Symbol Elimination in Program Analysis

(Invited Abstract)

Laura Kovács
TU Vienna
Austria
lkovacs@complang.tuwien.ac.at

Extended Abstract.

Automatic understanding of the intended meaning of computer programs is a very hard problem, requiring intelligence and reasoning. In this abstract we discuss a new method for program analysis, called symbol elimination which was introduced in [1], [2]. Symbol elimination uses first-order theorem proving techniques to automatically discover non-trivial program properties, such as loop invariants and loop bounds. Moreover, symbol elimination can be used as an alternative to interpolation for software verification.

This abstract is divided in two parts.

In the first part of the abstract we discuss how symbol elimination can be used for quantified invariant generation. Our method explores the power of a first-order theorem prover and allows one to generate first-order invariants containing alternations of quantifiers. To this end, we first deploy symbolic computation methods to generate polynomial invariants of the scalar loop variables [3]. Next, we make use of so-called update predicates over unbounded data structures, such as arrays. An update predicate for an array expresses updates made to the array. We observe that many properties of update predicates can be extracted automatically from the loop description and loop properties obtained by other methods such as a simple analysis of counters occurring in the loop, recurrence solving and quantifier elimination over loop variables. The first-order information extracted from the loop description can use auxiliary symbols, such as symbols denoting update predicates or loop counters. After having collected the first-order information, we run a saturation theorem prover to eliminate the auxiliary symbols and obtain loop invariants expressed as first-order formulas. Symbol elimination is thus the main ingredient in generating quantified loop invariants by a first-order theorem prover [1], [4]. Moreover, by a slight modification of our program analysis framework, we are also able to derive tight bounds on the number of loop iterations, as presented in [5]. To this end, we deploy pattern-based recurrence solving in conjunction with satisfiability modulo theory reasoning. The obtained loop bounds can further be used in the worst-case execution time analysis of programs.

In the second part of the abstract, we detail how interpolants can be automatically constructed from symbol-eliminating proofs of a special structure. In our approach to interpolation we restrict ourselves to so-called local proof. Local proofs are used in a context when some (predicate and/or function) symbols are declared to have colors. In local proofs every inference can use symbols of at most one color, as a consequence, every term or atomic formula used in such proofs can use symbols of at most one color. For extracting interpolants from local proofs we use the algorithm presented in [2]. The interpolants generated by our method are boolean combinations of conclusions of symbol eliminating inferences of a local first-order resolution proof. The obtained interpolants can further be used in bounded model checking, for the verification of safety properties of programs.

***Keywords*-program verification, loop invariants, loop bounds, interpolants, theorem proving, symbolic computation**

Acknowledgment

The work described in this talk is based on joint work with a number of authors, including Krystof Hoder and Andrei Voronkov (The University of Manchester), and Jens Knoop and Jakob Zwirchmayr (TU Vienna).

The author acknowledges funding from the FWF Hertha Firnberg Research grant (T425-N23) and the FWF National Research Network RiSE (S11410-N23).

References

[1] L. Kovács and A. Voronkov, "Finding Loop Invariants for Programs over Arrays Using a Theorem Prover," in *Proc. of FASE*, ser. LNCS, vol. 5503, 2009, pp. 470–485.

[2] L. Kovacs and A. Voronkov, "Interpolation and Symbol Elimination," in *Proc. of CADE*, ser. LNCS, vol. 5663, 2009, pp. 199–213.

[3] L. Kovacs, "Reasoning Algebraically About P-Solvable Loops," in *Proc. of TACAS*, ser. LNCS, vol. 4963, 2008, pp. 249–264.

[4] K. Hoder, L. Kovács, and A. Voronkov, "Case Studies on Invariant Generation Using a Saturation Theorem Prover," in *Proc. of MICAI*, ser. LNCS, vol. 7094, 2011, pp. 1–15.

[5] J. Knoop, L. Kovács, and J. Zwirchmayr, "Symbolic Loop Bound Computation for WCET Analysis," in *Proc. of PSI*, ser. LNCS, vol. 7162, 2011, to appear.