Report from Dagstuhl Seminar 23112

Unifying Formal Methods for Trustworthy Distributed Systems

Swen Jacobs[*1], **Kenneth McMillan**[*2], **Roopsha Samanta**[*3], **and Ilya Sergey**[*4]

1 **CISPA – Saarbrücken, DE.** jacobs@cispa.de
2 **University of Texas – Austin, US.** kenmcm@cs.utexas.edu
3 **Purdue University – West Lafayette, US.** roopsha@purdue.edu
4 **National University of Singapore, SG.** ilya@nus.edu.sg

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 23112 "Unifying Formal Methods for Trustworthy Distributed Systems".

Distributed systems are challenging to develop and reason about. Unsurprisingly, there have been many efforts in formally specifying, modeling, and verifying distributed systems. A bird's eye view of this vast body of work reveals two primary sensibilities. The first is that of semi-automated or interactive deductive verification targeting structured programs and implementations, and focusing on simplifying the user's task of providing inductive invariants. The second is that of fully-automated model checking, targeting more abstract models of distributed systems, and focusing on extending the boundaries of decidability for the parameterized model checking problem. Regrettably, solution frameworks and results in deductive verification and parameterized model checking have largely evolved in isolation while targeting the same overall goal.

This seminar aimed at enabling conversations and solutions cutting across the deductive verification and model checking communities, leveraging the complementary strengths of these approaches. In particular, we explored layered and compositional approaches for modeling and verification of industrial-scale distributed systems that lend themselves well to separation of verification tasks, and thereby the use of diverse proof methodologies.

Seminar March 12–15, 2023 – https://www.dagstuhl.de/23112
2012 ACM Subject Classification Computing methodologies → Distributed algorithms; Theory of computation → Concurrency; Theory of computation → Program verification
Keywords and phrases Deductive Verification, Distributed Algorithms, Formal Verification, Model Checking
Digital Object Identifier 10.4230/DagRep.13.3.32

1 Executive Summary

Swen Jacobs (CISPA – Saarbrücken, DE)
Kenneth McMillan (University of Texas – Austin, US)
Roopsha Samanta (Purdue University – West Lafayette, US)
Ilya Sergey (National University of Singapore, SG)

Dagstuhl Seminar 23112 Unifying Formal Methods for Trustworthy Distributed Systems took place on March 12–15, 2023 and had 25 participants: 9 female and 16 male, 22 from academia and 4 from industry, representing 9 different countries.

* Editor / Organizer

This was a short seminar spanning 2.5 days and included four one-hour keynotes, 16 regular and short (lightning) talks, as well as two two-hour whole-seminar plenary discussions. The keynote talks were given by

1. Peter Müller (ETH Zurich) on Verified Secure Routing
2. Ken McMillan (UT Austin) on Techniques for Decidable Verification
3. Swen Jacobs (CISPA) on Parameterized Model Checking and Synthesis
4. Murdoch Jamie Gabbay (Heriot-Watt University) on Semitopologies for Heterogeneous Consensus.

The abstracts of all talks appear in this seminar report, except for one of the keynotes and two impromptu talks for which we only give the titles here:

- "Taming Unbounded Distributed Systems with Modular, Bounded Verification" by Roopsha Samanta (Purdue University), and
- "Pushing Formal Methods Tools to Industry" by Mike Dodds (Galois).

The two plenary discussions that have taken place during the seminar were focusing on the topics of (1) performing comparative studies amongst different approaches for validating distributed systems and (2) grand challenges that call for joint efforts across different approaches and schools of thought in this area.

The outcome of the first discussion was an informal proposal on a "Distributed System Verification Competition" – a community effort in the spirit of the famous "VerifyThis" competition in software verification, which would offer, on a regular basis, a selection of micro-benchmarks and semi-artificial challenges in verification, validation, and bug-finding in distributed system, focusing on different aspects of safety, liveness and providing a landscape to showcase the recent advances in interactive or automated verification.

The second panel has concluded with several ideas of a large-scale verification/validation effort in distributed systems. The most viable option was suggested based on the topic of the first keynote talk on Verified Secure Routing, which is currently only partially achieved by a combination of two specific technologies and leaves a lot of room to improvement, both in terms of specification of the properties of interest (e.g., liveness) as well as for exploring possibilities for automating proofs as well as complementing sound verification methods with testing and dynamic analyses.

Given the short nature of this seminar, the social component of its program was limited to a dinner in local restaurant "Zum Schloßberg", during which possibilities for collaboration have been discussed between the participants. As one outcome of this social interaction, possible internship opportunities in system verification were offered by one of the industry participants, with one of the junior participants currently considering taking them for the Summer 2024.

