jh150031-NA19

Fast Multipole Method を用いた多種アーキテクチャ向けスーパーコンピュータ用ライブラリの開発と分子・流体シミュレーションでの評価

横田　理央（東京工業大学）

本研究では、エクサスケールを視野に入れた FMM のオープンソースライブラリである ExaFMM を様々なアーキテクチャのスーパーコンピュータ向けに最適化し、流体解析における疎行列の前処理と古典分子動力学における長距離力の計算に用いることで、その性能や精度を検証した。これまで GPU や XeonPhi 等にはある程度最適化出来ており SX-ACE や FX100 などの固有のアーキテクチャにも対応した最適化を行うことで、Top500 のリストにあるほぼ全てのプラットフォームに対応できる FMM ライブラリを構築した。また、ExaFMM に automake を導入することで全てのマシンでファイルやコマンドを改変することなくコンパイルできるようにし、buildbot を用いて全てのマシンで自動的にコードの動作確認をできる機能を付加した。

1.　共同研究に関する情報

(1) 共同研究を実施した拠点名

東北大学 サイバーサイエンスセンター
東京工業大学 学術国際情報センター
名古屋大学 情報基盤センター
京都大学 学術情報メディアセンター

(2) 共同研究分野

- ☑ 超大規模数値計算系応用分野
- □ 超大規模データ処理系応用分野
- □ 超大容量ネットワーク技術分野
- □ 超大規模情報システム関連研究分野

(3) 参加研究者の役割分担

成見哲（電気通信大学）GPU 向け動的負荷分散、耐故障性機能の開発、GPU カーネルの高速化
青木尊之（東京工業大学）流体解析コードと FMM 前処理の統合
泰岡顕治(慶應義塾大学)分子動力学コードと FMM の統合
高岩大輔(慶應義塾大学)FMM の分子動力学コードにおける計算精度の検証
sEdgar Noriega（電気通信大学）GPU 向け動的負荷分散、耐故障性機能の開発

2.　研究の目的と意義

N 体問題の高速化手法である Fast Multipole Method (FMM) は並列性、演算密度、データの局在性、非同期性などの観点からエクサスケールコンピュータにおいて高い性能を発揮できる手法として期待されている。汎用性の観点からも、FMM は N 体問題だけでなく音響・電磁場解析における密行列問題の行列・ベクトル積、流体・構造解析における疎行列問題の前処理、量子化学計算における固有値解析、信号処理やスペクトル解析における FFT などの代わりに用いることができることから、超大規模計算における代替手法として注目されている。

本研究では、エクサスケールを視野に入れた FMM のオープンソースライブラリ ExaFMM を様々なアーキテクチャのスーパーコンピュータ向けに最適化し、流体解析における疎行列の前処理と古典分子動力学における長距離力の計算に用いることで、その性能や精度を検証する。これまで GPU や Xeon Phi 等にはある程度最適化出来ているので SX-ACE や FX100 など固有のアーキテクチャにも対応した最適化を行うことで、Top500 のリストにあるほぼ全てのプラットフォームに対応できる FMM ライブラリを構築する。GPU については新たに動的負荷分散・対故障性機能を ExaFMM に付加する。また、100 億メッシュ規

模の流体シミュレーションにおける FMM とマルチグリッド法の比較と 1 兆ステップ規模の分子シミュレーションにおける FMM と PME 法の比較を通してその計算速度、並列化効率、計算精度を検証する。

本研究計画の将来性について考える際にカギとなるのが、計算機のアーキテクチャの変遷により疎行列ソルバや FFT などの既存の高速アルゴリズムの性能が出にくくなっているという事実である。このトレンドは一過性のものではく、 FMM の優位性は今後10年､20年と経つにつれますます大きくなると予想される。最適なアルゴリズムが変化すると分かっていても大規模な科学技術ソフトウェアを書き換えるのは容易でなく、実際にはどのタイミ ングで変更を行うかを皆が見計らっているのが現状である。本研究の大きな目標は、このアルゴリズムの変更を自ら働きかけることで促進し、オープンソースライブラリを提供することで次世代計算機を多くの科学技術アプリケーションコードが効率的に活用できることを可能にすることである。

