計算機アーキテクチャ 137－24
ハイパフォーマンス
コンピューティング 80－24
（2000．3．3）

三次元ポアソン方程式に対する領域分割法の分割方法による性能への影響

黒川原佳[†] 姫野龍太郎[††]
重谷隆之[††] 松澤照男[†††]

流体シミュレーションで用いられる 3 次元ポアソン方程式を差分法で離散化し，領域分割法で並列化した場合の分割次元と方向による計算性能全体への影響を示す．PC クラスタでは，静的な領域分割をどのように行なえば良いのかを数値実験的に明らかにした．162^3, 82^3 という問題量の領域分割を行う場合，明らかに通信するデータ量が多い場合を除いて，分割された計算領域形状を 1 PE の計算する性能が全体の性能に大きく依存することが分かった．また，通信による影響は，通信メッセージ量が明らかに多くない限り，全体性能には大きな影響を及ぼさなかった．静的に領域分割を行う場合，領域分割で得られる計算領域形状を 1 PE で計算した最も良い値のものを選べばほぼ適値が得られる．

Performance Efficiency of Partitioning Pattern for Three Dimensional Poisson Equation

Motoyoshi Kurokawa,[†] Ryutaro Himeno,[††] Takayuki Shigetani[†††]
and Teruo Matsuzawa[†††]

We studied parallel performance from partitioning pattern(the domain decomposition method) for three-dimensional Poisson equation using the finite difference method on a PC cluster. We discussed better static partitioning for three dimensional Poisson equation by the domain decomposition method.

Total performance would be estimated sequential performance in a domain shape. In problem size 162^3, 82^3, parallel performance dependent sequential performance in a computational domain shape by the domain decomposition method. On the other hand, communication performance had no serious effect unless total performance with a average message size on a PC cluster.

1. はじめに

近年の PC およびネットワークの高性能化，低価格化によって PC を 100 Base ネットワークで接続した PC クラスタが高性能かつ低価格な並列計算機として注目されている [1]．流体解析等のような大規模かつ高精度な数値シミュレーションを実用時間レベルで行うような解析では，これまでハイエンドのベクトル計算機や超並列計算機を用いて解析が行われてきた．これらの計算機システムは，非常に高価なため利用にはかなりの制限があった．PC クラスタが，これらの計算機システムに匹敵する能力を持つことは，シミュレーションを高速に制限なく行える環境を安価に作成でき，シミュレーションのための重要なキーとなるものと考えられる．

流体シミュレーションを考えた場合，三次元非圧縮 Navier-Stokes 方程式 (NS 方程式) のうち，圧力ポアソン方程式を解くことが，計算時間の中で大きな割合を占めることが知られている．特に非定常の NS 方程式では，少しづつ変化していく圧力ポアソン方程式を時間ステップ毎に解く必要がある．以上のことから，3 次元ポアソン方程式を解くことで，ある程度の流体シ

† 北陸先端科学技術大学院大学 情報科学研究科
School of Information Science, Japan Advanced Institute of Science and Technology, Hokuriku
†† 理化学研究所 情報環境室
Division of Computer and Information, RIKEN (The Institute of Physical and Chemical Research)
††† 北陸先端科学技術大学院大学 情報科学センター
Center for Information Science, Japan Advanced Institute of Science and Technology, Hokuriku

ミュレーションの予測が可能となる.

流体シミュレーションを並列で行うことは数多く取り組まれている [2]. 多くの流体シミュレーションで持ちいられている方法は, 計算領域を小領域に分割し, 各プロセッサを割り当てる領域分割法が広く用いられている. しかし, 分割方法に関してはあまり研究がなされておらず, 経験則で分割が行われている.

本報では, PC クラスタを用いて 3 次元の計算空間を静的に分割する方法を用い, 分割次元と分割数による並列計算性能への影響を考えた. 3 次元ポアソン方程式で PE(Processing Element) 数と問題量を固定し, 全ての分割パターンで領域分割を行った. その結果から分割次元と方向の違いによる計算性能への影響を考察した.

