非構造格子に基づく大規模非圧縮性粘性流れ解析のための超並列計算法

樫山和男[1]・玉井　典[2]

[1]正会員　工博　中央大学教授　理工学部土木工学科(〒112-8551 東京都文京区春日 1-13-27)
[2]正会員　工修　四国電力(株)橘湾火力建設所土木課(〒779-1631 徳島県阿南市橘町小勝1番地)

本論文は，非構造格子に基づく非圧縮粘性流れ解析のための，高効率な並列計算法を提案するものである．離散化手法としては，低次の混合補間（流速双1次/圧力区分0次）要素を用いた安定化有限要素法を用いている．並列計算手法としては，メッセージパッシングを用いた領域分割法に基づく方法を採用している．本手法は，大規模な非圧縮性粘性流れ解析において，問題および機種に依存しない有効な並列計算手法であることを例題を通じて示している．

KeyWords : *High performance computing, Large-scale incompressible Navier-Stokes flow, Element-by-Element solution technique, unstructured grids, domain decomposition*

1. はじめに

近年，コンピュータの性能と数値流体力学（CFD）手法の飛躍的な進歩により，非圧縮性粘性流体の高レイノルズ流れに対して安定で高精度な解析手法が多く提案されてきている．しかし，超長大橋や超高層ビルなど構造物の大型化が進む中で，CFDにより構造物周辺の風の流れの挙動や構造物に作用する流体力を定量的に評価するには，現行の逐次計算に基づく方法では，計算機容量および計算時間の点で適用できる計算条件（解析可能なレイノルズ数や解析領域等）に限界があるといえる．このような状況の中，近年マイクロプロセッサの飛躍的な性能向上と低廉化により複数のプロセッサを備えた並列計算機が多数商品化され，目ざましい勢いで普及しつつある．並列計算機には，専用並列計算機と単体のワークステーションやパーソナルコンピュータをネットワークで複数台接続したクラスター型の仮想並列計算機がある．また，メモリーを各プロセッサー毎に持つか否かによって，分散メモリー型と共有メモリー型に大別される[1),2)]．並列計算機の性能は，従来のベクトルスーパーコンピュータの性能をはるかに凌ぎ，理工学の様々な分野でこれまで不可能とされていた計算を可能にするものとして期待されている．とりわけ，構造解析に比べて一般に離散化自由度数を多く必要とする流体解析には，並列計算機は多大なる威力を発揮するものと考えられ，近年いくつかの並列計算手法が提案されている[3)-8)]．しかし，効率よい並列計算を行うには，解法のアルゴリズムを逐次処理から並列処理に変更するだけでなく，各プロセッサの計算負荷を均等化するとともに，通信量を最小化する必要があるなど工夫すべき点も多い[1),2)]．

本論文は，大規模な非圧縮性粘性流れ解析に着目し，その解析を高速かつ省メモリーに行うことを可能にする並列計算手法を提案するものである．非圧縮性粘性流れに対する並列計算手法としては，これまで構造格子に基づく方法[3)]と非構造格子に基づく方法[4)-8)]が提案されている．構造格子に基づく方法は，非構造格子に基づく方法に比べて各プロセッサの計算負荷の均等化が容易であり，効率の高い並列計算が可能となるが，複雑形状への適用には限界がある．そこで，本論文では，離散化手法としては，複雑形状を有する領域の計算に有利な非構造格子に基づく有限要素法を用いた．有限要素法を用いた並列計算法としては，高精度な安定化有限要素法であるSUPG/PSPG法[9),10)]やGLS法[7),11)]を用いた陰的な並列計算法[5),8)]が提案されている．

本論文では，それらの手法と精度的に同等であり，かつ計算時間と計算機容量の点で有利な準陰的手法であるQ1/P0（流速双1次・圧力区分0次）要素を用いた組み合わせ手法(SUPG/PSM+MSI法)[12)]に基づく並列計算法を提案する．なお，並列計算の具体的方法としては，共有メモリー型並列計算機に比べて超並列計算に有利な分散メモリー型並列計算機を対象として，領域分割法に基づく方法[8),13),14)]を採用した．各プロセッサの計算負荷の均等化を行うために，Farhat[13),14)]が提案した領域分割法の改良を行っている．また，並列化の方法には機種の依存性の無いメッセージパッシングライブラリーの一つであるMPI(Message Passing Interface)[15),16)]を用いた．本手法の数値流体解析手法としての有効性を

検討するために，三次元立方体キャビテー内流れおよび三次元円柱周りの流れを例題として並列化効率を調べ，計算速度の高速化と計算機容量の分散による省メモリー化について検討した．また，複数の性能の異なる並列計算機を用いて，提案する手法の機種の依存性に対する検討も行った．

2. 基礎方程式と境界条件

非圧縮粘性流れの場を支配する基礎方程式は，Navier-Stokesの運動方程式(1)と連続式(2)で表される．

$$\frac{\partial u_i}{\partial t} + u_j u_{i,j} + p_{,i} - \frac{1}{Re} u_{i,jj} = 0 \quad \text{in } \Omega \tag{1}$$

$$u_{i,i} = 0 \quad \text{in } \Omega \tag{2}$$

ここで，$_{,i}$ は i 方向の偏微分，u_i は i 方向の速度成分，p は圧力，$Re(=UD/\nu$，U：代表的流速，D：代表的長さ，ν：動粘性係数)はReynolds数，Ω は境界 Γ で囲まれた解析領域を示す．また，境界条件は，

$$u_i = \hat{u}_i \quad \text{on } \Gamma_g \tag{3}$$

$$\left(-p\,\delta_{ij} + \frac{1}{Re} u_{i,j}\right) n_j = \hat{t}_i \quad \text{on } \Gamma_h \tag{4}$$

で表される．ここで，n_j は境界外向き単位法線ベクトル，δ_{ij} はKroneckerの δ，Γ_g と Γ_h はそれぞれ，DirichletとNeumannの境界条件が与えられる境界を示す．

3. 安定化有限要素法による離散化

基礎方程式に対する離散化手法として，空間方向に流速双1次/圧力区分0次(Q1/P0)要素を用いたSUPG(Streamline upwind /Petrov-Galerkin)法と圧力安定化行列（pressure stabilization matrix:PSM)[17]に基づく安定化有限要素法[12]を，時間方向に修正準陰解法(Modified Semi Implicit:MSI)[18]を適用する．修正準陰解法に従い，運動方程式に関して流速を陽的，圧力を陰的に，連続式に関しては陰的に取り扱うことにより次のような離散化方程式を得る[12]．

$$\frac{\boldsymbol{M}_c \boldsymbol{u}_i^{n+1} - \boldsymbol{M}_c \boldsymbol{u}_i^{n}}{\Delta t} + \boldsymbol{K}(\boldsymbol{u}_j^n)\,\boldsymbol{u}_i^n - \boldsymbol{M}_c \boldsymbol{M}_L^{-1} \boldsymbol{C} \boldsymbol{p}^{n+1} + \frac{1}{Re} \boldsymbol{S}\,\boldsymbol{u}_i^n = \boldsymbol{o} \tag{5}$$

$$\boldsymbol{C}^T \boldsymbol{u}_i^{n+1} + \boldsymbol{D}\,\Delta t \boldsymbol{p}^{n+1} = \boldsymbol{o} \tag{6}$$