流体・構造解析の既存研究では、幅広い用途をもつ疎行列ソルバを出発点にとり、それをいかに次世代計算機で高速に計算するかを考えるのが主流であったが、本研究は逆に次世代計算機で高い計算性能は出ることは分かっているが用途の限られていた FMM を出発点にとり、その用途をどこまで広げられるかという方向性をとっているのが特徴である。分子シミュレーションにおいても、既存研究では高い精度が保証されている PME 法を出発点にとり、それをいかに次世代計算機で並列化するかを考えていたのに対して、本研究は逆に次世代計算機で高い並列化効率がでると分かっている FMM を出発点にとり、その精度をどこまで向上できるかを検討する。

既に GPU,BG/Q,Xeon Phi などの主要なプラットフォームに実装されている ExaFMM コードを東北大学の SX-ACE や名古屋大学の FX100 を利用することで日本固有のアーキテクチャにも拡張し、Top500 のリストにあるほぼ全ての情報基盤で使えるライブラリとして提供することができた。また、単に実装するだけでなく、各プラットフォー ムに固有の最適化を行うことで,性能の移植性も確保することができた。

3. 当拠点公募型共同研究として実施した意義

FMM のライブラリを開発する上で重要なのは、ソフトウェアスタックの上下の階層との連携である。本研究では、計算機科学分野の成見、Martinez が GPU 向けの動的負荷分散・対故障性機能・最適化を担当し、計算科学分 野の青木(流体)、泰岡(分子)、高岩(分子)がアプリケーションとの統合と精度の検証を担当し、その 2 つの分野をまたぐ形で横 田 (FMM)がライブラリの開発・最適化を進めた。これによりアプリケーション側からの現実的な要求 をもとにライブラリのインターフェイスを拡張することができたため、不必要な抽象化や時期尚早な最適化を行わずに済んだ。また、計算機科学分野の共同研究者によって行われたソフトウェアスタックの下層部での機能の追加やチューニングはそのまま自動的にライブラリとアプリケーションに反映される。特に FMM のような複雑なアルゴリズムの高速なライブラリを開発する場合、少なくともアプリケーション、アルゴリズム、データ構造、内部カーネルに関して各々が深い知識を持った研究者による共同研究が不可欠であった。

4. 前年度までに得られた研究成果の概要

継続課題ではないため、該当せず。

5. 今年度の研究成果の詳細

5.1 ExaFMM の動的負荷分散

非一様な計算点の分布を FMM で計算する際には負荷分散が重要な課題となる。一般的な方法では空間充填曲線を用いて粒子の領域分割を行い、そこから生じる大域的な木構造のうち必要な部分木のみを通信する。本研究では、領域分割の際に空間充填曲線を等分割するのではなく、直前の時間ステップの計算負荷で重みづけされた空間充填曲

線の分割を行った。これにより、前のステップで負荷の多かった MPI プロセスには次回のステップで負荷が軽減され、負荷の少なかった MPI プロセスは次のステップで負荷が増加するしくみができる。

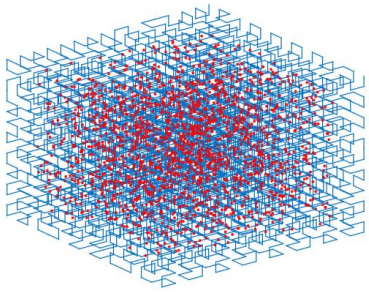

図 1　粒子の分割するための空間充填曲線の様子

負荷分散に用いる 3 次元空間充填曲線の様子を図 1 に示す。非一様に分布した粒子の周りを Hilbert 曲線が通過している様子が見てとれる。

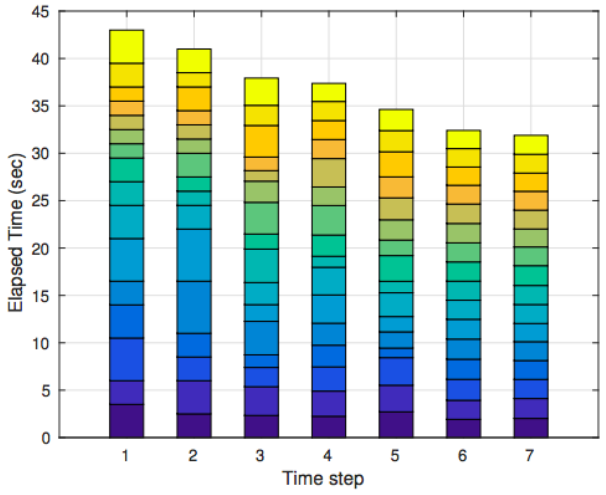

図 2　時間経過と負荷分散の関係

各時間ステップにおける各スレッドの計算時間を図 2 に示す。計算時間は各スレッドごとに区切られ、色分けされている。最初のステップでは各スレッドごとに計算時間にばらつきが見られるものの、時間の経過とともに負荷が均等になっていく様子がうかがえる。これは、前のステップの計算負荷をもとに領域分割をし直したためである。

