伸縮性領域を用いた並列適応格子法に関する研究

古山彰一[†] 松澤照男[††]

伸縮性領域を用いることで，適応格子法に関する並列計算手法を提案した．本手法を用いることにより，通信量コストを低く抑え，かつ，負荷分散を良好に行うことができた．その結果，一般的に並列計算が難しいこの問題に対して，32 PE 利用時に 73%の高い並列化効率を得ることができた．

Parallel Adaptive Mesh Refinement Solver Using Elastic Domain Decomposition

Shoichi Furuyama[†] and Teruo Matsuzawa[††]

Elastic Domain Decomposition Method (DDM) for a parallel Adaptive Mesh Refinement (AMR) Solver is described. To use this method saving a message passing cost, a good load balancing, and a high parallel performances (73% on 32 PEs) were achieved.

1. はじめに

本研究では，適応格子法（Adaptive Mesh Refinement method: AMR 法）に関する効果的な並列計算法として，伸縮性領域を用いた動的負荷分散法を提案する．

適応格子法は 1980 年代前半に Berger ら[1]によって提唱された手法である．この手法は，速度差，密度勾配などに応じて局所的に格子を細分化する手法で，一般的に局所的に高い精度が必要とされる数値流体力学分野では非常に効果的な手法として話題にのぼることも多い．近年の大型計算機の発達により，計算領域全体に細分化格子を作成し計算することも可能となったが，さらなる計算精度の向上や，計算時間の削減といった観点から，ハードウェアの発展がなされたとしても重要であり，かつ，今後ますます需要の高まる手法であると考えられる[2]．

適応格子法は，大きく 2 種類の手法に分けることができる．1 つは初期格子が正方もしくは立方格子で構成される構造格子であるものと，もう 1 つは，それが非構造格子で構成されるものである．本研究では，格子生成の容易さや，計算手法の簡潔さ，さらには，計算コストの低さから，ここ数年の数値流体力学分野で見直されつつある，正方，もしくは立方格子をベースとした初期格子が構造格子である手法に対して，適応格子法を採用した計算手法を取り上げる．このような適応格子法の主な研究としては以下のものがあげられる．Zeeuw ら[3]は，初期格子に正方格子を利用し，さらにカットセルを用いることで，物体近傍での物体表現を精密に行い，航空機周りなどの流れ場解析を行った．Wu ら[4]は Navier-Stokes 方程式の解法として適応格子法を採用しさらにマルチグリッド法を適用することで，非圧縮性粘性流れ場の解析を行った．Wang[5]は物体適合格子とのハイブリッドな格子系を用いることで，非圧縮性粘性流体に関する複雑形状物体周りの流れ場解析を行った．適応格子法は地形などの複雑形状を効率良く取り込み計算することが可能であるため，湖沼の水の流れ場解析などへの応用も試みられている（Borthwick ら[7]）．このように，様々な分野で流体の特徴に拘束されず適用されている．また，適応格子法の高精度・高速度であるという性質を生かし，太陽風のリアルタイムの予測にも使われつつある（Groth ら[6]）．

一方，数値流体力学分野において，並列計算に関する研究が急速に普及した．ハードウェア性能の向上とともに，さらなる大規模計算が可能となったが，計算をさらに高速に，かつ，さらに複雑な形状の物体が計算できるように，前述のような初期格子を構造格

† 富山商船高等専門学校情報工学科
Department of Information Engineering, Toyama National College of Maritime Technology
†† 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 (JAIST)

子とした場合の適応格子法に関する並列計算の研究もなされている．Calder ら[8]の PARAMESH を利用した FLASH，Parashar らの DAGH[9]，Quinlan の AMR++[10]，Kohn らの SAMRAI[11]などが代表的なものである．これらは，負荷分散を良好に行うために，複数の同細分化レベルの格子をブロックとしてまとめ，その領域内の格子に関してはすべて細分化を行うブロックという概念を用いて並列計算を行う手法である．上記の 4 つは，並列計算アルゴリズムとしてはほとんど同様の手法をとっており，現在行われている適応格子法に関する並列計算の研究はこれらをベースとしたものがほとんどである．これらは，ブロックを各 CPU に分散させるため，負荷分散を厳密に行うことが可能である．しかしながら，ブロックサイズをどの程度にとるべきか，または，負荷分散は厳密に行えるが，ブロックを増やせばそれにともない通信も増え，通信コストが非常に高くなるといった問題がある．さらに，適応格子法の持つ，格子単位での柔軟な格子細分化が制限されるために，不要な格子を生成してしまうことが問題である．特に 3 次元の流れ場についてブロックベースの手法を用いた場合，非常に多くの格子細分化を作成することとなる．

