

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, Lancaster, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Zurich, Switzerland

John C. Mitchell

Stanford University, Stanford, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

TU Dortmund University, Dortmund, Germany

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbrücken, Germany


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


Manuel V. Hermenegildo · Pedro Lopez-Garcia (Eds.)

Logic-Based Program Synthesis and Transformation

26th International Symposium, LOPSTR 2016
Edinburgh, UK, September 6–8, 2016
Revised Selected Papers

Editors

Manuel V. Hermenegildo 
IMDEA Software Institute and Universidad
Politécnica de Madrid
Madrid
Spain

Pedro Lopez-Garcia 
IMDEA Software Institute and CSIC
Madrid
Spain

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-319-63138-7 ISBN 978-3-319-63139-4 (eBook)
DOI 10.1007/978-3-319-63139-4

Library of Congress Control Number: 2017945732

LNCS Sublibrary: SL1 – Theoretical Computer Science and General Issues

© Springer International Publishing AG 2017

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

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 a selection of the papers presented at LOPSTR 2016, the 26th International Symposium on Logic-Based Program Synthesis and Transformation, held during September 6–8, 2016, at the University of Edinburgh, Scotland, UK. It was co-located with two other conferences: PPDP 2016, the 18th International ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming, and SAS 2016, the 23rd Static Analysis Symposium. The co-location of these three related conferences has been shown to be very productive and cross-fertilizing. Previous LOPSTR symposia were held in Siena (2015), Canterbury (2014), Madrid (2013 and 2002), Leuven (2012 and 1997), Odense (2011), Hagenberg (2010), Coimbra (2009), Valencia (2008), Lyngby (2007), Venice (2006 and 1999), London (2005 and 2000), Verona (2004), Uppsala (2003), Paphos (2001), Manchester (1998, 1992, and 1991), Stockholm (1996), Arnhem (1995), Pisa (1994), and Louvain-la-Neuve (1993). More information about the symposium can be found at: <http://www.cliplab.org/Conferences/LOPSTR16/>.

The aim of the LOPSTR series is to stimulate and promote international research and collaboration in logic-based program development. LOPSTR is open to contributions in all aspects of this area, including all stages of the software life cycle and dealing with issues related to both programming-in-the-small and programming-in-the-large. LOPSTR traditionally solicits contributions, in any language paradigm, in the areas of synthesis, specification, transformation, analysis and verification, specialization, testing and certification, composition, program/model manipulation, optimization, transformational techniques in SE, inversion, applications, and tools. LOPSTR has a reputation for being a lively forum that allows presenting and discussing both finished work and work in progress. Formal proceedings are produced only after the symposium so that authors can incorporate the feedback from the conference presentation and discussions.

In response to the call for papers, 45 abstracts were submitted to LOPSTR 2016, of which 38 resulted in full submissions, from 21 different countries. After the first round of reviewing, the Program Committee accepted two full papers for direct inclusion in the formal proceedings, and 18 full papers presented at the symposium were accepted after a post-conference revision and another round of reviewing. Each submission was reviewed by at least three Program Committee members or external reviewers. The paper “A Hiking Trip Through the Orders of Magnitude: Deriving Efficient Generators for Closed Simply-Typed Lambda Terms and Normal Forms” by Paul Tarau won the best paper award, sponsored by Springer. In addition to the 20 contributed papers, this volume includes the abstracts of the talks by our three outstanding invited speakers: Francesco Logozzo (Facebook, USA) and Greg Morrisett (Cornell University, USA), whose talks were shared with PPDP, and Martin Vechev (ETH Zurich, Switzerland), whose talk was shared with SAS.

We would like to thank the Program Committee members, who worked diligently to produce high-quality reviews for the submitted papers, as well as all the external reviewers involved in the paper selection. We are very grateful to the LOPSTR 2016 Organizing Committee composed by James Cheney (local organizer) and Moreno Falaschi for the wonderful job they did in managing the symposium. Many thanks also to Germán Vidal, the Program Committee chair of PPDP 2016, and Xavier Rival, the Program Committee chair of SAS 2016, with whom we often interacted to coordinate the three events. We would also like to thank Andrei Voronkov for his excellent EasyChair system that automates many of the tasks involved in chairing a conference. Special thanks go to the invited speakers and to all the authors who submitted and presented their papers at LOPSTR 2016. Finally, we also thank our sponsors, the School of Informatics of the University of Edinburgh, the IMDEA Software Institute, the European Association for Programming Languages and Systems, the European Association for Theoretical Computer Science, the Association for Logic Programming, and Springer for their cooperation and support in the organization of the symposium.

