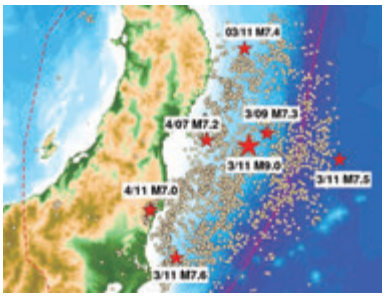
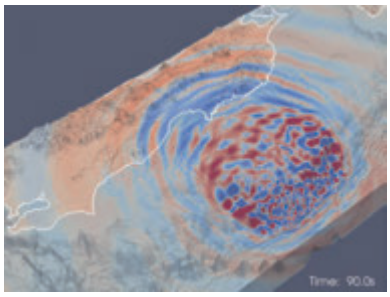
e-Science Journal
東京工業大学 学術国際情報センター
Global Scientific Information and Computing Center
Vol. 6

TSUBAME ESJ.





4096 GPUを用いた4096^{3}規模の一様等方性乱流の渦法解析

Turbulence Simulation Using 4096^3 Vortex Particles on 4096 GPUs

次世代シークエンサーから得られる大量メタゲノム情報の解析のための超高速パイプライン

An Ultra-fast Computing Pipeline for Metagenome Analysis with Next-Generation DNA Sequencers

大規模並列GPU計算による地震波伝播シミュレーション

GPU-Accelerated Large-Scale Simulation of Seismic-Wave Propagation

http://www.gsic.titech.ac.jp/TSUBAME_ESJ

4096 GPUを用いた4096^3規模の一様等方性乱流の渦法解析

横田 理央* Lorena Barba** 成見 哲*** 泰岡 顕治****

*King Abdullah University of Science and Technology **ボストン大学 機械工学科 ***電気通信大学 情報理工学部 情報・通信工学科
****慶應義塾大学 理工学部 機械工学科

高レイノルズ数の乱流解析はハイパフォーマンスコンピューティングの困難なアプリケーションの一つである。ここでは、4096^3の渦粒子を用いた渦法による一様等方性乱流の解析を行い同等の格子点数を用いた擬スペクトル法との比較を行った。二つの異なる手法を同等のアプリーケーションについて比較することで渦法の計算精度の定量的な検証を行うことができた。また、4096 GPUを用いたときの高速多重極展開法を用いた渦法とFFTを用いた擬スペクトル法の並列化効率の定量的な比較を行った。

はじめに 1

乱流は我々の身の回りに広く存在し、エネルギーや環境に関する様々な工学上の問題に現れる。乱流解析は支配方程式の非線形性により空間的にも時間的にも高い解像度を要するため大規模な計算資源が必要となる。近年、計算機の発達によりNavier-Stokes方程式の直接数値解析が多くの場面で利用可能になってきている。

多くの工学的アプリケーションは複雑な境界条件や様々な物理的過程との連成が必要となる一方で、このような複雑な現象から孤立した素過程を抽出し解析することで乱流の普遍的な性質が明らかになることもある。一様等方性乱流はこのような素過程の一つであり、平均せん断流や壁面の影響を含まない高レイノルズ数乱流の基本的な分析が可能となる。

一様等方性乱流の直接数値解析にはこれまで擬スペクトル法が用いられてきた[1]。最大規模のものでは地球シミュレータ上で2002年に行われた4096^3格子点、$Re_\lambda=1130$の計算が挙げられる[2]。石原ら[3]はこの大規模直接数値解析結果から、高次の乱流統計量は高レイノルズ数で異なった挙動を示すが、構造関数のスケール指数は普遍的である可能性が高いことを示した。このような結論は最高性能のスーパーコンピュータを利用して初めて得られるものであり、ハイパフォーマンスコンピューテイングの有用性を象徴しているといえる。

地球シミュレータから京コンピュータにかけてLINPACKベンチマークにおける演算性能が約300倍向上したにも関わらず、上述の4096^3格子点の計算規模の記録は現在でも破られていない[1]。これは以下の二通りの解釈が可能である。a) FFTはトーラスネットワークを用いた分散メモリ型の大規模計算機で性能を発揮できない。b) LINPACKベンチマークは実アプリケーションの性能を予測する指標としては適当でない。前者の解釈では、次世代計算機で性能を発揮できるアルゴリズムの開発が必要であるということになる。後者の見方では、LINPACKに変わる新たなベンチマークの採用を促進し、ハードウェアとソフトウェアの協調設計に用いることが求められる。

本研究では一様等方性乱流の計算を行うための新たな手法を提案し、TSUBAME2.0上で良好なスケーラビリティが得られることを示す。ただし、スケーラビリティはパフォーマンスとは直接関係のない指標であることに注意されたい。寧ろ計算の遅いコードほど通信を隠蔽することができるため、良好なスケーラビリティを得やすい。このため、本計算ではスケーラビリティだけでなく計算時間も比較し、次世代のハードウェアとアルゴリズムの協調設計に役立つような指標を提供するよう心がけた。

渦法 2

「渦法」と称される数値解析手法には様々なものがある[4]。離散化手法で大別すれば、格子を一切用いないLagrange型のものと、格子と粒子を併用するsemi-Lagrange型の手法がある。支配方程式の定式化で分けると渦度ー流れ関数によるものと渦度ー速度によるものが挙げられる。いずれの場合も渦度方程式を解くことになるが、このとき粘性拡散項の粒子上での扱いにも様々な手法が提案されている。全ての「渦法」に共通の特長としては、対流項がLagrange的に扱われるため、数値拡散や数値分散の問題から解放されるという点が挙げられる。さらに、圧力でなく渦度をベースとする定式化を行うことで計算領域を渦度がある場所に限定できる。これは渦輪や翼端渦のような渦運動が支配的な外部流において計算点の大きな節約につながるため、演算とメモリ使用量を大きく削減することができる。

本計算では完全メッシュフリーの離散化手法と渦度ー速度による定式化を採用した[5]。このためにまず渦度場をGauss分布を基底関数にもつ渦粒子の重ね合わせによって表す。渦度ー速度による定式化では速度のPoisson方程式

$$\nabla^2\mathbf{u}=-\nabla\times\boldsymbol{\omega}$$

[1] 8192^3点を用いたスペクトル法のベンチマークの報告はされているが乱流の計算を長時間行ったものは著者らの知る限りまだない。

と渦度方程式

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \mathbf{u}\cdot\nabla\boldsymbol{\omega} = \boldsymbol{\omega}\cdot\nabla\mathbf{u} + \nu\nabla^2\boldsymbol{\omega}$$

を交互に解く。速度のPoisson方程式はGreen関数を用いた積分方程式に変換するとBiot-Savart式になる。これは全ての粒子間の相互作用を伴うN体問題に帰着し、高速多重極展開法（FMM）を用いて解くことができる。渦度方程式の各項は個別に解くが、更新する変数が各項で異なるため部分段階法には相当しない。対流項は粒子の座標を更新することで正確に解くことができる。伸張項の計算は速度にBiot-Savart式を代入することで N体問題に変換できる。拡散項は基底関数であるGauss分布の標準偏差σを増大させることで考慮した。Lagrange型の離散化手法が収束するためには基底関数が十分重複する必要があるため、本計算では数ステップに一回渦粒子の座標を等間隔に再配置し、基底関数のσを再初期化し、渦粒子の強度は元の渦度場を再現するように調節した。このとき渦強度を調節する計算は動径基底関数を用いた補間によって行った。ただし、動径基底関数には渦要素の基底関数に合わせてGauss分布を用いた。

3 高速多重極展開法 (FMM)

Biot-Savart式と渦度方程式の伸張項の計算は全ての粒子同士の相互作用の計算になるため直接解くと$O(N^2)$の計算量になる。高速多重極展開法（FMM）を用いることでこのときの計算量を$O(N)$に低減することができる。

Biot-Savart式はPoisson方程式から導出できるためLaplace型のFMMカーネル関数を用いることができる。Biot-Savart式の右辺と渦度方程式の伸張項にはそれぞれ回転と勾配の空間微分演算子が存在するためLaplaceカーネルの二回微分までの項が必要となる。FMMで用いる多重極展開と局所展開にはこれらの微分が既に含まれているため、この情報をそのまま用いることができる。

3.1 負荷分散

FMMを並列化する際に問題となるのはN体問題に内在するデータの大域的な依存性と粒子のダイナミックな性質である。格子法とは異なり、N体問題では領域分割法を用いて粒子群を分割したとしても問題を局在化することはできない。これは、それぞれの部分領域にある粒子が必要とする情報が結局全領域にまたがっているためである。FMMでは大域的な木構造を用いた領域分割法が良く用いられるが、粒子の分布が非一様で木構造に偏りがある場合には領域を等分割すると逆に負荷は偏ってしまう。さらに、粒子が毎ステップ移動するため木構造を構築し直す必要があり、高速に木構造を生成、負荷分散する機構が不可欠となる。

FMMの負荷分散の問題を解決する巧妙な手法として前のステップの負荷のばらつきを元に現ステップの領域分割を更新するものが挙げられる。このような手法は９０年代前半から共有メモリ[8]、分散メモリ[9]の負荷分散にそれぞれ用いられてきた。前のステップの情報を元に負荷分散を微調整する発想自体はOrthogonal recursive bisection（ORB）[7]、Morton/Hilbert keyの分割法[9]、グラフ理論を用いた領域分割法[10]などの基本的な負荷分散の手法と併用することができる。本計算に用いるFMMコードはORB、Morton keyの分割の両方から最適なものを選ぶことができる仕組みになっている。