このような背景を基に，本研究では，既存の適応格子法を，格子ベースでの細分化という特性を失わずに，その柔軟性を維持したまま並列計算機上に実装することを検討する．また，近年普及している PC クラスタなどを考えた場合，通信コストが低い並列計算アルゴリズムを考える必要があり，通信コストを比較的低く抑えることが可能な並列計算手法である「伸縮性領域」を提案し，そのパフォーマンス評価を行う．

2. 手法（非並列部）

はじめに，非並列計算部について，流体解析手法と，適応格子法について言及する．

2.1 流体解析手法

解析する対象の流体は，圧縮性非粘性流体とし，有限体積形式の以下の Euler 方程式を解く．

$$\frac{d\mathbf{U}}{dt} = -\frac{1}{A}\sum_{faces}\left(\mathbf{F}\Delta y - \mathbf{G}\Delta x\right),$$
$$\mathbf{U} = (\rho,\ \rho u,\ \rho v,\ \rho E)^t,$$
$$\mathbf{F} = \left(\rho u,\ \rho u^2 + p,\ \rho uv,\ \rho uH\right)^t,$$
$$\mathbf{G} = \left(\rho v,\ \rho uv,\ \rho v^2 + p,\ \rho vH\right)^t. \qquad (1)$$

ここで，A はセルの面積を表し，Δx，Δy はセルの辺長を表す．ρ，u，v，E，H はそれぞれ，密度，速度成分 2 つ，エネルギー，エンタルピーを表す．また，周積分方向は半時計回りとする．

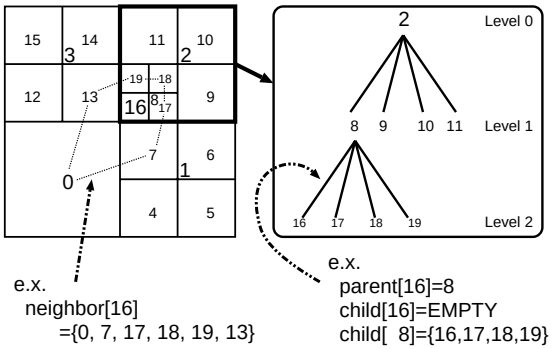

図 1 データ構造
Fig. 1 Data structure.

解析手法は，Zeeuw ら[3]の手法に従い，linear reconstruction method，近似リーマン解法，multi-stage time-stepping scheme を利用した．multi-stage time-stepping scheme については 5 段階のものを利用し，時間積分については，local time-stepping を用いた．

2.2 適応格子法

適応格子法を議論するにあたり，次の 2 点について言及する．

(1) 格子細分化指針（格子を細分化する場所）
(2) データ構造（データ配置，格子データの追加・削除について）

これらについて順を追って説明をし，この節の最後に，適応格子法を利用した場合の流れ場の様子を掲載する．

2.2.1 格子細分化指針

格子の細分化は，流れ場が定常状態（残差が収束）になったときに行うこととし，格子の細分化は，隣接格子との速度差が大きい格子について行うこととした．速度差の基準は，計算領域全体において隣接格子と最も速度差が大きいものとし，この値の 85%を超える速度差を持つ格子対に対して細分化フラグを立てる．格子細分化に際して，隣接格子とは細分化レベルの違いを 1 しか許さないこととする．また格子は等面積で 4 分割される．細分化された格子のデータは，親格子の値を用いて内挿を行うことで導出する．内挿の仕方については，Linear Reconstruction Method[3]に従う．

2.2.2 データ構造

適応格子法を実現するために，1 セルあたり合計 46 個（整数型 21 個，実数型 25 個）の情報を格納する[12]．

細分化格子は，1 つの格子から 4 つ生成されるため，データ表現には四分木を用いる．格子の配置とメモリ空間におけるデータの配置を**図 1** に示した．この図で

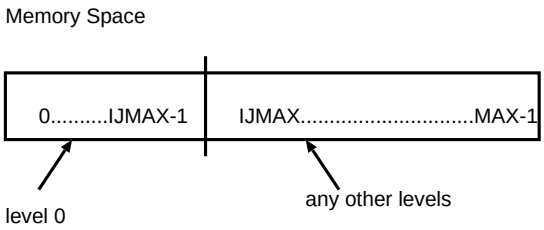

図 2 メモリ空間におけるデータの配置
Fig. 2 Numbering for meshes.

