A General Data Dependence Analysis to Nested Loop Using Integer Interval Theory

Jing Zhou 1,2, Guosun Zeng 1,2

1 Department of computer Science and Technology, Tongji University, Shanghai 200092, China
dinese@163.com, gszeng@mail.tongji.edu.cn
2 Tongji Branch, National Engineering & Technology Center of High Performance Computer, Shanghai 200092, China

Abstract

Many dependence tests have been proposed for loop parallelization in the case of arrays with linear subscripts, but little work has been done on the arrays with non-linear subscripts, which sometimes occur in parallel benchmarks and scientific and engineering applications. This paper focuses on array subscripts coupled integer power index variables. We attempt to use the integer interval theory to solve the above difficult dependence test problem. Some “interval solution” rules for polynomial equations have been proposed in this paper. Furthermore, based on the proposed rules, we present a novel approach to loop dependence analysis, which is termed the Polynomial Variable Interval test or PVI-test, and also develop a related algorithm. Some case studies show that the PVI-test is effective and efficient. Compared to the VI test, the PVI-test makes significant improvement, and is therefore a more general scheme of dependence test.

1 Introduction

Data dependence analysis is critical for parallel compiler to detect independent operations. Especially, a great deal of effort has been spent on loop dependence tests because of rich parallelism hidden in loops. The dependence analysis for array references, in fact, can be reduced to solving Diophantine equations [1]. It is a NP complete problem. Therefore a majority of dependence tests are heuristic methods in the literature. The GCD test [2] is based on the elementary number theory, which states that a linear equation has an integer solution if and only if the greatest common divisor of the coefficients divides the constant term. It is a necessary but not sufficient condition for data dependence. The Banerjee bound test [3] checks whether there is a real solution for the dependence equation. If the test produces yes answer, in practice, it implies the existence of dependence. The Omega test [4] is a general purpose algorithm that is based on Fourier-Motzkin variable elimination (FMVE) [5] and its integer extensions, while the Power test [6] combines the Generalized GCD test with FMVE [5] and takes loop limits and direction vector constraints into consideration. Though the two tests produce exact yes/no answers, they have worst-case exponential time complexity. The I-test [7] is a polynomial time test. Generally speaking, the I-test is a linear time exact test in most cases for single dimensional array references, but cannot precisely handle multidimensional array references involving coupled subscripts and rely on information known at compile time [8]. The DVI test [9] checks the existence of integer solutions and takes loop constant bounds and direction vector constraints into account for one-dimensional arrays. The GDVI test [10] extends the DVI test to handle loops with variable bounds. The IR test [11, 12] aims to a typical kind of loops with an arbitrary direction vector and triangular bound. The VI test [13], related to our research, has begun original work on dependence test using integer interval theory. But it can only prove or disprove the existence of data dependences in loops with linear subscripts. Other data dependence tests can be seen in the literature [14-21].

As mentioned above, in short, the most existing approaches to dependence test perform well only for nested loop with linear (affine) array subscripts, and in each test there is trade off between accuracy and efficiency. However, in the case of subscripts coupled integer power variable, such as \( b_{c_1} \cdot i_1^{A_1} + b_{c_2} \cdot i_2^{A_2} + \ldots b_{c_k} \cdot i_k^{A_k} \), where \( c_k \) is integer constants and \( i \) is loop index, the existing approaches can not work well. To our knowledge, for the more general form \( b_{c_i} \cdot \sum_{k=1}^n b_{c_i} \cdot i_k^{A_k} \) of array subscripts occurring in some science application, few works have been done to deal with the situation. Thus this paper focuses on this kind of complicated problem. In order to determine the data dependence in array references with nonlinear subscripts coupled integer power variable, we develop a general dependence analysis method, named
PVI-test, based on the integer interval theory. The PVI-test presents a significant improvement over the VI-test through experiment.

