ELSEVIER

Parallel Computing 27 (2001) 1783–1799

PARALLEL COMPUTING

www.elsevier.com/locate/parco

A multi-dimensional version of the I test

Weng-Long Chang [a], Chih-Ping Chu [b,*], Jia-Hwa Wu [b]

[a] *Department of Information Management, Kung Shan Institute of Technology, Tainan 701, Taiwan, ROC*

[b] *Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan 701, Taiwan, ROC*

Received 11 January 2001; received in revised form 13 April 2001; accepted 30 April 2001

Abstract

Two-dimensional arrays with linear subscripts occur quite frequently in real programs. In general, for *multi-dimensional linear* arrays under *constant* bounds the Lambda test is an efficient data dependence method to check whether there exist *real* solutions. In this paper, we propose a multi-dimensional version of the I test, *the multi-dimensional I test*, that can be applied to testing whether there are *integer* solutions for multi-dimensional linear arrays under constant limits. Experiments with benchmark showing the effects of the multi-dimensional I test on testing precision and testing efficiency are also presented. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Parallelizing/vectorizing compilers; Data dependence analysis; Loop parallelization; Loop vectorization

1. Introduction

The question of whether multi-dimensional array references with linear subscripts may be parallelized/vectorized depends upon the resolution of these multi-dimensional array aliases. The resolution of multi-dimensional array aliases is to ascertain whether two references to the same multi-dimensional array within a general loop may refer to the same element of this multi-dimensional array. This problem in general case can be reduced to that of checking whether a system of m linear equations with n unknown variables has a simultaneous integer solution, which

* Corresponding author. Tel.: +886-6-2757575x62527; fax: +886-6-2747076.
E-mail address: chucp@csie.ncku.edu.tw (C.-P. Chu).

PII: S 0 1 6 7 - 8 1 9 1 (0 1) 0 0 1 0 8 - 9

satisfies the constraints for each variable in the system. It is assumed that m linear equations in a system are written as

$$\begin{aligned} a_{1,1}X_1 + a_{1,2}X_2 + \cdots + a_{1,n-1}X_{n-1} + a_{1,n}X_n &= a_{1,0}, \\ &\vdots \\ a_{m,1}X_1 + a_{m,2}X_2 + \cdots + a_{m,n-1}X_{n-1} + a_{m,n}X_n &= a_{m,0}, \end{aligned} \tag{1.1}$$

where each $a_{i,j}$ is a constant integer for $1 \leqslant i \leqslant m$ and $1 \leqslant j \leqslant n$. It is postulated that the constraints to each variable in (1.1) are represented as

$$P_{r,0} + \sum_{s=1}^{r-1} P_{r,s}X_s \leqslant X_r \leqslant Q_{r,0} + \sum_{s=1}^{r-1} Q_{r,s}X_s, \tag{1.2}$$

where $P_{r,0}$, $Q_{r,0}$, $P_{r,s}$, and $Q_{r,s}$ are the constant integers for $1 \leqslant r \leqslant n$. That is, the bounds for each variable X_r are variable.

If each of $P_{r,s}$ and $Q_{r,s}$ is zero in the limits of (1.2), then (1.2) will be reduced to

$$P_{r,0} \leqslant X_r \leqslant Q_{r,0}, \quad \text{where } 1 \leqslant r \leqslant n. \tag{1.3}$$

That is, the bounds for each variable X_r are constants.

There are several well-known data dependence analysis algorithms applicable to *multi-dimensional linear* arrays under *constant* bounds: the Lambda test [2], an efficient method to check whether there exist *real* solutions; the Power test, a combination of Fourier–Motzkin variable elimination with an extension of Euclid's GCD algorithm [6]; the Omega test, which combines new methods for eliminating equality constraints with an extension of Fourier–Motzkin variable elimination to integer programming [3].

In this paper, the I test [1] and the Lambda test are integrated to check whether m linear equations (1.1) under *constant* bounds have *integer* solutions. (A dependence testing method determining if there exist *integer-valued* solutions is more precise than that determining if there exist *real-valued* solutions.) A theoretical analysis explains that we take advantage of the rectangular shape of the convex sets derived from m linear equations under constant limits in a data dependence testing. An algorithm called *the multi-dimensional I test* has been implemented and several measurements have also been performed.

The rest of this paper is proffered as follows. In Section 2, the summary accounts of the I test and the Lambda test are presented. In Section 3, the theoretical aspects and the worst-case time complexity of the multi-dimensional I test are described. Experimental results showing the advantages of the multi-dimensional I test are given in Section 4. Finally, brief conclusions are drawn in Section 5.

2. Background

The summary accounts of the I test and the Lambda test are introduced briefly in this section.

2.1. The I test

A linear equation with the bounds of (1.3) will be said to be integer solvable if the equation has an integer solution satisfying the bounds of each variable. The I test deals with a linear equation by first transforming it to an interval equation. Definitions 2.1 and 2.2 define integer intervals and interval equations [1].

Definition 2.1. Let $[\alpha_1, \alpha_2]$ represent the integer intervals from α_1 to α_2, i.e., the set of all integers between α_1 and α_2.

Definition 2.2. Let $a_1, \ldots, a_{n-1}, a_n$, L, and U be integers. A linear equation

$$a_1X_1 + a_2X_2 + \cdots + a_{n-1}X_{n-1} + a_nX_n = [L, U], \tag{2.1}$$

which is referred to as an interval equation, will be used to denote the set of ordinary equations consisting of

$$\begin{aligned} &a_1X_1 + a_2X_2 + \cdots + a_{n-1}X_{n-1} + a_nX_n = L, \\ &a_1X_1 + a_2X_2 + \cdots + a_{n-1}X_{n-1} + a_nX_n = L + 1, \\ &\qquad\qquad\vdots \\ &a_1X_1 + a_2X_2 + \cdots + a_{n-1}X_{n-1} + a_nX_n = U. \end{aligned}$$

An interval equation (2.1) will be said to be integer solvable if one of the equations in the set, which it defines, is integer solvable. The immediate way to determine this is to test if an integer in between L and U is divisible by the GCD of the coefficients of the left-hand-side terms. If $L > U$ in an interval equation (2.1), then there are no integer solutions for the interval equation. If the expression on the left-hand side of an interval equation (2.1) is reduced to zero items, in the processing of testing, then the interval equation will be said to be integer solvable if and only if $U \geqslant 0 \geqslant L$. The following definition and theorems, cited from [1], state how the I test determines integer solutions of an interval equation under constant bounds.

