三重対角行列の並列逆行列計算アルゴリズム

Parallel Algorithms for Inverting Tridiagonal Matrices

成　富　　敬
Naritomi Takashi

Abstract

In this paper, new parallel algorithms for inverting tridiagonal matrices are proposed. The algorithms are characterized by their parallel computability originates in bi-reduction method for tridiagonal linear systems. The efficiency of these algorithms are also discussed based on cost analysis. Parallel processing algorithm II is the most efficient one, which has the speedup factor of approximately 2 with two processors.

Keywords : tridiagonal matrix, matrix inversion, bi-reduction method, parallel processing algorithm

1　はじめに

三重対角行列の逆行列を求める問題は，さまざまな科学技術計算における重要な問題のひとつである。たとえば連立一次方程式を解く場合，通常は逆行列を求めずに，LU分解などにより求める。しかし，連立一次方程式の係数行列の一部分を変更して解きたい場合，改めてLU分解をせずに，すでに得られている逆行列をもとに変更後の係数行列の逆行列を求めたほうがよい場合があり，新しい逆行列計算法の開発は不可欠である。

三重対角行列の逆行列を求める研究は文献[1]，[2]，[4]，[5]などがあり，例えば文献[1]では計算量が約$4N^2$のアルゴリズム，また文献[2]では計算量が約$(10/3)N^2$のアルゴリズムが提案されている。いっぽう，本稿で提案し

ているアルゴリズムの計算量は約 $(11/4)N^2$である。

いま，正則で対角優位なN行N列の三重対角行列$\boldsymbol{A}$

$$\boldsymbol{A}=\begin{pmatrix} d_1 & e_1 & & & & & & \\ c_2 & d_2 & e_2 & & & & & \\ & c_3 & d_3 & e_3 & & & & \\ & & \cdots & \cdots & \cdots & & & \\ & & & \cdots & \cdots & \cdots & & \\ & & & & & c_{N-1} & d_{N-1} & e_{N-1} \\ & & & & & & c_N & d_N \end{pmatrix}, \tag{1}$$

の逆行列を考える。また，$\boldsymbol{A}$を係数行列に持つ三重対角連立一次方程式を次のように表す。

$$\boldsymbol{Ax}=\boldsymbol{h}. \tag{2}$$

ここで，$\boldsymbol{x}=(x_1,x_2,\ldots,x_N)^T$，$\boldsymbol{h}=(h_1,h_2,\ldots,h_N)^T$。

このとき，$\boldsymbol{Ax}=\boldsymbol{h}$を解くbi-reduction法は次の三つのステージで表される[7]。ただし，$N=2m$とする。

ステージI　次式により，d'_i, h'_i, d'_k, h'_kを各々計算する。

$$d'_1=d_1,\ d'_i=d_i-\frac{c_i}{d'_{i-1}}e_{i-1}\ (i=2,3,\ldots,m), \tag{3a}$$

$$d'_N=d_N,\ d'_k=d_k-\frac{e_k}{d'_{k+1}}c_{k+1}\ (k=N-1,\ N-2,\ldots,m+1), \tag{3b}$$

$$h'_1=h_1,\ h'_i=h_i-\frac{c_i}{d'_{i-1}}h'_{i-1}\ (i=2,3,\ldots,m), \tag{4a}$$

$$h'_N=h_N,\ h'_k=h_k-\frac{e_k}{d'_{k+1}}h'_{k+1}\ (k=N-1,\ N-2,\ldots,m+1). \tag{4b}$$