図 3 には 131,072 コアを用いた大規模計算における計算時間の内訳とそれぞれの負荷分散の様子を示す。横軸はスレッド ID、縦軸はそのスレッドの計算時間を表す。凡例にある Comm LET bodies は粒子の通信、Comm LET cells はセルの通信、Grow tree は木構造の構築、Traverse は FMM の相互作用計算、Up Down pass は木構造の上下へのデータの移動にかかった時間を表す。計算時間の大部分は FMM の相互作用計算が占めている。ただし、その計算負荷はほぼ一様に分散されていることが分かる。

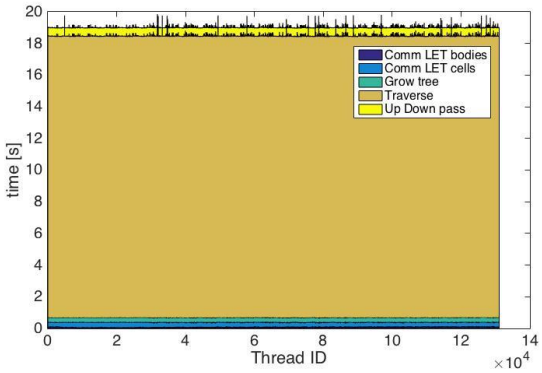

図 3　131,072 コアを用いた ExaFMM における各スレッドの計算時間

4.2 FMM とマルチグリッド法の比較

FMM は粒子の相互作用の計算のみならず疎行列の反復解法における前処理にも用いることができる。ここでは非圧縮性流体解析の圧力場算出に用いられるポアソン方程式の反復解法に用いたときの FMM の性能をマルチグリッド法と比較する。

ポアソン方程式の前処理に FMM を用いる場合、ディリクレ条件やノイマン条件などの境界条件を与えるために境界要素法を併用する必要がある。ただし、この境界要素法の計算は境界の点のみを用いた計算であるため、領域全体の計算量に比べると小さい。また、境界要素法の計算にも FMM を適用することができるため、全体で O(N)の手法であることに変わりはない。

ポアソン方程式の CG 法による解法をそれぞれ異なる前処理を行った場合の収束性の比較を図 4 に示す。横軸は CG 法の反復回数、縦軸は残差を表す。凡例の FMM は 3 種類あり上から順に近似精度がそれぞれ 10-6、10-4、10-2 のものを表す。AMG、

GMG は代数マルチグリッド、幾何マルチグリッドを表す。Inc Chol は不完全コレスキー分解を表す。

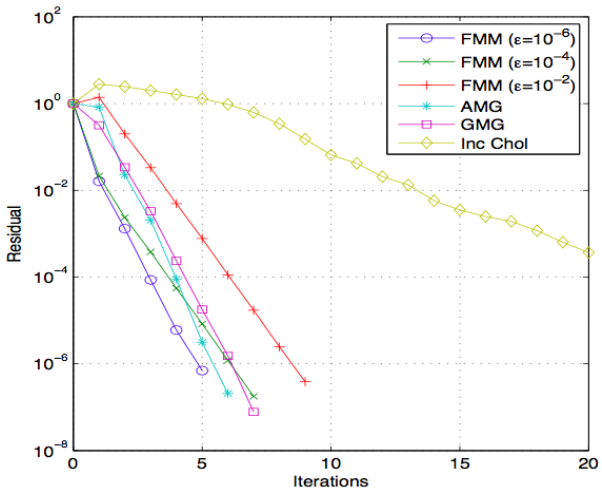

図 4　FMM とマルチグリッド法の収束性の比較

近似精度の最も高い FMM の収束性はマルチグリッド法を上回ることが図 4 より明らかである。FMM の近似精度を落としていくにしたがって収束性も悪くなる。ただし、近似精度を落とすことで反復あたりの計算時間が短くなるため、反復回数があまり増えない程度に近似精度を落とすほうが全体の計算としては速くなる。

