

自動チューニング機構を有する アプリケーション開発・実行環境 ppOpen-HPC

中島研吾^{†1, †7} 佐藤正樹^{†2, †7} 古村孝志^{†3, †7}
奥田洋司^{†4, †7} 岩下武史^{†5, †7} 阪口 秀^{†6, †7}

ヘテロジニアスなアーキテクチャによる計算ノードを有するポストペタスケールシステムの処理能力を十分に引き出す科学技術アプリケーションの効率的な開発、安定な実行に資する「自動チューニング機構を有するアプリケーション開発・実行環境：ppOpen-HPC」を開発する。対象離散化手法を有限要素法、差分法、有限体積法、境界要素法、個別要素法に限定し、各手法の特性に基づきハードウェアに依存しない共通インタフェースを有するアプリケーション開発用ライブラリ群、耐故障機能を含む実行環境を提供する。自動チューニング技術の導入により、様々な環境下における最適化ライブラリ、耐故障機能を持つ最適化アプリケーションの自動生成を目指す。本研究は 2014 年度に東京大学情報基盤センターに導入予定の数十ペタフロップス級システムをターゲットとし、同システム上で実アプリケーションによって検証、改良し、一般に公開する。

ppOpen-HPC: Open Source Infrastructure for Development and Execution of Large-Scale Scientific Applications with Automatic Tuning

Kengo Nakajima^{†1, †7} Masaki Satoh^{†2, †7}
Takashi Furumura^{†3, †7} Hiroshi Okuda^{†4, †7}
Takeshi Iwashita^{†5, †7} Hide Sakaguchi^{†6, †7}

We propose an open source infrastructure for development and execution of optimized and reliable simulation codes on post-peta (pp) scale parallel computers with heterogeneous computing nodes which consist of multicore CPU's and accelerators., named "ppOpen-HPC". ppOpen-HPC consists of various types of libraries, which covers various types of procedures for scientific computations. Source code developed on a PC with a single processor is linked with these libraries, and generated parallel code is optimized for post-peta scale system. Capability of automatic tuning is important and critical technology for further development on new architectures and maintenance of the framework.

1. はじめに

本稿では、平成 23 年度より開発を開発した自動チューニング機構を有するアプリケーション開発・実行環境、ppOpen-HPC [1] の概要を紹介する。本研究は、科学技術振興機構戦略的創造研究推進事業 (CREST) 研究領域「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出 (研究総括：米澤明憲 (理化学研究所) [2])」の一環として実施されている。

人類は 2008 年にペタスケールを達成し、2019 年～2020 年頃にはエクサスケールシステムが実現すると言われているが、そのためには様々な技術革新が必要である。エクサスケールシステムに向けて、数値アルゴリズム、アプリケーションにおけるキーワードは、以下の 5 点に集約され、これらを考慮した研究開発が必要である：

- ① ヘテロジニアスなアーキテクチャ (Multicore, Manycore, GPU)
- ② 混合精度演算 (Mixed Precision Computation)
- ③ 自動チューニング (Auto-Tuning, Automatic Tuning (AT))
- ④ 耐故障性 (Fault Tolerant)
- ⑤ 通信削減アルゴリズム (Communication Reducing Algorithms)

本研究ではこの 5 点を念頭に置き、ホスト CPU とアクセラレータ、コプロセッサによるヘテロジニアスなアーキテクチャによる計算ノードを有するポストペタスケールシステムを想定し、その処理能力を十分に引き出す科学技術アプリケーションの効率的な開発、安定で効率的な実行に資する「自動チューニング機構を有するオープンなアプリケーション開発・実行環境 (ppOpen-HPC, pp=post-petascale)」の研究開発を行う。

離散化手法としては、大規模シミュレーションに適した有限要素法 (Finite Element

^{†1} 東京大学情報基盤センター

Information Technology Center, The University of Tokyo

^{†2} 東京大学大気海洋研究所

Atmosphere and Ocean Research Institute, The University of Tokyo

^{†3} 東京大学大学院情報学環/地震研究所

Graduate School of Interdisciplinary Information Studies/Earthquake Research Institute, The University of Tokyo

^{†4} 東京大学人工物工学研究センター

Research into Artifacts, Center for Engineering, The University of Tokyo

^{†5} 京都大学学術情報メディアセンター

Academic Center for Computing and Media Studies, Kyoto University

^{†6} 海洋研究開発機構

Japan Agency for Marine-Earth Science and Technology (JAMSTEC)

^{†7} 科学技術振興機構 戦略的創造研究推進事業 (CREST)

Japan Science and Technology Agency (JST), Core Research for Evolutional Science and Technology (CREST)

Method, FEM), 差分法 (Finite Difference Method, FDM), 有限体積法 (Finite Volume Method, FVM), 境界要素法 (Boundary Element Method, BEM), 個別要素法 (Discrete Element Method, DEM) に限定し, 各手法の特性に基づきハードウェアに依存しない共通インタフェースを有するアプリケーション開発用ライブラリ群, 耐故障機能を含む実行環境を実現する。

ppOpen-HPC は, 図 1 に示すように:

- ppOpen-APPL: 各手法に対応した並列プログラム開発のためのライブラリ群
- ppOpen-MATH: 各離散化手法に共通の数値演算ライブラリ群
- ppOpen-AT: 科学技術計算のための自動チューニング (AT) コンパイラ
- ppOpen-SYS: ノード間通信, 耐故障機能に関連するライブラリ群