2. 計算条件

2.1 ポアソン方程式

一般的な非圧縮粘性流体の有限差分解析におけるポアソン方程式を用いた. 計算格子は, 正方形格子ではなく, 物体適合格子を仮定した. 解析領域は直方体計算領域に写像され, 計算領域を直接メモリ配列に割り当てることで計算を行った.

$$\frac{\partial p}{\partial x}+\frac{\partial p}{\partial y}+\frac{\partial p}{\partial z}=-f(x,y,z) \qquad (1)$$

$$\begin{cases} x=x(\xi,\eta,\zeta) \\ y=y(\xi,\eta,\zeta) \\ z=z(\xi,\eta,\zeta) \end{cases}$$

ξ,η,ζ は計算領域での座標を示し, x,y,z は物理領域での座標を示す. 物理領域での格子点 (x,y,z) の数値が既知であれば, それを用いて変数変換を行って得られる方式程の全ての係数が数値的に決定される.

このポアソン方程式を二次精度の中心差分で離散化した. 解法には領域分割の違いによる収束性への影響が出ないヤコビ法を用いた. ヤコビ法は, 自然点順序の緩和を行う. 実装は Fortran 77 と MPI を用い, 計算精度は単精度とした. 計測は, ある程度の大規模問題を考慮し, 格子点数 162 × 162 × 162 (425 万点), 82 × 82 × 82 (55 万点) の 2 通りとし, 200 反復の経過時間を計測した. 使用した PE 数は 8 とした. 図 1 に今回用いたコードを示す [3]. プログラムの最適化は, コンパイラのみによって行い, コードの最適化等は行っていない. 計算に用いるメモリ量は 162^2 で 240 MByte 程度, 82^3 で 30 MByte 程度となる.

```
do k=2,kmax-1
  do j=2,jmax-1
    do i=2,imax-1
      s0 = a(i,j,k,1)*p(i+1,j,k)
          +a(i,j,k,2)*p(i,j+1,k)
          +a(i,j,k,3)*p(i,j,k+1)
          +b(i,j,k,1)*(p(i+1,j+1,k)-p(i+1,j-1,k)
                      -p(i-1,j+1,k)+p(i-1,j-1,k))
          +b(i,j,k,2)*(p(i,j+1,k+1)-p(i,j-1,k+1)
                      -p(i,j+1,k-1)+p(i,j-1,k-1))
          +b(i,j,k,3)*(p(i+1,j,k+1)-p(i-1,j,k+1)
                      -p(i+1,j,k-1)+p(i-1,j,k-1))
          +c(i,j,k,1)*p(i-1,j,k)
          +c(i,j,k,2)*p(i,j-1,k)
          +c(i,j,k,3)*p(i,j,k-1)+wrk1(i,j,k)
      ss=(s0*a(i,j,k,4)-p(i,j,k))*bnd(i,j,k)
      gosa=gosa+ss*ss
      wrk2(i,j,k)=p(i,j,k)+omega * ss
    end do
  end do
end do

do k=2,kmax-1
  do j=2,jmax-1
    do i=2,imax-1
      p(i,j,k)=wrk2(i,j,k)
    end do
  end do
end do
```

図 1 Poisson Solver using Jacobi Method

2.2 領域分割法

領域分割法は, 解析対象となる領域を幾つかの部分領域に分割し, 各部分領域毎に解析を行うことで全体領域の解を求める手法の総称である. 領域分割法を実装する際に重要となる問題は, 並列化率の向上, 並列処理粒度, 各 PE での負荷の均等化, 並列処理時の通信の隠蔽等である. また, 小領域に分割することで, キャッシュの効果が大きく現れることが考えられる.

3 次元ポアソン方程式を反復解法で解く場合, 領域を 2 次元で分割し, パイプライン処理することで通信の隠蔽が行われ, 高並列による性能低下を抑えることが出来る [4),5)] が, 実装とプログラムの保守に手間が掛かるという欠点がある. そのため, 既存コードを用いたシミュレーションでは, 実装の手間を考えるとあまり現実的ではないため, 静的分割で分割領域内を通常の反復法を用いて解く場合の検討を行った.

