# データ依存解析のための二次不定方程式の分解

神戸和子[†] 加古富志雄[††]

自動並列化コンパイラ技術において，逐次ソースプログラムから並列性を検出するために多くのデータ依存解析手法が提案されてきた．ほとんどの手法は解析対象を一次式 $\Sigma_{i=1}^{n} a_i x_i + c$（$x_i$：整数変数，$a_i, c$：整数定数）に限定しているが，実際のプログラムコードには，二次の添え字式が頻繁に現れる．ベンチマークプログラム The Perfect Benchmarks や数値計算ライブラリ Eispack，Linpack，Lapack についてデータ依存解析を行った経験から，これらの二次式は三角行列を一次元化した配列参照において生じており，$X(X-1)/2+Y$，$1 \leq Y \leq X$（ただし $X$ と $Y$ は一次式）の形で表現できることが分かった．本論文では，この観察をもとに，二次不定方程式を一次不定方程式に分解する手法について述べる．一次不定方程式に分解してデータ依存解析を行うことで，効率的でより精度の高い解析結果が得られる．

# Reduction of Quadratic Equations for Data Dependence Tests

Kazuko Kambe[†] and Fujio Kako[††]

A number of data dependence tests have been proposed for detecting parallelism in sequential source programs. Despite the fact that quadratic subscript expressions often appear in real-life programs, most dependence tests restrict array subscripts to be linear functions of loop index variables $\Sigma_{i=1}^{n} a_i x_i + c$ ($x_i$: interger variables, $a_i, c$: an integer constant). The data dependence analyses of The Perfect Benchmarks and scientific libraries Eispack, Linpack and Lapack have shown that quadratic expressions appear only in array references of linearized triangular matrixes, and they can be expressed in the special form $X(X-1)/2+Y$, $1 \leq Y \leq X$ where $X$ and $Y$ are linear expressions. This paper presents a technique that reduces such a quadratic equation to two linear equations. This improves accuracy and efficiency of data dependence analyses.

## 1. はじめに

自動並列化コンパイラは，入力として受け取った並列実行されることがまったく考慮されていない逐次ソースプログラムを解析し，それと等価な並列オブジェクトコードを生成するコンパイラである．逐次原始プログラムからより多くの並列性を検出するために，配列変数に関するデータ依存解析はきわめて重要である．多くのデータ依存解析手法は，解析の対象を一次の添え字式に限定しているが，実際には配列参照の添え字式が二次である例は多い．これらの二次式は，プログラマによって明示的に入力されていたり，あるいは解析精度の向上を目的とした誘導変数消去の結果生じている．誘導変数消去とは，ループのイタレーションごとに一定の値で増減するスカラ変数を，インデックス変数で置き換えるプログラムコード変換手法の 1 つである．

以下に例を示す．

```
IJ = 0
DO J = 1,N
  IK = 0
  DO K = 1,J
    JK = IJ + K
    KK = IK + K
    A[JK] = ...
      ... = A[KK]
    IK = IK + K
  ENDDO
  IJ = IJ + J
ENDDO
```

上記のプログラムコードから誘導変数 **JK** と **KK** を消去すると，下記のプログラムコードが得られる．

† 奈良女子大学大学院人間文化研究科
Graduate School of Human Culture, Nara Women's University
†† 奈良女子大学理学部
Faculty of Science, Nara Women's University

```
DO J = 1,N
  DO K = 1,J
    A[(J * J - J + 2 * K) / 2] = ...
      ... = A[(K * K + K) / 2]
  ENDDO
ENDDO
```

この場合のデータ依存解析問題は，

$$\begin{cases} (j^2 - j + 2k)/2 = (\hat{k}^2 + \hat{k})/2, \\ 1 \le j \le n,\ 1 \le k \le j,\ 1 \le \hat{k} \le j \end{cases}$$

を満たす整数解が存在するかどうかを判定することである．この二次不定方程式を直接解くことは非常に計算負荷が大きい．ほとんどの依存解析手法では保守的な判定をせざるをえず，データ依存が存在するという結果になってしまう．しかし，この問題は一次不定方程式に分解することが可能で，分解後の依存解析問題は，

$$\begin{cases} j = \hat{k}, \quad 1 \le j \le n, \quad 1 \le \hat{k} \le j, \\ k = \hat{k}, \quad 1 \le k \le j \end{cases}$$

となり，ループ繰越依存（loop carried dependence）が存在しないことが容易に分かる．

ベンチマークプログラム The Perfect Benchmarks と数値計算ライブラリ Eispack，Linpack，Lapack のプログラムコードを，二次形式の添え字式の出現頻度について調査した結果を**表 1** と**表 2** に示す．自動並列化コンパイラ Parafrase-2[1] を用いて誘導変数を消去した後のプログラムコードを，調査の対象としている．表 1 は，全体のプログラム数と，どれだけのプログラムが二次式を含んでいたかを示している．表 2 では，二次式が現れたプログラムのみについて，配列参照を含む代入文（配列に関するデータ依存解析の対象となる文）の総数，そのうち添え字式が二次式であるもの，および二次式の出現回数を記している．さらに詳細に調べた結果，これらの二次式は三角行列を一次元化した配列要素の参照において生じていること，ならびにすべて同じ形式 $X(X-1)/2+Y$，$1 \le Y \le X$ で表せることが分かった．

