3.8 Making New Friends in Software Synthesis

*Ruzica Piskac (Yale University – New Haven, US)*

**License** © Creative Commons BY 4.0 International license  
© Ruzica Piskac

**Joint work of** Wonhyuk Choi, Bernd Finkbeiner, Ruzica Piskac, Mark Santolucito, Felix Klein

**Main reference** Wonhyuk Choi, Bernd Finkbeiner, Ruzica Piskac, Mark Santolucito: “Can reactive synthesis and syntax-guided synthesis be friends?”, in Proc. of the PLDI ’22: 43rd ACM SIGPLAN International Conference on Programming Language Design and Implementation, San Diego, CA, USA, June 13 – 17, 2022, pp. 229–243, ACM, 2022.

**URL** <https://doi.org/10.1145/3519939.3523429>

While reactive synthesis and syntax-guided synthesis (SyGuS) have seen enormous progress in recent years, combining the two approaches has remained a challenge. To overcome this obstacle, we introduced Temporal Stream Logic (TSL) [1], a new temporal logic that separates control and data. We developed a CEGAR-like synthesis approach for the construction of



implementations that are guaranteed to satisfy a TSL specification for all possible instantiations of the data processing functions. However, specifications often involve interpreted functions: for example, arithmetic functions or string manipulations. We extended TSL to Temporal Stream Logic modulo theories (TSL-MT) [2], a framework that unites the two approaches to synthesize a single program. In our approach, reactive synthesis and SyGuS collaborate in the synthesis process, and generate executable code that implements both reactive and data-level properties. We demonstrate the applicability of our approach over a set of real-world benchmarks [3].

## References

- 1 Bernd Finkbeiner, Felix Klein, Ruzica Piskac, Mark Santolucito, *Temporal Stream Logic: Synthesis Beyond the Booleans*. CAV (1) 2019: 609-629
- 2 Bernd Finkbeiner, Philippe Heim, Noemi Passing, *Temporal Stream Logic modulo Theories*. FoSSaCS 2022: 325-346
- 3 Wonhyuk Choi, Bernd Finkbeiner, Ruzica Piskac, Mark Santolucito, *Can reactive synthesis and syntax-guided synthesis be friends?*. PLDI 2022: 229-243

## 3.9 Synthesis Modulo Oracles

*Elizabeth Polgreen (University of Edinburgh, GB)*

**License** © Creative Commons BY 4.0 International license  
© Elizabeth Polgreen

**Joint work of** Elizabeth Polgreen, Andrew Reynolds, Sanjit A. Seshia  
**Main reference** Elizabeth Polgreen, Andrew Reynolds, Sanjit A. Seshia: “Satisfiability and Synthesis Modulo Oracles”, in Proc. of the Verification, Model Checking, and Abstract Interpretation – 23rd International Conference, VMCAI 2022, Philadelphia, PA, USA, January 16-18, 2022, Proceedings, Lecture Notes in Computer Science, Vol. 13182, pp. 263–284, Springer, 2022.  
**URL** [https://doi.org/10.1007/978-3-030-94583-1\\_13](https://doi.org/10.1007/978-3-030-94583-1_13)

In classic program synthesis algorithms, such as counterexample-guided inductive synthesis (CEGIS), the algorithms alternate between a synthesis phase and an oracle (verification) phase. Many (most) synthesis algorithms use a white-box oracle based on satisfiability modulo theory (SMT) solvers to provide counterexamples. But what if a white-box oracle is either not available or not easy to work with?

In this talk, I will present a framework for solving a general class of oracle-guided synthesis problems which we term synthesis modulo oracles (SyMO). In this setting, oracles are black boxes with a query-response interface defined by the synthesis problem. This allows us to lift synthesis to domains where using an SMT solver as a verifier is not practical.

