分散メモリ型並列計算機に向くHessenberg形への変換アルゴリズムとその有効性

片桐孝洋† 金田康正††

密行列の固有値を計算する場合において，Householder法を用いて多くの零要素を持つ行列に変換することは重要な最初の手順である．本論文ではこの変換アルゴリズムをデータ分割方式に基づく並列アルゴリズムについて，計算量と通信量の観点から詳細に検討し，256 PE (Processing Elements) からなる富士通の分散メモリ型並列計算機AP1000+，および日立のSR2201に実装して評価を行った．その結果，従来のアルゴリズムではプロセッサ台数が増加すると通信時間も増加するという問題が生じるのに対して，我々の提案するアルゴリズムではプロセッサ台数が増加するほど，通信時間を削減できることが明らかとなった．一方で，従来のアルゴリズムが我々のアルゴリズムに対して有効となる場合が理論上ありうることも明らかとなった．この条件を定量的に求め，その理論的な予測と実機での実行結果を比較した．

An Unblocked Hessenberg Reduction Algorithm for Distributed Memory Architecture Parallel Machines and Its Effectiveness

TAKAHIRO KATAGIRI† and YASUMASA KANADA††

Reduction of the matrix with many zero entries without changing its eigenvalue is the first important procedure for the eigenvalue problem. The reduction has been done by using Householder method, in many cases. In this paper, parallel algorithms for the reduction based on the data distribution stratagems were discussed in detail from the point of computing complexity and communication cost. We implemented these algorithms on the Fujitsu AP1000+ and the Hitachi SR2201 both with 256 PE (Processing Element) and evaluated these reduction performances. In our analysis, we found the fact that our parallel algorithm decreases communication times when the number of PE's is increased, but conventional algorithm increases this communication time. Moreover, we theoretically found that there is the case that the conventional algorithm is superior to our algorithm. We also could show the condition for the suitable algorithm. The condition was proved with the experiment with the AP1000+ and the SR2201.

1. はじめに

多くの工学的，物理的問題が固有値計算の問題に帰着できることはいうまでもない．同時に，固有値計算を高速処理しなくてはならないという問題に，現在も多くの技術者や研究者は直面している．この固有値計算において，相似変換を用いて多くの零要素を持つ行列に変換することは重要な最初の手順である．相似変換によく用いられる方法としてHouseholder法を用いた変換がある．

† 東京大学大学院理学系研究科情報科学専攻
Department of Information Science, Graduate School of Science, The University of Tokyo
†† 東京大学大型計算機センター
Computer Centre, The University of Tokyo

相似変換を非対称実数密行列に対して行う場合，Hessenberg形という形に変換 (Hessenberg reduction) するが，このHouseholder法を用いた相似変換の計算のオーダが $O(n^3)$ と膨大であるため，並列処理による高速化が数多く試みられている．たとえば，一般行列を処理する場合，行列データの分割方式で分類すると，列ブロック分割ではStewart[1]，列サイクリック分割ではDongarraら[2]が並列アルゴリズムの提案をしている．一方，対称行列用に特化した同種のアルゴリズムとして，列ブロック分割ではDongarraら[2]，Kalamboukis[3]，サイクリックサイクリック分割ではChangら[4]やHendricksonら[5]が並列アルゴリズムを提案している．

本論文では，次に示す3つの内容を含んでいる．いま，問題サイズを n，使用するPE (Processing Element)

数を p とする．このとき，Hessenberg reduction においてデータ分割のみを並列計算の前提とする従来の並列アルゴリズムでは，通信量が $O(n^2 \log_2 p)$ となり，PE 数が増えるにつれて通信時間が増加するという問題が生じていた[1]．それに対して，本論文で通信量が $O((n^2/\sqrt{p})\ \log_2 p)$ となるような PE 数が増加するにつれて通信量が減少する並列アルゴリズムを提案することが第 1 の内容である．第 2 の内容は，通信量の観点からは提案するアルゴリズムよりも低速であると予想される従来のアルゴリズムが，提案するアルゴリズムよりも高速となりうることを定性的に示し，またその正当性を実機で評価することである．第 3 の内容は，我々の提案するアルゴリズムは PE 数が増加すればするほど，従来のアルゴリズムに対して有効となることの定量的評価である．

まず次章で，並列変換プログラムが動作する計算機環境の仮定と行列等の表記法について述べる．3 章で逐次アルゴリズムについて述べ，4 章で合計で 4 種類のデータ分割方式による並列アルゴリズムを説明する．5 章では，提案するアルゴリズムを含む，計 4 種類の各分割におけるロードバランスを含めた計算量，メッセージの通信量の解析を行う．6 章で，それら各データ分割における計算法を 256 PE（Processing Element）からなる富士通の AP1000+，および日立の SR2201 に実装し，それぞれの実行時間を測定した結果を示す．また我々が求めた実行時間予測式の予測精度の評価と，それから得られる通信時間の占める割合を吟味することで，従来のアルゴリズムに対して，我々のアルゴリズムが大規模並列処理で有利となることを示す．最後に，本論文の結論および今後の課題を述べる．

2. 並列実行環境と表記法

ここで並列計算機は p 個の均質的な PE で構成され，それらは一次元的には $P_1, P_2, \cdots, P_p$ とラベル付けされているものとする．また二次元的には $d \equiv \sqrt{p}$ とすると，$d \times d$ のように PE が構成されているとする．さらに各 PE はメッセージの放送（一対多の通信）や，各 PE が局所的に所有するデータに対して総和演算を行うような計算（一対一もしくは全対全などの通信により実現）が行えるように，ネットワーク網により結合されているものとする．

定式化のために，**表 1** に示す表記を用いることとする．また詳細な行列のデータ分割方式と表記法の説明については文献 6) を参照されたい．

表 1 数学表記法とそれらの説明
Table 1 Mathmatical notations and its illustration.

表記	説明
α, μ	スカラ $\in \Re$
x, y, u	ベクトル $\in \Re^n$
A	行列 $\in \Re^{n \times n}$
$[A]_{i:j,k:l}$	A の行 $i, \cdots, j$，列 $k, \cdots, l$ 部分行列
$[A]_{i:j,k}$	A の行 $i, \cdots, j$，列 k 部分ベクトル
$A^{(k)}$	反復 k における行列 A

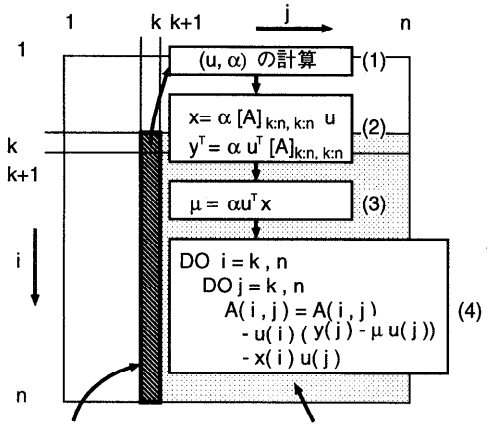


図 1 Hessenberg reduction の逐次アルゴリズム
Fig. 1 Sequential algorithm for Hessnberg reduction.

