三重対角線形方程式の分割並列消去法について†

陳 莘萌†† 井上知子††† 萩原 宏†††

並列処理計算機の出現以来，数値計算の分野に対しては並列処理に適したアルゴリズムの研究・開発が求められてきた．本論文では，三重対角線形方程式の解法に対する並列処理の新しいアルゴリズムとして，Wangの分割法に基づく分割並列消去アルゴリズム(I)，および(II)を提案する．アルゴリズム(I)は計算機の並列処理個数が十分大きい場合，アルゴリズム(II)は並列処理個数が7以下の場合に適していると考えられる．三重対角線形方程式を解くための並列処理のアルゴリズムとして既に Stone の recursive doubling 法や Hockney の cyclic reduction 法，Wang の分割法などがあるが，前者2つは特定のアーキテクチャの計算機向きであるとか，行列の次元数が2のべき乗であることが望ましいなどの制約がある．一方 Wang の分割法も本論文で述べるアルゴリズムも，SIMD 方式の計算機であれば，特に制約のない似た方法である．しかし，ベクトル演算総数においてアルゴリズム(I)は Wang の分割法より著しく少なく，その約 40% で済む．

1. まえがき

本論文では三重対角線形方程式を並列に解くために，本質的には Gauss の消去法であるが，それに並列処理を加味した分割並列消去アルゴリズム(I)，および分割並列消去アルゴリズム(II)と名付ける新しい方法を提案する．三重対角線形方程式を並列に解くためのアルゴリズムとしては，既に Stone が recursive doubling 法[1]とその改良法[2),3)]を提案しているが，その方法は Illiac IV のようなプロセッサアレイ計算機に適したものである．Hockney が提案し[4]，Lambiotte と Voigt が調べた cyclic reduction 法[5]はプロセッサアレイ計算機にも利用できるが，STAR-100 とか CRAY-1 のようなパイプライン処理計算機の方に適したアルゴリズムである．1981年 IBM の H. H. Wang が提案した方法[6]は，プロセッサアレイ方式にもパイプライン方式にも適用できるものである．本論文では Wang の提案した方法を "Wang の分割法" とよぶ．本論文で述べる分割並列消去アルゴリズム(I)は，この Wang の分割法に基づくものである．

Wang の分割法におけるベクトル演算総数は，recursive doubling 法や cyclic reduction 法より多くなり，スカラ演算総数も cyclic reduction 法より約5% 多くなるが recursive doubling 法より少なくなる．しかし，Wang の分割法の特徴は他の2方法と違って，特定のアーキテクチャの計算機にしか向かないということがなく，cyclic reduction 法のように行列の次元数が2のべき乗でなければ極端に効率が低下するということもないことである．これらの特徴は，本論文で提案する分割並列消去法についても同様である．

性質のよく似た Wang の分割法と比べると，分割並列消去アルゴリズム(I)のベクトル演算総数は約60% 減るうえ，スカラ演算総数も減少する．しかし，次元数 n が大きくなると，ベクトル演算総数は recursive doubling 法や cyclic reduction 法より多い．分割並列消去アルゴリズム (II) のスカラ演算総数は，直接法であるガウス消去法で必要とされる演算数とほぼ同じであり，他のいずれの方法よりも少ない．これは，並列プロセッサの個数が少ないときに適する方法であることを示唆している．

なお本論文では，対象とする並列処理計算機は SIMD 方式のものであり，複数のデータ領域に対して同種の演算を施す場合に，ベクトル演算1回という数え方をしている．また，本論文で対象とする三重対角線形方程式は必ず解が一つだけ存在し，かつ係数行列は対角優位であるか，あるいはそうでない場合も Pivoting を必要としない，などの条件の良い場合のものである．

2. 一般的な分割並列消去アルゴリズム (I)

$$Ax=r \qquad (1)$$

ただし次元数 n，ベクトル x, r および係数行列 A は

† Partitioned Parallel Elimination Method for Tridiagonal Linear Equations by CHEN XINMENG (Computer Science Faculty, Wuhan University), TOMOKO INOUE and HIROSHI HAGIWARA (Department of Information Science, Faculty of Engineering, Kyoto University).

†† 中国 武漢大学計算機科学部
††† 京都大学工学部情報工学科

$$A=\begin{vmatrix} a_1 & b_1 & & & & \\ c_2 & a_2 & b_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & c_{n-1} & a_{n-1} & b_{n-1} \\ & & & & c_n & a_n \end{vmatrix}$$

$$x=\begin{vmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{vmatrix},\quad r=\begin{vmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{vmatrix}$$

n は6以上の，二項に因数分解可能な数とする．

方程式(1)を解く分割並列消去アルゴリズム(I)の具体的なステップは次のとおりである．初等的な行の代数演算がその基本となっている．なお，消去の過程で同じステップ内で値が変更されたものは，ダッシュや上付き添字をつけて示している．さらに，新しいステップに移行した段階では $a_j=a_j^{(j)}$, $r_m=r_m''$ のように，ダッシュや上付き添字を取り払うように定義しなおされたものとして記述している．

ステップ(1) 小行列に分割する．

仮定により，$n=k\times p$ ($k\geqq 2$, $p\geqq 3$) とあらわせるので，A を k 次の $p\times p$ 個（ただし，$p\geqq 3$）の小行列に分割する．$n=16$ のときの様子を**図1**に示す．

