# Algorithmic Verification of Systems Software Using SMT Solvers

Shaz Qadeer

Microsoft Research, Redmond

**Abstract.** Program verification is an undecidable problem; all program verifiers must make a tradeoff between precision and scalability. Over the past decade, a variety of scalable program analysis tools have been developed. These tools, based primarily on techniques such as type systems and dataflow analysis, scale to large and realistic programs. However, to achieve scalability they sacrifice precision, resulting in a significant number of false error reports and adversely affecting the usability of the tool.

In this talk, I will present a different approach to program verification realized in the HAVOC verifier for low-level systems software. HAVOC works directly on the operational semantics of C programs based on a physical model of memory that allows precise modeling of pointer arithmetic and other unsafe operations prevalent in low-level software. To achieve scalability, HAVOC performs modular verification using contracts in an expressive assertion language that includes propositional logic, arithmetic, and quantified type and data-structure invariants. The assertion logic is closed under weakest precondition, thereby guaranteeing precise verification for loop-free and call-free code fragments. To reduce the effort of writing contracts, HAVOC provides a mechanism to infer them automatically. It allows the user to populate the code with candidate contracts and then searches efficiently through the candidate set for a subset of consistent contracts.

The expressive contract language in HAVOC has two important benefits. First, it allows the documentation and verification of properties and invariants specific to a particular software system. Second, it allows a user to systematically achieve the ideal of precise verification (with no false alarms) by interacting with the verifier and providing key contracts that could not be inferred automatically.

HAVOC has been implemented using the Boogie verification-condition generator and the Z3 solver for Satisfiability-Modulo-Theories. I will describe the design and implementation of HAVOC and our experience applying it to verify typestate assertions on medium-sized device drivers with zero false alarms. I will conclude with a discussion of remaining challenges and directions for future work.

J. Palsberg and Z. Su (Eds.): SAS 2009, LNCS 5673, p. 2, 2009.