Lecture Notes in Computer Science **11430**

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/7408

Jonathan P. Bowen · Zhiming Liu ·
Zili Zhang (Eds.)

Engineering Trustworthy Software Systems

4th International School, SETSS 2018
Chongqing, China, April 7–12, 2018
Tutorial Lectures

Springer

Editors
Jonathan P. Bowen
London South Bank University
London, UK

Zhiming Liu
Southwest University
Chongqing, China

Zili Zhang
Southwest University
Chongqing, China

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-030-17600-6 ISBN 978-3-030-17601-3 (eBook)
https://doi.org/10.1007/978-3-030-17601-3

LNCS Sublibrary: SL2 – Programming and Software Engineering

Cover illustration: Academic supervisor tree for Alan Turing. LNCS 11430, p. 213, used with permission.
Photograph on p. xiii: The photograph of the group was taken by Hui Xiang, used with permission.

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

Preface

The 4th School on Engineering Trustworthy Software Systems (SETSS 2018) was held during April 7–12, 2018, at Southwest University, Chongqing, China. It was aimed at PhD and Master students in particular, from around China and elsewhere, as well as being suitable for university researchers and industry software engineers. This volume contains tutorial papers related to a selection of the lecture courses and evening seminars delivered at the school.

SETSS 2018 was organized by the School of Computer and Information Science, in particular the Centre for Research and Innovation in Software Engineering (RISE), at Southwest University, providing lectures on leading-edge research in methods and tools for use in computer system engineering. The school aimed to enable participants to learn about state-of-the-art software engineering methods and technology advances from experts in the field.

The opening session was chaired by Prof. Guoqiang Xiao. A welcome speech was delivered by the Vice President of Southwest University, Prof. Yanqiang Cui, followed by an introductory briefing for SETSS 2018 by Prof. Zhiming Liu. The session finished with a photograph of participants at the school.

The following lecture courses (each consisting of six hour-long lecture sessions, with breaks) were delivered during the school, chaired by Jonathan Bowen, Zhiming Liu, Zhendong Su, and Shmuel Tyazberwicz:

- Mark Utting: An Introduction to Software Verification with Whiley
- Wang Yi: Model-Based Design of Real-Time Systems: From Timed Automata to Di-Graph and Back
- Zhendong Su: Randomized and Systematic Testing of Software
- Lijun Zhang: Omega-Automata Learning Algorithms and Its Application
- Jorge Cuellar: Securing the Future IoT Application
- Nikolaj Bjørner: Programming Constraint Services with Z3

In addition, there were two evening seminars:

- Yu Jiang: Fuzzing Testing in Theory and Practice
- Jonathan P. Bowen: From Alan Turing to Formal Methods

These additional presentations complemented the longer lecture courses.

Courses

An Introduction to Software Verification with Whiley

Lecturer: Dr. Mark Utting, University of the Sunshine Coast, Australia

Biography: Mark Utting is a senior lecturer in ICT at the University of the Sunshine Coast (USC), in Queensland, Australia. Prior to joining USC, he worked at UQ, QUT, and Waikato University in academic positions, and he has also worked in industry, developing next-generation genomics software and manufacturing software. Mark is passionate about designing and engineering good software that solves real-world problems, and has extensive experience with managing software development projects and teams both in academia and industry. He is author of the book *Practical Model-Based Testing: A Tools Approach*, as well as of more than 50 publications on model-based testing, verification techniques for object-oriented and real-time software, and language design for parallel computing.

Overview: This course introduced students to the fundamental ideas of software verification for imperative programming. It covered basic specification techniques, how to use preconditions and postconditions, the relationship between specifications and code, techniques for verifying conditional code, loops, arrays, records, and functions. The course included a series of hands-on verification exercises using the Whiley programming language and its online verification tool.