The seminar has generated several ideas for follow-up meetings. In particular, the following areas will likely benefit from more focused discussions and exchanges: (a) testing and dynamic validation of distributed systems; (b) addressing the challenge of so-called "latent proof" (ignored abstraction gap) in automated verification, and (c) programming-language based techniques for implementing large-scale systems with a support for formal reasoning and verification.

2 Table of Contents

3 Overview of Talks

3.1 Session types, time, timeout

Laura Bocchi (University of Kent – Canterbury, GB)

In this talk I give an introduction on binary session types, outline their relation with other formalisms, in particular communicating finite state machines, and with verification problems. I then discuss the links between session types and programming languages, and some of their usage scenarios that include static typing, run-time monitoring, and API generation. Finally, I present the extension of session types with time constraints and timeouts, discussing recent and ongoing work, as well as open problems.

3.2 Commutativity Quotients of Concurrent or Distributed Algorithms

Constantin Enea (Ecole Polytechnique – Palaiseau, FR)

Joint work of Constantin Enea, Parisa Fathololumi, Eric Koskinen
Main reference Constantin Enea, Parisa Fathololumi, Eric Koskinen: "The Commutativity Quotients of Concurrent Objects", CoRR, Vol. abs/2301.05740, 2023.
URL https://doi.org//10.48550/arXiv.2301.05740

Concurrent or distributed algorithms form the foundation of many modern automated services. Reasoning about the fine-grained complexities (interleavings, invariants, etc.) of these algorithms, however, is notoriously difficult. Formal proof methodologies for arguing about their correctness are still somewhat disconnected from the intuitive correctness arguments. Intuitions are often about a few canonical executions, possibly with few threads, whereas formal proofs would often use generic but complex arguments about arbitrary interleavings over unboundedly many threads. As a way to bring formal proofs closer to intuitive arguments, we introduce a new methodology for characterizing the interleavings of concurrent or distributed algorithms, based on their commutativity quotient. This quotient represents every interleaving up to reordering of commutative steps and, when chosen carefully, admits simple abstractions in the form of regular or context-free languages that enable simple proofs of correctness.

3.3 Checking Qualitative Liveness Properties of Replicated Systems with Stochastic Scheduling

Javier Esparza (TU München, DE)

Joint work of Michael Blondin, Javier Esparza, Martin Helfrich, Antonín Kucera, Philipp J. Meyer
Main reference Michael Blondin, Javier Esparza, Martin Helfrich, Antonín Kucera, Philipp J. Meyer: "Checking Qualitative Liveness Properties of Replicated Systems with Stochastic Scheduling", in Proc. of the Computer Aided Verification – 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21-24, 2020, Proceedings, Part II, Lecture Notes in Computer Science, Vol. 12225, pp. 372–397, Springer, 2020.
URL https://doi.org//10.1007/978-3-030-53291-8_20

We present a sound and complete method for the verification of qualitative liveness properties of replicated systems under stochastic scheduling. These are systems consisting of a finite-state program, executed by an unknown number of indistinguishable agents, where the next agent to make a move is determined by the result of a random experiment. We show that if a property of such a system holds, then there is always a witness in the shape of a Presburger stage graph: a finite graph whose nodes are Presburger-definable sets of configurations. Due to the high complexity of the verification problem (Ackermann-complete), we introduce an incomplete procedure for the construction of Presburger stage graphs, and implement it on top of an SMT solver. The procedure makes extensive use of the theory of well-quasi-orders, and of the structural theory of Petri nets and vector addition systems. We apply our results to a set of benchmarks, in particular to a large collection of population protocols, a model of distributed computation extensively studied by the distributed computing community.

3.4 The semitopology of permissionless consensus

Murdoch Jamie Gabbay (Heriot-Watt University – Edinburgh, GB) and Giuliano Losa (Stellar Development Foundation – San Francisco, US)

Main reference Murdoch Gabbay, Giuliano Losa: "Semitopology: a new topological model of heterogeneous consensus", CoRR, Vol. abs/2303.09287, 2023.
URL https://doi.org//10.48550/arXiv.2303.09287

A distributed system is *permissionless* when participants can join and leave the network without permission from a central authority. Many modern distributed systems are naturally permissionless, in the sense that a central permissioning authority would defeat their design purpose: this includes blockchains, filesharing protocols, some voting systems, and more. Due to their permissionless nature, such systems are also heterogeneous: participants may only have a partial view of the system, and they may also have different goals and beliefs. The traditional notion of consensus, i.e. system-wide agreement, may therefore not be adequate.

