オーダリングによる EBE データ並列有限要素法の効率化*

奥田 洋 司^{*1}, 阿南 統 久^{*2}Efficient Data-Parallel Computations of Element-by-Element
Finite Element Method by OrderingHiroshi OKUDA^{*3} and Norihisa ANAN^{*3} Yokohama National University, Dept. of Mechanical Engineering and Materials Science
Tokiwadai 79-5, Hodogaya-ku Yokohama, Kanagawa, 240-8501 Japan

This study aims to develop a partitioning algorithm for data parallel computation. The greedy partitioning algorithm for the message passing developed by Farhat was modified for the data parallel computation of an element-by-element finite element method. The influence of the numbering of nodes and elements onto the parallel efficiency is examined through the parallel computations using CRAY-T 3 D. It was shown that the proposed partitioning algorithm considering the ordering of nodes and elements for data parallel computation can save time of solving problems and will improve the efficiency of parallelism. In the case of CRAY-T 3 D with 128 PEs, the maximum number of communications of the cyclic ordering version in the flow inside a subway station model with about 1 410 000 elements is 89% smaller than the original one. The rate of CPU usage of the cyclic ordering version is 92%.

Key Words: Computational Fluid Dynamics, Numerical Analysis, Finite Element Method, Domain Decomposition Method, Data Parallel Computing, Partitioning, Ordering

1. 序 論

有限要素法の進歩と計算機性能の飛躍的な向上により、非線形三次元問題などの大規模解析が頻繁に行われるようになってきた。その結果、記憶容量の節約と計算の高速化が求められると同時に、分散メモリー形超並列計算機の仕組みによく適合するような計算手法が必要とされている。

超並列計算機を用いたシミュレーションでは、プロセッサ(PE)が分散配置されていることから、自然な考え方として領域分割法(Domain Decomposition Method)を用いるのが一般的である。領域分割の仕方(Partitioning)は、通信量、負荷バランス、反復法ソルバの前処理等に関連して、計算効率に大きな影響を及ぼすことが知られている。

一方、並列プログラミング手法としては、制御並列形とデータ並列形がある。前者は、MPI(Message Passing Interface), PVM(Parallel Virtual Machine)などのメッセージパッシングライブラリー

を利用することで、PE間通信等の明示的なコントロールが可能である。しかし、複数のPEを意識しなければならず、アプリケーション開発者への負担が大きい。後者は、最小限の付加的指示を挿入するだけでDOループの並列化が行えるので、従来のプログラム資産のインプリメンテーションが容易である。また、HPF(High Performance Fortran)の言語仕様も標準化しつつあることから今後の発展が期待される。しかしながら、通信等をユーザが陽には制御できないため、特に、三次元非構造格子問題における計算効率の低下が指摘されている。

そこで本研究では、データ並列計算に、制御並列計算向けであったPartitioningアルゴリズム(Greedy法)⁽¹⁾を適用し、さらに、オーダリング(要素番号および節点番号のリナンバリング)を施すことで、各PE間の負荷分散および通信量の低減化を可能にする手法を開発した。そして、それをElement-by-Element(EBE)データ並列有限要素法⁽²⁾に適用し、分散メモリー形超並列計算機CRAY-T 3 D上で、計算時間および各PE間のデータローカリティ(通信回数の少なさ)を測定することにより、三次元非構造格子問題(地下鉄駅構内の流れ解析)における計算効率を評価した。なお、分散メモリー形であることから、効率的な配列

* 原稿受付 1999年2月4日。

^{*1} 正員, 神奈川国立大学工学部 (〒240-8501 横浜市保土ヶ谷区常盤台79-5)。^{*2} 横浜国立大学大学院。

E-mail: okuda@typhoon.cm.me.ynu.ac.jp

確保を行うために、計算手法として 3 種類のオーダリング手法(ここでは, regular 版, dummy 版, cyclic 版と呼ぶ)を用いて、オーダリング前とオーダリング後を比較した。

