Science of Computer Programming 182 (2019) 86–87

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Preface

Fundamentals of Software Engineering (extended versions of selected papers of FSEN 2017)

This special issue contains the extended versions of the selected papers presented at the seventh IPM International Conference on Fundamentals of Software Engineering (FSEN), Tehran, Iran, April 26–28, 2017. This event was organized by the School of Computer Science at the Institute for Studies in Fundamental Sciences (IPM) in Iran, in cooperation with the ACM SIGSOFT and IFIP WG2.2.

FSEN attracts submissions concerning all aspects of formal methods in software engineering, in particular those related to advancing the application of formal methods in the software industry and promoting their integration with practical engineering techniques. For FSEN 2017, we received 49 submissions from 27 countries out of which we have accepted 16 regular papers for publication in the conference proceedings and presentation at the conference. From these papers. After a rigorous review process, we have accepted the following five extended papers for this special issue.

The first article entitled 'Abstract Machines for Open Call-by-Value' by Beniamino Accattoli and Giulio Guerrieri, builds on the theory of the call-by-values that relies on weak evaluation and closed terms, which are in turn natural hypotheses in the study of programming languages. However, in order to model proof assistants, strong evaluation and open terms are required. In this article, the authors focus on open call-by-value, which is the intermediate setting of weak evaluation with (possibly) open terms. They provide a theory of abstract and efficient machines for the fireball calculus, the simplest presentation of open call-by-value. The machine implements the fireball calculus with a bilinear overhead. The authors examine how different optimizations impact on the complexity of the overhead.

The second article entitled 'Behavioral Model Identification and Classification of Multi-component Systems' by Zeynab Sabahi-Kaviani and Fatemeh Ghassemi focuses on learning model from the execution traces. The authors extend the passive automata learning by considering the behavior of the depending components in addition to the observed behaviors. The proposed approach assumes that the models of applications can be abstractly defined in terms of how they execute their depending components. The generated models are then improved to cover unobserved behaviors as well. The learned models are used to distinguish the executions of applications in an interleaved execution trace of different systems. The authors utilize runtime verification techniques to enhance the performance of the matching process for a trace. The applicability of the approach is shown by some real-world applications.

The third article entitled 'Translating Active Objects into Colored Petri Nets for Communication Analysis' by Anastasia Gkolfi, Crystal Chang Din, Einar Broch Johnsen, Lars Michael Kristensen, Martin Steffen, and Ingrid Chieh Yu, addresses the problem of communication deadlock and livelock for the active object language ABS. The authors propose to translate the formal semantics of ABS into colored Petri nets, which provides a basic model of concurrency, causality and synchronization, and has been used to analyze communication patterns and deadlock. The authors show that the translation from ABS to CPN is sound. They demonstrate how to analyze communication deadlocks and livelocks for active objects in ABS using the implementation of this net in CPN Tools.

The fourth article entitled 'A Formal Model for Multi Software Product Lines' by Ferruccio Damiani, Michael Lienhardt, and Luca Paolini, proposes a formal model of Multi Software Product Lines. A Multi Software Product Line (MPL) is a set of interdependent Software Product Lines (SPLs) that are typically managed and developed in a decentralized fashion. The authors address the implementation problem of multi software product line by proposing extensions of existing modular Delta-Oriented Programming (DOP) approach for implementing software product lines to accommodate compositional analyses. The extensions are presented by means of a core calculus for delta-oriented MPLs of Java programs.

The last article entitled 'Bisimilarity of Open Terms in Stream GSOS' by Filippo Bonchi, Tom van Bussel, Matias David Lee, and Jurriaan Rot, addresses the problem of open terms in stream GSOS. Stream GSOS is a specification format for operations and calculi on streams. Bisimilarity provides canonical proof technique for equivalence of closed terms in such

https://doi.org/10.1016/j.scico.2019.07.004

specifications. The authors obtain a notion of bisimilarity over open terms that coincides with behavioral equivalence of all closed instances and provides a concrete proof technique for equivalence of open terms.

We would like to thank the authors and the reviewers of all the papers. We also thank the editorial team of the journal, specially the Editor-in-Chief, Mohammadreza Mousavi, and the Editorial Assistant, Bas van Vlijmen, for their constant help and support.

Mehdi Dastani *
Utrecht University, the Netherlands
E-mail address: M.M.Dastani@uu.nl

Marjan Sirjani
Mälardalen University, Sweden
Reykjavik University, Iceland
E-mail address: marjan.sirjani@mdh.se

Available online 26 July 2019

* Corresponding author.