Lecture Notes in Computer Science 10869

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

More information about this series at http://www.springer.com/series/7407

María del Mar Gallardo · Pedro Merino (Eds.)

Model Checking Software

25th International Symposium, SPIN 2018
Malaga, Spain, June 20–22, 2018
Proceedings

Springer

Editors
María del Mar Gallardo
University of Málaga
Málaga
Spain

Pedro Merino
University of Málaga
Málaga
Spain

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-319-94110-3 ISBN 978-3-319-94111-0 (eBook)
https://doi.org/10.1007/978-3-319-94111-0

Library of Congress Control Number: 2018947326

LNCS Sublibrary: SL1 – Theoretical Computer Science and General Issues

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG
part of Springer Nature
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This volume contains the proceedings of the 25th International Symposium on Model Checking Software, SPIN 2018, held in Málaga, Spain, June 20–22, 2018. SPIN is a well-recognized periodic event started in 1995 around the model checking tool SPIN. Since 1995, the event has evolved and has been consolidated as a reference symposium in the area of formal methods related to model checking. The previous edition of the SPIN symposium took place in Santa Barbara (USA) with a record number of submissions and participants.

The SPIN 2018 edition requested regular papers, short papers, and tool demos in the following areas: 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, although two of them were rejected by the chairs because they were not within the scope of the symposium. Each of the remaining submissions was carefully reviewed by three Program Committee (PC) members. The selection process included further online discussion open to all PC members. Only the papers with positive global score were considered for acceptance. In addition, within these papers, only those with no objections from the PC members were accepted. As a result, 16 papers were selected for presentation at the symposium and publication in Springer's proceedings. The program consisted of 14 regular papers, one short paper, and a demo-tool paper.

In addition to the accepted papers, the symposium included one invited tutorial by Irina Mariuca Asavoae and Markus Roggenbach entitled "Software Model Checking for Mobile Security, Collusion Detection in K," and three invited talks: "Efficient Runtime Verification of First-Order Temporal Properties" by Klaus Havelund and Doron Peled, "Applying Formal Methods to Advanced Embedded Controllers" by Rémi Delmas, and "Program Verification with Separation Logic" by Radu Iosif.

We would like to thank all the authors that submitted papers, the Steering Committee, the PC, the additional reviewers, the invited speakers, the participants, and the

local organizers for making SPIN 2018 a successful event. We also thank all the sponsors that provided logistics and financial support to make the symposium possible.

May 2018

María del Mar Gallardo
Pedro Merino

Organization

Steering Committee

Dragan Bosnacki (Chair)	Eindhoven University of Technology, The Netherlands
Susanne Graf	Verimag, France
Gerard Holzmann	Nimble Research, USA
Stefan Leue	University of Konstanz, Germany
Neha Rungta	Amazon Web Services, USA
Jaco Van de Pol	University of Twente, The Netherlands
Willem Visser	Stellenbosch University, South Africa

Program Committee

María Alpuente	Technical University of Valencia, Spain
Irina Mariuca Asavoae	Inria, France
Dragan Bosnacki	Eindhoven University of Technology, The Netherlands
Rance Cleaveland	University of Maryland, USA
Stefan Edelkamp	King's College London, UK
Hakan Erdogmus	Carnegie Mellon, USA
María del Mar Gallardo (Chair)	University of Málaga, Spain
Stefania Gnesi	CNR, Italy
Patrice Godefroid	Microsoft Research, USA
Klaus Havelund	NASA/Caltech Jet Propulsion Laboratory, USA
Gerard Holzmann	Nimble Research, USA
Radu Iosif	Verimag, France
Frédéric Lang	Inria, France
Kim Larsen	Aalborg University, Denmark
Stefan Leue	University of Konstanz, Germany
Alberto Lluch Lafuente	Technical University of Denmark, Denmark
Pedro Merino (Chair)	University of Málaga, Spain
Alice Miller	University of Glasgow, UK
Corina Pasareanu	CMU/NASA Ames, USA
Charles Pecheur	Université catholique de Louvain, Belgium
Doron Peled	Bar-Ilan University, Israel
Neha Rungta	Amazon Web Services, USA
Antti Valmari	University of Jyvskyl, Finland
Jaco Van de Pol	University of Twente, The Netherlands
Willem Visser	Stellenbosch University, South Africa
Farn Wang	National Taiwan University, Taiwan

Additional Reviewers

Peter Aldous
Mihail Asavoae
Giovanni Bacci
Georgiana Caltais
Laura Carnevali
Alessandro Fantechi
Grigory Fedyukovich
Martin Koelbl
Florian Lorber
Eric Mercer
Marco Muniz
Julia Sapiña
Andrea Vandin

Organizing Committee

Carlos Canal	University of Málaga, Spain
María del Mar Gallardo	University of Málaga, Spain
Pedro Merino	University of Málaga, Spain
Laura Panizo	University of Málaga, Spain

Sponsors

Abstracts of Invited Papers

Software Model Checking for Mobile Security – Collusion Detection in $\mathbb{K}$

Irina Măriuca Asăvoae[1], Hoang Nga Nguyen[2], and Markus Roggenbach[1]