### 3.10 A primer on SYNTCOMP

*Guillermo A. Pérez (University of Antwerp, BE)*

**License** © Creative Commons BY 4.0 International license  
© Guillermo A. Pérez

**Joint work of** Swen Jacobs, Guillermo A. Pérez

**Main reference** Swen Jacobs, Guillermo A. Pérez, Remco Abraham, Véronique Bruyère, Michaël Cadilhac, Maximilien Colange, Charly Delfosse, Tom van Dijk, Alexandre Duret-Lutz, Peter Faymonville, Bernd Finkbeiner, Ayrat Khalimov, Felix Klein, Michael Luttenberger, Klara J. Meyer, Thibaud Michaud, Adrien Pommellet, Florian Renkin, Philipp Schlehuber-Caissier, Mouhammad Sakr, Salomon Sickert, Gaëtan Staquet, Clément Tamines, Leander Tentrup, Adam Walker: “The Reactive Synthesis Competition (SYNTCOMP): 2018-2021”, CoRR, Vol. abs/2206.00251, 2022.

**URL** <https://doi.org/10.48550/ARXIV.2206.00251>

The Reactive Synthesis Competition (SYNTCOMP) is a competition for reactive synthesis tools. The competition’s goal is to collect benchmarks in a publicly available library and foster research in new tools for automatic synthesis of systems. SYNTCOMP is organized annually (since 2014) as a satellite event of CAV.

In this talk, the status of the competition (in particular the state of the benchmarks) and its evolution through the last couple of years is presented.

### 3.11 Reactive Synthesis as a Programming Language Paradigm

*Mark Santolucito (Barnard College, Columbia University – New York, US)*

**License** © Creative Commons BY 4.0 International license  
© Mark Santolucito

**URL** <https://barnard-pl-labs.github.io/CYOA-TSL>

There has been an explosion of interest in the use of LLMs to generate code in recent years, complimenting a long history of formal methods-driven program synthesis. However, code generation remains a largely all-or-nothing problem – either users can take advantage of the flexibility and adaptivity of LLMs and generate code that might not be correct, or they can rely on more rigid program synthesis tools which are guaranteed to generate correct results, but are limited in their generative grammars. In this talk, we propose a strategy for the combination of these techniques – leveraging formal methods to generate structures of provable correct programs, and allowing the LLM to complete the details with more flexibility. We focus on the problem of generating reactive programs, where these types of programs consume streams of input and produce streams of output. These programs are critical across application domains, including circuit design, self-driving cars, mobile apps, and chatbots – all of which we have been able to synthesize using our synthesis procedure. We end with an outline of the challenges still facing the integration of reactive synthesis into code generation paradigms.

### 3.12 Deep Learning for Reactive Synthesis

*Frederik Schmitt (CISPA – Saarbrücken, DE)*

License  Creative Commons BY 4.0 International license  
© Frederik Schmitt

Neural-symbolic computing offers a broad and largely unexplored spectrum of integrating neural and algorithmic components for developing new solutions to the reactive synthesis problem. At one end of the spectrum are purely symbolic and algorithmic methods that defined the field from its very beginning. In this talk, we will move to the other end of the spectrum and discuss how far we can get by relying on pure deep learning methods to solve synthesis problems. In particular, three approaches are presented that trace the current developments in deep learning:

1. training neural networks from scratch on data derived from specification patterns,
2. fine-tuning language and code generation models,
3. evaluating large language models and few-shot prompting on instances of parameterized specifications.

We specifically focus on representing the structure of synthesis problems in neural network architectures, the bundling of both algorithmic tools and neural networks, and we hope to spark discussions about approaches that are centred on the spectrum of neural-symbolic methods.

### 3.13 The power of feedback

*Anne-Kathrin Schmuck (MPI-SWS – Kaiserslautern, DE)*

License  Creative Commons BY 4.0 International license  
© Anne-Kathrin Schmuck

Feedback allows systems to seamlessly and instantaneously adapt their behavior to their environment and is thereby the fundamental principle of life and technology – it lets animals breathe, it stabilizes the climate, it allows airplanes to fly, and the energy grid to operate. During the last century, control technology excelled at using this power of feedback to engineer extremely stable, robust, and reliable technological systems. With the ubiquity of computing devices in modern technological systems, feedback loops become cyber-physical – the laws of physics governing technological, social or biological processes interact with (cyber) computing systems in a highly nontrivial manner, pushing towards higher and higher levels of autonomy and self-regulation. While stability, reliability and robustness remain to be of uppermost importance in these systems, a control-inspired utilization of cyber-physical feedback loops for this purpose is lacking far behind. In this talk, I will discuss how a control-inspired view on formal methods for reliable software design can enable us to utilize the power of feedback for robust and adaptable cyber-physical system design.