Model-Based Design of Real-Time Systems: From Timed Automata to Di-Graph and Back

Lecturer: Prof. Wang Yi, Uppsala University, Sweden

Biography: Wang Yi received a PhD in Computer Science from Chalmers University of Technology in 1991. He was appointed Professor of Embedded Systems at Uppsala University, Sweden, in 2000, and Distinguished Professor at North Eastern University, China, in 2007. He is a fellow of the IEEE and member of the Academy of Europe. His interests include embedded systems and formal verification. He was the recipient of the CAV 2013 Award for the development of UPPAAL, Best Paper Awards at RTSS17, RTSS15, RTSS09, ECRTS15, DATE13, and Outstanding Paper Award at ECRTS12. He is a board member of ACM SigBed and Award Committee Chair of the ACM SigBed Caspi Dissertation Award. He is a Steering Committee Co-chair of EMSOFT, and Steering Committee member of ESWEEK, FORMATS, LCTES, and SETTA. He is editor for the journals *ACM Transactions on Embedded Computing and Systems*, *IEEE Embedded Systems Letters*, *IEEE Design and Test*, and the *Journal of Computer Science and Technology*. Recent keynote talks were given at ETAPS15, SIES16, APSEC17, and ICFEM17.

Overview: The first part of my lecture focused on modeling and verification of real-time systems in the framework of timed automata, covering the theoretical foundation, modeling, and specification languages, as well as the central algorithms of UPPAAL, a tool developed jointly by Uppsala and Aalborg University. The second part of my lecture was based on our recent work on real-time scheduling and timing analysis. I presented a new graph-based model for timed systems, that allows us to precisely capture the timing behavior of real-time software and yet keep the analysis problems tractable. For the theoretically intractable cases of interest, we presented a refinement technique, which allows for effective guidance to significantly prune away the global search space and to efficiently verify the desired timing properties in real applications.

Randomized and Systematic Testing of Software

Lecturer: Prof. Zhendong Su, ETH Zürich, Switzerland

Biography: Zhendong Su received his PhD from the University of California, Berkeley, and until recently he was a professor and chancellor's fellow at the University of California Davis, before taking up a professorial position at ETH Zürich, Switzerland. His research focuses on methodologies, techniques, and tools for improving software quality and programming productivity. His work has been recognized with best paper awards from EAPLS, SIGSOFT, OOPSLA and PLDI, CACM Research Highlight, NSF CAREER Award, UC Davis Outstanding Engineering Faculty Award, and industrial research awards (Cisco, Google, IBM, Microsoft, Mozilla). He served as Associate Editor for ACM TOSEM, program (co-)chair SAS (2009), ISSTA (2012), and FSE (2016), and serves on the Steering Committees of ESEC/FSE and ISSTA.

Overview: Random testing (aka fuzzing) has been remarkably successful in finding important software flaws and vulnerabilities. There is also much exciting recent progress in developing more advanced systematic techniques and adapting them to different domains. This set of lectures introduced and highlighted several of these important advances, including EMI and SPE testing for compilers and interpreters, and mathematical execution for solving floating-point constraints and analyzing numerical software. It also discussed key open technical challenges and promising new applications.

Omega-Automata Learning Algorithms and Its Application

Lecturer: Prof. Lijun Zhang, State Key Laboratory of Computer Science, Institute of Software Chinese Academy of Sciences, China

Biography: Lijun Zhang is a research professor at State Key Laboratory of Computer Science, Institute of Software Chinese Academy of Sciences. Before this he was an

associate professor at the Language-Based Technology section, DTU Compute, Technical University of Denmark. He was a postdoctoral researcher at the University of Oxford and obtained gained a diploma degree and a PhD (Dr. Ing.) at Saarland University. His research interests include: probabilistic models, simulation reduction, decision algorithms for probabilistic simulation preorders, abstraction, and model checking. His recent work is in combining automata learning techniques with model checking. He is leading the development of the model checker IscasMC.

Overview: Learning-based automata inference techniques have received significant attention from the community of formal system analysis. In general, the primary applications of automata learning in the community can be categorized into two groups: improving efficiency and scalability of verification and synthesizing abstract system model for further analysis. Most of the results in the literature focus on checking safety properties or synthesizing finite behavior models of systems/programs. On the other hand, Büchi automaton is the standard model for describing liveness properties of distributed systems. Unlike the case for finite automata learning, learning algorithms for Büchi automata are very rarely used in our community. In this talk, we present algorithms to learn a Büchi automaton from a teacher who knows an omega-regular language. The algorithm is based on learning a formalism named family of DFAs (FDFAs) recently proposed by Angluin and Fisman. The main catch is that we use a classification tree structure instead of the standard observation table structure. The worst-case storage space required by our algorithm is quadratically better than the table-based algorithm. We implement the first publicly available library ROLL (Regular Omega Language Learning), which consists of all omega-regular learning algorithms available in the literature and the new algorithms proposed in this paper. Further, with our tool, we demonstrate how our algorithm can be exploited in classic automata operations such as complementation checking and in the model-checking context.