ステージII　次式により，x_mとx_{m+1}を各々計算する。

$$x_m=\frac{h'_m d'_{m+1}-h'_{m+1}e_m}{d'_m d'_{m+1}-c_{m+1}e_m}, \tag{5a}$$

$$x_{m+1}=\frac{h'_{m+1}d'_m-h'_m c_{m+1}}{d'_m d'_{m+1}-c_{m+1}e_m}. \tag{5b}$$

ステージIII　次式により，x_iとx_kを各々計算する。

$$x_i=\frac{h'_i-e_i x_{i+1}}{d'_i}\ (i=m-1,m-2,\ldots,1), \tag{6a}$$

$$x_k=\frac{h'_k-c_kx_{k-1}}{d'_k}(k=m+2,\ m+3,\ \ldots,N). \tag{6b}$$

三重対角連立一次方程式を解くアルゴリズムとしては，これまでガウスの消去法やサイクリックリダクション法をはじめさまざまなアルゴリズムが提案されているが[1]，[3]，[6]，本論文で用いるbi-reduction法は文献[6]で提案されているbi-recurrence法と同様，Gaussの消去法に匹敵するほど計算量が少ないという特徴を持っている。

2　Bi-reduction法による逆行列計算

2．1　$\boldsymbol{Ax}=\boldsymbol{h}$の求解と逆行列計算

いま，$\boldsymbol{X}$をN行N列の行列とし，次のように分割する。

$$\boldsymbol{X}=(\boldsymbol{x}_1,\boldsymbol{x}_2,\ \ldots,\boldsymbol{x}_N). \tag{7}$$

ここで，$\boldsymbol{x}_j$は，

$$\boldsymbol{x}_j=(x_{1,j},x_{2,j},\ \ldots,x_{N,j})^T. \tag{8}$$

また，$\boldsymbol{H}$をN行N列の単位行列とし，次のように表す。

$$\boldsymbol{H}=(\boldsymbol{h}_1,\boldsymbol{h}_2,\ \ldots,\boldsymbol{h}_N). \tag{9}$$

ここで，$\boldsymbol{h}_j$は，

$$\begin{aligned}\boldsymbol{h}_j&=(h_{1,j},h_{2,j},\ \ldots,h_{N,j})^T\\&=(\underbrace{0,\ \ldots,0}_{j-1},1,\underbrace{0,\ \ldots,0}_{N-j})^T.\end{aligned} \tag{10}$$

このとき，係数行列$\boldsymbol{A}$の逆行列を求める問題は，$\boldsymbol{AX}=\boldsymbol{H}$の解を求める問題に帰着される。すなわち，逆行列は次のN個の行列方程式を解くことによって得られる。

$$\boldsymbol{Ax}_j=\boldsymbol{h}_j(1\leq j\leq N). \tag{11}$$

2．2　Bi-reduction法による$\boldsymbol{Ax}_j=\boldsymbol{h}_j$の計算

三重対角連立一次方程式$\boldsymbol{Ax}_j=\boldsymbol{h}_j$を以下のように表現する。

$$\begin{pmatrix} d_1 & e_1 & & & & & \\ c_2 & d_2 & e_2 & & & & \\ & \cdots & \cdots & \cdots & & & \\ & c_j & d_j & e_j & & & \\ & & \cdots & \cdots & \cdots & & \\ & & & & c_{N-1} & d_{N-1} & e_{N-1} \\ & & & & & c_N & d_N \end{pmatrix} \begin{pmatrix} x_{1,j} \\ x_{2,j} \\ \vdots \\ x_{j,j} \\ \vdots \\ x_{N-1,j} \\ x_{N,j} \end{pmatrix} = \begin{pmatrix} h_{1,j} \\ h_{2,j} \\ \vdots \\ h_{j,j} \\ \vdots \\ h_{N-1,j} \\ h_{N,j} \end{pmatrix}. \quad (12)$$

このとき，式(12)の求解は，bi-reduction法を用いると次の三つのステージで表される。

ステージI　次式により，d'_i, d'_k, $h'_{i,j}$, $h'_{k,j}$を各々計算する。

$$d'_1=d_1,\ d'_i=d_i-\frac{c_i}{d'_{i-1}}e_{i-1}\ (i=2,3,\ldots,m), \quad (13a)$$

$$d'_N=d_N,\ d'_k=d_k-\frac{e_k}{d'_{k+1}}c_{k+1}\ (k=N-1,N-2,\ldots,m+1). \quad (13b)$$