[1] Swansea University, UK
{I.M.Asavoae,M.Roggenbach}@swansea.ac.uk
[2] Coventry University, UK
Hoang.Nguyen@coventry.ac.uk

Abstract. Mobile devices pose a particular security risk because they hold personal details and have capabilities potentially exploitable for eavesdropping. The Android operating system is designed with a number of built-in security features such as application sandboxing and permission-based access control. Unfortunately, these restrictions can be bypassed, without the user noticing, by colluding apps whose combined permissions allow them to carry out attacks that neither app is able to execute by itself. In this paper, we develop a software model-checking approach within the $\mathbb{K}$-framework that is capable to detect collusion. This involves giving an abstract, formal semantics to Android applications and proving that the applied abstraction principles lead to a finite state space.

Efficient Runtime Verification of First-Order Temporal Properties

Klaus Havelund[1] and Doron Peled[2]

[1] Jet Propulsion Laboratory, California Institute of Technology, USA
[2] Department of Computer Science, Bar Ilan University, Israel

Abstract. Runtime verification allows monitoring the execution of a system against a temporal property, raising an alarm if the property is violated. In this paper we present a theory and system for runtime verification of a first-order past time linear temporal logic. The first-order nature of the logic allows a monitor to reason about events with data elements. While runtime verification of propositional temporal logic requires only a fixed amount of memory, the first-order variant has to deal with a number of data values potentially growing unbounded in the length of the execution trace. This requires special compactness considerations in order to allow checking very long executions. In previous work we presented an efficient use of BDDs for such first-order runtime verification, implemented in the tool DejaVu. We first summarize this previous work. Subsequently, we look at the new problem of dynamically identifying when data observed in the past are no longer needed, allowing to reclaim the data elements used to represent them. We also study the problem of adding relations over data values. Finally, we present parts of the implementation, including a new concept of user defined property macros.

The research performed by the first author was carried out at Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The research performed by the second author was partially funded by Israeli Science Foundation grant 2239/15: “Runtime Measuring and Checking of Cyber Physical Systems”.

A Sample of Formal Verification Research for Embedded Control Software at ONERA

Rémi Delmas, Thomas Loquen, and Pierre Roux

ONERA Centre de Toulouse, 2 av. Édouard Belin, 31055 Toulouse, France
{Rémi Delmas,Thomas Loquen,Pierre Roux}@onera.fr

Abstract. This talk presents a sample of research work conducted by the French Aerospace Lab (ONERA) on tailoring and applying formal methods to advanced embedded controllers, at various phases of the development and verification process, illustrated by industrial projects and collaborations. A first line of work[1], carried out in partnership with Airbus, Dassault and LAAS-CNRS, aims at going beyond simulation for validating advanced hybrid control laws, by leveraging bounded reachability analysis and robustness analysis from the early design phases. This requires to bridge the representation gap existing between hybrid dataflow formalisms used to model control laws (e.g. Simulink, Scade-Hybrid,…), and the automata-based formalisms used by most hybrid model-checkers (e.g. SpaceEx, Flow*, dReach,…) and robustness analysis frameworks. We discuss the steps taken to handle the complexity and size of typical industrial models. A second line of work[1], carried out jointly with academic lab LRI (Paris-Sud, INRIA) and technology provider OcamlPro, addresses the sound combination of SMT-solvers and potentially unsound convex optimization engines to allow proving complex polynomial invariants on advanced control laws implementations. Such implementations are usually obtained by automatic time-discretization and code generation from a hybrid dataflow model. The proposed approach shows a notable performance improvement on controllers of interest with respect to earlier approaches based on interval arithmetic or purely symbolic methods such as cylindrical algebraic decomposition or virtual substitutions. Last, we present research conducted[2] in partnership with Liebherr Aerospace Toulouse and technology provider Systerel on leveraging model-checking techniques for unit-level test case generation for an air management system, taking into account the industrial setting and qualification constraints, following DO-178C and D0-333 guidelines.

Keywords: Hybrid dataflow models • Hybrid automata • Reachability analysis SMT solvers • Convex optimization • SAT solvers • Test case generation

[1] with funding from the French Civil Aviation Authority (DGAC) through the SEFA-IKKY program.

[2] with funding from the CIFRE program of the National Technological Research Agency (ANRT) and the RAPID program of the French Government Defense Procurement and Technology Agency (DGA) (project SATRUCT).

Program Verification with Separation Logic

Radu Iosif

CNRS/VERIMAG/Université Grenoble Alpes, Grenoble, France
Radu.Iosif@univ-grenoble-alpes.fr

Abstract. Separation Logic is a framework for the development of modular program analyses for sequential, inter-procedural and concurrent programs. The first part of the paper introduces Separation Logic first from a historical, then from a program verification perspective. Because program verification eventually boils down to deciding logical queries such as the validity of verification conditions, the second part is dedicated to a survey of decision procedures for Separation Logic, that stem from either SMT, proof theory or automata theory. Incidentally we address issues related to decidability and computational complexity of such problems, in order to expose certain sources of intractability.

Contents