This is a mathematical challenge; how should we understand what permissionless consensus means? And how can we use this understanding to build mathematical models to help us engineer simple, robust, effective, and secure practical systems?

We study a new definition of permissionless consensus, based on *semitopology* – like topology, but without the restriction that intersections of opens be open. Semitopologies have a rich theory which is related to topology, but with a distinct character and mathematics. We introduce novel well-behavedness conditions, including an anti-Hausdorff property and a new notion of '*topen set*, and we show how these structures relate to consensus. We give a restriction of semitopologies to *witness semitopologies*, which are an algorithmically tractable subclass corresponding to Horn clause theories, having particularly good mathematical properties.

3.5 Parameterized Model Checking (and Synthesis)

Swen Jacobs (CISPA – Saarbrücken, DE)

Joint work of Roderick Bloem, Swen Jacobs, Ayrat Khalimov, Igor Konnov, Sasha Rubin, Helmut Veith, Josef Widder, Nouraldin Jaber, Christopher Wagner, Milind Kulkarni, Roopsha Samanta

In this talk, I first gave a short overview of existing results in the area of parameterized model checking, including an introduction of basic techniques for obtaining decidability results, and more recent results that build on and extend these techniques [1]. In the second half of the talk, I presented our own recent work in the area. Here, I introduced the computational model of global synchronization protocols (GSPs), and described how we obtained decidability and cutoff results for its parameterized model checking problem [2, 3]. Finally, I showed how these results not only enable verification, but also synthesis, which furthermore can relieve the designer of the system from fitting the system into the decidable fragment, instead letting the synthesis algorithm take care of that [4].

References

1 Roderick Bloem, Swen Jacobs, Ayrat Khalimov, Igor Konnov, Sasha Rubin, Helmut Veith, Josef Widder. *Decidability of Parameterized Verification*. Synthesis Lectures on Distributed Computing Theory, Morgan & Claypool Publishers 2015

2 Nouraldin Jaber, Swen Jacobs, Christopher Wagner, Milind Kulkarni, Roopsha Samanta. *Parameterized Verification of Systems with Global Synchronization and Guards*. CAV (1) 2020: 299-323

3 Nouraldin Jaber, Christopher Wagner, Swen Jacobs, Milind Kulkarni, Roopsha Samanta. *QuickSilver: modeling and parameterized verification for distributed agreement-based systems*. Proc. ACM Program. Lang. 5(OOPSLA): 1-31 (2021)

4 Nouraldin Jaber, Christopher Wagner, Swen Jacobs, Milind Kulkarni, Roopsha Samanta. *Synthesis of Distributed Agreement-Based Systems with Efficiently-Decidable Verification*. TACAS (2) 2023: 289-308

3.6 Verifying Indistinguishability of Privacy-Preserving Protocols

Gowtham Kaki (University of Colorado – Boulder, US)

Joint work of Kirby Linvill, Gowtham Kaki, Eric Wustrow
Main reference Kirby Linvill, Gowtham Kaki, Eric Wustrow: "Verifying Indistinguishability of Privacy-Preserving Protocols". Under submission.

Internet users rely on the protocols they use to protect their private information including their identity and the websites they visit. Formal verification of these protocols can detect subtle bugs that compromise these protections at design time, but is a challenging task as it involves probabilistic reasoning about random sampling, cryptographic primitives, and concurrent execution. Existing approaches either reason about symbolic models of the protocols that sacrifice precision for automation, or reason about more precise models that are harder to automate and require cryptographic expertise. In this talk I describe a novel approach to verifying privacy-preserving protocols that is more precise than symbolic models yet more accessible than computational models. Our approach permits direct-style proofs of privacy, as opposed to indirect game-based proofs in computational models, by formalizing privacy as indistinguishability of possible network traces induced by a protocol. We ease automation by leveraging insights from the distributed systems verification community to create sound synchronous models of concurrent protocols. Our verification framework is implemented in F* as a library we call Waldo. I talk about two large case studies of using Waldo to verify indistinguishability; one on the Encrypted Client Hello (ECH) extension of the TLS protocol and another on a Private Information Retrieval (PIR) protocol. I describe subtle flaws we uncovered in the TLS ECH specification that were missed by other efforts.

3.7 Improving usability of TLA+ tools for blockchain engineers

Igor Konnov (Informal Systems – Wien, AT)

Joint work of Igor Konnov, Shon Feder, Jure Kukovec, Gabriela Moreira, Thomas Pani

In this talk, I gave a brief introduction into the Cosmos ecosystem and a summary of results on formal specification & model checking of blockchain protocols conducted at Informal Systems. We further discussed the benefits and practical challenges of applying Temporal Logic of Actions (TLA+) and the Apalache model checker in the blockchain industry. The talk concluded with an introduction of Quint, the new syntax for the logic of TLA+. We introduced a new specification development cycle, which accommodates the needs of the protocol designers, blockchain engineers, and verification engineers.