は，“2” とラベリングされた *level* 0 の格子に注目している．“2” は細分化されることにより，{8, 9, 10, 11} の *level* 1 格子に細分化される．さらにこの場合では，細分化格子中の “8” が {16, 17, 18, 19} の *level*2 格子に細分化されている．各格子は，親格子へのポインタ，子供格子へのポインタ，また，隣接格子へのポインタなどの情報を持つ．図 1 では，“16” に関する格子のこれらのポインタについて具体例をあげている．

メモリ空間における格子データの配置は（配列番号のラベリング）は，初期格子（*level* 0）のみ構造的に配置する（**図 2**）．すなわち，初期格子が “$IJMAX$” 個存在する場合，配列番号 [0]〜[$IJMAX-1$] までを構造的に初期格子で埋める．細分化格子については，その後ろ（すなわち [$IJMAX$] 以降）に追加する．これらについては，そのレベルに関係なく，後ろに必要なだけ追加する．また，格子の削除が必要になった場合には，その部分のデータ情報として “$DataEMPTY$” のラベリングをし，追加格子が優先してその部分に挿入されるようにする．追加格子がない場合は，それ以降にある格子を前方に詰めるようにする．つまり，全格子数が “MAX” 個である場合，データは配列番号 [0]〜[$MAX-1$] にすべて収まるようにする．

2.2.3 適応格子法による結果

計算対象は，**図 3** のように，256 × 100 の初期格子で覆われる計算空間（$\Delta x = \Delta y = 0.5$，横 128 × 縦 50）内に，11 × 4 の格子で生成される長方形の物体（5.5 × 2）を置き，左側から風を流し込む場合を取り上げた．物理量の初期値は，$\rho = 1.4$，$u = 2.2$，$v = 0$，$E = 6.875$ とし，計算を行っている間の流入口の境界条件も同じとした．また，CFL 数は 1.1508 で固定した．

図 4 では，この条件での残差の履歴を示した．ここで，収束した時間ステップを基準として（つまり格子の細分化が行われたステップ）計算区間を $T1$〜$T4$ で定義し，この区間については今後の議論で利用する．区間についての詳細は**表 1** に示した．Start，End，Steps，Ratio は，区間の始まりと終わりのステップ，

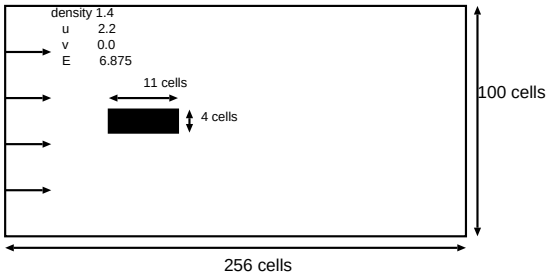

図 3 初期条件
Fig. 3 Initial condition.

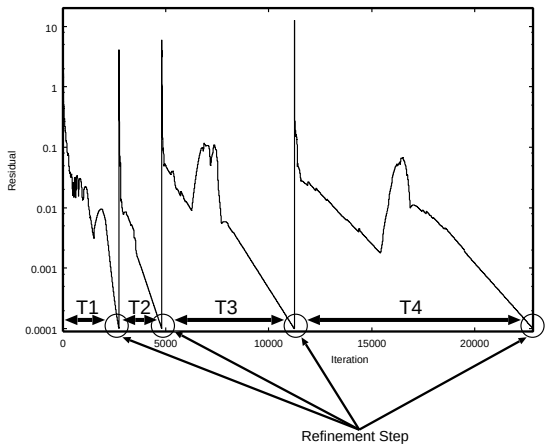

図 4 残差履歴
Fig. 4 Representative residual convergence history.

表 1 計算区間とステップ数
Table 1 Calculation term.

Term	Start		End	Steps	Ratio(%)
T1	0	-	2,722	2,723	11.90
T2	2,723	-	4,801	2,079	9.10
T3	4,802	-	11,238	6,438	28.19
T4	11,239	-	22,840	11,602	50.80

その区間のステップ数，全体の計算ステップ数に占めるその区間のステップ数割合をそれぞれ表す．

図 5，**図 6** では AMR を用いた場合の格子分布と密度分布を示した．AMR を用いることにより，衝撃波がシャープにとらえられ，物体後方での密度分布も詳細にとらえることができた．

3. 手法（並列計算部）

本手法では，分散型の並列計算環境で領域分割法を用いる並列計算法を採用するため，それぞれの PE は，完全にローカルな領域のデータのみ所持することとする．それにともない，それぞれの PE が担当する計算領域の周りに計算バッファを設け，その部分について