The remainder of this paper is structured as follows. Section 2 presents problem description and introduces the integer interval theory. Section 3 proposes theories served the PVI-test. On basis of proposed theories, section 4 develops a novel algorithm for recognizing the data dependence in multidimensional arrays with non-linear subscripts. Then a case study is illustrated in section 5. Finally, Section 6 concludes the paper and points out the future work.

2 Preliminaries

2.1 Nested Loop with Nonlinear Subscripts

Coupled Integer Power Variables

To describe clearly, we first depict the addressed loop program as shown in Figure 1. Without loss of generality, we assume that all bounds of the loop are linear functions of the outer loop indices, and each value of lower and upper bounds is positive integer. The loop step is 1. S1 and S2 are statements in the loop body. There may exist dependence between the two array references. Their array subscripts, coupled integer power variables, have the following form of polynomial expression:

\[ b_0 + b_1 \cdot i_1^{v_1} + b_2 \cdot i_1^{v_2} + \ldots + b_{m_2} \cdot i_2^{v_{m_2}} \]

where \( b_0 \) is an integer constant, \( b_j \) such as \( b_{ij} \) is integer constant coefficient and \( i_k \) is a loop index with power \( v_{xy} \), \( k, j, m \in \mathbb{Z}^+ \).

To make it easier for us to solve the problem, we will change the above equation (1) subject to (2) to the following polynomial interval equation (3) subjected to (4).

\[ \sum_{i=1}^{n-1} a_{ij} X_{ij} = [a_0, a_n] \]

(3)

\[ P_{2k-1}(x) \leq X_{2k-1} \leq Q_{2k-1}(x) \]

(4)

To make it easier for us to solve the problem, we will change the above equation (1) subject to (2) to the following polynomial interval equation (3) subjected to (4).

\[ \sum_{i=1}^{n-1} a_{ij} X_{ij} = [a_0, a_n] \]

(3)

\[ P_{2k-1}(x) \leq X_{2k-1} \leq Q_{2k-1}(x) \]

(4)

where \( X_{2k-1} = i_k \) and \( X_{2k} = i_k \) are two instances of the same loop iteration variable contained power \( v_{xy} \), and \( a_k = c_k - b_k \).

2.2 Integer Interval Theory

As discussed above, it is rather difficult to solve the equation (1) subjected to the inequalities (2). So we give up finding the exact solutions. But the existence of solutions may be determined by integer interval theory. The followings are some concepts of integer interval theory.

Definition 1 We define the positive and negative part of an integer a, respectively, as

\[ a^+ = \begin{cases} a & \text{if } a > 0 \\ 0 & \text{otherwise} \end{cases} \]

\[ a^- = \begin{cases} -a & \text{if } a < 0 \\ 0 & \text{otherwise} \end{cases} \]

Definition 2 Given an integer region \( R \in \mathbb{Z}^n \) and two functions \( L \) and \( U \) from \( R \) to \( \mathbb{Z} \), we define the variable integer interval, denoted \([L(x), U(x)]\), as the union of all integer intervals for all values \( x_i \)'s of \( x \) in \( R \):

\[ [L(x), U(x)] = \bigcup_{x_i \in R} [L(x_i), U(x_i)] \]

Definition 3 Given an integer region \( R \subseteq \mathbb{Z}^n \) and three functions \( F \), \( L \), and \( U \) from \( R \) to \( \mathbb{Z} \), we define a variable interval equation of the following form:

\[ F(x) = [L(x), U(x)] \]

The above equation is said to be integer solvable in \( R \) iff there exist a value of \( x \), \( x_0 \in R \) such that \( L(x_0) \leq F(x_0) \leq U(x_0) \).

3 Theories towards the Polynomial Variable Interval Test
To facilitate dependence test, we first present some “interval solution” rules for polynomial equations based on the integer interval theory.

**Theorem 1**
The polynomial interval equation $F(x)+aX^2=[L(x)+bX^2,U(x)+cX^2]$ is integer solvable subject to a set of constraints on $x$ in $R$ and the constraint $P(x)\leq Q(x)$, if the polynomial interval equation $F(x)+aX^2=\{L(x)+bX^2, U(x)+cX^2\}$ is integer solvable subject to the same constraints.

