Lecture Notes in Computer Science 10580

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

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

Dang Van Hung · Deepak Kapur (Eds.)

Theoretical Aspects of Computing – ICTAC 2017

14th International Colloquium
Hanoi, Vietnam, October 23–27, 2017
Proceedings

Springer

Editors

Dang Van Hung
Vietnam National University
Hanoi
Vietnam

Deepak Kapur
University of New Mexico
Albuquerque, NM
USA

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-319-67728-6 ISBN 978-3-319-67729-3 (eBook)
DOI 10.1007/978-3-319-67729-3

Library of Congress Control Number: 2017953430

LNCS Sublibrary: SL1 – Theoretical Computer Science and General Issues

Printed on acid-free paper

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

Preface

This volume contains the papers presented at the 14th International Colloquium on Theoretical Aspects of Computing (ICTAC), held in Hanoi, Vietnam, during October 23–27, 2017.

The International Colloquium on Theoretical Aspects of Computing (ICTAC) constitutes a series of annual conferences/schools, initiated in 2003 by the then United Nations University International Institute for Software Technology, to bring together researchers and provide them with an international venue to present their results, exchange ideas, and engage in discussions. There were two additional goals: (i) promote cooperation between participants from emerging and developed regions, and most importantly, (ii) provide an opportunity for students and young researchers to get exposed to topical research directions in theoretical aspects of computing technologies.

ICTAC 2017 received 40 full-paper submissions coauthored by researchers from 20 different countries. Each submission was reviewed by at least three Program Committee (PC) members with the help of reviewers outside the PC. After two weeks of online discussions, the committee decided to accept 17 papers for presentation at the conference.

We would like to express our gratitude to all the researchers who submitted their work to the symposium. We are particularly thankful to all colleagues who served on the Program Committee, as well as the external reviewers, whose hard work in the review process helped us prepare the conference program. The international diversity of the PC as well as the external reviewers is noteworthy as well: PC members and external reviewers have affiliations with institutes in 22 different countries. Special thanks go to the three invited speakers – Jun Andronick from UNSW, Australia; Joose-Pieter Katoen from RWTH Aachen University, Germany; and Ruston Leino from Microsoft Research, Redmond, USA. The abstracts of the invited talks are included in this volume.

Like previous ICTACs, the 14th ICTAC included four tutorials by Jun Andronick, Joose-Pieter Katoen, Ruston Leino, and Zhiming Liu (Southwest University, China). We thank them for agreeing to offer this valuable service.

A number of colleagues have worked very hard to make this conference a success. We wish to express our thanks to the local organizing committee, Hung Pham Ngoc, Hoang Truong Anh, Hieu Vo Dinh, and many student volunteers. The University of Engineering and Technology of the Vietnam National University, Hanoi, the host of the conference, provided support and facilities for organizing the conference and its tutorials. Finally, we enjoyed institutional and financial support from the National Foundation for Science and Technology Development (NAFOSTED) of Vietnam and the HUMAX VINA Company in Hanoi, Vietnam.

The conference program and proceedings were prepared with the help of EasyChair. We thank Springer for continuing to publish the conference proceedings.

October 2017

Dang Van Hung
Deepak Kapur

Organization

ICTAC 2017 was organized by the University of Engineering and Technology, Vietnam National University (UET-VNU), Hanoi, Vietnam.

Steering Committee

Ana Cavalcanti	University of York, UK
John Fitzgerald	Newcastle University, UK
Martin Leucker	University of Luebeck, Germany
Zhiming Liu	Southwest University, China
Tobias Nipkow	Technical University of Munich, Germany
Augusto Sampaio	Federal University of Pernambuco, Brazil
Natarajan Shankar	SRI International, USA

General Chair

Nguyen Viet Ha	University of Engineering and Technology, VNU, Vietnam

Organizing Committee

Ninh-Thuan Truong (Co-chair)	UET-VNU, Vietnam
Ngoc-Hung Pham (Co-chair)	UET-VNU, Vietnam
Truong Anh Hoang	UET-VNU, Vietnam
Vo Dinh Hieu	UET-VNU, Vietnam
To Van Khanh	UET-VNU, Vietnam
Dang Duc Hanh	UET-VNU, Vietnam
Vu Dieu Huong	UET-VNU, Vietnam

