TSUBAME 共同利用 平成 27 年度 学術利用 成果報告書

利用課題名 LRnLA アルゴリズムを用いた FDTD 法による電磁場伝搬解析
英文：FDTD simulation of electromagnetic wave propagation with LRnLA algorithms

善甫 康成
Yasunari Zempo

法政大学 情報科学部
Computer and Information Sciences, Hosei University
http://cis.k.hosei.ac.jp/

光学デバイスや電磁場解析は計算規模が非常に大きいが、これに適した FDTD 法(Finite Difference Time Domain Method)を用いたシミュレーションコードを作成した。我々の計算手法は LRnLA (Locally recursive non-Locally Asynchronous)アルゴリズムを基盤としている。その中でも GPGPU に最適な DiamondTorre 法に基づいて、ハードウェアの特徴に合わせチューニングを行った。DiamondTorre 法は他の計算手法や異なったアーキテクチャの計算機でも利用できるという特徴があるが、この報告では Yee スタッカード格子の計算を many-GPU クラスターで、その性能を検証することとした。実際サイズの光学のシミュレーションが可能になるよう、このアルゴリズムに基づくコードを調整した。事前のモデル計算により、アルゴリズムのパラメーター及びコンピューターモデルを用いて、コードの性能を推定しておき、これと実際の性能を単一の GPU 及び GPU クラスターで検証した。テストの結果はモデル上の理論値と一致することが判明した。Yee セル0.3×10^{12}を用いる 3 次元のシミュレーションにおいて、1 秒あたり0.65×10^{12}個のセルの更新を行うという、高い性能を達成することができた。

We have implemented FDTD (Finite Difference Time Domain Method) method for solution of optical and other electrodynamic problems of high computational cost. The implementation is based on LRnLA (Locally recursive non-Locally Asynchronous) algorithm DiamondTorre, which is developed specifically for GPGPU hardware. DiamondTorre can be used with other methods and computers. But now, just in this project, it was adapted to Yee grid and many-GPU cluster. The algorithm is implemented in software for real optics calculation. The software performance is estimated through algorithms parameters and computer model. The real performance is tested on one GPU device, as well as on many-GPU cluster. The result matches model estimations. The high performance of up to 0.65×10^{12} cell updates per second for 3D domain with 0.3×10^{12} Yee cells total is achieved.

LRnLA algorithm, GPU, FDTD, high performance

背景と目的

電磁場伝搬問題の数値解法において FDTD 方法はもっとも簡単で直接的な方法の一つである。ただし，非常に多くの格子点が必要となるので、大きな計算コストがかかる。通常スーパーコンピューターを使っても、計算できるサイズには限界がある。我々は独自のアルゴリズムを使い、この数値計算の限界を超え、10～100 倍程度の大きい領域でも計算が可能とした。今回、many-GPU クラスターに最適な DiamondTorre という LRnLA アルゴリズムを用いた FDTD コードを用いた計算を TSUBAME2.5 で実施した。既存の計算と比べ、極めて巨大なデータを使う計算でも最大のパフォーマンスを得ることができた。

我々が用いた FDTD 方は次のような特徴を持っている。まず 3 次元の Maxwell 方程式を直接差分法で解くものである。この差分では空間軸に沿って４次近似であり時間について２点を用いる３次元の Maxwell 方程式を解く FDTD 法である。境界は PML を用い、また TFSF 電磁場源を有している。また分散性媒質については、単純な Drude モデルを用いている。

概要

LRnLA アルゴリズムの特徴は、空間分割だけではなく、時間・空間分割によって並列化を行うところにある。図 1 は我々の時間発展プログラムでの「時空間分割」

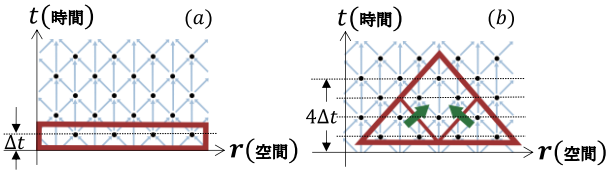

図 1. 通常の空間分割（左）と我々が開発してきた時空間分割（右）．基本的な形状．三角領域の中では独立して計算ができる．この計算の局所性を利用する．

の模式図である。通常の空間分割 (a) では 1 ステップ毎に同期を取り、次のステップの計算を行う。時空間分割 (b) では、局所性があれば部分的に時間発展をさせ、これを繰り返す。これによりある長さのステップ毎に同期をとるが、基本的に時空間を合わせて分割して計算を進める。我々はこの種の一連の手法を LRnLA (Locally Recursive non-Locally Asynchronous) アルゴリズムと呼んでいる。詳細は参考文献[1, 2, 3]参照。