April 2017

Manuel V. Hermenegildo
Pedro Lopez-Garcia

Organization

Program Chairs

Manuel V. Hermenegildo	IMDEA Software Institute and Universidad Politécnica de Madrid, Spain
Pedro Lopez-Garcia	IMDEA Software Institute and CSIC, Spain

Program Committee

Slim Abdennadher	German University in Cairo, Egypt
Maria Alpuente	Universitat Politècnica de València, Spain
Sergio Antoy	Portland State University, USA
Michael Codish	Ben-Gurion University of the Negev, Israel
Jérôme Feret	Inria/École Normale Supérieure, France
Fabio Fioravanti	University of Chieti-Pescara, Italy
Maurizio Gabbrielli	University of Bologna, Italy
Maria Garcia de La Banda	Monash University, Australia
Robert Glück	University of Copenhagen, Denmark
Miguel Gomez-Zamalloa	Complutense University of Madrid, Spain
Gopal Gupta	University of Texas at Dallas, USA
Patricia Hill	University of Leeds, UK and BUGSENG Srl, Italy
Jacob Howe	City University of London, UK
Viktor Kuncak	École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
Michael Leuschel	University of Düsseldorf, Germany
Heiko Mantel	TU Darmstadt, Germany
Jorge A. Navas	SRI International, USA
Naoki Nishida	Nagoya University, Japan
Catuscia Palamidessi	Inria, France
C.R. Ramakrishnan	Stony Brook University, New York, USA
Vítor Santos-Costa	Universidade do Porto, Portugal
Peter Schneider-Kamp	University of Southern Denmark, Denmark
Hirohisa Seki	Nagoya Institute of Technology, Japan

Organizing Committee

James Cheney	University of Edinburgh, Scotland, UK (Local Organizer)
Moreno Falaschi	University of Siena, Italy

Additional Reviewers

Ayala-Rincón, Mauricio
Ballis, Demis
Biernacki, Dariusz
Ceruelo, Víctor Pablos
Comini, Marco
Cristescu, Ioana
Cruz-Filipe, Luís
De Angelis, Emanuele
Denecker, Marc
Escobar, Santiago
Filinski, Andrzej
Fischer, Sebastian
Fortier, Jérôme
Fuhs, Carsten
Grygiel, Katarzyna
Guo, Hai-Feng
Hamann, Tobias
Isabel, Miguel
Kaarsgaard, Robin

Kafle, Bishoksan
Kawabe, Yoshinobu
Kjellerstrand, Håkan
Krings, Sebastian
Krustev, Dimitur
Kuraj, Ivan
Lanese, Ivan
Lescanne, Pierre
Libby, Steven
Lucanu, Dorel
Lämmel, Ralf
Marple, Kyle
Meo, Maria Chiara
Mera, Edison
Nieva, Susana
Ozono, Tadachika
Pettorossi, Alberto
Proietti, Maurizio
Riesco, Adrián

Salazar, Elmer
Sapiña, Julia
Schneider, David
Serrano, Alejandro
Sharaf, Nada
Stuckey, Peter J.
Stulova, Nataliia
Tarau, Paul
Tasch, Markus
Tiezzi, Francesco
Triska, Markus
Tsushima, Kanae
Urban, Caterina
Villanueva, Alicia
Weber, Alexandra
Zaffanella, Enea
Zaki, Amira

Abstracts of Invited Talks

Challenges in Compiling Coq

Greg Morrisett

Cornell University, Ithaca, USA

jgm19@cornell.edu

Abstract. The Coq proof assistant is increasingly used for constructing verified software, including everything from verified micro-kernels to verified databases. Programmers typically write code in Gallina (the core functional language of Coq) and construct proofs about those Gallina programs. Then, through a process of “extraction”, the Gallina code is translated to either OCaml, Haskell, or Scheme and compiled by a conventional compiler to produce machine code. Unfortunately, this translation often results in inefficient code, and it fails to take advantage of the dependent types and proofs. Furthermore, it is a bit embarrassing that the process is not formally verified.

Working with Andrew Appel’s group at Princeton, we are trying to formalize as much of the process of extraction and compilation as we can, all within Coq. I will talk about both the opportunities this presents, as well as some of the key challenges, including the inability to preserve types through compilation, and the difficulty that axioms present.

Static Analysis for Security at the Facebook Scale

Francesco Logozzo

Facebook, Seattle, USA
logozzo@fb.com

Abstract. The scale and continuous growth of commercial code bases are the greatest challenges for adoption of automated analysis tools in industry. Alas, scale is largely ignored by academic research. We developed a new static analysis tool for security to scale to Facebook scale. It relies on abstract interpretation to focus on the properties that really matter to security engineers and provides fine control on the cost/precision ratio. It was designed from day one for “real world” security and privacy problems at scale. Facebook codebase is huge, and we can analyze it, from scratch in 13 min. This talk will give attendees a peek at some of the secret sauce we use to achieve such amazing performance and precision.

