［共同研究成果］

Building-Cube Method に基づく解適合格子細分化と動的負荷分散

三坂孝志[1]，佐々木大輔[2]，大林茂[3]
[1]東北大学 学際科学フロンティア研究所
[2]金沢工業大学 工学部
[3]東北大学 流体科学研究所

分散メモリ型計算機において解適合格子細分化と動的負荷分散を行うためのフレームワークを Building-Cube Method（BCM）に基づき構築した．解適合格子細分化を行うと特定の MPI ランクに計算負荷が集中する場合があるが，動的負荷分散と組み合わせることでこれを解消する．負荷分散は各 MPI ランクにおける計算時間と一対一通信の時間をキューブ再分配の指標をすることで実現した．ラプラス方程式を用いて，この BCM フレームワークの解適合格子細分化や動的負荷分散に関する検証を行った．その後，上記の BCM フレームワークを 3 次元非圧縮性ナビエ・ストークス方程式に適用した．流れ場の解適合格子細分化では，無次元壁座標（y^+）を細分化指標とした解析を行い，その有効性を確認した．

1. はじめに

数値流体力学（Computational Fluid Dynamics, CFD）計算の大規模化が進み，計算効率や複雑形状に対する格子生成の容易さから直交格子法が再び注目を集めている．Building-Cube Method（BCM）は，キューブと呼ばれる立方体のマルチブロック構造直交格子を利用した CFD ソルバーのフレームワークである[1]．計算領域はサイズの異なるキューブで分割されるが，隣り合うキューブのサイズ比は，1:2，1:1，2:1 のいずれかである．各キューブは同一の格子点数を含んでいるため，格子解像度はキューブサイズによって決定される．格子生成においては，解析対象の形状に合わせた格子細分化が行われる．この BCM フレームワークを用いた各種 CFD ソルバーの開発は，中橋らのグループによって進められてきた[1-5]．GUI を含む格子生成ソフトウェアの整備は石田らによって行われた[2]．京コンピュータを含む大規模計算環境への展開も大西らによって進められている[3]．大規模計算結果データの効率的な処理を目的として，データ圧縮技術も開発されてきた[4]．BCM で曲面形状まわり流れの解析を行う方法に関しても，グリッドレス法とのカップリング[5]などいくつかの取り組みがあるが，主に埋め込み境界法（Immersed Boundary Method, IBM）を用いた方法が研究されてきた．

BCM を提案した中橋らの論文でも触れられているように，BCM のフレームワークは大規模計算に有効であり，実際に京コンピュータを用いた解析も行われてきている．BCM の残る課題として，高レイノルズ数流れにおける IBM の高精度化や動的なキューブデータの制御による動的負荷分散が挙げられる．後者に関連して，解適合格子細分化の実装も必要である．動的な格子細分化は，初期条件からの流れ計算が収束するまでの時間を短縮する効果もある．

本研究では，分散メモリ型計算機において解適合格子細分化と動的負荷分散を行うためのフレームワークを BCM に基づき構築する．解適合格子細分化を行うと特定の MPI ランクに計算負荷が集中する場合があるが，動的負荷分散と組み合わせることでこれを解消する方針とした．ラプラス方程式を用いてこの BCM フレームワークの解適合格子細分化や動的負荷分散に関する検証を行い，その後，3 次元非圧縮ナビエ・ストークス方程式に適用した．流れ場の解適合格子細分化では，無次元壁座標（y^+）を格子細分化指標とした解析を行った．

