Generating Loop Invariants via Polynomial Interpolation

Marc Moreno Mazza and Rong Xiao
University of Western Ontario
Ontario Research Center for Computer Algebra

The problem

A loop invariant is a condition of the loop variable that is always true at the entry point of the loop. We are interested in computing polynomial equations which are loop invariants for the following type of loops in programs:

while C do
  if C then 
    X := A(X);
  else
    Y := B(Y);
  end if
end while

Notations and assumptions:
- loop variables \( X, x_1, \ldots, x_k \) are real values in (practice non-finite) scalar variables;
- all conditions consist of polynomial constraints in \( X \); given a condition \( C \) in \( X \), denote by \( Z(C) \) all points in \( \mathbb{R}^n \) satisfying \( C \);
- \( C_0 \) denotes the initial condition;
- for \( t = 1, \ldots, m \), we denote by \( A_t \) a polynomial in \( \mathbb{Q}[X] \) and \( A_t(X) \) the corresponding map induced by \( A_t \);
- \( C_1, C_2, \ldots, C_m \) are pairwise-exclusive; each implies \( C_0 \).

Given a loop \( L \), it is easy to deduce that \( \{ p \in \mathbb{Q}[X] \mid \text{p is odd an invariant of } L \} \)
forms a polynomial ideal in \( \mathbb{Q}[X] \), which is called the invariant ideal of \( L \).

The proposed method

The proposed approach aims at computing all polynomial invariants up to a given total degree \( d \). It is based on polynomial interpolation and it is rather straightforward.

1. Compute the polynomial \( P \) of degree \( d \) which has \( S \) as zero, and \( x \) as a variable taking non-negative integer values. Let \( a_1, \ldots, a_k \) be independent variables. Then each polynomial in \( \mathbb{Q}[a_1, \ldots, a_k] \) is called a polynomial expression in \( x \) and \( \mathbb{Q}[a_1, \ldots, a_k] \).

2. Let \( A := \{ a_1, \ldots, a_k \} \subseteq \mathbb{Q}^n \) and \( \text{assoc} \), \( \text{mult} \) a variable \( \text{basis} \), the multiplicative relation ideal \((\text{MRI})\) of \( x \) is the ideal in \( \mathbb{Q}[a_1, \ldots, a_k] \) generated by

3. Example 1. Consider the following simple infinite loop:
   \( x := 1; y := 1; \text{while } \text{true do } x := x + 1; y := y + 1; \text{end do;} \)
   It is easy to deduce that the trajectory of the loop \( \mathbb{Q}[x, y] = \langle x+y \rangle = \ldots \), \( y \) is a loop invariant of the loop.

4. Example 2. Let \( A := \{ a_1, \ldots, a_k \} \subseteq \mathbb{Q}^n \) and \( \text{assoc} \), \( \text{mult} \) a variable \( \text{basis} \), the multiplicative relation ideal \((\text{MRI})\) of \( x \) is the ideal in \( \mathbb{Q}[a_1, \ldots, a_k] \) generated by

5. Example 3. The MRI of \( A := \langle 1, 2, 3 \rangle \) is associated with \( y_1, y_2, y_3 \).

Notations: \( \text{MRI} \) is a P-probable invariant relation defining a sequence \( \mathbb{Q}[x, y] \) with variable \( x_1, x_2, \ldots, x_k \). Let \( Z(\mathbb{Q}[x_1, x_2, \ldots, x_k]) \) be the invariant ideal of \( R \). Let \( M \)
be the MRI of non-zero eigenvalues of \( R \). Denoted by \( d(x) \) of \( \text{MRI} \).

Theorem. Suppose \( M \) is an MRI of degree \( d(x) \). Assume the total degree of polynomial expressions of \( R \) is estimated to be no more than \( d(x) \). Then we have

Moreover, if the degree of the variable is 0, then we have

The method is simple, but 3 non-trivial issues have to be handled:
A. A reasonable degree must be supplied. In the next section, we shall estimate the degree bound as well the dimension of the invariant ideals for certain loops.
B. A general criterion to check whether or not a condition is invariant must be developed. A criterion is proposed at the end of this paper.
C. The size of sample points might grow dramatically, direct implementation may not be efficient in practice. In our implementation, modular techniques are used to compute the interpolated polynomials.

Proposition. A general criterion consisting of polynomial constraints, if \( Z(I) \subseteq Z(I(\text{m})) \)
holds and if for each branch, the relation
\( M(\text{m}) \subseteq Z(I(\text{m})) \)
holds, then \( I \) is a loop invariant.

The criterion in the above proposition can be easily implemented by set-theoretical operation of semi-algebraic sets, see SemiAlgebraicSetTools of RegularChains package in MAPLE.

Conclusion

Though not complete, the proposed method is quite efficient in practice, and applies better situations than some other methods (e.g. FP and SE). Our degree and dimension estimates can be used to justify the completeness of our output as well as to supply a reasonable degree bound in other methods which also need a degree bound (e.g. CPI)

Implementation and benchmarks

The proposed method has been implemented in MAPLE, which is the command Equational-LoopInvariants().

The ProgramAnalysis package will contain functionality to: automatically generate invariants, verify specifications and verify termination for while loops optimize for while loops better data locality, etc.

> with(ProgramAnalysis):
> (loop)
> (\{ x := 1; y := 1; \text{while } \text{true do } x := x + 1; y := y + 1; \text{end do; } \}
> loop)
> (\text{end loop})
> (\text{end while loop})