図 4 は単一コアの計算結果であり、FMM が真価を発揮できる条件とは言いがたい。また、収束性だけ見てもその絶対的な計算時間は分からない。疎行列の前処理法としての FMM の有用性を評価するためには並列計算における正味の計算時間の比較を行う必要がある。

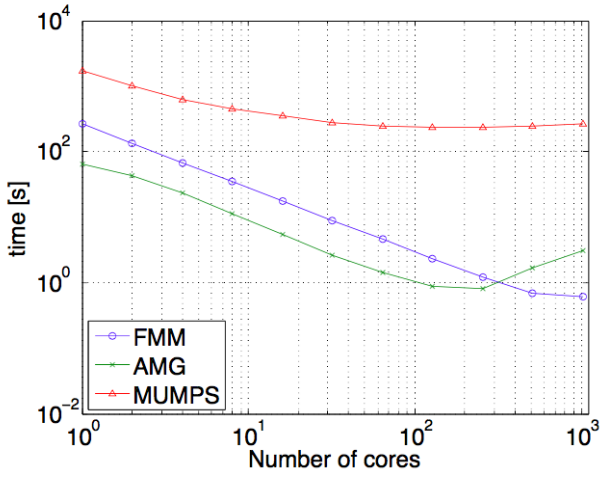

図 5　FMM、マルチグリッド法、直接解法のコア数を増加させたときの計算時間

図 5 にはコア数を増やしていったときの FMM、マルチグリッド法、マルチフロンタル法のソルバ全体にかかる計算時間の比較を示す。FMM は本研究で開発した exaFMM コード、AMG はマルチグリッド法の並列実装である BoomerAMG コード、MUMPS はマルチフロンタル法の並列実装である MUMPS コードの結果である。BoomerAMG と MUMPS はそれぞれの業界内で最大のシェアを誇るコードで、いずれも高度なチューニングが行われている高品質な実装であるといえる。FMM と AMG はともに MUMPS よりも 1 コアにおける計算時間が短く、コア数が増加するにしたがってその差はさらに大きくなり、最終的には 10 倍以上の計算時間の差になる。FMM は AMG よりの 1 コアにおける計算時間は長いが、コア数が増えるにしたがって優位になるのが見てとれる。今後メニーコアが標準的になるにしたがって FMM はマルチグリッド法に代わる O(N)の前処理の手法として用いられる可能性が高い。

4.3 FMM を用いた分子シミュレーション

古典分子動力学における長距離力の計算には通常 Particle mesh Ewald 法(PME)が用いられる。しかし、PME の内部では多数の FFT の計算が行われるため FFT の並列化効率の低さが最終的にはボトルネックとなる。FMM はもともとこのような粒子同士の長距離力の計算のために開発されたため、分子動力学計算には周期境界条件を付加するだけで対応できる。また、図 5 に示すように FMM の並列化効率は他の手法に比べて高く、FFT の代替手法として用いることで次世代計算機で高い性能を得ることができると期待される。

本研究では CHARMM や GROMACS などの分子シミュレーションソフトウェアに FMM を組み込み、PME との直接比較を行った。その結果、FMM の結果からは PME では見られないエネルギーの増加があることが分かった。図 6 に FMM の展開次数 P とオープニングアングル θ を変えたときのエネルギーの時間変化を示す。ただし、これらの P と θ の値は