上述の方法で粒子の領域分割が行われると、それぞれの領域で木構造が生成される。次に、この局所的な木構造の一部を各プロセスが相互に送信しながら各部分領域にある粒子が必要な大域的な木構造(Local Essential Tree）を構築する必要がある。このLETの構築に伴う通信は大域的な通信になるため、大規模な分散メモリの計算ではここがスケーラビリティのボトルネックとなる。LETの通信の最適化手法としてORBの二分木の各階層を次元に見立てた超立方体に沿った通信パターンを用いるものがある。これはtreecode[11]にもFMM[12]にも適用することができる。本計算で用いたFMMコードでは上記の手法をプロセス数が2の冪乗でない場合に拡張したものを用いた。さらに、周期境界FMMに上記の通信方法が使えるようにも拡張した。

3.2 Dual Tree Traversal

粒子を領域分割し、各プロセスで局所的に木構造を生成し、LETを通信するところまではtreecodeとFMMの手順は全く同様である。これら二手法間の差異はLETを走査する方法にある。ただし、以下の説明では粒子間相互作用の作用される側をターゲット、作用を及ぼす側をソースと定義する。また、木構造の末端にあるセルを葉セルと呼ぶこととする。 木構造の生成時には、葉セルあたりの粒子数ができるだけ均一になるようにする。Treecodeではターゲットの木構造の葉セルに関するループをまわし、各葉セルに対してソースの木構造を走査する。一方、FMMでは通常木構造の走査は行わず、ターゲットセルに関するループをまわして各ターゲットセルについて作用するソースセルのリストを逐次生成する。粒子の分布が不均一で木構造に偏りがある場合にはこのソースセルのリストの生成が煩雑になる。このため、FMMでは通常multipole acceptance criterion（MAC）の概念は用いず、影響距離の固定されたソースのリストを使用する。

Dual tree traversal[13]を用いることで上記のソースのリストの生成を簡略化・最適化することができる。これは、FMMでよく用いられるU,V,W,Xリスト[12]にMACの概念を付加したものと考えることもできる。図1にdual tree traversalの概念図を示す。基本的な手順はターゲットとソースの双方の木構造を同時に走査するだけである。この際に生じるターゲットとソースのセルのペアに対してMACを適用し、セル同士の計算を行うのに十分な距離にあるかを判定する。十分でなければ大きい方のセルを分割し、再びターゲットとソースのセルのペアに対してMACによる判定を行う。これにより双対な木構造の走査を行い、ターゲットとソースの両方が葉セルに達した時点で距離が未だ

に十分遠くなければ直接計算を行う。Dual tree traversalはターゲットの親セルから子セルへソースの候補を継承することで、各階層で排他的なMACベースのソースリストを構築する最適な手法を実現しているといえる。

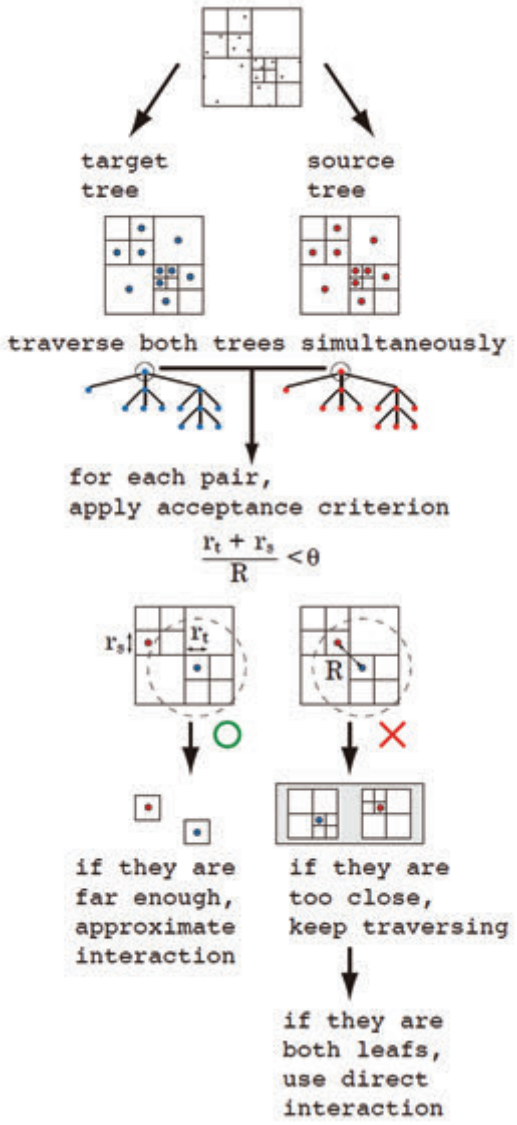


図1 Dual tree traversalの概念図

3.3 CPU/GPU 上での自動最適化

TreecodeやFMMなどの階層的なN体問題の高速化手法では近傍場のparticle-particle（P2P）カーネルと遠方場のmultipole-to-local（M2L）やmultipole-to-particle（M2P）カーネルが計算時間の大半を占める。また、この際の近傍場のP2Pと遠方場のM2L/M2Pの計算時間のバランスが重要である。CPUとGPUが混在する環境ではこれらのカーネルの加速率にばらつきがあるため[14]バランスをとることは容易ではない。さらに、多重極展開や局所展開自体にも様々な基底のものが存在し、計算機のアーキテクチャによってそのカーネルの性能は異なる。

これらのカーネルの中から最適なものを選ぶ簡単な方法として、各カーネルの処理時間を予め計測し実行時にその情報をもとに自動最適化を行うものがある。Dual tree traversal中にMACを満たすセルの各ペアについてP2P（direct）、M2P（treecode）、M2L（FMM）のカーネルから最適なものを選ぶことでいずれの単独の手法よりも常に高速な実装となる。このことは本コードを用いた性能試験で実証している[15]。

スケーラビリティ 4

本計算は全てTSUBAME 2.0上で行った。TSUBAME 2.0のthin-nodeはIntel Xeon5670を2ソケット、NVIDIA M2050を3カード、54GBのRAM、120 GBのローカルSSDを有するノード1408台からなる。ノード間はフルバイセクション・ノンブロッキングのfat-treeネットワークにより接続されており、各ノードはdual linkのQDR Infinibandでリンクされている。

図2にGPUあたりN=4096^2粒子を割り当てたときの1から4096GPUまでのweak scalingとそのときの計算時間の内訳を示す。本計算ではGPUあたりMPIプロセスを1つ割り当てたため、MPIプロセス数とGPU数は等しい。計算時間はBiot-Savart式と渦度方程式の伸張項の両方を含んでいる。 凡例の「P2P evaluation」は近傍場のP2Pカーネルの計算時間、「FMM evaluation」はその他のFMMカーネルの合計値、「MPI communication」は全ての通信時間の合計値、「GPU buffering」はGPU周りのデータのバッファリングとGPUとの送受信の合計、「Tree construction」は木構造の生成と領域分割、負荷分散の時間を表している。ただし、MPIの通信はローカルなP2Pカーネルの計算とオーバーラップしており棒グラフからはこの重複部分は引かれている。このため、棒グラフの合計の高さは全体の実行時間と一致している。最も大きい4096GPUを用いた計算ではN=4096^3の粒子の計算を100秒程度で行い1.01 Pflop/sを達成した。

図2を見ると「GPU buffering」に時間がかかっていることが分かる。ただし、これはFMMのカーネルをGPU上で効率的に実行するために必要であり、実行時間の合計値を低減する効果があることが分かっている。また、この部分は非常に良くスケーリングするため、FMM全体のスケーラビリティには全く影響しない。FMMのスケーラビリティに効いてくるのは「MPI communication」と「Tree construction」である。「Tree construction」の中にもMPI通信があり、これがスケーラビリティを低下させている。3.1節で述べた超立方体に沿った階層的なLETの通信を行う手法は、TSUBAME 2.0のネットワーク上では単純なMPI_Alltoallvに比べて低速であった。このため、図2ではMPI_

Alltoallvを用いたときの通信時間を示している。

擬スペクトル法に関しても同様なweak scalingの試験を行い、図3にこのときの並列化効率をFMMと比較したものを示す。ただし、本計算を行った時点ではCPU上のFFTWとGPU上のCUFFTの計算速度に（CPU-GPU間の通信時間を含めると）大きな差異が見受けられなかったため、擬スペクトル法の計算はノードあたり3プロセスを用いてCPU 上で行った。4096プロセスを用いたときの並列化効率はFMMでは74%でスペクトル法では14%であった。計算時間は両手法ともに1タイムステップあたり100秒程度であった。

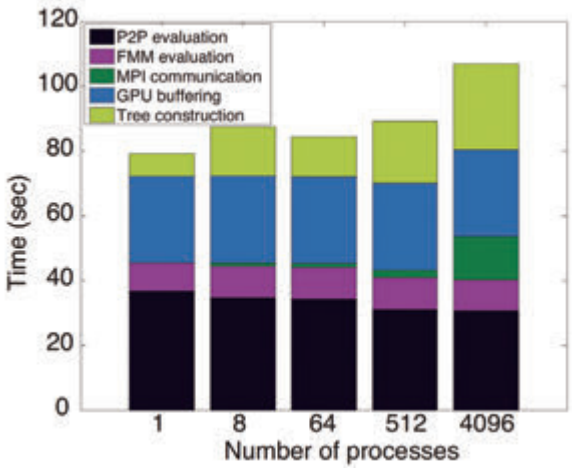


図2 GPUあたりN=4096^{2}粒子を用いた4096GPUまでのFMMのweak scaling

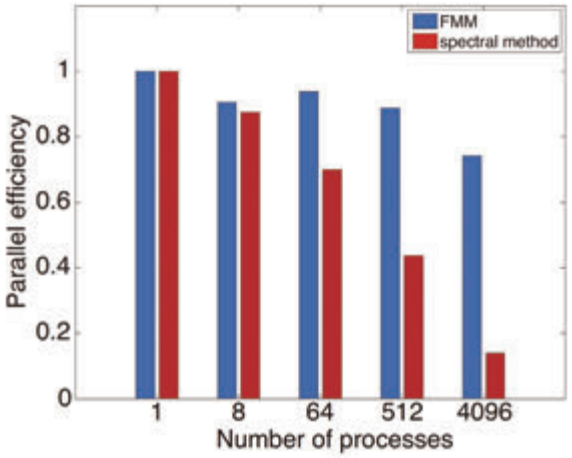


図3 FMMとスペクトル法の4096プロセスまでの並列化効率

一様等方性乱流の解析 5

ここでは前述のFMMコードにより高速化された渦法とFFTWによる擬スペクトル法を用いた Re_λ=500の一様等方性乱流の解析結果を示す。計算領域は$[-\pi,\pi]^3$の周期的な立方体で格子点および粒子の数は4096^3であった。FMMに関しては27^3の周期鏡像を置くことで周期境界条件を考慮した。また、FMMの級数展開の次数はp=14とした。初期条件は指定したエネルギースペクトルにランダムな位相を持たせたものを波数空間の速度場として生成した。擬スペクトル法はこの情報をそのまま初期条件として用いた。渦法では波数空間の速度場をまず実空間へと変換し各セルの中心点（スタガード格子における圧力の計算点）における渦度を計算した。次に、渦粒子を同じくセルの中心点に配置し、動径基底関数を用いた補間法によって渦度場を再現するように渦粒子の強度を計算した。渦粒子の渦核半径はオーバラップ比が1となるように$\sigma=\Delta x$とした。