3. Householder 変換の概要

本論文は Householder 変換を用いた Hessenberg reduction そのものに関する論文ではないので，詳細な説明は行わず概要のみを述べる．詳細な説明は文献 1), 2) を参照されたい．

いま行列 $A^{(1)} = A$ から Hessenberg 形 $A^{(n-2)}$ への変換を行うことを考える．この変換処理は**図 1** のようになることが知られている[1),2)]．

図 1 に示したように，反復 k における消去過程では消去対象行列 $[A]_{k:n,k:n}$ における列ベクトル $[A]_{k:n,k}$ が必要になる．このベクトルのことを枢軸ベクトル生成列と呼び，後に (u, α) の計算により計算されたベクトル u を枢軸ベクトル（pivot vector）と呼ぶことにする．ここで α は $[A]_{k:n,k}$ のノルムを用いて計算されるスカラ値である．

4. 並列アルゴリズムの説明

4.1 (*, Block), (*, Cyclic) 列分割方式の概要

説明を簡単にするため，行列の分割方式を (*, Block)，すなわち $n \times n/p$ の大きさの部分行列を各 PE に割り当てる分割方式を考える．このとき図 1 に

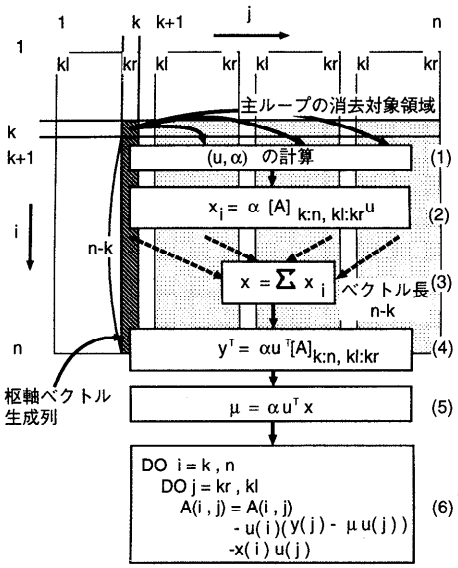


図2 Hessenberg reductionの並列アルゴリズム（列分割方式）
Fig. 2 Parallel algorithm for Hessenberg reduction (column wise distribution).

示す計算を行うことを考える．この分割方式による計算法は図2のようになる．

行列の分割方式が (*, Cyclic) の場合，すなわち各PEに1行のみ割り当てる場合を考え，さらに残りの列分だけ同じように循環して割り当てていく方式では，(*, Block) 列分割方式で各PEが1行のみ所有していると考えることができる．よって，同様な計算方式で処理できることが分かる．

(*, Cyclic) 列分割方式での本計算法は，Dongarraら[2]が提案したものと類似しているが，枢軸ベクトル u の計算を枢軸ベクトル生成列を受信後行うことで，スカラ α を放送する手間を削減している点が異なる．

4.2 (Block, Block), (Cyclic, Cyclic) 格子分割方式の概要

前述の従来の列分割方式では，枢軸ベクトル生成列すべてが得られるまで，計算を進めることはできなかった．ところがこの提案する格子分割方式でのアルゴリズムは，部分的に枢軸ベクトルを得るだけで計算を進めることができる．

格子分割方式は行および列の両方に，先述の列分割方式を適用した分割方式といえる．格子分割方式に基づく計算法☆では，枢軸ベクトル生成列に関する (a) 行方向の放送の後に行う処理（Au の積），(b) 列方向の放送の後に行う処理（u^TA の積），の2つに分けて考えることができる（図3の格子分割方式の例）．

(Cyclic, Cyclic) 格子分割方式についても各PEが行列の1要素ずつ所有していると考えると，同様な計算法で行えることが分かる．(Cyclic, Cyclic) 格子分割方式での本アルゴリズムのアイデアは，基本的にはChangら[4]やHendricksonら[5]によって提案された対称行列用の並列アルゴリズムにも用いられている．

☆ 対称行列を処理する場合 scattered square 方式[4]として知られている．この分割方式は本論文では (Cyclic, Cyclic) 格子分割方式に相当する．

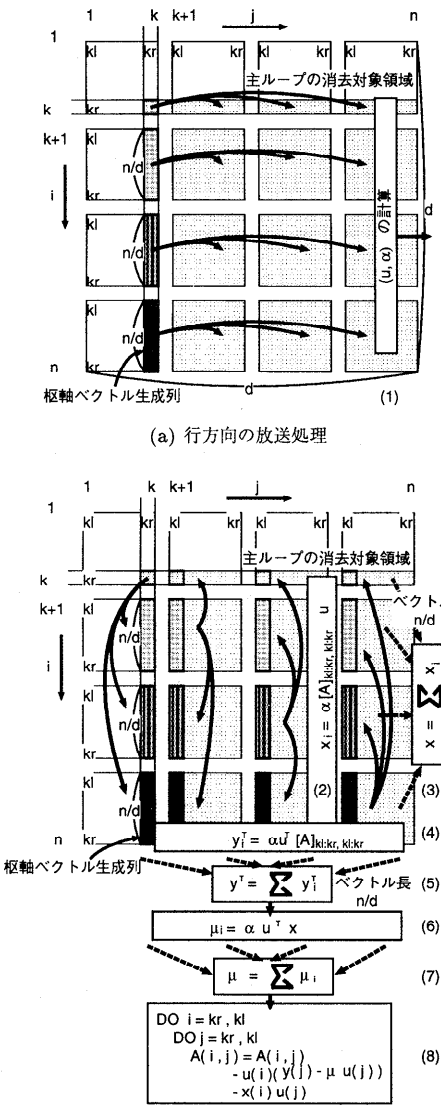


図3 Hessenberg reductionの並列アルゴリズム（格子分割方式）
Fig. 3 Parallel algorithm for Hessenberg reduction (grid wise distribution).

表2 並列計算機のパラメータ
Table 2 The parameter of parallel computers for our estimate model.