2. 数値計算手法

BCM は格子生成，高次精度スキームの組み込み，および，後処理の簡素化のために等間隔の直交格子を用いている．さらに，計算アルゴリズムを複雑化させることなく，格子を複雑形状および局所流れ場に適合させるために，ブロック構造格子を採用している．ブロック構造格子はデータ構造としても利点がある．計算領域はキューブと呼ばれる立方体ブロックに分割されるが，格子解像度がキューブサイズで決まり，また，キューブを用いた並列処理が効率的に行われるように，各キューブは同じ数の格子点を含んでいる．BCM の基本となるデータ構造に関しては過去の文献を参照されたい[1]．本研究では BCM フレームワークを用い，MPI ランク間でキューブ情報を動的に移動する方法を導入した．基本となるのは空間充填曲線（Space Filling Curve, SFC）による 3 次元キューブ位置情報の一次元並べ替えである．空間充填曲線は多くの分野で用いられており，直交格子 CFD においても多く利用されている[6]．本研究では同様の手法を BCM のキューブに適用する．図 1 に 2 次元の空間充填曲線と 1 次元マッピングの例を示す．図 1 のように空間上の座標値（ここでは各キューブの中心座標）が与えられれば，一意に一次元順列を生成することができる．この空間充填曲線は，一次元順列における隣接と元の空間での隣接がある程度両立するので，領域分割にも用いられる．図 1 に示すように，一次元順列に沿ってキューブを分割することによって，元の空間における領域分割が可能となる．本研究では，この分割を用いて各MPIランクにキューブを分配する．

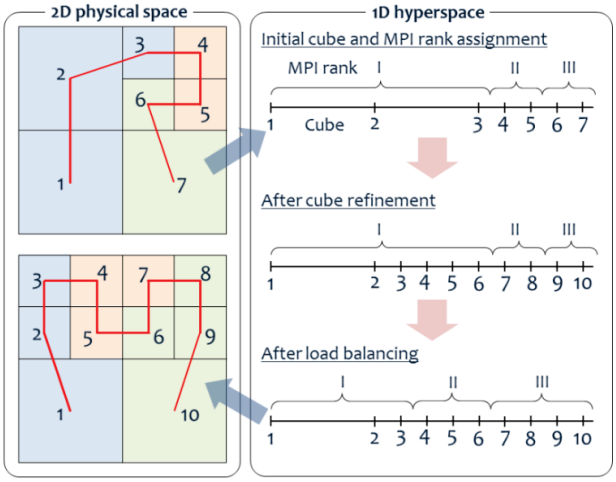

図 1　空間充填曲線と一次元マッピングの例．

この空間充填曲線を用いると，動的な負荷分散は各MPIランクで経過時間を計測し，そのばらつきに応じて一次元順列上で分割点を移動することで実装することができる．分割点が決まったら，MPIランク間の必要なデータ交換を行えばよい．図 2 に 3 並列の場合のキューブ情報移動の例を示す．あるステップ数の計算を行った後の各 MPI ランクにおける演算時間と一対一通信時間の和をt_i $(i = 1, \cdots, N)$とする．ここでNは総 MPI ランク数であり，今の場合は$N = 3$ である．式(1)のようにr_iを定義する．

$$r_i = \frac{1/t_i}{\sum_{j=1}^{N}(1/t_j)}. \tag{1}$$

ここでr_iは各 MPIランクの経過時間の逆数を，その和で正規化したものである．このr_iを用いて，空間充填曲線上の領域分割位置を以下のように定義する．

$$ec_i = n_{cube,all}\frac{\sum_{k=1}^{i} r_k}{\sum_{j=1}^{N} r_j} \tag{2}$$

ここで$n_{cube,all}$は総キューブ数である．図 2 には，負荷分散前に MPI ランク 2 の経過時間t_2が，MPI ランク 1 および 3 よりも大きい状況を示している．このとき，各 MPI ランクのキューブ数は一定である．負荷分散後には式(2)で計算した位置に分割点が移動し，このとき，MPI ランク 2 のキューブ数が少なくなり，各 MPI ランクの経過時間が近い値となる．上記のような処理を，例えば，数百タイムステップごとに繰り返すことで，各 MPI ランクの経過時間が均等になる．