### 3.14 Constraint-based synthesis

*Armando Solar-Lezama (MIT – Cambridge, US)*

**License** © Creative Commons BY 4.0 International license  
© Armando Solar-Lezama

In this talk, I describe some recent efforts to combine functional and reactive synthesis. In the first part of the talk, I describe the Sketch program synthesis system, which allows users to write partial programs and solves for the missing details using an SMT solver. The talk focused on a new feature in Sketch that allows the user to write temporal specifications describing the behaviour of the program through its execution and showed how such constraints could be used to speed up the synthesis process and to give the user more control over the resulting program.

During the second part of the talk, I described a new effort to use a combination of reactive and functional synthesis to derive models of an environment from observations. The idea was implemented in a tool called Autumn, which focuses on pixel-world domains and is able to synthesize functional reactive programs from observations.

### 3.15 Synthesizing Pareto-optimal Interpretations for Black-box Models

*Hazem Torfah (Chalmers University of Technology – Göteborg, SE)*

**License** © Creative Commons BY 4.0 International license  
© Hazem Torfah

**Joint work of** Hazem Torfah, Shetal Shah, Supratik Chakraborty, S Akshay, Sanjit Seshia

**Main reference** Hazem Torfah, Shetal Shah, Supratik Chakraborty, S. Akshay, Sanjit A. Seshia: “Synthesizing Pareto-Optimal Interpretations for Black-Box Models”, in Proc. of the Formal Methods in Computer Aided Design, FMCAD 2021, New Haven, CT, USA, October 19-22, 2021, pp. 153–162, IEEE, 2021.


**URL** [https://doi.org/10.34727/2021/ISBN.978-3-85448-046-4\\_24](https://doi.org/10.34727/2021/ISBN.978-3-85448-046-4_24)

We present a multi-objective optimization approach for synthesizing interpretations of black-box models. Existing methods for synthesizing interpretations use a single objective function and are often optimized for a single class of interpretations. In contrast, we provide a more general and multi-objective synthesis framework that allows users to choose (1) the class of syntactic templates from which an interpretation should be synthesized, and (2) quantitative measures on both the correctness and explainability of an interpretation. For a given black-box, our approach yields a set of Pareto-optimal interpretations with respect to the correctness and explainability measures. We show that the underlying multi-objective optimization problem can be solved via a reduction to quantitative constraint solving, such as weighted maximum satisfiability. To demonstrate the benefits of our approach, we have applied it to synthesize interpretations for black-box neural-network classifiers. Our experiments show that there often exists a rich and varied set of choices for interpretations that are missed by existing approaches.

## 4 Working groups

### 4.1 Quantitative Specification

*Shaul Almagor (Technion – Haifa, IL)*

License  Creative Commons BY 4.0 International license  
© Shaul Almagor

Our discussion was aimed at the following question: can we find a natural specification formalism for which winning strategies in the synthesis problem are captured by well-studied quantitative computational models, such as One Counter Automata, One Counter Nets, VASS, etc.

After discussing several game types and specifications, we came up with the following concrete specification. Consider inputs  $\{a, \#\}$  and outputs  $\{b, @\}$ , the specification is to accept only words of the form  $a^n \# b^n @$ . That is, the environment inputs a sequence of  $a$ 's, and the system should respond with a length-matching sequence of  $b$ 's. It is natural to model a winning strategy using a One Counter Automaton in this case. We therefore wonder if this is a particular case of a more general specification formalism. One possible candidate is “PSL with local variables”, which seems to be able to capture specifications in this spirit.

### 4.2 A programmatic approach for reactive synthesis

*Nathanaël Fijalkow (CNRS – Talence, FR)*

License  Creative Commons BY 4.0 International license  
© Nathanaël Fijalkow

I proposed this working group to discuss a novel approach to reactive synthesis, where instead of a finite state controller the goal is to output a controller in the form of a program in a high-level programming language.

The key questions are:

- What is the (or a?) right programming language for reactive synthesis?
- How to perform inference?
- Even model-checking of programs is not obvious

The working session gathered about a dozen participants, and the lively discussions touched upon the three key questions mentioned above. Different approaches were sketched. The matter will be investigated more thoroughly in the coming months, thanks to the interests it sparked during this working session.