**Proof**
Let us assume that the polynomial interval equation $F(x)=[L(x)+(b-a)X^2, U(x)+(c-a)X^2]$ is integer solvable, if there exist a value $x, x_0 \in R$ such that $L(x_0)\leq F(x_0)\leq U(x_0)$, which means $\forall x \in R, L(x_0)\leq F(x)\leq U(x_0)$.

Thus, there exist $x_0$ such that $L(x_0)+aX^2 \leq F(x_0)+aX^2 \leq U(x_0)+aX^2$.

According to Definition 3 we can prove the polynomial interval equation:

$$F(x)+aX^2=[L(x)+bX^2, U(x)+cX^2]$$

is integer solvable subject to the same constraints.

**Theorem 2**
The variable integer interval $[L+\mu X^2, U+\nu X^2]$, where $P^2 \leq Q^2$, is equal to the integer interval $[L+\mu P^2, U+\nu Q^2]$ if $\nu \mu \leq 0$, or $\mu \nu > 0$ and $U-L+(\mu-\nu)P^2-(\nu-\mu)Q^2+1 \geq \min(\mu, |\nu|)$.

**Proof**
The proof of the theorem can be divided into the following two parts:

-part[1]: if the condition of $\mu \leq 0$, $\nu > 0$ and $U-L+(\mu-\nu)P^2-(\nu-\mu)Q^2+1 \geq \min(\mu, |\nu|)$ is satisfied, the variable integer interval $[L+\mu X^2, U+\nu X^2]$ is equal to the integer interval $[L+\mu P^2, U+\nu Q^2]$.

-part[2]: if $[L+\mu X^2, U+\nu X^2]$ is equal to the integer interval $[L+\mu P^2, U+\nu Q^2]$, where $P^2 \leq Q^2$, we can conclude that the inequality $U-L+(\mu-\nu)P^2-(\nu-\mu)Q^2+1 \geq \min(\mu, |\nu|)$, exist when $\mu \nu > 0$. We will consider two conditions of $\mu \nu > 0$; one is $\mu \nu > 0$ and $\nu > 0$ and another is $\mu < 0$ and $\nu < 0$.

**Proof of part[1]**
Because of $[L(x), U(x)] = \bigcup_{x \in R} [L(x), U(x)]$, where $P^2 \leq Q^2$ and $\nu > 0$, we can prove that $\bigcup_{x \in R} [L(x), U(x)]$ is equal to the integer interval $[L+\mu X^2, U+\nu X^2]$.

**Proof of part[2]**
According to Definition 2 and hypothesis, $\bigcup_{x \in R} [L(x), U(x)]$ is equal to the integer interval $[L+\mu P^2, U+\nu Q^2]$, where $P^2 \leq Q^2$.

Thus, the final integer interval $[L+\mu Q^2, U+\nu Q^2]$ contains all of all the previous intervals. It means that $\bigcup_{x \in R} [L(x), U(x)]$ is equal to $[L+\mu P^2, U+\nu Q^2]$.

This completes the proof.
Theorem 3

Consider the variable integer interval 
\[[L(x_i)+\mu x_i^2, U(x_i)+\nu x_i^2]\], subject to a set of constraints on \(x_i\) in \(R\) and \(P^2(x_i) \leq Q^2(x_i)\), where \(X\) does not appear in any of the constraints in \(R\).

If \(\mu \leq 0\), or \(\mu > 0\) and \(\min(U(x_i)-L(x_i))+(\nu-\mu) P^2(x_i) - (\nu-\mu) Q^2(x_i)+1 \geq \min(\mu, |\nu|)\), then the above variable integer interval is equal to 
\[[L(x_i)+\mu P^2(x_i)-\mu Q^2(x_i), U(x_i)+\nu Q^2(x_i)-\nu P^2(x_i)]\], subject to the same constraints on \(x_i\) in \(R\) and the constraint \(P^2(x_i) \leq Q^2(x_i)\).