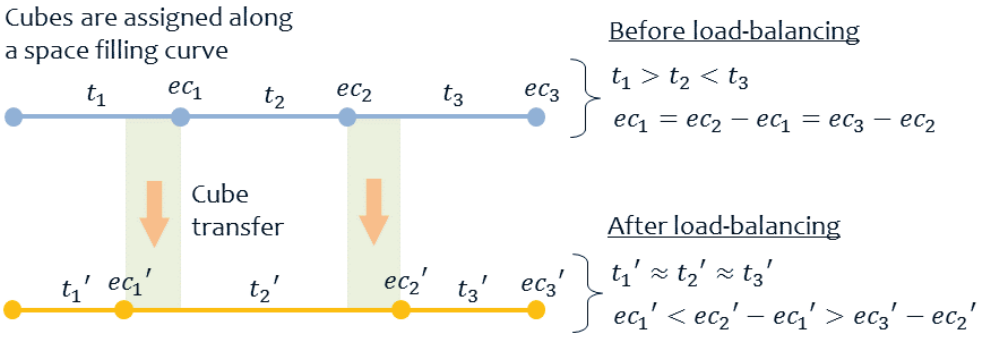

図 2　空間充填曲線を用いた動的負荷分散の例．

3. ラプラス方程式を用いた動的負荷分散および解適合格子細分化の検討

前節で説明した動的負荷分散や格子細分化に関する検討を行うために，ここでは 3 次元ラプラス方程式を BCM フレームワークにより解く．単位辺長さを持つ立方体領域を考え，上面で$\phi_s = 1$，その他の境界で$\phi_s = 0$の境界条件を課すと，解析的に式(3)に示す級数解を求めることができる．

$$\phi_s(x,y,z) = \frac{16}{\pi^2}\sum_{m,n=1}^{50}\frac{\sinh\gamma_{mn}z}{mn\sinh\gamma_{mn}}\sin m\pi x\sin n\pi y, \tag{3}$$

ここで，$\gamma_{mn} = \sqrt{(m\pi)^2 + (n\pi)^2}$である．式(3)は無限項の和をとるべきものであるが，実際に解を得るに当たり 50 までの和とした．図 3 に計算領域と中心断面における級数解の分布を示す．一方で，数値解は BCM フレームワークを用い，キューブ内では 2 次精度中心差分で離散化した式を SOR 法により反復的に解く．級数解は数値解との比較のために用いる．

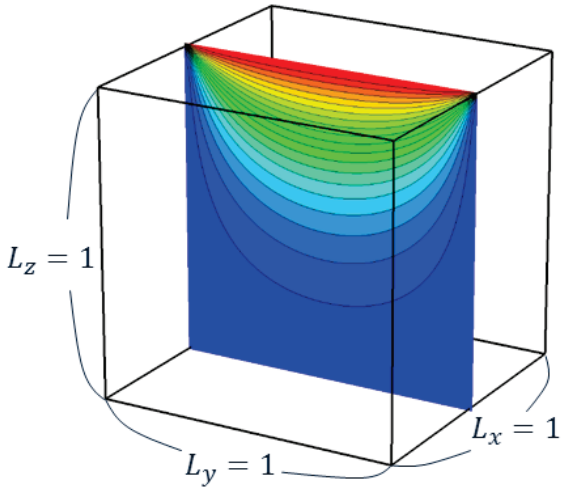

図 3　計算領域と中心断面における級数解の分布．

図 4 に各 MPI ランクにおける動的負荷分散前後の経過時間を示す．ここで経過時間は各ランクにおけるラプラス方程式の反復計算と MPI_Get による一対一通信にかかった時間を積算している．意図的に計算負荷を不均一にするために，各ランクで SOR 法の反復回数を `nmax=int(50.0*rand()+1.0)` とした．SGI ICE X を用い，スレッド数を 12 に固定して MPI/OpenMP ハイブリッド並列により計算を行った．図 4(左)に 512 キューブ，32 MPI プロセスの結果を示す．動的負荷分散の前では，各ランクの経過時間が大きく変動していることがわかる．このとき各ランクのキューブ数は一定である．動的負荷分散後には，経過時間の変動が小さくなっており，また，各ランクのキューブ数が大きく変動していることがわかる．図 4(右)に 4,096 キューブ，64 MPI プロセスの結果を示す．図 4(左)と同様の結果を示しているが，動的負荷分散後の経過時間のばらつきが小さくなっていることがわかる．これは各ランクが受け持っているキューブ数が多い方が，負荷分散を行う余地が大きいことを示している．

