Lecture Notes in Computer Science 12971

More information about this subseries at http://www.springer.com/series/7408

Zhe Hou · Vijay Ganesh (Eds.)

Automated Technology for Verification and Analysis

19th International Symposium, ATVA 2021
Gold Coast, QLD, Australia, October 18–22, 2021
Proceedings

Springer

Editors
Zhe Hou
Griffith University
Brisbane, QLD, Australia

Vijay Ganesh
University of Waterloo
Waterloo, ON, Canada

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-030-88884-8 ISBN 978-3-030-88885-5 (eBook)
https://doi.org/10.1007/978-3-030-88885-5

LNCS Sublibrary: SL2 – Programming and Software Engineering

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

Preface

This volume contains the papers presented at the 19th International Symposium on Automated Technology for Verification and Analysis (ATVA 2021). The ATVA series of symposia intends to promote research in theoretical and practical aspects of automated analysis, verification, and synthesis by providing a forum for interaction between the regional and international research communities and industry in related areas.

ATVA 2021 was planned to be hosted on the Gold Coast, Australia, in late October 2021. However, due to the COVID-19 pandemic and travel restrictions, the Steering Committee decided to host the conference virtually during October 18–22, 2021. ATVA 2021 received 75 submissions covering topics related to the theory of and applications in automated verification and analysis techniques. Each paper was reviewed by at least three reviewers, and the Program Committee (PC) accepted 19 regular papers and 4 tool papers, leading to a competitive and attractive scientific program.

This edition of ATVA was blessed by the presence of four prestigious keynote speakers. The first keynote was given by Sir Tony Hoare, a Turing Award and Kyoto Prize laureate. He discussed the link between algebra, geometry, and programming testing and verification using a unified theory. The second keynote speaker, Andrew Chi-Chih Yao from the Tsinghua University, is also a Turing Award and Kyoto Prize laureate. His expertise is in complexity theory and cryptography, and he presented novel ideas about computing and analysis from these angles. Moshe Vardi from the Rice University is another widely recognized top computer scientist and a Godel Prize winner. He talked about linear temporal logic and its applications in analysis and synthesis. Last, but not least, Jha from the University of Wisconsin presented insightful views on security, formal methods and adversarial machine learning. The four talks covered current hot research topics and revealed many new interesting research directions.

After the success of the workshops of the previous edition, we decided to co-host the conference with three workshops in related research areas: Security and Reliability of Machine Learning (SRML 2021), organized by Shiqi Wang, Huan Zhang, Kaidi Xu, and Suman Jana; the Workshop on Hyperproperties: Advances in Theory and Practice (HYPER 2021), organized by Daniel Fremont and Hazem Torfah; and the Workshop on Open Problems in Learning and Verification of Neural Networks 2021, organized by Anna Lukina, Guy Avni, Mirco Giacobbe, and Christian Schilling. All three workshops were hosted virtually on October 18, 2021. These workshops brought in additional participants to ATVA 2021 and helped make it an interesting and successful event. We thank all the workshop organizers for their hard work.

ATVA 2021 would not have been successful without the contributions and involvement of the Program Committee members as well as the external reviewers, who contributed to the review process (with more than 225 reviews) and the selection of the best contributions. This event would not exist if authors and contributors did not

submit their proposals. We thank every person, reviewer, author, PC member and organizing committee member involved in the success of ATVA 2021.

The EasyChair system was set up for the management of ATVA 2021 and supported the submission, review, and volume preparation processes. It proved to be a powerful framework.

Although ATVA 2021 was hosted virtually, the local host and sponsor Griffith University provided tremendous help with the registration and online facilities. The other sponsors, Formal Methods Europe, Springer, and Destination Gold Coast, contributed in different forms to help the conference run smoothly. Many thanks to all the local organisers and sponsors.

We wish to express our special thanks to the General Chair and Steering Committee members, particularly Jing Sun, Farn Wang, Jie-Hong Roland Jiang, and Yu-Fang Chen, for their valuable support.

October 2021

Zhe Hou
Vijay Ganesh

Organization

General Co-chairs

Jin Song Dong	National University of Singapore, Singapore
Jing Sun	University of Auckland, New Zealand

Program Co-chairs

Zhe Hou	Griffith University, Australia
Vijay Ganesh	University of Waterloo, Canada

Steering Committee

Teruo Higashino	Osaka University, Japan
Jie-Hong Roland Jiang	National Taiwan University, Taiwan
Doron A Peled	Bar Ilan University, Israel
Yu-Fang Chen	Institute of Information Science, Academia Sinica, Taiwan
Ichiro Hasuo	National Institute of Informatics, Japan
Yunja Choi	Kyungpook National University, South Korea

Advisory Committee

Insup Lee	University of Pennsylvania, USA
Allen Emerson	The University of Texas at Austin, USA
Hsu-Chun Yen	National Taiwan University, Taiwan
Farn Wang	National Taiwan University, Taiwan

Publicity Co-chairs

Giles Reger	The University of Manchester, UK
Meng Sun	Peking University, China

Workshop Co-chairs

Guy Katz	Hebrew University of Jerusalem, Israel
Rayna Dimitrova	CISPA Helmholtz Center for Information Security, Germany