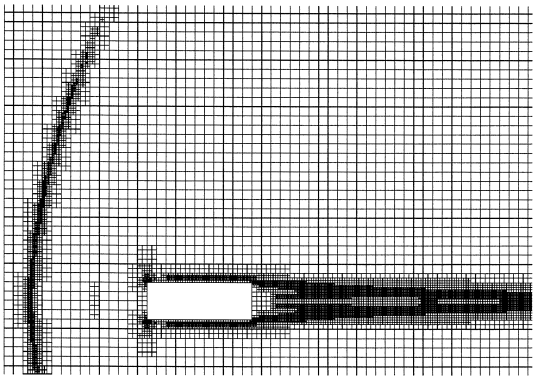

図 5 格子分布図
Fig. 5 Mesh distribution.

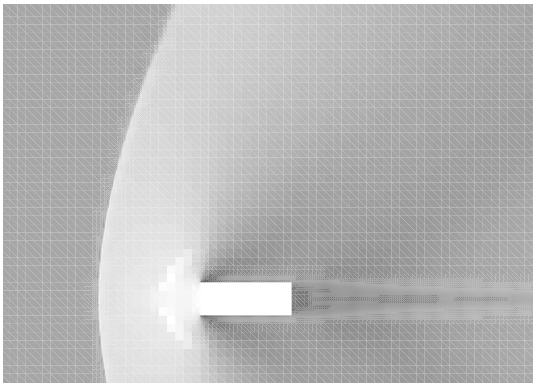

図 6 密度分布図
Fig. 6 Density distribution.

通信を行う必要がある．また，適応格子法に対して領域分割を用いて並列計算を行うには，動的な負荷分散が必要不可欠となる．これらのことより，本章では，
(1) 基本的な通信
(2) 負荷分散法
の 2 つの話題を取り上げる．なお，並列計算に関する通信ライブラリには，MPI を用いた．

3.1 基本的な通信

領域分割を用いて並列計算を行う場合，各 PE の担当領域の端に通信バッファを設け，その部分のデータを隣接 PE から取得する必要がある．本手法では計算領域を *level* 0 格子の辺に沿って $x = const.$ のラインで分割し，左右に *level* 0 格子帯 1 つ分をバッファ領域として設けた（**図 7**）．

それぞれの格子に対するデータは，実際に計算 PE 間で交換が必要なもの（つねに値が更新される物理量など）と，一度決めたら変化がないもの（座標値など）が構造体として格納されているため，通信に必要なものだけをそこから取り出し，*MPI_Pack* を用いてメモリ空間において連続的に配置して，その後，実際にデータを送受信する．

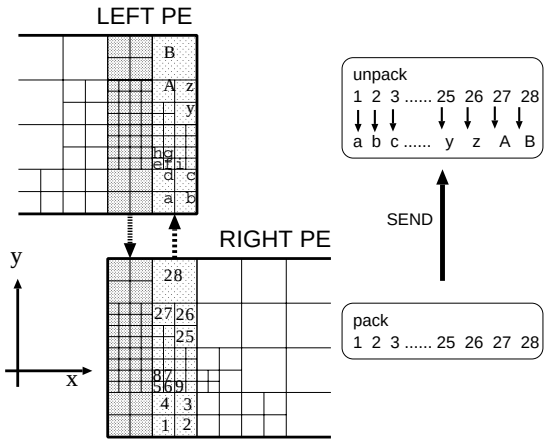

図 7 通信領域のデータ構造
Fig. 7 Message passing: pack & unpack data.

図 7 では，具体的なデータのパックの方法を示した．まずはじめに，LEFT PE の明灰色部分の格子の構成は，RIGHT PE の明灰色部分と同じ格子構成にする．そして，この図の RIGHT PE の明灰色の部分を LEFT PE の明灰色部分に送信する．Pack をしていく順番は *level* 0 格子の単位で考え，その子供を再帰的に Pack していく．Pack されたデータを，LEFT PE では同様の手順で *MPI_Unpack* を用いて Unpack する．

以上の手続きを時間ステップごとに行うことで，領域分割法を利用した基本的な並列計算が可能となる．

3.2 負荷分散法

適応格子法は局所的に細分化格子を生成するために，前述の単純な領域分割法を用いた並列計算だけでは，負荷分散に関して不十分である．そこで，本研究での主題となる伸縮性領域を提案し，これを用いて負荷分散を行う．伸縮性領域とは，各 PE の担当する領域を，伸縮・拡大することで担当格子数の増減を行い，負荷分散を行う手法である（**図 8**）．細分化格子数が増加する場所には，なるべく多くの計算 PE を割り当てるか，割り当てられている PE の担当領域を縮小し，1 次元的に計算領域をする手法が，本研究で提案する伸縮性領域分割法である．