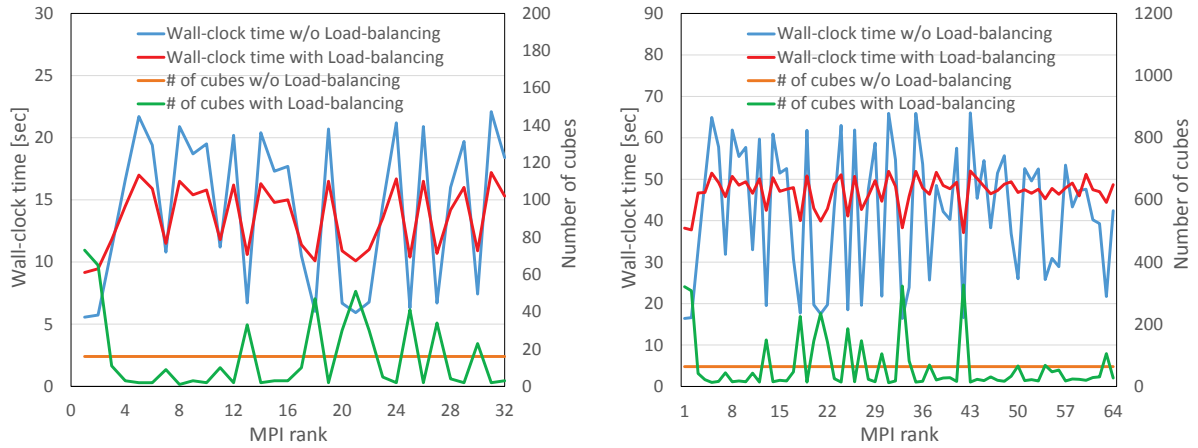

図 4　各 MPI ランクにおける動的負荷分散前後の経過時間，(左) 512 キューブ，32 MPI プロセス，(右) 4,096 キューブ，64 MPI プロセス．

図 5 に MPI 総ランク数を変えたときの経過時間の変化を示す．ここでは，ラプラス方程式の計算時間，MPI_Get および Reduction 通信にかかった時間を積算して経過時間としている．図 5 には NEC LX406Re-2 および SGI ICE X を用いて評価した結果を示している．共に MPI/OpenMP ハイブリッド並列を行い，スレッド数は 12 に固定している．ここでも各ランクに多くのキューブが割り当てられている場合に，動的負荷分散の効果が大きくなることがわかる．一方で，NEC LX406Re-2 では，動的負荷分散の効果があまり見られなかった．

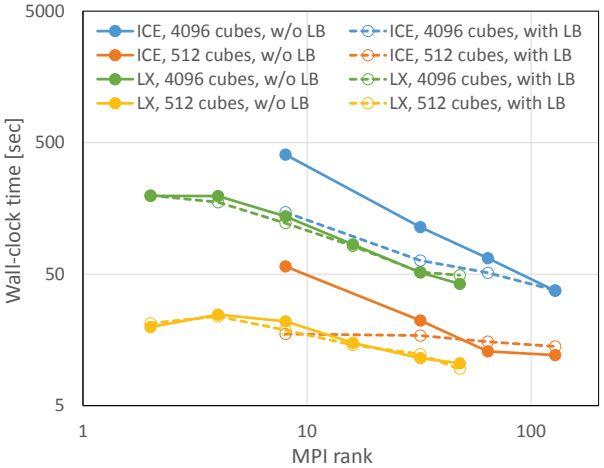

図 5　総 MPI ランク数を変化させたときの経過時間の変化（動的負荷分散有無）．

表 1 利用した計算機の諸元

	1 ノード（CPU，メモリ）	ノード数
NEC LX406Re-2	Intel Xeon E5-2695v2 (12 core)×2，128GB	68 ノード
SGI UV2000	Intel Xeon E5-4650v2（10 core）×128，8TB	1 式
	Intel Xeon E5-4650v2（10 core）×128，4TB	2 式
	Intel Xeon E5-4650v2（10 core）×64，2TB	4 式
SGI ICE X	Intel Xeon E5-2697v2 (12 core) ×2，128GB	176 ノード
	Intel Xeon E5-2697v2 (12 core) ×2，256GB	128 ノード
	Intel Xeon E5-2680v3 (12 core) ×2，128GB	136 ノード