Definition 2.3. Let a variable a_i be an integer for $1 \leqslant i \leqslant n$. The positive part a_i^+ and the negative part a_i^- of an integer a_i are defined by $a_i^+ = \mathrm{MAX}\{a_i, 0\}$ and $a_i^- = \mathrm{MAX}\{-a_i, 0\}$.

Theorem 2.1. *Given a linear equation subject to the constraints of* (1.3). *Let* $a_1, a_2, \ldots, a_n$, *L, and U be integers. For each* r, $1 \leqslant r \leqslant n-1$, *let each of* $P_{r,0}$ *and* $Q_{r,0}$ *be either an integer or an unknown limit, where* $P_{r,0} \leqslant Q_{r,0}$ *if both* $P_{r,0}$ *and* $Q_{r,0}$ *are integers. Let* $P_{n,0}$ *and* $Q_{n,0}$ *be integers, where* $P_{n,0} \leqslant Q_{n,0}$. *If* $|a_n| \leqslant U - L + 1$, *then the interval equation*

$$a_1X_1 + a_2X_2 + \cdots + a_nX_n = [L, U]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0};\ 1 \leqslant r \leqslant n)$-*integer solvable if and only if the interval equation*

$$a_1X_1 + a_2X_2 + \cdots + a_{n-1}X_{n-1} = [L - a_n^+Q_{n,0} + a_n^-P_{n,0},\ U - a_n^+P_{n,0} + a_n^-Q_{n,0}]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0};\ 1 \leqslant r \leqslant n-1)$-*integer solvable.*

Theorem 2.2. *Given a linear equation subject to the constraints of* (1.3). *Let* $a_1, a_2, \ldots, a_n$, *L, and U be integers. For each r,* $1 \leqslant r \leqslant n$, *let each of* $P_{r,0}$ *and* $Q_{r,0}$ *be either an integer or an unknown limit, where* $P_{r,0} \leqslant Q_{r,0}$ *if both* $P_{r,0}$ *and* $Q_{r,0}$ *are integers. Let* $g = \gcd(a_1, \ldots, a_{n-1}, a_n)$. *The interval equation*

$$a_1X_1 + a_2X_2 + \cdots + a_{n-1}X_{n-1} + a_nX_n = [L, U]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0};\ 1 \leqslant r \leqslant n)$*-integer solvable if and only if the interval equation*

$$(a_1/g)X_1 + (a_2/g)X_2 + \cdots + (a_{n-1}/g)X_{n-1} + (a_n/g)X_n = [\lceil L/g \rceil, \lfloor U/g \rfloor]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0};\ 1 \leqslant r \leqslant n)$*-integer solvable.*

2.2. The Lambda test

Coupled references are groups of reference positions sharing one or more index variables [2,6]. Geometrically, each linear equation in (1.1) defines a hyperplane π in $\mathbb{R}^n$ spaces. The intersection S of m hyperplanes corresponds to the common solutions to all linear equations in (1.1). Obviously, if S is empty then there is no data dependence. Inspecting whether S is empty is trivial in linear algebra [8]. The bounds of (1.3) define a bounded convex set V in $\mathbb{R}^n$. If any of the hyperplanes in (1.1) does not intersect V, then obviously S cannot intersect V. However, even if every hyperplane in (1.1) intersects V, it is still possible that S and V are disjoint. If S and V are disjoint, then there exists a hyperplane which contains S and is disjoint from V. Furthermore, this hyperplane is a linear combination of hyperplanes in (1.1). On the other hand, if S and V intersect, then no such linear combination exists [2]. In general, the Banerjee inequalities [5] are first applied to test each hyperplane in (1.1). If every hyperplane intersects V, then the Lambda test is employed to simultaneously check every hyperplane.The Lambda test is an efficient data dependence method to deal with (1.1) beneath V. The Lambda test is actually equivalent to the multi-dimensional Banerjee inequality because it can determine simultaneous constrained real-valued solutions. The test forms linear combinations of coupled references that eliminate one or more instances of index variables when direction vectors are not considered. Simultaneous constrained real-valued solutions exist if and only if the Banerjee inequalities find solutions in all the linear combinations generated [2].

3. The multi-dimensional I test

Given the data dependence problem of multi-dimensional arrays with linear subscripts and constant bounds, we propose a multi-dimensional version of the I test – the multi-dimensional I test. The multi-dimensional I test examines a system of linear equations and deduces whether the system has integer-valued solutions. The linear equations have to be first transformed by the multi-dimensional I to their corresponding interval equations. That is, the interval equations

$$a_{1,1}X_1 + a_{1,2}X_2 + \cdots + a_{1,n-1}X_{n-1} + a_{1,n}X_n = [a_{1,0}, a_{1,0}],$$
$$\vdots$$
$$a_{m,1}X_1 + a_{m,2}X_2 + \cdots + a_{m,n-1}X_{n-1} + a_{m,n}X_n = [a_{m,0}, a_{m,0}]$$

have to be obtained from the linear equations (1.1). It is straightforward that the linear equations are integer solvable if and only if its corresponding interval equations are integer solvable. In this section, the theoretical aspects and the worst-case time complexity of the multi-dimensional I test are provided.

Assume that there are m interval equations written as:

$$\begin{aligned} a_{1,1}X_1 + a_{1,2}X_2 + \cdots + a_{1,n-1}X_{n-1} + a_{1,n}X_n &= [L_1, U_1], \\ &\vdots \\ a_{m,1}X_1 + a_{m,2}X_2 + \cdots + a_{m,n-1}X_{n-1} + a_{m,n}X_n &= [L_m, U_m], \end{aligned} \tag{3.1}$$

where each $a_{i,j}$ is a constant integer for $1 \leqslant i \leqslant m$ and $1 \leqslant j \leqslant n$. The constraints to each variable in (3.1) are postulated to be

$$P_{r,0} \leqslant X_r \leqslant Q_{r,0}, \tag{3.2}$$

where $P_{r,0}$ and $Q_{r,0}$ are constant integers for $1 \leqslant r \leqslant n$. Let F_i be the ith interval equation in (3.1). Geometrically, F_i consists of $U_i - L_i + 1$ linear equations in which each linear equation is parallel each other. Hence, F_i contains $U_i - L_i + 1$ hyperplanes in which each hyperplane is parallel each other. The intersection S of m interval equations corresponds to the common solutions to all interval equations in (3.1). Obviously, if S is empty then there is no data dependence. The bounds of (3.2) define a bounded convex set V in $\mathbb{R}^n$. If any of the interval equations in (3.1) does not intersect V, then obviously S cannot intersect V. However, even if every interval equation in (3.1) intersects V, it is still possible that S and V are disjoint. It is assumed that two interval equations in (3.1), respectively, intersect V. But the intersection of them is outside of V. If one can find a new interval equation which contains S but is disjoint from V, then it immediately follows that S and V do not intersect. The following theorem is an extension of Theorem 1 in [2] and guarantees that if S and V are disjoint, then there must be an interval equation which consists of S and is disjoint from V. Furthermore, this interval equation is a linear combination of equations in (3.1). On the other hand, if S and V intersect, then no such linear combination exists.

