Finding Loop Invariants Based on Wu's Characteristic Set Method

Yong Cao and Qing-Xin Zhu
Institute of Computer Science and Engineering,
University of Electronic Science and Technology, China

Abstract: Loop invariants are important parts in program verification and proof. Correspondingly, techniques for automatically checking and finding invariants have been studied for many years. In present study, an approach using Wu's characteristic set method for automatically finding polynomial invariants of imperative programs is presented. Present method is based on the algebraic theory of polynomial set over polynomial rings, which have wider application domain. We implement this method with the computer algebra tools MMP. The application of the method is demonstrated on a few examples. Compared with other polynomial algebraic approaches, our method is more efficient through experiments.

Key words: Wu's method, loop invariants, characteristic set, computer algebra

INTRODUCTION

There has recently been a surge of interest in research on automatically finding loop invariants of imperative programs. Program verification based on Floyd-Hoare's inductive assertion method, using preconditions, post-conditions and loop invariants, was considered a major research problem in the seventies, leading to the development of many program verification systems. Karl first showed an arithmetic algorithm for finding invariant linear equalities at any program point of a procedure in literature (Rodriguez-Carbonell and Kapur, 2004a, b). Recently, Muller-Olm and Seidl (2004) proposed an interprocedural method for computing invariant polynomial of bounded degree in program. Due to the advance of computer algebra and there appears to be a revival of research activities relating to mechanically finding loop invariants, especially using abstract interpretations, quantifier elimination and the associated widening operators (Muller-Olm and Seidl, 2002, 2004; Rodriguez-Carbonell and Kapur, 2004a, b).

Wu's method is an algorithm of representing the zeros of a set of polynomials with the characteristic sets, which is invented by Chinese famous mathematician Wu Wen-Tsun. As an important method and tool, Wu's method performs mechanical geometry theorem proving better than other ways in symbolic computation and has aroused attention to symbolic computation once more. It mainly calculates a characteristic set of a finite set of polynomials and Wu Wen-Tsun successfully applied it to geometry theorem proving. Wu's work inspired the study of more efficient characteristic set algorithms and he becomes a leader in mathematics mechanization in China (Mao and Wu, 2005).

An invariant of a loop is hypothesized as a parameterized formula. Parameters in the invariant are discovered by generating constraints on the parameters by ensuring that the formula is indeed preserved by the execution path corresponding to every basic cycle of the loop. Constraints on parameters appearing in a parameterized formula hypothesized as an invariant can be successively generated by considering every basic cycle through the program location (typically the beginning of the loop) to which the invariant is associated.

The other methods for finding loop invariants are generally based on Grobner bases which are the theory of ideals over polynomial rings. Before we use the methods based on Grobner bases, we must prove that the methods are used over ideals (Sankaranarayanan et al., 2004). It is not necessarily for us to use Wu's characteristic set method because it is algebraic theory of polynomial set over polynomial rings. Wu's characteristic set method can be used to find loop invariants and we implement the approach with the computer algebra tools MMP (http://www.mmrc.iss.ac.cn/mmp). Compared with other approaches, our method is more efficient through experiments.

WU'S CHARACTERISTIC SET METHOD

Most approaches to geometric reasoning are formulated in terms of synthetic geometry which is a logical system expressed using properties of points, lines.
angles, etc., considered as primitive elements. In contrast, Wu's method is algebraic. Hypotheses (or given conditions) and conclusions (conjectures) in a geometry problem are represented as polynomial equations. Geometry theorem can be proved using a fairly simple and elegant algebraic method such as Wu's method. A geometry statement of the form a finite set of hypotheses (or given conditions) implying a conclusion is considered. Hypotheses (or given conditions) are polynomial equations expressing geometric relations and the conclusion is also a polynomial equation stating a geometric relation to be derived; a subset of variables corresponding to coordinates of points that can be arbitrarily chosen in a geometry statement are viewed as parameters. Unlike the methods of Tarski, Seidenberg, Monk, Collins and Ben-Or et al., variables in Wu's method range over an algebraically closed field.