利用した計算機の諸元を表 1 に示す．前述のように，NEC LX406Re-2 と SGI ICE X は図 5 に示すラプラス方程式ソルバーのスケーリング調査に計算に利用した．一方で，SGI UV2000 は後述の流体シミュレーションにおいて利用した．SGI UV2000 は大規模共有メモリ計算機であるため，表 1 のような表記とした．

図 6 に計算領域中心断面における数値解ϕ_n，数値解と級数解の差$\phi_n - \phi_s$，数値解ϕ_nの微分値を各キューブで積分した値の分布，そして，各キューブの MPI ランクを示す．図中の黒実線はキューブ境界を示している．境界条件により，数値解と級数解の誤差は上角部分で大きくなっており，解適合格子細分化においては，この付近のキューブが分割されることになる．図 7 に適合格子細分化による誤差の変化とキューブの分布を示す．全キューブを一様に細分化した場合の誤差も示している．ここでは数値解の微分値とそのキューブ内積分値を格子細分化の指標としている．数値解の微分値を細分化指標とした場合には，角部のみ格子細分化が繰り返され，結果として領域全体の誤差はそれほど減少しない．一方で，数値解の微分のキューブ内積分値を格子細分化指標とした場合には，計算領域の広い範囲が細分化され，より誤差が減少していることがわかる．全キューブを一様に細分化した結果と比較すると，キューブ内積分値を用いた解適合格子細分化により少ないキューブ数でより小さな誤差を実現している．

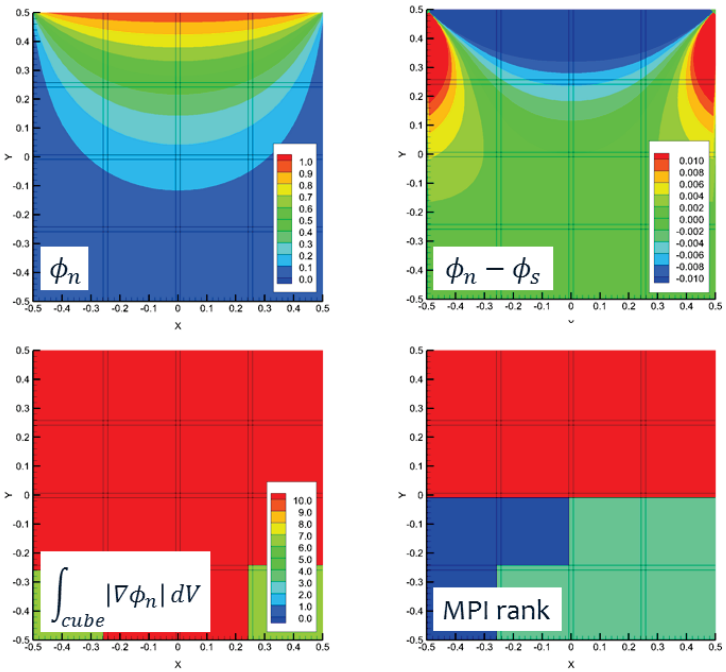

図 6 計算領域中心断面における数値解ϕ_n，数値解と級数解の差$\phi_n - \phi_s$，数値解ϕ_nの微分値を各キューブで積分した値の分布，そして，各キューブの MPI ランク．

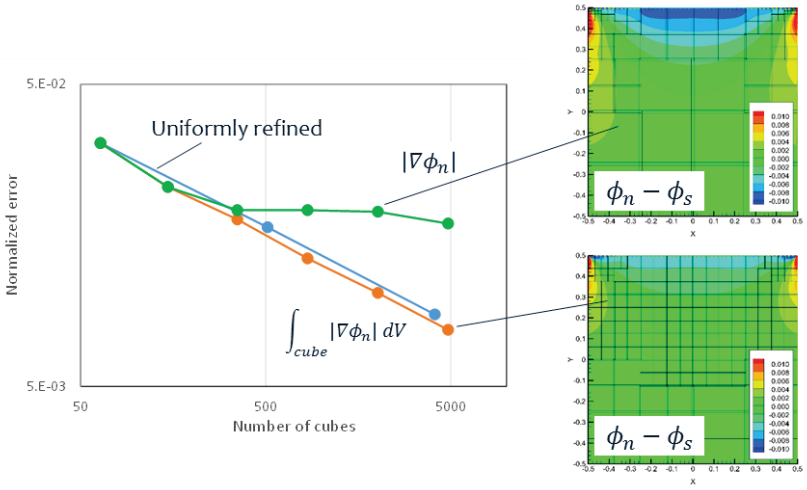

図 7　異なる指標を用いた解適合格子細分化による誤差の変化とキューブの分布.