【$1\leq j\leq m$のとき】

$$h'_{i,j}=0\ (i=1,2,\ldots,j-1), \quad (14a)$$

$$h'_{i,j}=1\ (i=j), \quad (14b)$$

$$h'_{i,j}=h_{i,j}-\frac{c_i}{d'_{i-1}}h'_{i-1,j}\ (i=j+1,j+2,\ldots,m), \quad (14c)$$

$$h'_{k,j}=0\ (k=N,N-1,\ldots,m+1). \quad (14d)$$

【$m+1\leq j\leq N$のとき】

$$h'_{i,j}=0\ (i=1,2,\ldots,m), \quad (14e)$$

$$h'_{k,j}=h_{k,j}-\frac{e_k}{d'_{k+1}}h'_{k+1,j}\ (k=j-1,j-2\ldots m+1), \quad (14f)$$

$$h'_{k,j}=1\ (k=j), \quad (14g)$$

$$h'_{k,j}=0\ (k=N,N-1,\ldots,j+1). \quad (14h)$$

ステージII　次式により，$x_{m,j}$と$x_{m+1,j}$を各々計算する。

$$x_{m,j}=\frac{h'_{m,j}d'_{m+1}-h'_{m+1,j}e_m}{d'_m d'_{m+1}-c_{m+1}e_m}, \quad (15a)$$

$$x_{m+1,j}=\frac{h'_{m+1,j}d'_m-h'_{m,j}c_{m+1}}{d'_m d'_{m+1}-c_{m+1}e_m}. \quad (15b)$$

ステージIII　次式により，$x_{i,j}$と$x_{k,j}$を各々計算する。

$$x_{i,j}=\frac{h'_{i,j}-e_i x_{i+1,j}}{d'_i}\ (i=m-1,m-2,\ldots,1), \tag{16a}$$

$$x_{k,j}=\frac{h'_{k,j}-c_k x_{k-1,j}}{d'_k}\ (k=m+2,\ldots,N). \tag{16b}$$

2．3　コスト計算

表1，表2は，逆行列の第1列目の要素（$\boldsymbol{Ax}_1=\boldsymbol{h}_1$の解）と第$N$列目の要素（$\boldsymbol{Ax}_N=\boldsymbol{h}_N$の解），逆行列の第2列目の要素（$\boldsymbol{Ax}_2=\boldsymbol{h}_2$の解）と第$N-1$列目の要素（$\boldsymbol{Ax}_{N-1}=\boldsymbol{h}_{N-1}$の解）を，それぞれbi-reduction法により求めた場合の時間コストを示している。ここで，$\boldsymbol{x}_3,\boldsymbol{x}_4,\ldots,\boldsymbol{x}_m,\boldsymbol{x}_{m+1},\ldots,\boldsymbol{x}_{N-3},\boldsymbol{x}_{N-2}$を求めるコストに関する表は省略している。

したがって，逐次処理により逆行列を求めるときのコスト：C_{seq}は，

$$\begin{aligned}C_{seq}&=3(N-2)\\&\quad+(m-2)+7+3(m-1)+2(N-m-1)+(N-m-2)+4+2(m-1)\\&\quad+3(N-m-1)\\&\quad+(m-3)+4+3(m-1)+2(N-m-1)+(N-m-3)+4+2(m-1)\\&\quad+3(N-m-1)\\&\quad\vdots\qquad\vdots\\&\quad+1+4+3(m-1)+2(N-m-1)+1+4+2(m-1)+3(N-m-1)\\&\quad+0+4+3(m-1)+2(N-m-1)+0+4+2(m-1)+3(N-m-1)\\&\quad+0+4+3(m-1)+2(N-m-1)+0+4+2(m-1)+3(N-m-1)\\&=\frac{11}{4}N^2-\frac{1}{2}N-1.\end{aligned} \tag{17}$$

ここで，式(17)の計算の際，式(13a)，式(13b)におけるd'_i，d'_k，あるいは式(15a)，式(15b)における$d'_m d'_{m+1}-c_{m+1}e_m$などは，$\boldsymbol{Ax}_1=\boldsymbol{h}_1$の求解のときに一度計算しておけばよいため，$\boldsymbol{Ax}_2=\boldsymbol{h}_2,\ldots,\boldsymbol{Ax}_N=\boldsymbol{h}_N$のコストには含めていない。