Securing the Future IoT Application

Lecturer: Prof. Dr. Jorge Cuellar, Siemens AG and University of Passau, Germany

Biography: Jorge Cuellar is a principal research scientist at Siemens AG. He was awarded the DI-ST Award for the best technical achievement for his work on modeling of operating systems and transaction managers. He has worked in several topics, including performance analysis, on learning algorithms, hand-writing recognition, formal verification of distributed system design, and security and he has co-authored 50 publications. He has done technical standardization work on privacy and security protocols at the IETF, 3GPP, and the Open Mobile Alliance. He has worked in several EU-funded research projects, mostly on security topics. He regularly serves in Program Committees for international conferences and he has held many short-term visiting teaching positions, in different universities around the world.

Overview: In the near future, computing devices – belonging to different owners with competing expectations and diverse security goals – will be embedded into all sort of

commonplace objects, including smart surfaces or devices in buildings and at home, wearables, city and transportation infrastructure, etc. The IoT promise is that those "things" will talk to each other and will create self-configuring systems. There is a need to negotiate compromises ("contracts") that manage their interactions and interoperate the security policies and functionality goals.

We require a formal language for specifying the possible interactions and contracts and to enforce the agreements reached. We propose to use Petri nets, smart contracts, and a public ledger (like a blockchain or a Merkle tree). The system resembles in some aspects Bitcoins, Etherum, or other cryptocurrencies, but instead of coins, the tokens represent mostly permissions ("authorization tokens") or information. To allow verification, we avoid Turing-complete contracts, but construct smart contracts using Petri nets based on building blocks with cryptographic functionality (secure or fair interactions) or guarded commands.

In this short course, we reviewed how to construct and to use authorization tokens for IoT, how to create workflows as Petri nets, how to define and implement basic cryptographic building blocks, how to use them to create more complex smart contracts, and how to use a public ledger for common information and for resolving disputes.

Programming Constraint Services with Z3

Lecturer: Dr. Nikolaj Bjørner, Microsoft Research, USA

Biography: Nikolaj Bjørner is a principal researcher at Microsoft Research, Redmond, USA, working in the area of automated theorem proving and network verification. His current main line of work with Leonardo de Moura, Lev Nachmanson, and Christoph Wintersteiger is on the state-of-the-art theorem prover Z3, which is used as a foundation of several software engineering tools. Z3 received the 2015 ACM SIGPLAN Software System Award, most influential tool paper in the first 20 years of TACAS in 2014, and the 2017 Skolem Award for the 2007 paper on *Efficient E-matching for SMT Solvers*. Another main line of activity is focused on network verification with colleagues in Azure, Karthick Jayaraman, and academia, George Varghese. Previously, he developed the DFSR, Distributed File System Replication, part of Windows Server since 2005, and before that worked on distributed file sharing systems at a startup, XDegrees, and program synthesis and transformation systems at the Kestrel Institute. He received his Master's and PhD degrees in computer science from Stanford University, and spent the first three years of university at DTU and DIKU in Denmark.

Overview: Many program verification, analysis, testing, and synthesis queries reduce to solving satisfiability of logical formulas. Yet, there are many applications where satisfiability, and optionally a model or a proof, is insufficient. Examples of useful additional information include interpolants, models that satisfy optimality criteria, generating strategies for solving quantified formulas, enumerating and counting solutions. The lectures describe logical services from the point of view of the Satisfiability Modulo Theories solver Z3. We cover their foundations, algorithmics, and ways to put these features to use.

As an overview, we provide a few types of queries below.