### 4.3 Minimization of deterministic parity automata

*Antonio Casares (University of Bordeaux, FR)*

License  Creative Commons BY 4.0 International license  
© Antonio Casares

In this working group, we discussed the complexity of the following decision problem: Input: A deterministic transition-based parity automaton  $\mathcal{A}$  and an integer  $k$ . Question: Is there a deterministic transition-based parity automaton of size at most  $k$  equivalent to  $\mathcal{A}$ ?

We considered the minimization procedure for good-for-games coBüchi automata by Abu Radi and Kupferman [1], and studied what are the kind of combinatorial problems that arise when trying to determinize these automata by adding a minimal number of states.

## References

- 1 Bader Abu Radi, Orna Kupferman. *Minimization and Canonization of GFG Transition-Based Automata*. Log. Methods Comput. Sci., 2022

## 4.4 Minimization of deterministic (co)Büchi automata

Rémi Morvan (*University of Bordeaux, FR*)

**License** © Creative Commons BY 4.0 International license  
© Rémi Morvan

This working group is a follow-up of Antonio Casares’ session “Minimization of deterministic parity automata”. We studied the following functional problem:

**input:** A deterministic transition-based (co)Büchi automaton  $\mathcal{A}$ .

**output:** A deterministic transition-based (co)Büchi automaton which is equivalent to  $\mathcal{A}$ , and state-minimal.

Contrary to minimization of parity automata, we believe this problem to be polynomial-time computable. We mostly focused on a congruence-based approach.

## 4.5 Positionality and memory

Pierre Ohlmann (*University of Warsaw, PL*)

**License** © Creative Commons BY 4.0 International license  
© Pierre Ohlmann

We worked on understanding memory requirements in games with topologically open winning conditions. The question can be phrased as follows. Let  $\Sigma$  be an alphabet:

**input:**  $L$  a language of  $\Sigma^*$ , say regular;

**output:** minimal  $k$  such that on any game graph with objective  $L \subseteq \Sigma^\omega$ , if Eve wins then she wins with a  $k$ -states memory

We discussed a few examples and motivations and talked about the case where  $L$  is a singleton which has a simple solution. We did not make substantial progress on the general case.

## 4.6 Graph neural networks and reactive synthesis

Guillermo A. Pérez (*University of Antwerp, BE*)

**License** © Creative Commons BY 4.0 International license  
© Guillermo A. Pérez  
**Joint work of** Guillermo A. Pérez, Isseine Calviac

During this working session, we discussed the possibility of using graph neural networks (GNNs) to reduce the state space of automata used for synthesis from LTL specifications. The state of the art (in current LTL-synthesis solvers) concerns the syntactic recognition

of subformulas that are treated with special transformations which may lead to minimal subautomata. Finally, the product of said automata is taken to obtain a final automaton. To further reduce the size of the final automaton, Spot /ltlsynt implements an approximation of an isomorphism check (rather, it resembles graded bisimulation). This option seems to be disabled by default as it takes too long. An alternative concerns guessing a bisimulation relation and checking (in logspace, using one step of the classical partition refinement algorithm for bisimulation) that it is indeed a bisimulation relation. This begs the question of whether machine learning can help us make such a guess.

On the GNN side, it was mentioned that they are a very hot topic in AI and that the main focus of current research concerns the proposal of new architectures to improve their “expressiveness”. GNNs can be seen as a “recolouring” function applied to a coloured graph. It is known that this recolouring function cannot colour nodes differently than the Weisfeiler-Leman (WL) algorithm would deem as equivalent. Conversely, for every coloured graph, there are weight matrices and bias vectors such that GNNs implement the WL algorithm. Guillermo A. Pérez mentioned an unpublished result: GNNs can also implement the (coarsest) bisimulation relation.

Open questions:

1. Can other relations of interest be implemented using different GNN architectures?
2. To leverage GNNs in graphs that come from verification applications, one needs to obtain useful and meaningful features for the vertices. How can these be obtained easily or even automatically?
3. For large graphs, recent works and libraries suggest sampling a number of neighbours of each node. Will this render GNNs useless for verification methods in large graphs?

