様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　２ 年　 ６ 月 ２５ 日現在

機関番号：１２６０１
研究種目：基盤研究(C)（一般）
研究期間：2017～2019
課題番号：１７Ｋ００１６５

研究課題名（和文）新しい動的負荷分散を用いたGPUスパコン向け適合細分化格子法フレームワークの開発

研究課題名（英文）Development of a framework for adaptive mesh refinement on GPU supercomputers using a novel dynamic load balancing.

研究代表者

下川辺　隆史（Shimokawabe, Takashi）

東京大学・情報基盤センター・准教授

研究者番号：４０６３６０４９

交付決定額（研究期間全体）：（直接経費）　　3,500,000円

研究成果の概要（和文）：近年、大規模GPU計算が可能となり、広大な計算領域の場所によって求められる精度が異なる問題に有効な手法が要求されている。本研究では、GPUが得意なステンシル計算を活用しながら、高精度が必要な領域を局所的に高精細にできる適合細分化格子法(AMR法) を様々なアプリケーションへ適用できるAMRフレームワークを構築した。GPU スパコンに適した動的負荷分散手法、通信削減技術、時間発展の最適化などを開発し導入した。AMRフレームワークに基づいた3 次元圧縮性流体計算で高い並列化効率を達成し、その有用性を示した。

研究成果の学術的意義や社会的意義
ペタスケールのスパコンでは、低消費電力かつ高性能を達成するため数千台を超えるGPUが搭載され、日本、米国などで稼働している。格子計算はスパコンを利用する代表的なアプリケーションで、局所的に高精細な大規模計算を実現させる意義は大きい。大規模な格子計算に向けて、通信やデータ移動の少ないアルゴリズムの開発は必須であり、スパコンに必須な通信隠蔽・削減技術と併用して、高性能AMR法を達成する試みの意義は大きい。本研究は、個別アプリケーションに特化したAMR法を構築するものでなく、汎用フレームワークを構築するものであり、大規模GPUアプリケーションの開発を支援する基盤技術となり、波及範囲は広い。

研究成果の概要（英文）：Recently grid-based physical simulations with multiple GPUs require effective methods to adapt grid resolution to certain sensitive regions of simulations. In this research, we have developed a high-productivity framework for adaptive mesh refinement (AMR); the AMR method is one of the effective methods on GPU to compute certain local regions that demand higher accuracy with higher resolution. This framework allows us to apply AMR to various stencil-based applications on GPU supercomputers. We have developed and implemented a dynamic load balancing method for GPU supercomputers, communication reduction techniques, and optimization techniques for time integration computations to enhance the framework. The 3D compressive fluid simulation based on this proposed framework has achieved a high parallel efficiency and demonstrated the high productivity of the framework.

研究分野：格子法に基づいた大規模物理計算

キーワード：適合細分化格子　動的負荷分散　高生産フレームワーク　高性能計算　スーパーコンピュータ　GPU

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９（共通）

１．研究開始当初の背景

研究代表者らは、格子に基づくアプリケーションを GPU (Graphics Processing Units)で実行する研究に取り組んできた。GPU は元々は画像処理用のプロセッサであったが、従来のプロセッサである CPU と比べ消費電力当たりの演算性能が高いため、 ペタスケールのスパコンでは GPU を大規模に搭載している。

大規模 GPU 計算が可能となり、近年、広大な計算領域の場所によって求められる精度が異なる問題に有効な手法が要求されている。GPU 計算では、GPU が得意なステンシル計算を活用しながら、高精度が必要な領域を局所的に高精細にできる適合細分化格子法 (Adaptive Mesh Refinement; AMR 法)が有効である。

GPU は大規模計算に適するが、その開発コストは高い。申請者らは、通常の C++を記述するだけで、 GPU スパコンに必須な最適化をアプリケーションへ簡便に導入できる構造格子用フレームワークを提案した。フレームワーク技術は GPU 実アプリケーションを高生産に開発する上で有用なことを示した。本研究では、このフレームワークを発展させ、 GPU スパコンの複数 GPU を用いた AMR アプリケーションを高生産に開発する基盤となる AMR 法フレームワークの構築を進める。

複数 GPU による AMR 法では、局所的に解像度が刻一刻と変化するため、GPU 間で計算負荷を動的に均等にすることが必須である。空間充填曲線を用いた動的負荷分散では、3 次元空間に配置された格子を 1 次元的に扱い、GPU 間で計算負荷を均等分配する。高精細領域が移動する度にスパコン全体にわたり格子データのソートや GPU 間の移動が発生し性能低下を招き、課題となっている。この解決には、各 GPU の計算負荷を動的に把握・予測し、これらをほぼ均等にしながら、各 GPU の扱う格子配置の変形を小さく保ち、データ移動を抑え、 アプリケーション実行時間の最小化を目的とした負荷分散を開発する必要がある。本研究では、これを導入した AMR 法フレームワークを構築する。