Theorem 3.1. *$S \cap V = \emptyset$ if and only if there exists an interval equation, β, only consisting of one linear equation, which corresponds to a linear combination of equations in* (3.1):

$$\left\langle \sum_{i=1}^{m} \lambda_i * \vec{a}_i, \vec{X} \right\rangle = \left[\sum_{i=1}^{m} \lambda_i * a_{i,0}, \sum_{i=1}^{m} \lambda_i * a_{i,0} \right],$$

where $L_i \leqslant a_{i,0} \leqslant U_i$ for $1 \leqslant i \leqslant m$ such that $\beta \cap V = \emptyset$. $\langle \vec{a}_i, \vec{X} \rangle$ denotes the inner product of $\vec{a}_i = (a_{i,1}, \ldots, a_{i,n})$ and $\vec{X} = (X_1, \ldots, X_n)$.

Proof. ($\Leftarrow$) The interval equation, β, contains S and is disjoint from V. So we can immediately derive that S is disjoint from V.

($\Rightarrow$) For the convenience of the proof, (3.1) are rewritten as $A * \vec{Y} = O$, where

$$A = \begin{pmatrix} -a_{1,0} & a_{1,1} & \cdots & a_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{m,0} & a_{m,1} & \cdots & a_{m,n} \end{pmatrix}_{m\times(n+1)}, \quad \vec{Y} = \begin{pmatrix} 1 \\ X_1 \\ \vdots \\ X_n \end{pmatrix}_{(n+1)\times 1},$$

O is an $m \times 1$ zero matrix, and $L_i \leqslant a_{i,0} \leqslant U_i$ for $1 \leqslant i \leqslant m$. We can let $S = \{(X_1, \ldots, X_n) : A\vec{Y} = O\}$, $V = \{(X_1, \ldots, X_n) : P_{r,0} \leqslant X_r \leqslant Q_{r,0}$ for $1 \leqslant r \leqslant n\}$, $S' = \{(1, X_1, \ldots, X_n) : \forall (X_1, \ldots, X_n) \in S\}$, and $V' = \{(1, X_1, \ldots, X_n) : \forall (X_1, \ldots, X_n) \in \mathrm{V}\}$. Because $S \cap V = \emptyset$, we can infer that $S' \cap V' = \emptyset$.

We let $\alpha = \mathrm{Span}(\vec{b}_1, \ldots, \vec{b}_m)$, where $\vec{b}_i = (-a_{i,0}, a_{i,1}, \ldots, a_{i,n})$. For all $\vec{C} \in \alpha$ and $\vec{D} \in S'$, we can obtain the inner product of $\vec{C}$ and $\vec{D}$ as follows:

$$\begin{aligned} \langle \vec{C}, \vec{D} \rangle &= \left\langle \sum_{i=1}^{m} \lambda_i * \vec{b}_i, \vec{D} \right\rangle \\ &= \lambda_1(-a_{1,0} + a_{1,1}X_1 + \cdots + a_{1,n}X_n) + \cdots \\ &\quad + \lambda_m(-a_{m,0} + a_{m,1}X_1 + \cdots + a_{m,n}X_n) \\ &= \lambda_1(0) + \cdots + \lambda_m(0) \\ &= 0. \end{aligned}$$

Therefore, we can at once derive that α is the orthogonal complementary space of S'. For any $\vec{Z}$ in V', consider $\vec{P}_Z$, the projection of $\vec{Z}$ on S'. Since $\|\vec{P}_Z - \vec{Z}\|$ is a continuous function on V' and V' is bounded, there must exist $\vec{Z}_0$ in V' such that $\|\vec{P}_{Z_0} - \vec{Z}_0\| = \min_{\vec{Z}\in V'} \|\vec{P}_Z - \vec{Z}\|$. This is the minimum distance between S' and V'. Since $\vec{Z}_0 - \vec{P}_{Z_0}$ is orthogonal to S', it must be in α. Hence, the equation $\langle \vec{Z}_0 - \vec{P}_{Z_0}, \vec{D} \rangle = 0$ is a linear combination of equations in (3.1), i.e., $\vec{Z}_0 - \vec{P}_{Z_0} = \lambda_1 * \vec{b}_1 + \cdots + \lambda_m * \vec{b}_m$. The equation $\langle \vec{Z}_0 - \vec{P}_{Z_0}, \vec{D} \rangle = 0$ is actually equal to

$$\left\langle \sum_{i=1}^{m} \lambda_i * \vec{a}_i, \vec{X} \right\rangle = \sum_{i=1}^{m} \lambda_i * a_{i,0}.$$

Therefore, the equation

$$\left\langle \sum_{i=1}^{m} \lambda_i * \vec{a}_i, \vec{X} \right\rangle = \sum_{i=1}^{m} \lambda_i * a_{i,0}$$

is transformed to one new interval equation

$$\left\langle \sum_{i=1}^{m} \lambda_i * \vec{a}_i, \vec{X} \right\rangle = \left[\sum_{i=1}^{m} \lambda_i * a_{i,0}, \sum_{i=1}^{m} \lambda_i * a_{i,0} \right].$$

Let β be the new interval equation. Hence, we can immediately conclude that the new interval equation, β, contains S. Since $\langle \vec{Z}_0 - \vec{P}_{Z_0}, \vec{Z} \rangle > 0$ for any $\vec{Z}$ in V' (refer to [2]), so each element $\vec{X}$ in V satisfies

$$\left\langle \sum_{i=1}^{m} \lambda_i * \vec{a}_i, \vec{X} \right\rangle > \sum_{i=1}^{m} \lambda_i * a_{i,0}.$$

Therefore, we can at once derive $\beta \cap V = \emptyset$. □

An array $(\lambda_1, \ldots, \lambda_m)$ in Theorem 3.1 determines an interval equation that contains S. There are infinite number of such interval equations. The tricky part in the multi-dimensional I test is to examine as few interval equations as necessary to determine whether S and V intersect. We start from the case of $m = 2$, both for convenience of presentation and for practical importance of two-dimensional arrays [9].

