International Journal on Software Tools for Technology Transfer (2020) 22:103–104
https://doi.org/10.1007/s10009-020-00550-7

INTRODUCTION

Introduction to the Special Issue devoted to SPIN 2018

María del Mar Gallardo[1] · **Pedro Merino**[1]

Published online: 21 January 2020

1 Introduction

This special issue contains the extended and improved versions of the best papers presented at the 25th International Symposium on Model Checking Software, SPIN 2018, held in Málaga, Spain, June 20–22, 2018. The original submissions to the conferences 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.

The symposium attracted 28 submissions, and finally the program [1] consisted of 14 regular papers, one short paper and a demo-tool paper. From the 14 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 were through a new review process with at least two phases for each paper. We selected the reviewers from the SPIN 2018 program committee, ensuring that at least one of them was also in the first review for the conference. The four accepted papers included in this issue addressed all the concerns by the reviewers and by the guest editors. In the context of the original call for papers of the conference, the papers selected for this journal mainly cover the topics of model checking, automated testing, static analysis and formal verification.

In the first paper, "Model-based testing of apps in real network scenarios" [2], authors Almudena Díaz, Laura Panizo and Bruno García present the application of model checking and model-based testing to emulate the behavior of the users of mobile applications. The starting point is the model of the interaction. With this model, the authors use the SPIN model checker to generate realistic interaction sequences driven by some extra-functional property of interest. The sequences are later executed in real phones connecting a testbed that emulates the whole mobile network. This is a practical paper useful for app developers that can find new ideas to debug their apps even with no specific knowledge of the model checking and model-based testing technologies. The tool is offered to experimenters through the project TRIANGLE.

The focus of the paper "Joint Forces for Memory Safety Checking Revisited" [3] by Marek Chalupa, Jan Strejček and Martina Vitovskà is to find usual memory usage errors in C programs, like invalid pointer dereference or memory leaking. The paper describes the whole approach, including the theory, the implementation and some results. The foundation of the analysis method is a combination of static pointer analysis, instrumentation, static program slicing and symbolic execution. The authors argue that they reduce the need for instrumentation thanks to the way of performing pointer analysis. The source code of the implementation to be used in the tool SYMBIOTIC and the benchmarks for the experimental results are included in GitHub as additional material to the readers.

In the paper "IC3 Software Model Checking" [4], the authors Tim Lange, Martin R. Neuhäußer, Thomas Noll and Joost-Pieter Katoen extend the application of the algorithm ic3 (Incremental Construction of Inductive Clauses for Indubitable Correctness) from hardware to software systems. The main contribution is the way to represent the program exe-

✉ Pedro Merino
pedro@lcc.uma.es

María del Mar Gallardo
gallardo@lcc.uma.es

[1] ITIS Software, Universidad de Málaga, Andalucia Tech, Malaga, Spain

cution flow maintaining information on the reachability of the different program locations. The new algorithm IC3CFA generalizes the use of SAT solving to SMT solving, allowing the analysis of infinite-state systems. The paper presents some implementation details of the tool supporting the new algorithms and a comparison with other model checkers.

The last paper "Counting Petri Net Markings From Reduction Equations" [5], by Bernard Berthomieu, Didier Le Botlan and Silvano Dal Zilioa, presents a method to count the reachable markings in the Petri net without explicit enumeration. The method starts building a reduced net from the original one with a number of potential rules or equations that are recorded. Then, the authors can count the reachable markings in the original net using the story of the reductions and the reduced net. The reduction and counting methods have been implemented as the tool *tedd* that is part of the Petri nets analysis toolbox called TINA, publicly available.

References

1. Gallardo, M.M., Merino, P.: Model Checking Software—25th International Symposium, SPIN 2018, Malaga, Spain, 20–22 June 2018, Proceedings. Lecture Notes in Computer Science, vol. 10869. Springer, ISBN 978-3-319-94110-3 (2018)
2. Panizo, L., Díaz, A., García, B.: Model-based testing of apps in real network scenarios. Int. J. Softw. Tools Technol. Transfer (2019). https://doi.org/10.1007/s10009-019-00518-2
3. Chalupa, M., Strejček, J., Vitovskà, M.: Joint forces for memory safety checking revisited. Int. J. Softw. Tools Technol. Transfer (2019). https://doi.org/10.1007/s10009-019-00526-2
4. Lange, T., Neuhäußerr, M.R., Noll, T., Katoen, J.-P.: IC3 software model checking. Int. J. Softw. Tools Technol. Transfer (2019). https://doi.org/10.1007/s10009-019-00547-x
5. Berthomieu, B., Le Botlan, D., Dal Zilioa, S.: Counting Petri net markings from reduction equations. Int. J. Softw. Tools Technol. Transfer (2019). https://doi.org/10.1007/s10009-019-00519-1