Program Committee

Bernhard K. Aichernig	TU Graz, Austria
Farhad Arbab	CWI and Leiden University, The Netherlands
Ana Cavalcanti	University of York, UK
Wei-Ngan Chin	National University of Singapore, Singapore
Hung Dang-Van (Co-chair)	UET-VNU, Hanoi, Vietnam
Martin Fränzle	Carl von Ossietzky Universität Oldenburg, Germany
Marcelo Frias	Buenos Aires Institute of Technology, Argentina
Dimitar P. Guelev	Bulgarian Academy of Sciences, Bulgaria

Deepak Kapur (Co-chair)	University of New Mexico, USA
Kim Guldstrand Larsen	Aalborg University, Denmark
Martin Leucker	University of Lübeck, Germany
Xuandong Li	Nanjing University, China
Xiaoshan Li	University of Macau, Macao
Zhiming Liu	Southwest University, China
Dominique Mery	Université de Lorraine, LORIA, France
Mohammadreza Mousavi	Halmstad University, Sweden
Thanh-Binh Nguyen	Da Nang University of Technology, Vietnam
Mizuhito Ogawa	JAIST, Japan
Jose Oliveira	Universidade do Minho, Portugal
Catuscia Palamidessi	Inria, France
Minh-Dung Phan	AIT, Thailand
Sanjiva Prasad	Indian Institute of Technology Delhi, India
Thanh-Tho Quan	Hochiminh City University of Technology, Vietnam
António Ravara	Universidade Nova de Lisboa, Portugal
Augusto Sampaio	Federal University of Pernambuco, Brazil
Emil Sekerinski	McMaster University, Canada
Hiroyuki Seki	Nagoya University, Japan
Deepak D'Souza	Indian Institute of Science, Bangalore, India
Hoang Truong-Anh	UET-VNU, Hanoi, Vietnam
Kazunori Ueda	Waseda University, Japan
Farn Wang	National Taiwan University, Taiwan
Jim Woodcock	University of York, UK
Hsu-Chun Yen	National Taiwan University, Taiwan
Naijun Zhan	IoS, Chinese Academy of Sciences, China
Huibiao Zhu	East China Normal University, China
Abdullah Mohd Zin	Universiti Kebangsaan Malaysia, Malaysia

Additional Reviewers

Ana Almeida Matos
Ullas Aparanji
Marius Bozga
Georgiana Caltais
Pablo Castro
Ugo Dal Lago
Duc-Hanh Dang
Kasper Dokter
Bertram Felgenhauer
Raul Fervari
Sebastian Gerwinn
Falk Howar
Huu Hung Huynh
Raghavendra K.R.
Tobias Kapp
Natallia Kokash
Martin Lange
Xinxin Liu
Kamal Lodaya
Ben Moszkowski
Matthias Niewerth
Masahiko Sakai
Vinicius Santos
Torben Scheffel
Karsten Scheibler
Malte Schmitz
Richard Schumi
Xiang Shuangqing

Mani Swaminathan
Martin Tappler
Daniel Thoma
Tinko Tinchev
Ionut Tutu
Mahsa Varshosaz
Hieu Vo
Xuan Tung Vu
Shuling Wang
Karsten Wolf
Zhilin Wu
Xiaoyuan Xie
Yilong Yang
Shoji Yuen
Hengjun Zhao
Jianhua Zhao
Liang Zhao
Quan Zu

Sponsoring Institutions

National Foundation for Science and Technology Development (NAFOSTED) of Vietnam, Hanoi, Vietnam
HUMAX VINA Company in Hanoi, Vietnam
University of Engineering and Technology, Vietnam National University (UET-VNU), Hanoi, Vietnam

Abstract of Invited Talks

From Hoare Logic to Owicki-Gries and Rely-Guarantee for Interruptible eChronos and Multicore seL4 (Extended Abstract)

June Andronick

Data61, CSIRO (formerly NICTA) and UNSW, Sydney, Australia
june.andronick@data61.csiro.au

In this talk we will be exploring the use of foundational proof techniques in the formal verification of real-world operating system (OS) kernels. We will focus on eChronos [2], a small interruptible real-time OS, and seL4 [7, 8], the landmark verified microkernel, currently undergoing verification of its multicore version. Both are deployed in various safety- and security-critical areas, and present challenging complexities due to their performance constraints and concurrency behavior. Foundational techniques have been and are being used for their verification, ranging for standard Hoare logic [5], to concurrency logics like Owicki-Gries [10] and Rely-Guarantee [6]. We will describe their use and combination with theorem proving and automation techniques to achieve impact on large-scale software.

