11-NA03

GPGPU の地震ハザード予測シミュレーションへの適応性評価

青井 真（独立行政法人 防災科学技術研究所）

概要　昨年度までに TSUBAME2.0 を使用した単純なモデルによる弱スケーリングの性能評価で格子数が数億規模の比較的大きなモデルまで高いスケーラビリティを得られることができた。本年度は、より高い精度で地震ハザード予測をめざし、より規模の大きなモデルによる超並列計算のパフォーマンス検証を行うと同時に、大量の計算結果を伴うシミュレーションの出力アルゴリズムに関して基礎的な検討を行った。

1　研究の目的と意義

兵庫県南部地震を契機に、地震に関する調査研究の成果が国民や防災を担当する機関に十分に伝達され活用される体制になっていなかったという課題意識の下に、行政施策に直結すべき地震に関する調査研究の責任体制を明らかにし、これを政府として一元的に推進するため、地震防災対策特別措置法に基づき総理府に設置(現・文部科学省に設置された)機関として、地震調査研究推進本部が設置されている。地震調査研究推進本部の地震調査委員会において、地震に関する観測、測量、調査または研究を行う関係行政機関、大学等の調査結果を収集、整理、分析し、並びにこれに基づき総合的な評価を行っており、この中で地震動予測地図の高度化を進めている。日本周辺で発生する大地震に関して、地震のリスク評価の基礎となり得る精度で地震ハザードを予測できるよう、手法・モデルの高度化を目指している。そのために、必要な精度、分解能を持つ地盤構造の開発を行うとともに、高精度かつ汎用性のある地震波伝播（強震動）シミュレーション手法の開発に関する研究を行っている。

地震波伝播シミュレーションにおいて詳細な 3 次元地下構造を十分な精度で離散化し、短周期の地震波まで計算するためには細かな格子が必要であるため、実用的な計算においては格子数が数億から数十億に及ぶ規模のモデルを扱うことになる。近年の計算機環境の劇的な進歩の恩恵を受けたとはいえ、大きな計算機リソース（CPU パワー及びメモリ）を必要とし、実務等で一般的に使用可能な計算機では数日以上の計算となる事もしばしばである。このような状況を打開する手段として、多くの数値計算分野で広く使われるようになりつつある GPGPU（General Purpose Computation on Graphics Processing Unit）の利用が考えられる。

本研究では、GPU コンピューティングをスーパーコンピュータシステムに積極的に取り入れている TSUBAME 2.0 で地震波伝播シミュレーションを試行し、高度な地震ハザード・リスク評価への適応性を評価することを目的とする。

2　当拠点公募型共同研究として実施した意義

地震の被害を軽減するためには、個々人の地震への防災意識を高め、地震に対する備えを促すことが不可欠である。このため、防災科学技術研究所では、日本全国で発生する地震を対象として、地震調査研究の成果の集大成である地震動予測地図を高度化し、地震ハザード・リスク評価に関する研究を行うとともに、WebGIS 等の技術を用いて、地震ハザード・リスク情報、地下構造データ等の関連情報を網羅的に提供可能な地震ハザード・リスク情報ステーションを構築している。

本研究では東京工業大学との超大規模数値計算応用分野での共同研究を行なっており、これによ

り地震ハザード・リスク評価に用いられる地震波伝播シミュレーションに対する GPGPU に適用について、実務等に使用される汎用計算機環境から TSUBAME 2.0 のような大規模なスーパーコンピュータシステムまでの様々な規模の計算機環境における評価が可能となり、日本の防災力向上に大きく貢献することが期待される。

3　研究成果の詳細と当初計画の達成状況

3.1　GMS による波動場の計算

本研究では、不連続な食い違い格子（Aoi and Fujiwara, 1999）を用いた、空間四次・時間二次精度の差分演算子による実用コードである GMS（Ground Motion Simulator、青井・他、2004）をベースに GPGPU の適用性の評価を行なった。GMS は、防災科学技術研究所によってパッケージ化された、3 次元有限差分法(FDM)により地震波伝搬シミュレーションを行うためのツール群でおり、主に Fortran90 で書かれた差分計算ソルバはソースコードも公開されている。