２．研究の目的

研究代表者らは、アプリケーション開発者の視点で、GPU スパコン上で高精度が必要な領域をより高精細な格子で計算できる高生産・高性能フレームワークを開発する。本研究では数千台を超える GPU を搭載したスパコン上で (1) 機械学習と通信隠蔽技術により実行時間の最小化を目的とした動的負荷分散技術を確立する。 これを基盤として、 (2) GPU スパコン上で適合細分化格子法 (AMR 法)を確立し、局所的に 100 倍の高解像度となる計算を実現することを目指す。この手法を様々なアプリケーションへ適用可能にする (3) GPU スパコンに向けた高性能・高生産 AMR 法フレームワークを構築する。フレームワークの開発を通して、AMR 法の適用技術を確立する。

３．研究の方法

本研究では、GPU スパコンで局所的な領域を高精細とできる適合細分化格子法(AMR 法)フレームワークを開発するため、 まず(1)通信削減手法を併用する動的負荷分散手法を開発し、これを基盤とし AMR 法を構築する。平行して、(2) 既存フレームワークを拡張し、高性能な AMR アプリケーション開発に必要なデータ構造と計算機構を構築する。さらに、 (3) 高精細な領域が大きく 移動する計算に対し有効な予測型動的負荷分散技術を開発する。これらを統合して、(4)大規模 GPU 計算のための AMR 法フレームワークを完成する。完成したフレームワークによって解析対象を高精細に計算する大規模アプリケーションが開発できることを検証するため、(4) フレームワークを様々なアプリケーションへ適用し、大規模 GPU 計算を実施する。

４．研究成果

本研究では、AMR データ構造と複数 GPU における管理手法、ステンシル計算関数の実行機構、時間ブロッキング法を用いた通信回数削減手法、複数木構造に基づいた動的負荷分散を開発した。これらを導入して、複数 GPU 向け AMR 法フレームワークを構築した。これを用いた 3 次元圧縮性流体計算において、東京工業大学の TSUBAME3.0 の 288 GPU を用い、84%の並列化効率を達成することに成功した。

ペタスケールのスパコンでは、低消費電力かつ高性能を達成するため数千台を超える GPU が搭載され、日本、米国などで稼働している。格子計算はスパコンを利用する代表的なアプリケーションで、本研究により局所的に高精細とした AMR 計算をフレームワーク技術を用いることで高い生産性で GPU 上で実現できることを示した意義は大きい。また、AMR 法フレームワークの開発を通して、高性能アプリケーションを高生産に開発する方法について有益な知見が得られた。

大規模に GPU を用いた AMR 計算では、当初想定していたよりも GPU 間の通信が性能低下に大きな影響を与えることがわかった。本研究では、通信回数削減技術や新しい領域分割法の導入により高度化を進めたが、さらなる最適化の余地があり、これは今後の課題である。

以下では主な研究成果について説明する。

(1)AMR 法フレームワーク

①AMR データ構造と複数 GPU における管理

本フレームワークの対象とする AMR 法では構造格子を再帰的に細分化し、その空間的配置を

木構造で表す。木構造の各リーフノードには、一つの格子ブロックを割り当てる。格子ブロックは典型的には 2 次元で 16 × 16 格子程度である。3 次元計算では八分木、2 次元空間では四分木となる。GPU は連続したメモリ領域へアクセスするときに高い実行性能となるため、格子ブロックは物理変数ごとに一つの大きな連続メモリ領域に確保する。各リーフノードは、直接は格子ブロックを保持せず、格子ブロックを特定する ID を保持する。この ID から割り当てられた格子ブロックの連続メモリ領域における位置を求め、格子ブロック上のデータを参照する。

複数 GPU による計算では MPI で並列化され、各プロセスが計算領域全体の木構造を表現する Field 型データ を保持する。各 MPI プロセスで発行する(1)各リーフに対する解像度の変更の指示、 (2)特定のリーフのデータをある GPU から別の GPU へ移行させる指示、は発行後に、MPI 通信によって全プロセスで共有する。共有後に、各プロセスは自分の保持する計算領域全体を表現する木構造を変更する。各プロセスの木構造自身は通信により明示的に同期することはないが、木構造を変更する上記の二つの「指示」を共有することで、全プロセスは常に同一の木構造を保持する。この計算領域全体の木構造の情報をもとに、適切に境界領域の交換などを行う。

