Lecture Notes in Computer Science 8829

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

Stephan Merz Jun Pang (Eds.)

Formal Methods and Software Engineering

16th International Conference
on Formal Engineering Methods, ICFEM 2014
Luxembourg, Luxembourg, November 3-5, 2014
Proceedings

Springer

Volume Editors

Stephan Merz
Inria Nancy - Grand Est
615 rue du Jardin Botanique
54602 Villers-lès-Nancy, France
E-mail: stephan.merz@loria.fr

Jun Pang
Université du Luxembourg
6 rue Richard Coudenhove-Kalergi
1359 Luxembourg, Luxembourg
E-mail: jun.pang@uni.lu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-319-11736-2 e-ISBN 978-3-319-11737-9
DOI 10.1007/978-3-319-11737-9
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014948936

LNCS Sublibrary: SL 2 – Programming and Software Engineering

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The International Conference on Formal Engineering Methods (ICFEM) is a premier conference for research in all areas related to formal engineering methods, such as verification and validation, software engineering, formal specification and modeling, software development, software security, and reliability. Since 1997, ICFEM has been an international forum for researchers and practitioners from academia, industry, and government. It is devoted to presentations and exchanges that advance the state of the art of applying formal methods in practice. Submissions that present combinations of conceptual and methodological aspects with their formal foundation and tool support are particularly encouraged.

In recent years, ICFEM has taken place in Queenstown, New Zealand (2013), Kyoto, Japan (2012), Durham, UK (2011), Shanghai, China (2010), and Rio de Janeiro, Brazil (2009). The 16th edition of ICFEM took place in Luxembourg during 3–5 November, 2014. The Program Committee (PC) received 73 full paper submissions, of which one was withdrawn. Each paper received at least 3 review reports from PC members or external reviewers. On the basis of these reports, each submission was extensively discussed in the virtual meeting of the PC, and the PC decided to accept 28 papers. The proceedings also include the abstracts from the 3 keynote speakers Nikolaj Bjørner, Lionel Briand, and Vincent Danos.

ICFEM 2014 was organized and sponsored by the Interdisciplinary Centre for Security, Reliability and Trust (SnT) at the University of Luxembourg. We are also grateful for the financial support received from the Fonds National de la Recherche (FNR - National Research Fund) in Luxembourg, the Computer Science and Communications Research Unit (CSC) and the Laboratory of Algorithmics, Cryptology and Security (LACS) at University of Luxembourg. We thank the Local Organizing Committee for their hard work in making ICFEM 2014 a successful and exciting event.

The main event was preceded by the seventh International Summer School on Verification Technology, Systems & Applications (VTSA 2014). The third International Workshop on Formal Techniques for Safety-Critical Systems (FTSCS 2014) and the fourth Workshop on SOFL + MSVL were co-located with ICFEM and took place immediately following the conference.

We thank all the PC members for their support, completing quality reviews on time, and being active in discussions during the review process. We thank the external reviewers for their reports that helped the PC decide on which submissions to accept. Most importantly, we thank the authors for submitting

their papers to the conference, and the participants for attending it. Finally, we also thank the EasyChair team for its great conference system and Springer Verlag for the smooth cooperation in the production of this proceedings volume.

July 2014

Stephan Merz
Jun Pang

Organization

Program Committee

Jonathan P. Bowen	Birmingham City University, UK
Michael Butler	University of Southampton, UK
Konstantinos Chatzikokolakis	CNRS & Ecole Polytechnique of Paris, France
Frank De Boer	CWI, The Netherlands
Zhenhua Duan	Xidian University, China
Colin Fidge	Queensland University of Technology, Australia
Stefania Gnesi	ISTI-CNR, Italy
Peter Gorm Larsen	Aarhus University, Denmark
Radu Grosu	Vienna University of Technology, Austria
Ian J. Hayes	University of Queensland, Australia
Michaela Huhn	Technische Universität Clausthal, Germany
Pierre Kelsen	University of Luxembourg, Luxembourg
Steve Kremer	Inria Nancy - Grand Est, France
Jean Krivine	CNRS & Paris Diderot University, France
Xuandong Li	Nanjing University, China
Shang-Wei Lin	National University of Singapore, Singapore
Shaoying Liu	Hosei University, Japan
Yang Liu	Nanyang Technological University, China
Sjouke Mauw	University of Luxembourg, Luxembourg
Dominique Mery	Université de Lorraine, LORIA, France
Stephan Merz	Inria Nancy - Grand Est, France
Mohammadreza Mousavi	Halmstad University, Sweden
Peter Müller	ETH Zürich, Switzerland
Shin Nakajima	National Institute of Informatics, Japan
Jun Pang	University of Luxembourg, Luxembourg
Ion Petre	Åbo Akademi University, Finland
Shengchao Qin	Teesside University, UK
Zongyan Qiu	Peking University, China
Jing Sun	The University of Auckland, New Zealand
Jun Sun	Singapore University of Technology and Design, Singapore
Kenji Taguchi	AIST, Japan
Viktor Vafeiadis	MPI-SWS, Germany
Jaco Van De Pol	University of Twente, The Netherlands
Hai H. Wang	University of Aston, UK
Wang Yi	Uppsala University, Sweden
Huibiao Zhu	East China Normal University, China