本報では 3 次元非圧縮性流れシミュレーションの中核部分である 3 次元ポアソン方程式を静的にブロック分割し並列化した. 領域分割のパターンは, 8 PE で 8 ブロックに分割することを考える. 1 次元分割では, i,j,k 各方向に 8 等分した $D_{(8,1,1)}$, $D_{(1,8,1)}$, $D_{(1,1,8)}$ の 3 パターン, 2 次元分割では, 1 方向に 4 等分と 2 等分した

$D_{(4,2,1)}$, $D_{(4,1,2)}$, $D_{(2,4,1)}$, $D_{(1,4,2)}$, $D_{(2,1,4)}$, $D_{(1,2,4)}$ の 6 パターン, 3 次元分割では, 全ての方向に 2 等分した $D_{(2,2,2)}$ の 1 パターンの計 10 パターンである. 分割パターンの表記は, $D_{(i,j,k)}$ として, i,j,k はそれぞれの方向の領域分割数を示す.

2.3 計算コスト

このポアソン方程式の計算には, $34\times(N-2)^3\times ITR$ 回の浮動小数点演算 (FLOPs) が必要になる. N は各方向の格子点の数, ITR は反復回数, p は使用 PE 数とした. 格子点 1 点の解を得るために 34 FLOPs が必要となる. 全体領域の境界はディレクレ境界のため, 両端の境界値の演算は行わない.

本報の条件での並列化に伴う演算量の減少は, どの分割パターンを用いても 1/p になる. 並列化した場合の 1 PE の 1 反復あたりの演算量 ($FLOPs_1$) は式 (2) で表され, 計算負荷はどの PE も一致している.

$$FLOPs_1 = 34 \times \frac{(N-2)^3}{p} \tag{2}$$

2.4 メッセージ通信

Point to point のメッセージ送受信のために MPI_SEND() と MPI_RECV() が呼び出される回数は, 1 PE に対して分割方向に対して各 2 回である. 1 次元分割で 2 回, 2 次元で 4 回, 3 次元で 6 回発生する. また, 通信のためのバッファコピーも回数分通信量だけ発生する. 1 反復中に MPI_SEND(), MPI_RECV() で発生するメッセージ通信量 (Byte) は以下のようになる.

- 1 次元

$$2\times N^2\times 4_{Byte} = 8\times N^2$$

- 2 次元

$$2\times\left(\frac{N^2}{4}+\frac{N^2}{2}\right)\times 4_{Byte} = 6\times N^2$$

- 3 次元

$$2\times\left(\frac{N^2}{4}\times 3\right)\times 4_{Byte} = 6\times N^2$$

計算の全体領域が立方体であるため, 2 次元分割と 3 次元分割は, MPI_SEND(), MPI_RECV() の呼び出し回数と 1 回あたりの通信量が異なるが, 通信の全体量はほぼ同量となる. また, 領域分割法の反復計算では, 残差の総和計算が発生する. 残差の総和計算は分割方法には依存しないため, 本報では反復計算中に総和計算を行ってない.

2.5 並列計算全体のコスト

本報のポアソン方程式を解くための全体コストは, 通信の隠蔽等を行っていないため, 通信時間と計算時間の和によってほぼ決定できる. f を 1 FLOPs の計算時間 (秒), l を通信の立ち上がり時間 (秒), b を 1 Byte の通信時間 (秒), E_α は 1 PE の計算領域における利用効率, C_α は明確に現れないバッファコピーや同期遅延等の通信の付加時間 (秒) とする. よって各分割における計算時間は次にように予測できる [5),6)].