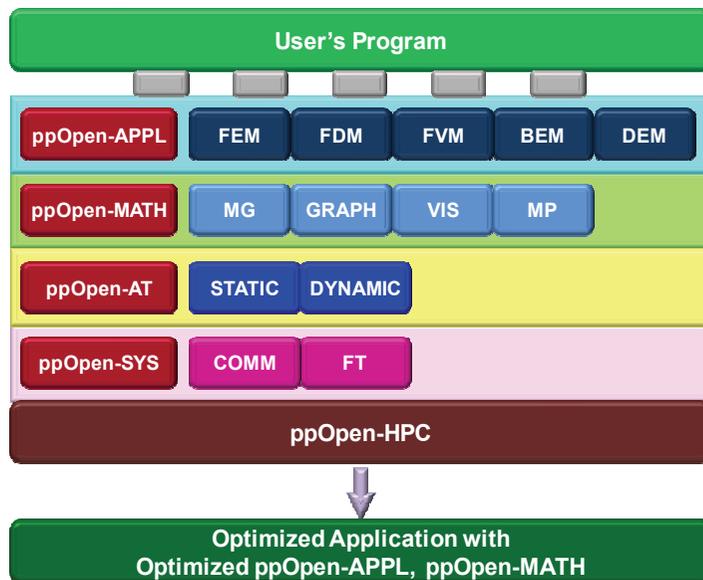


図 1 ppOpen-HPC の構成

から構成される。ユーザープログラムから ppOpen-APPL, ppOpen-MATH, ppOpen-SYS のライブラリ群を呼び出し, 自動チューニングコンパイラ ppOpen-AT を使用することによって, ハイブリッド並列化・CUDA・OpenCL 等に煩わされることなく, またハードウェアに依存したチューニング, ホスト CPU~アクセラレータ・コプロセッサ間の負荷分散等を考慮することなく, ポストペタスケールシステムに対して最適化された

耐故障機能をもつアプリケーションを容易に開発できる。また, 最適化されたライブラリ群が自動的に生成される。メモリアクセス最適化, 資源配分に関するハードウェア依存のチューニングのプロセスを ppOpen-AT に集約することにより, 計算機環境が変化しても, ppOpen-AT の変更によって対応できる。

本研究の成果は, 平成 26 年度に本研究の中核機関である東京大学情報基盤センター (以下 東大 ITC) [3] に導入予定の数十ペタフロップス (PF) 級システム上で検証, 改良, 更に公開し, 大規模シミュレーションによる計算科学の進歩に貢献する。更に改良を進めることによって, エクサスケールシステムにおけるアプリケーション開発・実行環境へと発展させていくことを目標とする。

2. 背景

科学技術計算に携わる科学者, 技術者にとって, 大規模化, 複雑化, 多様化するハイエンド計算機環境の能力を十分に引き出し, 効率的なアプリケーションプログラムを開発することは困難である。有限要素法等の科学技術計算手法は, プリ・ポスト処理, 行列生成, 線形方程式求解等の一連の共通プロセスから構成される。これらをハードウェアに応じた最適化を施したライブラリとして整備することで, アプリケーション開発者から共通プロセスに関わるプログラミング作業, 並列化も含むチューニング作業を隠蔽できる。このようなライブラリ群はアプリケーションとコンパイラ間のギャップを埋める HPC ミドルウェアと呼ばれ, 地球シミュレータについて 2000 年前後に開発された GeoFEM [4], HPC-MW [5] 等がある (図 2)。

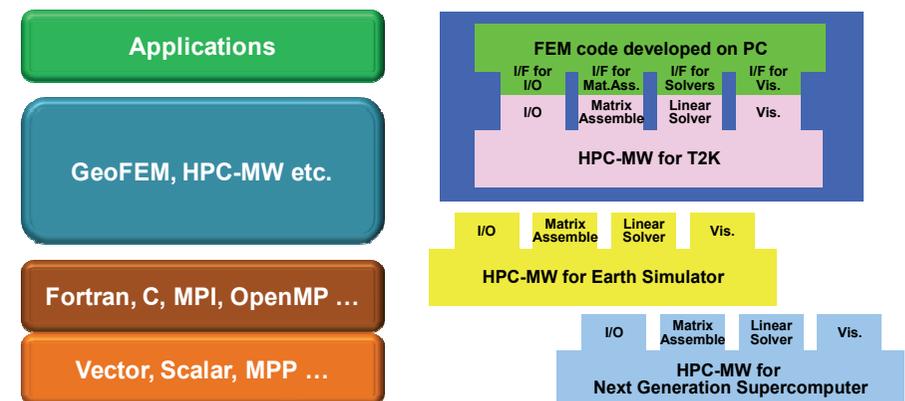


図 2 アプリケーションとコンパイラ, ハードウェア間のギャップを埋める HPC ミドルウェア, 有限要素法向け HPC-MW の例

GeoFEM, HPC-MW では有限要素法の共通プロセスのプログラミング, MPI による並列化, ベクトル計算機向け最適化チューニング関連の作業の隠蔽に成功した。スカラ計算機はより複雑なメモリ階層性を有し, アプリケーション性能がキャッシュ容量も含むアーキテクチャの変化の影響を大きく受けるため, 異なるアーキテクチャごと, あるいは計算機システムのリプレースごとに最適化されたライブラリを別個に整備しなければならず, コストを要し, 様々なノウハウを継承していくことも困難である。

近年, マイクロプロセッサのメモリバンド幅の相対的低下傾向の中で, 高いメモリバンド幅を持つ GPU 等のアクセラレータが科学技術計算向けハードウェアとして注目されており CPU と組み合わせたヘテロジニアスなアーキテクチャが使用されている。様々な技術的問題のために, 陽解法による差分法等に用途は限定されている。また, CUDA, OpenCL 等の環境が普及し, HMPP [6] に代表されるコードジェネレータも利用可能となつてはいるものの, 一般的にプログラミングは困難である。GeoFEM, HPC-MW の開発から約 10 年が経過したが, ハイエンド計算機環境はより多様化し, 変化も激しいため, HPC ミドルウェアの必要性は当時よりむしろ高くなっている。アプリケーション開発者の立場から考えると, アクセラレータ, コプロセッサの導入などにより複雑化の一途をたどるハイエンド計算機システムにおいて, 高い計算性能を達成できるようなプログラムを開発することは非常に困難であり, 高性能で安定に稼働するアプリケーションプログラムを開発するための環境が求められている。

ppOpen-HPC では, 大規模アプリケーション開発に豊富な経験と知見を有する諸機関の協力を得て, 代表的な 5 つの分散化手法をサポートする HPC ミドルウェアを研究開発し, それを支える数値演算ライブラリ, システムソフトウェア, ハードウェア依存チューニングを受け持つ自動チューニングコンパイラを開発する。数値演算ライブラリは個別のハードウェアに対するチューニングが必要であるが, 計算機システムが変わるとのチューニング作業は困難であり, 自動チューニング機構は重要である。