Aoi and Fujiwara（1999）や青井・他（2004）は、大きさの異なる格子を組み合わせることにより効率的かつ高精度に計算を行うことの出来る不連続格子による差分法の定式化を提案した。差分法による地震波伝播シミュレーションを行う場合、格子サイズは計算すべき最短波長により決定されるため、モデルのごく一部のみが低速度の媒質である場合でも計算領域全体を小さな格子に分割せざるを得ず、大規模なモデル計算の大きな障害となっていた。軟弱（＝地震波速度が低い）な表層が存在する浅い部分（領域 I）の格子点間隔は細かく、深い部分（領域 II）の格子点間隔は領域 I の 3 倍の粗い格子点間隔を有する格子モデルを用いている（図 1）。2 つの領域はオーバーラップしており、波動場の連続性が保たれるよう内挿される。典型的な盆地構造モデルの計算において、均質な格子による場合と比較し数倍から十数倍程度効率がよいことが分かっている。

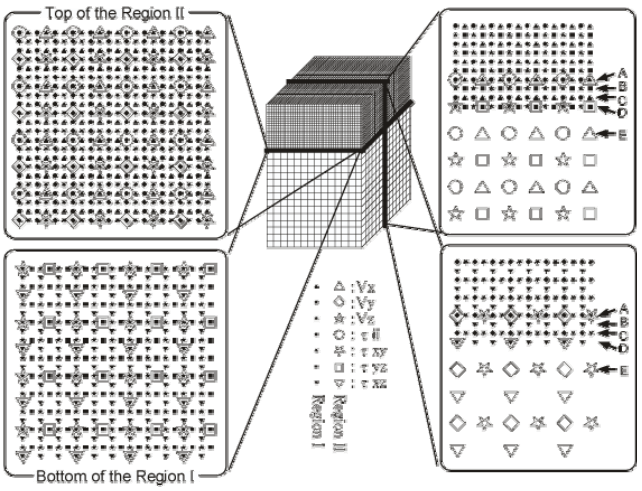

図 1　（中央）計算に用いる不連続格子。（右）不連続格子の垂直断面。領域 I と領域 II の接続部分で、内挿のために格子が重なっている。(左) 領域 II の最上面（A 面）と領域 I の最下面（D 面）における不連続格子の水平断面。

3.2　GPU での差分計算処理

GMS の差分法の計算処理部分については、ほぼ全て GPU 上で処理するように実装を行った。開発環境には、NVIDIA 社から提供されている CUDA（Compute Unified Device Architecture）を使用している。CUDA では、演算処理を担当する GPU と、その演算処理に必要なデータを格納するビデオメモリ（グローバルメモリ）により構成されるデバイスをアクセラレータとして用いることを前提としている(図 2 左)。

差分法の計算を行う際は、スレッド（GPU で計算処理を行う際の実行の最小単位）の集合であるブロックを 2 次元で構成し、計算の対象となる x-y 平面にブロックを敷き詰めるように配置している。ブロック内の各スレッドは z 方向の始点から終点まで 1 格子点ずつ計算処理を進め、3 次元領域全体の計算処理を行う（図 2 右)。差分法のように大量のメモリアクセスを伴う計算ではメモリバンド幅がボトルネックになることが多いため、GPU 上に搭載されている高速なレジスタやシェアードメモリをキャッシュのように使用して、低速なグローバルメモリへのアクセスを軽減する工夫を行っている。

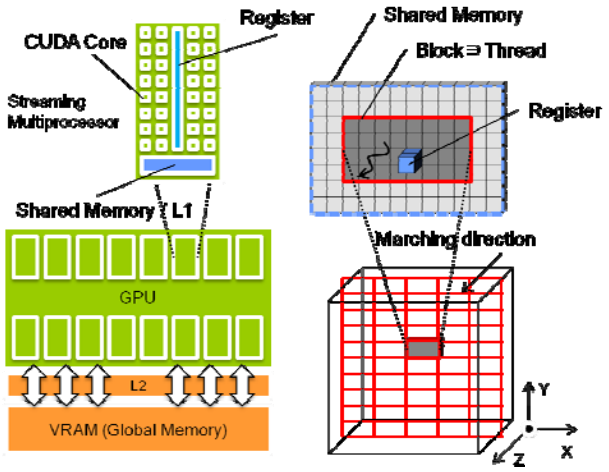

図 2　(左)GPU のアーキテクチャ、(右) GPU での計算処理

3.3　複数 GPU を用いた並列化

複数 GPU を用いた並列化においては、水平二方向（x 方向、y 方向）の領域分割法を用いている。分割した各部分領域には、それぞれ 1 つの GPU を割り付けて計算処理を行う。各部分領域には、隣接する部分領域のデータを収める袖領域を設け、データが更新された際は、各部分領域間で MPI(Message-Passing Interface)ライブラリを用いてデータを交換する。