**Definition 1:** Given a polynomial:

\[ f = L_1(x_1, ..., x_n)x_1^\alpha + L_1(x_1, ..., x_n)x_1^\beta + \ldots + L_n(x_1, ..., x_{n}) \]  

Its initial polynomial, \( \text{ln}(f) \) is defined to be the polynomial \( L_1(x_1, ..., x_{n}) \).

**Theorem 1:** For any two polynomials \( f \) and \( g \), there exist a non-negative number \( \alpha \) and polynomials \( q \) and \( r \), such that:

\[ \text{ln}(f)^\alpha g = qf - r \]  

which is called \( f \) pseudo-divides \( g \) and \( r \) is called pseudo-remainder of \( f \) pseudo-dividing \( g \).

In geometric proving we algebraize the proposition (conclusion or conjectures) which need to be proved and the given conditions (or hypotheses) firstly. Suppose that the algebraized conclusion is \( p \) and the algebraized given conditions (or hypotheses) are \( S = \{ f_1 = 0, f_2 = 0, \ldots, f_\ell = 0 \} \). Then we set \( p \) as goal and set \( S = \{ f_1 = 0, f_2 = 0, \ldots, f_\ell = 0 \} \) as constraint conditions. From the hypothesis (constraint condition) polynomials we compute a set of polynomials in a triangular form called Wu's characteristic set \( B = (g_1 = 0, ..., g_n = 0) \) which every polynomial in \( S \) is a linear combination of polynomials in \( B \), namely \( f_\ell \in S \) and

\[ \text{ln}(g_1)^\alpha \ldots \text{ln}(g_n)^\alpha f_\ell = \sum_{i=1}^{n} h_i g_i \]

Next we check whether the conclusion polynomial \( p \) pseudo-divides to 0 using the polynomials in the characteristic set, namely we use every element of the characteristic set pseudo-divide the conclusion \( p \) to obtain pseudo-remainder \( r \) (prim is abbreviation of pseudo-remainder).

\[ r = \text{prim}(p; g_1) \]
\[ r = \text{prim}(r_1, g_1) \]
\[ \vdots \]
\[ r = \text{prim}(r_\ell, g_1) \]

and we know there should be:

\[ \text{ln}(g_1)^\alpha \ldots \text{ln}(g_n)^\alpha p = \sum_{i=1}^{n} h_i g_i + r \]

If \( r = 0 \) and \( \text{ln}(g_1)^\alpha \ldots \text{ln}(g_n)^\alpha = 0 \), then we can draw a conclusion \( p = 0 \). Characteristic sets are computed using pseudo-division by introducing a total ordering on dependent variables. The inequations \( \text{ln}(g_1)^\alpha \ldots \text{ln}(g_n)^\alpha = 0 \), \( \text{ln}(g_1)^\alpha \ldots \text{ln}(g_n)^\alpha = 0 \), have been called nondegenerate conditions or subsidiary conditions in the literature (Mao and Wu, 2005).

**Definition 2:** We use \( \text{Zero}(S) \) to denote the zero set of \( S \):

\[ \text{Zero}(S) = \left\{ (a_1, a_2, ..., a_n) \in \mathbb{K}^n : \left( \forall p \in S \right) \left[ p(a_1, a_2, ..., a_n) = 0 \right] \right\} \]

**Theorem 2:** Let \( B \) be a characteristic set of \( S \). Then:

\[ \text{Zero}(S) = \left( \text{Zero}(B) \setminus \text{Zero}(\text{ln}(B)) \right) \cup \bigcup_{i=1}^{\ell} \left( \text{Zero}(S) \setminus \text{ln}(g_i) \right) \]

**Theorem 3:** Given each \( p_i \in B \), we calculate the corresponding characteristic set by using Wu's method. Let \( \{ B_i : 1 \leq i \leq u \} \) be all the characteristic sets obtained that include \( B \) and have

\[ \text{Zero}(P) = \bigcup_{i=1}^{u} \left( \text{Zero}(B_i) \setminus \text{Zero}(\text{ln}(B_i)) \right) \]