Additional Reviewers

Azadbakht, Keyvan
Battle, Nick
Beohar, Harsh
Bessling, Sara
Bezirgiannis, Nikolaos
Bodeveix, Jean-Paul
Boström, Pontus
Bu, Lei
Coleman, Joey
Colley, John
Craciun, Florin
Dghaym, Dana
Dima, Catalin
Dong, Naipeng
Fang, Huixing
Fantechi, Alessandro
Ferrari, Alessio
Gengler, Marc
Gheorghe, Marian
Gratie, Cristian
Gratie, Diana-Elena
Guck, Dennis
Gui, Lin
Hansen, Henri
Huang, Yanhong
Ishikawa, Fuyuki
Islam, Md. Ariful
Ivanov, Sergiu
Jongmans, Sung-Shik T.Q.
Jonker, Hugo
Kalajdzic, Kenan
Kant, Gijs
Kassios, Ioannis
Keiren, Jeroen J.A.
Khakpour, Narges
Kromodimoeljo, Sentot
Laarman, Alfons
Li, Jianwen
Lime, Didier
Lluch Lafuente, Alberto
Ma, Qin
Melnychenko, Oleksandr
Mizera, Andrzej
Mohaqeqi, Morteza
Nguyen, Truong Khanh
Noroozi, Neda
Ouchani, Samir
Petrocchi, Marinella
Petrucci, Laure
Qu, Hongyang
Rezazadeh, Abdolbaghi
Ruijters, Enno
Sanán, David
Selyunin, Konstantin
Singh, Neeraj
Solin, Kim
Song, Songzheng
Spagnolo, Giorgio Oronzo
Strejcek, Jan
Su, Wen
Sulskus, Gintautas
Trujillo, Rolando
van Dijk, Tom
Vanzetto, Hernán
Versari, Cristian
Wang, Ting
Wijs, Anton
Wildman, Luke
Winter, Kirsten
Wu, Xi
Würtz Vinther Jørgensen, Peter
Zhang, Tian
Zhao, Jianhua
Zou, Liang

Abstracts of Invited Talks

SecGuru: Azure Network Verification Using Z3

Nikolaj Bjørner[1] and Karthick Jayaraman[2]

[1] Microsoft Research
nbjorner@microsoft.com
[2] Microsoft Azure
karjay@microsoft.com

This talk describes the use of SMT solving for *Network Verification*. We take as starting point experiences using Z3 in checking network configurations in the Microsoft Azure public cloud infrastructure.

The Azure infrastructure is a prime example of a state-of-the art global and highly complex network infrastructure. It supports a wide range of usage scenarios and security is a principal concern. As a result, there is an urgent need for formal methods tools that provide diagnostic feedback when there are errors and otherwise correctness guarantees.

The Azure architecture enforces network access restrictions using ACLs that are placed on multiple routers and firewalls. Mis-configurations are a dominant source of network outages. The SecGuru tool uses the SMT solver Z3 to check contracts on firewall ACLs. ACLs are checked for containment and equivalence with contracts. SecGuru checks all routers on a continuous basis: each router is checked every 30 minutes against a data-base of contracts. SecGuru relies on checking satisfiability of bit-vector formulas. SecGuru's model extraction algorithm exploits that properties can be captured succinctly as combinations of ranges.