領域決定のプロセスを以下に述べる．**図 9** で示したように，伸縮を行う方向（この図では横方向）に対して *level*0 格子幅での帯を考え，その帯の中に，6，6，15，21，6，6 の格子が存在し，合計で 60 格子が計算空間に存在する状況を考え，これを 2 PE 利用して並

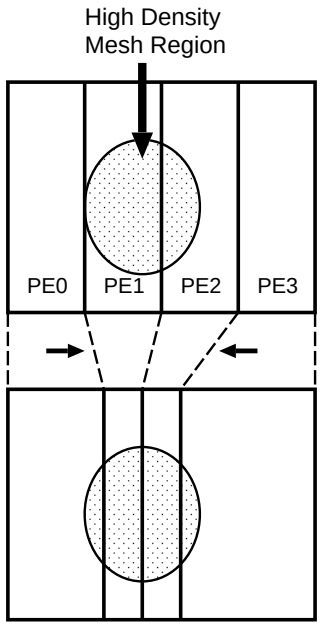

図 8 伸縮性領域
Fig. 8 Elastic domain.

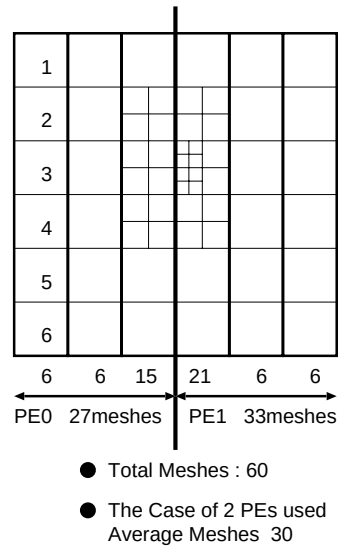

図 9 計算量の定量化
Fig. 9 The mesh in a belt is counted.

列計算を実現すると仮定する．

まずはじめに，それぞれの帯における格子数の情報を各 PE から PE0 に送信する．PE0 では，これらの情報から計算領域内の総格子数を求め，各 PE が担当すべき平均格子数を導出する．図 9 の場合は，平均格子数は 30（総格子数 60，使用 PE 数 2）となる．次に，計算領域の左端の帯から順次格子数をその平均値に達するまで加算していく．ある帯の格子数を加算したときに平均値を超えた格子数になった時点で，その前までの帯を足した時点での格子数と比較する．それが平均格子数に近い方を選択し，その PE の担当領域とする．図 9 の場合，6，6，15，21 と足し合わせた時点で 48 となり，平均格子数の 30 を超える．しかし，21 を足す前での格子数は 27 であり，最後の 21 の帯を足した場合よりも平均格子数に近い値となる．そのため，21 を足す前の帯までを PE0 の計算領域とし，図で示した位置での分割を行う．ここでは 2 PE 利用時のことを具体的に述べたが，PE 数がさらに多い場合には，残りの帯から担当平均格子数を新たに導出し，この手順を繰り返す．この手法により，それぞれの PE が平均値に近い格子数を担当するようになる．

次に，上記手法により，新たに担当することになった計算領域のデータの取得について述べる．本手法は，分散型の並列計算を前提としているため，各 PE は計算担当領域のデータしか保持していない．そのため，上記手法により得られた各 PE の担当する領域が，過去の担当領域と変動があった場合，その差分領域におけるデータを，通信によって他 PE より得ることが必要である（計算領域が縮小される場合は，他 PE にデータを与えることが必要）．また本手法では $level0$ の初期格子に関しては格子構造を維持するため，領域変動があった場合にその構造をどのように維持するかも問題となる．このことを以下に述べる．それぞれの PE は，新たに担当する計算領域と，これまで担当してきた計算領域との差分が図 9 で示した $level0$ で構成される帯でいくつあるのか情報を取得する．これは，PE0 で新領域を決定した際に，各 PE は，PE0 からその情報を取得することとする．そして，その値に応じて，現領域内に残される $level0$ 格子をメモリ空間内で移動する．