図4にt/T=2における渦法の計算結果から得られた速度勾配テンソルの第二不変量の等値面を示す。ただし、Tは大規模渦の回転時間である。微小な渦構造が多数見受けられるが、大規模はコヒーレント渦構造は計算時間が不十分なため確認することができなかった。計算時間はTSUBAME 2.0のフルノードを占有できる時間の制約により僅かなものに限られた。ただし、渦法の計算精度やスケーラビリティの違いを定量化するためにはこのような短時間の計算で十分であるといえる。

図5にt/T=2における擬スペクトル法と渦法の速度場から得られるエネルギースペクトルを示す。ただしここでは、擬スペクトル法の結果が正しいものとして渦法の計算精度を検証することを目的とする。高波数成分において僅かな差異はみられるものの両手法間のスペクトルは定量的に一致していることが見てとれる。過去の一様等方性乱流の渦法による解析例[5][16]と比べて高い精度が得られている要因として以下のことが挙げられる。今回の計算ではFMMの級数展開の次数が高く（p=14）、周期鏡像の数も多い（27^3）。また、再初期化の頻度が高く（5ステップに1回）、再初期化の際の動径基底関数の補間の収束判定を厳しく設定した（$|L|^2$=1e-5）。もちろん、膨大な数の渦粒子（4096^3）を用いていることも高い精度を実現できた要因の一つである。逆に、これらの条件を全て満たさない限り高精度な渦法計算は実現できないことが分かった。

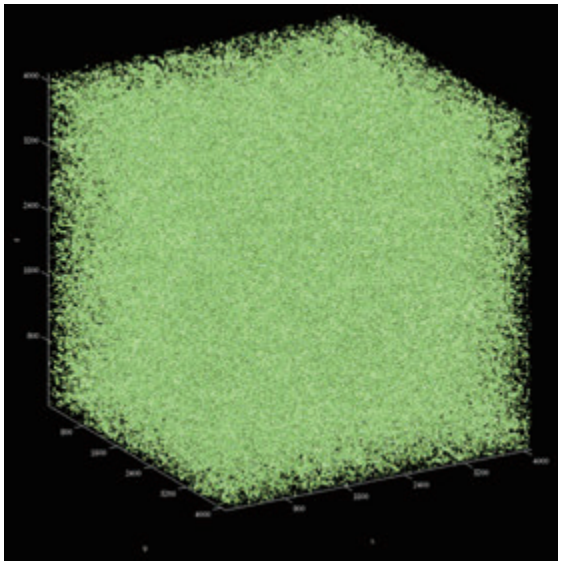


図4 渦法の計算結果から得られた
速度勾配テンソルの第二不変量の等値面

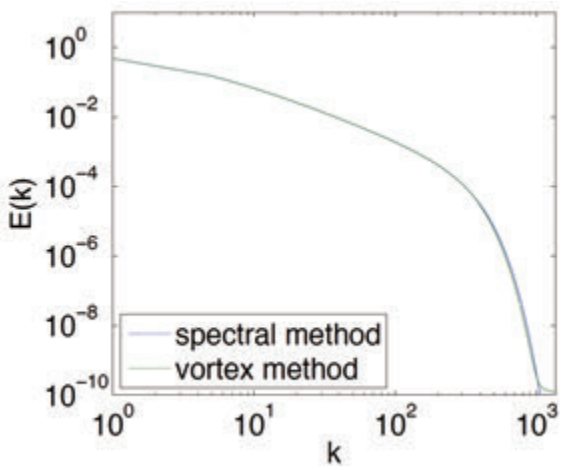


図5 擬スペクトル法と渦法の t/T=2 における
エネルギースペクトル

おわりに 6

本研究では一様等方性乱流を例にとって、二つの異なる計算手法による解析を同等の計算条件の元で行い、計算精度と計算速度、スケーラビリティに関する比較を行った。FMMをベースとする渦法による解析とFFTをベースとする擬スペクトル法の解析を4096^3の粒子・格子点を用いて行いTSUBAME 2.0上で最大4096 GPUまで用いた計算を行った。このときFMMの並列化効率は74%であったのに対してFFTの並列化効率は14%であった。両手法とも計算時間は1ステップあたり100秒程度であり、4096^3を用いた渦法計算は1.01Pflop/sの持続演算性能に相当する。渦法と擬スペクトル法のエネルギースペクトルは定量的に一致し、本手法の計算精度を検証することができた。ただし、TSUBAME 2.0全体を占有できる時間が限られていたため、高次の乱流統計量を得るのに十分な計算時間を確保することができなかった。今後、同様の計算資源が安価に利用できるようになれば、FMMを用いた乱流解析の有用性が明らかになると期待される。

謝　辞

本研究の計算機環境はTSUBAME2.0の平成23年秋期グランドチャレンジ大規模計算制度によるものである。本研究の一部は科学技術振興機構CREST「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出」から支援を頂いた。ここに感謝の意を表する。

参考文献

[1] S. A. Orszag and G. S. J. Patterson: Numerical Simulation of Three Dimensional Homogeneous Isotropic Turbulence. Phys. Rev. Lett. Vol. 28, pp. 76-79 (1972)

[2] M. Yokokawa, K. Itakura, A. Uno, T. Ishihara, and Y. Kaneda: 16.4-Tflops Direct Numerical Simulation of Turbulence by a Fourier Spectral Method on the Earth Simulator, Proc. ACM/IEEE Supercomput. Conf., Washington DC, pp. 50-66 (2002)

[3] T. Ishihara, T., Gotoh, and Y. Kaneda: Study of High-Reynolds Number Isotropic Turbulence by Direct Numerical Simulation, Annu. Rev. Fluid Mech., Vol. 41, pp. 165-180 (2009)

[4] G. H. Cottet and P. Koumoutsakos: Vortex Methods, Cambridge University Press (2000)

[5] R. Yokota, T. K. Sheel, and S. Obi: Calculation of Isotropic Turbulence Using a Pure Lagrangian Vortex Method, J. Comput. Phys., Vol. 226, pp. 1589-1606 (2007)

[6] H. Cheng, L. Greengard, and V. Rokhlin: A Fast Adaptive Multipole Algorithm in Three Dimensions, Vol. 155, pp. 468-498 (1999)

[7] M. S. Warren and J. K. Salmon: Astrophysical N-body Simulation Using Hierarchical Tree Data Structures, Proc. ACM/IEEE Supercomput. Conf., pp. 570-576 (1992)
[8] J. P. Singh, C. Holt, J. L. Hennessy, and A. Gupta: A Parallel Adaptive Fast Multipole Method, Proc. ACM/IEEE Supercomput. Conf., pp.54-65 (1993)
[9] M. S. Warren and J. K. Salmon: A Parallel Hashed Oct-tree N-body Algorithm. Proc. ACM/IEEE Conf. Supercomput., pp. 12-21 (1993)
[10] S.-H. Teng. Provably Good Partitioning and Load Balancing Algorithms for Parallel Adaptive N-body Simulation. SIAM J. Sci. Comput., Vol. 19, pp. 635-656 (1998)
[11] T. Hamada, K. Nitadori, K. Benkrid, Y. Ohno, G. Morimoto, T. Masada, Y. Shibata, K. Oguri, and M. Taiji: A Novel Multiple-walk Parallel Algorithm for the Barnes-Hut Treecode on GPUs – Towards Cost Effective, High Performance N-body Simulation, Comput. Sci. – Res. Dev., Vol. 24, pp. 21-31 (2009)
[12] I. Lashuk, A. Chandramowlishwaran, H. Langston, T.-A. Nguyen, R. Sampath, A. Shringarpure, R. Vuduc, L. Ying, D. Zorin, and G. Biros: A Massively Parallel Adaptive Fast Multipole Method on Heterogeneous Architectures, Proc. Conf. High Perform. Comput. Netw. Stor. Anal., (2009)
[13] W. Dehnen: A Hierarchical O(N) Force Calculation Algorithm, J. Comput. Phys., Vol. 179, pp. 27-42 (2002)
[14] R. Yokota and L. A. Barba: Treecode and Fast Multipole Method for N-body Simulation with CUDA, GPU Computing Gems, Morgan Kaufmann (2011)
[15] R. Yokota and L. A. Barba: Hierarchical N-body Simulations with Auto-tuning for Heterogenous Systems. Comput. Sci. Eng., Vol. 14, pp. 30-39 (2012)
[16] R. Yokota, T. Narumi, R. Sakamaki, S. Kameoka, S. Obi, and K. Yasuoka: Fast Multipole Methods on a Cluster of GPUs for the Meshless Simulation of Turbulence. Comput. Phys. Comm., Vol. 180, pp. 2066-2078 (2009)