**FINDING LOOP INVARIANTS BASED ON WU'S CHARACTERISTIC SET METHOD**

An invariant at any program point is a formula relating program variable such that whenever control passes through the point, the formula is true. A path in a program is assumed to be a sequence of assignments, possibly interspersed with Boolean tests (which is assumed to evaluate to true) arising due to conditional statements and loops. For a given loop \( L \), let \( p_1, \ldots, p_m \) be all possible basic cycles, consisting of Boolean tests and assignment statements, each of which starts at the beginning of the body of the loop and ends at the same point. For a formula \( f(x) \) to serve as a loop invariant, it should be the case that the Hoare triple \( \{ f(x) \} \ p_i \ \{ f(x) \} \) where \( p_i \) stands for the code fragment.
corresponding to the path \( p_i \), and \( \bar{x} \) is the tuple of all of the variables in the program.

Loops are supposed to have form as follow:

\[
\text{While } E(x) \text{ do} \nonumber \\
\text{path 1: } \langle p_1, \ldots, p_m \rangle \nonumber \\
\text{path 2: } \langle p_1, \ldots, p_m \rangle \nonumber \\
\vdots \nonumber \\
\text{path m: } \langle p_1, \ldots, p_m \rangle \nonumber \\
\text{End} \tag{7}
\]

where, \( p_i = 0 \) is a assignment statement (\( x_i = c_i(x) \Leftrightarrow x_i - c_i(x) = 0 \), where \( c_i \) is an expression) which is polynomial generally and \( \bar{x} = (x_1, x_2, \ldots, x_n) \) denotes the tuple of program variables and \( E \) are Boolean expressions.

Suppose that polynomials \( p \) and \( q \). Let \( r \) is pseudoremainder of \( p \) pseudo-dividing \( q \). \( \exists y \in \mathbb{R} \) s.t. \( y \neq 0 \), \( p(y) = 0 \Leftrightarrow q(y) = 0 \). When \( p = 0 \) and \( q = 0 \) we can also obtain \( r = 0 \). Therefore we have \( \text{Zero}(r) = \text{Zero}(q) \land \text{Zero}(p) \Leftrightarrow \text{Zero}(q) \) under nondegenerate conditions. Similarly if \( r \) is pseudoremainder of polynomial \( p \) pseudo-dividing polynomial set \( Q = (q_1, q_2, \ldots, q_n) \) then there should be:

\[
\text{Zero}(r) = \text{Zero}(p \land c_i^{m_i} \land q) = \text{Zero}(p) \land \text{Zero}(c_i^{m_i} \land q)
\]

under nondegenerate conditions.

Invariant polynomial \( I(\bar{x}) = 0 \) should satisfies \( E(\bar{x}) = 0 \) and \( p_i = 0 \) (\( i = 1, 2, \ldots, m; j = 1, 2, \ldots, k_i \)), namely:

\[
\text{Zero}(I(\bar{x})) \subset \nonumber \\
\text{Zero}(E(\bar{x}) \land \bigcap_{i=1}^{m_i} \text{Zero}(P_i)) \tag{8}
\]

Therefore we have:

\[
I(\bar{x}) \land (E(\bar{x}) \land c_i^{m_i} \land p_i) \Rightarrow I(\bar{x}) \tag{9}
\]

Suppose that the invariant assertion \( I(\bar{x}) \) is a parameterized polynomial equation over program variables or states:

\[
I(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} \sum_{j=1}^{m_i} a_{i,j} x_1^{j_1} x_2^{j_2} \ldots x_n^{j_n} = 0 \tag{10}
\]

