Acta Informatica (2017) 54:543–544
DOI 10.1007/s00236-017-0299-0

EDITORIAL

Special issue: Synthesis and SYNT 2014

Krishnendu Chatterjee[1] · Rüdiger Ehlers[2]

Published online: 12 April 2017

The ubiquity of computation in modern machines and devices imposes a need to assert the correctness of their behavior. Especially in the case of safety-critical systems, their designers need to take measures that enforce their safe operation. Formal methods has emerged as a research field that addresses this challenge: by rigorously proving that all system executions adhere to their specifications, the correctness of an implementation under concern can be assured. To achieve this goal, a plethora of techniques are nowadays available, all of which are optimized for different system types and application domains.

But formal methods do not stop at the idea to verify a system to be correct after it has been constructed. Already in the early years of computer science, the desire to automate the system engineering process to the greatest possible extent was formulated. For reactive systems, this is commonly referred to as Church's problem: given a specification over some set of propositions, and a partitioning of the propositions into outputs that the system to be designed can set, and inputs that it cannot, the question is to algorithmically determine if there exists a suitable implementation, and to compute one whenever there exists one. Starting from the definition of the problem in the 60s and the first principal solutions to the problem by Büchi, Clarke, Emerson, Rabin, Pnueli, and Rosner, the field of reactive synthesis made substantial progress on increasing the scalability of the synthesis process.

Meanwhile, the research area of software synthesis emerged from the desire to automate parts of the modern software development process. The focus in this case is slightly different than in reactive synthesis: rather than synthesizing non-terminating designs, most software synthesis methods aim at the automatic derivation of moderately sized terminating code fragments. The focus here is to help the software designer with writing the most difficult parts of a program, and only requiring a formal specification of such parts. Again, a multitude of

✉ Krishnendu Chatterjee
krish.chat@ist.ac.at

Rüdiger Ehlers
rehlers@uni-bremen.de

1 IST Austria, Klosterneuburg, Austria

2 University of Bremen, Bremen, Germany

techniques have emerged to tackle this problem, which base on different algorithmic ideas and target different application domains.

Despite the different origins of the reactive and software synthesis communities, their interaction intensified in the last few years. Both of them have started to hold annual competitions, namely the SYNTCOMP and SyGuS-Comp competitions, and despite their principal differences, a good number of techniques have been shown to be useful in both areas, such as inductive learning and SAT/SMT solving. Since software and reactive synthesis also have some problems in common that appear in their practical application, such as how to write good specifications, it makes sense to support the interaction of these two communities in a joint workshop.

This volume of Acta Informatica contains selected papers from the *Third workshop on Synthesis* (SYNT), which was held on the 23rd–24th July 2014 in Vienna, Austria, along with further contributions. The SYNT workshop was part of the Vienna Summer of Logic 2014 and was associated with the 26th International Conference on Computer-Aided Verification (CAV), which was also part of the Vienna Summer of Logic. Apart from contributed presentations, the workshop featured detailed presentations of the SyGuS and SYNTCOMP competition results and provided an open platform for scientific exchange on all aspects of synthesis, including algorithms and tools for software and reactive synthesis, specification languages, complexity and impossibility results, case studies of software or hardware synthesis, and connections between verification and synthesis. The workshop was rounded off by invited talks by Bernd Finkbeiner, Viktor Kuncak, Leonid Ryzhyk, and Ashish Tiwari.

Krishnendu Chatterjee and Rüdiger Ehlers
Editors of Acta Informatica Special Issue on Synthesis and SYNT 2014 organizers (along with Susmit Jha)