GPU の演算速度は極めて高速であるために相対的に通信に要する時間の割合が大きくなり、CPU のみを用いて並列化した際と比べて、並列性能が低くなる傾向がある。本研究では、これを改善させるため GPU での計算処理と通信をオーバーラップさせて、通信に要する時間の隠蔽を行っている。一般に通信の隠蔽を行う際には、通信の対象となる袖領域の計算を事前に行い、内部領域を計算している間に並行して通信を行う手法がとられることが多い。しかしこのような方法では、袖領域を計算する際に GPU が苦手とする不連続なメモリアクセスが生じるため、袖領域の事前計算そのものが足かせとなってしまう場合がある（図 3)。本研究では、GMS が格子最適化手法として採用している不連続格子が 2 つの異なる格子サイズを持つ領域（領域 I、領域 II）からなることに注目し、一方の領域を計算する間に他方の領域の通信を行うことで袖領域のみの計算を別途行う必要性を回避した（図 4)。なお、この方法は、不連続格子を用いていないアルゴリズムにも拡張する可能で、事前計算などの余分なオーバーヘッドが無いことから、通信を隠蔽する際には有効な手段と言える。

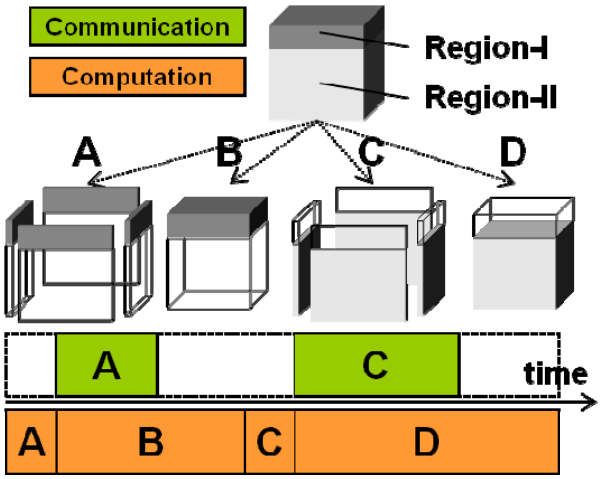

図 3　複数 GPU での並列計算(水平方向の領域分割に並列化)

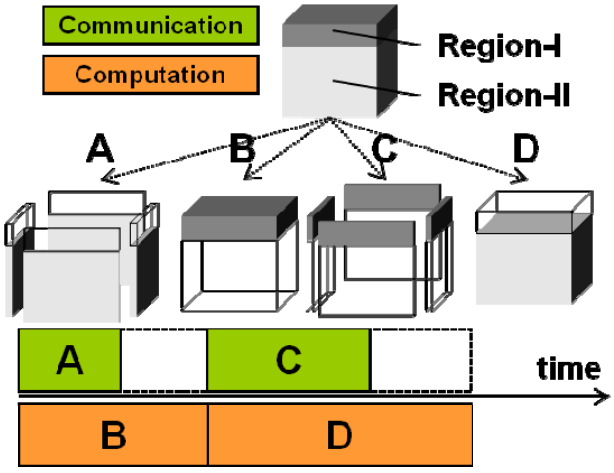

図 4　本研究で開発した通信時間の隠蔽方法（袖領域の事前計算が必要ない）

3.4　並列計算の性能試験

GPU での実行に対応した差分計算ソルバについて TSUBAME 2.0 を用いて性能試験を行った。使用したノードには GPU に NVIDIA Tesla M2050 x3、CPU に Intel Xeon X5670 2.93GHz x2 が搭載されており、ノード間は InfiniBand 4x QDR にて接続されている。

まず、単体 GPU での性能を評価するため、GPU

版プログラム、および比較用として CPU 版プログラムの 1 並列での性能試験を実施した。計算モデルには領域 I が 420x420x100、領域 II が 140x140x200 の 21,560,000 格子からなるモデル（Unit420）を使用し、浮動小数点演算は GPU 版プログラム、CPU 版プログラム共に単精度となる。比較用の CPU 版プログラムの実行には Intel Xeon X5670 の 1CPU コアを使用し、コンパイラには Intel Compiler 11.1 を用いている。