3. 開発内容

3.1 前提とするアーキテクチャ

本研究においてターゲットとするポストペタスケールシステムは, ヘテロジニアスな計算ノードを有するアーキテクチャとし, 以下のように仮定する:

- 各計算ノードは, 以下の 2 つの部分から構成されるものとする:
 - 8~16 コア程度を有するマルチコア低消費電力 CPU (ホスト CPU)

- 数十~数百のコアを有する複数のアクセラレータ, コプロセッサ
- アクセラレータ, コプロセッサは PCI Express に接続され以下の 2 つのケースを考慮する:
 - GPGPU: ホスト CPU から CUDA または OpenCL によって利用する
 - Intel Many Integrated Core (MIC) アーキテクチャ [7] に基づくメニーコア (Knights Ferry, Knights Corner 等): 軽量 OS, コンパイラが稼働する
- ノード間ネットワークは Infiniband のような高速ネットワークを想定する
- アクセラレータ, コプロセッサ間の通信は CPU を経ないで可能とする

本研究で前提とするヘテロジニアスな計算ノードでは, メッセージパッシングとスレッド並列によるハイブリッド並列プログラミングモデルが必須である。

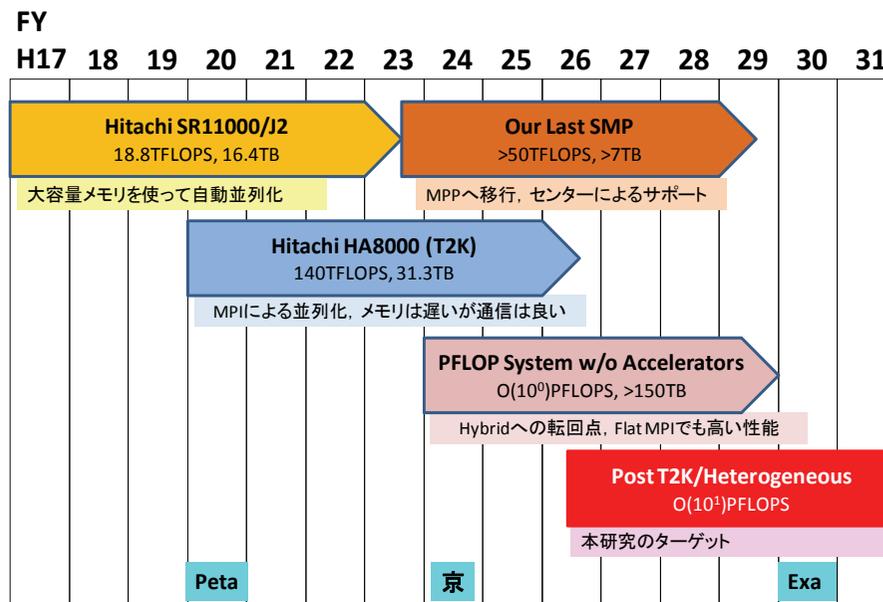


図3 東京大学情報基盤センター(東大 ITC)スーパーコンピュータシステムの導入スケジュール

図3は東京大学情報基盤センター(東大 ITC)におけるスーパーコンピュータシステムの導入スケジュールである。東大 ITC では現在 2 システム (Hitachi SR11000, Hitachi

HA8000 (T2K 東大 [8])) のを運用しているが、平成 23 年度から 24 年度にかけて、Hitachi SR11000 がリプレースされ、ペタフロップス級の超並列計算機システム (アクセラレータ, コプロセッサを持たない) が導入される。本研究で最終的なターゲットとしているのは平成 26 年度に導入が予定されている数十ペタフロップス級のヘテロジニアスな計算ノードを有するアーキテクチャによるポストペタスケールシステムであるが、それに至るまでの「京」も含めた様々なシステムを念頭に置いた研究開発を実施する。

3.2 開発内容の概要

1. で示した①~⑤の 5 つのキーワードを念頭に置き、上記アーキテクチャを有するポストペタスケールシステム向けのアプリケーション開発・実行環境として図 1 に示す 4 つのレイヤーを持つ ppOpen-HPC の研究開発を実施する。利用者がハイブリッド並列化, CUDA, OpenCL 等に煩わされることなく、また、ホスト CPU, アクセラレータ, コプロセッサの利用を意識することなく、ポストペタスケールシステムに対して最適化された耐故障機能を持つアプリケーションを開発できる環境を目指す。

ppOpen-HPC では利用者が独自にプログラム開発することを想定するが、OpenFOAM 等のオープンソースコードについても、若干のインタフェース改良により利用可能である。

近年、プロセッサのマルチコア化が進み、大規模システムにおいては、メッセージパッシング (ノード間, 例: MPI) とスレッド並列 (ノード内, 例: OpenMP, Pthread) を組み合わせたハイブリッド並列プログラミングモデル [9] の重要性は広く認識されている。しかしながら、MPI による並列化は普及しているものの、ハイブリッド並列プログラミングモデルの実アプリケーションへの導入は進んでいない。本研究では、すでに MPI による並列化を実施しているアプリケーションについて、ppOpen-HPC を使用することによって、図 3 に示したようなスパコンの導入スケジュールの中でハイブリッド化、ヘテロジニアスな構成の計算ノードへの対応をスムーズに実現することを念頭においている。