- 1 次元

$$\begin{aligned}&\left(FLOPs_1\times\frac{f}{E_\alpha}+2\times(4\times N^2\times b+l+C_\alpha)\right)\times ITR\\&=\left(FLOPs_1\times\frac{f}{E_\alpha}+(8N^2b+2l+C_\alpha)\right)\times ITR\end{aligned} \tag{3}$$

- 2 次元

$$\begin{aligned}&\left[FLOPs_1\times\frac{f}{E_\alpha}+2\times\left(\left(4\times\frac{N^2}{4}\times b+l\right)+\left(4\times\frac{N^2}{2}\times b+l\right)+C_\alpha\right)\right]\times ITR\\&=\left(FLOPs_1\times\frac{f}{E_\alpha}+(6N^2b+4l+C_\alpha)\right)\times ITR\end{aligned} \tag{4}$$

- 3 次元

$$\begin{aligned}&\left(FLOPs_1\times\frac{f}{E_\alpha}+2\times 3\times\left(4\times\frac{N^2}{4}\times b+l+C_\alpha\right)\right)\times ITR\\&=\left(FLOPs_1\times\frac{f}{E_\alpha}+(6N^2b+4l+C_\alpha)\right)\times ITR\end{aligned} \tag{5}$$

もし, 計算時間を推定するならば b,l,f は, 各システムの理論値を用いることが出来る. しかし, b は, 流体シミュレーション等のメモリに負担のかかる計算を行う場合, PE 性能は確実に CPU 性能より低下する. そのため, E_α を定義して実際の演算能力に対応する. E_α は, 領域分割によって現れる 1 PE 分の計算形状に合わせた領域ブロックを 1 PE で計算することで実測値が得られる.

分割方法の性能への影響は並列可能部分の性能である E_α が大きく影響するものと仮定し, 領域ブロックを 1 PE で計算した場合を計測した. その結果から C_α の全体性能に対する割合を算出した. C_α は通信関連の遅延, 処理時間の総和となる数値である. メッセージ量から得られる通信時間がほぼ一定とすると,

C_α という値を評価することで領域分割による通信処理全体の効率が悪い分割パターンが特定できる.

計算時間の予測式の E_α と C_α を除いた確定的な要素のみを見比べた時, 1 次元分割は明らかに通信量が多い. また, 2, 3 次元では, 通信の立ち上がり時間 l の係数の違いのみとなる.

3. クラスタ構成

本報で用いた PC クラスタ [7),8)] (表 1) と評価用として IBM RS/6000 SP (表 2) の構成を示す. PC クラスタでは, 100BASE-T Network を計算用とサービス用 (NFS, NIS, etc.) の 2 本持つ. その概要を図 2 に示す.

また, SP は, 100BASE-T と SP-Switch のネットワークで接続された 4 CPU/1 Node の SMP クラスタである. 本報では, 1 ノードに 1 プロセスを割り当てて 8 ノード用いた. SP 上の通信では, 1 ノードに 2 プロセス以上割り当てると, ユーザーレベルでの通信が行われないだけでなく, シングルステージの通信帯域を複数の PE で共有するため多少通信性能が落ちる.

表 1 PC Cluster Hardware Specification

PE 数	9
CPU	Pentium III 450 MHz
2nd Cache	512 KB / CPU
メモリ	128 MB / PE (Server:256 MB)
ディスク	4.5 GB / PE
ネットワーク	100Base-T × 2 DEC 41143 i82558
OS	Linux-2.2.13
コンパイラ	PGI v3.0
オプション	-O2 -Mvect -Munroll
MPI Library	MPICH (PGI 付属)

表 2 RS/6000 SP Hardware Specification

ノード数	64 (4 CPU/1 Node)
CPU	PowerPC 604e 332 MHz
2nd Cache	256 KB /CPU
メモリ	512 MB/node
ネットワーク	SP Switch (150MB/s)
コンパイラ	XL Fortran v6
オプション	-O3 -qstrict -qarch=604

4. 結　　果

4.1 PC クラスタ

PC クラスタでの結果を図 3 に示す. 図は, 縦軸に Mflop/s, 横軸には $D_{(i,j,k)}$ の (i,j,k) をとった.