More details about Quint may be found at the project webpage: `https://github.com/informalsystems/quint/`.

3.8 Random testing of Byzantine fault tolerant algorithms

Burcu Kulahcioglu Ozkan (TU Delft, NL)

Joint work of Levin N. Winter, Florena Buse, Daan de Graaf, Klaus von Gleissenthall, Burcu Kulahcioglu Ozkan
Main reference Levin N. Winter, Florena Buse, Daan de Graaf, Klaus von Gleissenthall, Burcu Kulahcioglu Ozkan: "Randomized Testing of Byzantine Fault Tolerant Algorithms", Proc. ACM Program. Lang., Vol. 7(OOPSLA1), pp. 757–788, 2023.
URL https://doi.org//10.1145/3586053

Byzantine fault-tolerant algorithms promise agreement on a correct value, even if a subset of processes can deviate from the algorithm arbitrarily. While these algorithms provide strong guarantees in theory, protocol bugs and implementation mistakes may cause them to violate fault tolerance in practice.

This talk discusses the challenges of testing Byzantine fault-tolerant systems and introduces ByzzFuzz, a method for automatically finding errors in implementations of Byzantine fault-tolerant algorithms through randomized testing. ByzzFuzz detects fault-tolerance bugs by injecting randomly generated network and process faults into their executions. To navigate the space of possible process faults, ByzzFuzz introduces small-scope message mutations which mutate the contents of the protocol messages by applying small changes to the original message either in value (e.g., by incrementing the round number) or in time (e.g., by repeating a proposal value from a previous message). The evaluation of ByzzFuzz on the implementations of popular blockchains show that small-scope mutations, combined with insights from the testing and fuzzing literature, are effective at uncovering protocol logic and implementation bugs in real-world fault-tolerant systems.

3.9 Verified Causal Broadcast with Liquid Haskell

Lindsey Kuper (University of California – Santa Cruz, US)

Joint work of Patrick Redmond, Gan Shen, Niki Vazou, Lindsey Kuper

Protocols to ensure that messages are delivered in causal order are a ubiquitous building block of distributed systems. For instance, distributed data storage systems can use causally ordered message delivery to ensure causal consistency, and CRDTs can rely on the existence of an underlying causally-ordered messaging layer to simplify their implementation. A causal delivery protocol ensures that when a message is delivered to a process, any causally preceding messages sent to the same process have already been delivered to it. While causal delivery protocols are widely used, verification of their correctness is less common, much less machine-checked proofs about executable implementations.

We implemented a standard causal broadcast protocol in Haskell and used the Liquid Haskell solver-aided verification system to express and mechanically prove that messages will never be delivered to a process in an order that violates causality. We express this property using refinement types and prove that it holds of our implementation, taking advantage of Liquid Haskell's underlying SMT solver to automate parts of the proof and using its manual theorem-proving features for the rest. We then put our verified causal broadcast implementation to work as the foundation of a distributed key-value store.

3.10 Potential-based semantics for causally consistent shared memory

Ori Lahav (Tel Aviv University, IL)

Joint work of Ori Lahav, Udi Boker
Main reference Ori Lahav, Udi Boker: "What's Decidable About Causally Consistent Shared Memory?", ACM Trans. Program. Lang. Syst., Vol. 44(2), pp. 8:1–8:55, 2022.
URL https://doi.org//10.1145/3505273

While causal consistency is one of the most fundamental consistency models weaker than sequential consistency, the decidability of safety verification for (finite-state) concurrent programs running under causally consistent shared memories is still unclear. In this paper, we establish the decidability of this problem for two standard and well-studied variants of causal consistency. To do so, for each variant, we develop an equivalent "lossy" operational semantics, whose states track possible futures, rather than more standard semantics that record the history of the execution. We show that these semantics constitute well-structured transition systems, thus enabling decidable verification. Based on a key observation, which we call the "shared-memory causality principle", the two novel semantics may also be of independent use in the investigation of weakly consistent models and their verification. Interestingly, our results are in contrast to the undecidability of this problem under the Release/Acquire fragment of the C/C++11 memory model, which forms another variant of causally consistent memory that, in terms of allowed outcomes, lies strictly between the two models studied here. Nevertheless, we show that all these three variants coincide for write/write-race-free programs, which implies the decidability of verification for such programs under Release/Acquire.