以下に図 1 に示す 4 つのレイヤーについて簡単に説明する。

3.2.1 ppOpen-APPL

ppOpen-APPL は、図 2 に示す HPC ミドルウェアに相当し、ユーザープログラムに最も近いレイヤーである。ppOpen-APPL は有限要素法 (FEM), 差分法 (FDM), 有限体積法 (FVM), 境界要素法 (BEM) (特に高速多重局展開法 (Fast Multipole Method, FMM) による FMM-BEM に注目), 個別要素法 (DEM) の 5 種類の離散化手法をサポートする (図 4)。境界要素法としては FMM-BEM を想定すると、これらはいずれも隣接領域との相互作用に基づくローカルな離散化手法であり、大規模シミュレーション

により適した手法である。ppOpen-APPL は、これらの手法に関して、

- (a) データ入出力部
- (b) 領域間通信
- (c) 係数マトリクス生成部 [10]
- (d) 各離散化手法向け前処理付き反復法
- (e) 適応格子・動的負荷分散 [11]

等の共通機能を提供するライブラリ群であり、FORTRAN, C, C++, OpenCL 等によるユーザープログラムから呼び出せるハードウェアに依存しない共通インタフェースを備える。

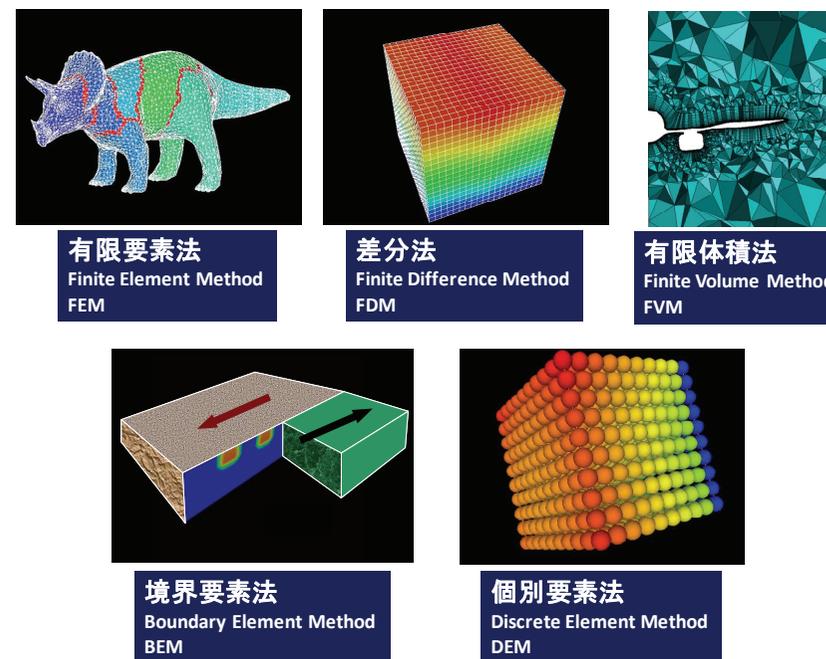


図 4 ppOpen-APPL の対象とする離散化手法

各離散化手法をサポートする ppOpen-APPL は以下のような機能を有する：

- FORTRAN, C, C++, OpenCL 等によるユーザープログラムから呼び出せるハードウェアに依存しない共通インタフェースを備える。
- FORTRAN, C, C++等のプログラミング言語, OpenMP+MPI ハイブリッド並列プログラミングモデルによって記述されている。
- 機能の一部については CUDA または OpenCL によって記述する。
- 陽解法, 陰解法の両方のプログラムを開発可能とする。
- 陽解法, 係数マトリクス生成部については自動的にプログラムを生成するためのテンプレート機能を備え, GPGPU 向けに最適化されたプログラムを生成するためのテンプレート機能を併せて備えるものとする。

3.2.2 ppOpen-MATH

ppOpen-MATH は, 大規模シミュレーションに必須の数値処理機能を実現する共通基盤ライブラリ群であり, MG (多重格子法), GRAPH (グラフ処理), VIS (可視化), MP (カップリング) から構成され, ppOpen-APPL, ユーザープログラムから呼び出せる。各離散化手法の特性を生かし, 様々な問題に対応可能とする。各コンポーネントの概要は以下の通りである：

- **ppOpen-MATH/MG**：幾何学的・代数学的多重格子法を前処理手法とし, 様々な問題を安定かつ効率的に解けるような機能を提供する Krylov 部分空間型反復解法ライブラリ [12]
- **ppOpen-MATH/GRAPH**：領域分割・再分割, リオーダーリングなどのグラフ処理を実施するライブラリ。
- **ppOpen-MATH/VIS**：局所細分化を含むボクセル型背景格子を使用したリアルタイム可視化ライブラリ [13]
- **ppOpen-MATH/MP**：複数のアプリケーションを結合し, 大規模データ転送, 変換を実施する弱連成のためのカップリングツール。以下のような機能を有する：
 - ppOpen-APPL のサポートする様々な離散化手法を全てサポートし, 二次元, 三次元カップリング機能を提供する。異なった離散化手法によるモデル間のカップリングも可能とする。時間積分手法に応じて複数のプロセスをパイプライン処理する機能を提供する。
 - 計算プロセスだけでなく, シミュレーションと ppOpen-VIS の並列実行も可能とする。
 - 複数のアプリケーションに対して, 利用可能な計算資源を最適に配分

する機能 (M×N 配分機能) を開発する。その際, ppOpen-MATH/GRAPH を利用する [14]。

3.2.3 ppOpen-AT

科学技術計算向け自動チューニングコンパイラ ppOpen-AT は, 本研究を特徴づけるものであり, ユーザープログラムへの簡単なディレクティブの挿入により, 以下の機能を実現する：