**表 1** 二次形式の添え字式を含むプログラム数
Table 1 Programes with quadratic subscript expressions.

| | プログラム数 | 二次の添え字式を含むプログラム数 |
|---|---|---|
| Perfect Benchmarks | — | 0 |
| Eispack | 76 | 2 |
| Linpack | 40 | 6 |
| Lapack | 1292 | 33 |

**表 2** 二次形式の添え字式の出現頻度
Table 2 Frequency of quadratic subscript expressions.

| | 配列参照を含む代入文の総数 | 二次の添え字式を含む代入文の総数 | 二次の添え字式の出現数 |
|---|---|---|---|
| Eispack | 26 | 9 | 10 |
| Linpack | 80 | 17 | 17 |
| Lapack | 553 | 187 | 234 |

本論文では，上記の観察をもとに限定した条件下で二次不定方程式を一次不定方程式に分解する手法を提案する．二次不定方程式に対応していない依存解析手法でも，一次式ならば適用することも可能であり，計算負荷も小さく，解析精度を高めることができる．

本論文の構成は以下のとおりである．2 章で，本論文で扱うデータ依存解析問題について定義し，二次不定方程式の解法について数学的観点から概説する．3 章で，分解アルゴリズムとアルゴリズムを導くために必要な定理を示し，いくつかの適用例をあげる．4 章では関連研究について述べ，5 章でまとめる．

## 2. 背　　景

本章では依存解析問題について定義をし，二次不定方程式の数学的アプローチによる解法について検討する．

### 2.1 データ依存解析

プログラム中のループを並列計算機で並列実行させるためには，ループ中の配列要素のデータ依存解析をする必要がある．以下のような多重ループモデルを考える．

```
DO i1 = L1, U1
  DO i2 = L2, U2
    ...
    DO in = Ln, Un
S1:   A[f1(i1,...,in),...,fm(i1,...,in)] = ...
S2:   ... = A[g1(i1,...,in),...,gm(i1,...,in)]
    ENDDO
    ...
  ENDDO
ENDDO
```

文 S1 の配列 **A** と S2 の **A** のうち少なくとも一方が代入文の左辺にあるとき，2 つの **A** の間の依存関係を調べることは，以下の不定方程式と不等式を満たす整数解 $i_j$ $(1 \le j \le n)$ を求める問題に帰着する[2),3)]．

$$\begin{cases} f_j(i_1, \ldots, i_n) = g_j(\hat{i}_1, \ldots, \hat{i}_n),\ 1 \le j \le m, \\ L_k \le i_k \le U_k,\ L_k \le \hat{i}_k \le U_k,\ 1 \le k \le n. \end{cases}$$

本論文では，イタレーション空間が三角領域，すなわち

$U_k$ がインデックス変数 $i_{k-1}$ あるいは $\hat{i}_{k-1}$ $(k>1)$ を含み，$m=1$ の二次不定方程式の依存解析問題を対象とする．

### 2.2 二次不定方程式の解法

二元二次不定方程式 $ax^2+bxy+cy^2=f$（ただし，$a\neq 0$，$c\neq 0$）について，この方程式を幾何学的に解釈すると，

$$\begin{cases} D<0 \text{ の場合} & \text{楕円,} \\ D=0 \text{ の場合} & \text{放物線,} \\ D>0 \text{ の場合} & \text{双曲線} \end{cases}$$

となり，曲線上の格子点（座標が整数値である点）が不定方程式の解である．ここで $D=b^2-4ac$ を判別式という．

- $D<0$ のとき
  楕円なので $x$（または $y$）の範囲は有限であり，この範囲に属する正または負の整数値を逐次 $x$（または $y$）に与えてすべての格子点を求めればよい．
- $D=0$ のとき
  この場合，不定方程式の左辺は完全平方式である．$a$，$b$，$c$ の最大公約数を $g$ とすれば，$f$ が $g$ で割り切れて，かつ $f/g$ が平方数であるならば，一次不定方程式の問題に帰する．
- $D>0$ のとき
  不定方程式は，$(2ax+by)^2-Dy^2=4af$ と同一である．
  - $D$ が平方数である場合は，左辺は 2 つの整係数の一次因数に分解できるので，一次不定方程式の問題に帰する．
  - $D$ が平方数でない場合は，一般には連分数展開を利用して解く．解があるならば，それらの解は有限個の同伴解に分かれる（詳しくは文献 4)，5) を参照されたい）．