具体的な例を**図 10**，**図 11** で示した．ここでは，横方向に $level0$ 格子 3 つで構成されていた領域（図 10 中，OLD Domain）が，上記のように領域の伸縮を行うことで，左に 2 つ，右に 1 つの領域を追加される場合（図 10 中，NEW Domain）を仮定している．この場合，たとえば図 10 で “0” とラベリングされていた格子は “1” と再ラベリングされ，“11” は “21” と再ラベリングが行われる．他の $level0$ の格子に関しても同様に再ラベリングを行い，格子構造の再構築を行う．それにともない，図 11 で示したように $level1$ 以上の細分化格子に関しては，追加される $level0$ の格子数分だけメモリ空間において後方に移動させる．さらに，新たに計算対象となる $level0$ 以外の細分化格子は，メモリ後方に順次追加していく．また，この移

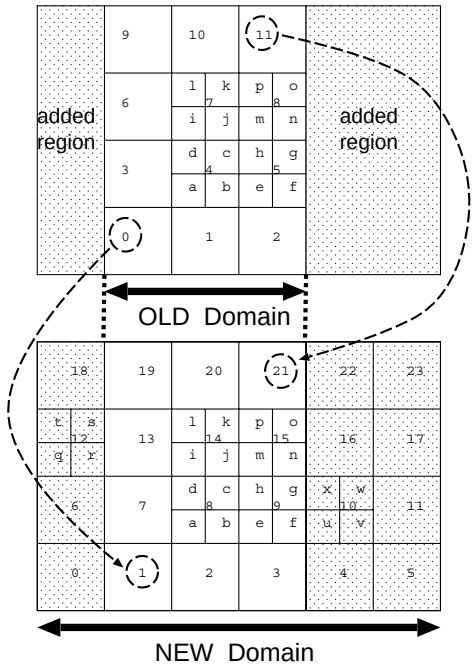

図 10 データの再配置
Fig. 10 Data reconstruction.

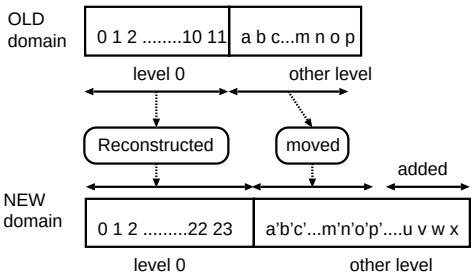

図 11 メモリ空間におけるデータの再配置
Fig. 11 Elastic domain decomposition, memory space.

動にともない，親子関係，隣接格子のポインタ，領域の変動による各格子の状態の変更（計算領域境界に位置する，などのラベリング）も修正する．このようにして既存の領域を新領域との差分領域で干渉をしないように移動してから，差分領域を各 PE どうしで通信しあい，新領域を作成する．その際，旧領域と新領域の間に最低 2 格子分は共有する制限を設けて，データの通信を，隣接した計算領域を担当している PE（一般的には 2 つの PE）とのみ行うようにする．

以上の手法と，前節の「基本的な通信」を併用して，負荷分散が行われた領域で並列計算が可能となる．

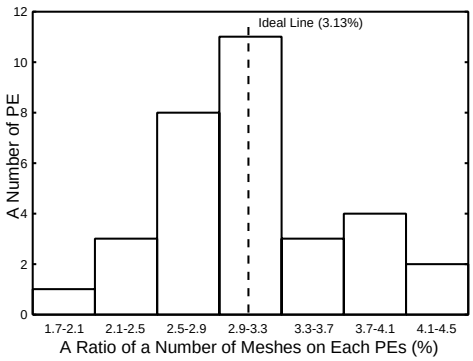

図 12 負荷分布
Fig. 12 Load distribution.

4. 結　　果

4.1 環　　境

並列計算機として Cray-T3E 1200E を利用した．このマシンは，3 次元トーラス相互結合網で結合した分散メモリ型の並列計算機で，128 個の PE ノードが 650 MB/s の転送能力を持つ結合リンクで 3 次元トーラスに結合されている．また各 PE ノードは，Compaq 社の Alpha 21164 600 MHz を搭載している．

4.2 解 析 対 象

パフォーマンス評価は，2.2.3 項で述べた解析対象と同条件の下で行う．なお，各 PE の担当する計算領域の初期形状は，伸縮方向について *level* 0 格子線で等間隔に分割したものとする．

4.3 負 荷 分 散

図 12 は，32 PE 利用時の負荷分散状況を示した．このグラフは，計算区間 $T4$ において，計算領域全体の格子数（本計算条件では 56,416）に対する担当格子数の割合と，その PE 数を示している．このグラフから，1 PE における担当格子数割合は 1.7〜4.5%の範囲に収まっていることが分かる．