Hoare logic is well known to be the foundation of formal verification for main-stream programs. It is what is taught to university students to prove formally the correctness of programs. Hoare logic can also be the basis of large-scale, real-world software verification, such as the verified seL4 microkernel. seL4 is a very small OS kernel, the core and most critical part of any software system. It provides minimal hardware abstractions and communication mechanisms to applications. seL4 additionally enforces strong access control: applications can be configured to have precise rights to access memory or to communicate, and seL4 guarantees the absence of unauthorised accesses. seL4 has undergone extensive formal verification [7, 8] when running on unicore hardware. The central piece of this verification is the proof of functional correctness: that seL4 source code satisfies its specification. This proof uses Hoare logic at its core, while the top-level theorem is a traditional refinement proof through forward simulation: we show that all behaviors of the source program are contained in the behaviors of the specification. For small programs, Hoare logic can be the central method to prove functional correctness, where the specification is defined as being the description of the state in the postcondition. For larger programs, and in particular for programs where further verification is desired (like seL4's further security proofs), having the specification as a separate standalone artifact saves significant overall effort. In this case a refinement proof links the concrete source code to the abstract specification, and often relies on global invariants to be maintained. In seL4 verification, invariant proofs represent the largest part of the effort [8]. They heavily use

Hoare logic reasoning, combined with important use of automation in the Isabelle/HOL theorem prover [9], both to generate the required invariant statements for each of the hundreds of seL4 functions and to discharge as many as possible without need for human interaction.

Following this verification of a large and complex, but sequential program, we investigated the impact of concurrency in settings where interrupts cannot be avoided (seL4 runs with interrupts mostly disabled), or where running on multiple processors is desired.

Reasoning about interrupt-induced concurrency is motivated by our verification of the eChronos [2] embedded OS. In an eChronos-based system, the kernel runs with interrupts enabled, even during scheduling operations, to be able to satisfy stringent latency requirements. The additional challenge in its concurrency reasoning is that racy access to shared state between the scheduler and interrupt handlers is allowed, and can indeed occur.

The modelling and verification approach we chose for this fine-grained concurrency reasoning is Owicki-Gries [10], the simple extension on Hoare logic with parallel composition and *await* statements for synchronisation. Owicki-Gries provided the low-level of abstraction needed for the high-performance shared-variable system code we were verifying. We could conveniently identify localised Owicki-Gries assertions at the points of the racy accesses, and tune them to enforce the overall correctness invariant of eChronos scheduler. In contrast, the Rely-Guarantee (RG) approach [6] would have required identification of global interference conditions, which was challenging for such racy sharing with no clear interface, unless we made heavier use of auxiliary variables to identify racy sections of code, but this defeats the compositionality of the RG approach, one of its principal purposes. The explosion of verification conditions inherent in the Owicki-Gries approach has been minimized by the controlled nature of the interrupt-induced concurrency, and mitigated by proof-engineering techniques and automation of a modern theorem prover. We were able to develop an abstract model of eChronos scheduling behavior and prove its main scheduling property: that the running task is always the highest-priority runnable task [4, 3]. Our models and proofs are available online [1].

We are currently exploring multicore-induced concurrency for seL4 in a setting where most but not all of the code is running under a big lock. Here we have explored the RG approach, on an abstracted model identifying the allowed interleaving between cores. In this setting, the relies and guarantees can express what shared state the lock is protecting, and what the conditions are under which shared state can be accessed without holding the lock. The main challenge is resource reuse. The kernel runs in privileged mode, and as such has access to everything; it can for instance delete objects on other cores to which critical registers point. This could create a system crash if later on in that core, the kernel code accesses these registers pointing to corrupted memory. Designs to solve this issue include forcing kernel operations on all other cores without holding the lock. The proof that this is sound needs to be expressed via relies and guarantees between cores. We proved, on our abstract model of the multicore seL4-system, that critical registers remain valid at all times.

The main challenge now, for both the eChronos verification and multicore seL4 one, is to transfer the verification down to the source code via refinement.

Acknowledgements. The author would like to thank the people that have worked on the research presented in this paper: Sidney Amani, Maksym Bortin, Gerwin Klein, Corey Lewis, Daniel Matichuk, Carroll Morgan and Christine Rizkallah. The author also thanks Carroll Morgan and Gerwin Klein for their feedback on drafts of this paper.