表記	説明
ϕ	倍精度演算1回あたりの演算時間
$\delta_c(p)$	通信立ち上がり時間（一対一通信）
$\tau_c(p)$	メッセージ1バイトの転送時間（一対一通信）
$\delta_b(p)$	通信立ち上がり時間（放送）
$\tau_b(p)$	メッセージ1バイトの転送時間（放送）
$\gamma(\sqrt{p})$	倍精度実数のリダクション演算時間

5. 性能予測と計算量および通信量の解析

5.1 性能予測モデル

ここでは，各分散方式を分散メモリ型並列計算機上で並列実行する場合の実行時間の予測をするためのモデルを記す．まず本モデルにおける分散メモリ型並列計算機の各種評価パラメータを**表2**に示す．

ここで一対一通信の時間は，実装方式により通信相手の距離に依存するが，ここでは，メッセージ N バイトを1回転送する時間は N に関する一次式で近似できると仮定する．このとき

$$\text{メッセージの転送時間} = \delta_c(p) + N\cdot\tau_c(p) \quad (1)$$

である．同様に放送の時間も近似できると仮定する．

各分割方式において，問題サイズ n，使用するPE数 p の条件で問題を解く場合の計算時間を $T_{calc}(n,p)$，通信時間を $T_{comm}(n,p)$ とする．ここで以下の仮定のうえで，通信時間と計算時間を見積もる．

[仮定1] 並列実行時間は計算時間，通信時間に分けられる．さらに通信時間は，一対一通信時間，放送時間，およびスカラリダクション演算時間に分けられる．

[仮定2] 並列Hessenberg reductionでの通信処理において，すべてのPEが必要となる通信データ（枢軸ベクトル u やスカラ α）を受信し計算可能となるまで，処理を進めることはない．

[仮定3] 計算時間は，データ分散にともなう各プロセッサでの計算時間のうち最大のもので定まるとする．

いま計算時間 $T_{calc}(n,p)$ は総演算量を $Calc\ (n,p)$ とすると

$$T_{calc}(n,p) = Calc(n,p) \times \phi \quad (2)$$

である．また一対一通信の起動回数を $CommT(n,p)$，一対一通信の総メッセージ転送量を $CommS(n,p)$ とする．さらに，放送の起動回数を $BroadT(n,p)$，放送の総メッセージ転送量を $BroadS(n,p)$ とする．また $ReductS(n)$ をスカラリダクション回数とする．このとき通信時間 $T_{comm}(n,p)$ は

$$\begin{aligned} T_{comm}(n,p) = {} & CommT(n,p)\cdot\delta_c(p) + CommS(n,p)\cdot\tau_c(p) \\ & + BroadT(n,p)\cdot\delta_b(p) + BroadS(n,p)\cdot\tau_b(p) \\ & + ReductS(n)\cdot\gamma(\sqrt{p}) \end{aligned} \quad (3)$$

となる．このとき全体の理論的な実行時間 $T_{theory}(n,p)$ は以下のようになる．

$$T_{theory}(n,p) = T_{calc}(n,p) + T_{comm}(n,p) \quad (4)$$

また倍精度演算1回あたりの演算時間 ϕ は，問題サイズやプロセッサ台数，分割方式の違いなどで一般に異なり，推定は困難である．しかしながら実行時間の予測をするためには，このパラメータを得なければならない．この値の見積りの方法の1つとして，プログラム中の通信処理をすべて取り除いて，その実行時間測定して推定する方法も考えられる．しかしここでは以下のようにこの値を推定する．

いま，ある問題サイズ $\hat{n}$，およびあるプロセッサ台数 $\hat{p}$ に固定して測定した実行時間を $T_{actual}(\hat{n},\hat{p})$ とする．さらに通信性能に関する各パラメータはすでに測定済とする．このとき通信処理の予測時間 $T_{comm}(\hat{n},\hat{p})$ も得ることが可能である．したがって，

$$T_{actual}(\hat{n},\hat{p}) = Calc(\hat{n},\hat{p})\phi + T_{comm}(\hat{n},\hat{p}) \quad (5)$$

と仮定すると，ϕ は

$$\phi = \frac{T_{actual}(\hat{n},\hat{p}) - T_{comm}(\hat{n},\hat{p})}{Calc(\hat{n},\hat{p})} \quad (6)$$

と推定される．この ϕ の値を $T_{theory}(n,p)$ の推定に用いることにする．

5.2 演 算 量

Householder法を用いたHessenberg reductionにおける基本的な演算（乗算および加算）回数について考える．なおここでの解析には，近似的な計算量を扱っている点に注意する．

いま各反復時における主ループの適用範囲を考えるが，各PEが局所的に所有する行列の適用範囲を $Row \times Col$ とする．このときHouseholder法を用いたHessenberg reductionを行う各分割方式の反復1から $n-2$ までの主ループ適用範囲は**表3**のようになる．

同一の分割方式でも，一般に各反復で各PE($P_1, P_2, \cdots, P_p$)における適用範囲の大きさが異なる．表3では各反復でPEがとりうる最大の適用範囲を示していることに注意されたい．この表3から分割方式の違いによる計算量の違いを計算することができる．

さて次に図1(1)のノルム計算と図1(3)の内積計算 $\mu = u^T x$ の演算量とを考えよう．この計算量はそれぞれ同じであるので，片方だけを**表4**に示す．

次に図1(2)の行列ベクトル演算 Au，$u^T A$ を考え

表 3 各分割における主ループの適用範囲とその表記
Table 3 The range of main loop and its notation.

分割方式	*Row*	*Col*
(*, Block)	$Row_b(i,k,n,p) = n-(k-1)\frac{n}{p} - i + 1$ $(i=1,\ldots,\frac{n}{p},\ k=1,\ldots,p)$	$Col_b(n,p) = \frac{n}{p}$
(*, Cyclic)	$Row_c(i,k,n,p) = n-(k-1)p - i + 1$ $(i=1,\ldots,p,\ k=1,\ldots,\frac{n}{p})$	$Col_c(k,n,p) = \frac{n}{p} - k + 1$ $(k=1,\ldots,\frac{n}{p})$
(Block, Block)	$Row_{bb}(n,p) = \frac{n}{\sqrt{p}}$	$Col_{bb}(n,p) = \frac{n}{\sqrt{p}}$
(Cyclic, Cyclic)	$Row_{cc}(k,n,p) = \frac{n}{\sqrt{p}} - k + 1$ $(k=1,\ldots,\frac{n}{\sqrt{p}})$	$Col_{cc}(k,n,p) = \frac{n}{\sqrt{p}} - k + 1$ $(k=1,\ldots,\frac{n}{\sqrt{p}})$

表 4 ノルム計算および内積計算の計算量
Table 4 The computing complexities for 2-norm or dot products.