②ステンシル計算関数の定義と実行の概要

AMR 法フレームワーク上でステンシル計算を定義する方法と実行方法について説明する。本フレームワークでは、ステンシル計算は、フレームワークの提供する MArrayIndex3D 等を用い、C++11 で導入されたラムダ式を使い定義する。当初は C++式テンプレートを活用したが、GPU とは異なるプロセッサである Xeon Phi では高い性能が得られないことが判明したため、フレームワークの汎用性を高めるため、C++11 のラムダ式を用いるようにフレームワーク全体を再構築した。フレームワークは、データ構造として MArray を用いることで、効率的な記述を実現する。MArray は、 大きさと位置の情報を持つ Range3D 型と配列を保持するクラスであり、この配列を連続メモリ領域として使う。3 次元の拡散計算では、次のようにステンシル計算関数を定義し実行することができる。

```
Engine_t engine;
engine.run(amrcon, inside, LevelGreaterEqual(1),
  [=] __host__ __device__ (const MArrayIndex3D &idx, const float *f, float *fn) {
     const float fn = cc*f[idx.ix()] + ce*f[idx.ix(1,0,0)] + cw*f[idx.ix(-1,0,0)]
       + cn*f[idx.ix(0,1,0)] + cs*f[idx.ix(0,-1,0)]+ ct*f[idx.ix(0,0,1)] + cb*f[idx.ix(0,0,-1)];
   }, idx(fa.range()), ptr(&fa), ptr(&fan));
```

engine は、第 1 引数に AMR データ構造を表す Field などを保持するオブジェクト、第 4 引数にラムダ式で表されたステンシル計算を取り、これに第 5 引数以降を渡す。fa、fan は MArray 型、inside は Range3D で、ptr() はステンシル計算関数中で MArray の inside の開始点のポインタを取得する。engine は、ステンシル関数を第 2 引数で渡された inside 領域内に対し、第 3 引数の条件を満たすレベルの格子ブロックに対して適用する。LevelGreaterEqual(1) を指定することで、このステンシル計算関数は、レベル 1 以上の格子ブロックに適用される。実行する格子のレベルを選択できるようにしたことで、異なる解像度（レベル）により異なる時間ステップ幅をもつ格子ボルツマン法へも適用可能となった。

③時間ブロッキング法を用いた通信回数削減手法

複数 GPU 計算では、時間発展させるためには、同一 GPU 内における格子ブロック間の袖領域データ交換とともに、異なる GPU に割り当てられた格子ブロックの袖領域データも交換する必要がある。図 1 に袖領域データ交換の手順を示す。ステンシル計算に必要な格子ブロックを隣接 GPU から転送するとき、まずフレームワークは、各プロセスにおいて、連続メモリ領域から未使用な格子ブロックの一部を一時領域として確保する。次に、MPI 通信を利用して実際に隣接 GPU からステンシル計算に必要となる格子ブロックを転送し、一時領域へ保存する。各プロセスでのステンシル計算は、この一時領域を参照して実行される。