次世代シークエンサーから得られる大量メタゲノム情報の解析のための超高速パイプライン

石田貴士　鈴木脩司　秋山泰
東京工業大学　大学院情報理工学研究科　計算工学専攻

近年注目を集めているメタゲノム解析は未知の微生物に関するゲノム情報が得られるだけでなく、その環境中の共生系の理解や環境汚染の監視等に有用であるが、解析を進める上で計算量の大きな配列相同性検索処理がボトルネックの一つとなっている。この問題に対処するため、我々はGPUを用いた高速な配列相同性検索プログラムを開発し、TSUBAME2上に大規模なメタゲノム解析向けの超高速パイプラインの構築をおこなった。

はじめに 1

従来のゲノム解析は培養された単一の種のゲノム情報を明らかにするものであったが、近年シークエンサーの性能向上に伴い、土壌、海洋、ヒト体内等の環境中に生息する微生物のゲノムを分離培養せずにそのまま丸ごとシークエンスして解析することが可能となってきた。この解析手法はメタゲノム解析と呼ばれ、未知の微生物に関してのゲノム情報が得られるだけでなく、その環境中の共生系の理解や環境汚染の監視等に有用であり、大きな注目を集めている[1]。さらに近年では次世代シークエンサーと呼ばれる新型のシークエンサーの登場により、膨大な量のゲノム情報が短時間のうちに入手可能となっており、その大規模な情報を用いることでメタゲノム研究が更に進展することが期待されている。

しかし、このメタゲノムの解析では、サンプルに含まれる多くの種のゲノム情報がデータベースに登録されていないため、遠縁の種の配列データとの間で比較が可能となる高感度な検索手法が必要となる。配列相同性検索と呼ばれるこの検索処理は多くの計算を必要とする処理であり、その結果、メタゲノム解析を進める上でのボトルネックの一つとなってしまっている[2]。

そこで、本研究では次世代シークエンサーによる大量のメタゲノム情報を現実的な時間内に解析することを目的とし、TSUBAME2の膨大な計算能力を利用可能とする大規模な全自動解析パイプラインを構築した。また、そのパイプライン中で行われる配列相同性検索そのものを高速化するため、従来のメタゲノム研究で標準的に利用されてきた配列相同性検索プログラムBLASTX[3]に加え、BLASTXと同等の検索感度を持つオリジナルのGPUプログラムであるGHOSTM[4]を開発し、利用可能とした。これによって次世代シークエンサーの一度の読み取り実行によって得られるゲノム情報を数時間の内に処理することが可能となり、今後このパイプラインによって次世代シークエンサーによるメタゲノム解析が促進されることが期待される。

メタゲノムマッピング 2

次世代シークエンサーと呼ばれる現在のシークエンサーは非常に高いスループットを持ち、1度の読み取り実行で数千億塩基以上が解読可能となっている。しかし、その出力は100塩基（bp）程度の短い断片配列であるため、そこから意味のある情報を得るにはその出力に対してアセンブリやマッピングといった処理を計算機上で行う必要がある。

マッピング処理は既知のゲノム配列に対して断片配列を貼り付け、その一致する位置を同定する処理である。従来行われてきたような単一の生物種に対する解析ではリファレンスゲノムが存在するため、多くの不一致やギャップを許容する必要がなく、通常の文字列検索に近い処理によって解析が可能であった。現在ではBWA[5]やBowtie[6]といった高速なプログラムが開発されており、数台のワークステーションがあれば次世代シークエンサーの出力に対しても十分に対処が可能となっている。

その一方、メタゲノム解析では環境中に含まれる微生物のすべてのゲノム配列が既知であることは稀であるため、マッピングにおいては近縁の種のゲノム情報を参照する必要があり、曖昧な一致まで検知可能となるような高い感度の検索が必要となる。そのような多くの不一致やギャップを許容する検索は一般に配列相同性検索と呼ばれ、従来のゲノムマッピングに比べて非常に多くの計算を必要とする。さらにメタゲノム解析ではより高い検索感度を得るため、A, T, G, C の4文字で表現されるDNA 配列のままで検索を行なわず、それらをコドン表に従い20種類のアミノ酸からなるタンパク質配列に翻訳してから解析を行う。DNA配列の比較では塩基の差は一致か不一致の2状態でしか区別しないことが多いのに対し、タンパク質配列ではアミノ酸間で性質の類似度の違いから、アミノ酸置換毎に異なるスコアを用いるため、計算量は更に大きなものとなる。

このようなタンパク質配列間での曖昧な一致も含めた検索を行うため、現在では比較的高速かつ高感度で配列比較が可能な近似的手法BLAST[3]が利用されてきた[7]。しかし、次世代シークエンサーは膨大な量のDNA 断片配列を出力するため、すべてのDNA断片配列

をアミノ酸配列に翻訳してからマッピングを行うのはBLASTXプログラムを用いても長い計算時間が必要となる。現在、最新のIllumina社のHiseq 2000 DNA シークエンサーの出力は1度の読み取りで合計600G 塩基にも達し、そのデータを解析するには約25,000CPU 日が必要となってしまう。 そのため、現在ではメタゲノムの解析においては、このマッピング処理が大きなボトルネックとなっており、その高速化が必要とされている。

GPU 配列相同性検索プログラム GHOSTM 3

我々はメタゲノム解析において処理のボトルネックとなっている配列相同性検索を高速化するため、GPUを利用した高速な配列相同性解析プログラムであるGHOSTM[4]を開発した。GHOSTMのアルゴリズムはBLASTと類似したものであるが、GPU上での実行により適したものとなっており、メタゲノムの解析に必要十分な検索感度を持つように設計されている。GHOSTMはNVIDIA社のCUDAを用いて実装されており、CUDA バージョン2.2以上が利用可能なNVIDIA社製のGPUが搭載された計算機で利用可能である。一部の処理を簡略化し、処理の大部分をGPU化することで大幅な高速化を達成しており、詳細は後述するが1GPUを用いた場合、1CPUコア上で実行されたBLASTXに対して約130倍の高速化を達成している。

3.1 アルゴリズム

図1に示すように、GHOSTMではBLASTと同様にまずK文字の部分文字列の一致を探索する事でアラインメントの候補となる部位を同定し、その後に各アライメント候補についてSmith-Watermanアルゴリズム[8]により局所的アラインメントを行う事で詳細なアラインメントとそのスコアを計算している。そして、最終的に全てのアラインメントのスコアがソートされ、そのスコア上位のヒットが検索結果として出力される。

GHOSTMではこれらの処理のうち、計算の大部分を占めるアラインメント候補探索と局所的アラインメントの双方をGPU上で処理している。GPUは多くのコアを搭載しており、処理を適切に並列化することで大きな高速化が可能であるが、GHOSTMでは大量の断片配列を処理する必要があるため、まず、アラインメント候補の探索の際には各断片配列の処理を各GPUスレッドで行い、局所的アラインメントの際には各アラインメント候補に対する処理をGPUスレッドに割り当てて処理することで並列化を行っている。

3.2 検索速度と検索感度

メタゲノム解析では高い検索感度が必要となるため、BLAT[9]のようなBLASTより高速であるがに検索感度の低いアルゴリズムを利用する事は困難であった。一方GHOSTMはメタゲノム解析に十分な検索感度を有しており、図2で示すように動的計画法によりアライメントを行うSSEARCH[10]の結果を正解としたテストではBLASTには劣るものの、BLATよりも高い検索感度を示し、特に実際に利用されるBitスコア50以上の領域ではBLASTとほぼ同程度の検索感度を示しており、BLASTの代替として利用可能な事が示されている。

また、表1は約75bpのリード100,000本をクエリとして、KEGGのgenes.pepタンパク質配列データベース（約2.5GB）に対して相同性検索を実行した際の実行時間であるが、GHOSTMの検索速度は1GPUを用いた場合、1CPUコア上で実行されたBLASTXに対して約130倍、4GPUを用いた際には約400倍の高速化を達成している。これは検索感度落とすことで高速化を実現しているBLATの高速化率約40倍に比べてもより高速であり、GHOSTMは高感度と高速化を同時に実現している。

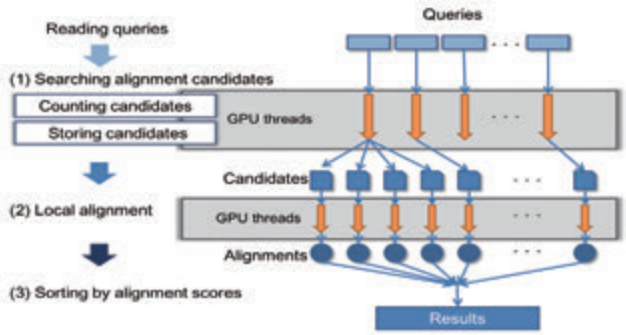


図 1 GHOSTM の処理の流れ

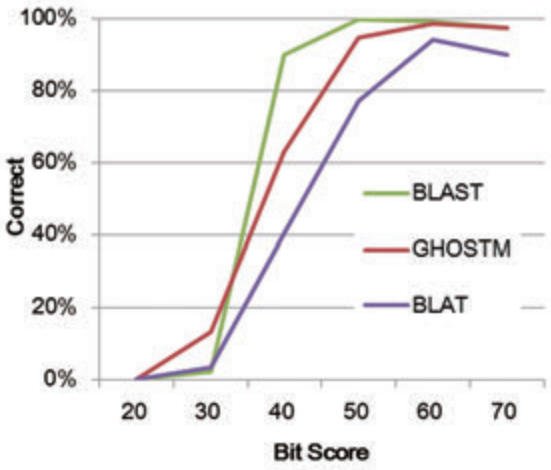


図 2 GHOSTM の検索感度

次世代シークエンサーから得られる大量メタゲノム情報の解析のための超高速パイプライン

Program	#GPUs	Time (sec.)	Acceleration ratio
GHOSTM	1	2855	129.5
GHOSTM	4	909	406.7
BLAT		9898	37.3
BLASTX (1 thread)		369678	1
BLASTX (4 threads)		102255	3.6