ステップ(2) c_t ($t=2,3,\cdots,n$) を消去する．

$$f_{ik+1}=c_{ik+1}$$
($i=1,2,\cdots,p-1$ に対して並列) (2)

$$c_{ik+j}'=c_{ik+j}/a_{ik+j-1}^{(j-1)}$$
($i=0,1,\cdots,p-1$ に対して並列) (3)

$$\begin{cases} a_{ik+j}^{(j)}=a_{ik+j}-c_{ik+j}'\times b_{ik-j-1} \\ \quad (i=0,1,\cdots,p-1 \text{ に対して並列}) \quad (4) \\ f_{ik+j}=-c_{ik+j}'\times f_{ik+j-1} \\ \quad (i=1,2,\cdots,p-1 \text{ に対して並列}) \quad (5) \\ r_{ik+j}^{(j)}=r_{ik+j}-c_{ik+j}'\times r_{ik+j-1}^{(j-1)} \\ \quad (i=0,1,\cdots,p-1 \text{ に対して並列}) \quad (6) \end{cases}$$

まず(2)の操作を行う．次に

$$a_{ik+1}^{(1)}=a_{ik+1},\ r_{ik+1}^{(1)}=r_{ik+1}$$
($i=0,1,\cdots,p-1$)

とおく．そのうえで $j=2,3,\cdots,k$ のそれぞれについて，(3)を計算した後(4)～(6)を並列に計算することを，$(k-1)$ 回繰り返す*．このとき，新しい要素 f_{ik+j} ($i=1,2,\cdots,p-1$, $j=1,2,\cdots,k$) が付加される．

ステップ(3) b_t ($t=1,2,\cdots,n-1$) を消去する．

$$g_{ik-1}=b_{ik-1}$$
($i=1,2,\cdots,p-1$ に対して並列) (7)

$$b_{j-1}'=b_{j-1}/a_j$$
($j=2,3,\cdots,n$. ただし，$j=ik$ ($i=1,2,\cdots,p-1$) は除く．) (8)

(7)，(8)それぞれを並列に1回計算した後，右下隅のブロックに g_j を付加しないための措置を，次の(9)，(10)式で講じておく**．

$$\begin{cases} f_{n-1}'=f_{n-1}-b_{n-1}'\times f_n & (9) \\ r_{n-1}'=r_{n-1}-b_{n-1}'\times r_n & (10) \end{cases}$$

この後，$f_{n-1}=f_{n-1}'$, $r_{n-1}=r_{n-1}'$ とおく．各対角小行列の下から2番目の行を最初の軸行として1つ上の行の b_j を消去し，g_j を付加する．計算の済んだばかりの行を次の軸行として b_{j-1} を消去し，g_{j-1} を付加することを繰り返す．この計算のために，まず

A =

a1	b1														
c2	a2	b2													
	c3	a3	b3												
		c4	a4	b4											
			c5	a5	b5										
				c6	a6	b6									
					c7	a7	b7								
						c8	a8	b8							
							c9	a9	b9						
								c10	a10	b10					
									c11	a11	b11				
										c12	a12	b12			
											c13	a13	b13		
												c14	a14	b14	
													c15	a15	b15
														c16	a16

図1 $n=16$, $p=4$, $k=4$ のときの小行列への分割
Fig. 1 The partitioned form in the Partitioned Parallel Elimination Method (I), where $n=16, p=4, k=4$.

* Wang の分割法では(3)，(4)，(5)，(6)をこの順序で $j=2,\cdots,k$ について計算しているが，(4)，(5)，(6)は互いに独立なので，(3)を別に求めておいて(4)～(6)を並列に計算することができる．

** Wangの分割法では右下隅のブロックに g_j を付加する．本アルゴリズム(I)で g_j を付加しないことによるメリットは次の2つである．
1. スカラ演算数を減らせる．
2. 演算ベクトル長を，(11)式およびステップ(4)のはじめの部分で短くできる．

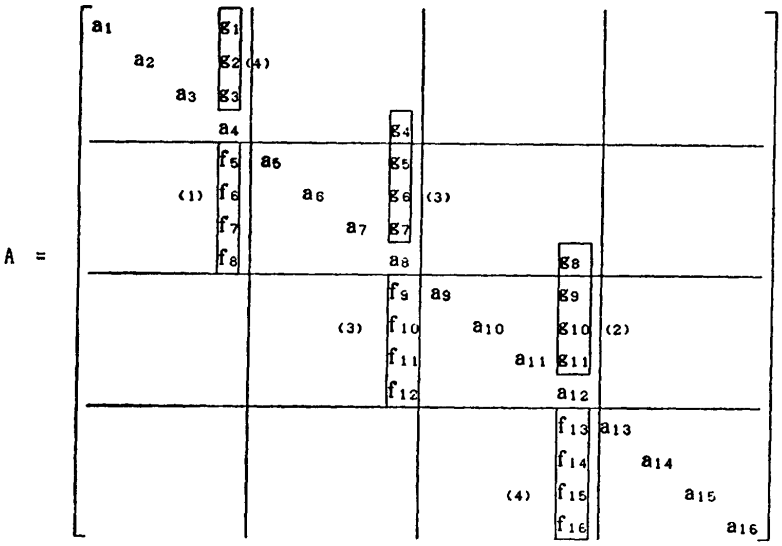

図 2 すべての c_i, b_i を消去したあとの行列
(1), (2), … などの番号は，ステップ(4)での消去の順番を示す．

Fig. 2 The block tridiagonal form when all c_i, b_i have been vanished. The numbers (1), (2), … show the order of the elimination in the Step (4) of the Partitioned Parallel Elimination Method (I).