2. 非圧縮性流れ解析の EBE アルゴリズム

2・1 非圧縮性流れ解析のアルゴリズム⁽²⁾ 非圧縮粘性流体の運動は, Navier-Stokes 方程式および連続の式に, Galerkin 法を適用することによって, 下記の行列方程式で表現される。

Navier-Stokes 方程式

$$\mathbf{M}\dot{\mathbf{u}}_i + \mathbf{B}(\mathbf{u})\mathbf{u}_i - \mathbf{C}_i\mathbf{p} + \mathbf{D}\mathbf{u}_i = \mathbf{f}_i \cdots \cdots (1)$$

連続の式

$$\mathbf{C}_i^t \mathbf{u}_i = 0 \cdots \cdots (2)$$

ここで, $\mathbf{u}_i$ は速度の節点量をすべて含む全体列ベクトル, $\mathbf{p}$ はすべての要素圧力からなる全体列ベクトルであり, $\mathbf{f}_i$ は外力ベクトルである。 $\mathbf{M}$, $\mathbf{B}(\mathbf{u})$, $\mathbf{C}_i$, $\mathbf{D}$, $\mathbf{C}_i^t$ はそれぞれ, 質量, 対流, こう配, 拡散, 発散を表す全体行列である。速度に関して一次, 圧力に関してゼロ次(要素内一定)の基本関数を用い, 形状に関しては一次の基本関数で近似する。

時間積分には, 非圧縮性流れの各種解法のうち分離形解法の一種である速度修正法という解法を用いた。式(1), (2)を流速に関して陽, 圧力に関して陰に取扱い, 次のように変形する。

$$\mathbf{u}_{n+1} = \mathbf{u}_n - \Delta t \mathbf{M}^{-1} \{ \mathbf{B}(\mathbf{u}_n) \mathbf{u}_n - \mathbf{C}_{p_{n+1}} + \mathbf{D} \mathbf{u}_n - \mathbf{f}_n \} \cdots \cdots (3)$$

$$(\mathbf{C}^t \mathbf{M}^{-1} \mathbf{C}) \mathbf{p}_{n+1} = -\frac{1}{\Delta t} \mathbf{C}^t \mathbf{u}_n + \mathbf{C}^t \mathbf{M}^{-1} \{ \mathbf{B}(\mathbf{u}_n) \mathbf{u}_n + \mathbf{D} \mathbf{u}_n - \mathbf{f}_n \} \cdots \cdots (4)$$

ここで, Δt は時間きざみ幅で, n は時間ステップである。

式(4)から, CG(Conjugate Gradient)法を用いて, 圧力 $\mathbf{p}_{n+1}$ を求め, 次に式(3)から, 流速 $\mathbf{u}_{n+1}$ を求めることで, 時間が進められる。

2・2 Element-by-Element スキーム 式(3), (4)を解くのに EBE スキームを用いると, 以下に示す式(5)のような連立一次方程式などを解く際に, 全体行列を作成することなく, エレメントワイズに演算が行える。また, ソルバに反復法である CG 法を用いると, 主要な計算部分が行列とベクトルの積で形成されているため, 解法のすべての過程を EBE スキームに取り込むことができる。

$$\mathbf{A} \mathbf{x} = \left(\sum_{e=1}^{N_{elem}} \mathbf{A}_e \mathbf{x}_e \right) = \left(\sum_{e=1}^{N_{elem}} \mathbf{b}_e \right) = \mathbf{b} \cdots \cdots (5)$$

EBE スキームを用いる利点は, 全体行列を作成する必要がないため記憶容量を節約でき, 要素ループが独立して計算されるのでデータ並列計算との整合性が非常によいことである。

