学際大規模情報基盤共同利用・共同研究拠点 公募型共同研究 平成31年度採択課題
11th Symposium
Joint Usage / Research Center for Interdisciplinary Large-scale Information Infrastructures
jh190064-NAH

下川辺 隆史（東京大学）

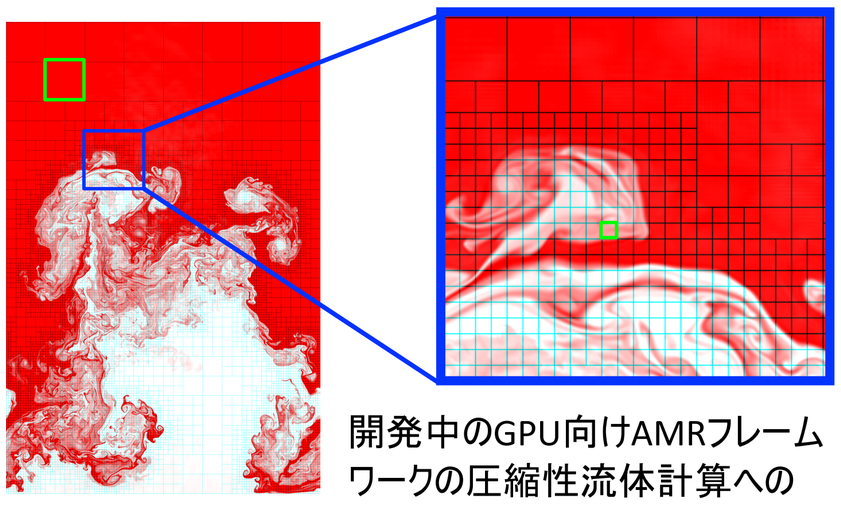
AMR法を適用したLBM計算の大規模化に向けたフレームワークの拡張

1 研究背景と研究目的

近年、ステンシル計算を用いた格子に基づいたシミュレーションでは、大規模な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きている。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。

本研究では、開発中のAMRフレームワークを流体中を流れ成長する金属凝固成長計算への適用を進めている。金属凝固成長計算は、格子ボルツマン法(Lattice Boltzmann Method; LBM)による流体計算とフェーズフィールド法による凝固成長計算で構成される。

本年度は、AMR法を適用したLBM計算の大規模・高性能計算に焦点をあてながら、これまで以上に大規模な計算に向け、AMR法フレームワークの高度化を行う。解像度により異なる時間ステップ幅をもつLBM計算のための通信をまとめて行う手法と、通信を効率的に行うことができる領域分割手法を構築する。これまで以上に大規模な計算の実現に向けて、一時メモリの効率的な利用を行う。最終的には、大規模LBM計算、さらに金属凝固成長計算の大規模計算を実現することを目指す。



開発中のGPU向けAMRフレームワークの圧縮性流体計算への適用例。緑のブロックは同一数の格子点を持つ。

2 AMR法フレームワーク

AMR法フレームワークの概要を述べる。前年度までに、複数GPUに対応したAMRフレームワークを構築し、100台を超える複数GPU計算を達成した。

2.1 フレームワークの対象

- ✓ 直交格子をベースとしたブロック型のAMR
- ✓ 各格子点上で定義される物理変数（配列）の時間変化を計算
- ✓ 物理変数の時間ステップ更新は陽的でステンシル計算で行う

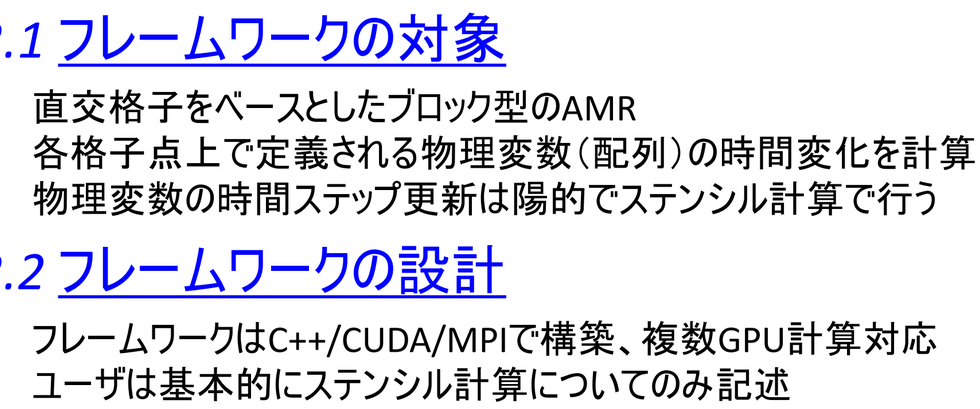
2.2 フレームワークの設計

- ✓ フレームワークはC++/CUDA/MPIで構築、複数GPU計算対応
- ✓ ユーザは基本的にステンシル計算についてのみ記述
- ✓ AMRでは様々な解像度のブロックが存在するが、ユーザはあたかも単一解像度への計算として記述できる
  → これを実現するため各ブロックは袖領域の格子を持つ
- ✓ AMRデータは木構造で管理するが、これを意識しないプログラミング
- ✓ 任意の数の物理変数（配列）を扱える