DiamondTorre 法の概要

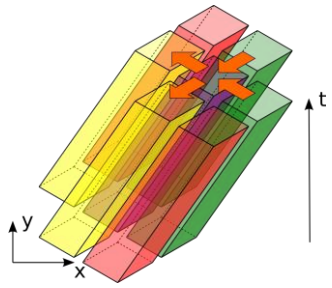

図 2. 時空間の分割（DiamondTorre）

図 1 に示したケースを基に、時空間を合わせた 4 次元領域に対して $X-Y-t$ 時空間分割を行うと、図 2~4 に示すようなプリズム形状に分割することができる。なおここでは Z 軸では分割しない。もちろん 4 次元領域では各時間ステップに対応した計算が行われる。図 2 は、このプリズム(DiamondTorre と呼ぶ)内での計算毎の依存関係を示している。また図3 に示すように

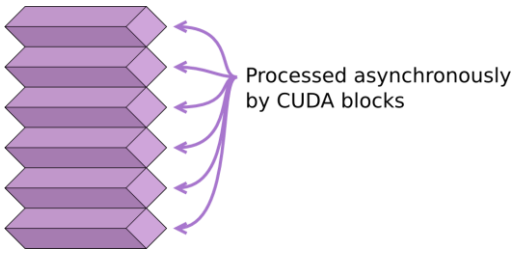

図 3. CUDA-block による非同期並列処理.

DiamondTorre 内ではデータ通信なしで並列処理が可能である（図 2）。これを実現するため非同期 CUDA-block で処理する。Z 軸（図には示していない）に沿っての格子点は CUDA-block 内の CUDA-thread で処理する。ノード内の並列化はこの DiamondTorre の並びを 3 つの GPU へ分散して実行する。ノード間の並列化は、図 4 に示したように、$X-t$ 時空間の分割によって行う。

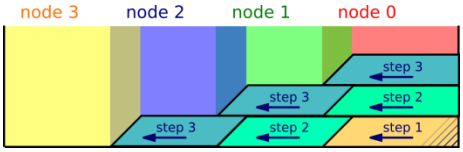

図 4. ノード間の並列化

結果および考察

TSUBAME2.5 での並列スケーリング

この節では、我々のコードを TSUBAME2.5 で検証した結果について説明する。今回、利用することができたノード数は 300 である。その性能は次の通りである。各ノードは、「NVIDIA Tesla K20X」GPU を 3 台搭載している。累計 GDDR5 メモリは 16.9 GB であり、各転送速度は 208 GBps である。各ノードは，54 GB のメモリを有し、一部 96 GB のメモリを有している。ただ計算で実際利用できるメモリは 40 GB 程度である。GPU 間で PCI-E2.0 インターフェイスが搭載され、転送速度は両方で 4 GBps である。各ノードは，120 GB の SSD メモリを有している。ノードは QDR InfiniBand ネットワークに接続しており，合計で最大 80 Gbps の通信性能を有している。