分割方式	$Calc(n,p)$
(*, Block)	$\sum_{i=1}^{n-2} 2(n-i+1) = n^2+n+O(1)$
(*, Cyclic)	$\sum_{i=1}^{n-2} 2(n-i+1) = n^2+n+O(1)$
(Block, Block)	$2\frac{n}{\sqrt{p}} \times \frac{n}{\sqrt{p}} \times (\sqrt{p}-1) + \sum_{k=1}^{n/\sqrt{p}-2} 2\left(\frac{n}{\sqrt{p}} - k + 1\right) = \left(\frac{2}{\sqrt{p}} - \frac{1}{p}\right)n^2 + \frac{1}{\sqrt{p}}n + O(1)$
(Cyclic, Cyclic)	$\sum_{k=1}^{n/\sqrt{p}} 2\left(\frac{n}{\sqrt{p}} - k + 1\right) \times \sqrt{p} = \frac{1}{\sqrt{p}}n^2 + n$

る．これらの各 PE での演算量も同一である．よって片方の計算量を示すと**表 5** のようになる．表 5 の計算量は PE 内のデータに対するものである．そのため，表 5 の計算の後に通信処理が必要であることに注意する．

最後に図 1 (4) の消去演算では，主ループ適用範囲の各要素について，3 回の乗算および 3 回の加算を行う．このため各分割方式の計算量は，表 5 の値を 3 倍したものと同値となる．

以上の演算を変換全体に対して合計した結果を**表 6** にまとめる．表 6 では各分割方式で総計算量が異なるが，これは各反復で最も計算量の多い処理を行う PE

表 5 行列ベクトル積の計算量
Table 5 The computing complexities for matrix-vector products.

分割方式	$Calc(n,p)$
(*, Block)	$\sum_{k=1}^{p-1}\sum_{i=1}^{\frac{n}{p}} 2Row_b(i,k,n,p)Col_b(n,p) + \sum_{k=\frac{n}{p}}^{2} 2\left(\frac{n}{p} - k + 1\right) = \left(\frac{1}{p} - \frac{1}{p^3}\right)n^3 + \left(\frac{1}{p} - \frac{2}{p^2}\right)n^2 + \frac{3}{p}n$
(*, Cyclic)	$\sum_{k=1}^{\frac{n}{p}}\sum_{i=1}^{p} 2Row_c(i,k,n,p)Col_c(k,n,p) = \frac{2}{3p}n^3 + \left(\frac{1}{2} + \frac{1}{3p}\right)n^2 + \left(\frac{1}{2} - \frac{1}{6}p\right)n$
(Block, Block)	$2Row(n,p)^2\frac{n}{\sqrt{p}}(\sqrt{p}-1) + \sum_{k=1}^{\frac{n}{\sqrt{p}}-2} 2\left(\frac{n}{\sqrt{p}} - k + 1\right)^2 = 2\left(\frac{1}{p} - \frac{2}{3p\sqrt{p}}\right)n^3 + \frac{1}{p}n^2 + \frac{1}{3\sqrt{p}}n + O(1)$
(Cyclic, Cyclic)	$\sum_{k=1}^{\frac{n}{\sqrt{p}}} 2Row(k,n,p)^2 \times \sqrt{p} = \frac{2}{3p}n^3 + \frac{1}{\sqrt{p}}n^2 + n$

表 6 並列 Hessenberg reduction の計算量
Table 6 The total computing complexities for parallel Hessenberg reduction.

分割方式	$Calc(n,p)$
(*, Block)	$\left(\frac{5}{p} - \frac{5}{p^3}\right)n^3 + \left(2 + \frac{5}{p} - \frac{10}{p^2}\right)n^2 + \left(2 + \frac{15}{p}\right)n + O(1)$
(*, Cyclic)	$\frac{10}{3p}n^3 + \left(\frac{9}{2} + \frac{15}{6p}\right)n^2 + \left(\frac{9}{2} - \frac{5}{6}p\right)n + O(1)$
(Block, Block)	$10\left(\frac{1}{p} - \frac{2}{3p\sqrt{p}}\right)n^3 + \left(\frac{4}{\sqrt{p}} + \frac{3}{p}\right)n^2 + \frac{11}{3\sqrt{p}}n + O(1)$
(Cyclic, Cyclic)	$\frac{10}{3p}n^3 + \frac{7}{\sqrt{p}}n^2 + \frac{11}{3}n$

の計算量の総和だからである．

5.3 通 信 量

通信処理の実装方式により通信量が異なるので，解析においては実装方式を定める必要がある．4 章で述べたとおり，Hessenberg reduction では各 PE が局所的に所有するベクトルに対して，その要素の総和演算が必要である．この通信と演算を含む操作を，一般にベクトルリダクション演算という．また同様に，スカラ量に対して同様の操作を行うことをスカラリダクション演算と呼ぶ．本アルゴリズムの実装では，この

ベクトルリダクション演算を行うのにPE数をpとすると，$\log_2 p$回の通信と1回の放送とを行う処理により実現する．以降，この実装方式の各反復時における最大の通信量について考察し，その総和をアルゴリズム全体における通信量とする．

まず枢軸ベクトル生成列を放送するのに必要な通信回数を$BroadT(n,p)$，メッセージ転送量を$BroadS(n,p)$としたときの通信量を**表7**にまとめる．表7から，(*, Cyclic)列分割方式は(Cyclic, Cyclic)格子分割方式に比べて，通信回数は半分で済むことが分かる．しかし，(Cyclic, Cyclic)格子分割方式ではpが増せば，通信量は減少していく利点があることに注意しておく．

同様に，各分割方式のベクトルリダクション操作に必要となる，通信起動回数，メッセージ転送量および，加乗算回数を計算すると，**表8**のようになる．一般に主ループ適用範囲が反復が進むにつれて減少するアルゴリズムにおいては，サイクリック分割はすべての反復で通信が必要である．ところが，ブロック分割にすると通信対象のPEが減少して，反復の終盤では通信が不要で，1PE内での演算のみとなる．この点がブロック分割の優れている特徴といえる．表8に示す値は通信回数，メッセージ転送量ともこの点を考慮したものである．表8から，(*, Block)，(*, Cyclic)列方向分割ではPE数が増すにつれ，ベクトルリダクションに費す通信量と計算量が増すが，(Block, Block)，(Cyclic, Cyclic)格子分割方式では，通信量と計算量がともに減少することが分かる．

表7 枢軸ベクトル生成列の放送量
Table 7 Total amount of broadcasted data for pivot vector column.