References

1 Ori Lahav. *Verification under causally consistent shared memory.* ACM SIGLOG News 6:2, April 2019, pages 43–56. `https://doi.org/10.1145/3326938.3326942`

2 Ori Lahav and Udi Boker. *Decidable verification under a causally consistent shared memory.* In *PLDI*, ACM, 2020. `https://doi.org/10.1145/3385412.3385966`

3 Lahav, O., Boker, U.: What's Decidable About Causally Consistent Shared Memory? ACM Trans. Program. Lang. Syst. **44**(2), 8:1–8:55 (2022), `https://doi.org/10.1145/3505273`

3.11 Parameterized Verification of Randomized Consensus Algorithms

Marijana Lazic (TU München, DE)

In this talk I showed the extension of threshold automata for modeling randomized consensus algorithms that perform an unbounded number of asynchronous rounds. Moreover, I presented techniques for parameterized verification of the three randomized consensus properties: agreement, validity and almost sure termination.

For non-probabilistic properties, I showed that it is necessary and sufficient to verify these properties under round-rigid schedules, that is, schedules where processes enter round r only after all processes finished round $r-1$.

For almost-sure termination, I proceed in 2 steps. First, I analyze these algorithms under round-rigid adversaries, that is, fair adversaries that only generate round-rigid schedules. This allows us to do compositional and inductive reasoning that reduces verification of the asynchronous multi-round algorithms to model checking of a one-round threshold automaton.

We apply this framework and automatically verify the following classic algorithms: Ben-Or's and Bracha's seminal consensus algorithms for crashes and Byzantine faults, 2-set agreement for crash faults, and RS-Bosco for the Byzantine case.

Second, I focus on weak adversaries, that express the property that the adversary (scheduler), which has to decide which messages to deliver to which process, has no means of inferring the outcome of random choices, and the content of the messages. I introduced a model for randomized distributed algorithms that allows us to formalize the notion of weak adversaries. I show that for verification purposes, the class of weak adversaries can be restricted to round-rigid adversaries. This new reduction theorem paves the way to the parameterized verification of randomized distributed algorithms under the more realistic weak adversaries.

References

1 Nathalie Bertrand, Marijana Lazic, Josef Widder. *A Reduction Theorem for Randomized Distributed Algorithms Under Weak Adversaries.* VMCAI 2021: 219-239

2 Nathalie Bertrand, Igor Konnov, Marijana Lazic, Josef Widder. *Verification of Randomized Consensus Algorithms Under Round-Rigid Adversaries.* CONCUR 2019: 33:1-33:15

3.12 A simple proof of the FLP impossibility result

Giuliano Losa (Stellar Development Foundation – San Francisco, US)

Joint work of Giuliano Losa, Eli Gafni
Main reference Eli Gafni, Giuliano Losa: "Time is not a Healer, but it Sure Makes Hindsight 20:20", CoRR, Vol. abs/2305.02295, 2023.
URL https://doi.org//10.48550/arXiv.2305.02295

We present a remarkably simple proof of the famous FLP impossibility result. We first observe that solving consensus in an asynchronous system where one process may fail implies solving consensus in the synchronous model of Santoro and Widmayer. Then, we build on insights from Volzer to obtain an almost trivial impossibility proof in the synchronous model.

3.13 Random Testing of Distributed Systems

Rupak Majumdar (MPI-SWS – Kaiserslautern, DE)

This talk was an overview of the main ideas behind the state-of-the-art techniques for effective and efficient fuzz-testing of realistic distributed systems. I have outlined the basic theory facts that explain why well-adopted "black-box" testing tools, such as Jepsen, are surprisingly effective in discovering interesting bugs in distributed systems. I have also described approaches that can be used to improve the algorithms that navigated through the very large space of possible distributed interactions by exploiting the ideas of partial synchrony and round-based formulation of distributed protocols. These algorithms define a sample space based on the underlying partial orderings of events in the distributed system and sample efficiently from that space.

The talk described work that appeared in the following papers and dissertation:

References

1 Rupak Majumdar and Filip Niksic. Why is random testing effective for partition tolerance bugs? Proc. ACM Program. Lang. 2(POPL): 46:1-46:24 (2018)

2 Burcu Kulahcioglu Ozkan, Rupak Majumdar, Filip Niksic, Mitra Tabaei Befrouei, Georg Weissenbacher. Randomized testing of distributed systems with probabilistic guarantees. Proc. ACM Program. Lang. 2(OOPSLA): 160:1-160:28 (2018)

3 Filip Niksic. Combinatorial Constructions for Effective Testing. Kaiserslautern University of Technology, Germany, 2019

