Polynomial Invariant Generation for Multi-Path Loops

Extended Abstract for Presentation Contribution

Andreas Humenberger, Maximilian Jaroschek, and Laura Kovács*

Technische Universität Wien
Institut für Informationssysteme 184
Favoritenstraße 9-11
Vienna A-1040, Austria
`ahumenbe@forsyte.at`
`maximilian@mjaroschek.com`
`lkovacs@forsyte.at`

Abstract. Computing loop invariants, that is properties which hold after any number of loop iterations, is a major task in program analysis. In this paper we study nested loops and loops with conditionals. In particular, we generalize a known invariant generating procedure for P-solvable multi-path loops to a wider class of admissible programs, so called extended P-solvable loops.

For reasoning about safety properties of software systems with non-trivial program flow, code blocks like while-loops and recursive functions, whose number of execution iterations is unknown during compile time, usually require the extraction of properties independent of the number of iterations. Most often, this additional information comes in the form of invariants, that is relations among the loop variables which hold before and after arbitrarily many executions of the loop body. In this work we are particularly interested in invariants which are expressible as polynomial equations in the loop variables. A polynomial invariant is required to equate to zero when evaluated at the values of the loop variables after any number of loop iterations. As the set of all polynomial invariants is not computable when conditions in a while loop of the form $\mathit{while}(\mathit{condition})\{\dots\}$ are considered [5], one usually moves to an abstraction $\mathit{while}()\{\dots\}$ with an undetermined loop condition.

In [3] a new approach was proposed for generating the set of all polynomial invariants of so-called *P-solvable loops*, a class of while-loops where the values of the loop variables are determined by C-finite sequences. In this procedure, the loop variables are modeled as linear recurrence equations and their closed form

* All authors are supported by the ERC Starting Grant 2014 SYMCAR 639270. Furthermore, we acknowledge funding from the Wallenberg Academy Fellowship 2014 TheProSE, the Swedish VR grant GenPro D0497701, and the Austrian FWF research project RiSE S11409-N23. We also acknowledge support from the FWF project W1255-N23.

solutions are computed. Then, by adding new variables representing exponential terms, the set of all algebraic relations among the exponential terms, the loop counter, and the loop variables is determined. This set forms an ideal, which allows the use of Gröbner basis computations to eliminate the newly added variables for exponential terms as well as the loop counter to obtain the ideal of all polynomial loop invariants.

Based on the observation that the values of the variables after a loop execution are the initial values of a succeeding loop, the approach of [3] was extended to multi-path loops in [4]. In this context, multi-path loops are nested loops with conditional branches. Every loop containing nested loops and conditionals can be transformed into a loop containing nested loops only and is then treated as a non-deterministic program (see [2]). More precisely, a program with loop bodies / conditional branches $L_1, L_2, \dots, L_n$ is interpreted as the non-deterministic program $(L_1^*; L_2^*; \dots; L_n^*)^*$, where L^* denotes the execution of program L arbitrarily often and $L_1; L_2$ is the sequential execution of L_1 and L_2.

A new class of loops, called *extended P-solvable*, was identified in [1] subsuming P-solvable loops. A loop with assignments only, with a loop counter n and program variables $v_1, \dots, v_t$ is called *extended P-solvable* if each of its recursively changed variables determines a sequence of the form

$$v_i(n) = \sum_{k \in \mathbb{Z}^\ell} p_{i,k}(n, \theta_1^n, \dots, \theta_s^n)((n+\zeta_1)^{\underline{n}})^{k_1} \cdots ((n+\zeta_\ell)^{\underline{n}})^{k_\ell} \tag{1}$$

where, for a computable field $\mathbb{K}$ of characteristic zero, $p_{i,k}$ is a polynomial in $\mathbb{K}(x)[y_1, \dots, y_s]$, θ_i, ζ_j are elements in $\mathbb{K}$ for $i = 1, \dots, s$, $j = 1, \dots, \ell$, and $\theta_i \neq \theta_j$ and $\zeta_i - \zeta_j \notin \mathbb{Z}$ for $i \neq j$. Furthermore, $r(n)^{\underline{n}}$ denotes the falling factorial for $r(n) \in \mathbb{K}(n)$. Sequences of the form (1) are Hadamard products of C-finite and hypergeometric sequences. Hence, reasoning about C-finite and hypergeometric sequences is employed to determine the ideal of polynomial invariants.

So far, only extended P-solvable single-path loops were considered. Based on the same observation as for P-solvable loops, we propose an algorithm for computing the set of all polynomial invariants of extended P-solvable multi-path loops.

In this regard we show that for two given sequences $u(m)$ and $v(n)$ of the form (1), substituting $v(n)$ for the initial value $u(0)$ of $u(m)$ yields again a sequence of the form (1). This closure property ensures that we can *merge* multiple loops into one single loop and apply the analogous reasoning as in the single-path case. The result of this process is the set of all polynomial invariants of sequentially composed loops $L_1^*; \dots; L_n^*$.

Let $\mathcal{I}_m(L_1, \dots, L_n)$ denote the invariant ideal of a composition

$$(L_1^*; \dots; L_n^*)^q; L_1^*; \dots; L_r^*$$

where q is the quotient and r the remainder of m divided by n, and P^q denotes q sequential executions of a non-deterministic program P. Now consider a loop L containing n inner loops $L_1, \dots, L_n$. In order to derive the invariant ideal for L we compute the invariant ideal $\mathcal{I}_m(L_1, \dots, L_n)$ for $m = 1, 2, \dots$ until a fixed-point is reached. Proving that such a fixed-point always exists is subject to current investigations.

The techniques and methods described in this paper are implemented in the open source Mathematica package ALIGATOR[1] which is available for download at

`https://ahumenberger.github.io/aligator/`

We would like to emphasize that the results presented in this work are not final and subject to ongoing research.

References

1. Humenberger, A., Jaroschek, M., Kovács, L.: Automated Generation of Non-Linear Loop Invariants Utilizing Hypergeometric Sequences. In: International Symposium on Symbolic and Algebraic Computation, ISSAC 2017 (2017), `https://arxiv.org/abs/1705.02863`
2. Kovács, L.: Automated Invariant Generation by Algebraic Techniques for Imperative Program Verification in Theorema. Ph.D. thesis, RISC, Johannes Kepler University Linz (October 2007)
3. Kovács, L.: Reasoning Algebraically About P-Solvable Loops. In: Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings. pp. 249–264 (2008)
4. Kovács, L.: A Complete Invariant Generation Approach for P-solvable Loops. In: Perspectives of Systems Informatics, 7th International Andrei Ershov Memorial Conference, PSI 2009, Novosibirsk, Russia, June 15-19, 2009. Revised Papers. pp. 242–256 (2009)
5. Müller-Olm, M., Seidl, H.: A Note on Karr's Algorithm. In: Automata, Languages and Programming: 31st International Colloquium, ICALP 2004, Turku, Finland, July 12-16, 2004. Proceedings. pp. 1016–1028 (2004)

[1] The unpublished results mentioned in this extended abstract are not yet implemented, but will be in the upcoming weeks.