表 1 実行速度の比較

4 大規模全自動解析パイプライン

我々はメタゲノム解析におけるボトルネックである配列相同性解析を高速化するため、GPUを利用した高速なプログラムであるGHOSTMを作成した。しかし、一台の計算機では、このGHOSTMを用いても次世代シークエンサーの出力を解析するには不十分である。そのため、我々はTSUBAME2の大量の計算ノードを解析に利用することで、次世代シークエンサーによるメタゲノム解析を現実的な時間で行うことを目指し、TSUBAME2上に全自動のメタゲノム解析パイプラインを構築した。多くの計算ノードを利用する大規模な計算ではデータベースのコピー、計算結果の書き込みといったファイルの入出力に関する処理が問題となるが、このパイプラインではノード間でコピーを2分木状に行う工夫を行う事で入出力に依存するボトルネックを解消している。

4.1 大規模実行

我々は本パイプラインについてTSUBAMEグランドチャレンジ（超大規模アプリケーション）制度を利用し、次世代シークエンサーによって得られた大規模なデータに対する解析速度の検証を行った。解析に用いたデータは汚染土壌に関するメタゲノムデータである。メタゲノム原データは各75bpのDNAリード224 million本であるが、実際に配列相同性検索が行われたのは、ここから低品質なデータを除去後した71million本のDNAリードであり、これらがクエリとして4.2GBのタンパク質配列情報が含まれたNCBI nrデータベースに対する検索が行われた。パイプラインの相同性検索エンジンとしてはCPU上で動作するBLASTXと我々の開発したGPU上で動作するGHOSTMのそれぞれについて実効性能の比較を行った。

パイプラインは計算コア数に対してほぼ線形の速度向上を示し、図3のようにBLASTXを相同性検索に用いた場合、TSUBAME2のCPU 16,008コア（1,334ノード）用いた際に1時間あたり約24 millionのDNAリードの処理を実現した。また、GHOSTMを用いた場合では図4のように、2,520 GPU（840ノード）を用いることで1時間あたり約60 millionのDNA リードの処理を実現した。GHOSTMと2,520 GPUを用いた場合では1,260GPUを用いた場合に対して速度の向上が2割程度に留まっているが、これは計算資源に対して今回用いたデータのサイズが小さくなりすぎてしまったため、負荷分散等に失敗したことが原因であり、最新のシークエンサーの出力や、複数回のシークエンシングの結果を同時に処理すればGPU数に対する速度向上はほぼ線形となると考えられる。

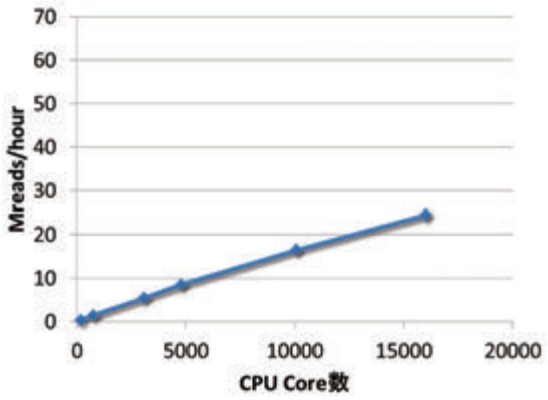


図 3 BLASTX を用いた際の
CPU コア数に対する処理速度の向上

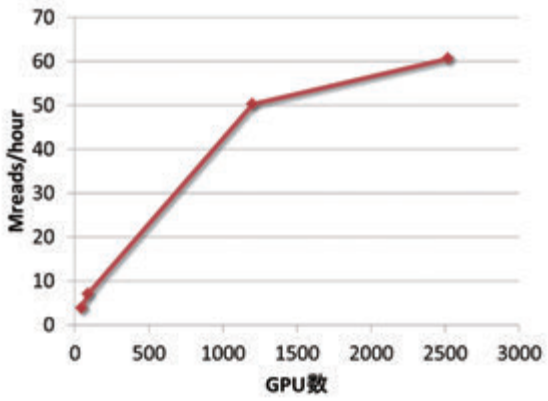


図 4 GHOSTM を用いた際の
GPU 枚数に対する処理速度の向上

おわりに 5

我々は次世代シークエンサーから得られるメタゲノム情報の解析において現在ボトルネックとなっている相同性検索を高速化することを目的とし、GPUによる効率的な相同性検索プログラムGHOSTMを開発し、またTSUBAME2の大量の計算機資源を利用可能とする大規模な解析パイプラインの構築を行った。解析パイプラインは使用したCPUコア数、GPU数に対してほぼ線形の速度向上を示し、TSUBAME2のほぼ全体を利用した場合、1時間あたり約60 millionのDNA リードの処理が可能であった。これは次世代シークエンサーの1度の読み取り実行から得られる出力に対する解析が数時間以内に可能となる事を示しており、今後我々のパイプラインによって次世代シークエンサーによるメタゲノム解析が促進されると考えている。

謝　辞

本研究の一部はHPCI戦略プログラム「予測する生命科学・医療および創薬基盤」、国立がんセンター研究所がん研究開発費、及びNVIDIA社CUDA COEプログラムの支援を受けて行われたものである。

参考文献

[1] M. Arumugam et al., “Enterotypes of the human gut microbiome.,” Nature, vol. 473, no. 7346, pp. 174-80, May 2011.
[2] J. C. Wooley, A. Godzik, and I. Friedberg, “A primer on metagenomics.,” PLoS computational biology, vol. 6, no. 2, p. e1000667, Jan. 2010.
[3] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, “Basic local alignment search tool.,” Journal of molecular biology, vol. 215, no. 3, pp. 403-10, Oct. 1990.
[4] S. Suzuki, T. Ishida, K. Kurokawa, and Y. Akiyama, “GHOSTM: A GPU-Accelerated Homology Search Tool for Metagenomics,” PLoS ONE, vol. 7, no. 5, p. e36060, May 2012.
[5] H. Li and R. Durbin, “Fast and accurate short read alignment with Burrows-Wheeler transform.,” Bioinformatics (Oxford, England), vol. 25, no. 14, pp. 1754-60, Jul. 2009.
[6] B. Langmead, C. Trapnell, M. Pop, and S. L. Salzberg, “Ultrafast and memory-efficient alignment of short DNA sequences to the human genome.,” Genome biology, vol. 10, no. 3, p. R25, Jan. 2009.
[7] P. J. Turnbaugh, R. E. Ley, M. a Mahowald, V. Magrini, E. R. Mardis, and J. I. Gordon, “An obesity-associated gut microbiome with increased capacity for energy harvest.,” Nature, vol. 444, no. 7122, pp. 1027-31, Dec. 2006.
[8] T. F. Smith and M. S. Waterman, “Identification of common molecular subsequences.,” Journal of molecular biology, vol. 147, no. 1, pp. 195-7, Mar. 1981.
[9] W. J. Kent, “BLAT--the BLAST-like alignment tool.,” Genome research, vol. 12, no. 4, pp. 656-64, Apr. 2002.
[10] W. R. Pearson, “Searching protein sequence libraries: comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms.,” Genomics, vol. 11, no. 3, pp. 635-50, Nov. 1991.

大規模並列GPU計算による地震波伝播シミュレーション

岡元太郎* 竹中博士** 中村武史*** 青木尊之****

*東京工業大学理工学研究科 **九州大学理学研究院 ***海洋研究開発機構 ****東京工業大学学術国際情報センター

2011年の東北地方太平洋沖地震は、強い地震動と巨大な津波によって東日本地域に計り知れないほどの被害をもたらした。この地震について、巨大地震の震源がどのようなものであったか、また巨大津波や地震動の励起はどのようになされたのかを探ることが、地球科学においてきわめて重要な課題となっている。そのためには大規模な地震波計算によって得られる理論波形を用いた定量的な解析が必要となる。我々はTSUBAMEのGPUを用いた高速・大規模計算によってこの問題に取り組んでいる。本稿では、我々が採用したGPU計算手法や計算性能について報告する。また、東北地方太平洋沖地震に関するシミュレーション例を紹介する。

はじめに 1

2011年3月11日に発生した東北地方太平洋沖地震（マグニチュード9、図1）は、強い地震動と巨大な津波によって東日本地域に計り知れないほどの被害をもたらした。しかし残念ながら、この地域ではこれほどの巨大地震の発生は想定されていなかった。そのため、巨大地震の発生条件や強震動・巨大津波励起の仕組みを探ることと、今後の地震防災に向けた研究を進めることが地球科学においてきわめて重要な課題となっている。

我々は地震波を解析することによって、この地震がどのようなものであったのかを推定し、地震時に発生した強震動や津波生成との関連を探る研究を進めている。 その研究では、地球内部不均質性や地形・海水層などの効果を考慮した大規模な地震波の理論計算を繰り返し実行しなければならない。そのため、大規模で高速な計算資源が必要となっている。そこで我々はGPUの演算性能に着目し、大規模並列GPU計算による地震波伝播計算プログラムの開発を進めている。

GPU（Graphics Processing Unit）は、その名称の通り、画像処理を高速に実行するためのデバイスである。GPUの特徴は、非常に多くの演算用コアを内蔵していることと、極めて高い数値演算性能を持つことである（図2）。最近のGPUは一千個以上の演算コアを内蔵し、単精度演算では2TFlops（テラ・フロップス）を越える理論ピーク演算性能を持っている。また、絶対性能と同時に電力あたりの性能も高い。さらに一般消費者向けに大量生産される製品であることから、高い演算性能にも関わらず安価にかつ容易に入手できるのもGPUの大きな特徴である。

GPUにはもう一つ、メモリ転送速度が非常に高速であるという重要な特徴がある（図2）。そのため、多量のメモリ読み出し・書き込みが発生するメモリ集中型（memory intensive）の計算問題でも高い性能を発揮する。地震波伝播のシミュレーションもメモリ集中型の問題であり、GPUを応用することには大きな利点がある。