3.1. The case of two-dimensional array references

In the case of two-dimensional array references, two interval equations in (3.1) are $F_1 = [L_1, U_1]$ and $F_2 = [L_2, U_2]$, where $F_i = a_{i,1}X_1 + \cdots + a_{i,n}X_n$ for $1 \leqslant i \leqslant 2$. An arbitrary linear combination of the two interval equations can be written as

$$\lambda_1 F_1 + \lambda_2 F_2 = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}],$$

where $L_1 \leqslant a_{1,0} \leqslant U_1$ and $L_2 \leqslant a_{2,0} \leqslant U_2$. The domain of (λ_1, λ_2) is the whole $\mathbb{R}^2$ space. Let

$$F_{\lambda_1,\lambda_2} = \lambda_1 F_1 + \lambda_2 F_2 = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}],$$

that is

$$F_{\lambda_1,\lambda_2} = -(\lambda_1 a_{1,0} + \lambda_2 a_{2,0}) + (\lambda_1 a_{1,1} + \lambda_2 a_{2,1})X_1 + \cdots + (\lambda_1 a_{1,n} + \lambda_2 a_{2,n})X_n = 0.$$

By [2], F_{λ_1,λ_2} is viewed in two ways. With (λ_1, λ_2) fixed, F_{λ_1,λ_2} is a linear function of $(X_1, \ldots, X_n)$ in $\mathbb{R}^n$. With $(X_1, \ldots, X_n)$ fixed, it is a linear function of (λ_1, λ_2) in $\mathbb{R}^2$. Furthermore, the coefficient of each variable in F_{λ_1,λ_2} is a linear function of (λ_1, λ_2) in $\mathbb{R}^2$, i.e., $\Psi^{(i)} = \lambda_1 a_{1,i} + \lambda_2 a_{2,i}$ for $1 \leqslant i \leqslant n$. The equation $\Psi^{(i)} = 0$, $1 \leqslant i \leqslant n$, is called a Ψ equation. Each Ψ equation corresponds to a line in $\mathbb{R}^2$, which is called a Ψ line. Each Ψ line separates the whole space into two closed halfspaces $\Psi_i^+ = \{(\lambda_1, \lambda_2) \mid \Psi^{(i)} \geqslant 0\}$ and $\Psi_i^- = \{(\lambda_1, \lambda_2) \mid \Psi^{(i)} \leqslant 0\}$ that intersect at the Ψ line.

A nonempty set $C \subset \mathbb{R}^m$ is a cone if $\varepsilon\vec{\lambda} \in C$ for each $\vec{\lambda} \in C$ and $\varepsilon \geqslant 0$ [8]. It is obvious that each cone contains the zero vector. Moreover, a cone that includes at least one nonzero vector $\vec{\lambda}$ must consist of the ray of $\vec{\lambda}$, namely $\{\varepsilon\vec{\lambda} \mid \varepsilon \geqslant 0\}$. Such cones can clearly be viewed as the union of rays. There are at most n Ψ lines which together divide $\mathbb{R}^2$ into at most $2n$ regions. Each region contains the zero vector. Any one nonzero element $\vec{\lambda}$ and the zero vector in the region form the ray of $\vec{\lambda}$, namely $\{\varepsilon\vec{\lambda} \mid \varepsilon \geqslant 0\}$. Therefore, each region can be viewed as the union of rays. It is very obvious from the definition of cone that each region is a cone [8].

In the following, Lemmas 3.1–3.3 are extended from [1,2]. Definitions 3.1 and 3.2 are cited from [2] directly.

Lemma 3.1. *Suppose that a bounded convex set V is defined simply by the limits of* (3.2). (*The dependence directions will not be taken account of.*) *If*

$$F_{\lambda_1,\lambda_2} = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$*-integer solvable for every* (λ_1, λ_2) *in every* Ψ *line, then*

$$F_{\lambda_1,\lambda_2} = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}]$$

is also $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$*-integer solvable for every* (λ_1, λ_2) *in* $\mathbb{R}^2$.

Proof.
1. From the I test in [1], because

$$F_{\lambda_1,\lambda_2} = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$-integer solvable for every (λ_1, λ_2) in every Ψ line, there must be at least one element in V such that $F_{\lambda_1,\lambda_2} - (\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}) = 0$.
2. We have that $F_{\lambda_1,\lambda_2} - (\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}) = 0$ for any point (λ_1, λ_2) on every Ψ line according to the assumption of the lemma. It is immediately concluded that

$$F_{\lambda_1,\lambda_2} = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$-integer solvable for every point (λ_1, λ_2) on the boundaries of each cone.
3. Every point in each cone can be expressed as a linear combination of some points on the boundary of the same cone, as being a well-known fact in the convex theory. Any point (λ_5, λ_6) in a cone is assumed to be capable of being represented as $(\varepsilon\lambda_1 + \tau\lambda_3, \varepsilon\lambda_2 + \tau\lambda_4)$, where (λ_1, λ_2) and (λ_3, λ_4) are points in the boundary of the cone and $\varepsilon \geqslant 0$ and $\tau \geqslant 0$. Because

$$\begin{aligned}
&F_{\lambda_5,\lambda_6}(X_1, \ldots, X_n) - (\lambda_5 a_{1,0} + \lambda_6 a_{2,0}) \\
&\quad = F_{\varepsilon\lambda_1+\tau\lambda_3,\varepsilon\lambda_2+\tau\lambda_4}(X_1, \ldots, X_n) - (\varepsilon\lambda_1 + \tau\lambda_3)a_{1,0} - (\varepsilon\lambda_2 + \tau\lambda_4)a_{2,0} \\
&\quad = \varepsilon * (F_{\lambda_1,\lambda_2}(X_1, \ldots, X_n) - (\lambda_1 a_{1,0} + \lambda_2 a_{2,0})) + \tau * (F_{\lambda_3,\lambda_4}(X_1, \ldots, X_n) \\
&\qquad - (\lambda_3 a_{1,0} + \lambda_4 a_{2,0})) \\
&\quad = \varepsilon * 0 + \tau * 0 = 0,
\end{aligned}$$

we thus secure that

$$F_{\lambda_5,\lambda_6} = [\lambda_5 * a_{1,0} + \lambda_6 * a_{2,0}, \lambda_5 * a_{1,0} + \lambda_6 * a_{2,0}]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$-integer solvable for any point (λ_5, λ_6) in each cone. Of course it is also true in the whole $\mathbb{R}^2$ space. Therefore, for any point (λ_1, λ_2) in $\mathbb{R}^2$ space,

$$F_{\lambda_1,\lambda_2} = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$-integer solvable in $\mathbb{R}^n$ space. □

It is indicated from Lemma 3.1 and Theorem 2.1 that variables in an interval equation can be moved to the right if the coefficients of variables have small enough values to justify the movement. If all coefficients for variables in an interval equation have no sufficiently small values to justify the movements, then Lemma 3.1 and Theorem 2.1 cannot be applied. While every variable in an interval equation cannot be moved to the right, Lemma 3.2 describes a transformation using the GCD test which may enable additional variables to be moved.