表 1 ：x_1とx_Nを求めるコスト

	$Ax_1=h_1$			$Ax_N=h_N$		
	±	×	÷	±	×	÷
$d'_i(1\leq i\leq m)$ $d'_k(m+1\leq k\leq N)$	m-1 N-m-1	m-1 N-m-1	m-1 N-m-1	m-1 N-m-1	m-1 N-m-1	m-1 N-m-1
$h'_{i,1},h'_{i,N}(1\leq i\leq m)$	0	m-2	0	0	0	0
$h'_{k,1},h'_{k,N}(m+1\leq k\leq N)$	0	0	0	0	N-m-2	0
$x_{m,1},x_{m,N}$	1	3	1	0	1	1
$x_{m+1,1},x_{m+1,N}$	0	1	1	1	3	1
$x_{i,1},x_{i,N}(m-1\geq i\geq 1)$	m-1	m-1	m-1	0	m-1	m-1
$x_{k,1},x_{k,N}(m+2\leq k\leq N)$	0	N-m-1	N-m-1	N-m-1	N-m-1	N-m-1

表 2 ：x_2とx_{N-1}を求めるコスト

	$Ax_2=h_2$			$Ax_{N-1}=h_{N-1}$		
	±	×	÷	±	×	÷
$d'_i(1\leq i\leq m)$ $d'_k(m+1\leq k\leq N)$	m-1 N-m-1	m-1 N-m-1	m-1 N-m-1	m-1 N-m-1	m-1 N-m-1	m-1 N-m-1
$h'_{i,2},h'_{i,N-1}(1\leq i\leq m)$	0	m-3	0	0	0	0
$h'_{k,2},h'_{k,N-1}(m+1\leq k\leq N)$	0	0	0	0	N-m-3	0
$x_{m,2},x_{m,N-1}$	1	3	1	0	1	1
$x_{m+1,2},x_{m+1,N-1}$	0	1	1	1	3	1
$x_{i,2},x_{i,N-1}(m-1\geq i\geq 1)$	m-1	m-1	m-1	0	m-1	m-1
$x_{k,2},x_{k,N-1}(m+2\leq k\leq N)$	0	N-m-1	N-m-1	N-m-1	N-m-1	N-m-1

3　逆行列計算の並列処理アルゴリズム

Bi-reduction法はもともと並列計算性を持っている。すなわち，各$Ax_j=h_j$ $(1\leq j\leq N)$の計算に対して，式(3a)，式(4a)，式(5a)，式(6a)におけるの添字iの系列：$d'_i,h'_{i,j},x_{i,j}(1\leq i\leq m)$の処理と式(3b)，式(4b)，式(5b)，式(6b)におけるの添字kの系列：$d'_k,h'_{k,j},x_{k,j}(m+1\leq k\leq N)$の処理をほぼ独立に実行することができる。以下，２個のプロセッサによる並列度２の並列処理アルゴリズムＩと並列処理アルゴリズムII，そしてN個のプロセッサによる並列度Nの並列処理アルゴリズムIIIについて述べる。

3．1　並列処理アルゴリズムＩ（並列度２）

表１，表２において，各jに対して，添字iの系列：$d'_i,h'_{i,j},x_{i,j}(1\leq i\leq m)$に関

する処理をプロセッサ1に割り当て，添字kの系列：$d'_k, h'_{k,j}, x_{k,j}$ $(m+1 \leq k \leq N)$に関する処理をプロセッサ2に割り当てる。アルゴリズムは以下のとおりである。

ステージI　$d'_i (1 \leq i \leq m)$，$d'_k (m+1 \leq k \leq N)$を，それぞれプロセッサ1，プロセッサ2に入力する。

ステージII　$\boldsymbol{A}\boldsymbol{x}_j = \boldsymbol{h}_j (1 \leq j \leq N)$，を順番に解く。

ステージIII　$x_{i,j} (m \leq i \leq 1)$，$x_{k,j} (m+1 \leq k \leq N)$より逆行列の各要素を構成し，出力する。