Type of Query	Query in symbolic form
Satisfiability	$\varphi \rightsquigarrow$ sat, unsat, timeout
Certificates	$\varphi \rightsquigarrow$ model, proof, unsat core
Interpolation	$\varphi[x, y] \rightarrow I[x] \rightarrow \psi[x, z]$
Optimization	$\max x \mid \varphi$
Consequences	$\varphi \rightarrow \varphi_1 \wedge \ldots \wedge \varphi_n$
Sat subsets	$\psi_1 \wedge \psi_2$, $\psi_1 \wedge \psi_3$
Unsat cores	$\neg(\psi_1 \wedge \psi_2)$, $\neg(\psi_1 \wedge \psi_3)$
Model counting	$\lvert\{x \mid \varphi\}\rvert$
All models	$Ideal(\varphi)$, $M_1 \models \varphi$, $M_2 \models \varphi$, $\ldots$
Model probability	$\ldots$

The first type of query is the most typical query posed to SMT solvers: whether a formula φ is satisfiable and a corresponding yes/no/don't know answer. This conveys some information, but applications typically need to retrieve additional output. At the very least they may need a certificate. An assignment of values to variables for satisfiable formulas, e.g., a model is very commonly used. Dually, proofs or cores for unsatisfiability can be used for unsatisfiability formulas. Other queries include asking to find models that optimize objective values, finding formulas that are consequences, count or enumerate models.

Seminars

Fuzzing Testing in Theory and Practice

Lecturer: Dr. Yu Jiang, Tsinghua University, China

Biography: Yu Jiang received his PhD degree in computer science from Tsinghua University in 2015, worked as a postdoc at the University of Illinois at Urbana-Champaign in 2016, and is currently an assistant professor at Tsinghua University in Beijing, China. His research focuses on safety and security assurance of modern software systems such as deep learning systems and big data systems, and proposed systematic methods for the reliability analysis and testing of those systems, which has been applied in the design and mass production of train control system (MVB/WTB) of CRRC. He has published 40+ papers in international journals (TPDS, TC, TCPS, etc.) and conferences (ICSE, ASE, ICCAD, etc.). He won the China Computer Association Outstanding Doctoral Dissertation Award in 2015, and the Excellent Guide Teacher Award for a national software test competition in 2017.

Abstract: Fuzzing is a widely used software testing technique for bug and vulnerability detection, and the testing performance is greatly affected by the quality of initial seeds and the effectiveness of mutation strategy. In this presentation, we introduced some

basic concepts about fuzzing and then presented SAFL, an efficient fuzzing testing tool augmented with qualified seed generation and efficient coverage-directed mutation. After conducting thoroughly repeated evaluations on real-world program benchmarks against state-of-the-art versions of fuzzing tools, we also presented the obstacles encountered in industrial practice, and how we finally solved these obstacles to detect real-world vulnerabilities. Finally, we described some potential domains where fuzzing can be applied and customized.

From Alan Turing to Formal Methods

Lecturer: Prof. Jonathan P. Bowen, Southwest University, China

Biography: Jonathan Bowen, FBCS FRSA, is Adjunct Professor in the Centre for Research and Innovation in Software Engineering (RISE) at Southwest University, Chongqing, China. He is also Chairman of Museophile Limited (founded in 2002) and Emeritus Professor of Computing at London South Bank University in the UK, where he established and headed the Centre for Applied Formal Methods from 2000. Previously, he worked at Imperial College London, the Oxford University Computing Laboratory, the University of Reading, and Birmingham City University, as well as in industry. He has been a visitor at the United Nations University (Macau) and East China Normal University (Shanghai). His interests have ranged from software engineering, formal methods, safety-critical systems, the Z notation, provably correct systems, rapid prototyping using logic programming, decompilation, hardware compilation, software/hardware co-design, linking semantics, and software testing, to the history of computing, museum informatics, and virtual communities. In 2017, he co-authored *The Turing Guide*, a book on the work of the computing pioneer Alan Turing.