For a given loop \( L \), suppose that \( p_1, \ldots, p_n \) are all possible closed paths, consisting of Boolean tests and assignment statements, each of which starts at the beginning of the body of the loop and ends at the same point. Suppose a closed path \( p_i \) which includes a sequence of assignment statements \( p_i, p_{i+1}, \ldots, p_n \). We compute characteristic set \( B = \{ p_i, p_{i+1}, \ldots, p_n \} \) and \( I(\bar{x}) \) pseudo-divide the characteristic set \( B \) to obtain pseudoremainder \( I(\bar{x}) \). \( s = I(\bar{x}) - I(\bar{x}) = 0 \) where \( s \) is a polynomial formula. Because \( s \) equals zero for all the possible values of \( x_1, x_2, \ldots, x_n \), (Sankaranarayanan et al., 2004) and its all coefficients are identical zero \( (I(\bar{x}) \) is an invariant). The coefficients of \( s \) are linear combination of coefficients of invariant assertion \( I(\bar{x}) \). In the closed path we can determine the partial or total coefficients of invariant assertion \( I(\bar{x}) \). Let the invariant assertion of which partial coefficients are ascertainment be \( I_{<i}(\bar{x}) \) and obviously \( I_{<i}(\bar{x}) \) are derived from \( I(\bar{x}) \). Because the same parameterized assertion can be used for multiple closed paths, \( I_{<i}(\bar{x}) \) is invariant assertion of another closed path. Repeat the above way and we can compute other coefficients of invariant assertion \( I_{<i}(\bar{x}) \). Similarly, the rest coefficients of \( I(\bar{x}) \) may be deduced by analogy. Finally we compute constant coefficient of invariant by substituting initial values of variables. In this way, we can obtain the invariant assertion \( I(\bar{x}) \) for every closed path.

Algorithm 1:

\begin{itemize}
  \item Input: Algebraized goal \( I(\bar{x}) \) with parameters and constraint conditions of all closed paths \( p_1, p_2, \ldots, p_n \), which have been represented as equations.
  \item Output: Invariant assertion of loop
\end{itemize}

\begin{algorithm}
\textbf{Begin}
\begin{itemize}
  \item for \( i=1 \) to \( m \) \{
    \item Compute characteristic set \( \{ p_i, p_{i+1}, \ldots, p_n \} \) of constraint conditions of \( p_i, p_{i+1}, \ldots, p_n \) of the \( i \)th closed path;
    \item \( I(\bar{x}) \) pseudo-divide \( B \) to obtain \( I(\bar{x}) \):
    \item \( s = I(\bar{x}) - I(\bar{x}) \):
    \item Because all coefficients of \( s \) are identical zero, we compute part parameters of \( I(\bar{x}) \):
    \item Substitute initial values of variables to compute constant coefficient of invariant;
    \item return \( I(\bar{x}) \);/* its coefficients are determined*/
\end{itemize}
\textbf{End}
\end{algorithm}

\textbf{In order to demonstrate our method, we give an example which is shown as follow.}

\textbf{Example 1:}

\begin{verbatim}
function product (X, Y: integer) return z: integer
  var x, y, z: integer;
  \{ x, y, z \} = (X, Y, 0);
  while \( y < 0 \) do
    \{ x, y, z \} = (2x, \( y \div \text{div} 2, z \));
    \[\{ x, y, z \} = (2x, (y-1) \div \text{div} 2, s+z) \];
  end while
end
\end{verbatim}

351
We transform the assignment statements of each closed path in Example 1 into polynomial equations.

\[ \begin{align*}
\tau_1 p_1 &= x - 2u \\
p_1 &= y - (1/2)v \\
\tau_2 p_2 &= x - 2u \\
p_2 &= y - (1/2)(v - 1) \\
p_3 &= z - (u + w)
\end{align*} \] (11)

Suppose that \( I(x, y, z) \) is a polynomial equation in \( x, y, z \) in which the degree of every variable is at most 1.

\[ I(x, y, z) = a_1xyz + a_2xy + a_3xz + a_4yx + a_5x + a_6y + a_7z + a_8 = 0 \] (12)

At first we compute characteristic set \( B \) of \( \{p_{11} = x - 2u, p_{12} = y - (1/2)v \} \). Then we compute the pseudo-remainder \( r \) of \( I(x, y, z) \) pseudo-dividing the characteristic set \( B \) through MMP.

\[ \begin{align*}
&> \text{css:}=[a_1x^2y^3z + a_2x^3y^2z + a_3x^2y^3z + a_4x^2y^3z + a_5xy + a_6y + a_7y^2 + a_8], [x-2u], [y-(1/2)v]; \\
&> \text{ord:}=[x, y, z, u, v]; \\
&> \text{chuser(csl, ord)};
\end{align*} \]

where, variables’ ordering is given by \( \text{ord} \) and \( \text{chuser} \) is a function in MMP.

