Interpolant Automata
(Invited Talk)

Andreas Podelski

University of Freiburg, Germany

Abstract. We will cover the principles underlying a recent approach to the verification of different classes of correctness properties (safety, termination) for different classes of programs (sequential, recursive, concurrent). The approach is based on the notion of interpolant automata. A Floyd-Hoare correctness proof of the correctness of a trace (i.e., a sequence of statements) consists of a sequence of assertions, the *interpolants* of the trace. The sequence can be constructed, e.g., by static analysis or by an SMT solver with interpolant generation. We use the interpolants as the states of an automaton which has a transition $\varphi \xrightarrow{a} \varphi'$ if the Hoare triple $\{\varphi\}a\{\varphi'\}$ is valid. The resulting *interpolant automaton* recognizes a language over the alphabet of statements. The language is a set of *correct traces*, i.e., traces that obey the given correctness specification of a given program. The program is proven correct if the regular language of its program traces is contained in a union of interpolant automata. The new proof method consists of accumulating interpolant automata until the inclusion holds. Checking the inclusion is comparable to finite-model checking (the finite model defines the set of program traces, the property defines the set of correct traces). Interpolant automata are a modular, succinct, and program-independent presentation of a correctness argument.

The talk is based on joint work with Matthias Heizmann and Jochen Hoenicke and on joint work with Azadeh Farzan and Zachary Kincaid.

S. Chakraborty and M. Mukund (Eds.): ATVA 2012, LNCS 7561, p. 26, 2012.