異なる GPU 間の袖領域の交換では、ステンシル計算で必要となる格子ブロックの袖領域だけ

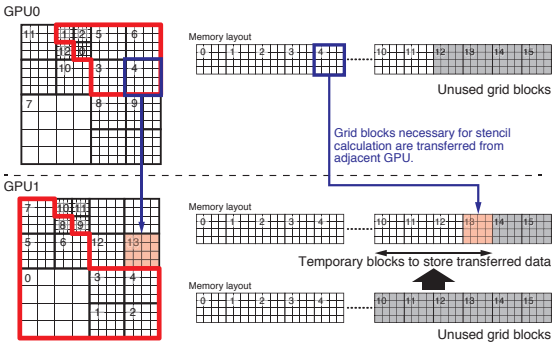

図 1 異なる GPU 間における格子ブロック間の袖領域データ交換

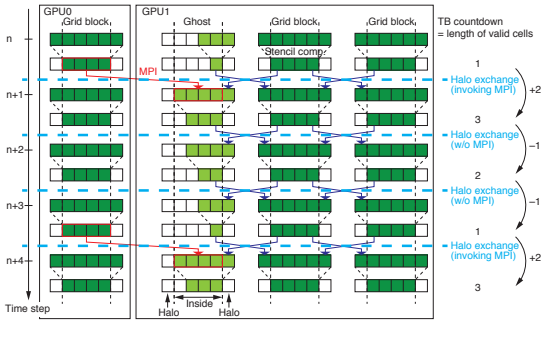

図 2 カウントダウン方式を利用した時間ブロッキング法の AMR 法フレームワークへの導入

でなく、格子ブロックの全領域のデータを転送する（図 1 のオレンジ色の領域）。従来の実装では、このデータ転送により大幅な性能低下につながっていたが、ステンシル計算では複数の時間ステップに必要な通信をまとめて実行できる時間ブロッキング法が有効であることがわかり、図 2 に示すように、直交格子用フレームワークで導入したカウントダウン方式の時間ブロッキング法の実行方法を AMR 法に適用し導入することに成功した。カウントダウン方式の時間ブロッキング手法を用いることで、フレームワークを使う側のユーザコードはほぼ変更を行う必要はなく、通信による性能低下を抑えることに成功した。

④複数木構造に基づいた動的負荷分散

AMR 法では、時間発展とともに高解像度となる局所的な領域の大きさや位置が変化する。複数 GPU による AMR 法では、性能低下を抑えるために、常にそれぞれの GPU にほぼ同数の格子ブロックが割り当てられるようにして、計算負荷を均等にする必要がある。

フレームワークは、格子ブロックをある GPU から他の GPU へ移行させることを指示する関数を提供する。格子ブロックのデータ移行前に各 MPI プロセスは、全リーフノードの格子ブロックの移行元と移行先を共有する。その後、実際の格子ブロックのデータ移行を並列で実行する。本フレームワークでは任意の物理空間を複数の木構造で表現する。格子ブロックは、全ての木構造を順に辿りながら、木構造上の全リーフノードへ割り当てられる。このため、同一の MPI プロセスが管理する格子ブロックは木構造の空間的分布に従い局在化する。フレームワークでは GPU が管理する格子ブロック数に偏りが生じ、ある閾値を超えた場合、全木構造を最初から順に辿り、その上の全リーフノードを各 GPU へ再割り当てする。木構造自体の空間的位置は変わらないため、この再割り当ては効率的に行うことができる。本研究では明示的に全領域にわたる空間充填曲線を導入することなく動的負荷分散に成功している。

領域分割は木構造単位と関係なく各 GPU の担う計算量が均等となるように分割すると、各 GPU の担う計算領域の幾何学的形状が複雑となる。これにより各領域の表面積が増大するために、通信パターンの複雑化とともに通信量が増加し、性能低下の要因となる。そこで、均等に配置された木構造を単位とした領域分割方法を導入し、性能向上に貢献した。

(2)3 次元圧縮性流体計算への適用と実行性能

複数 GPU に対応した AMR フレームワークの実行性能を確認するため、3 次元 3 次精度風上手法を用いた圧縮性流体計算へ適用する。

図 3 に東京工業大学の TSUBAME3.0 スパコンの 8 ノードを用い、各ノードから 4 台、合計 32 台の NVIDIA Tesla P100 GPU を用いた圧縮性流体計算によるレイリーテイラー不安定性の計算結果を示す。2048 × 2048 × 4096 の均一格子と同等の計算領域を 5 レベルの AMR を適用する。1 つの格子ブロックは 203 格子で、最大格子の幅は最小格子の幅の 16 倍となる。

図 4 は、この計算が各時間ステップに要した計算時間を示す。時間ブロッキング手法(TB)の有無で測定した。導入した時間ブロッキング法を用いることで、GPU 間の袖領域交換手法の実行性能が向上し、各時間ステップに要する計算時間が短くなっている。同等の計算を均一格子で行う場合、1 ステップで 2.7 秒かかるため、AMR 法により大幅な実行時間の減少が確認された。なお、動的負荷分散のためのデータ移行は、200 ステップに一度実行する。

GPU 数に合わせて水平方向の空間を拡大する弱スケーリングでは、TSUBAME3.0 の 288GPU を利用した計算が可能であることを示した。288GPU の結果は 8GPU に対して 84%の並列化効率を達成した。

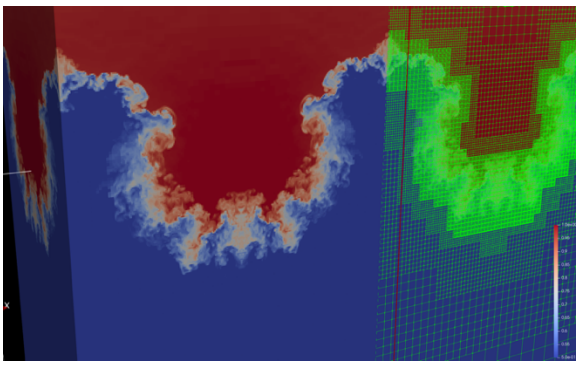

図 3 AMR 法フレームワークを用いた複数 GPU による 3 次元圧縮流体計算によるレイリーテイラー不安定性。格子状の緑色線は各リーフノードが持つ格子ブロックを表す。