同じ近似精度になるように設定している。

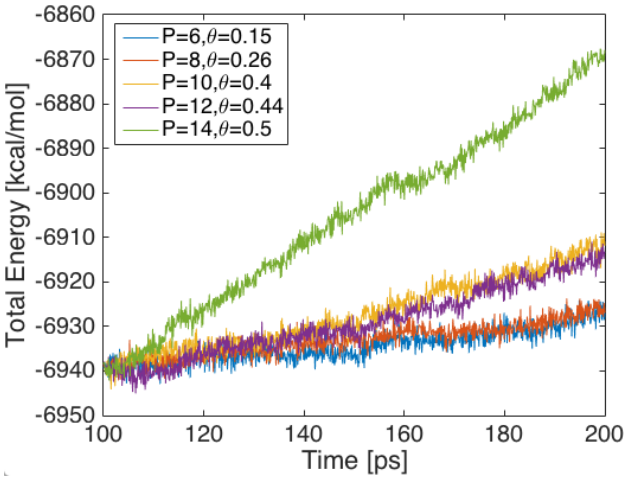

図 6　FMM の計算におけるエネルギーのドリフト

同じ近似精度でもエネルギーの増加率が異なることから、単に FMM の誤差が熱に変わっているわけではないことが分かる。θ の値が小さいときのほうがエネルギーの増加が少ないことが図 6 より見てとれる。θ の値が小さいと直接計算される近傍領域が拡大され、近似計算される遠方領域は遠のく。これがエネルギーの増加を抑えることから、原因は直接計算される近傍場と近似計算される遠方場の境目に生じる不連続面を振動する原子が何度も行き来することであると予想される。

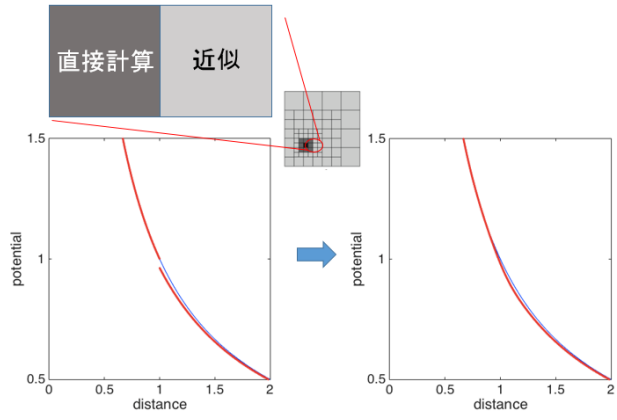

図 7　FMM の不連続面を接続する手法

本研究では前述の FMM の近傍領域と遠方領域の不連続面の問題を解決するため、これらを滑らかにつなぐ関数をその境界に適用する方法を提案する。この手法はまだ実験段階にあり、分子シミュレーションに応用するにはいたっていない。

4.4 多種アーキテクチャ用のフレームワーク

本研究で開発中の ExaFMM は既に x86 のみならず、GPU、BG/Q、FX10、Xeon Phi などへも実装されている。これは Top500 の リストにある情報基盤の全てにおいて動作するコードを目指しているためである。今後もこのようなポータビリティ を維持するために、常に最新のアーキテクチャへの実装を行なっていく必要がある。東北大学サイバーサイエンスセンターでは SX-ACE 用のベクトル演算のチューニングを、東京工業大学学術国 際情報センターでは GPU 用のチューニングを、名屋大学情報基盤センターでは FX10 と FX100 用のチューニングを、京都大学サイバーメディアセンターでは Xeon Phi 用のチューニングを行った。

ExaFMM は C++の演算子オーバーロードを用いて内部カーネルの全ての演算子をコンパイル時 に SSE, AVX, BG/Q, FX10 のベクトル・イントリンシック(もしくはインライン・アセンブリコード)に変換するヘッダファイルを用いている。これにより、内部カーネルはマシン環境によらず常に強制的にベクトル化、最適化される。また、automake を導入することで configure、make によって X86、SX-ACE、FX100、GPU、Xeon Phi の全てのマシンで最適なコンパイラとオプションを自動的に選択してビルドを行うことができた。さらに、buildbot を用いたビルドテスト環境も構築し、開発者が github にコードをプッシュするたびに、自動的に X86、SX-ACE、FX100、GPU、Xeon Phi の全てのマシンでテストが実行され正常にコンパイル・実行が終了したかどうかがウェブサイトに表示されるようになった。