各プロセッサにおける計算負荷を表3に示す。表3において，たとえば$j=1$のとき，プロセッサ1によるd'_i，$h'_{i,1}$，$x_{i,1}$のコストがそれぞれ$3(m-1)$，$m-2$，$5+3(m-1)$，また，プロセッサ2によるd'_k，$h'_{k,1}$，$x_{k,1}$のコストがそれぞれ$3(N-m-1)$，0，$5+2(N-m-1)$であることを示している。

したがって，並列処理アルゴリズムIにより逆行列を求めるときのコスト：$C_{para\,I}$は，

$$C_{para\,I} = \frac{7}{4}N^2 - N + 2 . \tag{18}$$

3.2　並列処理アルゴリズムII（並列度2）

並列処理アルゴリズムIでは，たとえば，$j=1$のときに$h'_{i,1}$と$h'_{k,1}$を求める際，プロセッサ1の方はコスト$m-2$の処理を実行する必要があるが，プロセッサ2の方は遊んでいる状態にある。しかも，次の$x'_{i,1}$と$x'_{k,1}$の計算を開始するためには互いのプロセッサ間で$h'_{m,1}$と$h'_{m+1,1}$を交換する必要があるため，ここで同期を取らなければならない。そのため，$1 \leq j \leq m$のときプロセッサ1の負荷が重く，$m+1 \leq j \leq N$ではプロセッサ2の負荷が重くなり，計算負荷が不均等になってしまう。

並列処理アルゴリズムIIでは，並列処理アルゴリズムIで発生した計算負荷の偏りを解消するため，たとえば，$j=1$に対する処理と$j=N$に対する処理を組み合わせて実行する手法を取り入れている。これにより各プロセッサにかかる計算負荷の均等化を実現している。このとき，添字iの系列：$d'_i, h'_{i,j}, x_{i,j}$

表3：並列処理アルゴリズムIによる各プロセッサの計算負荷

		プロセッサ1	プロセッサ2
$(j=1)$	d'_i, d'_k	3(m-1)	3(N-m-1)
	$h'_{i,1}, h'_{k,1}$ $x_{i,1}, x_{k,1}$	m-2 5+3(m-1)	0 5+2(N-m-1)
$(j=2)$	$h'_{i,2}, h'_{k,2}$ $x_{i,2}, x_{k,2}$	m-3 2+3(m-1)	0 2+2(N-m-1)
⋮	⋮	⋮	⋮
$(j=N-1)$	$h'_{i,N-1}, h'_{k,N-1}$ $x_{i,N-1}, x_{k,N-1}$	0 2+2(m-1)	N-m-3 2+3(N-m-1)
$(j=N)$	$h'_{i,N}, h'_{k,N}$ $x_{i,N}, x_{k,N}$	0 2+2(m-1)	N-m-2 2+3(N-m-1)

$(1\leq i\leq m)$に関する処理をプロセッサ1に割り当て，添字kの系列：$d'_k, h'_{k,j}$, $x_{k,j}(m+1\leq k\leq N)$に関する処理をプロセッサ2に割り当てることに関しては，並列処理アルゴリズムIと同じである。アルゴリズムを以下に示す。

ステージI　$d'_i(1\leq i\leq m)$，$d'_k(m+1\leq k\leq N)$を，それぞれプロセッサ1，プロセッサ2に入力する。

ステージII　$\boldsymbol{A}\boldsymbol{x}_j=\boldsymbol{h}_j$と$\boldsymbol{A}\boldsymbol{x}_{N-j+1}=\boldsymbol{h}_{N-j+1}$，$1\leq j\leq m$，を組にして順番に解く。たとえば，プロセッサ1は$j=1$と$j=N$の添字$i$の系列に関する処理を担当し，プロセッサ2は$j=1$と$j=N$の添字$k$の系列に関する処理を担当する。