単体 GPU での性能試験の結果から、CPU 版プログラムに比べて GPU 版プログラムは 20.4 倍の性能となり、相当な高速化が図られたことが確認された（表 1）。また、NVIDIA CUDA に付属する性能解析ツールを用いて GPU-VRAM（グローバルメモリ）間のメモリ転送速度を計測したところ、最大で理論性能の 70%以上となることから、律速の要因が演算処理ではなくメモリ転送速度であることが確認された。計算の主要部分において、メモリ転送が理論性能に対して十分に高速に行われており、ハードウェアの性能を生かしていることから、更なる高速化の余地はそれほど大きくはないと考えられる。

表 1　単体 GPU での性能評価

Unit420 1000ステップ

プログラム	実行時間 (sec)	FLOPS (FP32)	高速化率
GPU版プログラム	5.48E+01	7.95E+10	20.42
CPU版プログラム	1.12E+03	3.89E+09	1.0

※時間計測においてはファイル入出力等を除いた差分法の計算処理部分のみを対象とした。

続いて、複数 GPU を用いた並列性能の評価を実施した。計算モデルには、Unit420 を単位モデルとし、これを水平方向に 2x2、3x3、4x4、8x8、10x10、16x16、32x32 個並べて、全体の格子数がそれぞれ 4 倍、9 倍、16 倍、64 倍、100 倍、256 倍、1024 倍となるようなモデルを使用した。

複数 GPU での性能試験の結果の結果から弱スケーリングに関しては、ほぼ線形（実際には線形を超える）の性能を得られていることが分かる（図 5）。一方、モデルを単位モデルに固定し使用する GPU 数を増加させる強スケーリングに関しては、4 GPU では 3.2 倍の演算性能が得られるものの、16 GPU では 7.3 倍の性能となり、効率が落ちてゆくことが分かった。これは、並列数の増加に伴い各 GPU が担当する格子数が減少するため、相対的に通信の時間が大きくなること、および GPU で生成されるスレッド数が減少し計算の効率が低下したことが要因として考えられる。通常行われるシミュレーションのタイムステップ数は数万ステップであり、モデルサイズに応じた計算機資源が確保されている場合には計算に要する時間（turn around time）は数分から数十分程度となるため、過剰な並列度による更なる高速化の必要性は実用的にはほとんど無いと考えられる。

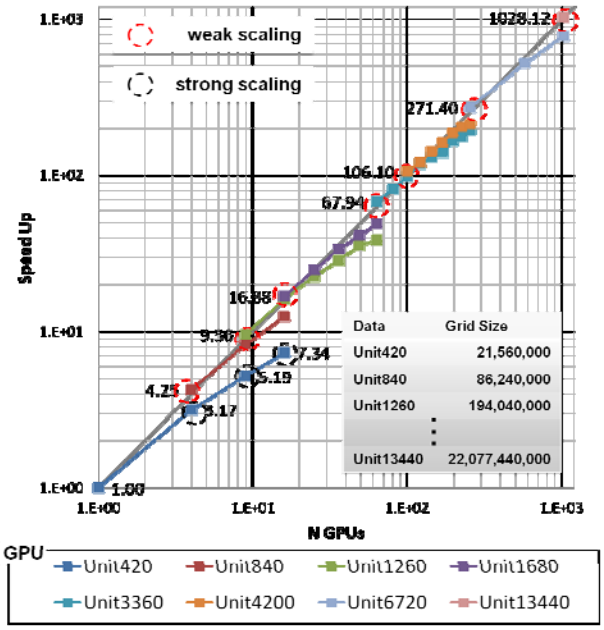

図 5　並列数 － 高速化率の関係

また、実行時間と浮動小数点演算数を用いて FLOPS（Floating-point Operations Per Second）を算出した（図 6）。GPU 版プログラムでは、Unit6720（約 55 億格子）の 256GPU による性能が 21.1TFLOPS、Unit13440（約 220 億格子）の 1024GPU による性能が 79.7TFLOPS になることが確認された。CPU 版プログラムでは Unit13440 の 1024 並列での性能が 2.3TFLOPS であることから、約 34 倍の