ここで，$\boldsymbol{M}_c$，$\boldsymbol{M}_L$，$\boldsymbol{K}$，$\boldsymbol{S}$，$\boldsymbol{C}$，$\boldsymbol{D}$ はそれぞれ整合質量，集中化質量，移流，拡散，勾配，圧力安定化行列である．なお，質量行列と移流行列に関しては［Galerkin項］＋［SUPG項］という二つの行列から構成されている．得られた運動方程式と連続式に対する有限要素方程式(5)，(6)を連立することにより，次のような連立1次方程式が得られる．

$$\begin{bmatrix} \boldsymbol{M}_c & -\boldsymbol{M}_c \boldsymbol{M}_L^{-1} \boldsymbol{C} \\ \boldsymbol{C}^T & \boldsymbol{D} \end{bmatrix} \begin{Bmatrix} \boldsymbol{u}_i^{n+1} \\ \Delta t \boldsymbol{p}^{n+1} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{b}_i^n \\ \boldsymbol{o} \end{Bmatrix} \tag{7}$$

ここで，$\boldsymbol{b}_i^n$ は運動方程式の既知項をまとめたものであり

$$\boldsymbol{b}_i^n = \boldsymbol{M}_c \boldsymbol{u}_i^n - \Delta t \left(\boldsymbol{K}(\boldsymbol{u}_j^n)\,\boldsymbol{u}_i^n + \frac{1}{Re} \boldsymbol{S}\,\boldsymbol{u}_i^n \right) \tag{8}$$

となっている．式(7)において，記憶容量低減のために圧力の部分を掃き出すと，次のような圧力のPoisson方程式を導くことができる．

$$(\boldsymbol{C}^T \boldsymbol{M}_L^{-1} \boldsymbol{C} + \boldsymbol{D})\,\Delta t \boldsymbol{p}^{n+1} = -\boldsymbol{C}^T \boldsymbol{M}_c^{-1} \boldsymbol{b}_i^n \tag{9}$$

なお，圧力のPoisson方程式を解く際にはElement-by-Element共役勾配法(SCG法)[19),20),21)]を用いている．この方法は，共役勾配法によってPoisson方程式を解く際に毎ステップ行う以下の繰り返し計算

$$\boldsymbol{\omega} = (\boldsymbol{C}^T \boldsymbol{M}_L^{-1} \boldsymbol{C})\,\boldsymbol{\phi} \tag{10}$$

を全体行列を組み立てないで $\boldsymbol{C}$ の要素行列 $\boldsymbol{C}^e$ を用いて以下のように段階的に行うというものである．これにより，記憶容量の低減および計算の高速化が可能となる．

$$\boldsymbol{\theta} = \sum_e (\boldsymbol{C}^e\,\boldsymbol{\phi}) \tag{11a}$$

$$\boldsymbol{\varphi} = \boldsymbol{M}_L^{-1} \boldsymbol{\theta} \tag{11b}$$

$$\boldsymbol{\omega} = \sum_e (\boldsymbol{C}^{eT}\,\boldsymbol{\varphi}) \tag{11c}$$

ここで，要素行列 $\boldsymbol{C}^e$ は2次元計算では各方向に4つの成分を持つ行列（列ベクトル）であり，式(11a),(11b),(11c)を解くには1要素あたり8成分(3次元計算の場合は24成分)を記憶させるだけでよい．従って，問題が大規模になっても要素数に比例しただけの記憶容量で計算が行えるため，大規模計算に対して有効な方法であると言える．

以上をまとめると，非圧縮Navier-Stokes流れ解析の本計算手順は次の通りである．

1. 中間流速求解

$$\tilde{\boldsymbol{u}}_i = \boldsymbol{u}_i^n - \Delta t \boldsymbol{M}_c^{-1} \left(\boldsymbol{K}(\boldsymbol{u}_j)\,\boldsymbol{u}_i^n + \frac{1}{Re} \boldsymbol{S}\,\boldsymbol{u}_i^n \right) \tag{12}$$

2. 圧力求解

$$(\boldsymbol{C}^T \boldsymbol{M}_L^{-1} \boldsymbol{C} + \boldsymbol{D})\,\Delta t \boldsymbol{p}^{n+1} = -\boldsymbol{C}^T \tilde{\boldsymbol{u}}_i \tag{13}$$

3. 流速求解

$$\boldsymbol{u}_i^{n+1} = \tilde{\boldsymbol{u}}_i + \boldsymbol{M}_L^{-1} \boldsymbol{C}\,\Delta t \boldsymbol{p}^{n+1} \tag{14}$$

なお，式(12)の解法には3-pass algorithmを用いている[12)]．離散化の詳細については文献[12)]に詳しい．

4. 並列計算法

(1) 並列計算法の概要

3．で示した離散化手法に対する並列計算の手順は以下の通りである(**Fig.1**参照).

1) 与えられた有限要素メッシュに対して領域分割法を適用し，部分領域データと部分領域境界上の節点データを作成する.
2) 各プロセッサは担当する部分領域に関するデータを入力する.
3) 各プロセッサは担当する部分領域について，中間流速求解式(12)を作成する.
4) 隣接プロセッサ通信を行い，中間流速を求める.
5) 各プロセッサは担当する部分領域について，圧力求解式(13)を作成する.
6) 隣接プロセッサ通信と全プロセッサ通信を行い，圧力を求める.
7) 各プロセッサは担当する部分領域について，流速求解式(14)を作成する.
8) 隣接プロセッサ通信を行い，流速を求める.
9) 時間ステップが終了するまで 3)~8) を繰り返す.

一般に，効率よい並列計算を実現するためのには，次の三つの点を実現する必要がある.

a) 各プロセッサの計算負荷の均等化
b) プロセッサ間の通信量の最小化
c) 解析プログラムの並列処理部分の拡大

これらを実現するために，本論文ではa)に対しては，各プロセッサに割り当てる要素数の均等化，b)に対しては，通信する節点数の最小化，c)に対しては，プログラム全体の並列化を行っている．これらのうち，a)とb)については並列計算の前処理として行う領域分割により実現する．領域分割法については，本章の**(3)**において，並列化手法については**(4),(5)**において詳しく述べる．なお，並列化に伴うデータ通信には機種の依存性のないMPI (Message Passing Interface) を用いている.