上記の方法を基本として，$ax^2+bxy+cy^2+dx+ey+f=0$ を解くことに拡張できる[6]が，現実の依存解析問題における二次不定方程式は 3 つ以上の変数を含むもっと複雑なものもある．整数論の観点から論じると，多元高次不定方程式はいろいろの特殊例についても，また一般的にも多く論じられているが，これらは概して答を与えることがきわめて困難である．

しかし，プログラムコードの持つ特性を利用すれば一次不定方程式の問題に簡単化することができる．我々は対象とする式を，三角行列を一次元化した配列に現れる二次式に限定して，一次不定方程式に分解する方法をとる．

## 3. 二次不定方程式の分解

本章では，三角行列の要素が一次元化された配列において，$X(X-1)/2+Y$，$1\leq Y\leq X$（ただし $X$ と $Y$ は一次式）の形で表現できることを示し，これを利用した二次不定方程式の分解アルゴリズムについて述べる．また，いくつかの適用例を示す．

### 3.1 三角行列と二次不定方程式

**図 1** は下三角行列の $(x,y)$ 要素が一次元配列においてどの要素に対応しているかを示している．一次元化するときに優先させる方向によって，対応する一次元配列の要素を表す添え字式は，

(A) $y$ 方向を優先　　$(x(x-1)/2+y)$
(B) $x$ 方向を優先　　$((y-1)(2n-y)/2+x)$

となる☆．

(A) と (B) の場合では，添え字式はまったく異なるように見える．(B) において一次元配列の任意の要素 $I$ に対して，$I\rightarrow n(n+1)/2+1-I$ という変換を考えると，

$$\begin{aligned} &n\,(n+1)\,/2+1-((y-1)\,(2n-y)\,/2+x) \\ &\quad=(n-y+1)\,(n-y)\,/2+n-x+1, \end{aligned}$$

$1\leq n-x+1\leq n-y+1$ となる．ここで $n-y+1\rightarrow X$，$n-x+1\rightarrow Y$ とおいてやると，(A) における一次元配列要素の添え字式と同じ形になっていることに気づく．$I\rightarrow n(n+1)/2+1-I$ という変換は，三角行列では要素を $(x,y)\rightarrow(n-y+1,n-x+1)$ で変換させることに相当している．また，上三角行列は下三角行列を転置したものなので，(A)，(B) と同様に考えることができる．

以上のことから，三角行列における $(X,Y)$ 要素は一次元化した配列において，以下の形で表せることが分かる．

$$\begin{cases} X(X-1)/2+Y, \\ 1\leq Y\leq X, \\ X,Y:\ \text{整係数一次式.} \end{cases} \tag{1}$$

定理 3.1 は，二次不定方程式が三角行列を一次元化

☆ ここでは，$x$，$y$ 方向に一次元化する場合についてのみ述べているが，斜め方向に要素を格納する場合も考えられる．しかしそのような二次式においても，式 (1) の形にできる一次変換が存在するならば，本論文で提案する分解手法を適用できる．たとえば，

4
2 5
1 3 6

のように要素を格納した場合，$n-x+y\rightarrow X$，$y\rightarrow Y$ の変換により，式 (1) の形になる．

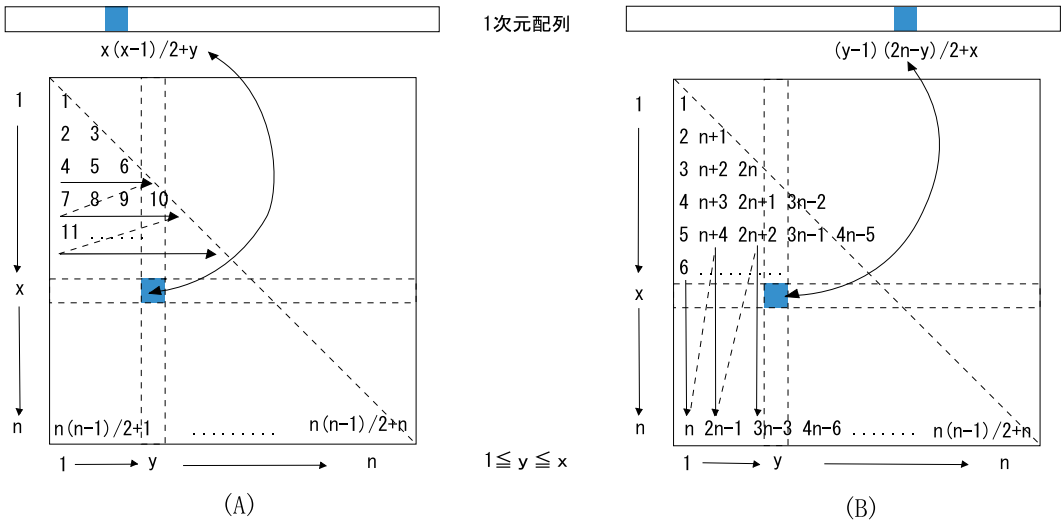


**図 1** 三角行列の一次元化
Fig. 1 Linearizing triangular matrix.

した配列参照の添え字式の場合は，一意に一次不定方程式に分解できることを保証し，定理 3.2 で，より一般的な二次不定方程式が，一次不定方程式に分解できる条件を与えている．