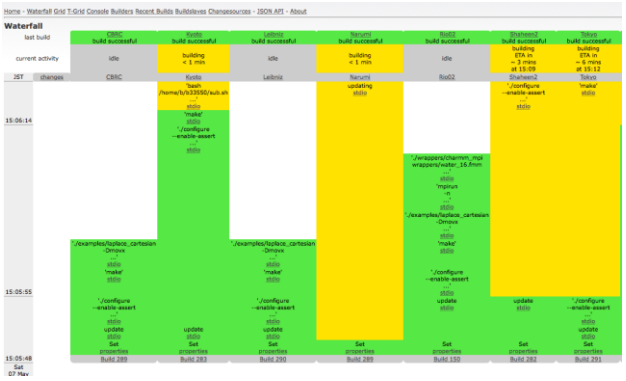

図８　Buildbot の各拠点におけるビルド状況

図８に Buildbot のウェブサイトに各拠点のビルド状況が表示されている様子を示す。Buildbot は各拠点に github から最新の exaFMM を複製し、configure, make, 実行を buildbot の master に指示された通りのフラッグを用いて行う。このとき、configure, make, 実行のいずれかの段階でエラーが生じた場合は図８の画面に赤色のブロックが表示される。テスト中のブロックは黄色、正常に動作が完了したブロックは緑色で表される。

一つのコードを複数の開発者がそれぞれ複数の拠点で動作するように開発を行う場合、github のようなバージョン管理ツールと buildbot のような継続的インテグレーションツールは極めて有用である。いずれの開発者が行った変更も即座に github に反映され、その変更によって動作に異常が発生したかどうかを自動的に全ての拠点で configure の段階から buildbot でテストできるため、マシン環境依存性、コンパイラ依存性、コードのバージョン依存性などを心配する必要がこれでなくなった。また、github のリモートレポジトリに push する前に開発者は commit されている手元のレポジトリに対して buildbot のテストを行うことができるため、自分の行った変更がコードの他の部分を壊さないかをパラメータを網羅的に変えながら全ての開発拠点で予め動作確認を行うこともできるようにした。

ExaFMM の autoconf の設定は FFTW と mpich をベースにしており、これらのライブラリがサポートするマシン環境では動作が保証されている。ただし、SX-ACE や FX100 では長いオプションを configure に渡す必要があるため、ExaFMM の autoconf では ax_compiler_vendor.m4 を書き換えることで、単に「./configure」とするだけでコンパイラやパスを指定せずとも、SX-ACE や FX100 で configure できるようにした。これは buildbot から configure を呼ぶ際に x86 のマシンと統一的なコマンドでテストできるようになったという意味で有用である。これにより SX-ACE や FX100 などのユーザーがライブラリのインストールを x86 のマシンと全く同様の手順で行えるようになった。

ExaFMM の多種アーキテクチャ用演算子オーバーロードは Agner Fogg の vector class をベースにしており、SSE、AVX、AVX512 の大半のイントリンシック命令を対応する演算子でオーバーロードしている。これを GPU や FX100 のイントリンシックへと拡張したものが ExaFMM の vec.h というヘッダファイルに定義されており、vec<4,float>のようにベクトル長とスカラー型を定義することで自動的にそのベクトル型に対して行う演算をイントリンシック命令に変換することができる。これにより、高度に最適化された内部カーネルですら同一のコードを SSE、AVX、AVX512、GPU、FX100 で用いることができるため、コードの開発速度を飛躍的に向上させることができた。

4.5 FMM を前処理に用いた流体解析

FMM は非圧縮性流体解析における Poisson 方程式の反復解法において前処理として用いることができる。通常の前処理にはマルチグリッド法を用いるのが一般的であるが、FMM のマルチグリッド法に対する優位性は 4.2 節に示した通りである。代数マルチグリッド法の中でも最高性能を誇る BoomerAMG との同等条件における直接比較において、並列数を増やしていくと FMM が優位になるという結果は流体解析において大きな意味を持つ。

FMM を実際の流体アプリケーションコードと統合する際には PETSc を用いた。PETSc は流体解析で多用されている分散メモリ並列用の大規模連立一次方程式の解法を多数収めたライブラリで、Argonne 国立研究所で開発されている。PETSc ではユーザーが前処理の関数を定義することができる

ため、このインターフェイスを使って FMM を様々な反復法の前処理に使えるようにした。また、PETSc にはマルチグリッド法 BoomerAMG も統合されているため、実行時のオプション切り替えで AMG と FMM を切り替えながら Poisson ソルバとしての直接比較を行うことができた。

図 9 にこの時の setup と apply にかかる時間を FMM と AMG についてそれぞれ図示した。Setup にかかる時間は AMG、FMM ともほぼ同等であるが、実際