## References

- 1 Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, Martin Grohe: Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks. AAAI 2019: 4602-4609
- 2 Floris Geerts, Filip Mazowiecki, Guillermo A. Pérez: Let’s Agree to Degree: Comparing Graph Convolutional Networks in the Message-Passing Framework. ICML 2021: 3640-3649
- 3 Tobias Hecking, Swathi Muthukrishnan, Alexander Weinert. Predicting Winning Regions in Parity Games via Graph Neural Networks. Deep Learning-aided Verification @ CAV 2023.

## 4.7 SYNTCOMP benchmarking

*Guillermo A. Pérez (University of Antwerp, BE)*

License © Creative Commons BY 4.0 International license

© Guillermo A. Pérez

Joint work of Swen Jacobs, Guillermo A. Pérez, Philipp Schlehuber-Caissier

Benchmark sets for existing and upcoming tracks of the reactive synthesis competition (SYNTCOMP) were discussed. The main points that were touched during the discussion can be summarized as follows.

1. PDDL is a well-established family of languages in the planning community for describing tasks. In the short term, we can try to translate (non-deterministic) PDDL specifications to TLSF. In the long term, PDDL itself may become a reasonable format for some tracks, and we may further extend it with goals expressed in LTL over finite words.

2. While reports of SYNTCOMP usually include graphs covering all benchmarks, it may be of more importance to highlight how well tools scale per parametric family of benchmarks as the parameter values increase. This is already present in the report for the 2018-2021 editions of the competition. A proposal is to include such graphs in the results of every forthcoming edition of the competition.
3. PSL is a well-established extension of LTL that seems to be used in industry. We will survey the literature on PSL case studies and perhaps even approach IBM, Synopsys, and other companies to ask whether they have such specifications so that we can enrich the set of benchmarks.
4. Regarding the output format of synthesis tools: The current one is quite succinct and it matches the input of model checking tools. Namely, SYNTCOMP requires output in AIGER format. However, semi-explicit representations such as HOA for Mealy/Moore machines may be easier to visualize. Hence, for future editions of the competition, an “explainable badge” will be awarded to tools that produce such semi- or fully explicit versions of their output (as an additional option, not as the official output for the competition).
5. Finally, we noted that the parity game track is currently biased as it measures the time for parsing and minimization of the parity game together with the time it takes to actually solve the game. Instead, a proposal is to split this into preprocessing time and solving time. In the short term, tools will have to output a message and timestamp stating when preprocessing is finished so that we can split the two processes. In the long term, we can consider more succinct formats of a parity game to have parsing become faster (after using, perhaps, the BDD version of *hoa-tools*) and to be able to succinctly encode even larger games.

## 4.8 IPASIR-UP: User Propagators for CDCL

*Andre Schidler (TU Wien, AT)*

**License** © Creative Commons BY 4.0 International license

© Andre Schidler

**URL** <https://simons.berkeley.edu/talks/katalin-fazekas-tu-wien-2023-04-17>

This talk is a teaser talk about a new SAT solver API – IPASIR-UP – that provides interactions with the solver during the solving process. Hence, instead of calling the solver incrementally, it is possible to interact with the solver during the solving process, i.e., whenever the solver makes a decision, propagates a variable, finds a conflict, finds a model, etc.

IPASIR-UP allows, among other things, implementing CEGAR approaches or theories quickly and cleanly. So far, this API is available in Cadical.



## 4.9 Reactive Synthesis Beyond the Booleans

*César Sánchez (IMDEA Software Institute – Madrid, ES)*

License  Creative Commons BY 4.0 International license  
© César Sánchez

There has been a growing interest in the last few years in increasing the expressivity of reactive synthesis from propositional LTL into richer languages that can handle data. One impact-full line of research includes temporal stream logic (TSL) that has managed to synthesize sophisticated controllers. Another example was Raina Dimitrova’s work presented in this seminar. However, most attempts to increase the expressivity quickly render the realizability decision problem undecidable.

In the first part of this working session we discussed the recent work on realizability of “LTL modulo theory” that shows decidability of an extension of LTL where the propositions are replaced by literals from a first order theories. The Boolean abstraction method generates an equi-realizable propositional LTL specification as long as the first-order theory enjoys a decidable exists-forall (validity) decision problem. Moreover, the resulting specification remains in the same temporal class and is amenable of Boolean reactive synthesis. Then we discussed how the resulting (Boolean) controllers obtained using existing synthesis tools can be extended to “theory controllers”, thus obtaining a full LTL modulo theory synthesis algorithm.