(2) 並列計算機

本論文で主に使用した並列計算機は，分散メモリー型並列計算機である日立社のSR2201である．この計算機は，1024台のプロセッサが**Fig.2**に示すように3次元クロスバーネットワーク(300MB/sec)により接続されている．SR2201のプロセッサの速度性能 は300MFlops，メモリーは256MBあり，1024台のプロセッサ使用時においては，300GFlopsのシステム性能を有し，262GBまでの計算が可能である．また，3次元クロスバーネットワークはデータ転送時にバッファを介さないため高速なデータ転送が行える特徴がある[22].

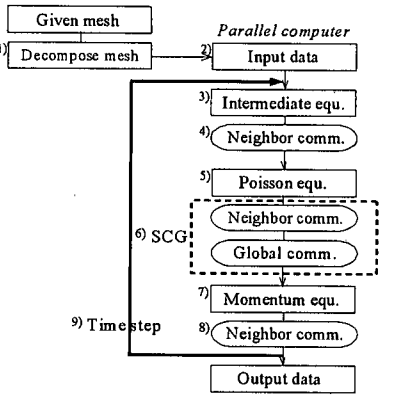


Fig.1 並列計算の流れ

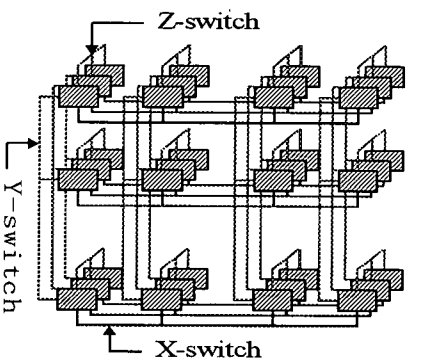


Fig.2 SR2201のネットワーク構成図

(3) 領域分割法

本論文では，領域分割法に基づく並列計算法を提案する．この方法は，与えられた有限要素メッシュをいくつかの部分領域に分割し，部分領域ごとにプロセッサを割り当てて並列計算を行う方法である．効率の良い並列処理を実現するためには領域を分割するにあたり，(1)で述べたようにa)各部分領域における計算負荷の均等化，b)部分領域間の通信量の最小化，の二点の実現が要求される.

これら二点を同時に実現させる領域分割法は，大きく，隣接要素の関係を記憶しておき近隣の要素を必要数だけ探索するGreedy法に基づく方法と数学的手法を用いて計算領域を次々に二分していく二分法（Bisection法）に基づく方法に大別されるが[23]，ここではアルゴリズムの単純なGeedy法に基づくFarhatにより提案された領域分割法を用いた[13),14)] (**Fig.3**, **Fig.4**参照)．この方法は，a)に対しては部分領域に割り当てる要素数の均等化，b)に対しては領域境界上の節点数の最小化を行っている．有限要素法は要素ごとに処理を進め

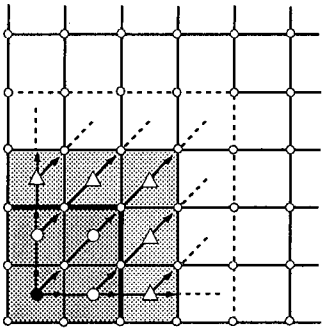


Fig.3 Greedyアルゴリズムのための要素モデル

ていく手法であるので，部分領域ごとの要素数が均等になれば計算負荷の均等化も実現する．また，領域境界上では流速および圧力の連続条件を満たすために通信が必要となるが，領域境界上での節点数を最小化するように領域分割することで，通信量の最小化も実現する．

しかし，Farhatが提案したGreedyアルゴリズム(**Fig.4**) をそのまま採用すると，条件によってはプロセッサ(部分領域)間の負荷均等が崩れる場合がある．すなわち，この手法により領域分割を行うと各プロセッサにはELE/PE (ELE:要素数，PE:プロセッサ数)個の要素が割り当てられるが，ELE/PEが割り切れない場合に，最後の一つの領域にELE/PE+割り切れないで余った要素個の要素が割り当てられる．この計算負荷の不均等により同期待ちを発生させ，これがプロセッサを多数使う計算（超並列計算）や粒度が細かい計算の場合には並列化効率を下げる原因となる．そこで本論文では，まずELE/PEの商を全プロセッサに割り当て，割り切れないで余った要素を各プロセッサに1つずつ振り分けていき，プロセッサ間の要素数の差が最大でも1になるように改良している．この改良により，非構造格子に基づく任意のメッシュに対してプロセッサ間の負荷の均等化がさらに実現される．

(1) まず，各要素についてそれぞれの要素に隣接している要素を記憶しておく．ここで言う隣接とは，2次元要素では「辺」か「点」，3次元要素では「面」,「辺」,「点」のいずれかで接していることを指す．

(2) 各部分領域(プロセッサ)に割り当てる要素数を求める．各部分領域には次式で求まる均等数の要素が割り当てられる．

$$ELEp = ELE \ / \ PE$$

ここで，ELEpは部分領域内の要素数，ELEは領域全体の要素数，PEは並列実行時に使用するプロセッサ数である．

(3) ある要素に着目し，(1)の結果を利用して，その要素に隣接している要素を認識する．**Fig.4**では，●要素に着目し，その周辺の○要素を認識する．

(4) (3)が終了した段階で認識された要素数がELEpに達していなければ，(3)で新たに認識された要素に隣接している要素を順次探索し，認識させる．**Fig.4**では，○要素に隣り合う△要素を順次探索していく．

(5) 認識された要素がELEpに達するまで(4)を繰り返す．この結果，認識された全ての要素が同一の部分領域に属する要素となる．**Fig.4**では，●要素を含めて色のついた要素が，一つの部分領域を形成する．

(6) (5)で確定した部分領域に接する要素のうち，要素番号が最小のものを出発要素(●)として(3)に戻る．

(7) 最終的にPE個の部分領域が完成するまで，(3)~(6)を繰り返す．

Fig.4 Greedyアルゴリズム

(4) 運動方程式の並列計算法

有限要素方程式の並列計算法として，まず流速求解式(14)の並列化について述べる．いま，式(14)を以下のように表す[26),27)]．

$$\boldsymbol{M}_L \boldsymbol{X} = \boldsymbol{F} \qquad (15)$$

ここで，$\boldsymbol{M}_L$は集中化質量行列，$\boldsymbol{X}$は未知ベクトル，$\boldsymbol{F}$は既知ベクトルである．**Fig.5**に示す要素モデルにおいて，節点A（節点が部分領域内にある場合）と節点B（節点が部分領域境界上にある場合）の未知量の求解方法を述べる．なお，**Fig.5**において要素(1)~(4)はプロセッサ1に含まれ，要素(5),(6)はプロセッサ2に含まれているものとする．