3. データ並列計算

3・1 CRAY-T3D の性能 本研究で用いた CRAY-T3D⁽³⁾⁽⁴⁾ は, RISC マイクロプロセッサをもつ 128 個の PE を, 三次元トラス構造のインタコネクトネットワークにより結合した MIMD(Multiple Instruction Multiple Data stream)形超並列計算機システムである。表 1 にもな性能を示す。また, CRAY-T3D 専用の Fortran コンパイラを用いることによって, SIMD(Single Instruction Multiple Data stream)形共有メモリーのような使い方ができる。

3・2 データシェアリングとワークシェアリング

CRAY-T3D には, データ並列化手法として, コンパイラディレクティブ `cdir$ shared A(:block)` を用いることによって, 配列 A を各 PE に分散させる Data Sharing, および, コンパイラディレクティブ `cdir$ doshared(i) on A(i)` を用いることによって, 配列 A(i) の i で回す DO ループを並列化する Work Sharing とがある。下記に配列 A(8) を Data Sharing で 2 個の PE に block 分割し, DO ループを Work Sharing で並列化した例を図 1 に示す。各 PE の所有するデータは, メッセージパッシングライブラリーを用いることなく, 他のどの PE からでも参照可能である。また, A(i) のデータを所有する PE にその DO ループ

Table 1 Performance of CRAY-T3D

	Number of PEs	
	1PE	128PEs
Peak performance	150MFLOPS	19.2GFLOPS
Main memory	16MB	2GB
Comm. transfer speed	—	300MB/s

Fig.1 The array A(8) of Data-Sharing and Work-Sharing

処理が割り当てられる。

3.3 配列サイズの制限 分散メモリ形超並列計算機でデータ並列を行う場合、配列を確保する際に、配列サイズに注意する必要がある。例えば CRAY-T 3 D では、実際に必要とされる配列 A の大きさが 2 のべき乗に一致しない。例えば A(100) のとき、Data Sharing によって、各 PE (4 個の PE とする) に分散されるデータ量に不均衡が生じる。つまり、最外 DO ループを配列総数 (100) で回す場合、それよりも大きい 2 のべき乗個 (128) の配列 A' (128) が確保されるため、DO ループ処理を行わない PE が生じ、各 PE は不均等な負荷分散になる (図 2)。

高い計算効率を得るためには、各 PE の負荷を均等にする必要があり、Data Sharing では、データを分散させる際に、block の大きさ (分散させるデータサイズ n) を指定して、負荷の均等化を考慮できるようになっている。なお、 $n=1$ のとき負荷の均等化が最適

Fig. 2 The limitation of the number of shared array elements to powers of 2 are influential in load imbalance

Fig. 3 Distribution of the array data A(100); [block distribution, $n=16$, $n=1$ (cyclic distribution)]

となり、このことを cyclic 分割という (図 3)。しかしながら、cyclic 分割は、三次元非構造格子問題の場合、複雑なデータ構造となることから、何も工夫しなければ、各 PE 間の通信量は逆に増える原因となる。

4. EBE アルゴリズムにおける通信量の定量的評価

4.1 通信の発生 EBE スキームの場合、Work Sharing する最外 DO ループを総要素数で回し、その内側をその要素を構成する節点の DO ループとする演算が多用される。そのため、要素データと、その構成節点データが異なる PE 上に存在する場合、計算に必要なデータを他の PE から参照しなければならず通信が発生する。通信が発生すると、自分のメモリーを参照するよりも大幅に時間がかかることから、計算効率の向上のためには、データローカリティが重視される。つまり、EBE スキームで計算効率を向上させるためには、要素と、その構成節点をなるべく同じ PE に分散させなければならない。非構造格子の並列計算が、構造格子の場合よりも一般に効率が低いのは、まさにこの理由による。下記に総要素数 8、総節点数 15 のメッシュを 2 個の PE に cyclic 分割し、オーダリング前後の各 PE ごとの通信回数での最大値の違いを示す (図 4)。オーダリング後の通信回数が、5 減少していることから、その分計算効率が向上することがわかる (なお、ここでは、 $5 \cdot 2 \cdot 4$ 項で示す Cyclic 版オーダリング手法を用いている)。