Mflop/s の最高値 $D_{(2,2,2)}$ と最低値 $D_{(8,1,1)}$ は, 問題量で一致した. 各分割パターンの違いによる Mflop/s の差は, 問題量 162^3 では, $D_{(8,1,1)}$, $D_{(1,8,1)}$, $D_{(2,2,2)}$ を除いて 540 Mflop/s 程度と大きな差は見られなかった. 82^3 では, $D_{(8,1,1)}$ が処理性能が低く, $D_{(1,8,1)}$, $D_{(1,4,2)}$, $D_{(2,2,2)}$ の場合が 500 Mflop/s 以上と良い値が得られた. また, 2, 3 次元分割の差である立ち上がり時間の影響によって 3 次元分割を用いても遅くなるということは見られなかった. 逆に 3 次元分割を行った方が問題量を問わず良い結果が得られた. 通常 1 次元分割を行う場合, $D_{(1,1,8)}$ の分割を用いることが多いが, この分割パターンでは, 良い結果が得られなかった.

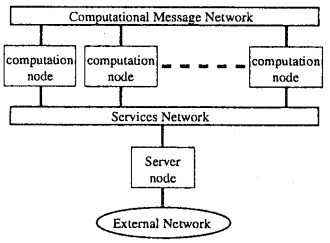

図 2 PC Cluster

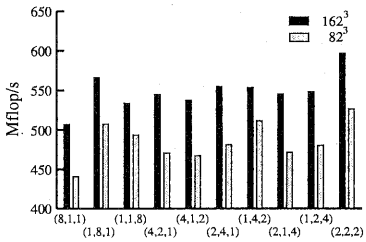

図 3 PC Cluster Performance (Mflop/s)

4.2 RS/6000 SP

RS/6000 SP での Mflop/s の結果を図 4 に示す. PC クラスタと異なり, 経過時間の最低値は, $D_{(8,1,1)}$ で一致したが, 最高値は, 問題量によって異なり一致しなかった. 図の縦軸, 横軸は PC クラスタと同様.

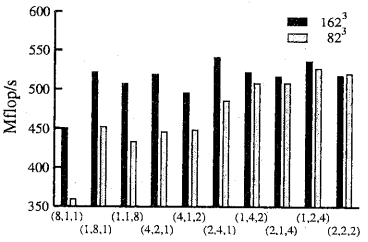

図 4 RS/6000 SP Performance (Mflop/s)

問題量 162^3 では, $D_{(8,1,1)}$ を除いて, 500 Mflop/s 以上と大きな差は見られなかった. 82^3 では, 1 次元分割と i 方向の分割が 4 以上の場合では良い値は得られなかった. また, PC クラスタと異なり, 3 次元分割では最良値が得られなかった.

4.3 E_α の導出

E_α は, 1 PE の実際の処理能力と 1 CPU の理論性能との比である.

$$E_\alpha = \frac{Mflop/s(DS_{(i,j,k)}, 1PE)}{Peak\ Mflop/s(1CPU)} \qquad (6)$$

E_α はハードウェアやコンパイラの能力, またコーディングや計算アルゴリズムによって, 計算サイズと計算領域形状に依存すると考えられる. 各分割パターンでの計算領域形状 ($D_{(i,j,k)}$ における 1 PE の計算領域形状 $DS_{(i,j,k)}$) を 1 PE で計算した計算時間と 1 CPU の理論性能を元に E_α を算出した. 用いたコードと反復回数は, 並列計算と同じである. 図 5 に PC クラスタ, 図 6 に SP の結果を示す. 図は, 縦軸に E_α, 横軸に $DS_{(i,j,k)}$ の (i,j,k) とした. CPU 理論性能は, PC クラスタは 450 Mflop/s, SP は 332 Mflop/s とした.

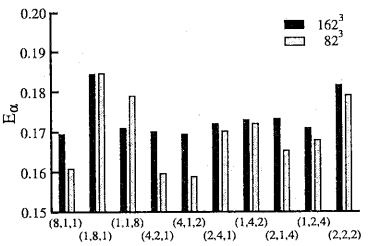

図 5 PC Cluster E_α

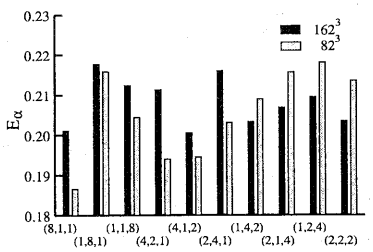

図 6 RS/6000 SP E_α

実効性能は, PC クラスタでは CPU 理論性能の平均 0.175 程度, SP では平均 0.203 程度が得られた. また, 分割パターンによって, PC クラスタは, 6.8 ~ 11.6 Mflop/s 程度, SP では, 5.6 ~ 10.5 Mflop/s 程度の性能差があった. PC クラスタでは, 問題量を問わず $DS_{(1,8,1)}$ の場合が良い結果が得られた. その他の分割パターンでは, 問題量 162^3 では大きな違いは見られないが, 問題量 82^3 では, $DS_{(8,1,1)}$, $DS_{(4,2,1)}$, $DS_{(4,1,2)}$ が 0.16 程度まで落ちた.