Parts of the work presented are supported by the Air Force Office of Scientific Research, Asian Office of Aerospace Research and Development (AOARD) and U.S. Army International Technology Center - Pacific under grant FA2386-15-1-4055.

References

1. eChronos model and proofs. https://github.com/echronos/echronos-proofs
2. The eChronos OS. http://echronos.systems
3. Andronick, J., Lewis, C., Matichuk, D., Morgan, C., Rizkallah, C.: Proof of OS scheduling behavior in the presence of interrupt-induced concurrency. In: Blanchette, J.C., Merz, S. (eds.) ITP 2016. LNCS, pp. 52–68. Springer, Cham (2016)
4. Andronick, J., Lewis, C., Morgan, C.: Controlled owicki-gries concurrency: reasoning about the preemptible eChronos embedded operating system. In: van Glabbeek, R.J., Groote J.F., Höfner, P. (eds.) Workshop on Models for Formal Analysis of Real Systems, MARS 2015, pp. 10–24. Suva, Fiji (Nov 2015)
5. Hoare, C.A.R.: An axiomatic basis for computer programming. CACM **12**, 576–580 (1969)
6. Jones, C.B.: Tentative steps towards a development method for interfering programs. ACM Trans. Program. Lang. Syst. **5**(4), 596–619 (1983)
7. Klein, G., Andronick, J., Elphinstone, K., Heiser, G., Cock, D., Derrin, P., Elkaduwe, D., Engelhardt, K., Kolanski, R., Norrish, M., Sewell, T., Tuch, H., Winwood, S.: seL4: Formal verification of an operating-system kernel. CACM **53**(6), 107–115 (2010)
8. Klein, G., Andronick, J., Elphinstone, K., Murray, T., Sewell, T., Kolanski, R., Heiser, G.: Comprehensive formal verification of an OS microkernel. Trans. Comp. Syst. **32**(1), 2:1–2:70 (2014)
9. Nipkow, T., Paulson, L., Wenzel, M.: Isabelle/HOL— A Proof Assistant for Higher-Order Logic. LNCS, vol. 2283, Springer, Heidelberg (2002)
10. Owicki, S., Gries, D.: An axiomatic proof technique for parallel programs. Acta Informatica **6**, 319–340 (1976)

Tweaking the Odds: Parameter Synthesis in Markov Models (Abstract)

Joost-Pieter Katoen[1,2]

[1] RWTH Aachen University, Germany
[2] University of Twente, The Netherlands

Markov decision processes (MDPs) are the prime model in sequential decision making under uncertainty. Their transition probabilities are fixed. Transitions in *parametric* Markov models are equipped with functions (e.g., polynomials or rational functions) over a finite set of parameters x_1 through x_n, say. Instantiating each variable x_i with a value v_i induces a MDP or a Markov chain (MC) if non-determinism is absent. We present recent advances on the *parameter synthesis problem*: for which parameter values — and for MDPs, for which policy — does the instantiated Markov model satisfy a given objective? For objectives such as reachability probabilities and expected costs we consider (1) an *exact* procedure and (2) an *approximative* technique. Both approaches come with a CEGAR-like procedure to obtain a good coverage of the parameter space indicating which parameter regions satisfy the property and which ones do not.

The exact approach first obtains symbolic representations of the synthesis problem at hand. This can be done using e.g., Gaussian elimination or a technique introduced by Daws at ICTAC 2004 [4] that is based on an automata-to-regular expression conversion. These symbolic representations (in fact, rational functions in x_1 through x_n) can be solved using satisfiability-modulo-theory techniques over non-linear real arithmetic [7]. This technique is applicable to parametric MCs only but extendible to conditional reachability objectives too. Using advanced reduction and implementation techniques [5] it is practically applicable to MCs of up to a few million states and 2–3 parameters.

The approximative approach removes parameter dependencies at the expense of adding new parameters and then replaces them by lower and upper bounds [9]. It reduces parameter synthesis to standard model checking of non-parametric Markov models that have one extra degree of non-determinism. Its beauty is the simplicity and applicability to both MCs and MDPs. It is applicable to models of up to about ten million states and 4–5 parameters.