4 Cezara Dragoi, Constantin Enea, Burcu Kulahcioglu Ozkan, Rupak Majumdar, Filip Niksic: Testing consensus implementations using communication closure. Proc. ACM Program. Lang. 4(OOPSLA): 210:1-210:29 (2020)

3.14 Verified Secure Routing

Peter Müller (ETH Zürich, CH)

SCION is a new Internet architecture that addresses many of the security vulnerabilities of today's Internet. Its clean-slate design provides, among other properties, route control, failure isolation, and multi-path communication. The verifiedSCION project is an effort to formally verify the correctness and security of SCION. It aims to provide strong guarantees for the entire architecture, from the protocol design to its concrete implementation. The project uses stepwise refinement to prove that the protocol withstands increasingly strong attackers. The refinement proofs assume that all network components such as routers satisfy their specifications. This property is then verified separately using deductive program verification in separation logic. This talk will give an overview of the verifiedSCION project and explain, in particular, how we verify code-level properties such as memory safety, I/O behavior, and information flow security.

3.15 Interactive Synthesis of Distributed Protocols

Kedar Namjoshi (Nokia Bell Labs – Murray Hill, US)

Joint work of Kedar Namjoshi, Senthil Rajasekaran

It is difficult to verify distributed protocols: one must prove that all configurations satisfy a global property, which is in general an undecidable question. Could one synthesize such protocols instead? That is undecidable, too; but we suggest using a process of successive refinement on specifications, with the goal of obtaining a specification that is localized to a generic process and a generic neighborhood, which is simpler to synthesize. A protocol designer suggests the sequence of specifications, with automated help in establishing refinements.

3.16 Reasoning about Byzantine Accountability

Ilya Sergey (National University of Singapore, SG) and George Pîrlea (National University of Singapore, SG)

Modern Byzantine distributed consensus protocols can achieve agreement in the presence of a bounded number of faulty nodes trying to corrupt the network, yet they fail to identify or disincentivise Byzantine behaviour by malicious nodes. Accountable Byzantine Consensus (ABC) is a protocol transformation that, when combined with any Byzantine consensus protocol, guarantees both consensus and accountability.

I this talk, I presented the key ideas of Byzantine accountability, its semantic model, as well some preliminary results on formalising and verifying accountable consensus protocols.

3.17 Finding Infinite Counter Models in Deductive Verification

Sharon Shoham Buchbinder (Tel Aviv University, IL)

Joint work of Sharon Shoham Buchbinder, Neta Elad, Oded Padon

First-order logic, and quantifiers in particular, are widely used in deductive verification of programs and systems. Quantifiers are essential for describing systems with unbounded domains, but prove difficult for automated solvers. Significant effort has been dedicated to finding quantifier instantiations that establish unsatisfiability of quantified formulas, thus ensuring validity of a system's verification conditions. However, in many cases the formulas are satisfiable—this is often the case in intermediate steps of the verification process, e.g., when an invariant is not yet inductive. For such cases, existing tools are limited to finding finite models. Yet, some quantified formulas are satisfiable but only have infinite models, which current solvers are unable to find. Such infinite counter-models are especially typical when first-order logic is used to approximate the natural numbers, the integers, or other inductive definitions, which is common in deductive verification.

In this work, we tackle the problem of finding such infinite models, specifically, finite representations thereof that can be presented to the user of a deductive verification tool. These models give insight into the verification failure, and allow the user to identify and fix bugs in the modeling of the system and its properties. Our approach consists of three parts. First, we introduce templates as a way to represent certain infinite models, and show that formulas can be efficiently model checked against them. Second, we identify a new decidable fragment of first-order logic that extends and subsumes EPR, where satisfiable formulas always have a model representable by a template of a bounded size. Finally, we describe an effective decision procedure to symbolically explore this (usually vast) search space of templates.

We evaluate our approach on examples from a variety of domains: distributed consensus protocols, linked lists, and axiomatic arithmetic. Our implementation quickly finds infinite counter-models that demonstrate the source of verification failures in a simple way, even in cases beyond the decidable fragment, while state-of-the-art SMT solvers and theorem provers such as Z3, cvc5, and Vampire diverge or return "unknown".