SP では, PC クラスタと異なり分割パターンによる変動が明確に現れた. 良い結果が得られた分割パターンは, 162^3 では $DS_{(1,8,1)}$, $DS_{(1,1,8)}$, $DS_{(4,2,1)}$, $DS_{(2,4,1)}$ のパターンで 0.21 以上の結果が得られた. また, 82^3 では $DS_{(1,8,1)}$, $DS_{(2,1,4)}$, $DS_{(1,2,4)}$, $DS_{(2,2,2)}$ で 0.21 以上の結果が得られた.

4.4 C_α の影響

通信における付加時間 C_α の影響を式 (3), (4), (5) と実測値から導いた. C_α が並列処理経過時間に占める割合を図 7 に PC クラスタ, 図 8 に SP の値を示す.

$$C_\alpha \text{ Ratio} = 1 - \frac{Mflop/s_{(C_\alpha=0)}}{Mflop/s_{measure}} \qquad (7)$$

この時 f には, E_α という補正値が掛かるため理論値を用い, b, l には Ping-Pong ベンチマーク [9] から得られた実測値を用いた. 値を表 3 に示す.

表 3 Calcurate Parameter

	PC Cluster	RS/6000 SP
Bandwith(MByte/s)	10.6	82.4
Startup Time(μs)	161.9	106.8

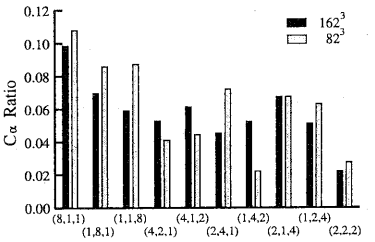

図 7 PC Cluster C_α Ratio

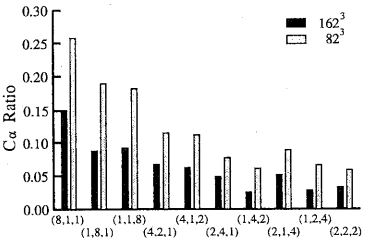

図 8 RS/6000 SP C_α Ratio

ここでは, 分割パターンによるデータ通信時間を一定とした. PC クラスタでは, 0.02 ~ 0.11 程度,

SP では, 0.03 ～ 0.26 程度になった. 1 次元分割では C_α に大きな影響があるが, 計算量が多い場合 $D_{(8,1,1)}$ と $D_{(2,2,2)}$ を除いて大きな差は見られない. 問題量 162^3 で $D_{(2,2,2)}$ が他と比べ小さい値を示し, 82^3 で $D_{(1,4,2)}$ が最小値となった. SP では, それぞれ $D_{(1,4,2)}$, $D_{(2,2,2)}$ が最小値となったが $D_{(1,4,2)}$, $D_{(1,2,4)}$, $D_{(2,2,2)}$ が同程度の良い値を示した.

PC クラスタでは, 問題量が大きくなっても, C_α の割り合いは大きく変化しなかった. 問題量が 8 倍増加した場合, 経過時間に占める C_α の割合は下がるはずであるが, 82^3 から 162^3 という問題量の変化では, ほとんど下がらず, 逆に上がった場合が見られた. SP では, 問題量の増加によって C_α の占める割合が低下したが PC クラスタのように $D_{(2,2,2)}$ が特に低い値を示さなかった.