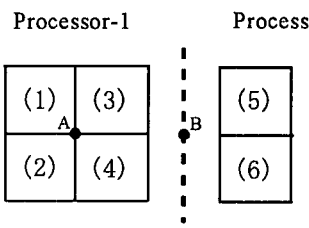


Fig.5 陽的並列計算のためのメッシュモデル

a) 節点Aの未知量の求解法

1) 節点Aを共有している各要素において，$M_{L,A(n)}$, $F_{A(n)}$を計算する．ここで，nは節点Aを共有する要素を示す．

2) $M_{L,A(n)}$, $F_{A(n)}$を足し合わせることにより，節点量$M_{L,A}$, F_Aが求まる．

$$M_{L,A} = M_{L,A(1)} + M_{L,A(2)} + M_{L,A(3)} + M_{L,A(4)}$$

$$F_A = F_{A(1)} + F_{A(2)} + F_{A(3)} + F_{A(4)}$$

3) 節点Aにおける未知ベクトルX_Aは次式から得られる．

$$X_A = F_A / M_{L,A}$$

このように，節点が領域内部に存在する場合には，プロセッサ間の通信が発生しないことになる．

b) 節点Bの未知量の求解法

1) 節点Bを共有する各プロセッサでは，次式のように各プロセッサ内での節点量$M_{L,B(m)}$, $F_{B(m)}$が求まる．ここで，mは節点Bを共有する要素を示す．

プロセッサ1：

$$M_{L,B1} = M_{L,B(3)} + M_{L,B(4)}$$

$$F_{B1} = F_{B(3)} + F_{B(4)}$$

プロセッサ2：

$$M_{L,B2} = M_{L,B(5)} + M_{L,B(6)}$$

$$F_{B2} = F_{B(5)} + F_{B(6)}$$

2) プロセッサ間で境界節点データの送受信をした後，次式により節点Bにおける節点量$M_{L,B}$, F_Bを得る．

$$M_{L,B} = M_{L,B1} + M_{L,B2} \qquad F_B = F_{B1} + F_{B2}$$

3) 節点Bにおける未知ベクトルX_Bは次式から得られる．

$$X_B = F_B / M_{L,B}$$

以上のように，節点が部分領域境界上にある場合には，その節点に関係する要素の重ね合わせを行うため，隣接するプロセッサ間でのデータ送受信を行う必要がある．このデータ送受信は，各時間ステップ毎に行われるが，集中化質量行列M_Lについては不変であるため送受信は1度行うだけでよい．

なお，中間流速求解式(12)の並列化についても上記と同様に行うことができる．

(5) 圧力Poisson方程式の並列計算法

有限要素方程式(12)～(14)のうち，解を得るのに最も時間を要するのは圧力Poisson方程式(13)を解く部分であり，時間ステップ区間に占めるその割合は三次元計算の場合約60% である．従って，この圧力の求解部分をいかに効率よく並列化するかが重要となる．本論文では，解法として並列処理向きであるElement-by-Element SCG法を用いる．**Fig.6**にElement-by-Element SCG法の計算アルゴリズムを示す．ただし，圧力方程式を$Ax = b$とし，ϕ, ψは作業ベクトル，rは残差ベクトルを表している．本論文では圧力を要素内一定で補間するP0要素を用いているため，**Fig.6**中の全ての計算を要素に関して並列に処理でき，効率良い並列計算が行える．

(i) 初期設定

$$r_0 = b - Ax_0 = b - \underbrace{\sum_e A^{(e)} x_0}_{①} \tag{16}$$

$$\phi_0 = r_0 \tag{17}$$

(ii) 解が収束するまで式(18)～(23)をk回繰り返す．

$$\psi_k = A\phi_k = \underbrace{\sum_e A^{(e)} \phi_k}_{①} \tag{18}$$

$$\alpha_k = \underbrace{(r_k, r_k)}_{②} / \underbrace{(\phi_k, \psi_k)}_{②} \tag{19}$$

$$x_{k+1} = x_k + \alpha_k \phi_k \tag{20}$$

$$r_{k+1} = r_k - \alpha_k \psi_k \tag{21}$$

$$\beta_k = \underbrace{(r_{k+1}, r_{k+1})}_{②} / \underbrace{(r_k, r_k)}_{②} \tag{22}$$

$$\phi_{k+1} = r_{k+1} + \beta_k \phi_k \tag{23}$$

Fig.6 Element-by-Element SCG法の計算アルゴリズム

Fig.6において，①，②はアルゴリズムの並列化により通信を伴う計算であることを示している．①は隣接するプロセッサ間の通信であり，②は全プロセッサ間の通信である．Element-by-Element計算部分(①)では隣接するプロセッサ間との一対一通信が必要となる．この通信は，部分領域境界上の節点に対する要素の重ね合わせを完成させるために必要となるもので，流速求解（陽的解法）に用いた通信と同じものである．また，ベクトルの内積計算部分(②)ではスカラー量の総和を求めるために，全プロセッサ間の通信が必要となる．

次に，上記の隣接プロセッサ通信（①）と全プロセッサ間通信（②）について**Fig.7**を用いて述べる．

a) 隣接プロセッサ間通信

この通信は部分領域境界上の節点に対して要素の重ね合わせを完成させるために行う通信であり，式(18)を式(11a)-(11c)の3段階で処理する際に表れる．一例として，式(11a)の並列計算について示す．

$$\theta = \sum_e (C^e \phi) \tag{24}$$

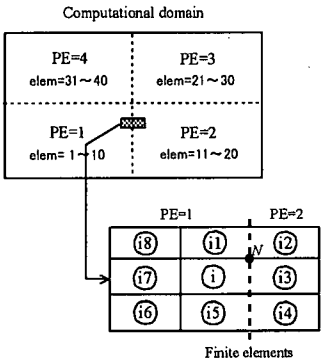


Fig.7 領域分割モデルとメッシュモデル

この計算は，以下の手順で行なう．

1) **Fig.7**(下)の節点Nに着目すると，各プロセッサにおいて次の計算が行われる．

$$\text{プロセッサ1:}\quad \theta_{N1} = C_{N,i}\,\phi_i + C_{N,i1}\,\phi_{i1} \tag{25}$$

$$\text{プロセッサ2:}\quad \theta_{N2} = C_{N,i2}\,\phi_{i2} + C_{N,i3}\,\phi_{i3} \tag{26}$$