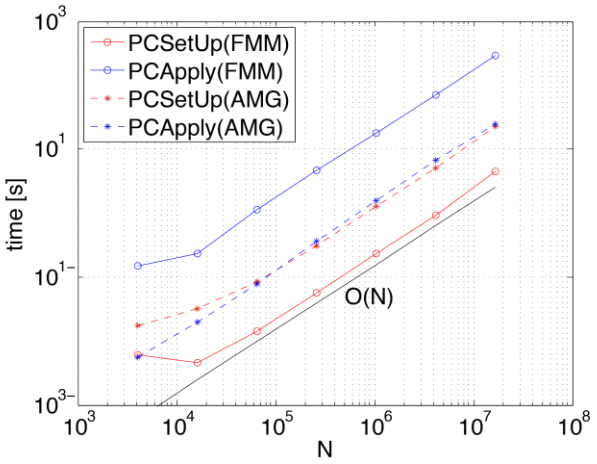

図 9　FMM と AMG の問題サイズと計算時間の関係

の apply にかかる時間は AMG の方が一桁程短いことが分かる。ただし、これは単一ノードにおける逐次計算の結果であり、FMM が AMG に対して優位になるのは並列数が増えた場合である。この他に、図 9 から分かることは FMM と AMG もともに問題サイズに比例した計算時間 O(N)になっていることである。ここに示されている計算時間「time [s]」は反復法が収束するまでにかかった合計時間であり、これが O(N)になっているのは Poisson 方程式の反復回数が AMG、FMM ともに問題サイズに依存せず一定になっていることを表している。よって、FMM は実際の流体計算においてもマルチグリッド法と同じ O(N)の解法であることが実証された。

FMM と AMG の並列数を増やした時の計算時間を図 10 に示す。FMM のスケーラビリティの方が AMG よりも良いものの 256 コアまでの範囲では逆転は見られなかった。4.2 節に示した比較ではコア数を上げていくと逆転したが、ここではコア数を増やしても逆転が見られなかった理由として、前者が 2 次元の計算であったのに対して、後者が 3 次元の実アプリケーション計算であったことが挙げられる。2 次元の FMM は複素数を用いた 1 次元のローラン展開を用いることができるのに対して、3 次元の FMM は球面調和関数を用いた展開を行うため、同じ問題サイズ、同じ精度の計算をするのにかかる時間はほぼ 10 倍となる。この 2 次元 FMM と 3 次元 FMM の計算時間の違いを図 11 に示す。問題サイズを増やしていった時の計算時間の増え方は

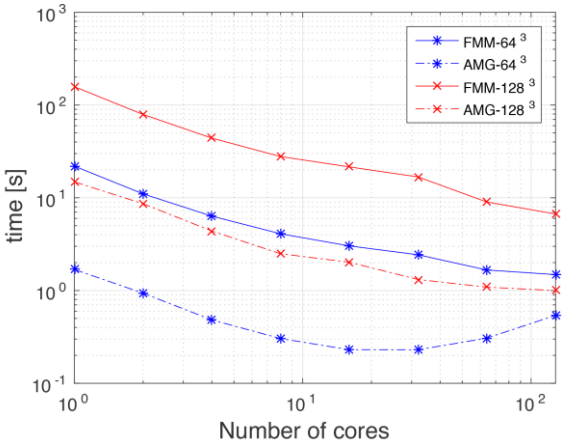

図 1 0　FMM と AMG の強スケーリング

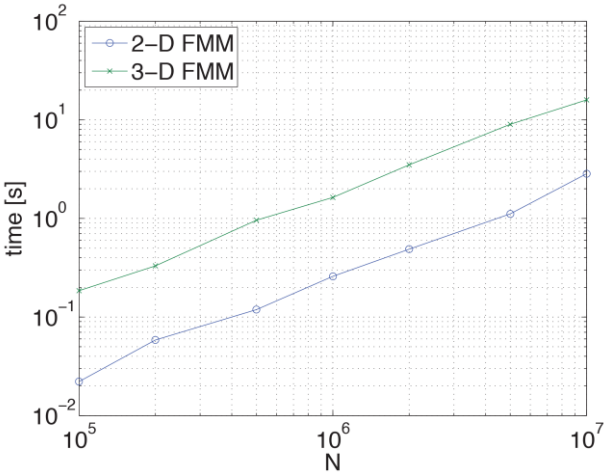

図 1 1　2 次元 FMM と 3 次元 FMM の計算時間

ともに O(N)であるが、2 次元 FMM の方が 1 桁程速い。ただし、図 10 においてもコア数が 256 からさらに増えた場合には FMM が優位になる可能性は残っており、3 次元 FMM のさらなる最適化によって逆転が起きる並列数は減少していくことが予想される。