**定理 3.1** 以下の依存方程式とループのインデックス変数に関する制約条件が与えられたとき，

$$\begin{cases} x(x-1)/2+y=\hat{x}(\hat{x}-1)/2+\hat{y}, \\ 1\le y\le x,\ 1\le \hat{y}\le \hat{x}, \\ x,y,\hat{x},\hat{y}:\text{整数変数}, \end{cases}$$

これを満たす解集合は

$$\begin{cases} x=\hat{x}, \quad 1\le y\le x, \\ y=\hat{y}, \quad 1\le \hat{y}\le \hat{x}. \end{cases}$$

を満たす解集合と一致する．

（証明）$1\le y\le x$ の任意の $x$ と $y$ に対して，$z=x(x-1)/2+y$ とする．$\hat{x}(\hat{x}-1)/2+\hat{y}=z$ かつ $1\le\hat{y}\le\hat{x}$ であるような整数 $\hat{x}$ を選ぶ．

$x<\hat{x}$ とすると，$x=\hat{x}-\alpha$．ただし，$\alpha\ge 1$．$x(x-1)/2+y=\hat{x}(\hat{x}-1)/2+\hat{y}$ から，$\hat{x}$ を $x+\alpha$ で置き換える．$y=\alpha x+\alpha(\alpha-1)/2+\hat{y}$ となり，$y>x$ が得られる．これは $y$ のとりかたに矛盾する．$x>\hat{x}$ の場合も同様にして矛盾に至る．したがって，$x=\hat{x}$ である． □

さらに一般的な二次不定方程式について考える．

**定理 3.2** 以下のような二次形式の依存方程式と制約条件が与えられたとき，

$$\begin{cases} \sum_{j=1}^{n}\sum_{i=1}^{n}A_{ij}x_ix_j+\sum_{i=1}^{n}B_ix_i+C \\ =\sum_{j=1}^{n}\sum_{i=1}^{n}\hat{A}_{ij}\hat{x}_i\hat{x}_j+\sum_{i=1}^{n}\hat{B}_i\hat{x}_i+\hat{C}, \\ L_i\le x_i\le U_i,\ L_i\le\hat{x}_i\le U_i, \\ A_{ij},B_i,C,\hat{A}_{ij},\hat{B}_i,\hat{C}\ :\ \text{実数定数}, \\ L_i:\ \text{ループインデックス変数の下限}, \\ U_i:\ \text{ループインデックス変数の上限}, \\ (1\le i\le n,\ 1\le j\le n). \end{cases} \tag{2}$$

もし，次の条件

$$\begin{cases} A_{ii}=a_i^2/2, \\ A_{ij}+A_{ji}=a_ia_j, \\ B_i=ca_i+b_i-a_i/2, \\ C=d+c(c-1)/2, \\ L_i\le x_i\le U_i\ \text{のもとで} \\ 1\le\sum_{i=1}^{n}b_ix_i+d\le\sum_{i=1}^{n}a_ix_i+c, \\ (1\le i\le n,\ 1\le j\le n,\ i\ne j) \end{cases} \tag{3}$$

かつ

$$\begin{cases} \hat{A}_{ii}=\hat{a}_i^2/2, \\ \hat{A}_{ij}+\hat{A}_{ji}=\hat{a}_i\hat{a}_j, \\ \hat{B}_i=\hat{c}\hat{a}_i+\hat{b}_i-\hat{a}_i/2, \\ \hat{C}=\hat{d}+\hat{c}(\hat{c}-1)/2, \\ L_i\le\hat{x}_i\le U_i\ \text{のもとで} \\ 1\le\sum_{i=1}^{n}\hat{b}_i\hat{x}_i+\hat{d}\le\sum_{i=1}^{n}\hat{a}_i\hat{x}_i+\hat{c}, \\ (1\le i\le n,\ 1\le j\le n,\ i\ne j) \end{cases} \tag{4}$$

を満たす整数定数 $a_i$，$b_i$，$c$，$d$，$\hat{a}_i$，$\hat{b}_i$，$\hat{c}$，$\hat{d}$ $(1\le i\le n)$ が存在するならば，式 (2) の二次不定方程式は