分割方式	$BroadT(n,p)$	$BroadS(n,p)$
(*, Block)	$\frac{n}{p}(p-1)$	$\sum_{k=1}^{p-1}\sum_{i=1}^{n/p} Row_b(i,k,n,p)$ $= \left(\frac{1}{2}-\frac{1}{2p^2}\right)n^2 + \left(\frac{1}{2}-\frac{1}{2p}\right)n$
(*, Cyclic)	$n-2$	$\sum_{k=1}^{n-2}(n-k+1)$ $= \frac{1}{2}n^2+\frac{1}{2}n-3$
(Block, Block)	$2\left(1-\frac{1}{\sqrt{p}}\right)n$	$2Row_{bb}(n,p)^2(\sqrt{p}-1)$ $= 2\left(\frac{1}{\sqrt{p}}-\frac{1}{p}\right)n^2$
(Cyclic, Cyclic)	$2(n-2)$	$2\sum_{k=1}^{n/\sqrt{p}} Row_{cc}(k,n,p)\sqrt{p}$ $= \frac{1}{\sqrt{p}}n^2+n$

さらに加えて，(Block, Block)，(Cyclic, Cyclic)格子分割方式では，図3(a)(1)のノルム計算と図3(b)(7)のμの計算でスカラリダクション演算が$2(n-2)$回必要となる．これらの演算をまとめれば，**表9**のようになる．

5.4 性能解析

ここですべての演算に必要な計算量と通信量が分かったので，各分割方式における実行性能について解析を行う．なおここでの性能解析には，基本的には演算に関するファクタϕを単位とする具体的な計算時間の予測モデルについて述べている点に注意しておく．表6におけるn^3の係数を比較すると，計算量では

$$(\text{Cyclic, Cyclic}) = (*, \text{Cyclic})$$
$$(*, \text{Cyclic}) < (*, \text{Block}) < (\text{Block, Block})$$

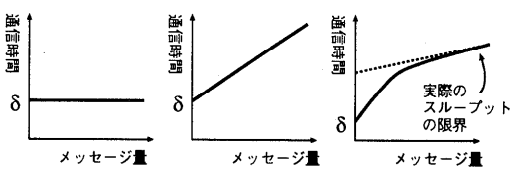
の順となる．この理論計算量の比較から(Cyclic, Cyclic)と(*, Cyclic)の両分割方式が有効といえる．さらに逐次の場合の総計算量が$10/3n^3$で，各PEの計算量が$10/(3p)n^3$であることから，これらの分割方式ではロードバランスが完全になされていることも分かる．一方，通信時間に関しては並列計算機の通信時間のモデルの違いにより変化する．次に，その点を明確とするために**図4**に示した3種類の通信性能を前提としたモデル上での議論を行うこととする．

図4に示された通信時間モデルで，我々がさきほど求めた表9に示す通信量を考察する．これまでの計算量の考察から(*, Cyclic)，(Cyclic, Cyclic)の両分割方式が有効になると予想されるので，この2つの分割方式についてのみ検討する．

零次近似型［図4(a)］：

理想的な並列計算機の通信時間を示す型である．すなわち，どんなに通信メッセージ長が長くなっても通信時間は一定である型である．我々の通信時間のモデルでは，$\tau=0$を意味している．ゆえに通信時間は通信起動回数$CommT(n,p)$および$BroadT(n,p)$のみで決まる．

このモデルでは，(*, Cyclic)列分割方式が(Cyclic, Cyclic)格子分割方式に対して必ず通信時間が少なくなる．



(a)零次近似型 (b)一次近似型 (c)複合近似型

図4 3種類の通信時間モデル
Fig. 4 The three types of communication time model.

表 8 ベクトルリダクション演算の通信量と計算量
Table 8 Total number of communication complexities and computing complexities for vector reduction operation.

(a) 通信回数

分割方式	$CommT(n,p)$	$BroadT(n,p)$
(*, Block)	$\sum_{k=1}^{\log_2 p-1}(\log_2 p-k+1)\frac{n}{2^k}$ $=(\log_2 p-1)n$	$\left(1-\frac{1}{p}\right)n$
(*, Cyclic)	$(n-2)\log_2 p$	$n-2$
(Block, Block)	$2\sum_{k=1}^{\log_2\sqrt{p}-1}(\log_2\sqrt{p}-k+1)\frac{n}{2^k}$ $=2(\log_2\sqrt{p}-1)n$	$2\left(1-\frac{1}{\sqrt{p}}\right)n$
(Cyclic, Cyclic)	$2(n-2)\log_2\sqrt{p}$	$2(n-2)$

(b) 通信量

分割方式	$CommS(n,p)$	$BroadS(n,p)$
(*, Block)	$\sum_{k=1}^{\log_2 p-1}\sum_{i=1}^{n/2^k}\left(n-\sum_{j=1}^{k-1}\frac{n}{2^j}-i+1\right)(\log_2 p-k+1)$ $=\left(\frac{1}{2}\log_2 p-\frac{1}{6}-\frac{1}{3p}\right)n^2+\left(\frac{1}{2}\log_2 p-\frac{1}{2}\right)n$	$\sum_{k=1}^{p-1}\sum_{i=1}^{n/p}\left(n-(k-1)\frac{n}{p}-i+1\right)$ $=\frac{1}{2}\left(1-\frac{1}{p^2}\right)n^2+\frac{1}{2}\left(1-\frac{1}{p}\right)n$
(*, Cyclic)	$\sum_{i=1}^{n-2}(n-i+1)\log_2 p$ $=\frac{\log_2 p}{2}(n^2+n-6)$	$\sum_{i=1}^{n-2}(n-i+1)$ $=\frac{1}{2}(n^2+n-6)$
(Block, Block)	$2\sum_{k=1}^{\log_2\sqrt{p}-1}\frac{n}{2^k}(\log_2\sqrt{p}-k+1)\frac{n}{\sqrt{p}}$ $=2\frac{\log_2\sqrt{p}-1}{\sqrt{p}}n^2$	$2Row_{bb}(n,p)^2(\sqrt{p}-1)$ $=2\frac{\sqrt{p}-1}{p}n^2$
(Cyclic, Cyclic)	$2\sum_{k=1}^{n/\sqrt{p}}Row_{cc}(k,n,p)\times\sqrt{p}\log_2\sqrt{p}$ $=\frac{1}{\sqrt{p}}\log_2\sqrt{p}\,n^2+\log_2\sqrt{p}\,n$	$2\sum_{k=1}^{n/\sqrt{p}}Row_{cc}(k,n,p)\times\sqrt{p}$ $=\frac{1}{\sqrt{p}}n^2+n$

(c) 計算量：(b) $CommS(n,p)$ と同じ

一次近似型［図 4 (b)］：

通信するメッセージ長が長くなるほど，それに比例して転送時間が長くなる型である．このモデルでは，通信パラメータ δ, τ, γ の各値により，通信時間が少なくなる分割方式が異なる．このとき，表 9 の値から δ, γ は n の一次の項，τ は n の二次の項で比較する．ここで議論を単純にするため，

- 演算時間のファクタ ϕ を各分割方式で同一

と仮定する．

この仮定のうえで，(*, Cyclic) 列分割方式が (Cyclic, Cyclic) 格子分割方式に対して必ず通信時間が少なくなる条件は

$$2(\varphi_{\gamma b}+1)\frac{1}{n} > \left(1-\frac{2}{\sqrt{p}}\right)\varphi_{bb}+\left[\frac{1}{2}\left(1-\frac{1}{\sqrt{p}}\right)\log_2 p\right]\varphi_{cb} \quad (7)$$

となる．ここで放送の通信立ち上がりに関する比を

$$\varphi_{\gamma b}\equiv\frac{\gamma(\sqrt{p})}{\delta_b(p)},\ \varphi_{bb}\equiv\frac{\tau_b(p)}{\delta_b(p)},\ \varphi_{cb}\equiv\frac{\tau_c(p)}{\delta_b(p)} \quad (8)$$

とおいた．式 (7) から，リダクション時間に関する比 $\varphi_{\gamma b}$ が大きいほど，また単位時間あたりのメッセージ転送量に関する比 φ_{bb}, φ_{cb} が小さいほど，(*, Cyclic) 列分割方式が有利となるといえる．

ここで式 (7) は演算ファクタ ϕ を各分散方式で同じ

表9 Hessenberg 形への変換処理における総通信量とベクトルリダクション演算における計算量
Table 9 The global total communication complexities for Hessenberg reduction, and total number of computing complexities for vector reduction operation.