Finally, we treat parameter synthesis for (3) *multiple objectives* for parametric MDPs. Whereas multi-objectivemodel-checking of MDPs can be cast as a linear programming problem [6], its analogue for parametric MDPs results in a non-linear programming (NLP) problem. An approximate solution of this NLP problem can be obtained in polynomial time using geometric programming [3]. This technique is extendible to richer objectives such as weighted combinations of single objectives.

Initial experiments indicate that this approach seems scalable to models with tens of parameters.

Parameter synthesis has abundant applications. These include *model repair* [2, 8] — how to adapt certain probabilities in a Markov model that refutes a given objective such that the tweaked model satisfies it — and finding *minimal recovery times for randomized self-stabilizing algorithms* [1].

Reference

1. Aflaki, S., Volk, M., Bonakdarpour, B., Katoen, J.-P., Storjohann, A.: Automated fine tuning of probabilistic self-stabilizing algorithms. In: SRDS (2017, to be published)
2. Bartocci, E., Grosu, R., Katsaros, P., Ramakrishnan, C. R., Smolka, S.A.: Model repair for probabilistic systems. In: Abdulla, P.A., Leino, K.R.M. (eds.) TACAS 2011. LNCS, vol. 6605, pp. 326–340. Springer, Heidelberg (2011)
3. Cubuktepe, M., Jansen, N., Junges, S., Katoen, J.-P., Papusha, I., Poonawala, H.A., Topcu, U.: Sequential convex programming for the efficient verification of parametric MDPs. In: Legay, A., Margaria, T. (eds.) TACAS 2017. LNCS, vol.10206, pp. 133–150, Springer, Heidelberg (2017)
4. Daws, C.: Symbolic and parametric model checking of discrete-time markov chains. In: Liu, Z., Araki, K. (eds.) ICTAC 2004. LNCS, vol. 3407, pp. 280–294. Springer, Heidelberg (2004)
5. Dehnert, C., Junges, S., Jansen, N., Corzilius, F., Volk, M., Bruintjes, H., Katoen, J.-P., Ábrahám, E.: Prophesy: A probabilistic parameter synthesis tool. In: Kroening, D., Păsăreanu, C. (eds.) CAV 2015. LNCS, vol. 9206, pp. 214–231. Springer, Cham (2015)
6. Etessami, K., Kwiatkowska, M., Vardi, M.Y., Yannakakis, M.: Multi-objective model checking of Markov decision processes. Log. Methods Comput. Sci. **4**(4), (2008)
7. Jansen, N., Corzilius, F., Volk, M., Wimmer, R., Ábrahám, E., Katoen, J.-P., Becker, B.: Accelerating parametric probabilistic verification. In: Norman, G., Sanders, W. (eds.) QEST 2014. LNCS, Vol. 8657, pp. 404–420. Springer, Cham (2014)
8. Pathak, S., Ábrahám, E., Jansen, N., Tacchella, A., Katoen, J.-P.: A greedy approach for the efficient repair of stochastic models. In: Havelund, K., Holzmann, G., Joshi, R. (eds.) NFM 2015. LNCS, vol. 9058, pp. 295–309. Springer, Cham (2015)
9. Quatmann, T., Dehnert, C., Jansen, N., Junges, S., Katoen, J.-P.: Parameter synthesis for Markov models: faster than ever. In: Artho, C., Legay, A., Peled, D. (eds.) ATVA 2016. LNCS, vol. 9938, pp. 50–67. Springer, Cham (2016)

Directions to and for Verified Software

K. Rustan M. Leino[1,2]

[1] Microsoft Research, Redmond
[2] Imperial College London

Abstract. There are many techniques and tools aimed at creating and maintaining reliable software. At one extreme of the reliability spectrum is deductive verification, where software designers write specifications for what the software is supposed to do and where programs are developed together with proofs that show that the specifications are met. The journey of research and development behind deductive verification spans many decades, from early visions of the idea, through criticism and doubt, through the development of automated techniques, to education, to experience in using tools in practice, and to the streamlining of the process.

In this talk, I give a perspective of where this journey of program-verification research has brought us today, give a demo of a state-of-the-art system for writing verified programs [1], and discuss directions for what may be possible in the future.

Reference

1. Leino, K.R.M.: Dafny: an automatic program verifier for functional correctness. In: Clarke, E. M., Voronkov, A. (eds.) LPAR 2010. LNCS, vol. 6355, pp. 348–370. Springer, Heidelberg (2010)

Contents

Logics

Software Components and Concurrency

Automata

SMT Solvers and Algorithms

Security