- ppOpen-APPL の係数マトリクス生成部 (陽解法を含む), 疎行列ソルバーについて, 自動チューニングによりメモリアクセスパターンの最適化を環境 (計算機アーキテクチャ, 計算資源, 問題サイズ) に応じて適応的に実施する。
- ppOpen-MATH/MG の演算カーネルについて, 自動チューニングによりメモリアクセスパターンの最適化を環境 (計算機アーキテクチャ, 計算資源, 問題サイズ) に応じて適応的に実施する。
- ヘテロジニアスなノード構成を持つアーキテクチャにおいて, ホスト CPU とアクセラレータ, コプロセッサ間の最適な資源配分を環境 (計算機アーキテクチャ, 計算資源, 問題サイズ) に応じて適応的に決定する。
- アクセラレータを使用する場合に最適化された CUDA, OpenCL により記述されるコードを自動生成することを可能とする。

ppOpen-APPL の 5 つの離散化手法はいずれもメモリ bound な手法であるため (境界要素法としては FMM-BEM を想定), メモリアクセスパターンの最適化に主眼を置くことは重要である。ppOpen-AT/STATIC は独立したコンパイラであり, プログラム, ppOpen-APPL, ppOpen-MATH の各機能の最適化を静的に行う。ppOpen-AT/DYNAMIC はランタイム時に最適化を行うライブラリである。本研究では既存の AT 言語 ABCLibScript [15] を元に, ppOpen-APPL, ppOpen-MATH の主演算部分に特化した最適化機能を新たに開発する。ppOpen-APPL, ppOpen-MATH 大規模並列計算システム上でのアプリケーション開発を容易にするが, 必ずしも最適の性能が得られるわけではない。ppOpen-AT は, ppOpen-APPL, ppOpen-MATH, 及びユーザープログラムの一部の機能に限定して, 環境 (計算機アーキテクチャ, 計算資源, 問題サイズ) に応じて最適化されたプログラムを自動生成する機能を提供する。ppOpen-AT を使用すれば, 計算機環境が変化しても最適な ppOpen-APPL, ppOpen-MATH を自動生成することが可能であり, アプリケーション開発者は FORTRAN, C, C++等で記述した同じプログラムを利用し続けることができる。

3.2.4 ppOpen-SYS

ppOpen-SYS はシステムソフトウェア寄りのレイヤーであり、ノード間通信ライブラリ (ppOpen-SYS/COMM)、耐故障機能記述ライブラリ (ppOpen-SYS/FT) から構成される。大規模並列計算において、計算ノード間の通信は必須であるが、本研究で想定しているアプリケーションは隣接領域との通信が主であるため、通信ライブラリに必要な機能も限定される。本研究のターゲットとするポストペタスケール環境でより効率の良い通信を実現するため、ノード間通信ライブラリ ppOpen-SYS/COMM を開発する。また、アプリケーション開発者に、容易に耐故障アルゴリズムを記述できる環境を提供するライブラリとして ppOpen-SYS/FT を開発する。ポストペタスケール、エクサスケールシステムは数百万から数千万規模のコアを有するため、耐故障性 (Fault Tolerance) はシステムの備えるべき必須の機能である。ppOpen-SYS/FT はアプリケーションの特性に応じた柔軟な対応が可能であり、効率的に耐故障機能を提供する。

3.3 開発体制・スケジュール

本研究は本稿の6人の著者が所属する6つの研究機関の、数値ライブラリ、システ

ムソフトウェア、アプリケーションの専門家の緊密な協力のもとに実施する。各機関は地球シミュレータ、T2K オープンスパコン等の大規模計算機システムの利用経験が豊富である。ppOpen-APPL は実際に稼働しているアプリケーションに基づき、その分野の専門家が、様々な知見、経験によるノウハウを取り入れて開発する。成果物は表1に示すように適宜公開し、ppOpen-APPL 開発者自身、全国の大型計算機センター及び「京」のユーザーの協力も得て、ペタスケールシステム上での予備的検証、ターゲットとするポストペタスケールシステム (平成26年度東大ITCに導入予定、数十PF級)での検証、改良を実施する。また、平成25年度前半に試験環境 (メニーコア搭載小型クラスタ) を導入し、ターゲットとするポストペタスケールシステム稼働開始と同時に ppOpen-HPC が利用できるように準備をおこなう。各機関と協力して講習会を実施し、大学・大学院での科学技術計算プログラミング教材としての利用により更なる普及に資する。本研究では、平成24年秋以降、研究期間中に4回、成果物を公開する予定である。その時期に合わせてマイルストーンを設定する。第3回、第4回の公開はターゲットとするポストペタスケールシステムを対象とするが、それ以外のマルチコアクラスタ、GPUクラスタ向けの改良、最適化も並行して継続する。

表1 ppOpen-HPC の公開スケジュール

	公開内容	趣旨・目標	ターゲットマシン	開発環境
第1回 H24秋	ppOpen-APPLプロトタイプ (適応格子・動的負荷分散を除く)、領域分割ユーティリティ、ppOpen-MATH/VIS・ppOpen-AT/STATICプロトタイプ	ppOpen-APPL, ppOpen-ATが様々なマルチコアクラスタ、GPUクラスタで稼働することを検証する。大学 (学部、大学院) 講義への導入、講習会開催を継続的に実施する。	東大T2K 東大次期システム 京コンピュータ GPUクラスタ	東大T2K 東大次期システム GPUクラスタ (東大) 京コンピュータ
第2回 H25秋	ppOpen-HPC (ppOpen-AT/DYNAMICを除く)	ppOpen-HPCの各機能 (ppOpen-AT/DYNAMICを除く) をこの時期までに一通り完成する。実アプリケーションへ適用することにより更なる改良を図る。	東大次期システム 京コンピュータ GPUクラスタ	東大次期システム GPUクラスタ (東大) 京コンピュータ
第3回 H26秋	ポストペタスケールシステム向けppOpen-HPCプロトタイプ	ターゲットとするポストペタスケールシステム向けのppOpen-HPCをマシン稼働開始と同時に利用可能とする。	ポストT2Kシステム	東大次期システム 小型メニーコアクラスタ 京コンピュータ
第4回 H27秋	ポストペタスケールシステム向けppOpen-HPC完成版	第3回で公開したppOpen-HPCを実アプリケーションへ適用することにより、更に改良、最適化した最終バージョンを公開する。	ポストT2Kシステム	ポストT2Kシステム 東大次期システム 小型メニーコアクラスタ 京コンピュータ