分割方式	$BroadT(n,p)$	$BroadS(n,p)$	$ReductS(n)$	$Calc(n,p)$
(*, Block)	$2\left(1-\frac{1}{p}\right)n$	$\left(1-\frac{1}{p}\right)n^2+\left(1-\frac{1}{p}\right)n$	—	$\left(\frac{1}{2}\log_2 p-\frac{1}{6}-\frac{1}{3p}\right)n^2$ $+\left(\frac{1}{2}\log_2 p-\frac{1}{2}\right)n$
(*, Cyclic)	$2(n-2)$	n^2+n-6	—	$\frac{\log_2 p}{2}(n^2+n-6)$
(Block, Block)	$4\left(1-\frac{1}{\sqrt{p}}\right)n$	$2\frac{\sqrt{p}-1}{p}n^2$	$2(n-2)$	$2\left(\frac{\log_2\sqrt{p}}{\sqrt{p}}+\frac{\sqrt{p}-2}{p}\right)n^2$
(Cyclic, Cyclic)	$4(n-2)$	$\frac{2}{\sqrt{p}}n^2+2n$	$2(n-2)$	$\frac{1}{\sqrt{p}}(1+\log_2\sqrt{p})n^2$ $+\frac{1}{\sqrt{p}}(1+\log_2\sqrt{p})n$

$CommT(n,p)$：表8 (a) $CommT(n,p)$ と同じ
$CommS(n,p)$：表8 (b) $CommS(n,p)$ と同じ

と仮定した場合において，通信パラメータ，使用するPE数，および問題サイズにより，最適な分割方式が理論上異なるということを意味している点に注意しておく．

複合近似型［図4 (c)］：

実際の並列計算機の通信時間に近い型である．この型は零次近似型と一次近似型との複合型と考えることができる．なぜなら1回あたりのメッセージ転送量が十分小さな領域では，通信立ち上がり時間 δ を小さくする方が高速な通信が期待できるからである．この領域では一次近似型で近似ができると考えてよい．しかし逆に，転送メッセージ量が十分多く，通信立ち上がり時間 δ が無視できるときは，通信立ち上がり時間 δ が大きくなっても，スループットを上げる方が高速な通信が期待できる．この領域では一次近似型もしくは零次近似型で近似できる．

このモデルでは，メッセージが十分小さな領域では一次近似型と同じ傾向を示すが，メッセージ長が大きくなるにつれ (*, Cyclic) 列分割方式が有効になる状況になることも考えられる．

6. 性能評価

この章では，富士通AP1000+，および日立SR2201を用いて，前章の解析で得られた性能の逆転現象が観測されるか検証する．以降の議論では，通信時間モデルは一次近似型を仮定する．また，この論文で提案する (Cyclic, Cyclic) 格子分割方式にともなうアルゴリズムは，PE数 p が大きい場合に有効になるものと考えられるので，問題サイズ n を固定してPE数 p を増やす場合について性能を評価する．

6.1 AP1000+による性能

AP1000+の各PEの理論ピーク性能は50 MFlops，PE間は二次元トーラス網で結合されており，その最大転送性能は25 Mbytesである☆．

問題サイズ n を1024に固定し，PE数 p を4〜256まで変化させた場合の実行時間を**図5**に示した．図5より，AP1000+ではすべてのPEで (Cyclic, Cyclic) 格子分割方式が高速となった．

ここで我々の解析と，得られた実測結果が一致するか検討する．そのための並列計算機の通信性能は**表10**に示す値を仮定する☆☆．

表10の数値を用いて，式 (7) から導かれる (*, Cyclic) 列分割方式の方が通信時間が高速となる問題サイズを，PE数が p のとき n_p と表すと，$n_4 < 349$，$n_{16} < 87.5$，$n_{64} < 72.2$，$n_{256} < 75.4$ となる．$n=1024$ での図5の結果では，理論値どおり，すべて (Cyclic, Cyclic) 格子分割方式が高速となっている．一方，$p=4$，$n=340$ での実行結果は，(*, Cyclic) 列分割方式では2.89秒，(Cyclic, Cyclic) 格子分割方式2.95秒であり，この場合は理論値どおり速度の逆転が観測された．

6.2 SR2201による性能

SR2201の各PEの理論ピーク性能は300 MFlops，PE間は三次元クロスバ網で結合されており，その最

☆ 東京大学情報科学専攻が所有しているAP1000+のうち256 PEすべてを使用した．また，セルプログラム用のコンパイラとしてgcc version2.6.3，オプションとして -O3 -msupersparc -fcaller-saves -fomit-frame-pointer -funroll-loops を指定した．測定日は1996年6月24日から7月8日である．

☆☆ AP1000用セルライブラリの測定値を基に，最小二乗近似して得た値である．

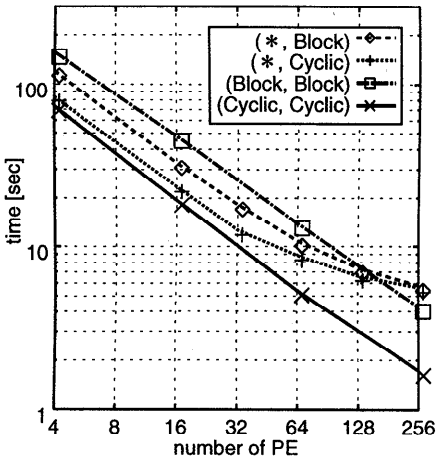


図5 AP1000+での $n = 1024$ における問題サイズを変化させた場合の実行時間

Fig. 5 Execution time for $n = 1024$ and variable number of PE's on the AP1000+.