2) 隣接プロセッサ間通信により，両プロセッサ間で互いのデータを交換し，節点Nに関する重ね合わせを行う．

$$\theta_N = \theta_{N1} + \theta_{N2} \tag{27}$$

この通信は流速求解にあらわれる通信と同様である．

b) 全プロセッサ間通信

この通信は，式(19),(22)で表されるベクトルの内積計算を行うために必要となる．

1) **Fig.7**(上)において各プロセッサで式(19)の分子(ここでは$\bar{\alpha}$とする)を計算すると，以下のような部分和が求まる．

プロセッサ1：
$$\bar{\alpha}_1 = \sum_{e=1}^{10} r_e^2$$

プロセッサ2：
$$\bar{\alpha}_2 = \sum_{e=11}^{20} r_e^2$$

プロセッサ3：
$$\bar{\alpha}_3 = \sum_{e=21}^{30} r_e^2$$

プロセッサ4：
$$\bar{\alpha}_4 = \sum_{e=31}^{40} r_e^2$$

2) 各プロセッサが有するデータの総和を，全プロセッサ間通信を用いて求める．

$$\bar{\alpha} = \bar{\alpha}_1 + \bar{\alpha}_2 + \bar{\alpha}_3 + \bar{\alpha}_4 \equiv \sum_{e=1}^{40} r_e^2 \tag{28}$$

この通信を行って求まる式(19)のα_kは未知ベクトルx_kを修正するための補正量である．

(6) PSM法と周期境界条件の並列処理

a) PSM法の並列処理

本論文では，流速・圧力の補間関数の組み合わせが下限上限条件を満たさないことにより生じる数値的不安定性を回避するために，圧力安定化行列（PSM）を用いており，この行列作成において，隣接プロセッサ間通信が必要となる．**Fig.7**において要素ⓘについて求めるべき安定化行列D_{ij}は式(29)で与えられる[17),12)]．

$$\sum_j D_{ij}p_j = a_i a_{i1}(p_i - p_{i1}) + a_i a_{i3}(p_i - p_{i3}) + a_i a_{i5}(p_i - p_{i5}) + a_i a_{i7}(p_i - p_{i7}) \tag{29}$$

ただし，a_kは要素kごとに決められる安定化パラメータで次のように与えられる．

$$a_k = \alpha\sqrt{(C^T M_L^{-1} C)_{kk}} \tag{30}$$

ここで，M_Lは集中化質量行列，Cは勾配行列，αは安定化の度合いを決定する無次元パラメータである．なお，本論文の3次元計算では文献[12)]に従いαの値を0.25とした．

要素ⓘの安定化行列を求めるために，この場合，プロセッサ1で

$$\sum_j D_{ij}p_j = a_i a_{i1}(p_i - p_{i1}) + a_i a_{i5}(p_i - p_{i5}) + a_i a_{i7}(p_i - p_{i7}) \tag{31}$$

の計算を，プロセッサ2で以下の計算を行う．

$$D_{i,i3}p_{i3} = a_i a_{i3}(p_i - p_{i3}) \tag{32}$$

そして，式(31)と式(32)を隣接プロセッサ間通信により足し合わせることで，式(29)で与えられる安定化行列の計算を行う．

b) 周期境界条件の並列処理

ここでは，周期境界条件が課せられる場合の並列処理法について**Fig.8**を例に述べる．**Fig.8**は，数値解析例で示す上下方向に周期境界条件が課せられた三次元の円柱周りの解析におけるある鉛直断面の領域分割例である．このような場合，例えば点Aと点Bのように上層面と下層面で対応する節点どうしで通信が必要となる．点Aと点Bのように，周期境界上の節点が隣接し合う領域境界上にない場合には問題ないが，点Dのように領域境界上の節点については配慮が必要となる．たとえば，点Dに関する計算をプロセッサ1，2の2台が行うために，点Cの情報を点Dに送信する場合には，プロセッサ2はプロセッサ1，2の二つのプロセッサにデータを送信する必要がある．また，点Dの情報を点Cに送信する場合には，プロセッサ1と2が 点Dの節点値を送信するので点Cには二倍の 点D情報が送られるので，どちらか一方のみのプロセッサが送信を行うようにする必要がある．本論文ではこのような重複を回避するために，**Table 1**（表中の数字はプロセッサ番号）に示す通信のための対応表を前処理の領域分割の段階で作成して，それを参照して通信を行っている．

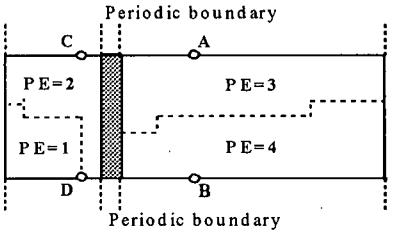


Fig.8 周期境界条件のための領域分割例

Table 1 周期境界に関する通信の対応付け

(a) 上層→下層

	1	2	3	4
A			send	
B				recv
C		send		
D	recv	recv		

(b) 下層→上層

	1	2	3	4
A			recv	
B				send
C		recv		
D	send	—		

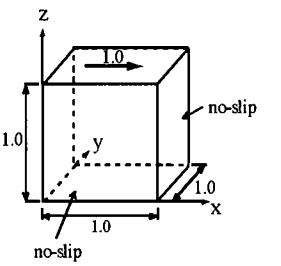
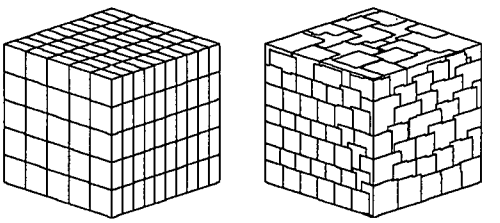


Fig.9 Cavity流れの境界条件と有限要素メッシュ



(a) スライス分割　　(b) Greedy分割

Fig.10 領域分割図 (250PE)

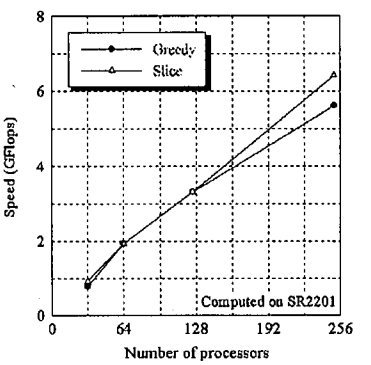


Fig.11 時間ステップ区間における速度性能

5. 数値解析例

本章では，本論文で提案するQ1/P0安定化有限要素法による並列計算手法の有効性について検討する．また，領域分割の差異が並列計算効率に与える影響や，機種の依存性についても検討する．数値解析例としては，三次元立方体Cavity内流れ解析と三次元円柱周りの高Reynolds流れ解析の二つを取り上げる．

(1) 立方体Cavity内流れ

ここでは，本手法を立方体Cavity内流れ解析に適用し，並列化の有効性と領域分割が並列化効率に及ぼす影響について検討を行った．

Cavity内流れ解析に対する並列化評価を行うため，**Fig.9**-(a)に示す領域の各辺を100等分割した有限要素メッシュを用いた．節点総数は1,030,301，要素総数は1,000,000であり，自由度数は4,090,903である．領域分割データとしては，**Fig.10**に示すGreedy分割によるものと，スライス分割によるものとの2種類を用いた．計算条件はΔt=0.01，Re=10^3とした．使用した計算機は日立SR2201であり，プロセッサ(PE)数は32，64，125，250と変化させた．