(a) p が偶数である場合

次に記述する式について，イ～ニの手順で計算する．

$$f_{ik+j}'=f_{ik+j}/a_{ik} \quad (j=1,2,\cdots,k \text{ に対して並列}) \tag{17}$$

$$\begin{cases} a_{(i+1)k}'=a_{(i+1)k}-f_{(i+1)k}'\times g_{ik} & (18)\\ g_{ik+j}'=g_{ik+j}-f_{ik+j}'\times g_{ik} \quad (j=1,2,\cdots,k-1 \text{ に対して並列}) & (19)\\ r_{ik+j}'=r_{ik+j}-f_{ik+j}'\times r_{ik} \quad (j=1,2,\cdots,k \text{ に対して並列}) & (20) \end{cases}$$

$$g_{ik+j-1}'=g_{ik+j-1}/a_{(i+1)k} \quad (j=1,2,\cdots,k \text{ に対して並列}) \tag{21}$$

$$a_{ik}'=a_{ik}-g_{ik}'\times f_{(i+1)k} \tag{22}$$

$$f_{ik+j-1}'=f_{ik+j-1}-g_{ik+j-1}'\times f_{(i+1)k} \quad (j=2,3,\cdots,k \text{ に対して並列}) \tag{23}$$

$$r_{ik+j-1}'=r_{ik+j-1}-g_{ik+j-1}'\times r_{(i+1)k} \quad (j=1,2,\cdots,k \text{ に対して並列}) \tag{24}$$

$$f_{ik+j}''=f_{ik+j}'/a_{ik}' \quad (j=1,2,\cdots,k \text{ に対して並列}) \tag{25}$$

$$r_{ik+j}''=r_{ik+j}'-f_{ik+j}''\times r_{ik}' \quad (j=1,2,\cdots,k \text{ に対して並列}) \tag{26}$$

$$g_{ik+j-1}''=g_{ik+j-1}'/a_{(i+1)k}' \quad (j=1,2,\cdots,k \text{ に対して並列}) \tag{27}$$

$$r_{ik+j-1}''=r_{ik+j-1}'-g_{ik+j-1}''\times r_{(i+1)k}' \quad (j=1,2,\cdots,k \text{ に対して並列}) \tag{28}$$

イ．$i=1$ に対して，(17)の計算の後，(18)～(20)を並列に計算する．さらに，$g_k'=g_k$, $r_k'=r_k$, $a_{2k}=a_{2k}'$, $r_{2k}=r_{2k}'$, とする．

ロ．$\begin{bmatrix} i=2 \\ l=p-2 \end{bmatrix}$, $\begin{bmatrix} i=3 \\ l=p-3 \end{bmatrix}$, …, $\begin{bmatrix} i=p/2-1 \\ l=p/2+1 \end{bmatrix}$

のそれぞれについて，(17)，(21)を並列に計算した後，(18)～(20)，(22)～(24)を並列に計算する．ただし i,l の各値の計算の後，次のように置く．

$a_{(i+1)k}=a_{(i+1)k}'$, $r_{ik}'=r_{ik}$,

$r_{(i+1)k}=r_{(i+1)k}'$, $g_{ik}'=g_{ik}$,

$f_{ik+k-1}^{(k-1)}=f_{ik+k-1}$ $(i=1,2,\cdots,p-1)$

$r_{ik+k-1}^{(k-1)}=r_{ik+k-1}$ $(i=0,1,\cdots,p-1)$

と定義し，$j=k-1,k-2,\cdots,2$ のそれぞれに対して次の(11)～(13)式を並列に計算することを $(k-2)$ 回繰り返す．

$$\begin{cases} g_{ik+j-1}=-b_{ik+j-1}'\times g_{ik+j} \quad (i=0,1,\cdots,p-2 \text{ に対して並列}) & (11)\\ f_{ik+j-1}^{(j-1)}=f_{ik+j-1}-b_{ik+j-1}'\times f_{ik+j}^{(j)} \quad (i=1,2,\cdots,p-1 \text{ に対して並列}) & (12)\\ r_{ik+j-1}^{(j-1)}=r_{ik+j-1}-b_{ik+j-1}'\times r_{ik+j}^{(j)} \quad (i=0,1,\cdots,p-1 \text{ に対して並列}) & (13) \end{cases}$$

さらに b_{ik} $(i=1,2,\cdots,p-1)$ 消去のために $g_{(p-1)k+1}=0$ として，次の(14)～(16)を $i=1,2,\cdots,p-1$ に対して並列に1回計算する．

$$\begin{cases} a_{ik}^{(0)}=a_{ik}-b_{ik}'\times f_{ik+1}^{(1)} & (14)\\ g_{ik}=-b_{ik}'\times g_{ik+1} & (15)\\ r_{ik}^{(0)}=r_{ik}-b_{ik}'\times r_{ik+1}^{(1)} & (16) \end{cases}$$