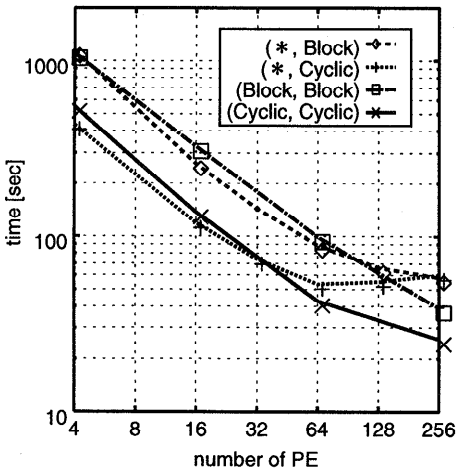


図6 SR2201での $n = 4096$ における問題サイズを変化させた場合の実行時間

Fig. 6 Execution time for $n = 4096$ and variable number of PE's on the SR2201.

表10 AP1000+の通信性能パラメータ

Table 10 The communication performance parameter of the AP1000+.

パラメータ	時間 [μ秒]
$\delta_c(4), \delta_c(16), \delta_c(64), \delta_c(256)$	14.0, 13.9, 16.4, 17.1
$\tau_c(4), \tau_c(16), \tau_c(64), \tau_c(256)$	0.40, 0.43, 0.38, 0.32
$\delta_b(4), \delta_b(16), \delta_b(64), \delta_b(256)$	16.7, 17.3, 18.6, 21.1
$\tau_b(4), \tau_b(16), \tau_b(64), \tau_b(256)$	0.76, 0.76, 0.76, 0.76
$\gamma(\sqrt{4}), \gamma(\sqrt{16}), \gamma(\sqrt{64}), \gamma(\sqrt{256})$	18, 28, 38, 50

表11 SR2201の通信性能パラメータ

Table 11 The communication performance parameter of SR2201.

パラメータ	時間 [μ秒]
$\delta_c(4), \delta_c(16), \delta_c(64), \delta_c(256)$	62.3, 55.6, 59.4, 81.4
$\tau_c(4), \tau_c(16), \tau_c(64), \tau_c(256)$	0.07, 0.08, 0.07, 0.01
$\delta_b(4), \delta_b(16), \delta_b(64), \delta_b(256)$	80.8, 159, 299, 635
$\tau_b(4), \tau_b(16), \tau_b(64), \tau_b(256)$	0.15, 0.31, 0.48, 1.08
$\gamma(\sqrt{4}), \gamma(\sqrt{16}), \gamma(\sqrt{64}), \gamma(\sqrt{256})$	70, 129, 189, 220

大転送性能は300 Mbytesである☆.

問題サイズ n を4096に固定し，PE数 p を4～256まで変化させた場合の実行時間を図**6**に示した．図6よりSR2201では，$p = 64$ 以下では (*, Cyclic) 列分割方式が高速であり，それ以上では (Cyclic, Cyclic) 格子分割方式の実行時間が高速となる結果が得られた．この結果は，我々の性能解析からも予測されうる結果である．

図6の現象を説明するため，理論値と実測値との差を考察してみる．SR2201の通信性能パラメータとして**表11**の値を仮定した☆☆.

表11の数値と，式(7)から導かれる (*, Cyclic) 列分割方式の方が通信時間が高速となる問題サイズは，$n_4 < 7713$, $n_{16} < 2073$, $n_{64} < 1726$, $n_{256} < 1184$ となる．ここで，$p = 4$, 64, 256 の場合については理論値どおりの結果が得られた．しかし，$p = 16$ の場合については理論値の約2倍の問題サイズでも性能の逆転現象が観測されなかった．

6.3 実行時間予測式の精度および通信時間の詳細解析

ここでは，我々が解析した演算時間，通信時間の見積り式の詳細な解析を行う．そこで以降の性能の見積りには，1回あたりの倍精度演算時間のファクタ ϕ を単位とする具体的な実行時間の予測モデルについて論じる．

まず日立SR2201で我々の予測が大きく外れる理由として，各分割方式での ϕ が異なるという理由が考えられる．よって，式(6)から得られる ϕ の推定値を計算する．その結果を**表12**に示す．

☆ 東京大学大型計算機センターが所有している1024 PEのSR2201のうち256 PEを使用した．また，コンパイラとして日立の最適化 FORTRAN90 V02-03，オプションとして -W0,'PVEC(PVFUNC(1), VERCHK(0), DIAG(1)), opt(o(s), fold(2), prefetch(1), rapidcall(1), ischedule(3), reroll(1), scope(1), split(2), uinline(2)) を指定した．測定日は1997年4月1日から4月7日である．

☆☆ SR2201の通信特性は複合近似型である．この型での性能評価は，先述のとおり困難である．そこで，SR2201用MPI (Message Passing Interface) の環境変数を変更し，ショートプロトコルのみで通信が行われるようにした．このことにより，一次近似型と同じ通信特性を示すようになる．このようにして測定した値を，最小二乗近似して得た．

表12 各並列計算機の ϕ の推定値
Table 12 The estimated values of ϕ on each parallel machines.

(a) AP1000+ (単位は 10^{-8} 秒)

分割方式	$p=4$	$p=16$	$p=64$	$p=256$
(*, Cyclic)	9.09	9.18	9.81	12.7
(Cyclic, Cyclic)	8.02	8.07	8.09	7.66

(b) SR2201 (単位は 10^{-9} 秒)

分割方式	$p=4$	$p=16$	$p=64$	$p=256$
(*, Cyclic)	7.44	7.27	8.78	2.91
(Cyclic, Cyclic)	9.46	8.75	8.04	2.51

表12 (b) から分かるように，SR2201 の $p=16$ では (*, Cyclic) 列分割方式の演算時間が (Cyclic, Cyclic) 格子分割方式に対して高速となっている．このことから，我々が $p=16$ で (Cyclic, Cyclic) 格子分割方式が (*, Cyclic) 列分割方式に対して高速であると予想して外れた要因の1つとして，(Cyclic, Cyclic) 格子分割方式の推定演算時間を実際よりも速く見積もっていた可能性が高い．一方で，表12 (a) から分かるように，AP1000+ではすべての p で (*, Cyclic) 列分割方式の演算時間が (Cyclic, Cyclic) 格子分割方式の演算時間に対して遅くなっており，予想よりも (Cyclic, Cyclic) 格子分割方式が有利となる場合が，実際は多いと予想される．また，SR2201 の演算時間が p が増加すると 2.5 倍ほど遅くなる理由は，p が増加すると $p=4$ の場合に対してベクトル長が短くなり，ベクトル演算の立ち上がりオーバヘッドが見えてくるからと考えている．

次に我々の提案した手法により，どれぐらいベクトルリダクション時間が減少するか解析する．いま AP1000+において $p=4$ の ϕ の値を演算時間の推定値として固定した場合に，式 (4) から得られる，実行時間（計算時間と各通信時間）の予測値と実際に測定した測定値との相対誤差を**図7**，**図8** に示す．