4.2 通信回数の計測 CRAY-T 3 D の組込み関数の一つで、データを所有する PE 番号を知ることができる `my_pe()` を用いることにより、通信回数の計測を行う手法を提案する (図 5)。`me=my_pe()` として、データが格納されている PE 番号 `me` を得る。節点と要素のそれぞれを所属する PE 番号を `my_pe()`

Fig. 4 Difference of Data-Locality

Case for PE0

PE number of element 7; ME = 0

PE number of node 5; me = 0

ME - me = 0 (Don't Add 1 to Num. of Comm.)

PE number of node 2; me = 1

ME - me = -1 (Add 1 to Num. of Comm.)

Fig. 5 Count number of communications among PEs

で求め、要素と、その構成節点の所属する PE 番号同士を引く。同じ PE に属していれば零になり、異なる PE ならば、零以外の数字になる。この零以外の数字の個数をカウントすることにより、最外 DO ループを総要素数で回し、その内側を構成節点数の DO ループとする演算の通信回数を計測することができ、データローカリティの定量的な比較が可能となる。

5. データ並列計算向けオーダリング手法

5.1 Partitioning Partitioning に必要な条件は以下のとおりである。

- (1) 分割数の任意性と不規則な幾何学図形の処理
- (2) 各部分領域 (PE) の負荷の均等化
- (3) 各部分領域 (PE) の通信量の低減化

Partitioning アルゴリズムとして用いた Greedy 法⁽¹⁾は、要素分割された解析対象を、各 PE の要素数が等しくなるように要素をグループ化して部分領域に分割する。そして領域分割を行った後、各部分領域 (PE) ごとに、節点番号を部分領域の内部の節点から付け、部分領域の境界上にある節点の番号を最後にまとめて付ける処理を行う。これにより、各部分領域のアスペクト比(縦横比)が小さく保たれるような領域分割が施される。

5.2 オーダリング手法の提案 regular 版, dummy 版, cyclic 版⁽⁵⁾の 3 種類のオーダリング手法を提案する。

5.2.1 初期データ (original) 初期データをここでは何もオーダリングせずに、そのまま解析を行う場合を original と呼ぶ。例として図 6 に、総要素数 12, 総節点数 22 のメッシュを 4 領域で分割したときのデータ分散を示す。なお、2 のべき乗の配列サイズ制限のため、要素配列 E(12), 節点配列 N(22)は、PE 3 にデータが格納されない。すなわち、不均等な

Fig. 6 Block distribution of original mesh data

Fig. 7 Block distribution of regular version

Fig. 8 Block distribution of dummy version

データ分散となり、PE 3 では計算負荷が発生しない。

5.2.2 regular 版 初期データに Greedy 法を用いて Partitioning を施しただけでは、実は有効な負荷分散とはならない。なぜなら、図 6 の例では、使用される PE は 3 個であり、Greedy 法を用いて、4 領域に分割すると、データは均等に 4 分割され、適合しないからである。そこで、配列サイズから実際に使用される PE 数を割り出し、その PE 数分の Partitioning を施す。その後、要素番号および節点番号が各 PE 内で、

なるべく連続になるように、オーダリング(要素番号および節点番号のリナンバリング)を施す(これを regular 版と呼ぶ)。図 7 に regular 版のデータ分散例を示す。これにより、負荷分散は改善されないが、データローカリティは改善される。

5・2・3 dummy 版 使用するすべての PE に負荷分散されるように、初期データとは別空間に dummy メッシュを作成し、データローカリティを考慮しながら、dummy データを挿入する(これを dummy 版と呼ぶ)。図 8 に dummy 版のデータ分散例を示す。これにより、dummy 分だけ総要素数および総節点数が増加するが、負荷分散は改善される。