After computation we obtain:

\[ I(x, y, z) = a_1xyz + a_2xy + 2a_3xz + 2a_4y + a_5x + a_6y + a_7z + a_8 = 0 \] (13)

Successively we compute the coefficients of \( I(x, y, z) \) and \( I(x, y, z) \) to obtain:

\[ s = a_5x - 2a_4y + a_5x + \frac{1}{2}a_6y = 0 \] (14)

According to literature (Sankaranarayanan et al., 2004) a polynomial \( s \) is zero for all the possible values of \( x_1; x_2; \ldots; x_k \) if all its coefficients are identically zero. This implies that the coefficient of every term in the polynomial is 0, namely \( a_3 = a_4 = a_5 = a_6 = a_7 = 0 \).

\[ I(x, y, z) = a_1xyz + a_2xy + a_3xz + a_1 \] (15)

In succession we compute characteristic set \( B \) of \( \{p_{11} = x - 2u, p_{12} = y - (1/2)v - 1, p_{13} = z - (u + w)\} \). We compute the pseudo-remainder \( r \) of \( I(x, y, z) \) pseudo-dividing the characteristic set \( B \) through MMP.

\[ \begin{align*}
&> \text{css:}=[a_1x^3y^3z + a_2x^2y^4z + a_3x^2y^5z + a_8], [x - 2u], [y - (1/2)v - 1], [z - (u + w)]; \\
&> \text{ord:}=[x, y, z, u, v, w]; \\
&> \text{chuser(csl, ord)};
\end{align*} \]

We obtain:

\[ I(x, y, z) = 2a_5x^2y^3 + 2a_3x^2y^3z + a_2xy + (a_1 - a_3)x + a_5z + a_8 = 0 \] (16)

We compare the coefficients of \( I(x, y, z) \) and \( I(x, y, z) \) to obtain \( a_1 = 0 \) and \( a_2 - a_3 = 0 \). Now we have \( I(x, y, z) = a_2xy + a_3z + a_8 = 0 \). When loop is entered, the initial values \( x^k \) and \( y^k \) of the variables will be substituted in \( I(x, y, z) \) to obtain \( I(x^k, y^k, 0) = a_2x^ky^k + a_8 = 0 \) which gives \( a_2 = -a_1x^ky^k \).

In this way the loop invariant is found:

\[ I(x, y, z) = a_2xy + a_3z - a_2x^ky^k = 0 \Rightarrow xy + z - x^ky^k = 0 \] (17)

In the above computation we hypothesize that the bounded degree is 1. When we hypothesize that the bounded degree is greater than 1, we can still obtain loop invariant \( xy + z - x^ky^k = 0 \).

**EXPERIMENTS**

Impressive results achieved using Wu’s method have once again sparked a great deal of interest in automatic geometry theorem proving. Kapur and Wan (1990) in particular has reported success in using Wu’s method for all of these problems and his program took much less time. Based on the experiments of Kapur and Wan (1990), it appears that the method using Wu’s characteristic set computation is more efficient than the method using Grobner basis algorithm.