ステップ(3)までの変換を終了した $n=16$ のときの行列の形を**図2**に示す．

ステップ(4)　対角行列にする．

p が偶数であるか奇数であるかによって，f_{ik+j} と g_{ik+j} を並列に消去してゆく過程に若干の違いがある☆．

★ Wang の分割法では ik $(i=1,2,\cdots,p-1)$ カラムごとに並列に f_{ik+j} を消去した後，別のステップで ik $(i=p,\cdots,1)$ カラムごとに並列に g_{ik+j} を消去している．本アルゴリズムでは，これらの非対角要素の独立性を最大限に利用して並列処理をしており，そのために p が偶数であるか奇数であるかを区別する必要が生じた．

$a_{ik}=a_{ik}'$, $r_{ik}=r_{ik}'$,

$f_{(i+1)k}'=f_{(i+1)k}$, $r_{(i+1)k}'=r_{(i+1)k}$.

ハ. $l=p/2$ について，(21)の計算の後，(22)～(24)を並列に計算する．そして

$f_{(i+1)k}'=f_{(i+1)k}$, $r_{(i+1)k}'=r_{(i+1)k}$

とする．

ニ. $\begin{bmatrix} i=p/2 \\ l=p/2-1 \end{bmatrix}$ $\begin{bmatrix} i=p/2+1 \\ l=p/2-2 \end{bmatrix}$ $\cdots$, $\begin{bmatrix} i=p-1 \\ l=0 \end{bmatrix}$

のそれぞれについて，(25)，(27)を並列に計算の後，(26)，(28)を並列に計算する．ここに，各 i,l の計算の後 $a_{(i+1)k}'=a_{(i+1)k}$, $r_{(i+1)k}'=r_{(i+1)k}''$, $a_{ik}'=a_{ik}$, $r_{ik}'=r_{ik}''$ と置く．

ただし，一番上と一番下の小行列の計算のために次のように定義しておく．

$g_j'=g_j$, $r_j'=r_j$ $(j=0,1,\cdots,k-1)$,

$f_j'=f_j$, $r_j'=r_j$ $(j=(p-1)k+1,\cdots,pk)$,

ここに $g_0=r_0=0$ である．

（b） p が奇数である場合

p が偶数である場合と考え方は基本的に同じである．偶数の場合は f_{k+1}～f_{2k} を並列に消去することから始めたが，今度は f_{k+1}～f_{2k} と $g_{(p-2)k}$～$g_{(p-1)k-1}$ を並列に消去することから始める．アルゴリズムの記述は省くがその消去の順番を，$p=7$ について**図3**に示す．計算式は p が偶数である場合に用いたものを利用すればよい．

ステップ(5) 対角線形方程式を解く．

$x_i \leftarrow r_i/a_i$

($i=1,2,\cdots,n$ に対して並列) (29)

3. 分割並列消去アルゴリズム（II）

今まではベクトルの次元数には一切注意を払わずに，論理的に並列処理可能なものにのみ着目して議論してきたが，現実の計算機では並列演算できる個数に制限のあるのが普通である．並列演算できる個数をMとすると，Mが小さい場合には，次に述べる簡単なアルゴリズムが有効な場合がある．行列の次元数 n が偶数のときについて述べるが，奇数のときには $x_{n+1}=1$ なる方程式を追加して考えればよい．