Proof

According to the hypothesis \(\mu \leq 0\), or \(\mu > 0\) and \(\min(U(x_i)-L(x_i))+(\nu-\mu) P^2(x_i) - (\nu-\mu) Q^2(x_i)+1 \geq \min(\mu, |\nu|)\), for all \(x_i \in \mathbb{Z}\) such that \(P^2(x_i) \leq Q^2(x_i)\).

According to Theorem 1, each variable integer interval 
\[[L(x_i)+\mu x_i^2, U(x_i)+\nu x_i^2]\], where \(P^2(x_i) \leq Q^2(x_i)\), is equal to the integer interval 
\[[L(x_i)+\mu P^2(x_i)-\mu Q^2(x_i), U(x_i)+\nu Q^2(x_i)-\nu P^2(x_i)]\]. Now according to Definition 2:

\[
\bigcup_{P^2(x_i) \leq Q^2(x_i)} \bigcup_{x_i \in \mathbb{Z}} [L(x_i)+\mu x_i^2, U(x_i)+\nu x_i^2] = [L(x_i)+\mu P^2(x_i)-\mu Q^2(x_i), U(x_i)+\nu Q^2(x_i)-\nu P^2(x_i)]
\]

for all \(x_i \in \mathbb{Z}\), where \(P^2(x_i) \leq Q^2(x_i)\).

By Definition 2 the double union of the integer intervals on the left hand side of the above equation is equal to the variable integer interval 
\[[L(x_i)+\mu x_i^2, U(x_i)+\nu x_i^2]\], where \(x_i \in \mathbb{Z}\) and \(P^2(x_i) \leq Q^2(x_i)\). Thus we conclude that the variable integer interval 
\[[L(x_i)+\mu P^2(x_i)-\mu Q^2(x_i), U(x_i)+\nu Q^2(x_i)-\nu P^2(x_i)]\], where \(P^2(x_i) \leq Q^2(x_i)\).

Theorem 4

Consider the following variable interval equation:

\[F(x)+aX^2=[L(x)+bX^2, U(x)+cX^2]\]

subject to a set of constraints on \(x\) in \(Z^2\) and \(P^2(x) \leq Q^2(x)\), where \(X\) does not appear in any of the constraints in \(R\). If \((b-a)(c-a) \leq 0\), or \((b-a)(c-a) > 0\) and \(\min(U(x)-L(x))+(b-c) P^2(x) - (b-c) Q^2(x) +1 \geq \min(b-a, c-a)\), then the equation above is integer solvable iff the variable interval equation:

\[F(x)=[L(x)+bX^2, U(x)+(c-a)^+ Q^2(x)-(c-a)^+ P^2(x)]\]

is integer solvable subject to the same constraints on \(x\) in \(R\) and the constraint \(P^2(x) \leq Q^2(x)\).

Proof

According to the Theorem 1, the polynomial interval equation

\[F(x)+aX^2=[L(x)+bX^2, U(x)+cX^2]\]

is integer solvable subject to a set of constraints, where \(P^2(x) \leq Q^2(x)\), iff the polynomial interval equation:

\[F(x)=[L(x)+(b-a)X^2, U(x)+(c-a)^+ X^2]\]

is integer solvable subject to the same constraints. Also it can be concluded from Theorem 2, \(F(x)=[L(x)+(b-a)X^2, U(x)+(c-a)^+ X^2]\) is integer solvable if the variable interval equation

\[F(x)=[L(x)+(b-a) P^2(x)-(b-a) Q^2(x), U(x)+(c-a)^+ Q^2(x)-(c-a)^+ P^2(x)]\]

is integer solvable with the constraints that \((b-a)(c-a) \leq 0\), or \((b-a)(c-a) > 0\) and \(\min(U(x)-L(x))+(b-c) P^2(x) - (b-c) Q^2(x) +1 \geq \min(b-a, c-a)\) and that concludes our proof.