Let Wu’s characteristic set \( B = \{g_1, g_2, \ldots, g_n\} \subset K(x_1, x_2, \ldots, x_k) \) and \( \text{deg}(g_i) \leq d, i = 1, \ldots, n \). By literature (Gallo and Mishra, 1994) the complexity of computing Wu’s characteristic set is \( O(n^{2m}(d + 1)^{n^2}) \). Then the Wu’s algorithm for finding loop invariants takes \( O(n^{2m}(d + 1)^{n^2} + m) \), where \( m \) is the number of closed paths. Therefore, the complexities are all \( O(n^{2m}(d + 1)^{n^2}) \) which is called a singly exponential complexity w.r.t. n. If we adopt Grobner base algorithm the complexity is:

\[ O((n^{2m}(d + 1)^{n^2} + mn) \]

(Lakshman, 1990), namely:

\[ O((n^{2m}(d + 1)^{n^2}) \]

which is double exponential w.r.t n. The method using Wu’s algorithm is superior to Grobner base obviously. The experiments also validate the case.
Table 1: The time comparison of finding loop invariants between the experimental results using Groebner bases (Pentium 4 with a 3.4 GHz processor and 2 GB of memory in literature (Rodríguez-Carbonell and Kapur, 2007) and our results using Wu's method (Pentium 4 with a 1.8 GHz processor and 512MB of memory) and obviously the computer performance applying Groebner bases is superior to mine) in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Source</th>
<th>Groebner bases</th>
<th>Wu's method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dijkstra</td>
<td>√</td>
<td>(Dijkstra)</td>
<td>1.5</td>
<td>0.56</td>
</tr>
<tr>
<td>Divbin</td>
<td>√</td>
<td>(Kaldewaij)</td>
<td>2.1</td>
<td>0.99</td>
</tr>
<tr>
<td>Freire1</td>
<td>√</td>
<td>(Freire)</td>
<td>0.7</td>
<td>0.48</td>
</tr>
<tr>
<td>Freire2</td>
<td>√</td>
<td>(Freire)</td>
<td>0.7</td>
<td>0.51</td>
</tr>
<tr>
<td>Cohenea</td>
<td>√</td>
<td>(Cohen)</td>
<td>0.7</td>
<td>0.53</td>
</tr>
<tr>
<td>Fermat</td>
<td>Factor</td>
<td>(Bressoud)</td>
<td>0.8</td>
<td>0.43</td>
</tr>
<tr>
<td>Wensley</td>
<td>Division</td>
<td>(Wegbreit)</td>
<td>1.1</td>
<td>0.47</td>
</tr>
<tr>
<td>Lenn</td>
<td>Lenn</td>
<td>(Dijkstra)</td>
<td>1.0</td>
<td>0.44</td>
</tr>
<tr>
<td>Peter3</td>
<td>Power sum</td>
<td>(Peter)</td>
<td>1.3</td>
<td>0.33</td>
</tr>
<tr>
<td>Peter4</td>
<td>Power sum</td>
<td>(Peter)</td>
<td>1.3</td>
<td>0.38</td>
</tr>
<tr>
<td>Peter5</td>
<td>Power sum</td>
<td>(Peter)</td>
<td>1.4</td>
<td>0.43</td>
</tr>
</tbody>
</table>

We have presented a sound method based on Wu's characteristic set method for finding polynomial invariants of imperative programs. The technique has been implemented using the algebraic geometry tool MMP. The implementation has successfully computed invariants for many non-trivial programs. Its performance is satisfactory as can be seen in Table 1. Compared with other approaches (Rodríguez-Carbonell and Kapur, 2007), our method is more efficient through these experiments.

CONCLUSION

An invariant of a program at a location can be seen as an assertion (we transform the assertion into polynomial equation in this study) that is true of any program state reaching the location. The importance of the automatic invariant finding problem for the verification and the analysis of programs are well-known. We have presented an approach for automatically finding invariants of loop programs using Wu's characteristic method and implement it on auto reasoning platform MMP. The technique has many advantages; first of all, our method has wider application domain. Secondly compared other methods by experiments we prove that our method is more efficient. Automatic proving and verification of program is a trend and algebraic method can be mechanized easily. For future research, we are interested in exploring the proposed research along such directions: generalize and extend our method to other program models and integrate this and other methods for mechanically program verification and proving etc.

ACKNOWLEDGMENTS

The study described in this research was supported by a grant from the National Science Foundation of China (NSFC) (60671033) and the Doctoral Foundation of the Ministry of Education of China (2006061405).

REFERENCES