Lemma 3.2. *Suppose that a bounded convex set V is defined simply by the limits of* (3.2). *Let*

$$g = \gcd(\lambda_1 a_{1,1} + \lambda_2 a_{2,1}, \ldots, \lambda_1 a_{1,n} + \lambda_2 a_{2,n}).$$

If

$$(1/g) * F_{\lambda_1,\lambda_2} = [\lceil(\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0})/g\rceil, \lfloor(\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0})/g\rfloor]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0};\ 1 \leqslant r \leqslant n)$*-integer solvable for every* (λ_1, λ_2) *in every* Ψ *line, then*

$$(1/g) * F_{\lambda_1,\lambda_2} = [\lceil(\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0})/g\rceil, \lfloor(\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0})/g\rfloor]$$

is also $(P_{r,0} \leqslant X_r \leqslant Q_{r,0};\ 1 \leqslant r \leqslant n)$*-integer solvable for every* (λ_1, λ_2) *in* $\mathbb{R}^2$.

Proof. Similar to Lemma 3.1. □

Lemma 3.3. *Suppose that a bounded convex set V is denoted by the limit of* (3.2). *Let*

$$g = \gcd(\lambda_1 a_{1,1} + \lambda_2 a_{2,1}, \ldots, \lambda_1 a_{1,n} + \lambda_2 a_{2,n}).$$

Given a line in $\mathbb{R}^2$ *corresponding to an equation* $a\lambda_1 + b\lambda_2 = 0$, *if*

$$F_{\lambda_1,\lambda_2} = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}]$$

or

$$(1/g) * F_{\lambda_1,\lambda_2} = [\lceil(\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0})/g\rceil, \lfloor(\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0})/g\rfloor]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0};\ 1 \leqslant r \leqslant n)$*-integer solvable in* $\mathbb{R}^n$ *space for any fixed point* $(\lambda_1^0, \lambda_2^0) \neq (0, 0)$ *in the line, then for every* (λ_1, λ_2) *in the line,*

$$F_{\lambda_1,\lambda_2} = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}]$$

or

$$(1/g) * F_{\lambda_1,\lambda_2} = [\lceil(\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0})/g\rceil, \lfloor(\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0})/g\rfloor]$$

is also $(P_{r,0} \leqslant X_r \leqslant Q_{r,0};\ 1 \leqslant r \leqslant n)$*-integer solvable in* $\mathbb{R}^n$ *space.*

Proof. Similar to Lemma 3.1. □

Definition 3.1. Given an equation of the form $a\lambda_1 + b\lambda_2 = 0$ where a, b are not zero simultaneously, a canonical solution of the equation is defined as follows:

$(\lambda_1, \lambda_2) = (1, 0)$ if $a = 0$;
$(\lambda_1, \lambda_2) = (0, 1)$ if $b = 0$;
$(\lambda_1, \lambda_2) = (b, -a)$ if neither of a, b is zero;
$(\lambda_1, \lambda_2) = (1, 1)$ if both of a and b are zero.

Definition 3.2. The Λ set is denoted to be the set of all canonical solutions to Ψ equations. Each element, (λ_1, λ_2), in the Λ set corresponds to one interval equation

$$F_{\lambda_1,\lambda_2} = [\lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}, \lambda_1 * a_{1,0} + \lambda_2 * a_{2,0}].$$

There are at most n Ψ equations if V is denoted by the bounds of (3.2). Each of the Ψ equations generates a canonical solution according to Definition 3.1. Each canonical solution forms a new interval equation, only containing the only linear equation in light of Definition 3.2. Obviously, new interval equations tested are at most n if V is defined by the constraints of (3.2).

The multi-dimensional I test is employed to simultaneously check every interval equation. It examines the subscripts from two dimensions, and then figures out the Λ set from Ψ equations. Each element in the Λ set determines a new interval equation. The new interval equation is tested to see if it intersects V, by moving variables in one interval equation as done in the I test for testing each *single* dimension.

We now use an example to explain how the multi-dimensional I test works. Consider the following equations:

$$X_1 - X_4 = 0,$$
$$-X_2 + X_3 = 0$$

subject to the constant bounds $1 \leqslant X_1, X_2, X_3, X_4 \leqslant 100$.

According to Definition 3.1, the Ψ equations have two canonical solutions $(0, 1)$ and $(1, 0)$. According to Definition 3.2, canonical solutions $(0, 1)$ and $(1, 0)$, respectively, yield the following interval equations:

$$-X_2 + X_3 = [0, 0], \tag{Ex1}$$
$$X_1 - X_4 = [0, 0]. \tag{Ex2}$$

Now the multi-dimensional I test applies Lemma 3.1 and Theorem 2.1 to resolve the interval equation (Ex1a), and the term $-X_2$ in the interval equation is moved to the right-hand side to gain the new interval equation

$$X_3 = [1, 100]. \tag{Ex1a}$$

Now the length of the right-hand side interval has been increased to 100, so Lemma 3.1 and Theorem 2.1 are again employed to move the term X_3 in the interval equation (Ex1a) to the right-hand side to acquire the new interval equation

$$0 = [-99, 99]. \tag{Ex1b}$$

Because $-99 \leqslant 0 \leqslant 99$, it is at once derived that the interval equation (Ex1) is integer solvable. Next, the multi-dimensional I test applies Lemma 3.1 and Theorem

2.1 to resolve the interval equation (Ex2), and the term $-X_4$ in the interval equation (Ex2) is moved to the right-hand side to gain the new interval equation

$$X_1 = [1, 100]. \tag{Ex2a}$$

Now the length of the right-hand side interval has been increased to 100, so Lemma 3.1 and Theorem 2.1 are again employed to move the term X_1 in the interval equation (Ex2a) to the right-hand side to acquire the new interval equation

$$0 = [-99, 99]. \tag{Ex2b}$$

Because $-99 \leqslant 0 \leqslant 99$, it is right away inferred that the interval equation (Ex2) is integer solvable. Therefore, the multi-dimensional I test in light of Lemmas 3.1–3.3 infers that there is integer-valued solution.