本稿では、我々が開発してきたGPUプログラムについて紹介する[1-3]。我々のプログラムは格子計算型の手法、すなわち食い違い格子型の差分法にもとづいたものである。この手法は地震波伝播シミュレーションにおける時間領域解法の標準的な手法の一つとして広く用いられているものである。プログラムの主要部分の開発には東京工業大学学術国際情報センターのTSUBAME-1.2（2010年10月まで）、およびTSUBAME-2.0（2010年11月以降）を利用させていただいた。ここではTSUBAME-2.0による結果について報告する。

なお本稿は地震の研究をテーマとしているが、地震波計算は地球・惑星内部構造を推定するうえでも非常に重要な手法であることを付言しておきたい。特に資源探査の分野では地震波伝播を高速に計算する手法や高性能ハードウェアへの要請が非常に高く、GPU計算を応用した地震波計算や内部構造推定の研究が世界的に進められている。

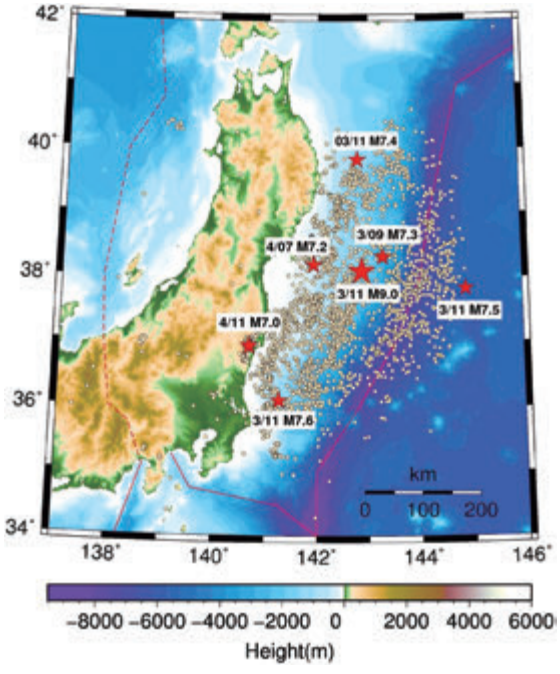


図1
東北地方太平洋沖地震（2011年3月11日）の余震分布（2011年3月9日から2011年4月11日、$M_{JMA} \geqq 4.0$）。マグニチュード7以上の地震の震央を星印で示した。震源データは気象庁による。

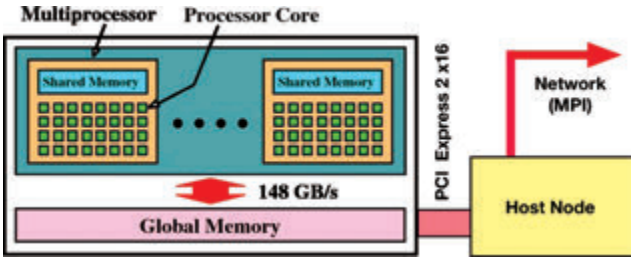


図 2 本研究で利用した GPU（NVIDIA M5020）の構成の概念図。

計算スキームについて 2

本稿で紹介するプログラムでは、時間領域の差分法（FDTD: finite-difference time domain）を用いている[4]。この差分法では媒質の粒子速度（v_i; i=x, y, z）と応力（τ_{ij}; i,j=x, y, z）とを変数として、図3に示す食い違い格子を用いて計算領域を離散化する。このような食い違い格子は地震波計算のほか、電磁界シミュレーションでも広く利用されている[5]。なお、本稿で紹介するプログラムでは、数値分散性を向上させるために空間差分精度は4次精度とした。時間差分精度は2次精度である。 完全弾性体の場合には1単位セルあたりのデータは粒子速度・応力が計9個、物性パラメータが3個（τ_{ij}と同じ格子点に置く）、合わせて12変数となる。非弾性減衰を含めた場合には変数の個数がこの数倍に増加する。（本稿で紹介する結果は全て弾性体版のものである。）

このスキームでは食い違い格子を用いるため、粒子速度変数と応力変数の時刻が時間刻みの半刻みだけずれることになる。計算上は、粒子速度と応力を片方ずつ更新していくことになる。

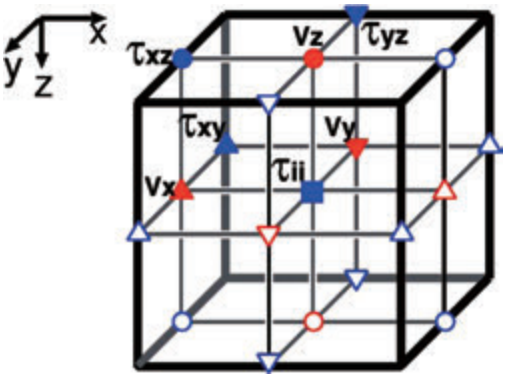


図 3 食い違い格子と変数の配置。図示した格子は単位セルに相当する。

GPU 計算 3

メモリ最適化

地震波の差分法計算のようなメモリ集中型の問題では、GPU内部のメモリ（グローバルメモリ）に差分法領域の変数全てを保存し、その変数に対して演算を行うことになる（図4a）。このグローバルメモリと演算ユニット間のメモリ帯域幅は、通常のCPUの帯域幅に比べると非常に大きい。しかしそれでもグローバルメモリから演算ユニットへのデータ転送には400-600サイクルの遅延が発生する。そのため、演算ユニット内部にある高速な共有メモリとレジスタを「キャッシュメモリ」として利用することが性能向上のためには重要である。

先に述べたように地震波計算では単位セルあたりの変数が多く、その一方で共有メモリのサイズは小さいため（M2050の場合 16 kBまたは48 kB）、共有メモリに3次元のブロックを確保することは困難である。そこで我々のプログラムでは共有メモリとレジスタとを併用する（図4a）。すなわち横方向への差分操作が必要になる2次元平面上のセルのデータは共有メモリに配置し、同時にこの2次元平面上の各セルに一つのスレッドを割りあてる。こうすると図の縦方向に位置するセルのデータについては縦方向への差分のみが発生しスレッド間で共有する必要がないので、これらのデータは各スレッドのレジスタに配置すればよい（図4b）。

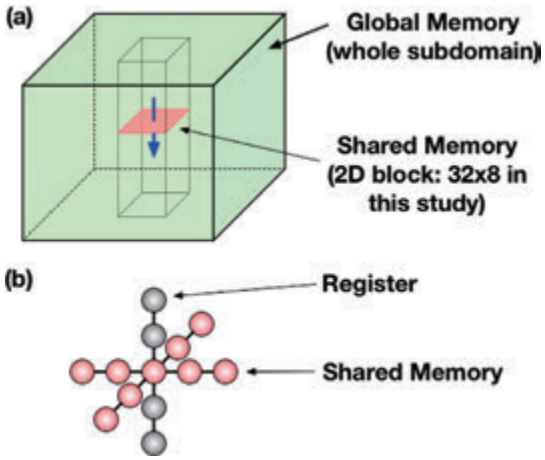


図 4
共有メモリとレジスタを併用する方法の概念図。(a) 副領域の変数をすべて GPU のグローバルメモリに収める。さらに、その中の小さな 2 次元ブロック（赤色）を共有メモリにコピーする。青矢印は計算が進行する方向を示す。(b) 差分法計算に関係するデータ配置の概念図。赤色の格子点のデータは共有メモリ、灰色の格子点のデータはレジスタにそれぞれコピーする。

そして、縦方向へ計算を進めていく際に共有メモリまたはレジスタに一度読み込んだデータを、一種のパイプラインのようにレジスタと共有メモリとの間で交換することによって、グローバルメモリへのアクセスを低減することができる。このようなレジスタと共有メモリの利用法は地震波シミュレーション[1,6-9]や気象シミュレーション[10]で利用されて効果をもたらしている。

この最適化のほかに、物性パラメータに関して参照テーブル方式を採用することや、単位セル中心以外の格子点での物性パラメータをセル中心の物性値を用いて毎回計算すること[11]などの方法によって、グローバルメモリへのアクセスを減らすようにしている。

このようにして作成したプログラムの性能を図5に示す。この性能は、次に述べるマルチGPU計算の機能を組み込んだプログラムを用いて、単一GPUで測定したものである。比較のために、TSUBAME-2.0 のホストノードCPU を用いて袖領域処理やMPI機能を含まない差分法プログラムの性能を測定した。参考的な比較となるが、これらのプログラムに関してはGPU 1基はホストノードのおよそ3倍の性能を持つ。

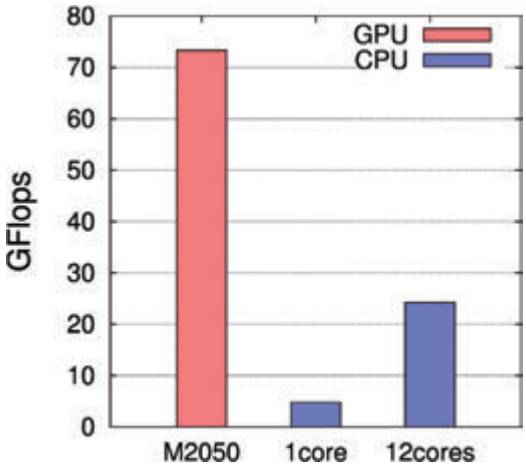


図5
GPU と CPU での地震波計算プログラムの性能。GPU の性能は領域サイズ 320 × 320 × 320 の場合について単一 GPU を用いて測定した。CPU では、袖領域処理を含まない差分法プログラムを Fortran と Open MP を用いて作成し、PGI Fortran コンパイラ（-fastsse オプションを利用）でコンパイルした。領域サイズは 320 × 320 × 320（1 コアの場合）、および 320 × 320 × 3840（12 コアの場合）とした。