ウィーク・スケーリング

ウィーク・スケーリングではプロセッサあたりの計算領域の大きさは変えず、ノード数に比例して領域が増加する。検証したのは次のスケーリングである。

(1) Y 軸スケーリング。データ通信は計算時間に隠蔽される。各 GPU では 112 CUDA-block を利用する。

(2) Y 軸スケーリング。データ通信はパーフォマンに直接影響する。各 GPU では 42 CUDA-block を利用する。

(3) X 軸スケーリング。各 GPU では 1890 Yee セルのデ

ータを保持する。ノード内の GPU の全てを利用し、Y 軸の並列化を行う。

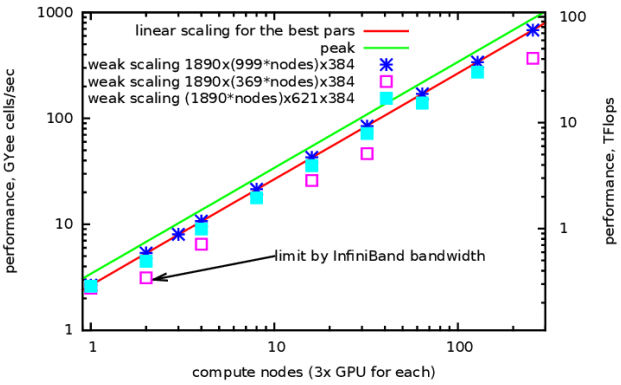

図 5. ウィーク・スケーリングの結果. ピーク性能(緑)と、リニアスケーリングの最高値（赤）に対する計算実測値

これらの計算結果を、図 5 に示す。ケース(1)では並列効率は予想通りで 99 %以上である。最大のパフォーマンスは、Yee セル0.3×10^{12}を用いる 3 次元のシミュレーションにおいて、1 秒あたり0.65×10^{12}個のセルの更新であった。この計算では、データ量が約 10 TB あり、256 ノードを利用した。ケース(2)では、1 ノードから 2 ノードまではパフォーマンスが減少してしまう。理由は、Infiniband のデータスループットが GPU 間のスループットより低いことによるものである。それ以外はノード数の増加に伴い、パフォーマンスの目立った減少はない。ケース(3)では、並列効率は理想値に近いがケース(1)のより低い。理由は、ノード利用の微妙な不均衡によるものと考えられる。また上記同様、1 から 2 ノードにパフォーマンスの減少がある。

ストロング・スケーリング

ストロング・スケーリングでは、全体の問題の大きさは変えず、プロセッサ数の増加に伴い、領域の分割数を増やし並列処理を行う。検証したのは次のスケーリングである。（図 6(a) 参照）

(1) $720 \times 3312 \times 384$ サイズの計算を 1～32 ノードで実行

(2) (1)の 4 倍サイズ $(720 \times 13248 \times 384)$の計算を 4 ~64 ノードで実行

(3) (1)の 16 倍サイズ $(720 \times 52992 \times 384)$ 計算を 16~256ノードで実行

上記のテストはノードあたり計算可能なサイズにより決定した。上限は各ノードのメモリ（約 40 GB 利用可能）である。下限は LRnLA アルゴリズムで決まる。この下限以下のテストも可能だが、パフォーマンスの減少が当然予想されるため、ケース(2)(3)では 1 ノードまで下げることは行わなかった。

総てのケースで、並列プロセスの増加とともに、ノードあたりのパフォーマンスが減少する。理由は GPU あたりの CUDA-block 数の減少によるものである。そこ