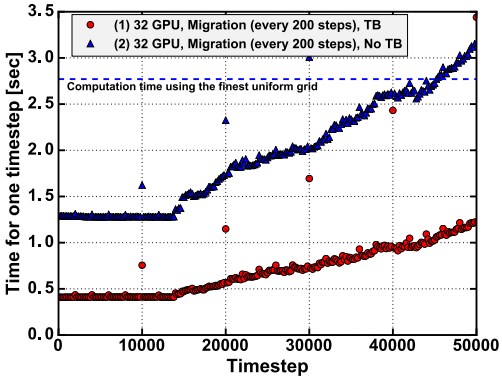

図 4 AMR 法フレームワークを適用した 3 次元圧縮流体計算における各時間ステップの計算時間

５．主な発表論文等

〔雑誌論文〕　計3件（うち査読付論文　3件／うち国際共著　0件／うちオープンアクセス　0件）

１．著者名 Shimokawabe Takashi、Onodera Naoyuki	４．巻 11536
２．論文標題 A High-Productivity Framework for Adaptive Mesh Refinement on Multiple GPUs	５．発行年 2019年
３．雑誌名 International Conference on Computational Science (ICCS) 2019	６．最初と最後の頁 281～294
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1007/978-3-030-22734-0_21	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

１．著者名 Takashi Shimokawabe, Toshio Endo, Naoyuki Onodera and Takayuki Aoki	４．巻 2017
２．論文標題 A Stencil Framework to Realize Large-scale Computations Beyond Device Memory Capacity on GPU Supercomputers	５．発行年 2017年
３．雑誌名 2017 IEEE International Conference on Cluster Computing (CLUSTER)	６．最初と最後の頁 525-529
掲載論文のDOI（デジタルオブジェクト識別子） https://doi.org/10.1109/CLUSTER.2017.97	査読の有無 有
オープンアクセス オープンアクセスではない、又はオープンアクセスが困難	国際共著 －

〔学会発表〕　計11件（うち招待講演　0件／うち国際学会　6件）

１．発表者名 下川辺 隆史, 小野寺 直幸
２．発表標題 複数GPUを用いた高精細計算を実現するAMR法フレームワークの開発
３．学会等名 第23回計算工学講演会
４．発表年 2018年

１．発表者名 Takashi Shimokawabe
２．発表標題 AMR Framework with multiple GPUs to Realize Effective High-resolution Simulations
３．学会等名 GPU Technology Conference (GTC) 2019（国際学会）
４．発表年 2019年

１．発表者名
下川辺 隆史，青木 尊之，小野寺 直幸

２．発表標題
高精細計算を実現するAMR法フレームワークの開発

３．学会等名
第22回計算工学講演会

４．発表年
2017年

１．発表者名
Takashi Shimokawabe, Takayuki Aoki, and Naoyuki Onodera

２．発表標題
An AMR Framework for Realizing Effective High-Resolution Simulations on Multiple GPUs

３．学会等名
18th SIAM Conference on Parallel Processing for Scientific Computing（国際学会）

４．発表年
2018年

１．発表者名
Takashi Shimokawabe, Takayuki Aoki and Naoyuki Onodera

２．発表標題
AMR Framework for Realizing Effective High-Resolution Simulations on GPU

３．学会等名
GPU Technology Conference (GTC) 2018（国際学会）

４．発表年
2018年

１．発表者名
下川辺 隆史、小野寺 直幸

２．発表標題
AMR 法フレームワークの大規模 GPU 計算に向けた発展

３．学会等名
第24回計算工学講演会

４．発表年
2019年

１．発表者名 Takashi Shimokawabe and Naoyuki Onodera
２．発表標題 AMR Framework to Realize Effective High-resolution Simulations on Multiple GPUs
３．学会等名 International Conference on High Performance Computing in Asia-Pacific Region (HPCAsia) 2020（国際学会）
４．発表年 2020年

１．発表者名 Takashi Shimokawabe and Naoyuki Onodera
２．発表標題 AMR Framework for Large-Scale Simulations on Multiple GPUs
３．学会等名 SIAM Conference on Parallel Processing for Scientific Computing 2020（国際学会）
４．発表年 2020年

〔図書〕　計0件

〔産業財産権〕

〔その他〕

―

６．研究組織

	氏名 （ローマ字氏名） （研究者番号）	所属研究機関・部局・職 （機関番号）	備考