ステージIII　$x_{i,j}(m\geq i\geq 1)$，$x_{k,j}(m+1\leq k\leq N)$より逆行列の各要素を構成し，出力する。

各プロセッサにおける計算負荷を表4に示す。表4において，たとえば，$j=1/j=N$のとき，スラッシュ記号(/)の左側は$j=1$に対するコスト，右側は$j=N$に対するコストである。すなわち，プロセッサ1では，$h'_{i,1}, h'_{i,N}$のコストがそれぞれm-2，0であり，$x_{i,1}, x_{i,N}$のコストがそれぞれ5+3$(m$-1$)$，2+2$(m$-1$)$であることを示している。

したがって，並列処理アルゴリズムIIにより逆行列を求めるときのコスト：$C_{para\,II}$は，

表4：並列処理アルゴリズムIIによる各プロセッサの計算負荷

		プロセッサ1	プロセッサ2
$(j=1/j=N)$	d'_i, d'_k	3(m-1)	3(N-m-1)
	$h'_{i,1}/h'_{i,N}, h'_{k,1}/h'_{k,N}$ $x_{i,1}/x_{i,N}, x_{k,1}/x_{k,N}$	m-2/0 5+3(m-1)/2+2(m-1)	0/N-m-2 5+2(N-m-1)/2+3(N-m-1)
$(j=2/j=N-1)$	$h'_{i,2}/h'_{i,N-1}, h'_{k,2}/h'_{k,N-1}$ $x_{i,2}/x_{i,N-1}, x_{k,2}/x_{k,N-1}$	m-3/0 2+3(m-1)/2+2(m-1)	0/N-m-3 2+2(N-m-1)/2+3(N-m-1)
⋮	⋮	⋮	⋮
$(j=m/j=m+1)$	$h'_{i,m}/h'_{i,m+1}, h'_{k,m}/h'_{k,m+1}$ $x_{i,m}/x_{i,m+1}, x_{k,m}/x_{k,m+1}$	0/0 2+3(m-1)/2+2(m-1)	0/0 2+2(N-m-1)/2+3(N-m-1)

$$C_{paraII}=\frac{11}{8}N^2-\frac{1}{4}N+1. \tag{19}$$

3.3　並列処理アルゴリズムIII（並列度N）

並列処理アルゴリズムIIIは，並列処理アルゴリズムIIの並列計算性を高めて並列度Nにしたアルゴリズムであり，アルゴリズムは以下のとおりである。

ステージI　$d'_i(1\leq i\leq m)$，$d'_k(m+1\leq k\leq N)$を，それぞれN個のプロセッサに入力する。

ステージII　$\boldsymbol{A}\boldsymbol{x}_j=\boldsymbol{h}_j$と$\boldsymbol{A}\boldsymbol{x}_{N-j+1}=\boldsymbol{h}_{N-j+1}$，$1\leq j\leq m$，に関する処理を，それぞれプロセッサ$2j-1$とプロセッサ$2j$に割り当て，N個のプロセッサで同時に処理する。

ステージIII　$x_{i,j}(m\geq i\geq 1)$，$x_{k,j}(m+1\leq k\leq N)$より逆行列の各要素を構成し，出力する。

このアルゴリズムでは，もっとも計算負荷の重いプロセッサは$j=1$と$j=N$の処理を担当するプロセッサ1とプロセッサ2であり，もっとも計算負荷の軽いプロセッサは$j=m$と$j=m+1$の処理を担当するプロセッサ2m-1とプロセッサ2mである。

表4より，並列処理アルゴリズムIIIにより逆行列を求めるときのコスト：$C_{paraIII}$は，

$$C_{paraIII}=\frac{9}{2}N-3. \tag{20}$$

4　考察とまとめ

本稿で提案した，それぞれの並列処理アルゴリズムのコスト，スピードアップ率，効率の比較を表5に示す。表5において，S_{paraI}，S_{paraII}はそれぞれ並列処理アルゴリズムI，並列処理アルゴリズムIIのスピードアップ率を，また，E_{paraI}，E_{paraII}はそれぞれ並列処理アルゴリズムI，並列処理アルゴリズムIIの並列処理の効率を示している。ここで，並列処理アルゴリズムIIIについては表5に記述していないが，その効率は約0.61である。各並列処理アルゴリズムのなかでは，明らかに並列処理アルゴリズムIIがスピードアップ，効率ともに勝っている。特に並列処理アルゴリズムIIでは，スピードアップは2個のプロセッサでほぼ2倍となり並列処理の効率もほぼ1である。