4.4 通 信 時 間

図 13 に 32 PE を使用した場合の全体の計算時間に対する通信に費やした時間（待ち時間も含む）の割合とその PE 数を載せた．またこの通信時間の割合は，動的負荷分散にかかわる通信も含んでいる．通信時間割合が 4〜12%の PE が 32 PE 中 27 であり，ほとんどの PE は 10%程度に通信時間割合を抑えており，最大のものでも 18%程度であることが分かる．また，グラフとしては掲載していないが，動的負荷分散にかかわるコストは，全計算時間の 0.16%に抑えられている．

4.5 速度向上比

図 14 では速度向上比の結果を載せた．横軸が PE

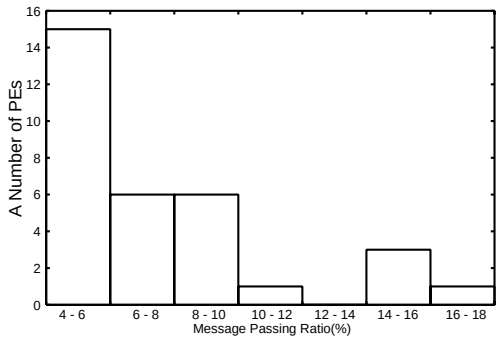

図 13 通信時間割合
Fig. 13 Ratio of message passing.

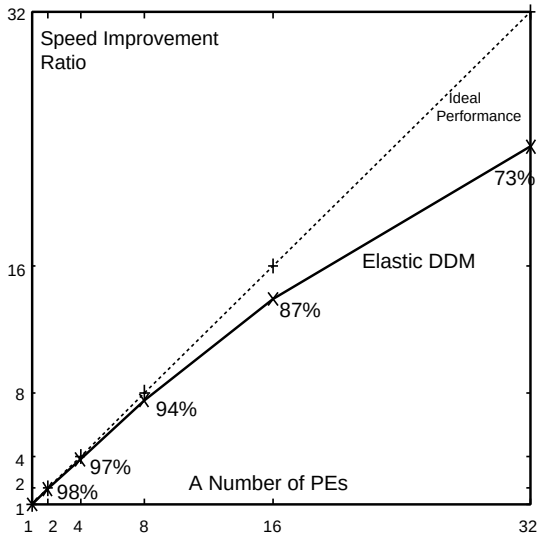

図 14 速度向上比
Fig. 14 Speed improvement ratio (cray T3E).

数で，縦軸が速度向上比を示している．点線で理想パフォーマンスを示し，実線で実測値を示した．また，各 PE 数において，理想パフォーマンスの何%のパフォーマンスが得られたかを示した．

5. 議　　論

負荷分散について議論を行う．32 PE を用いて並列計算を行う場合，1 PE あたりの平均担当格子数割合は約 3.13%となり，最大負荷を持つものがこの値に近ければ近いほど負荷分散が良好に行われているといえる．図 12 をみると，今回は，それが 4.1%程度である結果が得られた．

通信コストについては，図 13 から分かるように，最大でも 18%に収まっている．ここで通信時間として定義しているのは，計算領域境界のデータ交換に費やされる時間と領域伸縮に費やす時間であり，特に領域伸縮に費やす時間の割合は全計算時間の 0.16%にとどめている．そのため，ここで示した通信コストは，ほとんどが境界領域データの通信にかかる時間であるといえる．実際に，図 13 で通信割合が高い PE は，担当計算領域の境界に細分化格子を多数持っているものである．このことから，今後は負荷分散と通信量を考慮した計算領域境界を検討する必要がある．

速度向上比をみると，2，4，8，16，32 PE 利用時で，それぞれ 98，97，94，87，73%の高い並列化効率を得ることができた．負荷分散の点で理想的な値からややずれていたものが存在し，さらに，通信コストが 18%の PE も存在したが，速度向上比の評価の点でそれらが許容の範囲であることがこの結果から理解できる．

最後に計算対象について議論する．本手法では，領域の伸縮により負荷分散を行うために，領域変動の自由度が問題となる．領域の伸縮方向に，使用 PE に対して十分に大きい計算領域が設定されているような問題では，領域伸縮の自由度が増え，負荷分散を行いやすくなると考えられる．近年，生体力学の分野で血管内流れが注目されているが，このような管内流れ場解析では，1 方向に対して長い計算領域をとることが必要であり，本手法が高い能力を発揮できる計算対象であると考えられる．