Theorem 5

Consider the following variable interval equation:

\[F(x)+aX^2=[L(x)+bX^2, U(x)+cX^2]\]

subject to a set of constraints on \(x\) in \(R\), \(P^2(x) \leq Q^2(x)\) and \(n \in \mathbb{Z}^+\), where \(X^2\) does not appear in any of the constraints in \(R\). If \((b-a)(c-a) \leq 0\), or \((b-a)(c-a) > 0\) and \(\min(U(x)-L(x))+(b-c) P^2(x) - (b-c) Q^2(x) +1 \geq \min(b-a, c-a)\), then the equation above is integer solvable iff the variable interval equation:

\[F(x)=[L(x)+(b-a) P^2(x)-(b-a) Q^2(x), U(x)+(c-a)^+ Q^2(x)-(c-a)^+ P^2(x)]\]

is integer solvable subject to the same constraints on \(x\) in \(R\) and the constraint \(P^2(x) \leq Q^2(x)\).

Proof

The proof is the same as Theorem 4.

Especially, when \(n=1\), theorem 5 just is reduced to theorem 5 in the VI-test [13]. Also obviously theorem 2 is the instance of theorem 5. So theorem 5 is a general rule to solve the problem of array references either with linear subscripts or non-linear.
4 The Algorithm for the PVI-test

4.1 Basic idea

Our algorithm consists of three steps: the first is to rewrite the Diophantine equations (1) into the form of polynomial interval equation (3) subjected to inequalities (4). The second is to repeatedly eliminate variables which do not appear in any constraints in the equation, in order to decrease the constraints. The last is to judge whether dependence exists. During the step 2, according to Theorem 5, we treat the $X_i$ as a variable with its own constrain like a usual variable X. At the end of the step 2, an integer interval with zero on the right-hand side, is obtained. If zero belongs to the integer interval on the right-hand side, there exists integer solutions to the equation (1), that is to say that dependence exists in the loop.

Furthermore, before eliminating each variable from the polynomial interval equation, we should check following two accuracy conditions [13]:

**Accuracy Condition 1** For every variable $X$, eliminated from the variable interval equation:
\[F(x)+aX^n=[L(x)+bX^n, U(x)+cX^n], \quad \text{ne Z}\]
subject to a set of constraints on $x$ in R and $P^i(x) \leq X^n \leq Q^i(x)$, where $X$ does not appear in any of the constraints in R, the following inequalities need to be satisfied:

\[(b-a)(c-a) \leq 0, \quad (b-a)(c-a) > 0, \quad \text{min}(U(x)-L(x)+(c-b)P(x)^n - (c-b)Q(x)^n+1) \leq \text{min}((b-a)(c-a))\]

**Accuracy Condition 2** For every variable $X$, eliminated from the variable interval equation,
\[F(x)+aX^n=[L(x)+bX^n, U(x)+cX^n], \quad \text{subject to a set of constraints on } x \text{ in R and the constraint } P^i(x) \leq X^n \leq Q^i(x),\]
where $X$ does not appear in any of the constraints in R, the following inequality needs to be satisfied: $\text{min}(Q^i(x)-P^i(x)) \geq 0, \text{ne Z}$.

4.2 The algorithm

According to the theories and analysis about the PVI-test, we propose an algorithm for our dependence test towards the loop shown in Figure 1. The variables, which can be eliminated, are found by the procedure called Detector and pushed into a stack. The procedure acc_condition is used to judge whether variable satisfies above two accurate conditions, or adjust the lower bound to satisfy conditions and turn back the revised value of lower bound. The procedure Pos and Neg return the positive and negative part of an integer x, respectively.

**Algorithm: PVI-test for data dependence**

**Input:** $I^m\{a_{11}, \ldots, a_{1m}, a_{21}, \ldots, a_{2m}, \ldots, a_{n1}, \ldots, a_{nm}\};$

Output: true means that the equation (1) is integer solvable and false means that the equation (1) is not integer solvable.