Learning from Programs: Probabilistic Models, Program Analysis and Synthesis

Martin Vechev

ETH Zurich, Zurich, Switzerland
`martin.vechev@inf.ethz.ch`

Abstract. The increased availability of massive codebases (e.g., GitHub) creates an exciting opportunity for new kinds of programming tools based on probabilistic models. Enabled by these models, tomorrow's tools will provide statistically likely solutions to programming tasks difficult or impossible to solve with traditional techniques. An example is our JSNice statistical program de-minification system (<http://jsnice.org>), now used by more than 150,000 users in every country worldwide. In this talk, I will discuss some of the latest developments in this new inter-disciplinary research direction: the theoretical foundations used to build probabilistic programming systems, the practical challenges such systems must address, and the conceptual connections between the areas of statistical learning, static analysis and program synthesis.

Contents

Program Transformation

Partial Evaluation of Order-Sorted Equational Programs Modulo Axioms. . . .	3
<i>María Alpuente, Angel Cuenca-Ortega, Santiago Escobar, and José Meseguer</i>	
A Formal, Resource Consumption-Preserving Translation of Actors to Haskell	21
<i>Elvira Albert, Nikolaos Bezirgiannis, Frank de Boer, and Enrique Martin-Martin</i>	
Verification of Time-Aware Business Processes Using Constrained Horn Clauses	38
<i>Emanuele De Angelis, Fabio Fioravanti, Maria Chiara Meo, Alberto Pettorossi, and Maurizio Proietti</i>	

Constraint Programming

MiniZinc with Strings	59
<i>Roberto Amadini, Pierre Flener, Justin Pearson, Joseph D. Scott, Peter J. Stuckey, and Guido Tack</i>	
Slicing Concurrent Constraint Programs	76
<i>Moreno Falaschi, Maurizio Gabbrielli, Carlos Olarte, and Catuscia Palamidessi</i>	

Compilation and Optimization

A New Functional-Logic Compiler for Curry: SPRITE	97
<i>Sergio Antoy and Andy Jost</i>	
Ipopt: A Rule Optimization Tool for Answer Set Programming.	114
<i>Manuel Bichler, Michael Morak, and Stefan Woltran</i>	
Symbolic Execution and Thresholding for Efficiently Tuning Fuzzy Logic Programs	131
<i>Ginés Moreno, Jaime Penabad, José A. Riaza, and Germán Vidal</i>	

Analysis and Verification

Hierarchical Shape Abstraction for Analysis of Free List Memory Allocators	151
<i>Bin Fang and Mihaela Sighireanu</i>	
A Productivity Checker for Logic Programming	168
<i>Ekaterina Komendantskaya, Patricia Johann, and Martin Schmidt</i>	
Symbolic Abstract Contract Synthesis in a Rewriting Framework	187
<i>Maria Alpuente, Daniel Pardo, and Alicia Villanueva</i>	

Testing

On the Completeness of Selective Unification in Concolic Testing of Logic Programs	205
<i>Fred Mesnard, Étienne Payet, and Germán Vidal</i>	
CurryCheck: Checking Properties of Curry Programs.	222
<i>Michael Hanus</i>	
A Hiking Trip Through the Orders of Magnitude: Deriving Efficient Generators for Closed Simply-Typed Lambda Terms and Normal Forms	240
<i>Paul Tarau</i>	

Semantics and Model Checking

A Reversible Semantics for Erlang	259
<i>Naoki Nishida, Adrián Palacios, and Germán Vidal</i>	
Scaling Bounded Model Checking by Transforming Programs with Arrays. . . .	275
<i>Anushri Jana, Uday P. Khedker, Advaita Datar, R. Venkatesh, and Niyas C.</i>	
Intuitionistic Logic Programming for SQL	293
<i>Fernando Sáenz-Pérez</i>	

Types, Unification, and Logic

Coinductive Soundness of Corecursive Type Class Resolution	311
<i>František Farka, Ekaterina Komendantskaya, and Kevin Hammond</i>	
Nominal Unification of Higher Order Expressions with Recursive Let	328
<i>Manfred Schmidt-Schauß, Temur Kutsia, Jordi Levy, and Mateu Villaret</i>	
Automata Theory Approach to Predicate Intuitionistic Logic	345
<i>Maciej Zielenkiewicz and Aleksy Schubert</i>	

Author Index	361
-------------------------------	-----