性能が得られていることが分かる。

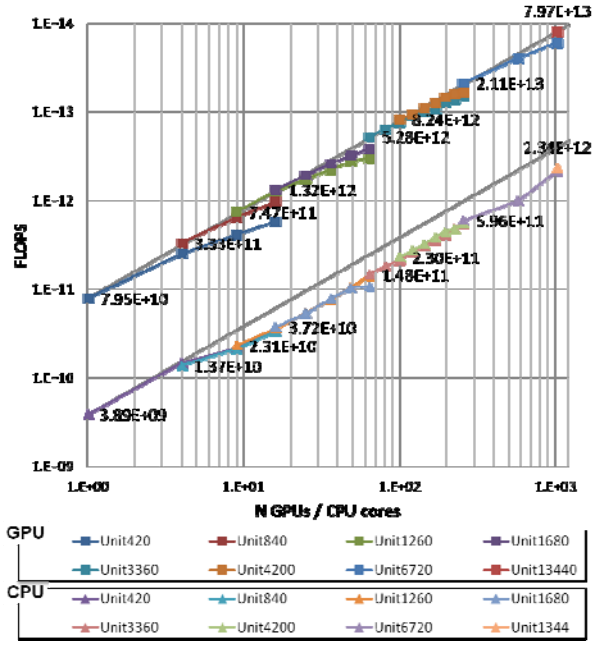

図 6　並列数 － FLOPS の関係
（GPU：GPU 版プログラム、CPU：CPU 版プログラム）

3.5　ファイル出力機能の追加と高速化

GPU の演算速度は極めて高速であるため、各タイムステップの Open/Close 処理などオーバーヘッド処理が差分計算ソルバ全体のボトルネックとなることが予想されたため、CPU のメモリを利用して出力のバッファリングを行うことにより、出力時間を低減する実装を行い評価した。

GMS は計算結果の出力に、ネットワーク透過で自己記述的なバイナリファイルを入出力できる HDF5 ライブラリ（http://www.hdfgroup.org/HDF5）を利用している。GMS の差分計算ソルバは、用途に応じていろいろな出力条件を指定できるが、ここでは最も自由度の高い 1D Dump 機能（任意の位置の計算結果を出力刷る機能）を GPU 版の差分計算ソルバに実装した。この出力機能では、毎タイムステップに指定した各観測点での波形を HDF5 でファイルに追記する。

モデル規模が大きくなると出力のすべきデータも膨大になるが、一般的に出力の並列性能を高めることは困難である。また、GPU での計算処理が極めて高速であるため、相対的にファイル出力に要する時間の割合が高くなってしまう。そこで、本研究では試験的に単体 GPU での実行に対応したプログラムに対して、GPU での計算処理とファイル出力を同時に行う隠蔽アルゴリズムの開発を行った。GPU 側の処理は基本的に非同期に進行する。そのため、GPU に計算処理を投入した後、前のタイムステップのファイル出力を行うことで、GPU での計算処理とファイル出力をオーバーラップさせ、ファイル出力に要する時間を隠蔽することが出来る（図 7）。単体 GPU ではこのようなアプローチは有効であるが、並列計算を行う場合には、既に述べた通信時間の隠蔽と重複するため、大量の出力に対しては有効な手法とは言えない。

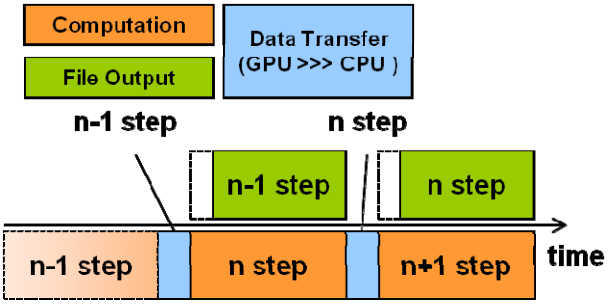

図 7　ファイル出力の隠蔽方法

ここでは、GPU での計算結果を CPU に転送後、メモリを利用して出力のバッファリングを行うことにより、出力時間を低減する実装を行い評価した。評価方法としては、複数 GPU を使用した際の GMS ソルバの実行時間を測定し、並列度の変化に伴う性能評価を実施した。約 2.2 千万格子からなるモデル Unit420 を単位モデルとし、これを水平方向に 2x2、4x4、8x8 個並べて、全体の格子数がそれぞれ 4 倍(Unit840)、16 倍(Unit1680)、64 倍(Unit3360)となるようなモデルを使用して、出力のパフォーマンスを評価した。出力対象観測点数は全モデルで一定とし、現在実用上最大規模のモデル計算で設定している規模の 64 万点とした。

まず、ディスク I/O が比較的低速な PC (CPU:Intel Xeon E5520, GPU:NVIDIA Tesla S2050, HDD:SATA300 1TB 7200rpm)を用い、Unit420 で評価を実施した。図 8 に書き出しに関