**Procedure** PVI (input, output)

```plaintext
{ /* Initialization */
A←{a_{11}, \ldots, a_{1m}, a_{21}, \ldots, a_{2m}, \ldots, a_{n1}, \ldots, a_{nm} \};
B←{l^{(1m)}, \ldots, l^{(2n)}, \ldots, l^{(2nm)}},
\quad u^{(2nm)}, \ldots, u^{(2nm)} \};
X←{X^{(1m)}, \ldots, X^{(2n)}, \ldots, X^{(2nm)}},
\quad X^{(2n)}, \ldots, X^{(2nm)} \};
F←{a_{11}X_{11}, \ldots, a_{12}X_{12}^2 + \ldots + a_{2n}X_{2n}^m + \ldots, a_{2n}X_{2n}^m + a_{2n}X_{2n}^m + \ldots + a_{2nm}X_{2nm}^m};
L←a_0; U←a_0;

While (B≠∅)
{ Detector(X, F, stack);
While (stack≠∅)
{ elim←stack;
    if elim<0 then
        flag2←true;
    else
        flag2←false;
    
    /*elim is the integer constant coefficient of the X^\ell, l^0 and
    u^0 are the lower bound and upper bound of X^\ell, respectively, as described above*/
    Output: true means that the equation (1) is integer solvable and false means that the equation (1) is not integer solvable.
}
}

Procedure Pos(x)
{ if x>0 then return x else return 0; }

Procedure Neg(x)
{ if x<0 then return -x else return 0; }

Procedure Detector(X, F, stack)
{ for ∀X_i ∈ X
    /* X is the union of variables in the polynomial interval equation */
    if (X_i not exist in B) & (X_i not exist in B) then
    stack←X_i;
}

Procedure acc_condition (L, U, l^0, u^0, a, b, c)
{ flag2←false;
    if (b-a)(c-a)≤0 then flag2←true;
    else
}
```
if \( \min (U-L+\ l\times \text{Pos}(c-b)^{+} - \text{Neg}(c-b) + 1) \geq \min [b-a, c-a] \) then
flag2=true;
else
while not flag2
{\( l' = l' - 1; \)
acc_condition \((L, U, l', l', a, b, c)\)}
if flag2 then
if \( \min [u', l'] \geq 0 \) then return true;
else return false;

5 Case Study

To verify the proposed PVI-test in previous session, we demonstrate how to handle loops with subscripts coupled integer power index variable and detect the dependence based on interval solution rules.

DO \( i = 1, 5 \)
DO \( j = 5 - i, 2 \times i + 7 \)
S: \( A [i+1+j]^2 = A [j^2 - 2xi + 10 + \ldots] \)
ENDDO
ENDDO

Figure 2 An example loop program with integer variable interval

There is data dependence between the two array references in statement \( S \) in Figure 2, if we can conclude that the right of the final interval equation includes zero through the proposed algorithm. We give the polynomial interval equation \((5)\) subjected to inequalities \((6)\) as follows:

\[
X_1^4 + X_1^2 + 2X_2^2 + X_2^4 \leq \left[ 10, 10 \right]
\]

\(1 \leq X_1 \leq 5\)
\(1 \leq X_2 \leq 5\)

\(5-X_1 \leq X_2 \leq 7\)

\((5-X_1)^2 \leq X_2 \leq (2X_1 + 7)^2\)

\(5-X_1 \leq X_2 \leq 7\)

\((5-X_1)^2 \leq X_2 \leq (2X_1 + 7)^2\)

where \(X_1\) and \(X_2\) are instances of the loop variable \(i\) and \(X_1, X_2\) are instances of the loop variable \(j\).

The variables that may be eliminated, according to our algorithm, are \(X_1^2\) and \(X_2^2\). We may start with \(X_2^2\). In this case \(a=1, b=0, c=0, L(x)=x_{10}, U(x)=10, P(x)=(5-X_2)^2, Q(x)=(2X_1 + 7)^2\). In addition, before eliminating, we should check two accuracy conditions as shown in the algorithm.