5・2・4 cyclic 版 負荷の均等化を実現する手法である cyclic 分割を利用する。ただし、初期データ(original)にそのまま cyclic 分割を適用しても、負荷分散は改善するが、データローカリティは悪化する(図 9)。そこで、データローカリティを向上させるオーダリング処理を以下の法則で施す(これを cyclic 版と呼ぶ)。図 10 に cyclic 版のデータ分散例を示す。これにより、各 PE 間の均等な負荷分散および通信量の低減化が可能になる。

〈cyclic 版オーダリング処理〉

Used number of PEs: 4

Maximum number of communications: 10

Fig. 9 Cyclic distribution of original mesh data

Used number of PEs: 4

Maximum number of communications: 5

Fig. 10 Cyclic distribution of cyclic version

- (1) Partitioning により、部分領域(PE)に分割
- (2) PE(n) に分散された要素に、式(6)に従って新たな要素番号を設定(リナンバリング)

新要素番号 = $i \times \text{使用 PE 数} + n + 1$

($i=0, l_{\text{sub}}-1$) ..... (6)

l_{sub} = 各 PE に含まれる要素数

= 総要素数 / 使用 PE 数

n = PE 番号

- (3) 節点についても、(2)の法則で、リナンバリング処理

6. 計算結果

6・1 解析モデル 非構造格子問題として地下鉄駅構内における流れ解析を取り上げる。各種オーダリング手法を適用し、通信量削減の効果および計算効率を比較する。なお、約 180 000 要素と約 1 410 000 要素の要素分割モデルを用いた。図 11 に Greedy 法を用いて約 180 000 要素モデルを 11 個の領域に分割した例を示す。

Fig. 11 Partitioning of a subway station model (11 distributions)

Fig. 12 Maximum number of comm (180 000 elements)

6.2 通信回数 約 180 000 要素地下鉄駅構内問題の流れ解析について、3 種類のオーダリング手法 (regular 版, dummy 版, cyclic 版) を適用し、4.2 節で提案した手法を用いて、オーダリング前後の各 PE ごとの通信回数での最大値を比較する。ただし、プログラム全体の通信回数ではなく、4.1 節の Work Sharing された DO ループ、一つに対する通信回数を計測する。その結果を図 12 に示す。縦軸は一つの Work Sharing における最大通信回数、横軸は使用 PE 数とし、8, 16, 32, 64, 128 の五つを評価点とする。なお、約 180 000 要素の問題は、CRAY-T 3 D では、8 PEs からしか計算できない。

オーダリングを施すことにより、cyclic 版, dummy 版の通信回数が激減することがわかる。その効果は、8 PEs 使用時に、regular 版を適用する前の最大通信回数と比較すると、cyclic 版は 97%, dummy 版は 98%, regular 版は 25% 減少する。regular 版の効果があまり表れないのは、要素データと節点データの実際に分散される PE 数が異なるためである。また、使用 PE 数が増加すると、各 PE のグラニュラリティ (他の部分とデータを交換し合うことなく実行できる計算の大きさ)⁽⁶⁾ が減少することから、それらの効果は圧縮され、128 PEs 使用時の cyclic 版では 79% の減少になる。

また、約 1 410 000 要素の問題は、CRAY-T 3 D では、64 PEs からしか計算できないが、使用 PE 数が多いときでも、各 PE のグラニュラリティが大きいことから、同様の効果が表れる。その効果は、128 PEs 使用時に、regular 版を適用する前の最大通信回数 131 072 と比較すると、cyclic 版では 89% の減少になる。

なお、この各 PE ごとの通信回数での最大値は、たった一つの Work Sharing におけるものであり、EBE データ並列有限要素法のアルゴリズムには、この DO ループが数多く含まれているため、オーダリングの効果の大きさがうかがえる。

6.3 計算時間 約 180 000 要素地下鉄駅構内問題の流れ解析の 1 step (打切り CG 回数 100 回) について、3 種類のオーダリング手法 (regular 版, dummy 版, cyclic 版) を適用し、CG 法を用いた式 (4) などの解析部分の計算時間を測定した。