3.2. The case of multi-dimensional array references

We take account of m interval equations in (3.1) with $m > 2$ for generalizing the multi-dimensional I test. All m interval equations are assumed to be connected; otherwise they can be partitioned into smaller systems. As stated before, we can hypothesize that there are no redundant equations. An arbitrary linear combination of m interval equations in (3.1) can be written as

$$F_{\lambda_1,\ldots,\lambda_m} = \left\langle \sum_{i=1}^{m} \lambda_i * \vec{a}_i, \vec{X} \right\rangle = \left[\sum_{i=1}^{m} \lambda_i * a_{i,0}, \sum_{i=1}^{m} \lambda_i * a_{i,0} \right],$$

where $L_i \leqslant a_{i,0} \leqslant U_i$ for $1 \leqslant i \leqslant m$ and $\langle \vec{a}_i, \vec{X} \rangle$ denotes the inner product of $\vec{a}_i = (a_{i,1}, \ldots, a_{i,n})$ and $\vec{X} = (X_1, \ldots, X_n)$.

Assume that

$$g = \gcd\left(\sum_{i=1}^{m} \lambda_i * a_{i,1}, \ldots, \sum_{i=1}^{m} \lambda_i * a_{i,n} \right).$$

It is to be determined whether

$$F_{\lambda_1,\ldots,\lambda_m} = \left[\sum_{i=1}^{m} \lambda_i * a_{i,0}, \sum_{i=1}^{m} \lambda_i * a_{i,0} \right]$$

or

$$(1/g) * F_{\lambda_1,\ldots,\lambda_m} = \left[\left\lceil \left(\sum_{i=1}^{m} \lambda_i * a_{i,0} \right) \Big/ g \right\rceil, \left\lfloor \left(\sum_{i=1}^{m} \lambda_i * a_{i,0} \right) \Big/ g \right\rfloor \right]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$-integer solvable in $\mathbb{R}^n$ space for arbitrary $(\lambda_1, \ldots, \lambda_m)$. By [2], the coefficient of each variable in $F_{\lambda_1,\ldots,\lambda_m}$ is a linear function of $(\lambda_1, \ldots, \lambda_m)$ in $\mathbb{R}^m$, which is

$$\Psi^{(i)} = \sum_{j=1}^{m} \lambda_j a_{j,i} \quad \text{for } 1 \leqslant i \leqslant n.$$

The equation $\Psi^{(i)} = 0,\ 1 \leqslant i \leqslant n$, is called a Ψ equation. A Ψ equation corresponds to a hyperplane in $\mathbb{R}^m$, called a Ψ plane. Each Ψ plane divides the whole space into two closed halfspaces

$$\Omega_i^+ = \left\{(\lambda_1, \ldots, \lambda_m) \,\middle|\, \Psi^{(i)} \geqslant 0\right\} \quad \text{and} \quad \Omega_i^- = \left\{(\lambda_1, \ldots, \lambda_m) \,\middle|\, \Psi^{(i)} \leqslant 0\right\}.$$

If V is defined by the constraints of (3.2), then a nonempty set $\bigcap_{i=1}^{n} \Omega_i$, where $\Omega_i \in \{\Omega_i^+, \Omega_i^-\}$, is called a λ region. Every λ region is a cone in $\mathbb{R}^m$ space [8]. The λ regions in $\mathbb{R}^m$ space have several lines as the frame of their boundaries. Each line (called a λ line) is the intersection of some Ψ equations.

The following lemmas are extended from [1,2].

Lemma 3.4. *Suppose that a bounded convex set V is defined simply by the limits of* (3.2). *Let*

$$g = \gcd\left(\sum_{i=1}^{m} \lambda_i * a_{i,1}, \ldots, \sum_{i=1}^{m} \lambda_i * a_{i,n}\right).$$

If

$$F_{\lambda_1,\ldots,\lambda_m} = \left[\sum_{i=1}^{m} \lambda_i * a_{i,0}, \sum_{i=1}^{m} \lambda_i * a_{i,0}\right]$$

or

$$(1/g) * F_{\lambda_1,\ldots,\lambda_m} = \left[\left\lceil \left(\sum_{i=1}^{m} \lambda_i * a_{i,0}\right) \Big/ g \right\rceil, \left\lfloor \left(\sum_{i=1}^{m} \lambda_i * a_{i,0}\right) \Big/ g \right\rfloor\right]$$

is $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$*-integer solvable for every* $(\lambda_1, \ldots, \lambda_m)$ *in every* λ *line, then*

$$F_{\lambda_1,\ldots,\lambda_m} = \left[\sum_{i=1}^{m} \lambda_i * a_{i,0}, \sum_{i=1}^{m} \lambda_i * a_{i,0}\right]$$

or

$$(1/g) * F_{\lambda_1,\ldots,\lambda_m} = \left[\left\lceil \left(\sum_{i=1}^{m} \lambda_i * a_{i,0}\right) \Big/ g \right\rceil, \left\lfloor \left(\sum_{i=1}^{m} \lambda_i * a_{i,0}\right) \Big/ g \right\rfloor\right]$$

is also $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$*-integer solvable for every* $(\lambda_1, \ldots, \lambda_m)$ *in* $\mathbb{R}^m$ *space.*

Proof. Similar to Lemma 3.1. □

Lemma 3.5. *Given a line in* $\mathbb{R}^m$ *which crosses the origin of the coordinates and let*

$$g = \gcd\left(\sum_{i=1}^{m} \lambda_i * a_{i,1}, \ldots, \sum_{i=1}^{m} \lambda_i * a_{i,n}\right).$$