これまで，実際の処理において問題となる，プロセッサ間でのデータ交換に伴う通信時間については考慮していない。並列処理アルゴリズムIでは，各プロセッサ間で交換する必要のあるデータは，各jに対して$h'_{m,j}$と$h'_{m+1,j}$のみである。いっぽう，並列処理アルゴリズムIIでは，たとえば($j=1$ / $j=N$)のとき，プロセッサ1が二つの方程式の系列iの計算を担当し，プロセッサ2も同じく二つの方程式の系列kの計算を担当している。このため，方程式の求解終了時点で，$\boldsymbol{A}\boldsymbol{x}_1=\boldsymbol{h}_1$と$\boldsymbol{A}\boldsymbol{x}_N=\boldsymbol{h}_N$の解系列のうち，$x_{i,1}(1\leq i\leq m)$と$x_{i,N}(1\leq i\leq m)$はプロセッサ1に保持されており，$x_{k,1}(m+1\leq k\leq N)$と$x_{k,N}(m+1\leq k\leq N)$はプロセッサ2に保持されている。そのため，求解により得られたベクトルをもとに逆行列の各要素を構成し，出力する点については，並列処理アルゴリズムIIのほうが手間がかかると考えられる。

ところで，逆行列計算の並列度を向上するためには，分割統治法を利用することもできる[6]。また，メモリを共有する場合と共有しない場合での処理効率への影響の違いも考えられる。これらは残された課題である。

表5：並列処理アルゴリズムのコスト，スピードアップ率，効率の比較

N	C_{seq}	C_{para} I	C_{para} II	S_{para} I	S_{para} II	E_{para} I	E_{para} II
10	269	167	141	1.6107784	1.9078014	0.8053892	0.9539007
100	27,449	17,402	13,776	1.5773474	1.9925232	0.7886737	0.9962616
1,000	2,749,499	1,749,002	1,375,251	1.5720388	1.9992707	0.7860194	0.9996353
10,000	274,994,999	174,990,002	137,502,501	1.5714898	1.9999273	0.7857449	0.9999636
100,000	27,499,949,999	17,499,900,002	13,750,025,001	1.5714347	1.9999927	0.7857173	0.9999964
1,000,000	2,749,999,499,999	1,749,999,000,002	1,375,000,250,001	1.5714292	1.9999993	0.7857146	0.9999996

謝辞

本研究の一部は山口大学経済学部学術振興基金の助成による。ここに記して深謝します。

参考文献

[1] Bondeli, S., Divide and conquer : a parallel algorithm for the solution of a tridiagonal linear system of equations, Parallel Computing, vol.17, no.4-5, pp.419-434, 1991.

[2] Climent, J. J., Tortosa L. and Zamora A., A BSP Recursive Divide and Conquer Algorithm to Compute the Inverse of a Tridiagonal Matrix, J. Parallel and Distributed Computing, vol.59, pp.423-444, 1999.

[3] Evans, D. J., Parallel numerical algorithms for linear systems, Parallel Processing Systems by Evans, D. J. ed., pp.357-384, Cambridge, 1982.

[4] Lewis, J. W., Inversion of Tridiagonal Matrices, Numer. Math., vol.38, pp. 333-345, 1982.

[5] Meurant, G., A Review on the Inverse of Symmetric Tridiagonal and Block Tridiagonal Matrices, SIAM J. Matrix Anal. Appl., vol.13, no.3, pp.707-728, 1992.

[6] Naritomi, T. and Aso, H., A highly parallel systolic tridiagonal solver, IEICE Trans. Inf. & Syst., Vol.E79-D, No.9, pp.1241-1247, 1996.

[7] Naritomi, T, Bi-reduction : A parallel algorithm for tridiagonal linear systems, INFORMATION, vol.3, no.4, pp.479-484, 2000.