大規模問題ではメモリー容量の制限から単一 PE での解析が不可能となることから、並列計算を評価する手法の一つとして、 n 個の PE を使用した際に式 (7) で定義される CPU 利用率⁽⁷⁾ R_n が用いられている。

$$R_n = \frac{\sum (\text{Working Time})}{\sum \{(\text{Working Time}) + (\text{Idling Time})\}} \quad \dots\dots\dots (7)$$

Fig. 13 Rate of CPU usage (180 000-elements)

ここで、 $\{(\text{Working Time}) + (\text{Idling Time})\}$ は各 PE での計算時間、(Idling Time) は通信時間と待ち時間の和である。

そこで本研究では、regular 版オーダリングを施した約 180 000 要素の 1/8 規模である約 20 000 要素地下鉄駅構内問題の流れ解析 1 step (打切り CG 回数 100 回) を 1 PE で行い、その計算時間をもとにして、1 要素あたりの Working Time を概算した。なお、各 PE における Working Time は、概算した値に各 PE に含まれる要素数を掛けたものとし、Idling Time は、得られた計算時間とその Working Time を引いたものとしている。また、dummy 版については、dummy 要素分の Working Time は除いてある。

以上をもとに、その結果を図 13 に示す。縦軸は CPU 利用率、横軸は使用 PE 数とし、8, 16, 32, 64, 128 の五つの評価点とする。なお、約 180 000 要素の問題は、CRAY-T 3 D では、8 PEs からしか計算できない。

128 PEs 使用時での CPU 利用率を比較すると、cyclic 版は 86%, dummy 版は 62%, regular 版は 57% となる。CRAY-T 3 D では、非構造格子問題の場合、データ分散方式として、block 分割よりも、cyclic 分割を用いたほうが計算効率が大幅に改善する。また、dummy を挿入して負荷分散を均等にしても、それほど効果は表れない。すなわち、オーダリン

グ手法よりも、データ分散手法の変更による効果のほうが大きい。ただし、cyclic 版のオーダリング前後の CPU 利用率が、1%程度の効率改善でとどまる他の手法に比べ、80%から 86%と大幅に改善したことから、本手法の有用性が確信される。

同様に、約 1 410 000 要素の問題の場合、使用 PE 数が多いときでも、相対的に各 PE の通信量よりグラニュラリティが大きくなることから、CPU 利用率が向上し、128 PEs 使用時の cyclic 版の CPU 利用率は 92%となる。ただし、約 1 410 000 要素の問題は、CRAY-T 3 D では、64 PEs からしか計算できない。

なお、上記の計算時間とは、1 step(打切り CG 回数 100 回)での測定であり、通常の大規模計算では、数百 step、打切り CG 回数百回が、普通であることから、さらに大きな効果が得られることが期待できる。

6・4 考察 最大通信回数と CPU 利用率との関連性からみると、オーダリングを施すことで、cyclic 版の CPU 利用率が増加したのは、通信回数が激減したことが最大の要因であることが確かめられる。そして、dummy 版、cyclic 版の通信回数が、あまり変わらないのに、dummy 版の CPU 利用率が cyclic 版よりも大幅に悪化したのは、負荷分散の均等化のため、配列に挿入した dummy データを含めて計算しているからである。

また本手法は、2 のべき乗制限のない CRAY-T 3 D 以外の分散メモリー形超並列計算機におけるデータ並列環境においても有用であり、約 180 000 要素地下鉄駅構内問題に、cyclic 版オーダリング手法を適用し、SR 2201、VP 300 の 128 PEs 使用時の CPU 利用率 R_n を式(8)を用いて推定する。

$$R_n = \frac{\sum \left(\frac{P_{cray}}{P_{mpp}} W_{cray} \right)}{\sum \left\{ \left(\frac{P_{cray}}{P_{mpp}} W_{cray} \right) + \left(\frac{C_{cray}}{C_{mpp}} I_{cray} \right) \right\}} \quad \cdots (8)$$