2.3 AMR法のデータ構造

- ✓ 構造格子を再帰的に細分化し、木構造で表す
- ✓ 物理空間ではリーフノードに格子ブロックを配置
- ✓ メモリ空間では複数の格子ブロックを単一配列内に配置
- ✓ リーフノードとメモリ上の格子ブロックは整数値(Id)で対応付け

物理空間



メモリ空間

2.4 ステンシル関数の記述と実行

- ✓ メモリレイアウトをフラットな構造とすることで、c++11ラムダ式で定義された直交格子用のステンシル計算関数を、一度に全格子ブロックに適用できる

```
// User-written stencil function
auto diffusion3d = [=] __host__ __device__
(const MArrayIndex &idx, int level,
 float ce, float cw, float cn, float cs, float ct,
 float cb, float cc, const FLOAT *f, float *fn) {
    fn[idx.ix()] = + 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)];
 }};
```

(i, j, k)

(i, j, k-1)

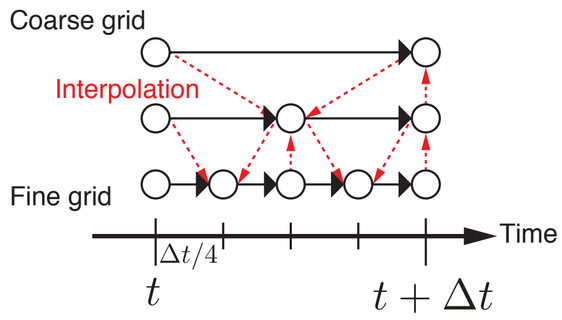
- ✓ フラットなメモリレイアウトにより複数の格子ブロックを同時に計算できる

```
Range3D inside; // 3D rectangular range where stencil functions are applied.
Engine_t engine;
engine.run(amrcon, inside, LevelGreaterEqual(1), diffusion3d,
           idx(f.range()), level(), ce,cw,cn,cs,ct,cb,cc, ptr(f), ptr(fn));
```

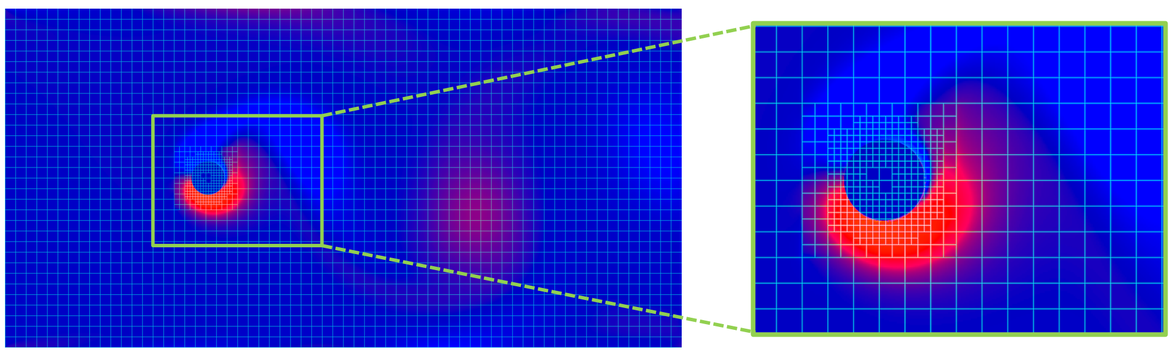
3 格子ボルツマン法（LBM）へのAMR法の適用

3.1 AMR法を適用したLBM計算で必要な補間計算

- ✓ AMR法を適用したLBMは、解像度ごとに時間ステップ幅が異なり、空間方向に加えて時間方向にも物理量の補間が必要となる。
- ✓ 本フレームワークでは、この補間計算が実装されている。



3.2 LBM計算へのAMR法の適用例



- ✓ AMRフレームワークを適用したLBM計算による円柱周りの流体計算の例
- ✓ 円柱周りの格子解像度が細かくなっている。

4 今後の研究計画

AMR法を適用したLBM計算の大規模・高性能計算に焦点をあてながら、これまで以上に大規模な計算に向け、AMR法フレームワークの高度化を行う。

大規模な計算の実現に向けて、GPU間通信の削減のため、使用するGPU数の動的最適化、領域分割方法の高度化を行い、一時メモリの効率的な利用を行う。

- ✓ 現状のフレームワークは、各GPUのメモリ使用量を均等として負荷を均一に分散することを最優先する。領域分割により各GPUの担う計算領域の幾何学的形状が複雑となり、通信量が増加し、大規模計算では性能低下の大きな要因となる。
- ✓ 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、これを改善する。
- ✓ 計算の随所でGPU間の通信量を最適化するため、計算領域全体の解像度に見合ったGPU数を動的に割り当てる手法を開発する。

解像度により異なる時間ステップ幅をもつLBM計算のための通信をまとめて行う手法と領域分割手法を構築する。

- ✓ 同一の時間ステップ幅を持つ計算手法に対し、通信をまとめて行うカウントダウン式の時間ブロッキング手法が有効であり、これをLBM計算にも適用できるようにする。
- ✓ 木構造を単位とした領域分割方法では、分割面で解像度に基づいた通信量の評価を取り入れ、なるべく通信量が少なくなるように領域を分割する。

最終的には、大規模LBM計算向けの高度化技術を導入したフレームワークを金属凝固成長計算の流体部分へ適用し、より大規模な計算を実現することを目指す。