If

$$F_{\lambda_1,\ldots,\lambda_m} = \left[\sum_{i=1}^{m} \lambda_i * a_{i,0}, \sum_{i=1}^{m} \lambda_i * a_{i,0}\right]$$

or

$$(1/g) * F_{\lambda_1,\ldots,\lambda_m} = \left[\left\lceil \left(\sum_{i=1}^{m} \lambda_i * a_{i,0}\right) \Big/ g \right\rceil, \left\lfloor \left(\sum_{i=1}^{m} \lambda_i * a_{i,0}\right) \Big/ g \right\rfloor\right]$$

is $(P_{r,0} X_r \leqslant Q_{r,0})$*-integer solvable in* $\mathbb{R}^n$ *space for any fixed point* $(\lambda_1^0, \ldots, \lambda_m^0) \neq (0, \ldots, 0)$ *in the line, then for every* $(\lambda_1, \ldots, \lambda_m)$ *in the line,*

$$F_{\lambda_1,\ldots,\lambda_m} = \left[\sum_{i=1}^{m} \lambda_i * a_{i,0}, \sum_{i=1}^{m} \lambda_i * a_{i,0}\right]$$

or

$$(1/g) * F_{\lambda_1,\ldots,\lambda_m} = \left[\left\lceil \left(\sum_{i=1}^{m} \lambda_i * a_{i,0}\right) \Big/ g \right\rceil, \left\lfloor \left(\sum_{i=1}^{m} \lambda_i * a_{i,0}\right) \Big/ g \right\rfloor\right]$$

is also $(P_{r,0} \leqslant X_r \leqslant Q_{r,0})$*-integer solvable in* $\mathbb{R}^n$ *space.*

Proof. Similar to Lemma 3.3. □

The details of the multi-dimensional I test in the general case are not considered here since the discussion is similar to the case of $m = 2$.

3.3. Time complexity

The main phases for the multi-dimensional I test include (1) calculating λ values and (2) examining each interval equation. λ Values are easily determined according to Ψ equations and Definition 3.1. It is clear that the time complexity to compute a λ value is $O(y)$ from Definition 3.1, where y is a constant. Each λ value corresponds to an interval equation. Each interval equation is tested to see if it intersects V, by moving variables in left-hand side of one interval equation to the right-hand side of the interval equation as done in the I test for one single dimension. The worst-case time complexity of the I test is $O(n^2 * y + n * y)$ [1], where n is the number of variables in interval equations. Hence, the time complexity of for the multi-dimensional I test examining an interval equation is derived to be $O(n^2 * y + n^* y + y)$. The number of interval equations checked in the multi-dimensional I test is at most

$$\prod_{i=1}^{m} (U_i - L_i + 1) * \binom{n}{m-1},$$

where m is the number of *original* linear references and L_i and U_i are the lower and upper bounds in the right-hand side of original interval equations for $1 \leqslant i \leqslant m$, in light of statements in Sections 3.1 and 3.2 and [2]. Therefore, the worst-case time complexity for the multi-dimensional I test is immediately inferred to be

$$O\left(\begin{bmatrix} n \\ m-1 \end{bmatrix} * \left(n^2 * y + n * y + y\right) * \left(\prod_{i=1}^{m} (U_i - L_i + 1)\right)\right).$$

Two-dimensional arrays with linear subscripts appear quite frequently in real programs [9]. As the lower and upper bounds are initially the same in the right-hand side of an initial interval equations in linear references in real programs, therefore, the number of interval equations examined in each two-dimensional array tested is at most n according to statements in Section 3.1. If the multi-dimensional I test is applied to deal with two-dimensional arrays, then their worst-case time complexity is $O(n * (n^2 * y + n * y + y))$. The worst-case time complexity for the Lambda test dealing with the same array is $O((3n/2) * (n + y))$. However, in general, the efficiency of the multi-dimensional I test is only slightly poorer than that of the Lambda test and the I test because the number of variables, n, in the interval equation tested is generally very small.

4. Experimental results

We tested the multi-dimensional I test and performed experiments for the benchmark codes cited from one numerical package SPEC77 in Perfect Benchmark [4,7]. There are totally 276 pairs of multi-dimensional array references found in the tested package. Of the 276 pairs of multi-dimensional array references, 220, 7, and 49 pairs were observed to have constant bounds, *variable* constraints, and *symbolic* (*unknown*) limits, subsequently. The multi-dimensional I test is only applied to test 220 arrays with constant bounds.

The results obtained (Table 1) reveal the multi-dimensional I test determined that there were integer-valued solutions for 188 pairs of multi-dimensional arrays with constant bounds. The "accuracy rate" in Table 1 refers to, when given a set of multi-dimensional arrays with constant bounds, how often the multi-dimensional I test detects a case where there is an integer-valued solution. Let b be the number of multi-dimensional arrays with constant bounds found in our experiments, and c be the number that is detected to have integer-valued solutions. Thus the accuracy rate is denoted to be equal to c/b. In our experiments, 220 pairs of array references were found to have constant limits, and 188 of them were found to have integer-valued solutions. So the accuracy rate for the multi-dimensional I test was about 85.4%. Similarly, the "improvement rate" refers to how often the multi-dimensional I test

Table 1
Testing capability of the multi-dimensional I test for 276 pairs of multi-dimensional array references from Perfect Benchmark

Loop bounds	Pairs of multi-dimensional arrays	Testing results (pairs)		Accuracy rate	Improvement rate
		Definitive	Maybe		
Constant bounds	220	188	32	85.4%	68.1%
Variable bounds	7	–	–	–	–
Symbolic bounds	49	–	–	–	–

gives a definitive (yes/no) result for a set of multi-dimensional arrays with constant, variable and symbolic bounds. Let d be the number of multi-dimensional arrays with constant, variable, and symbolic bounds found in our experiments. Thus the improvement rate is denoted to be equal to c/d. In our experiments, 276 pairs of array references were found to have multi-dimensional linear references, and 188 of them were found to have *definite* (*yes* or *no*) results. So the improvement rate for the multi-dimensional I test was equal to 68.1%.

We also implemented the Omega test and the Power test based on [3,6] to compare their effects with those of the multi-dimensional I test. The Omega test and Power test were applied to resolve 220 pairs of multi-dimensional arrays with constant bounds. These two tests were found to obtain the same accurate results as the multi-dimensional I test. Let k_{MI}, k_{P}, and k_{O} be the execution time to treat data dependence problem of a multi-dimensional array for the multi-dimensional I test, the Power test, and the Omega test, subsequently. The speed-up in Table 2 is defined to be the set of $k_{\mathrm{P}}/k_{\mathrm{MI}}$ and $k_{\mathrm{O}}/k_{\mathrm{MI}}$. Each row in Table 2 shows how many times the execution time of the Power test and the Omega test took longer than that of the multi-dimensional I test. For example, the first row shows that there are 10 subroutines in which the execution time of the Power test took from 5.3 to 8.9 times longer than that of the multi-dimensional I test. This table indicates that for multi-dimensional arrays with constant bounds the efficiency of the multi-dimensional I test is much better than that of the Power test and the Omega test.