またさらに，非定常計算のもとでも本手法は有効であると考えられる．適応格子法は，流れ場の物理量に応じて細分化格子を生成するため，格子の生成は，当然物理現象に応じたものとなる．非定常問題の場合は，定常問題と違い，格子の生成と削除が計算領域内のいたるところで行われることとなるが，格子は時間の進行とともに徐々に生成されて，それが移動するのが一般的である．本手法では，格子の生成具合にあわせて各 PE の担当する計算領域の変動を行うため，物理現象への変動にシンクロするような問題に対してはきわめて有効であると考えられる．またさらに，前述したように領域伸縮に関する通信コストは非常に低く抑えられているため，非定常問題を扱うことによってこの回数が多くなったとしてもそのコストを低く保ったまま計算が可能であると考えられる．これらのことから非定常問題に対しても十分に対応可能であると考えられ，本手法は様々な場面で適用可能であると考えられる．

6. おわりに

伸縮性領域分割法を提案し，適応格子法に関する並列計算手法に関する考察を行った．本手法の特徴とし

て，以下の点をあげることができた．

- 適応格子法の格子ベースでの細分化という柔軟性をまったく失っていない．
- 1 方向に対して長い計算対象に特に有効な手法．
- 通信コストが比較的小さい．
- 動的負荷分散に対するコストが小さいため，非定常計算にも有効．

今後は，3 次元計算について本手法を適用し，その能力を検討する．

参 考 文 献

1) Berger, M.J. and Oliger, J.: Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations, *J. of Comp. Phys.*, Vol.53, p.484 (1984).
2) 青木：複雑形状の複雑でない計算法，情報処理, Vol.42, No.6, p.550 (2001).
3) Zeeuw, D.D. and Powell, K.G.: An Adaptively Refined Cartesian Mesh Solver for the Euler Equations, *J. of Comp. Phys.*, Vol.104, p.56 (1993).
4) Wu, J., Ritzdorf, H., Oosterlee, K., Steckel, B. and Schuller, A.: Adaptive Parallel Multigrid Solution of 2D Incompressible Navier-Stokes Equation, *Int. J. Num. Meth. Fluids*, Vol.24, p.875 (1997).
5) Wang, Z.J.: A Quadtree-based Adaptive Cartesian/Quad Grid Flow Solver for Navier-Stokes Equations, *Comp. & Fluids*, Vol.27, No.4, p.529 (1998).
6) Groth, C.P.T., De Zeeuw, D.L., Gombosi, T.I. and Powell, K.G.: A Parallel Adaptive 3D MHD Scheme for Modeling Coronal and Solar Wind Plasma Flows, *Space Sci. Rev.*, Vol.87, p.193 (1999).
7) Borthwick, A.G.L., Leon, S.C. and Jozsa, J.: The Shallow Flow Equations Solved on Adaptive Quadtree Grids, *Int. J. Numer. Meth. Fluids*, Vol.37, p.691 (2001).
8) Calder, A.C., Curtis, B.B., Dursi, L.J., et al.: High-Performance Reactive Fluid Flow Simulations Using Adaptive Mesh Refinement on Thousands of Processors, *SC2000* (2000).
9) Parashar, M., et al.: DAGH: Data-Management for Parallel Adaptive Mesh-Refinement Techniques. http://www.caip.rutgers.edu/ parashar/DAGH/
10) Philip, B., et al.: AMR++: Object-Oriented Parallel Adaptive Mesh Refinement, UCRL-ID-137373, LLNL.
11) SAMRAI: Structured Adaptive Mesh Refinement Application Infrastructure. http://www.llnl.gov/CASC/SAMRAI/
12) 古山，松澤：正方格子を用いた適応格子方の並列計算に関する考察，第 15 回数値流体力学シンポ講演論文要旨集&CDROM (2001).

(平成 14 年 6 月 7 日受付)
(平成 14 年 9 月 26 日採録)

古山 彰一（正会員）

1971 年生．1994 年岩手大学教育学部卒業．1996 年北陸先端科学技術大学院大学博士前期課程（情報科学）修了．1999 年同大学院博士後期課程修了．同年より富山商船高等専門学校情報工学科助手．現在に至る．主に数値流体解析に関する並列計算アルゴリズムの研究に従事．博士（情報科学）．日本機械学会，日本流体力学会各会員．

松澤 照男（正会員）

1948 年生．1973 年信州大学大学院工学研究科修士課程修了．同年同大学医学部助手．1986 年沼津工業高等専門学校助教授．1991 年北陸先端科学技術大学院大学情報科学センター助教授．1995 年同教授．現在に至る．数値流体力学における並列計算の研究に従事．医学博士．日本機械学会，日本流体力学会各会員．