入力：2 をかけて分母をはらった $n$ 変数の二次式：$\Sigma_{j=1}^{n}\Sigma_{i=1}^{n}A_{ij}x_{ij}+\Sigma_{i=1}^{n}B_ix_i+C$，ただし，$x_1$ は $A_{11}>0$ の最外側のループインデックス変数；インデックス変数とその上下限：$L_i\le x_i\le U_i$，$1\le i\le n$.
出力：一次式：$\Sigma_{i=1}^{n}a_ix_i+c$ と $\Sigma_{i=1}^{n}b_ix_i+d$．もし分解ができない場合は，false を返す.
**if** $A_{ii}=a_i^2$，$A_{ij}+A_{ji}=2a_ia_j$ を満たす $a_i$ $(a_1>0)$ が存在する $(1\le i\le n, 1\le j\le n, i\ne j)$ **then**
  $\min(\Sigma_{i=1}^{n}a_ix_i)+c_{\rm ini}=1$ となる $c_{\rm ini}$ を求める
  $b_1\leftarrow(B_1-2a_1c_{\rm ini}+a_1)/2$; $H\leftarrow\lfloor b_1/a_1-1\rfloor$
  **if** $H<0$ **then**
    $H\leftarrow 0$
  **endif**
  $c\leftarrow c_{\rm ini}+H$; $b_i\leftarrow(B_i+a_i)/2-a_i(c_{\rm ini}+H)$; $d\leftarrow C/2-(c_{\rm ini}+H)(c_{\rm ini}+H-1)/2$
  **if** $1\le\Sigma_{i=1}^{n}b_ix_i+d\le\Sigma_{i=1}^{n}a_ix_i+c$ **then**
    return $\Sigma_{i=1}^{n}a_ix_i+c$ と $\Sigma_{i=1}^{n}b_ix_i+d$
  **else**
    $c'\leftarrow c+1$; $b'_i\leftarrow b_i-a_i$; $d'\leftarrow d-c$
    **if** $1\le\Sigma_{i=1}^{n}b'_ix_i+d'\le\Sigma_{i=1}^{n}a_ix_i+c'$ **then**
      return $\Sigma_{i=1}^{n}a_ix_i+c'$ と $\Sigma_{i=1}^{n}b'_ix_i+d'$
    **endif**
  **endif**
**endif**
return false

**図 2** 分解アルゴリズム
Fig. 2 Reduction algorithm.

次の一次不定方程式に分解できる．

$$\begin{cases}\sum_{i=1}^{n}a_ix_i+c=\sum_{i=1}^{n}\hat{a}_i\hat{x}_i+\hat{c},\\ \sum_{i=1}^{n}b_ix_i+d=\sum_{i=1}^{n}\hat{b}_i\hat{x}_i+\hat{d},\\ L_i\le x_i\le U_i,\ L_i\le\hat{x}_i\le U_i,\\ (1\le i\le n)\end{cases}\tag{5}$$

（証明）式 (2) が条件 (3) と (4) を満たすときに，一次不定方程式 (5) が導かれることを証明すれば十分である．式 (2) の $A_{ij}$，$B_i$，$C$，$\hat{A}_{ij}$，$\hat{B}_i$，$\hat{C}$ を $a_i$，$b_i$，$c$，$d$，$\hat{a}_i$，$\hat{b}_i$，$\hat{c}$，$\hat{d}$ で置き換えて整理すると，式 (2) の左辺は

$$\frac{1}{2}\left(\sum_{i=1}^{n}a_ix_i+c\right)\left(\sum_{i=1}^{n}a_ix_i+c-1\right)+\sum_{i=1}^{n}b_ix_i+d,$$

右辺は

$$\frac{1}{2}\left(\sum_{i=1}^{n}\hat{a}_i\hat{x}_i+\hat{c}\right)\left(\sum_{i=1}^{n}\hat{a}_i\hat{x}_i+\hat{c}-1\right)+\sum_{i=1}^{n}\hat{b}_i\hat{x}_i+\hat{d}$$

となる．ここで，$\sum_{i=1}^{n}a_ix_i+c\to X$，$\sum_{i=1}^{n}b_ix_i+d\to Y$，$\sum_{i=1}^{n}\hat{a}_i\hat{x}_i+\hat{c}\to\hat{X}$，$\sum_{i=1}^{n}\hat{b}_i\hat{x}_i+\hat{d}\to\hat{Y}$ とおくと，$X(X-1)/2+Y=\hat{X}(\hat{X}-1)/2+\hat{Y}$ となる．また式 (3) と (4) の制約条件は，$1\le Y\le X$，$1\le\hat{Y}\le\hat{X}$ と表される．したがって定理 3.1 より，二次不定方程式 (2) は一次不定方程式 (5) に分解される．□

