International Journal on Software Tools for Technology Transfer (2020) 22:563–564
https://doi.org/10.1007/s10009-020-00580-1

STTT

Special Issue: SPIN 2019

Introduction to the special issue for SPIN 2019

Fabrizio Biondi[1] · Thomas Given-Wilson[2] · Axel Legay[2,3]

Published online: 4 June 2020

Abstract
This short introduction presents the selected papers from SPIN 2019.

Keywords Statistical model checking · Applications · Security · Safety

1 Introduction

This special issue contains the extended and improved versions of the best papers presented at the 26th International Symposium on Model Checking Software, SPIN 2019, held in Beijing, China, July 15–16, 2019.

The original submissions to the conference were addressing topics like: formal verification techniques for automated analysis of software; formal analysis for modeling languages, such as UML/state charts; formal specification languages, temporal logic, design-by-contract; model checking, automated theorem proving, including SAT and SMT; verifying compilers; abstraction and symbolic execution techniques; static analysis and abstract interpretation; combination of verification techniques; modular and compositional verification techniques; verification of timed and probabilistic systems; automated testing using advanced analysis techniques; combination of static and dynamic analyses; derivation of specifications, test cases, or other useful material via formal analysis; case studies of interesting systems or with interesting results; engineering and implementation of software verification and analysis tools; benchmark and comparative studies for formal verification and analysis tools; formal methods education and training; and insightful surveys or historical accounts on topics of relevance to the symposium.

✉ Axel Legay
axel.legay@uclouvain.be

[1] Avast Software, Prague, Czech Republic

[2] Université Catholique de Louvain, Louvain-La-Neuve, Belgium

[3] Aalborg University, Aalborg, Denmark

The symposium attracted 29 submissions that were carefully reviewed by three Program Committee (PC) members. The selection process included further online discussion open to all PC members. As a result, 13 papers were selected for presentation at the symposium and publication in Springer's proceedings. The program also includes one invited talk by Kim G. Larsen (Aalborg University), and one invited talk by Kuldeep S. Meel (National University of Singapore)

From the 13 regular papers, we invited the best ones based on the reviews and the presentations in the event. The authors of four papers accepted to improve and to extend their works. The extended submissions went through a new review process with at least two phases for each paper. The four accepted papers included in this issue addressed all the concerns by the reviewers and by the guest editors.

- In the first paper, "Extracting Safe Thread Schedules from Incomplete Model Checking Results" [3], the authors present a technique that uses the results of incomplete concurrent verification attempts to construct a (fair) scheduler that allows the safe execution of the partially verified concurrent program. This scheduler restricts the execution to schedules that have been proven safe (and prevents executions that were found to be erroneous). The authors evaluated the performance of their technique and show how it can be improved using partial-order reduction. While constraining the scheduler results in a considerable performance penalty in general, results show that in some cases their approach—somewhat surprisingly—even leads to faster executions.
- In the second paper, "Swarm Model Checking on the GPU" [1], the authors present Grapple, a new and powerful framework for explicit-state model checking on

GPUs. Grapple is based on swarm verification (SV), a model-checking technique wherein a collection or swarm of small, memory- and time-bounded verification tests (VTs) are run in parallel to perform state-space exploration. The authors conducted a comprehensive performance analysis of Grapple focused on various design parameters. Results show that Grapple performs favorably compared to the SPIN swarm and a prior non-swarm GPU implementation. Although a recently debuted FPGA swarm is faster, the deployment process to the FPGA is much more complex than Grapple's.

– In the third paper, "Data Structure Invariants, Machine Learning, Korat, Learnability" [4], authors study and compare 10 machine learning algorithms to learn correct data structures for invariants. Results show that most of the machine learning algorithms are good in learning these properties, but they also identified algorithms that are not suitable for learning data structure properties. In this latter case, detailed justification is provided.
– In the fourth paper, "VeriVANca: An Actor-Based Framework for Formal Verification of VANET Applications" [5], the authors propose a framework to provide model checking facilities for the analysis of VANET applications. Authors have extended the Rebeca actor-based modeling language with the inheritance mechanism to support model-specific message passing among vehicles. To illustrate the applicability of this framework, authors modeled and analyzed two warning message dissemination schemes. Reviewing the results of using the model checking technique supports the claim that concurrent behaviors of the system components in VANETs may cause uncertainty which may not be detected by simulation-based techniques. Authors also observed that considering the interleaving of concurrent executions of the system components affects its performance metrics of it.

The proceedings also contains an extended technical paper, "Dependency Graphs with Applications to Verification" [2], from the invited talk by Kim. G. Larsen. Dependency graphs, as introduced more than 20 years ago by Liu and Smolka, are oriented graphs with hyperedges that connect nodes with sets of target nodes in order to represent causal dependencies in the graph. Numerous verification problems can be reduced into the problem of computing a minimum or maximum fixed-point assignment on dependency graphs. In the original definition, assignments link each node with a Boolean value, however, in recent work the assignment domains have been extended to a more general setting, even including infinite domains. In this survey paper, the authors present an overview of the recent results on extensions of dependency graphs in order to deal with verification of quantitative, probabilistic, parameterized and timed systems.

References

1. DeFrancisco, R., Cho, S., Ferdman, M., Smolka, S.A.: Swarm model checking on the GPU. In this issue
2. Enevoldsen, S, Larsen, K.G., Srba, J.: Dependency graphs with applications to verification. In this issue
3. Metzler, P., Suri, N., Weissenbacher, G.: Extracting safe thread schedules from incomplete model checking results. In this issue
4. Usman, M., Wang, W., Wang, K., Yelen, C., Dini, N., Khurshid, S.: Data structure invariants, machine learning, korat, learnability. In this issue
5. Yousefi, F., Khamespanah, E., Gharib, M., Sirjani, M., Movaghar, A.: Verivanca: An actor-based framework for formal verification of warning message dissemination schemes in vanets. In this issue