Checking Accuracy Condition 1
\((0-(-1))(0-(-1))\geq 0\), and
\(\min (10-10 + (5-X_1)^2 - 0(2X_1 + 7)^2 + 1) \geq 0 = \min (0-(-1), 0-(-1))\).

Checking Accuracy Condition 2
\(\min ((2X_1 + 7)^2 - (5-X_1)^2) = \min (2X_2 + 7 + 5-X_2) (2X_1 + 7 - 5 + X_2) \geq 0\).

Since both accuracy conditions are satisfied, we can indeed eliminate \(X_1\) and the polynomial interval equation is transformed into:

\[
X_1^4 + X_1^2 + 2X_2^2 + X_2^4 = \left[ 10 + 0(-(-1))(5-X_1)^2, 10 + 0(-(-1))(2X_1 + 7)^2 \right] \implies X_1^4 + X_1^2 + 2X_2^2 + X_2^4 = \left[ X_1^2 - 10X_2 + 35, 4X_1^2 - 28X_2 + 59 \right] = \left[ 10 + (5-X_1)^2, 10 + (2X_1 + 7)^2 \right].
\]

We continue by eliminating variable \(X_1^2\). In this case \(a=1, b=0, c=0, L(x)=X_1^2 - 10X_2 + 35, U(x)=4X_1^2 - 28X_2 + 59, P(x)=(5-X_1)^2, Q(x)=(2X_1 + 7)^2\). And check two accuracy conditions.

Checking Accuracy Condition 1
\((0-1)(0-1)\geq 0, \) and
\(\min (4X_1^2 - 28X_2 + 59 - (X_1^2 - 10X_2 + 35) + 0(5-X_1)^2 - 0(2X_1 + 7)^2 + 1) = \min (10 + (2X_1 + 7)^2 - (10 + (5-X_1)^2) + 1) = \min (0-(-1), 0-(-1))\).

Checking Accuracy Condition 2
\(\min ((2X_1 + 7)^2 - (5-X_1)^2) \geq 0\).

Since both accuracy conditions are satisfied we can indeed eliminate \(X_1^2\) and the polynomial interval equation is transformed into:

\[
X_1^4 + X_1^2 + 2X_2^2 + X_2^4 = \left[ 10 + 0(-(-1))(5-X_1)^2, 10 + 0(-(-1))(2X_1 + 7)^2 \right] \implies X_1^4 + X_1^2 + 2X_2^2 + X_2^4 = \left[ X_1^2 - 10X_2 + 35, 4X_1^2 - 28X_2 + 59 \right] = \left[ 10 + (5-X_1)^2, 10 + (2X_1 + 7)^2 \right].
\]

We continue by eliminating \(X_2^2\). In this case \(a=0, b=1, c=4, L(x)=4X_1^2 - 28X_2 + 24, U(x)=28X_2 - X_1^2 + 18X_1^4 - 14, P(x)=1, Q(x)=25\).

Checking Accuracy Condition 1
\((1-0)(4-0)\geq 0, \) and
\(\min (28X_2 - X_1^2 + 18X_1^4 + 24 - 10X_2 - 4X_1^2 - 28X_1^2 + 144) \geq 0 = \min (0-(-1), 0-(-1))\).

Checking Accuracy Condition 2
\(\min (25-1) = 24 \geq 0\).

Since both accuracy conditions are satisfied we can indeed eliminate \(X_2\) and the polynomial interval equation is transformed into:

\[
X_1^4 + X_1^2 + 2X_2 = \left[ 10 + (5-X_1)^2 + (2X_1 + 7)^2, 10 + (2X_1 + 7)^2 - (5-X_1)^2 \right] \implies X_1^4 + X_1^2 + 2X_2 = \left[ X_1^2 - 10X_2 + 35, 4X_1^2 - 28X_2 - 14, 4X_1^2 + 28X_2 - X_1^2 + 18X_1^4 + 34 \right].
\]

Next the variable \(X_1^2\) can be eliminated. In this case \(a=0, b=1, c=4, L(x)=4X_1^2 - 28X_2 + 24, U(x)=28X_2 - X_1^2 + 18X_1^4 - 14, P(x)=1, Q(x)=25\).