### 3.2 アルゴリズム

**図 2** に $n$ 変数二次式を 2 つの一次式に分解するアルゴリズムを示す．二次不定方程式の両辺に対してアルゴリズムを適用し，ともに分解可能ならば，式 (5) のように簡単化された依存問題が得られる．

アルゴリズム適用前に，係数を整数として扱うために方程式の両辺に 2 をかけて，分母を消去しておく．アルゴリズムは右辺と左辺それぞれに適用される．入力は，各係数が整数で，$x_1$ が $A_{11}>0$ の最外側のループインデックス変数の二次式，出力は 2 つの一次式である．

入力が二次式 $\sum_{j=1}^{n}\sum_{i=1}^{n}A_{ij}x_ix_j+\sum_{i=1}^{n}B_ix_i+C$ のとき，定理 3.2 の条件を満たす $a_i$，$c$，$b_i$，$d$ を決定することが，アルゴリズムの目的である．二次の項の係数から整数 $a_i$ は求まり，$c$ が決まれば $b_i$ と $d$ は二次式の剰余として求まる．不等式 $1\le\sum_{i=1}^{n}b_ix_i+d\le$

$$
\begin{array}{ll}
(i^2+3i+2)/2=(\hat{i}^2+\hat{i}-2)/2 & \Longrightarrow \quad i+1=\hat{i} \\
2\le i\le n,\ 2\le \hat{i}\le n & \qquad\quad i+1=\hat{i}-1 \\
 & \qquad\quad 2\le i\le n,\ 2\le \hat{i}\le n
\end{array}
$$

**図 3** 例（1）
Fig. 3 Example (1).

$$
\begin{array}{ll}
(k^2-2kj+j^2+k-j+2)/2 & \Longrightarrow \quad k-j+1=\hat{k} \\
=(\hat{k}^2+\hat{k}-2\hat{j}+2)/2 & \qquad\quad 1=\hat{k}-\hat{j}+1 \\
1\le k\le n,\ 1\le \hat{k}\le n & \qquad\quad 1\le k\le n,\ 1\le \hat{k}\le n \\
1\le j\le k,\ 1\le \hat{j}\le \hat{k} & \qquad\quad 1\le j\le k,\ 1\le \hat{j}\le \hat{k}
\end{array}
$$

**図 4** 例（2）
Fig. 4 Example (2).

$$
\begin{array}{ll}
(-j^2+2nj-2n+j+2i)/2 & \Longrightarrow \quad i=\hat{i} \\
=(-\hat{j}^2+2n\hat{j}-\hat{j}+2\hat{i})/2 & \qquad\quad j=\hat{j}+1 \\
3\le i\le n,\ 3\le \hat{i}\le n & \qquad\quad 3\le i\le n,\ 3\le \hat{i}\le n \\
1\le j\le i-2,\ 1\le \hat{j}\le \hat{i}-2 & \qquad\quad 1\le j\le i-2,\ 1\le \hat{j}\le \hat{i}-2
\end{array}
$$

**図 5** 例（3）
Fig. 5 Example (3).

$\sum_{i=1}^{n} a_i x_i + c$ を満たすように $c$ を決定しなければならない．もしそのような $c$ が存在しなければ一次式への分解はできない．

**ステップ 1**

まず，$A_{ij}$ から整数 $a_i$ を求める．$a_1 > 0$ とすれば $a_i$ は一意に決定できる．このとき矛盾を起こすならば分解できない．

**ステップ 2**

$L_i \le x_i \le U_i$ のもとで $\min(\sum_{i=1}^{n} a_i x_i) + c_{\mathrm{ini}} = 1$ を満たす $c_{\mathrm{ini}}$ と，$c_{\mathrm{ini}}$ に対する $b_1$ を計算する．$1 \le \sum_{i=1}^{n} a_i x_i + c$ なので少なくとも $c_{\mathrm{ini}} \le c$，かつ $\sum_{i=1}^{n} b_i x_i + d \le \sum_{i=1}^{n} a_i x_i + c$ を満たす $c$ を求める．$c = c_{\mathrm{ini}} + H$ として，不等式 $1 \le \sum_{i=1}^{n} b_i x_i + d \le \sum_{i=1}^{n} a_i x_i + c$ を満たしているか調べる．満足していれば $\sum_{i=1}^{n} a_i x_i + c$ と $\sum_{i=1}^{n} b_i x_i + d$ を返して終了する．

**ステップ 3**

ステップ 2 で求めた $c$ が不等式を満たしていない場合は，$c' = c+1$ として，$\sum_{i=1}^{n} b'_i x_i + d'$ を計算し直す．この場合も不等式を満足しなければ，当該二次式は分解不可能であり，アルゴリズムは false を返す．満足していれば $\sum_{i=1}^{n} a_i x_i + c'$ と $\sum_{i=1}^{n} b'_i x_i + d'$ を返して終了する．