Fig.11に時間ステップ区間における実効計算速度の計測結果を示した．この図より，Greedy分割とスライス分割を用いたいずれの計算においても，演算速度がほぼプロセッサ数に比例して線形的に向上しており，高い並列化効率が得られていることがわかる．また，64PEの場合には，両者とも32PEの場合より2倍以上の計算速度が得られているが，これについてはキャッシュの効果があらわれたものといえる．また，250PEにおいてスライス分割を用いた並列計算の方がGreedy分割を用いた並列計算のそれを上回る速度性能が得られていることがわかる．この原因は，隣接プロセッサ数の差異にあり，PSM法の隣接プロセッサー間通信を行う場合，領域分割法による分割では最大26領域となったのに対して，単純なスライス分割では最大6領域であり，隣接

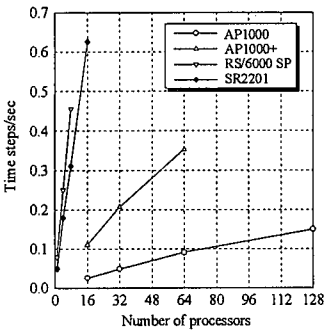


Fig.12 各種並列計算機における速度性能

PE数の少ないスライス分割の方が通信時間が少ないためと考えられる．また，250PEでこのような差が明確に表れたのは，プロセッサー数が多くなると，粒度が細かくなり計算時間に比べて通信時間が無視し得ないものとなることによる．従って，この例のように適用可能な問題は限られるが，構造格子で要素分割が可能でかつ，スライス分割により負荷均等が実現可能な場合には，スライス分割による領域分割が有利であることが分かる．

(2) 三次元円柱周り流れ解析

a) 異機種並列計算機による並列化性能の比較

本並列計算手法を，性能の異なる四種類の分散メモリー型並列計算機（日立SR2201，IBM RS6000/SP[24]，富士通AP1000およびAP1000+[25]）に適用して，機種の依存性ついて検討した．例題としては，基本的でかつ構造格子の適用の困難な円柱周りの流れ解析を取り上げた．

計算領域としては**Fig.13**に示すものを用いている．ただし，有限要素メッシュは，計算機性能に差異があるため最も性能の低い計算機の能力を考慮して比較的小規模な，節点総数 1,254×31=38,874，要素総数 1,200×30=36,000のメッシュ（節点数 1,254，要素数 1,200の2次元メッシュを高さ方向$3D$(Dは円柱直径)に30層等間隔($\Delta z = 0.1$)に積み重ねた）を用いた．また，計算条件としては，時間増分量Δt=0.01，Reynolds数 Re=1,000 とした．

Fig.12に計算の実効速度の測定結果を示す．図より，いずれの並列計算機においてもPE数の増加に伴って計算速度がほぼ直線的に向上しており，本手法は，計算機の種類に依存せず高い並列化効率が得られることが分かる．

b) 高Reynolds数流れの大規模計算

つぎに，大規模問題への適用例として**Fig.13,14**に示す微細メッシュを用いて，3次元円柱周り流れの高Reynolds数域の流れ解析を行い，計算速度と記憶容量の両面から本手法の大規模問題への有効性について検討する．また，幅広いReynolds数に対してシミュレーションを行い，特性値(抗力係数，Strouhal数)をCantwellの実験値[28]と比較することで，並列計算結果の妥当性についても示す．なお，この例題は文献[12]において著者らがすでに解析した例題であるが，ここではその文献よりも細かいメッシュを用いて解析することでメッシュによる計算結果の差異についても比較検討する．文献[12]で用いたメッシュの水平方向の最小メッシュ幅は，約 $6.0\times10^{-3}D$ であったのに対して，本計算では，その約1/6である $1.0\times10^{-3}D$ のメッシュを用いた(**Fig.14**)．なお，このメッシュは円柱周方向に160分割，径方向に60分割されており，Re=10^5の境界層(層厚 $3.2\times10^{-3}D$)が2分割されている．3次元メッシュには，この2次元メッシュを高さ方向 $3D$ に60層等間隔(Δz=0.05)に積み重ねたものを用いた．節点総数は 11,400×61=695,400，要素総数は 11,200×60=672,000 であり，自由度数は2,758,000である．

計算条件としては，Re=10^3～10^6の間を 0.5乗おきに計算し，Re=10^3～10^5で 時間増分量Δt=0.001，Re=$10^{5.5}$と10^6でΔt=0.0005とした．初期条件は各Reynolds数においてt=150の2次元解析結果を高さ方向に積み重ねたものを用い，3次元性を早く引き起こすため $150\leq t\leq151$ において撹乱を与えた[12]．なお，並列計算機としては，日立SR2201を使用した．**Fig.15**には，領域分割の一例として64PEで並列計算を行う場合の領域分割図を示した．

Fig.16にはReynolds数と抗力係数(C_D)の関係をCantwellの実験値[28]と比較したものを，**Fig.17**にはReynolds数とStrouhal数(S_t)の関係を同様に比較したものを示した．**Fig.16**中のLargeは本計算結果(約70万要素)であり，Smallは著者らが同一条件で行った計算結果(約25万要素)[12]である．図中のLargeに着目すると，実験値との比較においてRe=10^3～$10^{4.5}$までは良い一致を示している．また，Re=10^4～10^5において，実験結果と同様に抗力の値が大きくなる特徴が捉えられている．Re=10^5～10^6においては定量的には一致していないものの，Re=10^5～10^6で低下するDrag crisisの定性的な特徴は捉えられている．**Fig.17**はReynolds数とStrouhal数の関係を示したものであるが，Re=10^5付近まではLargeの計算結果はSmallに比べて実験値とよい一致を示している．以上，抗力係数およびStrouhal数ともに，微細メッシュを用いたLargeの計算結果はSmallに比べて実験結果と良い一致を示しており，これは微