4. 非圧縮性流体シミュレーションへの適用例

流れ解析には式(4)に示す 3 次元非圧縮性ナビエ・ストークス方程式と式(5)の連続の式を用いる.

$$\frac{\partial u_j}{\partial t} + \frac{\partial u_j u_k}{\partial x_k} = -\frac{1}{\rho_0}\frac{\partial p'}{\partial x_j} + \frac{\partial}{\partial x_k}\left[(\nu + \nu_t)2S_{jk}\right], \tag{4}$$

$$\frac{\partial u_j}{\partial x_j} = 0. \tag{5}$$

ここで, u_jとp'はそれぞれ3次元速度成分($j,k = 1, 2, \text{or } 3$)と, 基準圧力からの変動分$p = p_0 + p'$である. $S_{jk} = \left(\partial u_j/\partial x_k + \partial u_k/\partial x_j\right)/2$はひずみ速度テンソルである. 式(4)および(5)では速度成分u_jに関してアインシュタインの縮約記法を用いている. 密度は一定値($\rho_0 = 1.2$ kg/m^3)とし, 式(4)に現れる渦粘性係数はコヒーレント構造サブグリッドスケールモデルより得られる[7].

空間充填曲線や, それを用いた動的負荷分散は, 計算中に流れ場に応じて局所的に格子解像度を変化させる解適合格子細分化との相性が良い. 解適合格子細分化には流れ場から得られる格子細分化のための指標が必要であるが, 本研究では式(6)で定義される無次元壁座標(y^+)を用いた格子細分化を行った. 式(6)でτ_wは壁面せん断応力である. 境界層内の速度分布を y^+を用いた鉛直座標を用いてプロットすることで, 異なるレイノルズ数の流れから得られる速度分布を同一の曲線上に乗せることができる. すなわち, y^+を格子細分化の指標とすることで, レイノルズ数にかかわらず境界層分布を解像するために必要な格子点数を確保することができる.

$$y^+ = \frac{\rho u^+ \Delta x}{\mu}, u^+ = \sqrt{\tau_w/\rho} \tag{6}$$

解適合格子細分化および動的負荷分散の適用例として, 球周りの非圧縮性流体解析を行った. 図 8 に球中心を通る断面における物体近傍のy^+分布を示す. 図中の実線はキューブ境界を示している. ここでは$y^+ = 5$ を閾値とし, キューブ内のy^+の最大値が 5 を超える場合に, そのキューブを細分化(分割)する. 図 8(左)に示すように, 初期キューブで$y^+ > 5$ となっている赤い領域は, 格子細分化が行われ, その結果をして, 図 8(中, 右)に示すようにy^+の値が小さくなっていることが確認できる.

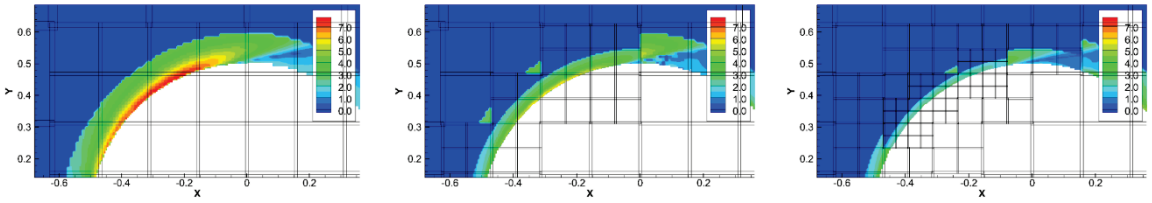

図 8　無次元壁座標(y^+)に基づく解適合格子細分化の様子.