注：①東大T2K：平成20年6月より稼働、ピーク性能140TFLOPS、②GPUクラスタ (東大)：平成23年4月に東大ITCに導入されるGPUクラスタ、NVIDIA TeslaM2050 32台搭載、③東大次期システム：平成24年4月に東大ITCに導入されるスーパーコンピュータ、ピーク性能1PFLOPS程度、④小型メニーコアクラスタ：平成25年度前半に東大ITCに導入されるメニーコアクラスタ、⑤ポストT2Kシステム：本研究のターゲットとするポストペタスケールマシン、平成26年度に東大ITCに導入、ピーク性能数十PFLOPS

4. 研究上の着目点

本研究は、1. の冒頭であげた 5 つの項目に着目して実施するが、本章では特に注目する以下の事項について説明する：

- マトリクス格納方法を含めたデータ構造
- 通信量・通信回数削減
- 混合精度演算

4.1 マトリクス格納方法を含めたデータ構造

データ構造が計算効率に大きく影響することはよく知られている。例えば GPU を利用した計算で効率を高めるために一般に利用されている *coalesced memory access* は、T2K（東大）のような ccNUMA アーキテクチャに基づくマルチコア CPU では高い効率を得ることができず、*sequential memory access* が一般に使用されている（図 5, 図 6）。

また、疎行列計算では、係数マトリクス格納法が性能に大きく影響し、最適な格納法は行列の性質、サイズ、計算機環境に左右されることも広く知られている [16]。また、ICCG 法等におけるデータ依存性を回避するために導入されている *reordering* の最適な色数が疎行列ベクトル積の性能、ICCG 法の収束でそれぞれ大きく異なっているような例もある [12]。

計算効率を高めるためにデータの *reordering* は不可欠であるが、ppOpen-AT による自動チューニングも含めて全体の計算プロセスを効率的に実施するためには、ppOpen-HPC 内部におけるデータ移動量が少なくなるような配慮が必要である。差分

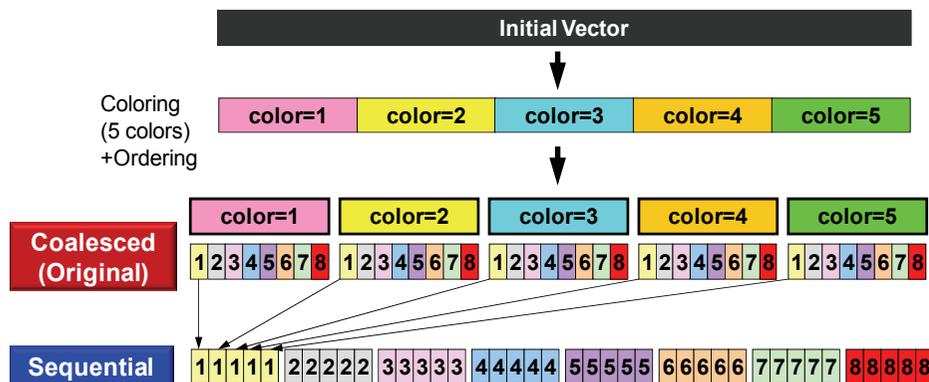


図 5 連続データアクセスのためのデータ再配置（5色，8スレッドの場合）[9]

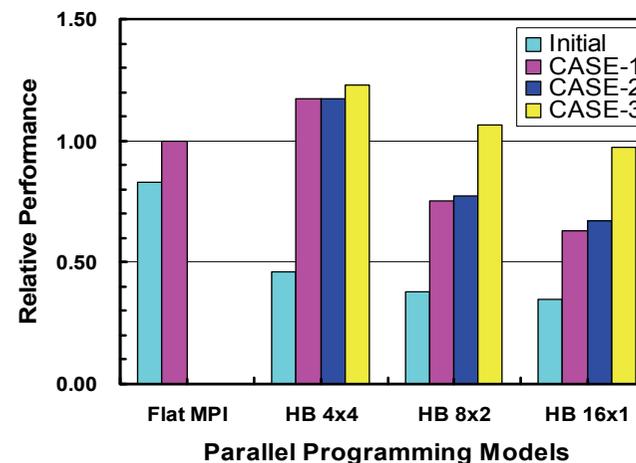


図 6 三次元有限要素法（不均質場，固体力学）による前処理付き反復法の計算性能（128³要素，6,291,456 自由度，T2K（東大）512 コア（32 ノード）使用）[9]

Initial ■ : Coalesced, CASE-1 ■ : Coalesced+NUMA, CASE-2 ■ : Coalesced+NUMA + First-Touch, CASE-3 ■ : Sequential+NUMA+First-Touch

法のような規則性のあるデータ構造では、最適な番号付けは事前に予測することが可能である。有限要素法のような不規則なデータ構造の場合でも、要素～節点の関係から事前に最適な番号付けの候補は絞り込むことが可能である。本研究では、そのような検討も行う。