The superiority of testing efficiency of the multi-dimensional I test over that of the Omega test for the stated dependence problem can also be deduced from time complexity analysis. The Omega test based on the least remainder algorithm, a variation of Euclid's algorithm, and Fourier's elimination method [3,10] consists of three major computations: eliminating equality constraints, eliminating variables in inequality constraints, and finding integer solutions (that is an integer programming problem). The time complexities for these steps are $\mathrm{O}(mn\log|c| + mnp + mn)$, $\mathrm{O}(n^2s^2)$, and $\mathrm{O}(k^n)$ [3,10,11], respectively, where m, n, c, p, s, and k denote the number of equality constraints, the number of variables, the coefficient with the largest absolute value in equality constraints, the number of passes to eliminate all the variables that become unbound, the number of inequality constraints, and the absolute value of coefficient of variable in inequality constraints, subsequently. So the overall time complexity of the Omega test is $\mathrm{O}(mn\log|c| + mnp + mn + n^2s^2 + k^n)$. Obviously, compared with the time complexity analysis shown in

Table 2
The speed-up of the multi-dimensional I test to the Power test and the Omega test for 188 pairs of multi-dimensional arrays with constant bounds from Perfect Benchmark

	Speed-up	Total number of subroutines involved
$k_{\mathrm{P}}/k_{\mathrm{MI}}$	5.3–8.9	10
$k_{\mathrm{P}}/k_{\mathrm{MI}}$	10.5–14.8	13
$k_{\mathrm{O}}/k_{\mathrm{MI}}$	6.1–9.9	9
$k_{\mathrm{O}}/k_{\mathrm{MI}}$	13.1–19.2	14

Section 3, the multi-dimensional I test is significantly superior to that of the Omega test in terms of testing efficiency. In [3] it is reported that the Omega test has *exponential* worst-case time complexity. Wolfe and Tseng [6] and Triolet et al. [12] also found that Fourier–Motzkin variable elimination for dependence testing takes from 22 to 28 times longer than the Banerjee method, that is a part of the multi-dimensional I test.

The study in [3] stated that: (1) the cost of scanning array subscripts and loop bounds to build a dependence problem was typically 2–4 times of the copying cost (the cost of building a system of dependence equations) for the problem, and (2) the dependence analysis cost for more than half of *simple* arrays tested was typically 2–4 times of the copying cost, but the dependence analysis cost for other simple arrays and all of the *regular*, *convex*, and *complex* arrays tested was more than 4 times of the copying cost. Based on such results we can figure out that, for simple arrays, the analysis cost of data dependence for parallelizing/vectorizing compilation occupies generally about 29–57% of total compiling time. But, for complex arrays the analysis cost of dependence testing takes more than 57% of total compiling time. Therefore, enhancing dependence testing performance may result in a significant improvement in the compiling performance of a parallelizing/vectorizing compiler.

5. Conclusions

When testing array references with multi-dimensional linear subscripts and constant bounds, the Lambda test can determine whether real-valued solutions exist. As we know in dependence analysis a testing strategy concluding the existence of real-valued solutions may sometimes lose the accuracy and results in false dependency. In this paper we propose the multi-dimensional I test. The multi-dimensional I test can ascertain whether integer-valued solutions exist for multi-dimensional array references with linear subscripts and constant bounds. Obviously, the significance of the multi-dimensional I test lies in that it enhances the testing precision, eliminates the possible false dependency and exploits the degree of loop parallelization and vectorization.

The Power test is a combination of Fourier–Motzkin variable elimination with an extension of Euclid's GCD algorithm [6]. The Omega test combines new methods for eliminating equality constraints with an extension of Fourier–Motzkin variable elimination [3]. The two tests currently have the highest precision and the widest applicable range in the field of data dependence analysis for arrays with linear subscripts. Such a fact is also reflected in our experimental results. However, the cost of the two tests is very expensive [3,6]. It is found in our experiment that the Power test takes 5.3–14.8 times longer in execution than the multi-dimensional I test and the Omega test takes 6.1–19.2 times longer in execution than the multi-dimensional I test when testing the dependence of multi-dimensional arrays.

According to the time complexity analysis, the multi-dimensional I test performs slightly poorer than that of the Lambda test. Therefore, it is suggested that depending on the application domains, the multi-dimensional I test can be applied independently or together with the Power test or the Omega test to analyze data dependence for multi-dimensional array references.

Acknowledgements

This work was partially supported by the National Science Council of the Republic of China under grant NSC89-2213-E-168-013.

References

[1] X. Kong, D. Klappholz, K. Psarris, The I test, IEEE Transaction on Parallel and Distributed Systems 2 (3) (1991) 342–359.
[2] Z. Li, P.-C. Yew, C.-Q. Zhu, An efficient data dependence analysis for parallelizing compilers, IEEE Transaction on Parallel and Distributed Systems 1 (1) (1990) 26–34.
[3] W. Pugh, A practical algorithm for exact array dependence analysis, Communication of the ACM 35 (8) (1992) 102–114.
[4] R. Eigenmann, J. Hoeflinger, D. Padua, On the automatic parallelization of the Perfect Benchmarks, IEEE Transactions on Parallel and Distributed Systems 9 (1) (1998) 5–23.
[5] U. Banerjee, Dependence Analysis for Supercomputing, Kluwer Academic Publishers, Norwell, MA, 1988.
[6] M. Wolfe, C.W. Tseng, The power test for data dependence, IEEE Transaction on Parallel and Distributed Systems 3 (5) (1992) 591–601.
[7] Y. Paek, J. Hoeflinger, D. Padua, Simplification of array access patterns for compiler optimizations, ACM SIGPLAN'98, Conference on Programming Languages Design and Implementation (PLDI).
[8] Vaughan, W. Jeffrey, A Residuals Management Model of the Iron and Steel Industry: A Linear Programming Approach, Mich.: Univ. Microfilms International, Ann Arbor, 1986.
[9] Z. Shen, Z. Li, P.-C. Yew, An empirical study of Fortran programs for parallelizing compilers, IEEE Transaction on Parallel and Distributed Systems 1 (3) (1992) 356–364.
[10] U. Banerjee, Dependence Analysis, Kluwer Academic Publishers, Norwell, MA, 1997.
[11] U. Banerjee, Loop Transformations for Restructuring Compilers: The Foundations, Kluwer Academic Publishers, Dordrecht, 1993.
[12] R. Triolet, F. Irigoin, P. Feautrier, Direct parallelization of call statements, in: Proc. SIGPLAN Symp. Compiler Construction, Palo Alto, CA, 1986, pp. 176–185.