5. 考　　察

図 5, 6 から計算性能を見る限り, PC クラスタ, SP ともに 1 次元分割である $D_{(1,8,1)}$ の分割を行った場合にかなり良い性能が得られた. 通信処理時間が無限小であれば, 1 次元分割 $D_{(1,8,1)}$ で分割を行っておけば, 分割に手間をかけず良い性能が得られる. 実際には図 3, 4 に見られるように, $D_{(1,8,1)}$ の分割も悪い結果ではないが, 1 次元分割のためデータ通信量の多さとそれに伴う C_α がかなり影響しているものと考えられる. しかし, 1 次元分割を除くと E_α と並列性能の変動はほぼ一致した. そのため, 並列性能は E_α の影響が強く, 1 次元分割のように通信量が多い場合を除いて, 通信性能の影響は少ないと考えられる. 特に PC クラスタでの $D_{(2,2,2)}$ の分割パターンは, $DS_{(2,2,2)}$ の E_α の性能と C_α の影響の低さのため高性能が得られたと考えられる.

図 7, 8 から PC クラスタの分割パターン $D_{(2,2,2)}$ の通信付加時間 C_α は, 他の場合に比べ良い結果が得られた. 通信の処理に要する付加時間の影響を考察すると $D_{(2,2,2)}$ 分割では, 各方向 2 ブロックとなり 1 PE の通信処理において, 送信だけ受信だけの状態となるため, 通信効率が落ちないことが考えられる. そのため, 1 PE が送受信状態となる 1, 2 次元の場合を見ると, $D_{(2,2,2)}$ の場合より C_α は 0.03 以上大きくなった. PE 数 8 程度の分割では, 必ず上記の状態となるため考慮すべき問題である. また, SP では PC クラスタのような現象は見られず, i 方向で分割しない 2 次元分割で十分な性能が得られた. このように, 分割パターンによる性能への影響が大きい場合, はっきりと現れるため, C_α の影響をある程度正確に把握できる.

PC クラスタで問題量 162^3, 82^3 の領域分割の分割パターンを決めるには, 分割パターンによる通信の影響もはっきりと現れるため, 分割パターンの決定はある程度可能であるため, 分割ブロックを 1 PE で計算し, その性能の最も良いパターンを選定すればほぼ最適値が得られると考えられる.

6. おわりに

3 次元ポアソン方程式を静的に領域分割を行って解く場合の分割パターンによる性能への影響を調べた. その結果, PC クラスタで 82^3, 162^3 といった問題量を解く場合では, 通信付加時間の影響をある程度把握し, 領域分割法で得られる 1 PE の計算領域形状を 1 PE で解いた場合の最良値を用いれば, 並列実行した場合の結果が最良であることが分かった. 今後, この結果がその他の並列計算機にも適用が可能かどうか, また C_α の影響を検討したいと考えている.

参 考 文 献

1) 太田高志, 清水大志: PC クラスター環境による並列流体力学計算, 日本流体力学会'99 年会講演論文集, pp371-372, 1999
2) 木村俊哉: 圧縮性流体コードを用いた異機種並列計算機の性能評価, JAERI-Data/Code 97-008
3) http://w3cic.riken.go.jp/HPC/HimenoBMT
4) D. Bailey, T. Harris, W. Saphir, R. Van der Wijngaart, A. Woo, M. Yarrow : *The NAS Parallel Benchmarks 2.0*, NAS Applied Research Branch Report RNR-95-020, 1995.
5) M. Yarrow and R. Van der Wijngaart : *Communication Improvement for the LU NAS Parallel Benchmark: A Model for Efficient Parallel Relaxation Schemes*, NAS Technical Report NAS-97-032, 1997.
6) 折居茂夫: 並列処理数値シミュレーションのためのスケーラビリティ検討方法, HPC58-5, pp27-32 1991
7) T. Sterling, J. Salmon, D. Becker and D. Savarese : *How to Build a Beowulf*, The MIT Press, 1999
8) D. Ridge, D. Becker, P. Merkey and T. Sterling : *Beowulf: Harnessing the Power of Parallelism in a Pile-of PCs*, IEEE Aerospace, 1997
9) R. Hockney, M. Berry: *PARKBENCH Committee: Report-1, Public International Benchmakrs for Parallel Computers*