Program Committee

Erika Abraham	RWTH Aachen University, Germany
Mohamed Faouzi Atig	Uppsala University, Sweden
Christel Baier	TU Dresden, Germany
Stanley Bak	Stony Brook University, USA
Ezio Bartocci	Vienna University of Technology, Austria
Saddek Bensalem	VERIMAG, France
Armin Biere	Johannes Kepler University Linz, Austria
Nikolaj Bjorner	Microsoft, USA
Udi Boker	Interdisciplinary Center (IDC) Herzliya, Israel
Borzoo Bonakdarpour	Michigan State University, USA
Luca Bortolussi	University of Trieste, Italy
Jalil Boudjadar	Aarhus University, Denmark
Martin Brain	University of Oxford, UK
Franck Cassez	ConsenSys and Macquarie University, Australia
Supratik Chakraborty	IIT Bombay, India
Krishnendu Chatterjee	Institute of Science and Technology (IST), UK
Yu-Fang Chen	Academia Sinica, Taiwan
Chih-Hong Cheng	Denso Automotive Deutschland GmbH, Germany
Alessandro Cimatti	Fondazione Bruno Kessler, Italy
Hung Dang Van	Vietnam National University, Vietnam
Tien V. Do	Budapest University of Technology and Economics, Hungary
Alexandre Duret-Lutz	LRDE, EPITA, France
Javier Esparza	Technical University of Munich, Germany
Bernd Finkbeiner	CISPA Helmholtz Center for Information Security, Germany
Pascal Fontaine	Université de Liège, Belgium
Martin Fränzle	Carl von Ossietzky Universität Oldenburg, Germany
Pierre Ganty	IMDEA Software Institute, Spain
Alberto Griggio	Fondazione Bruno Kessler, Italy
Dimitar Guelev	Bulgarian Academy of Sciences, Bulgaria
Keijo Heljanko	University of Helsinki, Finland
Guy Katz	The Hebrew University of Jerusalem, Israel
Siau-Cheng Khoo	National University of Singapore, Singapore
Xuandong Li	Nanjing University, China
Anthony Widjaja Lin	TU Kaiserslautern, Germany
Alexander Nadel	Intel, Israel
Pham Ngoc Hung	Vietnam National University, Vietnam
Aina Niemetz	Stanford University, USA
Tobias Nipkow	Technical University of Munich, Germany
Doron Peled	Bar Ilan University, Israel
Mathias Preiner	Stanford University, USA

Markus Rabe	Google, USA
Andrew Reynolds	University of Iowa, USA
Olli Saarikivi	Aalto University, Finland
Indranil Saha	Indian Institute of Technology Kanpur, India
Sven Schewe	University of Liverpool, UK
Anne-Kathrin Schmuck	Max-Planck-Institute for Software Systems, Germany
Daniel Selsam	Microsoft Research, USA
Gagandeep Singh	VMWare Research and University of Illinois at Urbana-Champaign, USA
Sadegh Soudjani	Newcastle University, UK
Jun Sun	Singapore Management University, Singapore
Sofiene Tahar	Concordia University, Canada
Michael Tautschnig	Queen Mary University of London, UK
Tachio Terauchi	Waseda University, Japan
Aditya Thakur	University of California, Davis, USA
Cesare Tinelli	University of Iowa, USA
Hoang Truong	Vietname National University, Vietname
Bow-Yaw Wang	Academia Sinica, Taiwan
Zhilin Wu	Chinese Academy of Sciences, China

Geometric Theory for Program Testing (Abstract of a Keynote Talk)

Bernhard Möller[1], Tony Hoare[2] and Zhe Hou[3]

[1] Universität Augsburg
[2] University of Cambridge and Honorary Member of Griffith University
[3] Griffith University

Abstract. Formal methods for verification of programs are extended to testing of programs. Their combination is intended to lead to benefits in reliable program development, testing, and evolution. Our geometric theory of testing is intended to serve as the specification of a testing environment, included as the last stage of a toolchain that assists professional programmers, amateurs, and students of Computer Science. The testing environment includes an automated algorithm which locates errors in a test that has been run, and assists in correcting them. It does this by displaying, on a monitor screen, a stick diagram of causal chains in the execution of the program under test. The diagram can then be navigated backwards in the familiar style of a satnav following roads on a map. This will reveal selections of places at which the program should be modified to remove the error.

Summary

The relevant formal methods for testing are due to the pioneers who provided the ideas: Euclid and Descartes for geometry; Carl Adam Petri, whose nets model execution of programs; Noam Chomsky, whose structured method defines the syntax of many programming languages. Their pioneering theories are simplified and adapted to meet current needs of programmers.

A Euclidean diagram is formed by executing a set of constructors, whose feasibility is postulated by axioms and definitions. The geometric features of the diagram (axes, coordinates, points, lines, figures, ...) are labelled by identifiers chosen in drawing the diagram. These identifiers relate the diagram to the proof of a Euclidean proposition, or the text of a program under test.

As an example, we take a structured programming language, with program executions represented by Chomsky's Abstract Syntax Trees. A multiple simultaneous assignment labels the leaves of the tree with atomic commands, and constructors label the branching points. Operators are sequential composition, object class declaration, and concurrent composition of various kinds. Individual operations of the language are defined by specifying the properties of a correct interface between their operands. Errors in arithmetic expressions can be detected by labelling a tree by the value that it produces. Detection of zero divide is then just a matter of calculation. Other errors (eg. deadlock) can be defined by defining a pattern (eg. a cyclic chain of arrows).

This makes it easy to define a new language feature separately by a new constructor. A new language can be defined as the union of its features. A testing tool should be automatically extensible to deal with any combination of features.

Contents

Model Checking

Probabilistic Analysis

Software and Hardware Verification

System Synthesis and Approximation