ステップ(1) 小行列に分割する．

図4に示すように $(n/2)\times(n/2)$ の小行列に分割する．

ステップ(2) c_i $(i=2,3,\cdots,n/2)$,

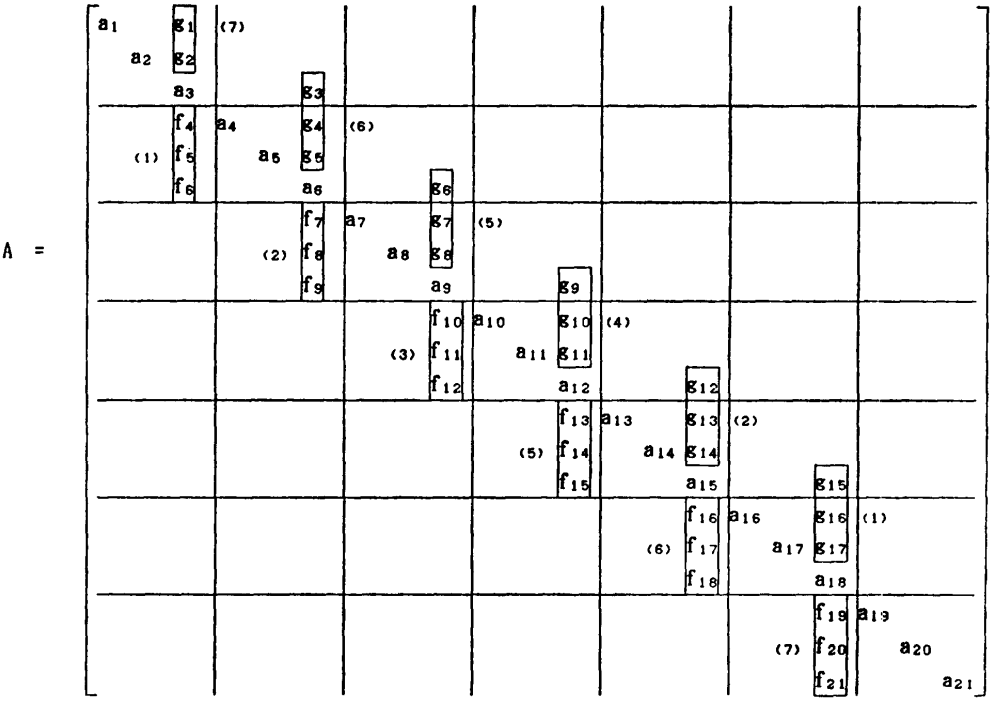

図3 分割数 p が奇数である時の消去の順序（$p=7, k=3$ のとき）
(1), (2), … などの番号は，ステップ(4)での消去の順番を示す．

Fig. 3 The order of the elimination when the partitioned number p is odd.
The numbers (1), (2), … show the eliminating order.

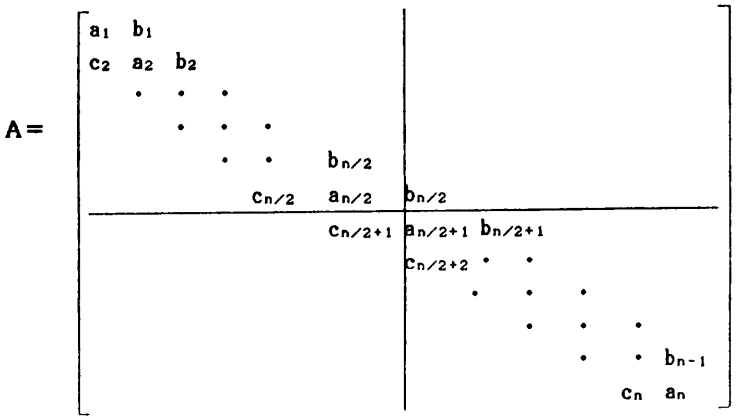

図 4 行列の分割

Fig. 4 The partitioned form in the Partitioned Parallel Elimination Method (II).

b_i $(i=n/2+1,\cdots,n-1)$ を消去する．

c_2 と b_{n-1}, c_3 と $b_{n-2},\cdots,c_{n/2}$ と $b_{n/2+1}$ を並列に消去してゆく．

$$\begin{cases} c_j'=c_j/a_{j-1}^{(j-1)} & (30)\\ b_{n-j+1}'=b_{n-j+1}/a_{n-j+2}^{(j-1)} & (31)\end{cases}$$

$$\begin{cases} a_j^{(j)}=a_j-c_j'\times b_{j-1} & (32)\\ a_{n-j+1}^{(j)}=a_{n-j+1}-b_{n-j+1}'\times c_{n-j+2} & (33)\\ r_j^{(j)}=r_j-c_j'\times r_{j-1}^{(j-1)} & (34)\\ r_{n-j+1}^{(j)}=r_{n-j+1}-b_{n-j+1}'\times r_{n-j+2}^{(j-1)} & (35)\end{cases}$$

まず，$a_1^{(1)}=a_1$, $a_n^{(1)}=a_n$, $r_1^{(1)}=r_1$, $r_n^{(1)}=r_n$ と定義する．$j=2,3,\cdots,n/2$ のそれぞれについて，(30)，(31)を並列に計算した後(32)～(35)を並列に計算することを，$(n/2-1)$ 回繰り返す．

ステップ(3)　$n/2$ 行と $(n/2)+1$ 行を対角要素だけにする．

$$\begin{cases} b_{n/2}'=b_{n/2}/a_{n/2+1} & (36)\\ c_{n/2+1}'=c_{n/2+1}/a_{n/2} & (37)\end{cases}$$

$$\begin{cases} a_{n/2}'=a_{n/2}-b_{n/2}'\times c_{n/2+1} & (38)\\ a_{n/2+1}'=a_{n/2+1}-c_{n/2+1}'\times b_{n/2} & (39)\\ r_{n/2}'=r_{n/2}-b_{n/2}'\times r_{n/2+1} & (40)\\ r_{n/2+1}'=r_{n/2+1}-c_{n/2+1}'\times r_{n/2} & (41)\end{cases}$$

(36)，(37)を並列に計算した後，(38)～(41)を並列に計算する．

実際に計算機で実行するときは，同じ下付き添字の場合，ダッシュがついていてもついていなくてもおなじ記憶域を参照することになるので，$t_1=c_{n/2+1}$, $t_2=b_{n/2}$ として(36)～(39)の式を変形すること．また，少しでも時差のおそれのあるときは $y_1=r_{n/2}$, $y_2=r_{n/2+1}$ としておいて，(40)のかわりに $r_{n/2}=y_1-b_{n/2}\times y_2$ を，(41)のかわりに $r_{n/2+1}=y_2-c_{n/2+1}\times y_1$ を用いる方が無難である．