Abstract: Alan Turing (1912–1954) has been increasingly recognized as an important mathematician and philosopher, who despite his short life developed ideas that have led to foundational aspects of computer science and related fields, such as the Turing machine and the Turing test. This seminar talk provided an overview of the diverse aspects related to Turing's remarkable achievements, in the context of the production of a book, *The Turing Guide*, a collected volume of 42 chapters, published by Oxford University Press in 2017. In particular, the talk considered Turing's foundational work with respect to the development of formal methods. Although the story of Turing is partly one of tragedy, with his life cut short while still at the height of his intellectual powers, just short of his 42nd birthday, from a historical viewpoint Turing's contribution to science and even culture has been triumphant.

From the courses and seminars, a record of the school has been distilled in five chapters in this volume as follows:

- David J. Pearce, Mark Utting, and Lindsay Groves: An Introduction to Software Verification with Whiley
- Yong Li, Andrea Turrini, Yu-Fang Chen, and Lijun Zhang: Learning Büchi Automata and Its Applications
- Prabhakaran Kasinathan and Jorge Cuellar: Securing Emergent IoT Applications
- Nikolaj Bjørner, Leonardo de Moura, Lev Nachmanson, and Christoph M. Wintersteiger: Programming Z3
- Jonathan P. Bowen: The Impact of Alan Turing: Formal Methods and Beyond

For further information on SETSS 2018, including lecture material, see:
http://www.swu-rise.net.cn/SETSS2018

SETSS 2018 was supported by IFIP Working Group 2.3 on Programming Methodology. The aim of WG 2.3 is to increase programmers' ability to compose programs, which fits very well with the themes of SETSS.

We would like to thank the lecturers and their co-authors for their professional commitment and effort, the reviewers for their help in improving the papers in this volume, the strong support of Southwest University, and the enthusiastic work of the local organization team, without which SETSS 2018 and these proceedings would not have been possible. Finally, we are grateful for the support of Alfred Hofmann and Anna Kramer of Springer's *Lecture Notes in Computer Science* (LNCS) in the publication of this volume.

February 2019

Jonathan P. Bowen
Zhiming Liu
Zili Zhang

Group photograph at SETSS 2018. Front row, left to right: Zhiping Shi (attendee), Bo Liu (organizer), Weiwei Chen (attendee), Zhiming Liu (organizer), Jonathan Bowen (organizer, lecturer), Yanqiang Cui (Vice President, SWU), Zili Zhang (Dean, SWU), Mark Utting (lecturer), Jorge Cuellar (lecturer), Shmuel Tyazberwicz (organizer), Guogiang Xiao (Dean, SWU), Maoling Zhang (attendee)

Organization

School Chairs

Zili Zhang	Southwest University, China
Guoquiang Xiao	Southwest University, China

Academic Instructors

Jonathan P. Bowen	RISE, Southwest University, China and London South Bank University, UK
Zhiming Liu	RISE, Southwest University, China

Organizing Committee

Bo Liu (Chair)	RISE, Southwest University, China
Rao Dan	RISE, Southwest University, China
Huazhen Liang	RISE, Southwest University, China
Xiao Qin	RISE, Southwest University, China
Shmuel Tyszberowicz	RISE, Southwest University, China and Tel Aviv University, Israel
Qing Wang	RISE, Southwest University, China
Xia Zeng	RISE, Southwest University, China
Tingting Zhang	RISE, Southwest University, China
Yukun Zhang	RISE, Southwest University, China
Hengjun Zhao	RISE, Southwest University, China

School Academic Committee

Michael Butler	University of Southampton, UK
Yixiang Chen	East China Normal University, China
Zhi Jin	Peking University, China
Zhiming Liu	RISE, Southwest University, China
Cong Tian	Xi'Dian University, China
Ji Wang	National University of Defence Science and Technology, China
Yi Wang	Uppsala University, Sweden and Northeast University, China
Jim Woodcock	University of York, UK
Jianhua Zhao	Nanjing University, China

Paper Reviewers

Troy Astarte	Newcastle University, UK
Nikolaj Bjørner	Microsoft Research, USA
Jorge Cuellar	Siemens AG, Germany and University of Passau, Germany
Bo Liu	RISE, Southwest University, China
Andrea Turrini	Institute of Software, China
Shmuel Tyszberowicz	RISE, Southwest University, China and Tel Aviv University, Israel
Mark Utting	University of the Sunshine Coast, Australia
Hengjun Zhao	RISE, Southwest University, China

Contents