また、悪条件問題の場合、各要素のサイズ、材料特性等から ILU 型前処理手法におけるフィルイン位置を特定するなど、極力計算の早い段階で最適な番号付けを決定できるような手法を検討する。

4.2 通信量・通信回数削減

ポストペタスケールシステムではコア数が数百万から数千万のオーダーになるため、プロセス間の通信の効率化は必須である。本研究の場合、通信は：

- 領域境界値の周辺領域との交換（一対一通信）
- スカラー値，小規模配列の Allreduce 集合通信（内積，最大・最小値の計算）

などが通信で、通信量そのものは比較的少なく、レイテンシが問題となるケースが多くなる。マルチコアクラスタではハイブリッド並列プログラミングモデルの採用によ

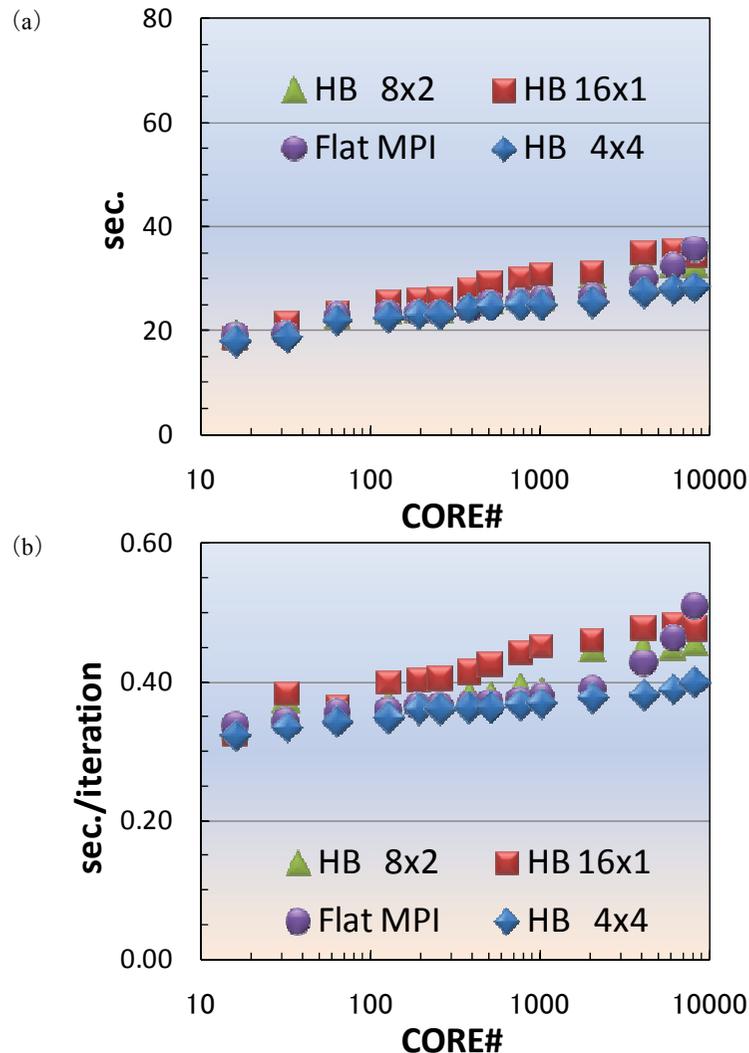


図7 MGCG法の計算性能 ((a) MGCG ソルバー計算時間, (b) MGCG ソルバー1 反復あたり計算時間) (最適化されたソルバーを適用, CM-RCM(2), 16~8,192 コア, T2K (東大)) (Weak Scaling, 262,144 セル/コア, 最大問題規模: 2,147,483,648 セル) (不均質多孔質媒体中の三次元地下水流れ) [12]

り MPI プロセス数を減らすことができる。図7は三次元不均質場におけるポアソン方程式をマルチグリッド前処理付き共役勾配法 (MGCG 法) で T2K (東大) の 512 ノード (8,192 コア) までを使用して解いた例 (Weak Scaling) である [12]。1 ソケット単位で MPI プロセスを発生させ、ソケット内は 4 スレッドの OpenMP によっている HB4 × 4 が、メモリアクセス、通信のバランスがとれており最も性能が良い。反復回数は問題規模に関わらずほぼ一定であるが、各プログラミングモデルともコア数が増えるに従って一反復当たりの計算時間が増す傾向にある。

マルチグリッド系の手法では、特に粗い格子レベルにおいて、通信のオーバーヘッドによる性能低下が顕著になる。図7に示した手法では格子数が MPI プロセス数の合計より小さい場合は、1 コアのみでマルチグリッド法の計算を実施しているため、コア数が増加するこの並列化されていない部分の影響が顕著となる可能性がある。

ポストペタスケールシステムではこのような事態に対処するため、計算途中での MPI プロセス数の変更などを適応的に実施する機能の研究開発が必要である。

4.3 混合精度演算

混合精度演算 (Mixed Precision Computation) は 1. であげたポストペタスケール、エクサスケールシステムのキーワードとなっており、特に必要最低限の精度で計算を実施することは、計算効率、電力消費量という観点からも重要である。

本研究では前処理付反復解法について Iterative Refinement [17] に基づき、行列の性質に応じて最適なパラメータの自動設定手法を、本研究でターゲットとするヘテロジニアスなアーキテクチャを有する計算ノードを対象として実施する予定である。

5. 関連研究

科学技術計算の手法ごとに共通処理をライブラリ化するという HPC ミドルウェア (図2) のアイデアは、ごく自然な発想に基づくものであり、既存の類似研究は数多い。近年は更なる利用者の負担軽減を目的として、Domain Specific Language (DSL)、スクリプト型言語によるプログラム自動生成環境の研究も盛んになっている [18]。