わるデバイス-ホスト間（D2H）、ホスト-ホスト間（H2H）、ホスト-ファイル間(H2F)、それぞれの転送時間を個別に計測して示す。バッファリングをしない場合（ステップ数 1）に比べ、バッファリングステップ数を増やしていくと、ホスト-ファイル間の転送時間が減少し、バッファリングステップ数 100 では約 1/3.5 となった。PC 環境では、1 回の書き込みデータ量を増やすバッファリングを用いた実装が効果的であることがわかった。

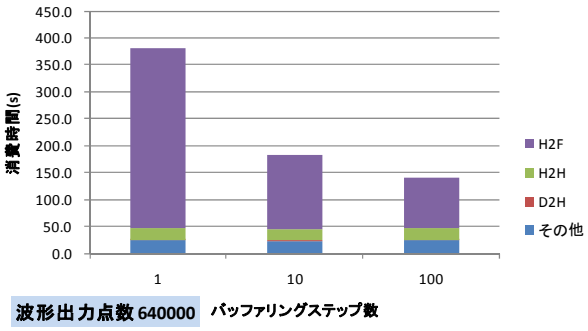

図 8　波形出力点数 640,000 の際のバッファリング効果（PC ローカルディスク）

次に TSUBAME2.0 において、Unit420 を用いて評価した結果を図 9 に示す。TSUBAME2.0 の場合、バッファリングステップ数を増やすとホストーファイル間の転送時間はむしろ増大することがわかった。これは、TSUBAME のファイルシステムが極めて高スループットで、かつ Open/Close 処理が高速であるため、I/O よりもバッファリング処理に係るオーバーヘッドが比較的高コストになっていることが原因と考えられる。

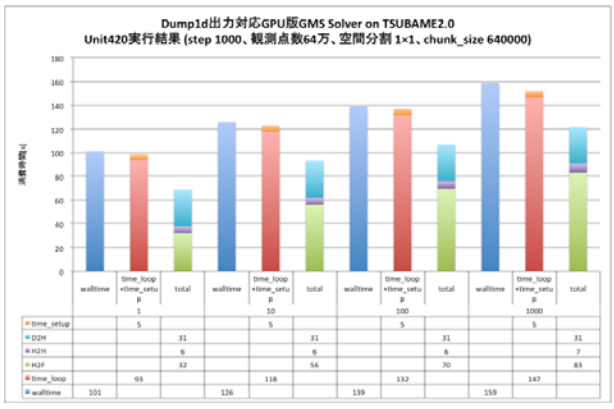

図 9　ファイル出力時間の評価結果

4　今後の展望

ハザード評価を行う場合、面的に計算結果を出力刷る必要がある。PC 環境のように出力環境が比較的低速な場合にはバッファリング等の工夫により出力が高速化されることが分かったが、TSUBAME2.0 のように高速な出力環境を備えるシステムにおいては、このようなアプローチでは不十分であることが分かった。大量の面的出力を得るには、現在のパフォーマンスでは計算時間よりも出力時間の方が長くかかることから、根本的に出力容量を減らすことが、計算全体のターンアラウンドタイムを短縮には必要であることが分かった。ポスト処理の 1 つであるフィルタ処理を行うためには十分細かい時間間隔での出力が必要であるが、ソルバにおける計算と同時にフィルタ処理を行うことで、出力量を劇的に削減出来る可能性がある。そのためには、GPU 側にデシメーション処理を逐次フィルタとして実装する必要がある。今後、ソルバ内外のファイル I/O を最適化することにより、シミュレーションだけでなくフィルタ処理、可視化処理等のポスト処理を含む全ターンアラウンドタイムを短縮し、より実用的なシミュレーション環境を構築し、実際の地球をモデル化した非常に大きな不均質性をもつ地下構造モデルや面的広がりをもつ断層モデルを用いた現実的で大規模モデルによるシミュレーションを目指す。

5　研究成果リスト

(1) 学術論文

(2) 国際会議プロシーディングス

S. Aoi, N. Nishizawa, T. Aoki (2012) Large scale simulation of seismic wave propagation using GPGPU, 15th World Conference on Earthquake Engineering, in Lisbon「投稿中」.

(3) 国際会議発表

(4) 国内会議発表

前田宜浩・森川信之・青井真・藤原広行, 2011, 南海トラフの巨大地震による長周期地震動に関する検討, 日本地震学会秋季大会, P2-51.

(5) その他（特許, プレス発表, 著書等）