図7，図8 では p が増加するにつれ，予想値がはずれてくる．この原因の1つに ϕ の推定値を小さく，または大きく見積もっていることが考えられる．しかしながら，相対誤差の最大値は (*, Cyclic) 列分割方式で約 14.7%，(Cyclic, Cyclic) 格子分割方式で約 3.09%であった．これから我々の予測モデルの精度は，3.09%から 14.7%であると考えられる．図7 で $p=256$ の場合，ベクトルリダクション時間が全体の時間の 56.2%を占めている．一方，我々が提案した (Cyclic, Cyclic) 格子分割方式のアルゴリズムでは，この時間は 24.4%足らずである．この値と精度を考慮すると，我々の提案したアルゴリズムは従来のものよりベクトルリダクション時間の占める割合を 1.9 倍～2.7 倍程度削減できる可能性を示している．この削減の割合は，一般に問題サイズ n を固定して考えた場合，PE 数 p が大規模になるほど大きくなると期待される．よって我々の示したアルゴリズムは，PE 数 p が大きくなるほど従来のアルゴリズムに対して有効となるといえる．

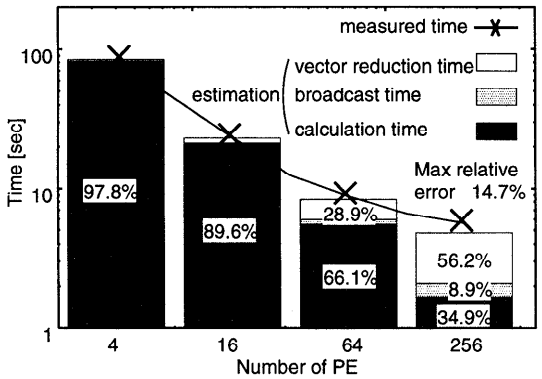


図7 AP1000+における (*, Cyclic) 列分割方式の実行時間の予測と通信時間の占める割合 ($n=1024$)
Fig. 7 The estimated execution times and the ratios of communication times in the (*, Cyclic) column-wize distributon on the AP1000+ ($n=1024$).

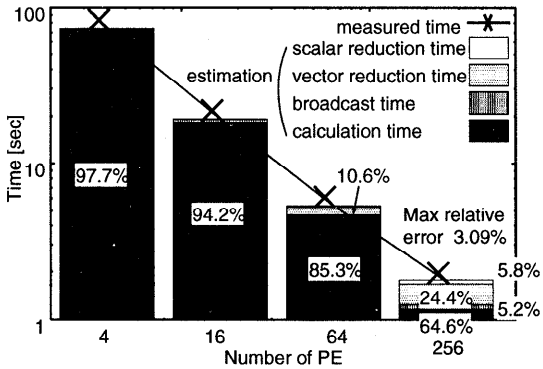


図8 AP1000+における (Cyclic, Cyclic) 格子分割方式の実行時間の予測と通信時間の占める割合 ($n=1024$)
Fig. 8 The estimated execution times and the ratios of communication times in the (Cyclic, Cyclic) grid-wize distributon on the AP1000+ ($n=1024$).

7. おわりに

Householder 法を用いた Hessenberg reduction において，PE 数が増加するに従って通信量が減少する並列アルゴリズムを提案した．また PE 数が増加するに従って通信量が増加する従来のアルゴリズムも，少なくとも並列計算機の通信性能，問題サイズ，使用する PE 数などの要因により，提案するアルゴリズムより高速となる場合が理論上ありうることが明らかとなっ

た．また実際に，富士通AP1000+と日立SR2201の実験結果でその現象が観測された．

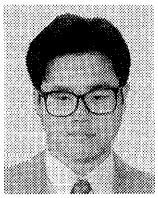
我々の示したアルゴリズムの実行時間予測式から，従来のアルゴリズムでは全体の実行時間の65%程度が通信処理になるような場合に対して，通信処理の時間を35%程度に削減できる可能性があることも明らかとなった．この傾向はプロセッサ台数を増やすとさらに顕著になることが予想されることから，我々の示したアルゴリズムは超並列処理に対して効果がある．

謝辞 日頃討論いただいた，東京大学大学院理学系研究科情報科学専攻金田研究室の諸氏に感謝いたします．有益な意見をいただいた査読者の各位に感謝いたします．また本研究の一部は，文部省科学研究費特定領域研究(A)「発見科学」（課題番号10143103）の支援により行った．

参考文献

1) Stewart, G.: A Parallel Implementation of the QR-Algorithm, *Parallel Computing*, Vol.5, pp.187–196 (1987).
2) Dongarra, J.J. and van de Geijn, R.A.: Reduction to Condensed Form for the Eigenvalue Problem on Distributed Memory Architectures, *Parallel Computing*, Vol.18, pp.973–982 (1992).
3) Kalamboukis, T.: The Parallel Algorithm for the Dense Symmetric Eigenvalue Problem on a Transputer Array, *Parallel Computing*, Vol.18, pp.207–212 (1992).
4) Chang, H., Utku, S., Sakama, M. and Rapp, D.: A Parallel Householder Tridiagonalization Stratagem Using Scattered Square Decomposition, *Parallel Computing*, Vol.6, pp.297–311 (1988).
5) Hendrickson, B.A. and Womble, D.E.: The Tours-Wrap Mapping for Dense Matrix Calculation on Massively Parallel Computers, *SIAM Sci. Comput.*, Vol.15, No.5, pp.1201–1226 (1994).
6) 片桐孝洋，金田康正：分散メモリ型並列計算機によるHouseholder法の性能評価，情報処理学会研究報告，96-HPC-62, pp.111–116 (1996).

(平成9年7月3日受付)
(平成10年9月7日採録)



片桐 孝洋（学生会員）
1973年生．1994年豊田工業高等専門学校情報工学科卒業．1996年京都大学工学部情報工学科卒業．1998年東京大学大学院理学系研究科情報科学専攻修士課程修了．現在，同大学院博士課程在学中．並列計算機による行列計算アルゴリズムの研究に従事．大規模固有値問題，並列数値計算に興味を持つ．日本応用数理学会，ACM各学生会員．

金田 康正（正会員）1949年生．1973年東北大学理学部物理第二学科卒業．1978年東京大学理学系研究科博士課程修了．理学博士．1978年名古屋大学プラズマ研究所助手．1981年東京大学大型計算機センター助教授を経て現在同教授．その間英国ケンブリッジ大学計算機研究所客員研究員，名古屋大学プラズマ研究所客員助教授，核融合科学研究所客員助教授．昭和58年度情報処理学会論文賞受賞．著書「πのはなし」（東京図書），共著「アドバンスド・コンピューティング—21世紀の科学技術基盤」（培風館），編著「Trends in Supercomputing」（World Scientific）．日本応用数理学会，プラズマ・核融合学会，ACM，SIAM各会員．研究テーマは「大規模数値計算」および「研究の研究」．