3.18 Deadlock-free asynchronous message reordering in Rust with multiparty session types

Nobuko Yoshida (University of Oxford, GB)

Joint work of Zak Cutner, Nobuko Yoshida, Martin Vassor
Main reference Zak Cutner, Nobuko Yoshida, Martin Vassor: "Deadlock-free asynchronous message reordering in rust with multiparty session types", in Proc. of the PPoPP '22: 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Seoul, Republic of Korea, April 2 – 6, 2022, pp. 246–261, ACM, 2022.
URL https://doi.org//10.1145/3503221.3508404

Rust is a modern systems language focussed on performance and reliability. Complementing Rust's promise to provide "fearless concurrency," developers frequently exploit asynchronous message passing. Unfortunately, sending and receiving messages in an arbitrary order to maximise computation-communication overlap (a popular optimisation in message-passing applications) opens up a Pandora's box of subtle concurrency bugs. To guarantee deadlock-freedom by construction, we present Rumpsteak: a new Rust framework based on multiparty session types. Previous session type implementations in Rust are either built upon synchronous and blocking communication and/or are limited to two-party interactions. Crucially, none support the arbitrary ordering of messages for efficiency. Rumpsteak instead targets asynchronous async/await code. Its unique ability is allowing developers to arbitrarily order send/receive messages whilst preserving deadlock-freedom. For this, Rumpsteak incorporates two recent advanced session type theories: (1) k-multiparty compatibility, which globally verifies the safety of a set of participants, and (2) asynchronous multiparty session subtyping, which locally verifies optimisations in the context of a single participant. Specifically, we propose a novel algorithm for asynchronous subtyping that is both sound and decidable. We first talk about Rumpsteak and show the new algorithm. We then talk about our evaluation against other Rust implementations and asynchronous verification tools. We conclude the talk with a demonstration of Rumpsteak.

3.19 A (not very simple) Protocol whose Mechanized Proof is ????

Lenore D. Zuck (University of Illinois – Chicago, US)

In JACM 41(6) Afek et al described a protocol [1], originally conceived by Wang and Zuck [2], then simplified by Afek, whose goal is to transmit an infinite sequence of messages, from a finite alphabet, over bi-directional channels that can reorder and delete messages. While the impossibility of transmitting such a sequence over channels that can also duplicate messages was well known at the time, it was conjectured that reordering and deleting channels suffice to render the problem impossible. The protocol served to refute this conjecture. The original description of the protocol was "flat," and Afek's suggestion to embed it into the "probe" mechanism transformed it into a layer protocol. The top layer implements a FIFO transmission over a lossy channel, for which the well-studied and verified Alternating Bit Protocol suffices. The bottom layer implements a lossy channel over a bi-directional channel that can loss and reorder messages. This protocol, as well as others using on the "probe" mechanism, was never mechanically verified. The talk introduces the protocols and describes the challenges in using existing tools to formally verify probe-based protocols.

References

1 Yehuda Afek, Hagit Attiya, Alan D. Fekete, Michael J. Fischer, Nancy A. Lynch, Yishay Mansour, Da-Wei Wang, Lenore D. Zuck. *Reliable Communication Over Unreliable Channels*. J. ACM 41(6): 1267-1297, 1994

2 Da-Wei Wang, Lenore D. Zuck. *Tight Bounds for the Sequence Transmission Problem*. PODC 1989: 73-83

4 Panel discussions

4.1 A Competition for Distributed Systems Verification?

Swen Jacobs (CISPA – Saarbrücken, DE), Kenneth McMillan (University of Texas – Austin, US), Roopsha Samanta (Purdue University – West Lafayette, US), and Ilya Sergey (National University of Singapore, SG)

Competitions have a long-standing tradition in the fields of verification and automated reasoning. They serve to unify, energize and provide guidance to the research community by establishing a universal format in which verification problems are stated, collecting a library of interesting and challenging benchmark problems, and providing an independent and unbiased platform for the comparison of verification tools. Competitions for different flavours of verification and automated reasoning have been very successful in achieving these goals [4, 1, 2, 3, 5], and were able to draw positive attention to their respective fields in the process.

While competitions have been successful in areas where verification tasks can be fully automated, this seems to be an overly ambitious goal for distributed verification in general: if we consider the verification of realistic distributed algorithms or systems, then the problem is arguably more difficult than in any of the areas where competitions of push-button tools exist. While a restricted scope of the competition would allow us to make it amenable to fully automatic tools, this would make it uninteresting for a large part of the community.

Thus, we concluded that for distributed systems verification an interactive competition would be more suitable, in the style of the VerifyThis competition [6], where a verification team tries to solve verification problems interactively with their tool of choice. This could either be a stand-alone solution, or be separated into multiple tracks, some of which are limited in scope and only only allow fully automated tools. Finally, a third choice would be to try to achieve the benefits of a competition without actually hosting one, i.e., trying to establish a standard format for problems and collecting a library of challenging problems that are offered to the research community, but without a dedicated comparison of tools at specific fixed times.