袖領域転送の効率化

GPUのグローバルメモリのサイズは大きくない（M2050の場合では3 GB）。そのため大規模計算では、計算領域を多数の副領域に分割して複数のGPUに副領域を割り当てることが不可欠となる。本研究では大規模計算で標準的に利用される3次元領域分割を用いる（図6）。副領域間の通信にはMPI を用いる。

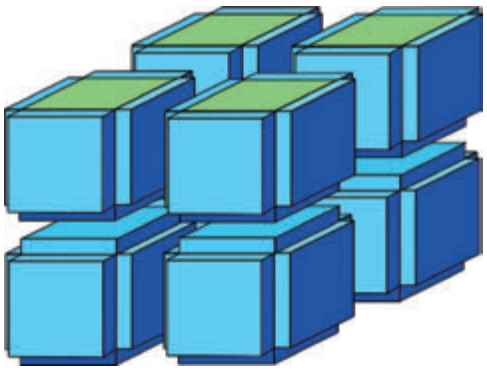


図6 3 次元領域分割の概念図。青い部分が袖領域を示す。

3次元領域分割では隣接する副領域との間で「袖領域」（ghost zone）を交換する必要がある。GPU計算では、この袖領域のメモリ並びに関して問題が発生する。これは、現状ではGPU間で直接データを交換することはできないことと関係している。図2に示すように、データを交換するためにはまずホスト計算機のメモリにGPUからデータを転送し、そのデータを別のホストに送るという段階を経なければならない。

袖領域のメモリを割り当てる方法としては、図7（a）のように内部領域の配列を延長する方法がしばしば採用される。しかし、袖領域内部ではメモリ並びは不連続となるため、袖領域のデータをホスト計算機のメモリに転送する際にはcudaMemcpyなどのデータ転送関数を細切れのデータごとに繰り返し呼び出すことになる。そのため袖領域全体を転送するには非常に長い時間がかかることになり、この方法はGPU計算では実用にならない。そこで、本研究では図7（b）のように内部領域とは独立の連続なメモリを袖領域に割り当てる。こうすると、一度のデータ転送関数呼び出しで袖領域全体をGPUからホストに転送することが可能となり、データ転送時間を（a）の場合よりも大幅に短くすることができる。ただしGPU内部では、袖領域メモリのデータを共有メモリのブロックに転送するための複雑なマッピングが追加される。

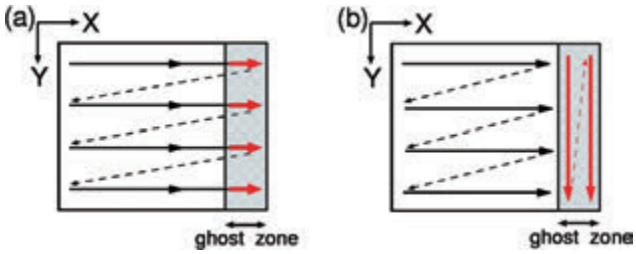


図7
袖領域（ghost zone）のメモリ並びの概念図 [1]。（a）内部領域の配列の延長として割り当てた場合。袖領域内部では数多くの小さなデータが不連続に並んだメモリ並びになる。（b）内部領域とは独立の連続なメモリ領域を割り当てた場合。

計算と通信のオーバーラップ

食い違い格子では粒子速度成分と応力成分を交互に時間積分していく。そのため、それぞれの積分演算を行うGPUカーネルも別々のものになる。さらに我々のプログラムでは、計算に要する全体の時間を短縮するために、内部領域についての積分と袖領域通信とをオーバーラップさせている。そのため、側面ブロックのみを処理するカーネルと内部ブロックのみを処理するカーネルの2つのカーネルを作成して利用している。計算と通信のオーバーラップのためには、GPUとホスト間のメモリ転送においてCUDA の非同期通信関数(cudaMemcpyAsync)を用いる必要がある。また、各 MPIプロセス間のデータ転送にはMPIの non-blocking 関数(MPI_Isend、MPI_Irecv)を用いた。

このようにして作成したマルチGPUプログラムについて、TSUBAME-2.0で全領域サイズを変えて計算を実行したときの、GPU数に対する性能を図8に示す。この例では副領域サイズを固定しているので、グラフは弱スケーリング性能を示すことになる。図からわかるように、800 GPU の場合まで理想的なスケーリングに近い(GPU数にほぼ比例する)実効性能が得られた。絶対値としても、800 GPUの場合に約50 TFlops、1200 GPU の場合に約61 TFlops という、非常に高い実効性能を達成できた(いずれも単精度性能)[3]。

なお、1200 GPU のときには理想的な場合よりも少し性能が低下しているように見える。800 GPU の場合の性能は理想的な数値の約85%、1200 GPUの場合には約69%である。この計測では800GPUまでは2 GPU/node、1200 GPU のときは3 GPU/nodeの条件で計算した。そのため、後者の場合にはノード間通信が増加してスケーラビリティがやや低下したと考えられる。この点は今後改良を進める必要がある。

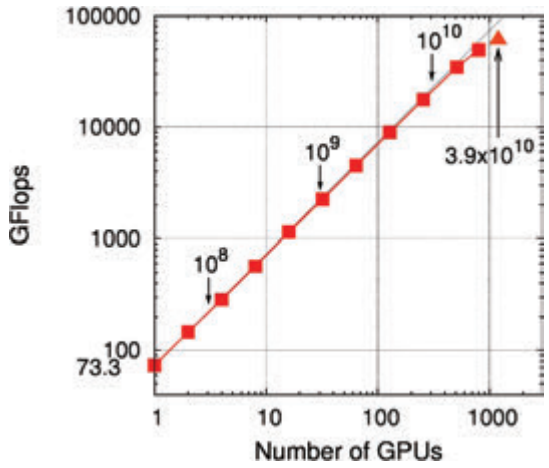


図 8
TSUBAME-2.0 での実効性能。副領域サイズを 320 × 320 × 320 に固定して、全領域サイズを変えて(つまり分割数 =GPU 数を変えて) 弱スケーリング性能を測定したもの。 いくつかの点での単位セル数も矢印と数値のペアで示す。

4 東北地方太平洋沖地震

ここで紹介する計算例では、全世界観測網の広帯域地震波形データを用いて推定した震源モデル[12]を用いる。遠方の観測点のデータを用いているため分解能に制限があり、やや長波長側の震源モデルになっている。この震源モデルは断層面上におよそ20kmおきに配置した格子点(23個×11個)に割り当てた点震源から構成される。東日本地域の3次元不均質速度構造モデルは、複数の既存構造モデル[13-16]をコンパイルして作成した。差分法計算のパラメータは次の通りである:格子間隔0.15 km、時間間隔 0.005 s、格子サイズ6400×3200×1600、時間ステップ数44000、周波数帯域の上限0.61 Hz。この計算ではTSUBAME-2.0の1000 GPUを利用した。これはTSUBAME-2.0の全GPUの約24%にあたる。 なお、計算時間は5768 s、演算性能は 33.2 TFlops である。図8に示した例よりも性能が低下したが、これは計算結果のファイル出力に伴うオーバーヘッドが大きいためと考えられる。この点も今後の改良が必要な部分である。

このようにして計算した地震波動場を可視化したものを図9と図10に示す[2]。 断層各部からの波動場が、地形や構造による散乱も加わって非常に複雑に干渉しながら伝播していく様子がわかる。また130秒後から150秒後にかけて、上向き速度と下向き速度の領域が顕著なペアになって福島県海岸線付近に現れ、震央付近から到来する波動のパタンを掻き消すような様子になっていることもわかる(図10、黄色の円形枠)。このペアの領域をもたらした波源域は福島県海岸線付近の深さ約60kmにあるやや大きなすべり領域と考えられる。このような領域は陸上観測点の強震動記録からも推定されており、福島県海岸線付近を始め数か所のSMGA (strong-motion-generation-area)が推定されている(例えば[17])。このような領域を含めた震源モデルの全体像について、研究を続ける必要がある。

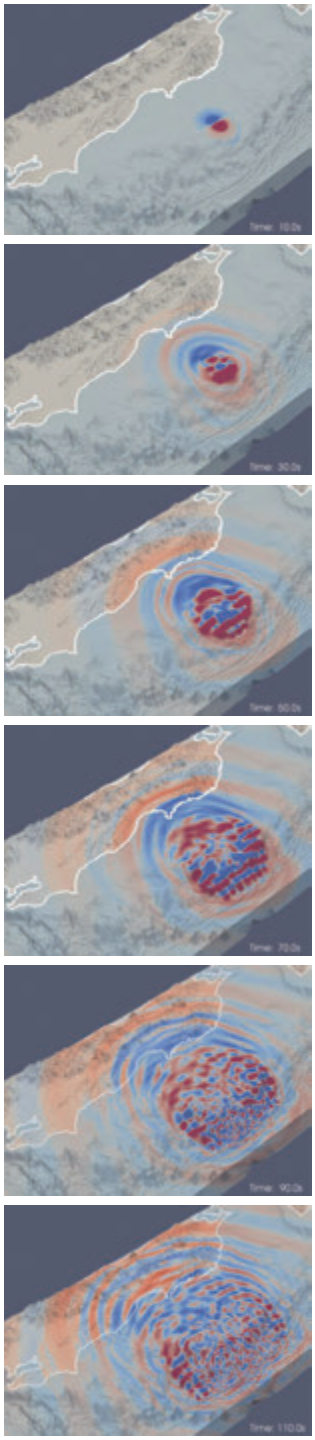


図 9
シミュレーションによる地震波伝播のスナップショット。地表および海底の固体側表面での鉛直地動速度を可視化したもの。赤と青はそれぞれ上方向、下方向への動きを示す。上から下にかけて、破壊開始時刻から 10 秒後、30 秒後、50 秒後、70 秒後、90 秒後、110 秒後のスナップショットを示す。海水中の音波伝播は計算で考慮されている。