### 3.3 不等式の判定

与えられた $c$ が不等式 $1 \le \sum_{i=1}^{n} b_i x_i + d \le \sum_{i=1}^{n} a_i x_i + c$ を満たしているかどうかの判定が問題となる．定理 3.1 において，もし $x$ と $y$ が $1 \le y \le x$ を満足しなければ，二次不定方程式を一意に一次方程式に分解することはできない．したがってこの条件は重要である．不等式は $1 - d \le \min(\sum_{i=1}^{n} b_i x_i)$ かつ $d - c \le \min(\sum_{i=1}^{n} (a_i - b_i) x_i)$ と考えることができるので，$\min(\sum_{i=1}^{n} b_i x_i)$ と $\min(\sum_{i=1}^{n} (a_i - b_i) x_i)$ を計算すればよい．

$\min(\sum_{i=1}^{n} k_i x_i)$（$k_i$：整係数）について，最内側ループのインデックス変数 $x_n$ から最外側ループの $x_1$ まで以下の条件で置き換えを行い，順次 $x_i$ を消去していく．

$$
x_i \to \begin{cases} L_i & (k_i \ge 0), \\ U_i & (k_i < 0). \end{cases}
$$

$x_1$ の上下限は定数なので，$x_1$ の置き換え終了後は $\min(\sum_{i=1}^{n} k_i x_i)$ の値は定数となり，ゆえに不等式の判定が可能である．

ループの制約条件のもとで，配列参照の一次添え字式がとりうる最小値および最大値を求める方法についてはすでに Range テスト[7] で述べられているので，ここでは簡単な説明にとどめる．

### 3.4 適 用 例

本節では，アルゴリズムの適用例をいくつか示す．

**図 3** では，左側の二次式不定方程式は，

$$(i+1)i/2 + i + 1 = \hat{i}(\hat{i}-1)2 + \hat{i} - 1.$$

と表され，右側の一次式が得られる．ループ繰り越し依存が存在しないことが容易に分かる．

**図 4** の二次不定方程式は，以下のように表せる．

$$(k-j+1)(k-j)/2+1=\hat{k}(\hat{k}-1)/2+\hat{k}-\hat{j}+1.$$

したがって依存問題は右側のように簡単化される．

**図 5** の二次式は図 1 の (B) の例である．アルゴリズム適用前に両辺を変換しておく．左辺は

$$\begin{aligned} &n(n+1)/2+1-(-j^2+2nj-2n+j+2i)/2\\ &\quad=(n-j+1)(n-j)/2+n-i+1, \end{aligned}$$

右辺は

$$\begin{aligned} &n(n+1)/2+1-(-\hat{j}^2+2n\hat{j}-\hat{j}+2\hat{i})/2\\ &\quad=(n-\hat{j})(n-\hat{j}-1)/2+n-\hat{i}+1 \end{aligned}$$

となる．結果は図の右側の連立一次式である．この二重ループは $i$ と $\hat{i}$ が外側ループのインデックス変数なので，外側ループが並列実行できることが分かる．

図 4 の簡単化された連立一次不定方程式においては，変数 $\hat{k}$ が，両方の一次方程式に現れているので，分離して依存テストを行えない．このような場合は，多次元配列の添え字式を同時に解析できる手法，たとえばスタンフォード大学による SUIF コンパイラ[8]，Power テスト[9] または Omega テスト[10] などを利用するほうが解析精度が良いことに注意すべきである．

## 4. 関 連 研 究

一般的によく利用されるデータ依存テストは，GCD テストと Banerjee テスト[2] の組合せである．しかし，これらは二次式を解析することはできない．

一次式であれ二次式であれ，データ依存テストにとって処理の重い複雑な方程式を扱う場合に 2 通りのアプローチが考えられる．1 つは多変数多項式に対応したデータ依存テスト手法を考案すること，もう 1 つはプログラムコードの特性を利用したヒューリスティックな方法で依存解析問題をできるだけ簡単にして，依存テストの負荷を減らすことである．

前者の試みとして，イリノイ大学の Polaris[11] に採用されている Range テストは二次の添え字式を解析できると報告されている．この依存テストは，多重ループにおけるあるイタレーションでアクセスされる配列要素の範囲が，他のイタレーションでアクセスされる要素の範囲と，オーバラップしなければ依存がないという考えを基本としている．しかし，依存がないことを証明することが目的で，もし解が存在しても解集合を求めることはできない．したがって二元一次不定方程式のような簡単に解が求まる問題でも方向ベクトルしか得られない．

後者のアプローチとしては，Maslov が rectanglur delinearization[12] と呼ばれる多変数一次式を単純な一次式に簡単化する手法を，Omega テストに実装している．多次元の長方形配列を一次元化した配列を参照するため，多変数の一次式になっている添え字式を，次元ごとに分解して，1 つの変数のみを含む一次式に変換する．手法適用後の配列は，ループの多重度と同じ次元数を持つ．しかし，この方法は二次式に対応したものではない．そこでさらに多項式を一次式に簡単化する方法[13] も提案している．これは代数学的な発想によるもので，因子化と線形化という操作で減次を繰り返し，一次方程式と不等式の集合に変換する．しかし，操作を繰り返すたびに方程式と不等式が増えてしまい，操作終了後にこれらをふたたび統合して簡単化する必要がある．計算負荷も大きく，実装面からもおよそ現実的とはいえない．