ただし、 P_{cray} を CRAY-T 3 D の、 P_{mpp} を SR 2201 および VP 300 のピーク性能、 C_{cray} および C_{mpp} を転送速度、 W_{cray} を Working Time、 I_{cray} を Idling Time とする。表 2 にそれぞれの性能および CPU 利用率を示す。

CPU 利用率は、CPU の処理速度と転送速度の比に大きく左右されるものなので、VP 300 での CPU 利用率が極端に低いように思えるが、オーダリング手法適用前の CPU 利用率の推定値が、34.6%となることから、オーダリングを施すことは、必要不可欠である。

なお一般に、リナンバリングオーダリングという計算時間がかかるというイメージがあるが、本研究で

Table 2 Estimated rate of CPU usage for SR 2201 and VPP 300

	Peak perf. (1PE): P	Comm. transfer speed: C	Rate of CPU usage: R
Cray-T3D	150Mflops	300MB/s	85.9%
SR2201	300Mflops	300MB/s	75.3%
VPP300	2.2Gflops	570MB/s	44.2%

提案したオーダリング手法の計算時間は、使用 PE 数に応じて解析前に一度だけ適用すればよく、大規模な EBE 計算の計算時間に比べれば短くて済み、減少する計算時間を考えると大変有用な手法である。

7. 結 論

データ並列計算の最大の特徴は、ユーザが複数の PE を意識することなく、仮想的に一つの PE 上で問題を解析できることである。しかし、分散メモリー形超並列計算機において、計算効率を向上させようとすると、入力データ依存となり、通信を最小限にするような、効率的な配列の取り方を考慮しなければならない。そのため、非構造格子問題での計算効率の低下が指摘されてきた。

データ並列計算環境において、非構造格子問題を解析する場合、分散メモリー形超並列計算機の最高性能を引き出すためには、Greedy 法などの Partitioning の概念を導入し、cyclic 分割向けのオーダリング(初期データの節点番号および要素番号のリナンバリング)を施したものが、各 PE 間の負荷分散およびデータローカリティの両方を同時に満足することができ、最良であった。

<cyclic 版の効果>

・約 180 000 要素地下鉄駅構内問題(128 PEs 使用時)

通信回数削減率 79%

CPU 利用率 86%

・約 1 410 000 要素地下鉄駅構内問題(128 PEs 使用時)

通信回数削減率 89%

CPU 利用率 92%

したがって、本研究により、非構造格子問題のデータ並列計算環境における、通信量の定量的な評価および計算効率の向上が可能となった。

本研究の並列計算にあつては動力炉・核燃料開発事業団(現:核燃料サイクル開発機構)の並列計算機 CRAY-T 3 D を使用させていただいた。ここに記して謝意を表す。

文 献

- (1) Farhat, C., A Simple and Efficient Automatic FEM Domain Decomposer, *Comput. Struct.*, **28**-5 (1988), 579-602.
  - (2) Yagawa, G., Nakabayashi, Y. and Okuda, H., Large-Scale Finite Element Fluid Analysis by Massively Parallel Processors, *Parallel Comput.*, **23** (1997), 1365-1377.
  - (3) CRAY-T3D System Architecture Overview HR-04033, Cray Research, Inc., (1994).
  - (4) CRAY MPP Fortran Reference Manual SR-2504 6.2, Cray Research, Inc., (1993).
  - (5) 奥田洋司・阿南統久, データ並列有限要素法のためのパーティショニングおよびリネンバリング, *機講論*, **II**, No. 98-3 (1998), 49-50.
  - (6) 矢川元基・曾根田直樹, パラレルコンピューティング, (1991), 培風館.
  - (7) 矢川元基・塩谷隆二, 超並列有限要素解析, (1998), 朝倉書店.
-