References

1 Gianpiero Cabodi, Carmelo Loiacono, Marco Palena, Paolo Pasini, Denis Patti, Stefano Quer, Danilo Vendraminetto, Armin Biere, Keijo Heljanko. *Hardware Model Checking Competition 2014: An Analysis and Comparison of Solvers and Benchmarks.* J. Satisf. Boolean Model. Comput. 9(1): 135-172 (2014)

2 Dirk Beyer. *Competition on Software Verification – (SV-COMP).* TACAS 2012: 504-524

3 Clark W. Barrett, Morgan Deters, Leonardo Mendonça de Moura, Albert Oliveras, Aaron Stump. *6 Years of SMT-COMP.* J. Autom. Reason. 50(3): 243-277 (2013)

4 Geoff Sutcliffe. *The CADE ATP System Competition – CASC.* AI Mag. 37(2): 99-101 (2016)

5 Swen Jacobs, Roderick Bloem, Romain Brenguier, Rüdiger Ehlers, Timotheus Hell, Robert Könighofer, Guillermo A. Pérez, Jean-François Raskin, Leonid Ryzhyk, Ocan Sankur, Martina Seidl, Leander Tentrup, Adam Walker. *The first reactive synthesis competition (SYNTCOMP 2014).* Int. J. Softw. Tools Technol. Transf. 19(3): 367-390 (2017)

6 Gidon Ernst, Marieke Huisman, Wojciech Mostowski, Mattias Ulbrich. *VerifyThis – Verification Competition with a Human Factor.* TACAS (3) 2019: 176-195

4.2 Grand Challenges for Distributed Systems Verification

Swen Jacobs (CISPA – Saarbrücken, DE), Kenneth McMillan (University of Texas – Austin, US), Roopsha Samanta (Purdue University – West Lafayette, US), and Ilya Sergey (National University of Singapore, SG)

Grand challenges in science are considered as beneficial in energizing the scientific community and focusing its efforts on meaningful goals. According to their name, they should be sufficiently challenging such that they cannot be completely solved by any research group in a single project, but require a long-term effort and collaboration between different research groups and communities. Solving them should have a major positive impact, not only on the scientific community, but also on society as a whole.

For grand challenges in the area of distributed systems verification, we discussed several ideas. However, we concluded that the effort it would take to design a project that is sufficiently large-scale and challenging, while at the same time allowing a large part of the distributed systems verification community to participate without major obstacles, would go well beyond what could be discussed in this short time frame.

Instead, we found that a particularly promising idea is to take "Verified Secure Routing" (as presented in the talk by Peter Müller) as a grand challenge. The reasons are that this includes challenging aspects and sub-problems for many different research directions, ranging from low-level protocol design over path exploration to information-flow properties. Moreover, the verification tasks are sufficiently difficult to be suitable (at different levels of abstraction) for different flavors of verification, from mechanized interactive proofs to partially or fully automated proof techniques. Finally, a lot of the groundwork in defining the problem and many of its sub-problems has already been done in the large-scale project "verified SCION" at ETH. This project concentrates on mechanized proofs with tight interaction of the protocol and system engineers, and leaves open many details, as well as aspects of automating the verification. This results in a low entry bar for even small research groups to contribute to this grand challenge.

Participants

- Laura Bocchi
University of Kent – Canterbury, GB
- Ahmed Bouajjani
Université Paris Cité, FR
- Andreea Costea
National University of Singapore, SG
- Mike Dodds
Galois – Portland, US
- Constantin Enea
Ecole Polytechnique – Palaiseau, FR
- Javier Esparza
TU München, DE
- Murdoch Jamie Gabbay
Heriot-Watt University – Edinburgh, GB
- Swen Jacobs
CISPA – Saarbrücken, DE
- Gowtham Kaki
University of Colorado – Boulder, US
- Igor Konnov
Informal Systems – Wien, AT
- Burcu Kulahcioglu Ozkan
TU Delft, NL
- Lindsey Kuper
University of California – Santa Cruz, US
- Ori Lahav
Tel Aviv University, IL
- Marijana Lazic
TU München, DE
- Giuliano Losa
Stellar Development Foundation – San Francisco, US
- Rupak Majumdar
MPI-SWS – Kaiserslautern, DE
- Kenneth McMillan
University of Texas – Austin, US
- Peter Müller
ETH Zürich, CH
- Kedar Namjoshi
Nokia Bell Labs – Murray Hill, US
- George Pîrlea
National University of Singapore, SG
- Roopsha Samanta
Purdue University – West Lafayette, US
- Ilya Sergey
National University of Singapore, SG
- Sharon Shoham Buchbinder
Tel Aviv University, IL
- Nobuko Yoshida
University of Oxford, GB
- Lenore D. Zuck
University of Illinois – Chicago, US