本研究の独創性、新規性は自動チューニングコンパイラ ppOpen-AT によりあらゆる計算機環境において適応的に最適化されたライブラリ、アプリケーションを自動生成でき、アプリケーション開発者だけでなくライブラリ開発者の負担も大きく軽減できることにある。例えば、OpenMM (Stanford 大学) [19] は GPU 向けに最適化された機能を有する分子動力学コード開発用ライブラリ群である。ハードウェアに依存しない共通インタフェースを有するが、自動チューニングのアイデアは取り入れられていない。

Oak Ridge National Laboratory (ORNL) が中心となって、米国エネルギー省 (US Department of Energy, US-DOE) の研究所群、各ベンダーにより Hybrid Multicore Consortium (HMC) [20] が組織され、様々な分野の専門家が協力して CPU・GPU 複合システムに向けた取り組みを実施している。現在はターゲットアプリケーションの選定と特徴抽出の段階で具体的な方策は未提示である。基本的考え方は本研究の目指すところと全く同じであり、エクサスケールシステム実現へ向けての国際的な枠組みである、International Exascale Software Project (IESP) [21] も含めて国際的に連携して行きたい。

6. まとめ・将来展望

自動チューニング機構を有するアプリケーション開発・実行環境、ppOpen-HPC の概要について述べた。

ppOpen-HPC によって、ポストペタスケールシステムにおける大規模アプリケーション開発環境が提供され、計算機システムの性能を最大限引き出すことのできるアプリケーションを短期間で容易に開発可能となり、大規模シミュレーションによる計算科学研究の進展を加速することが期待される。

ppOpen-HPC は東大 ITC に平成 26 年度に導入されるポストペタスケールシステムをターゲットとしているが、様々なマルチコアクラスタ、GPU クラスタ、メニーコアクラスタでの利用が可能である。また、PC、小規模クラスタなどでのアプリケーション開発環境（「並列」に限らない）としても利用できる。オープンソースとして公開することによって、研究室から大型計算機センターまで、共通のアプリケーション開発環境として展開が可能であり、教育に使用することにより、持続的な人材育成に貢献できる。大学・大学院での講義に利用する他、各センター、学会、産業界と共同した集中講義、講習会も実施する。並列プログラミング教育の共通教材として利用することで、全国的、国際的な普及を図る予定である。

参考文献

- [1] ppOpen-HPC : <http://ppopenhpc.cc.u-tokyo.ac.jp/>
- [2] 科学技術振興機構戦略的創造研究推進事業 (CREST) 研究領域「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出」
<http://www.jst.go.jp/kisoken/boshuu/h22/2ndterm/teian/koubo/04-ch2202.html>
- [3] 東京大学情報基盤センター (Information Technology Center, The University of Tokyo) :
<http://www.cc.u-tokyo.ac.jp/>
- [4] Okuda, H., Nakajima, K., Iizuka, M., Chen, L. and Nakamura, H., Parallel Finite Element Analysis

- Platform for the Earth Simulator: GeoFEM, Lecture Notes in Computer Science 2659, 773-780 (2003)
- [5] 奥田洋司・中島研吾, HPC ミドルウェア, 生産研究 55-3, 49-56 (2003)
 - [6] HMPP : http://www.caps-entreprise.com/fr/page/index.php?id=36&p_p=36
 - [7] Intel Many Integrated Core Architecture :
<http://www.intel.com/technology/architecture-silicon/mic/index.htm>
 - [8] The T2K Open Supercomputer Alliance: <http://www.open-supercomputer.org/>
 - [9] Nakajima, K.: Flat MPI vs. Hybrid: Evaluation of Parallel Programming Models for Preconditioned Iterative Solvers on “T2K Open Supercomputer”, IEEE Proceedings of the 38th International Conference on Parallel Processing (ICPP-09), 73-80 (2009)
 - [10] 大島聡史・林雅江・片桐孝洋・中島研吾, 三次元有限要素法アプリケーションの CUDA 向け実装と性能評価, 情報処理学会研究報告 (HPC-129-20) (2011)
 - [11] 中島研吾, 六面体メッシュの適合型並列局所細分化と負荷分散, 情報処理学会研究報告 (HPC-116), 163-168 (2008)
 - [12] 中島研吾, マルチコア時代の前処理手法, 京都大学数理解析研究所講究録 No.1374 「科学技術計算アルゴリズムの数理的基盤と展開」, 1-10 (2011)
 - [13] 中島研吾・陳 莉, ボクセル型背景格子を使用した大規模並列可視化手法, 情報処理学会研究報告 (HPC-107), 91-96 (2006)
 - [14] 中島研吾・長嶋利夫・江連真一・市村強・奥田洋司, M×N 通信による大規模連成シミュレーション, 情報処理学会研究報告 (HPC-109), 215-220 (2007)
 - [15] Katagiri, T., Kise, K., Honda, H. and Yuba, T., ABCLibScript: a directive to support specification of an auto-tuning facility for numerical software, Parallel Computing 32-1, 92-112 (2006)
 - [16] 片桐孝洋・櫻井隆雄・黒田久泰・直野健・中島研吾, Xabclib : 汎用的自動チューニングインターフェース OpenATLib を利用した反復解法ライブラリの開発, 特集: 数値計算のための自動チューニング, 応用数理 (日本応用数理学会会誌) 20-4, 25-37 (2010)
 - [17] Georgescu, S. and Okuda, H., Conjugate Gradients on Graphic Hardware: Performance & Feasibility, Proceedings of 9th International Workshop on State-of-the-Art in Scientific and Parallel Computing (PARA 2008) (2008)
 - [18] Escript/Finley : <http://www.uq.edu.au/esscc/escrpt-finley>
 - [19] OpenMM : <https://simtk.org/home/openmm>
 - [20] Hybrid Multicore Consortium (HMC) : <http://computing.ornl.gov/HMC/>
 - [21] International Exascale Software Project (IESP) : http://www.exascale.org/iesp/Main_Page