ステップ(4)　対角行列にする．

$$\begin{cases} b_j'=b_j/a_{j+1} \\ \qquad (j=1,2,\cdots,n/2-1 \text{ に対して並列}) & (42)\\ c_{n/2+j+1}'=c_{n/2+j+1}/a_{n/2+j} \\ \qquad (j=1,2,\cdots,n/2-1 \text{ に対して並列}) & (43)\end{cases}$$

$$\begin{cases} r_{n/2-j}^{(j)}=r_{n/2-j}-b_{n/2-j}'\times r_{n/2-j+1}^{(j-1)} & (44)\\ r_{n/2+j+1}^{(j)}=r_{n/2+j+1}-c_{n/2+j+1}'\times r_{n/2+j}^{(j-1)} & (45)\end{cases}$$

(42)，(43)を並列に計算した後，$r_{n/2}^{(0)}=r_{n/2}$, $r_{n/2+1}^{(0)}=r_{n/2+1}$ として，$j=1,2,\cdots,n/2-1$ のそれぞれについて(44)，(45)を並列に計算する．

ステップ(5)　対角線形方程式を解く．

$$x_i=r_i/a_i \quad (i=1,2,\cdots,n \text{ に対して並列}) \quad (46)$$

4. 各アルゴリズムの演算総数の比較

表1に分割並列消去アルゴリズム(I)のベクトル演算総数を**表2**にスカラ演算総数を列記する．表中「三重対角線形方程式を解く場合」というのは，右辺ベクトルが1個のとき，解を求めるのに必要となる演算回数である．「右辺の計算」というのは，同じ係数行列で右辺ベクトルだけ異なっている時に，1個の右辺につき増加する回数である．**表3**に分割並列消去アルゴリズム (II) の場合のベクトル演算総数を示す．

(1)　ベクトル演算総数の比較

表4に各方法の最適のベクトル演算総数を示す．これらは，次の式に基づくものである．

分割並列消去アルゴリズム(I)　$5k+3p-1$
($k\fallingdotseq 0.775\,n^{\frac{1}{2}}$, $p\fallingdotseq 1.29\,n^{\frac{1}{2}}$ のとき最小値)

Wang の分割法　$12k+8p-16$*
($k\fallingdotseq 0.8\,n^{\frac{1}{2}}$, $p\fallingdotseq 1.25\,n^{\frac{1}{2}}$ のとき最小値)

recursive doubling 法　$17\times\log 2n-3$*

cyclic reduction 法　$19\times\log 2n+1$*

表4から，分割並列消去アルゴリズム(I)は，Wang の分割法の約 40% で済むことがわかる．他の recursive doubling 法や cyclic reduction 法よりも，次元数が大きくなると分割並列消去法のベクトル演算総数

* これらの計算式およびスカラ演算数の式は，文献 6) 表Vより引用．

表 1 分割並列消去アルゴリズム（I）のベクトル演算総数
Table 1 Vector operation counts in the Partitioned Parallel Elimination Method (I).

		三重対角線形方程式を解く場合				右辺の計算			
		ベクトル長	除	乗	加(減)	ベクトル長	除	乗	加(減)
ステップ（2）		p $3p-1$ $2p$	$k-1$	$k-1$	$k-1$	p		$k-1$	$k-1$
ステップ（3）		$n-p$ 2 $3p-2$ $2p-1$ $3p-3$ $2p-2$	1	1 $k-2$ 1	1 $k-2$ 1	1 p $p-1$		1 $k-2$ 1	1 $k-2$ 1
ステップ（4）	p: 偶数	k $2k$ $4k$	2 $p-2$	$p/2+2$ $p/2-2$	$p/2+2$ $p/2-2$	k $2k$		2 $p-2$	2 $p-2$
	p: 奇数	k $2k$ $4k$	2 $p-2$	1 $[p/2]+1$ $[p/2]-1$	1 $[p/2]+1$ $[p/2]-1$	k $2k$		2 $p-2$	2 $p-2$
ステップ（5）		n	1			n	1		
ステップ（2）～（5） p: 偶数			$p+k+1$	$p+2k-1$	$p+2k-1$		1	$p+2k-1$	$p+2k-1$
p: 奇数			$p+k+1$	$p+2k-1$	$p+2k-1$		1	$p+2k-1$	$p+2k-1$
合計		$3p+5k-1$				合計	$2p+4k-1$		

ただし，記号 $[m]$ は m 以下の最大の整数を表す．

表 2 分割並列消去アルゴリズム（I）のスカラ演算総数
Table 2 Scalar operation counts in the Partitioned Parallel Elimination Method (I).

	三重対角線形方程式を解く場合			右辺の計算		
	除	乗	加(減)	除	乗	加(減)
ステップ（2）	$n-p$	$3n-3p-k+1$	$2n-2p$		$n-p$	$n-p$
ステップ（3）	$n-p$	$3n-3p-2k+3$	$2n-2p-k+2$		$n-p$	$n-p$
ステップ（4）						
p: 偶数	$2n-2k$	$3n-4k$	$3n-4k$		$2n-2k$	$2n-2k$
p: 奇数	$2n-2k$	$3n-4k$	$3n-4k$		$2n-2k$	$2n-2k$
ステップ（5）	n			n		
合計						
p: 偶数	$5n-2p-2k$	$9n-6p-7k+4$	$7n-4p-5k+2$	n	$4n-2p-2k$	$4n-2p-2k$
p: 奇数	$5n-2p-2k$	$9n-6p-7k+4$	$7n-4p-5k+2$	n	$4n-2p-2k$	$4n-2p-2k$
スカラ演算総数						
p: 偶数	$21n-12p-14k+6$			$9n-4p-4k$		
p: 奇数	$21n-12p-14k+6$			$9n-4p-4k$		