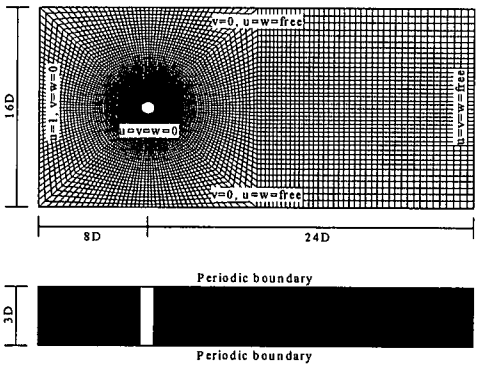


Fig.13 境界条件と有限要素メッシュ

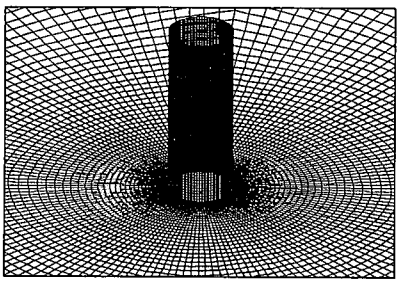


Fig.14 円柱近傍の有限要素メッシュ

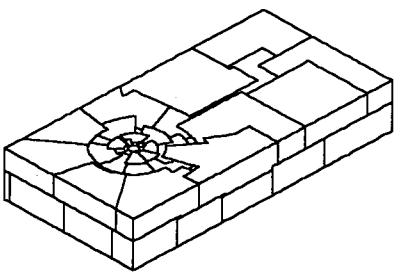


Fig.15 領域分割図 (64PE)

細メッシュを用いたことにより，発生する渦の挙動や三次元性をより正確に捉えることができたためと考えられる．Re=$10^{5.5}$以上の超臨界域において特性値を定量的に捉えるためには，より微細な有限要素メッシュの適用，乱流モデルの導入等が必要であると言える．

Fig.18と**Fig.19**はRe=10^5，Δt=0.001の計算（倍精度）において，計算時間と記憶容量負荷について調べた結果である．**Fig.18**は，時間ステップ区間全体(Time step)と，その区間内の圧力求解(SCG法; Pressure)，中間流速求解(Intermediate)部分での平均の演算速度を測定した結果である．この図から，時間ステップ区間において演算速度がほぼ直線的に向上しており，計算時

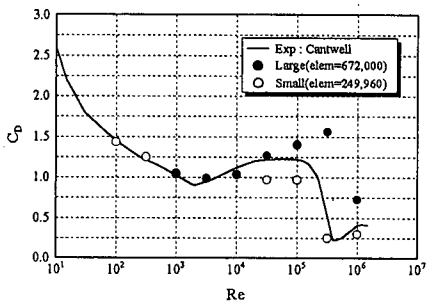


Fig.16 Reynolds数と抗力係数の関係

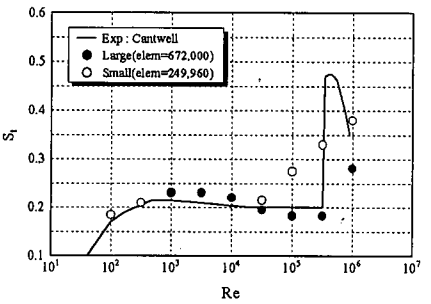


Fig.17 Reynolds数とStrouhal数の関係

間が大幅に短縮されていることが分かる．128PEでは約6.5倍(対16PE)の速度向上率を得られ，その演算速度は約3.0GFlopsであった．このように時間ステップ区間において演算速度を大きく向上できた最大の要因は，計算時間の約60%を占める圧力求解部分の演算速度が128PEで約6.0倍(対16PE)の高い向上率を得たためである．また，計算時間の約15%を占める中間流速の求解においてほぼ理想的な演算速度の伸び(区間効率 約96%)を得ており，この部分が全体の速度向上に寄与したことも一因である．なお，128プロセッサを利用してRe=10^5の計算を行った場合，1ステップあたりの計算時間は約0.79秒であった．

Fig.19は並列計算を行う上で1プロセッサ当たりが必要とする記憶容量を示したものである．図中，Serial partは通信を除く計算に必要とする記憶容量を，Comm partは並列化に伴う通信を行うための記憶容量を，Totalは両者の和をそれぞれ表している．本計算を単体で逐次処理すると約500MBの記憶容量を必要とするが，**Fig.19**から，PE数の増加に伴って計算に必要な記憶容量が大きく分散され，128PEでは50MB程度のメモリーを確保できれば計算が行えることがわかる．この結果から，本手法は省メモリーな並列計算を実現しており，大規模問題に対して記憶容量の点でも有効であることがわかる．

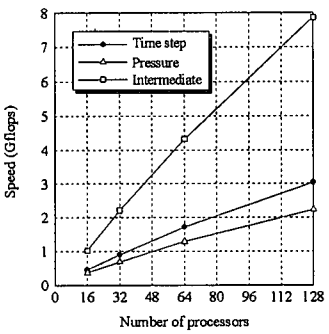


Fig.18 速度パフォーマンス

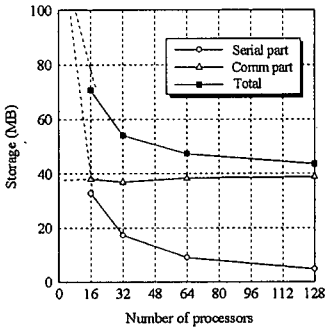


Fig.19 記憶容量負荷

6. おわりに

本論文では，Q1/P0（流速双1次/圧力区分0次）安定化有限要素法を用いた，非構造格子に基づく非圧縮性粘性流れ解析に対する並列計算法を提案した．本手法の有効性を検討するために，本手法を三次元立方体Cavity内流れの解析と三次元円柱周りの流れの解析に適用し，計算速度の向上と記憶容量の分散について検討した．また，領域分割の差異が並列計算効率に与える影響や，機種の依存性についても検討した．その結果得られた結論を以下に示す．

(1) 本手法をいくつかの問題に適用した結果，問題に依存せず高速かつ省メモリーな計算がおこなえることが明らかとなった．

(2) 高い並列化効率が得られた要因としては，領域分割により計算負荷の均等化と通信の最小化をほぼ実現できたこと，結果計算時間の約60%を占める圧力方程式の解法に，並列処理向きのElement-by-element処理に基づくSCG法を用いたことと，また，圧力の離散化にP0要素を用いたことにより，要素毎に独立して並列化が行われることがあげられる．

(3) 本手法を性能の異なる複数の分散メモリー型計算機上で計算した結果，機種の依存性がないことが明らかとなった．

(4) 本手法は領域分割に基づく手法であるため，計算効率を高めるためには，負荷の均等化と通信量の最小化を図ることが重要であり，この条件を満足しかつ構造格子で分割可能な場合には，スライス分割が有利であることを示した．

以上により，本手法は大規模問題に対して，その有効性がさらに発揮されるものと考える．今後の課題としては，より大規模でかつ極めて複雑な領域に対して有効な領域分割法の開発，前処理および後処理の並列化手法の構築があげられる．

謝辞： 本研究を進めるにあたり，元中央大学学生の斉藤克矢氏（現北海道開発コンサルタント），猪股渉氏（現東京ガス（株）），増田巧見氏（現（株）アイ・エヌ・エー）に協力を得た．また，並列計算機AP1000およびAP1000+の使用を許可された富士通並列処理研究センターに感謝いたします．なお，本研究の一部は文部省科学研究費・奨励研究（A）の補助を受けて行われたことを付記する．