図 9 に格子細分化中の主流方向流速およびキューブの分布を示す. 図 9(左)はレイノルズ数Re = 10^4 における格子細分化前の流れ場である. y^+に基づく格子細分化を行うことによって, 図 9(中)に示すように球前部のキューブが細分化されている. この状態からレイノルズ数を 2 倍にすることによって, 物体付近のy^+が大きくなり, 図 9(右)のように, 格子細分化がさらに行われる. 図 10 に解適合格子細分化中の物体表面と圧力係数分布を示す. 物体表面の境界条件として IBM を用いているため, 図 10 に示す階段状の表面が直接流れ場に影響するわけではないが, 格子細分化によりその領域の解析誤差を小さくすることができる.

図 11 に解適合格子細分化中の経過時間の変化を示す. SGI UV2000 を用いて, MPI/OpenMP ハイブリッド並列により計算を行った. OpenMP のスレッド数は 8 に固定した. 図 11(左)から解適合格子細分化後に動的負荷分散によってキューブが分配されている様子がわかる. これにより, 格子細分化も各 MPI ランクで経過時間の差がそれほど大きくなっていない. 図 11(右)に解適合格子細分化中の Reduction 通信を含む経過時間を示す. 図中の AMR は解適合格子細分化, LB は負荷分散を示している. キューブ分割によりキューブ数が増加して経過時間も増えるが, 負荷分散により経過時間が減少していることがわかる.

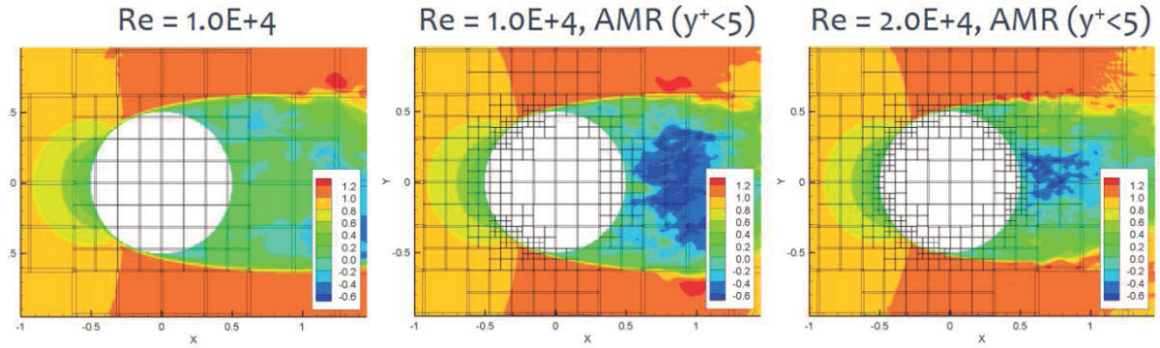

図 9　解適合格子細分化中の主流方向流速およびキューブの分布.

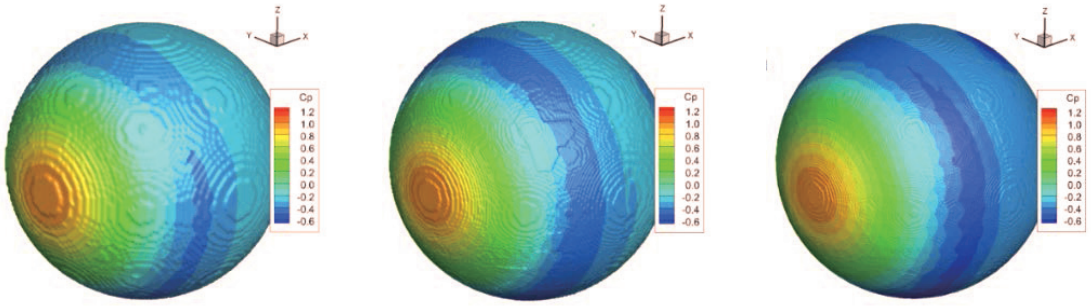

図 10　解適合格子細分化中の物体表面と圧力係数分布.