However, current LTL modulo theory does not allow to transfer data across time (which is the main reason for undecidability of richer formalisms like TSL). We briefly discussed possibilities for enriching LTL modulo theories with controlled data transferred, particularly based on known decidability results from register automata. We concluded that a first promising approach should be based on specific characteristics of each theory and not (as for LTL modulo theories described above) agnostic to the theory in question.

## 4.10 Reactive synthesis of Linear Temporal Logic on finite traces

*Shufang Zhu (University of Oxford, GB)*

License  Creative Commons BY 4.0 International license  
© Shufang Zhu

In this working group, we discussed the following problem in reactive synthesis of Linear Temporal Logic on finite traces (LTLf):

Consider an autonomous system immersing in an environment, where there are multiple abstraction levels of how the environment behaves, depending on how much environment dynamics to be concerned:

- $\text{Env}_1$  (any possible env behaviours),
- $\text{Env}_2$ ,
- ...
- $\text{Env}_m$  (most restricted env behaviours).

The system is given multiple tasks, ranging from most difficult to easiest:

- $\text{Task}_1$  (most difficult),
- $\text{Task}_2$ ,
- ...
- $\text{Task}_n$  (easiest).

Question: How to achieve a good balance to achieve a more difficult task considering a more flexible environment?

We discussed different ways of dealing with this problem. A good direction might be looking at robust LTL [1]. Another interesting direction is MaxSAT or Weighted MaxSAT.

#### References

- 1 Paulo Tabuada, Daniel Neider. *Robust Linear Temporal Logic*

## Participants

- Shaull Almagor  
Technion – Haifa, IL
- Guy Avni  
University of Haifa, IL
- Mrudula Balachander  
Free University of Brussels, BE
- Véronique Bruyère  
University of Mons, BE
- Michaël Cadilhac  
DePaul University – Chicago, US
- Antonio Casares  
University of Bordeaux, FR
- Rayna Dimitrova  
CISPA – Saarbrücken, DE
- Alexandre Duret-Lutz  
EPITA – Le Kremlin Bicêtre, FR
- Rüdiger Ehlers  
TU Clausthal, DE
- Nathanaël Fijalkow  
CNRS – Talence, FR
- Emmanuel Filiot  
UL – Brussels, BE
- Bernd Finkbeiner  
CISPA – Saarbrücken, DE
- Dana Fisman  
Ben Gurion University –  
Beer Sheva, IL
- Hadar Frenkel  
CISPA – Saarbrücken, DE
- Swen Jacobs  
CISPA – Saarbrücken, DE
- Ayrat Khalimov  
TU Clausthal, DE
- Bakh Khoussainov  
Univ. of Electronic Science &  
Technology – Chengdu, CN
- Rupak Majumdar  
MPI-SWS – Kaiserslautern, DE
- Théo Matricon  
University of Bordeaux, FR
- Niklas Metzger  
CISPA – Saarbrücken, DE
- Rémi Morvan  
University of Bordeaux, FR
- Anca Muscholl  
University of Bordeaux, FR
- Pierre Ohlmann  
University of Warsaw, PL
- Guillermo A. Pérez  
University of Antwerp, BE
- Ruzica Piskac  
Yale University – New Haven, US
- Elizabeth Polgreen  
University of Edinburgh, GB
- Mickael Randour  
F.R.S.-FNRS & UMONS –  
Université de Mons, BE
- César Sánchez  
IMDEA Software Institute –  
Madrid, ES
- Mark Santolucito  
Barnard College, Columbia  
University – New York, US
- Andre Schidler  
TU Wien, AT
- Frederik Schmitt  
CISPA – Saarbrücken, DE
- Anne-Kathrin Schmuck  
MPI-SWS – Kaiserslautern, DE
- Martina Seidl  
Johannes Kepler Universität  
Linz, AT
- Armando Solar-Lezama  
MIT – Cambridge, US
- Hazem Torfah  
Chalmers University of  
Technology – Göteborg, SE
- Tom van Dijk  
University of Twente –  
Enschede, NL
- Shufang Zhu  
University of Oxford, GB
- Martin Zimmermann  
Aalborg University, DK