参考文献

1) Hord,R.M.: Understanding Parallel Supercomputing, 356p., IEEE Press, 1999.
2) Kumar,V., Grama, A., Gupta, A. and Karypis, G.: Introduction to Parallel Computing, Design and Analysis of Algorithms, 597p., The Benjamin/Cummings Publishing Company, Inc., 1994.
3) Kawamura,H.: Direct numerical simulation of turbulence by parallel computation, *Reprints of 10th Int. Conf. On Parallel CFD*, pp.19-21, 1998.
4) Tezduyar,T.E., Aliabadi,S., Behr,M., Johnson,A., Kalro,V., and Litke,M.: Flow simulation and high performance computaing, *Comp. Mech. 18*, pp.397–412, 1996.
5) Johnson,A.A. and Tezduyar,T.E.: Parallel computation of incompressible flows with complex geometries, *Int. J. Numer. Methods Fluids*, Vol.24, pp.1321-1340, 1997.
6) Johan,Z., Mathur,K.K., Johnsson,S.L. and Hughes, T.J.R.: A case study in parallel computation: viscous flow around an onera M6 wing, *Int. J.Numer. Meth. Fluids*, Vol.21, pp.877-884, 1995.
7) Hughes, T.J.R., Franca,L.P. and Hulbert, G.M.: A new finite element formulation for CFD:VIII. The Galerkin-least-squres method for advection-diffusive equations, *Comput. Methods Appl. Mech. Engng.*, 79, pp.173-189, 1989.
8) Johan,Z., Mathur,K.K., Johnsson,S.L. and Hughes, T.J.R. : An efficient communication strategy for finite element methods on the Connection Machine CM-5 system, *Comp. Meth. Appl. Mech. Engrg.*, 113, pp.363–387, 1994.
9) Brooks, A.N. and Hughes, T.J.R.: Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations, *Comput. Methods Appl. Mech. Engng.*, 87, pp.364-384, 1991.

10) Tezduyar, T.E., Mittal, S. and Shih, R.: Time accurate incompressible flow computations with quaadrilateral velocity-pressure element, *Comput. Methods Appl. Mech. Engng.*, 79, pp.71-86, 1990.
11) Franca, L.P. and Frey, S.L.: Stabilized finite element methods: II. The incompressible Navier-Stokes equations, Comput. Methods Appl. Mech. Engng., 99, pp.209-233, 1992.
12) 樫山和男, 猪股渉：混合補間要素を用いた非圧縮粘性流れ解析のための高精度安定化有限要素法, 土木学会論文集, No.591/I-43, pp.125-137, 1998.
13) Farhat,C.: A simple and efficient automatic FEM domain decomposer, *Computers & Structures*, Vol.28, No.5, pp.579–602, 1988.
14) Farhat,C. and Lesoinne,M.: Automatic partitioning of unstructured meshes for the parallel solution of problems in computational mechanics, *Int. J. Num. Meth. Engrg.*, Vol.36, pp.745–764, 1993.
15) Gropp, W.: Using MPI, Portable Parallel Programming with the Message-Passing Interface, 307p.The MIT Press, 1994.
16) Snir, M., Otto, S.T., Huss-Lederman,S., Walker,D.W. and Dongarra,J.: MPI, The Complete Reference, 336p., The MIT Press, 1996.
17) 水上昭：Q_1-P_0要素によるFEM流れ解析のための安定化行列, 第8回数値流体力学シンポジウム講演論文集, pp.647–650, 1994.
18) Gresho,P.M., Chan,S.T., Lee,R.L. and Upson,U.D.: On the theory of semi-implicit projection methods for viscous incompressible flow and its implementation via a finite element method that also introduces a nearly consistent mass matrix, Part 1: Theory, Part 2: Implementation, *Int. J. Num. Meth. Fluids*, 11, pp.587–620, pp.621–659, 1990.
19) Barragy,E. and Carey,G.F.: A parallel Element-by-Element solution scheme, *Int. J. Num. Meth. Engrg.*, Vol.26, pp.2367–2382, 1988.
20) 水上昭：非圧縮流れ解析に現れるコンシステントな圧力方程式のElement-by-Element解法, 第7回計算力学シンポジウム報文集, pp.1–6, 1993.
21) 水上昭：Element-by-Element PCG法のベクトル化と流れ解析への応用, 第2回計算力学シンポジウム報文集, pp.1–6, 1988.
22) 東京大学大型計算機センター：センターニュース (Vol.28–29), 1996–1997.
23) Saad,Y.: Iterative Methods for Sparse Linear Systems, 447p., PWS Publishing Company, 1996.
24) 青山幸也：RS/6000 SP 並列プログラミング虎の巻 (MPI版), 日本アイ・ビー・エム, 1996.
25) 富士通株式会社：AP1000プログラム開発手引書 (FORTRAN言語インタフェース第四版), 1994.
26) Kashiyama,K., Saitoh,K., Behr,M. and Tezduyar,T.E.: Parallel Finite Element Methods for Large-scale Computation of Storm Surges and Tidal Flows, *Int. J. Num. Meth. Fluids*, Vol.24, pp.1371–1389, 1997.
27) Kashiyama,K., Ito,H., Behr,M. and Tezduyar, T.E.: Three-step Explicit Finite Element Computation of Shallow Water Flows on a Massively Parallel Computer, *Int. J. Num. Meth. Fluids*, Vol.21, pp.885–900, 1995.
28) Cantwell,B. and Coles,D.: An experimental study of entrainment and transport in the turbulent near wake of a circular cylinder, *J. Fluid Mech.*, Vol.136, pp.321–374, 1983.
29) Tamai,T. and Kashiyama,K.: 3-D paraallel finite element analysis of incompressible flow based on unstructured grid, *Proc. 7th Int. Conf. on Computing in Civil and Building Engineering.*, Vol.3, pp.1573-1578, 1997.
30) 玉井典, 樫山和男：非構造格子に基づく三次元非圧縮粘性流れの並列有限要素解析, 第11回数値流体力学シンポジウム講演論文集, pp.557–558, 1997.

(1999. 3. 31 受付)

MASSIVELY PARALLEL COMPUTATIONAL METHOD FOR LARGE-SCALE INCOMPRESSIBLE VISCOUS FLOW BASED ON UNSTRUCTURED GRIDS

Kazuo KASHIYAMA and Tsukasa TAMAI

A parallel finite element method based on domain decomposition method is presented for large-scale computation of incompressible Navier-Stokes flow. The finite element computations, carried out using unstructured grids, are based on semi-implicit formulation. The pressure Poisson equation, which consume the most of computational time, is completely parallelized using element-by-element SCG method. The present method is shown to be a useful and powerful tool for the large-scale computation of incompressible Navier-Stokes flow.