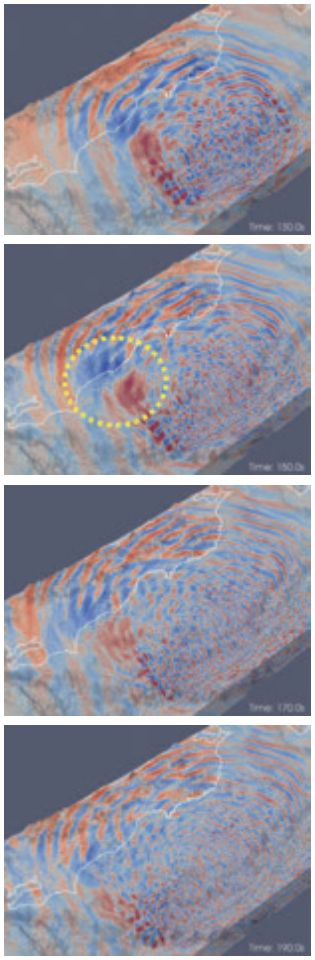


図 10 破壊開始点から 130 秒後、150 秒後、
170 秒後、190 秒後のスナップショット。

まとめ 5

地震波伝播シミュレーション、特に東北地方太平洋沖地震のような巨大地震のシミュレーションを行うために、マルチGPUを用いた並列差分法計算手法を開発した。今回、TSUBAME-2.0の1000 GPUを使うことによって、330億個の単位セルからなる領域について上限周波数が0.61 Hz であるようなシミュレーションを実用的な時間内で実行することができた。計算された地震波動場は、観測データの（やや長周期側での）特徴を定性的に再現している。このような計算が可能になったことから、今後は、日本列島の各地で観測された地震波波形データを用いた震源モデル解析を行うための『グリーン関数波形』をTSUBAME-2.0によって生成できる。そのようにして計算したグリーン関数波形を用いて震源モデルの改良と推定を進める必要がある。

謝　辞

構造モデルデータを提供してくださった各機関の方々に感謝申し上げます。また、TSUBAME-2.0における大規模計算（グランドチャレンジ）の機会をご提供くださった学術国際情報センターの方々に感謝申し上げます。この研究の一部に科学研究費補助金（23310122）を使用しました。

参考文献

[1] Okamoto, T., Takenaka, H., Nakamura, T. and Aoki, T. "Accelerating large-scale simulation of seismic wave propagation by multi-GPUs and three-dimensional domain decomposition", Earth, Planets and Space, vol. 62, no. 12, pp. 939-942 (2010).

[2] Okamoto, T., Takenaka, H., Nakamura, T., and Aoki, T. "Large-scale simulation of seismic-wave propagation of the 2011 Tohoku-Oki M9 earthquake", Proceedings of the International Symposium on Engineering Lessons Learned from the 2011 Great East Japan Earthquake, pp. 349-360 (2012).

[3] Okamoto, T., Takenaka, H., Nakamura, T. and Aoki, T. "GPU-accelerated simulation of seismic wave propagation", in GPU Solutions to Multi-scale Problems in Science and Engineering, Yuen, D., Wang, J., Johnsson, L., Chi, C.-H., Shi, Y. (Eds.), 250 pp., Springer, in press.

[4] Graves, R. W. "Simulating seismic wave propagation in 3D elastic media using staggered-grid finite differences", Bull. Seism. Soc. Am., vol. 86, no. 4, pp. 1091-1106 (1996).

[5] Yee, K. "Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media", IEEE Transactions on Antennas and Propagation", vol. 14, no. 3, pp. 302-307 (1966).

[6] Abdelkhalek, R., Calandra, H., Coulaud, O., Roman, J. and Latu, G. "Fast seismic modeling and reverse time migration on a GPU cluster", International Conference on High Performance Computing & Simulation, pp. 36-43 (2009).

[7] Micikevicius, P. "3D finite-difference computation on GPUs using CUDA", GPGPU-2: Proc. 2nd Workshop on General Purpose Processing on Graphics Processing Units, pp. 79-84, Washington DC, USA (2009).

[8] Michéa, D. and Komatitsch, D. "Accelerating a three-dimensional finite-difference wave propagation code using GPU graphics cards", Geophys. J. Int., doi: 10.1111/j.1365-246X.2010.04616.x (2010).

[9] 岡元太郎, 竹中博士, 中村武史. "GPUによる地震波伝播シミュレーション", 先進的計算基盤システムシンポジウム(SACSIS2010)論文集, pp. 141-142 (2010).

[10] 下川辺隆史・青木尊之, "次世代気象モデルのフルGPU計算 - TSUBAME2.0の3990 GPUで145 TFlops ー ", TSUBAME e-Science Journal, vol. 2, pp. 9-13 (2010).

[11] Takenaka, H., Nakamura, T., Okamoto, T. and Kaneda, Y. "A unified approach implementing land and ocean-bottom topographies in the staggered-grid finite-difference method for seismic wave modeling", Proc. 9th SEGJ Int. Symp., CD-ROM Paper No.37 (2009).

[12] Okamoto, T., Takenaka, H., Hara, T., Nakamura, T. and Aoki, T. "Rupture Process And Waveform Modeling of The 2011 Tohoku-Oki, Magnitude-9 Earthquake", American Geophysical Union, Fall Meeting, U51B-0038, San Francisco, USA (2011).

[13] Kisimoto, K. "Combined bathymetric and topographic mesh data: Japan250m.grd", Geological Survey of Japan, Open-file Report, No. 353 (1999).

[14] Fujiwara, H., Kawai, S., Aoi, S., Morikawa, N., Senna, S., Kudo, N., Ooi, M., Hao, K. X.-S., Hayakawa, Y., Toyama, N., Matsuyama, H., Iwamoto, K., Suzuki, H. and Liu, Y. "A study on subsurface structure model for deep sedimentary layers of Japan for strong-motion evaluation", Technical Note of the National Research Institute for Earth Science and Disaster Prevention, No.337 (2009).

[15] Baba, T., Ito, A., Kaneda, Y., Hayakawa, T. and Furumura, T. "3-D seismic wave velocity structures in the Nankai and Japan Trench subduction zones derived from marine seismic surveys", Abstr. Japan Geoscience Union Meet., S111-006, Makuhari, Japan (2006).

[16] Nakamura, T., Okamoto, T., Sugioka, H., Ishihara, Y., Ito, A., Obana, K., Kodaira, S., Suetsugu, D., Kinoshita, M., Fukao, Y. and Kaneda, Y. "3D FDM simulation for very-low-frequency earthquakes off Kii Peninsula", Abstr. Seism. Soc. Japan, P1-06, Hiroshima, Japan (2010).

[17] Kurahashi, S. and Irikura, K. "Source model for generating strong ground motions during the 2011 off the Pacific coast of Tohoku earthquake", Earth Planets Space, Vol. 63, 571-576 (2011).

● TSUBAME e-Science Journal No.6
2012 年 7 月 31 日　東京工業大学 学術国際情報センター発行 ©
ISSN 2185-6028
デザイン・レイアウト：キックアンドパンチ
編集：　TSUBAME e-Science Journal 編集室
青木尊之　ピパットポンサー・ティラポン
渡邊寿雄　佐々木淳　仲川愛理
住所：　〒 152-8550 東京都目黒区大岡山 2-12-1-E2-1
電話：　03-5734-2087　FAX：03-5734-3198
E-mail：tsubame_j@sim.gsic.titech.ac.jp
URL：　http://www.gsic.titech.ac.jp/

TSUBAME

TSUBAME 共同利用サービス

**『みんなのスパコン』TSUBAME共同利用サービスは、
ピーク性能 2.4PFlops、18000CPUコア、4300GPU搭載
世界トップクラスの東工大のスパコンTSUBAME2.0を
東工大以外の皆さまにご利用いただくための枠組みです。**

課題公募する利用区分とカテゴリ

共同利用サービスには、「学術利用」、「産業利用」、「社会貢献利用」の3つの利用区分があり、さらに「成果公開」と「成果非公開」のカテゴリがあります。
ご利用をご検討の際には、下記までお問い合わせください。

TSUBAME 共同利用とは…

他大学や公的研究機関の研究者の**学術利用**［有償利用］

民間企業の方の**産業利用**［有償・無償利用］

その他の組織による社会的貢献のための**社会貢献利用**［有償利用］

共同利用にて提供する計算資源

共同利用サービスの利用区分・カテゴリ別の利用課金表を下記に示します。TSUBAME 2.0における計算機資源の割振りは口数を単位としており、1口は標準1ノード（12 CPUコア, 3GPU, 55.82GBメモリ搭載）の3000時間分（≒約4ヵ月）相当の計算機資源です。
1000 CPUコアを1.5日利用する使い方や、100 GPUを3.75日利用する使い方も可能です。

利用区分	利用者	制度や利用規定等	カテゴリ	利用課金
学術利用	他大学または研究機関等	共同利用の利用規定に基づく	成果公開	1口：100,000円
産業利用	民間企業を中心としたグループ	「先端研究施設共用促進事業」に基づく	成果公開	トライアルユース（無償利用）
				1口：100,000円
			成果非公開	1口：400,000円
社会貢献利用	非営利団体、公共団体等	共同利用の利用規定に基づく	成果公開	1口：100,000円
			成果非公開	1口：400,000円

産業利用トライアルユース制度（先端研究施設共用促進事業）

東工大のスパコンTSUBAMEを、より多くの企業の皆さまにご利用いただくため、初めてTSUBAMEをご利用いただく際に、無償にてご試用いただける制度です。

（文部科学省 先端研究施設共用促進事業による助成）

詳しくは、下記までお問い合わせください。

お問い合わせ

●東京工業大学　学術国際情報センター　共同利用推進室
●e-mail　kyodo@gsic.titech.ac.jp　Tel. 03-5734-2085　Fax. 03-5734-3198
詳しくは http://www.gsic.titech.ac.jp/tsubame/をご覧ください。

GSIC
Global Scientific Information and Computing Center