6. 今年度の進捗状況と今後の展望

5.1 今年度の進捗状況

申請時の目標に掲げた「ExaFMM の動的負荷分散」、「FMM とマルチグリッド法の比較」、「FMM を用いた分子シミュレーション」、「多種アーキテクチャ用のフレームワーク」「超大規模な流体計算」はそれぞれ本報告書の 4.1、4.2、4.3、4.4、4.5 節に示した通り全て達成されている。

各拠点で利用した計算資源は以下の通り当初の見積もりの通りであった。

100 億格子点を用いた流体計算を SX-ACE 256 ノードで行う場合、マルチグリッド法、FMM ともに 1 ステップあたり 10 秒かかった。時間積分を 10000 ステップ行い 10×10000×256/3600=約 7000 ノード時間積を要した。

10000 原子の系を TSUBAME 1 ノードで行った場合、1 ステップあたり 1ms 程度かかった。20 通りのパラメータに関して精度検証を行った結果 $10^{-3}\times10^{9}\times20/3600$=約 6,000 ノード時間積を要した。

京都大学の Xeon Phi でも FMM を前処理に用いた流体解析を行った。TSUBAME と同様の 100 億格子点を 用いた計算を 128 ノード(最大値)で行う場合、ステップあたり 100 秒程度かかった。このため、時間積分は 10^{3} ステップ行い、$100\times10^{3}\times128/3600$=約 3500 ノード時間積を要した。

名古屋大学の FX100 においても同様の FMM を用いた流体解析を行った。100 億格子点を用いた計算を 256 ノードで行う場合、ステップあたり 10 秒程度かかった。時間積分を 10^{4} ステップ行い $10\times10^{4}\times256/3600$=約 7000 ノード時間積を要した。

5.2 今後の展望

平成 28 年度は JHPCN の国際共同研究課題「Hierarchical low-rank approximation methods on distributed memory and GPUs」が採択されており、FMM の代数学的拡張である H 行列の研究へと移行する予定である。当該研究課題は密行列の高性能実装の専門家である Jack Dongarra のグループとの共同研究であり、こちらの階層的アルゴリズムに関する知識と相手側の線形代数のノウハウをうまく組み合わせることを目指している。

7. 研究成果リスト

(1) 学術論文

1. R. Yokota, L. A. Barba, ``exaFMM: An Exascalable Fast Multipole Method Library", Communications in Computational Physics, accepted

2. J. Castrillon, R. Yokota, M. Genton, ``Multi-Level Restricted Maximum Likelihood Covariance Estimation and Kriging for Large Non-Gridded Spatial Datasets", Spatial Statistics, accepted

3. H. Ibeid, R. Yokota, D. Keyes, ``A Performance Model for the Communication in Fast Multipole Methods on HPC Platforms", International Journal of High Performance Computing Applications, accepted.

(2) 国際会議プロシーディングス

4. R. Yokota, ``Fast Multipole Method as a Matrix-free Hierarchical Low-rank Approximation", International Workshop on Eigenvalue Problems, Tsukuba, Japan, 14-16 September, 2015

(3) 国際会議発表

5. R. Yokota, ``Various implementations of FMM and their performance on future architectures", Multi-resolution Interactions Workshop, Durham, USA, 28-29 August, 2015

6. R. Yokota, ``Fast Multipole Method as a Matrix-free Hierarchical Low-rank Approximation, International Workshop on Eigenvalue Problems", Tsukuba, Japan, 14-16 September, 2015

7. R. Yokota, H. Ibeid, D. E. Keyes, ``Preconditioning Sparse Matrices Using a Highly Scalable Fast Multipole Method", 3rd International Workshops on Advances in Computational Mechanics, Tokyo, Japan, 12-14, October, 2015

8. R. Yokota, F.H.Rouet, X.S. Li, `` Comparison of FMM and HSS at Large Scale", SIAM Conference on Applied Linear Algebra, Atlanta, USA, 26-30 October, 2015

(4) 国内会議発表

9. 横田、第 7 回 自動チューニング技術の現状と応用に関するシンポジウム(2015/12/25)、招待講演