Each Azure data-center is built up around a hierarchy of routers that facilitate high-bandwidth traffic in and out as well as within the data-center. Traffic that leaves and enters the data-center traverses four layers of routers, while traffic within the data-center may traverse only one, two or at most three layers depending on whether the traffic is within a logical partition called a cluster. We describe a set of invariants that capture reachability properties of the Azure architecture. Data-centers are instantiations of this general architecture and we describe how SecGuru is used for checking network invariants on a continuous basis while data-centers are built out and updated.

Scalable Software Testing and Verification through Heuristic Search and Optimization

Lionel C. Briand

SnT Centre for Security, Reliability and Trust, University of Luxembourg
Email: lionel.briand@uni.lu

Testing and verification problems in the software industry come in many different forms, due to significant differences across domains and contexts. But one common challenge is scalability, the capacity to test and verify increasingly large, complex systems. Another concern relates to practicality. Can the inputs required by a given technique be realistically provided by engineers?

This talk reports on 10 years of research tackling verification and testing as a search and optimization problem, often but not always relying on abstractions and models of the system under test. Our observation is that most of the problems we faced could be re-expressed so as to make use of appropriate search and optimization techniques to automate a specific testing or verification strategy. One significant advantage of such an approach is that it often leads to solutions that scale in large problem spaces and that are less demanding in terms of the level of detail and precision required in models and abstractions. Their drawback, as heuristics, is that they are not amenable to proof and need to be thoroughly evaluated by empirical means. However, in the real world of software development, proof is usually not an option, even for smaller and critical systems. In practice, testing and verification is a means to reduce risk as much as possible given available resources and time.

Concrete examples of problems we have addressed and that I will cover in my talk include schedulability analysis, stress/load testing, CPU usage analysis, robustness testing, testing closed-loop dynamic controllers, and SQL Injection testing. Most of these projects have been performed in industrial contexts and solutions were validated on industrial software. There are, however, many other examples in the literature, a growing research trend that has given rise to a new field of study named search-based software testing.

Further information is available in the following selected references:

References

1. Ali, S., et al.: Generating test data from ocl constraints with search techniques. IEEE Transactions on Software Engineering Journal (2013)
2. Matinnejad, R., et al.: Search-based automated testing of continuous controllers: Framework, tool support, and case studies. Information and Software Technology Journal (2014)

3. Briand, L.C., et al.: Using genetic algorithms for early schedulability analysis and stress testing in real-time systems. Genetic Programming and Evolvable Machines Journal (2006)
4. Iqbal, M.Z.Z., et al.: Empirical investigation of search algorithms for environment model-based testing of real-time embedded software. In: ISSTA (2012)
5. Nejati, S., et al.: Identifying optimal trade-offs between cpu time usage and temporal constraints using search. In: ISSTA (2014)
6. Nejati, S., Di Alesio, S., Sabetzadeh, M., Briand, L.: Modeling and analysis of CPU usage in safety-critical embedded systems to support stress testing. In: France, R.B., Kazmeier, J., Breu, R., Atkinson, C. (eds.) MODELS 2012. LNCS, vol. 7590, pp. 759–775. Springer, Heidelberg (2012)

Approximations for Stochastic Graph Rewriting

Vincent Danos[1], Tobias Heindel[1], Ricardo Honorato-Zimmer[1], and Sandro Stucki[2]

[1] School of Informatics, University of Edinburgh, Edinburgh, United Kingdom
[2] Programming Methods Laboratory, EPFL, Lausanne, Switzerland

In this note we present a method to compute approximate descriptions of a class of stochastic systems. For the method to apply, the system must be presented as a Markov chain on a state space consisting in graphs or graph-like objects, and jumps must be described by transformations which follow a finite set of local rules.

The method is a form of static analysis and uses a technique which is reminiscent of theories of critical pairs in term rewriting systems. Its output is a system of coupled ordinary differential equations (ODE) which tracks the mean evolution of the number of (typically small) subgraphs. In some cases, these ODEs form an exact and finite description of these mean numbers. But even when the ODE description is only an approximation, it can often reveal interesting properties of the original system.

The method was first conceived in relation to a special type of graphs, namely the site graphs which form the basis of the Kappa language [3]. Recently, the authors have taken again this method with the goal to extend it to a broader class of objects. In this note, the goal is rather the opposite. We narrow down the construction to consider only simple graphs and invertible rules, to not be distracted by technicalities, and give a simple account. The exposition is mostly informal.

Table of Contents