は多くなってしまうが，他の2方法のように適用計算機や方程式の次元数が2のべき乗である方がよいというような制限はない．分割並列消去アルゴリズムやWangの分割法は，n が2数の積に分解されねばならないという条件はつくが，n が素数の場合は $x_s=1$ $(s=n+1, n+2, \cdots)$ という式を適当な個数付け加える

だけで，いつでも簡単に条件を満たすように変形できる．

（2） スカラ演算総数の比較

各方法のスカラ演算数は，次のようである．

分割並列消去アルゴリズム（Ⅰ）　$21n-12p-14k+1$

分割並列消去アルゴリズム（Ⅱ）　$9n-6$

Wang の分割法　$21n-12p-8k-4$

recursive doubling 法　$\fallingdotseq 7n+17n\times\log_2 n$

cyclic reduction 法　$20n$

これから，並列できるプロセッサの個数 M が小さいときは，分割並列消去アルゴリズム（Ⅱ）が最も良く，ついで cyclic reduction 法と分割並列消去アルゴリズム（Ⅰ）が良いことがわかる．しかし，アルゴリズム（Ⅱ）は，$M\geqq 3$ の場合使用しないプロセッサがあるため，適切な M の値があるはずである．それを調べているのが**表5**である．

表5の計算は次の2つの式に基づく．ただし，記号 $\lceil x\rceil$ は，x 以上の一番小さい整数を示すものとする．

アルゴリズム（Ⅰ）の場合　（p が偶数のとき）

$$\{\lceil p/M\rceil+\lceil(3p-1)/M\rceil+\lceil 2p/M\rceil\}\times(k-1)+\{\lceil(3p-2)/M\rceil+\lceil(2p-1)/M\rceil\}\times(k-2)+\lceil(n-p)/M\rceil+\lceil 2/M\rceil\times 2+\lceil(3p-3)/M\rceil+\lceil(2p-2)/M\rceil+\lceil k/M\rceil\times 2+\lceil 2k/M\rceil\times(2p+2)+\lceil 4k/M\rceil\times(p-4)+\lceil n/M\rceil$$

アルゴリズム（Ⅱ）の場合

$$\lceil 2/M\rceil\times(3n/2-2)+\lceil 4/M\rceil\times n+\lceil(n-2)/M\rceil+\lceil n/M\rceil$$

M が次元数 n に比べて十分大きければ，ベクトル演算回数はアルゴリズム（Ⅰ）の方がはるかに少なくて済む．しかし，$M\leqq 7$ の場合は，アルゴリズム（Ⅱ）の方の処理回数が少なくて済むことを表5は示している．

（3） 誤差について

ガウス消去法のスカラ演算総数は $9n-8$ で，これに最も近い演算数はアルゴリズム（Ⅱ）の $9n-6$ である．誤差の伝播という点からは演算数の少ない方が良い．また，残差を右辺ベクトルとする直接的な反復改良法で解を補正して計算機の精度一杯の解を得ようとするなら，右辺ベクトル1個当りの計算回数の少ない方が有利である[7]．

なお本論文では，添字の計算のために必要な処理や効率的な初期データの格納の仕方の技術はプログラミング段階での問題であるので，一切考慮にいれていないことを断っておきたい．

表3 分割並列消去アルゴリズム（Ⅱ）のベクトル演算総数
Table 3 Vector operation counts in the Partitioned Parallel Elimination Method (Ⅱ).

	ベクトル長	三重対角方程式を解く場合			右辺の計算		
		除	乗	加(減)	除	乗	加(減)
ステップ (2)	2	$n/2-1$				$n/2-1$	$n/2-1$
	4		$n/2-1$	$n/2-1$			
ステップ (3)	2	1				1	1
	4		1	1			
ステップ (4)	$n-2$	1					
	2		$n/2-1$	$n/2-1$		$n/2-1$	$n/2-1$
ステップ (5)	n	1			1		
ステップ (2)～(5)		$n/2+2$	$n-1$	$n-1$	1	$n-1$	$n-1$
総合計		$5n/2$			$2n-1$		

表4 各方法のベクトル演算総数の比較—特にアルゴリズム（Ⅰ）と Wang の分割法の比較—
Table 4 Comparison of the vector operation counts—especially between the Parallel Elimination Method (Ⅰ) and Wang's Parallel Method—.

n ＼ 方法	分割並列消去アルゴリズム(Ⅰ)	Wang の分割法	アルゴリズム(Ⅰ)／Wang の分割法	recursive doubling 法	cyclic reduction 法
$2^5=$ 32	43	96	0.448	82	96
$2^7=$ 128	87	208	0.418	116	134
$2^9=$ 512	175	432	0.405	150	172
$2^{11}=$ 2,048	351	880	0.399	184	210
$2^{13}=$ 8,192	703	1,776	0.396	218	248
$2^{15}=$ 32,768	1,407	3,568	0.394	252	286
$2^{17}=$131,072	2,815	7,152	0.394	286	324
$2^{19}=$524,288	5,631	14,320	0.393	320	362