我々の手法は後者に属し，依存解析に先だって実行すれば，二次の添え字式も従来のデータ依存テスト手法で正しく解析できるというメリットがある．また非常にシンプルなアルゴリズムになっている．

## 5. ま と め

本論文では，三角行列を一次元化した配列の参照に現れる二次式が，$X(X-1)/2+Y$，$1\le Y\le X$（ただし $X$ と $Y$ は一次式）の形で表現できることを示し，これをもとに二次不定方程式を一次不定方程式に分解する手法を提案した．またいくつかの適用例をあげ，依存解析問題がかなり簡単化されることを示した．我々の手法はシンプルなアルゴリズムで処理時間も短く，実装も容易である．

本手法を用いることで，二次不定方程式を直接データ依存解析するよりもはるかに計算負荷が少なく，より正確な解析結果を得ることができる．また，GCD テストと Banerjee テストの組合せといった簡便で近似的なデータ依存解析手法に対しても，精度良く解析できる機会を与える．

多次元データ依存解析テストが必要な場合も生じるが，依存解析問題はたかだか二連立不定方程式を扱うものであり，大きな計算負荷とはならない．

## 参 考 文 献

1) Polychronopoulos, C.D., Girkar, M.B., Haghighat, M.R., Lee, C.L., Leung, B.P. and Schouten, D.A.: The Structure of Parafrase-2: an Advanced Parallelizing Compiler for C and Fortran, *Proc. Workshop on Languages and Compilers for Parallel Computing*, pp.423–453 (1990).

2) Banerjee, U.: *Dependence Analysis for Supercomputing*, Kluwer Academic Publishers (1988).
3) 中田育夫：コンパイラの構成と最適化，朝倉書店 (1999).
4) 高木貞治：初等整数論講義，共立出版 (1971).
5) 河田敬義：数論 I，岩波書店 (1978).
6) Alpern, D.A.: *Methods to solve* $Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$. http://members.tripod.com/˜alpertron/METHODS.HTM.
7) Blume, W. and Eigenmann, R.: The Range Test: A Dependence Test for Symbolic, Nonlinear Expressions, *Proc. Conference on Supercomputing*, pp.528–537 (1994).
8) Maydan, D., Hennesy, J. and Lam, M.: Efficient and Exact Data Dependence Analysis for Parallelizing Compilers, *Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp.1–14 (1991).
9) Wolfe, M. and Tseng, C.-W.: The power test for data dependence, *IEEE Trans. Parallel and Distributed Systems*, Vol.3, No.5, pp.591–601 (1992).
10) Pugh, W.: The Omega test: A fast and practical integer programming algorithm for dependence analysis, *Proc. Supercomputing'91* (1991).
11) Blume, W., Eigenmann, R., Faigin, K., Grout, J., Hoeflinger, J., Padua, D., Petersen, P., Pottenger, W., Rauchwerger, L., Tu, P. and Weatherford, S.: Polaris: The Next Generation in Parallelizing Compilers, Technical Report, No.1375, Center for Supercomputing Research and Development (CSRD), University of Illinois Urbana-Champaign.
12) Maslov, V.: Delinearization: An Efficient Way to Break Multiloop Dependence Equations, *Proc. SIGPLAN'92 Conference on Programming Language Design and Implementation*, Vol.27, No.7, pp.152–161 (1992).
13) Maslov, V. and Pugh, W.: Simplifying Polynomial Constraints Over Integers to Make Dependence Analysis More Precise, Technical Report, CS-TR-3109, Dept. of Computer Science, University of Maryland, College Park (1993).

(平成 12 年 8 月 30 日受付)
(平成 12 年 12 月 1 日採録)

**神戸 和子**（学生会員）

昭和 55 年広島大学総合科学部総合科学科卒業．同年シャープ（株）入社．CAD システムの研究開発に従事．昭和 58 年退職．平成 11 年奈良女子大学理学研究科情報科学専攻修士課程修了．理学修士．現在，奈良女子大学大学院人間文化研究科複合領域専攻博士後期課程在籍．自動並列化コンパイラの依存解析と最適化について研究中．

**加古富志雄**

昭和 52 年神戸大学工学部電気工学科卒業．昭和 54 年同大学院修士課程修了．昭和 57 年九州大学大学院工学研究科応用原子核工学専攻博士後期課程修了．工学博士．昭和 58 年より広島大学助手，助教授を経て，現在奈良女子大学理学部教授．非線形発展方程式とコンピュータ代数アルゴリズムに関する研究に従事．日本応用数理学会，日本数式処理学会，SIGSAM/ACM 各会員．