Noname manuscript No.
(will be inserted by the editor)

Special Issue: Software Verification and Testing

Mohammad Reza Mousavi · Jun Pang

Received: date / Accepted: date

Welcome to the special issue dedicated to the selected extended papers of the Software Verification and Testing (SVT 2012) track held at the 27th ACM Symposium on Applied Computing (ACM SAC 2012). SVT 2012 attracted 43 submissions of which 12 papers were selected for presentation and publication in the proceedings of ACM SAC 2012. From these 12 papers, six were invited to the special issue and after several rigorous rounds of reviews (by at least three expert reviewers), five papers were accepted for inclusion in this special issue. This means that less than 12 percent of our initial submissions, after several rounds of review and revision, made their way into the special issue before you.

The SVT track brings together academic researchers and industry R&D expertise to discuss new results in formal verification and testing, as well as development of technologies to improve the usability of formal methods in software engineering. The SVT track's topics of interest include:

- tools and techniques for verification of large scale software systems,
- real world applications and case studies applying software verification,
- static and run-time analysis,
- abstract interpretation,
- model checking,
- theorem proving,
- refinement and correct by construction development,
- model-based testing,
- verification-based testing,
- run-time verification,
- symbolic execution and partial evaluation,
- analysis methods for dependable systems,
- software certification and proof carrying code.

For this special issue, the following papers were accepted after careful rounds of review for their significance, novelty and soundness:

1. Affeldt presents an approach to the construction of a library of formally verified low-level arithmetic functions. To achieve his research goal, Affeldt first introduces a formalisation of data structures for signed multi-precision arithmetic in low-level programs. He uses this formalisation to verify the implementation of several primitive arithmetic functions using Separation Logic. A concrete implementation of the binary extended GCD algorithm is used to illustrate the overall approach.
2. Dasgupta, Karkare and Reddy propose a field sensitive shape analysis technique to infer shapes of heap structures. They use field-based boolean variables and field-sensitive path matrices to maintain necessary information in order to infer the shapes of pointer variables in terms of boolean equations. With an implementation of their technique, the authors have evaluated their work on standard benchmarks and shown that the results are more precise than an existing field-insensitive analysis.
3. Perrone, Debois and Hildebrandt present a model checker – the BigMC tool for bigraphical reactive systems. They introduce the syntax of BigMC, and

M. Mousavi
Center for Research on Embedded Systems
Halmstad University, Sweden
E-mail: m.r.mousavi@hh.se

J. Pang (✉)
Faculty of Science, Technology and Communication
University of Luxembourg, Luxembourg
E-mail: jun.pang@uni.lu
Phone: +352 4666445625
Fax: +352 4666445500

exemplify its use with two examples. Then they give a tractable heuristic to approximate interference between reaction rules, and prove the analysis to be safe. Based on the proposed heuristic, an algorithm for checking properties expressed in terms of matching is developed.

4. Salaün et al. discuss their experience and draw conclusions from their effort in the fast few years on applying formal verification to cloud applications: complex distributed applications composed of multiple software components running on separate virtual machines. In their study, they mainly use the Lotus-NT process algebra to specify the applications and use the CADP verification toolbox to conduct formal verification.
5. Siddiqui and Khurshid present a novel approach in their paper to increase the efficiency of symbolic execution for systematic testing of object-oriented programs. Their main contribution lies in applying symbolic execution in different stages, rather than the traditional all-at-once approach. They have evaluated the proposed approach through extensive experiments and proved it to substantially save in testing effort, when compared to a state-of-the-art implementation of symbolic execution.

We would like to thank all the members of the program committee of SVT 2012 and anonymous reviewers of the special issue for their great work during the two review processes, the authors for extending and submitting their high-quality papers to the special issue and the editor-in-chief and the editorial assistants of Innovations in Systems and Software Engineering for their help and support during this intensive process.

In the meanwhile, SVT 2013 was held in Coimbra, Portugal, continuing the successful tradition of this scientific event, and its best papers have been invited to a forthcoming special issue of the Science of Computer Programming journal.

We hope that you will enjoy reading this special issue.