表 5 分割並列消去アルゴリズム(I),(II)のベクトル演算数と並列プロセッサ数 M との関係
Table 5 The relation between the vector operation counts and the available processor number M.

n \ M	$M=4$				
	アルゴリズム(I)				アルゴリズム(II)
	$k=2$	$k=4$	$k=8$	$k=16$	
$2^5=$ 32	118	136	138	***	94
$2^7=$ 128	478	568	606	610	382
$2^9=$ 512	1,918	2,296	2,478	2,554	1,534
$2^{11}=$ 2,048	7,678	9,208	9,966	10,330	6,142
$2^{13}=$ 8,192	30,718	36,856	39,918	41,434	24,574
$2^{15}=$ 32,768	122,878	147,448	159,726	165,850	98,302
$2^{17}=$131,072	491,518	589,816	638,958	663,514	393,214

n \ M	$M=7$				
	アルゴリズム(I)				アルゴリズム(II)
	$k=2$	$k=4$	$k=8$	$k=16$	
$2^5=$ 32	97	107	101	***	88
$2^7=$ 128	385	409	392	413	355
$2^9=$ 512	1,537	1,630	1,560	1,541	1,425
$2^{11}=$ 2,048	6,145	6,515	6,185	6,151	5,704
$2^{13}=$ 8,192	24,577	26,041	24,728	24,605	22,819
$2^{15}=$ 32,768	98,305	104,158	98,904	98,309	91,281
$2^{17}=$131,072	393,217	416,627	395,561	393,223	365,128

n \ M	$M=8$				
	アルゴリズム(I)				アルゴリズム(II)
	$k=2$	$k=4$	$k=8$	$k=16$	
$2^5=$ 32	78	70	81	***	86
$2^7=$ 128	306	286	304	306	350
$2^9=$ 512	1,218	1,150	1,240	1,278	1,406
$2^{11}=$ 2,048	4,866	4,606	4,984	5,166	5,630
$2^{13}=$ 8,192	19,458	18,430	19,960	20,718	22,526
$2^{15}=$ 32,768	77,826	73,726	79,864	82,926	90,110
$2^{17}=$131,072	311,298	294,910	319,480	331,758	360,446

*** は，p が条件を満たさないため値が存在しないことを示す．

参考文献

1) Stone, H. S.: An Efficient Parallel Algorithm for the Solution of a Tridiagonal Linear System of Equations, *J. ACM*, Vol. 20, No. 1, pp. 27-38 (1973).
2) Stone, H. S.: Parallel Tridiagonal Equation Solvers, *ACM Trans. Math. Softw.*, Vol. 1, No. 4, pp. 289-307 (1975).
3) Dubois, P. and Rodrigue, G.: *An Analysis of the Recursive Doubling Algorithm Organization*, Kuck, D. J., Lawrie, D. H. and Sameh A. H., eds., Academic Press, New York (1977).
4) Hockney, R. W.: A Fast Direct Solution of Poisson's Equation Using Fourier Analysis, *J. ACM*, Vol. 12, No. 1, pp. 95-113 (1965).
5) Lambiotte, J. J. Jr. and Voigt, R. G.: The Solution of Tridiagonal Linear System on the CDC STAR-100 Computer, *ACM Trans. Math. Softw.*, Vol. 1, No. 4, pp. 308-329 (1975).
6) Wang, H. H.: A Parallel Method for Tridiagonal Equation, *ACM Trans. Math. Softw.*, Vol. 7, No. 2, pp. 170-183 (1981).
7) Forsythe, G. and Moler, C. B.: *Computer Solution of Linear Algebraic Systems*, pp. 49-54, Prentice-Hall, Englewood Cliffs, N.J. (1967).

(昭和 62 年 3 月 16 日受付)
(昭和 62 年 7 月 9 日採録)

陳 華萌

昭和 14 年生．昭和 33 年中国武漢大学数学学部数学学科卒業．同年同大学に勤務．電子計算機，自動化などの研究に従事．昭和 55 年同計算機科学学部助教授，現在に至る，計算機アーキテクチャ，分散処理，並列処理，人工知能などに興味をもつ．昭和 54～57 年中国電子計算機学会委員．昭和 56～57 年度京都大学情報工学科招へい外国人学者．

井上 知子

昭和 20 年生．昭和 43 年奈良女子大学理学部数学科卒業．同年京都大学数理解析研究所勤務．昭和 51 年香川大学経済学部管理科学科勤務．昭和 55 年京都大学工学部情報工学科に文部技官として勤務，現在に至る．

萩原 宏（正会員）

大正 15 年生．昭和 25 年京都大学工学部電気工学科卒業．NHK を経て，昭和 32 年京都大学工学部助教授，昭和 36 年同教授，現在に至る．工学博士．情報理論，パルス通信，電子計算機などの研究に従事．昭和 31 年度稲田賞受賞．昭和 50 年本学会論文賞受賞．昭和 56～58 年度本学会副会長．著書「電子計算機通論 1～3」「マイクロプログラミング」など．電子情報通信学会，ACM，IEEE 各会員．