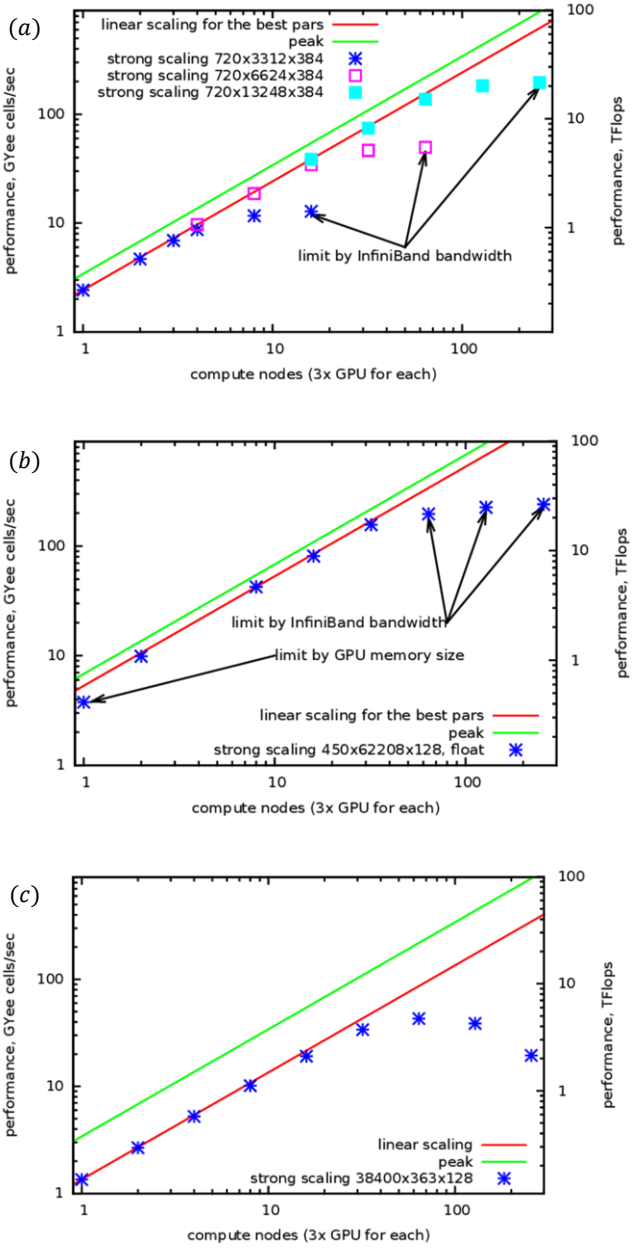

図 6. ストロング・スケーリングの結果. ピーク性能(緑)と、リニアスケーリングの最高値（赤）に対する計算実測値. (a) 計算サイズの違いによる結果. (b) Y 軸スケーリングの結果. (c) X 軸スケーリングの結果.

で問題サイズを調整して、1~256 ノードのスケーリングを実施可能とし、Y 軸と X 軸の並列処理を行った。

Y 軸スケーリングテストの結果を図 $6(b)$ に示した。計算サイズは$450 \times 62208 \times 128$ セル。なお 1 ノードから 8 ノードまで並列効率は 100 % 以上となっている。これはノードあたりサイズの増加に伴い、最適なアルゴリズムパラメーターを設定できるためである。

X 軸スケーリングの検証は $38400 \times 363 \times 128$ セルにて行った（図 $6(c)$ 参照）。ノードあたりのパフォーマンスはピーク性能の 3 割程度である。理由は Z 軸のセル数（アルゴリズムのパラメーターの 1 つ）が最適ではないためである。同時に起こるトランザクションが低いため、このパラメーターは GDDR5 アクセスレイテンシを隠蔽するには十分ではない。その場合でも並列高速化は約 40 倍である。128 ノード以上のときだけ、データ通信が計算よりも長い時間を要するため、計算レートが下がっている。また 1 ノードのメモリのサイズ（GPU メモリの約 3 倍程度）が高速化の上限を決定している。もちろんメモリの大きいノードを使えば、この上限を超えることができると予想される。

まとめ、今後の課題

実際のデバイスサイズでの光学過程のシミュレーションが可能な FDTD コードを開発した。コードの特徴は DiamondTorre LRnLA アルゴリズムを利用していることである。GPU デバイスのあたりのパフォーマンスを最適化し、many-GPU クラスターで 99% 以上の並列効率を示している。このアルゴリズムでは GPU メモリの大きさが問題を解くための制限にはならないうえ、パフォーマンスを落とさず CPU-RAM 及び SSD メモリまでも利用できる。

このコードを TSUBAME2.5 で実行して、高い性能を示すことが分かった。Yee セル 0.3×10^{12}を用いる 3 次元のシミュレーションにおいて、1 秒あたり 0.65×10^{12}個のセルの更新を行うという、高い性能を示すことができた。GPU 数が ~1000 の計算では、ストロング・スケーリングで約 100 倍高速化、ウィーク・スケーリングで約 1000 倍高速化を実現することができた。

Yee セル 0.3×10^{12} を用いる 3 次元のシミュレーションにおいて、1 秒あたり 0.65×10^{12}個のセルの更新を行うことができる性能を示すことができた。このサイズの計算が可能ということは、たとえば $1\ \mathrm{mm}^3$ 領域で光学の電磁場伝搬のシミュレーションできることになる。これは実デバイスサイズの計算が可能ということであり、数値光学の躍進であると考えられる。もちろんその応用範囲は広い。

また、ノードあたりのメモリ量がもっと充実しているか、あるいはもっと Fat ノードを利用できれば、パフォーマンスの減少なしで、それに比例して更に大きな領域を用いる計算が可能であると予想できる。

参考文献

[1] A. Perepelkina, V. Levchenko, ″DiamondTorre Algorithm for High-Performance Wave Modeling″, Keldysh Institute preprints, 2015, #18

[2] A. Perepelkina, V. Levchenko, “DiamondTile Algorithm for High-Performance Wave Modeling,”, GPU Technology conference GTC2015, S5315

[3] A. Zakirov, V. Levchenko, A. Perepelkina, Y. Zempo, “High performance FDTD algorithm for GPGPU supercomputers”, submitted to J. Phys.: Conf. Ser.