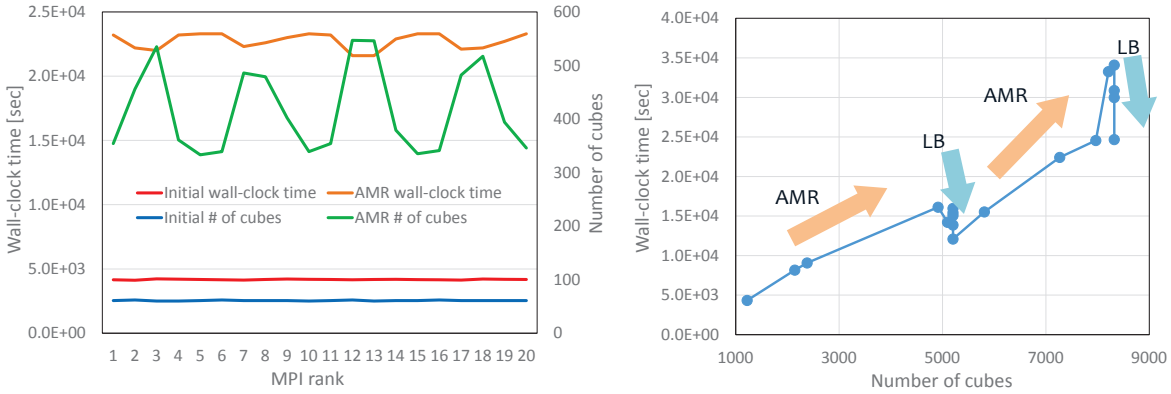

図 11　解適合格子細分化中の経過時間とキューブ数の変化.

5. おわりに

本研究では，分散メモリ型計算機において解適合格子細分化と動的負荷分散を行うためのフレームワークを Building-Cube Method（BCM）に基づき構築した．解適合格子細分化を行うと特定の MPI ランクに計算負荷が集中する場合があるが，動的負荷分散と組み合わせることでこれを解消した．ラプラス方程式を用いて，この BCM フレームワークの解適合格子細分化や動的負荷分散に関する検証を行い，その後，3 次元非圧縮ナビエ・ストークス方程式に適用した．流れ場の解適合格子細分化では，無次元壁座標（y^+）を格子細分化指標とした解析を行い，その有効性を確認した．

謝辞

本研究の結果は東北大学サイバーサイエンスセンターの並列コンピュータ LX402Re-2，東北大学流体科学研究所の SGI UV2000，そして，統計数理研究所の SGI ICE X を利用することにより得られた．関係各位に感謝の意を表します．

参考文献

[1] Nakahashi, K., “High-Density Mesh Flow Computations with Pre-/Post-Data Compressions,” AIAA Paper 2005-4876, 2005.
[2] Ishida, T., Takahashi S., and Nakahashi K., “Efficient and Robust Cartesian Mesh Generation for Building-Cube Method,” Journal of Computational Science and Technology, Vol. 2, No. 4, pp. 435-446, 2008.
[3] Onishi, K., Tsubokura, K., Obayashi S., and Nakahashi K., "Vehicle Aerodynamics Simulation for the Next Generation on the K Computer: Part 2 Use of Dirty CAD Data with Modified Cartesian Grid Approach," SAE International Journal of Passenger Cars- Mechanical Systems, Vol. 7, pp. 528-537, 2014.
[4] Sakai, R., Sasaki, D., Obayashi, S., and Nakahashi, K., “Wavelet-based Data Compression for Flow Simulation on Block-Structured Cartesian Mesh,” International Journal for Numerical Methods in Fluids, Vol. 73, No. 5, pp. 462-476, 2013.
[5] Su, X., Yamamoto S., Nakahashi K., “Analysis of a Meshless Solver for High Reynolds Number Flow,” International Journal for Numerical Methods in Fluids, Vol. 72, No. 5, pp. 505-527, 2012.
[6] Aftosmis, M. J., Berger, M. J., and Murman, S. M., “Applications of Space-Filling Curves to Cartesian Methods in CFD,” AIAA Paper 2004-1232, 2004.
[7] Kobayashi, H., “The Subgrid-Scale Models based on Coherent Structures for Rotating Homogeneous Turbulence and Turbulent Channel Flow,” Physics of Fluids, Vol. 17, pp. 045104, 2005.