Checking Accuracy Condition 1
\((1-0)(4-0)\geq 0, \) and
\(\min (28X_2 - X_1^2 + 18X_1^4 + 24 - 10X_2 - 4X_1^2 - 28X_1^2 + 144) \geq 0 = \min (0-(-1), 0-(-1))\).

Checking Accuracy Condition 2
\(\min (25-1) = 24 \geq 0\).

Since both accuracy conditions are satisfied we can indeed eliminate \(X_2\) and the polynomial interval equation is transformed into:

\[
X_1 + X_1^3 + 2X_2 = \left[ 10X_2 - 4X_1^2 - 28X_1 - 14 + (1-0) \times 128X_2 - X_1^2 + 18X_1^4 + 34 \right] \implies X_1 + X_1^3 + 2X_2 = \left[ 10X_2 - 4X_1^2 - 28X_1 - 13, 28X_2 - X_1^2 + 18X_1^4 + 134 \right].
\]

And we can eliminate \(X_1\) in the same way, and get the polynomial interval equation:
\(X_1^5 + X_1^3 = [4X_2^2 - 28X_2 - 73, -X_2^2 + 18X_1^2 + 264]\)

We continue by eliminating variable \(X_1^3\). In this case \(a=1, b=0, c=0, L(x)=-4X_2^2 - 28X_2 - 73, U(x)=-X_2^2 + 18X_1^2 + 234, P(x)=1, Q(x)=5^2\).

Checking Accuracy Condition 1
\((0-1)(0-1) = 0, \) and
\(\min (X_2^2 + 18X_1^2 + 264 + (4X_2^2 - 28X_2 - 73) + 1) \geq 0 = \min (0-1, 0-1)\).

Checking Accuracy Condition 2
\(\min (-X_2^2 + 18X_1^2 + 264) + (-4X_2^2 - 28X_2 - 73) \geq 0\).

Since both accuracy conditions are satisfied we can indeed eliminate \(X_2\) and the polynomial interval equation is transformed into:

\[
X_1 = \left[ -4X_2^2 - 28X_2 - 198, 28X_2 - X_1^2 + 18X_1^2 + 263 \right].
\]

We continue by eliminating variable \(X_1^2\), \(X_1\) following the algorithm, and obtain the final interval equation as follows:
\(0=\left[ -433, 317 \right]\).
The PVI-Test concludes that there indeed exists an integer solution to the polynomial interval equation subject to the constraints.

Conclusion

Though many well-known research works have been done on the dependence analysis such as the Banerjee test, the I-test, the Omega test and the Power test, they tend to ignore the case of arrays with non-linear subscripts. But loops with non-linear subscripts are sometimes found in scientific source code, especially with polynomial subscripts coupled with integer power variables.

We have developed a novel dependence analysis scheme, the PVI-test, which can accurately handle loops with subscripts couple polynomial expressions and trapezoidal bounds. The test is based on “interval solution” theories for polynomial equation and solves the problem by repeatedly eliminating variables from the polynomial interval equation. In addition we illustrate how to use it to handle data dependence problems in loops and testify its accuracy. Compared to the others tests, the PVI-test is much more general and efficient to detect dependence in arrays either with linear subscripts or non-linear. Furthermore, the PVI-test extends the VI test which fails to solve problem with non-linear subscripts, as well as inherits all the benefits of the VI test. Our future work is to add the PVI-test into our parallel compiler.

References

[20] W.Paugh, A Practical algorithm for extract array depend-


Jing Zhou received her B.S and M.S degrees in computer science from Jiangxi Normal University in 2000 and 2003 respectively, and she is now a doctoral candidate in computer software and theory at Tongji University. Her main research interests include grid computing and high performance computing technology.

Guo-Sun Zeng achieved his Ph.D. degree from Shanghai Jiaotong University in 2000. He is concurrently a professor and Ph.D. supervisor of Tongji University. He is engaged in high performance computing technology.