「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」
平成 23 年度採択研究代表者

H27 年度
実績報告書

滝沢寛之

国立大学法人東北大学大学院情報科学研究科
准教授

進化的アプローチによる超並列複合システム向け開発環境の創出

§1. 研究実施体制

(1)滝沢グループ(東北大学大学院情報科学研究科)
①研究代表者:滝沢　寛之(東北大学大学院情報科学研究科、准教授)
②研究項目
・システムの複合性を扱うためのプログラミングインタフェースの設計
・HPC アプリケーションのためのリファクタリングツールの開発

(2)高橋グループ(筑波大学)
①　主たる共同研究者:高橋　大介　(筑波大学システム情報系、教授)
②　研究項目
・超並列複合システムに向けた高速フーリエ変換(FFT)ライブラリおよび代数的多重格子法ライブラリの開発
・混合精度計算に対応した基本線形計算ライブラリの基本設計

(3)須田グループ(東京大学)
①　主たる共同研究者:須田礼仁　(東京大学情報理工学系研究科、教授)
②　研究項目
・ドメイン特化型ツールによる超並列複合システムの階層的抽象化に関する研究

(4)江川グループ(東北大学サイバーサイエンスセンター)
①　主たる共同研究者:江川　隆輔(東北大学サイバーサイエンスセンター、准教授)
②　研究項目
・HPC のためのリファクタリングカタログの設計
・超並列複合システム向けのアプリケーション開発手法の確立

§2. 研究実施の概要

現在、多くの高性能計算アプリケーションは特定のシステムを想定し、その想定システム上で高性能を達成できるように開発・最適化されている。このため、そのアプリケーションを他のシステム上で実行する場合には、高い性能を達成できない恐れがある。すなわち、性能可搬性が低いことが問題となっており、既存のアプリケーション資産を新システムに移行する場合にも同様の問題が発生する。このため本研究では、既存のアプリケーション資産の性能可搬性を高めることで、その資産の将来のシステムへの円滑な移行を実現する。そのためには性能可搬性の阻害要因を大規模なアプリケーションコードから分離する必要があり、書き直しの困難な規模のアプリケーション資産を対象に考える場合には、コードを段階的に再構成することで分離を実現する必要がある。この分離作業を、本研究では HPC リファクタリングと呼ぶ。

本研究の大きな目標の一つは、事例に基づいて性能可搬性の阻害要因を体系的にまとめ、それらを切り離す HPC リファクタリングの方法論をまとめることである。平成 27 年度には、継続してリファクタリングカタログの拡充に継続して取り組んだ。特に、アクセラレータや、東北大学サイバーサイエンスセンターに導入されたベクトル型計算機への移植および最適化を通じて、カタログのコンテンツの充実に取り組んだ。また、他グループで開発しているツールを活用することで、カタログの事例の変換ルールを生成し、カタログのコンテンツに変換ルールを加えた。 これらの取り組みの有用性を大気海洋大規模シミュレーションである MSSG カーネル群の OpenACC への移植を通して明らかにしている。

また、HPC リファクタリングによってアプリケーションコードから切り離された情報は、特定のシステムで性能を出すための情報と言い換えることができ、単に切り離すだけでは性能が低下する。このため、各システムで性能を出すための情報をアプリケーションコードとは別の存在として抽象化し、システムに合わせて使い分けることで、それらのシステム間の性能可搬性を実現することを考えている。この抽象化のための最も一般的な方法は、汎用性の高い計算を数値計算ライブラリとして実装することである。平成 27 年度には、数値計算ライブラリにより超並列複合システムの階層的抽象化を行うために、アプリケーションプログラムをなるべく修正せずに高い性能を達成できるライブラリの研究開発を行った。また、疎行列ベクトル積や高速フーリエ変換などにおいて Xevolver フレームワークを用いたコード変換について検討を行い、性能可搬性を改善できることを確認した。また、既存のアプリケーション資産を利用して高精度計算を行うために、Xevolver フレームワークを用いて、C 言語の倍精度コードを GMP（GNU Multi-Precision Library）を用いた任意多倍長コードへ自動変換する機構を開発した。さらには、ポストペタ時代の計算科学で必要性が明確化されてきた、新世代アルゴリズムの研究も進めている。超大規模計算で主要なオーバーヘッドとなる通信を削減したアルゴリズムとして、チェビシェフ基底 CG 法、ブロックチェビシェフ基底 CG 法の研究を行った。またこれらのアルゴリズムの基本演算である行列冪カーネルについて、従来法では通信回数が1回で済むが計算の重複が大量に発生するところを、通信回数はやや増えるが計算の重複がないアルゴリズムを提案した。

上記のように抽象化する仕組みを用意しても、全てを抽象化することはできないため、アプリケーションコードの修正が求められる。そのようなコード修正は一般化が難しく、特定のシステムやアプリケーション向けの修正となる傾向にある。そのようなコード変換も何らかの形でアプリケーションコー

ドから切り離し、再適用できる仕組みがなければ、HPC リファクタリングされたコードで元のコードと同じ性能を出すことはできない。このため本研究では、そのようなシステム特有、アプリケーション特有のコード修正を、ユーザ定義変換として定義するための仕組みを構築してきた。平成 27 年度は、これまでに構築した Xevolver フレームワークの適用事例を積み重ね、その有用性と課題について議論を深めた。典型的なループ最適化やデータ構造の変換などをユーザ定義コード変換として表現し、これらの重要な性能最適化技法を、アプリケーションコードを大きく修正することなく適用可能であることを実証した。さらに、OpenACC 向けの最適化技法を事前定義のコード変換ルールとして用意し、その事前定義変換へ実際のアプリケーションコードを適応させるためにユーザ定義変換を用いるという使い分けを検討した。ユーザ定義コード変換のルールを記述するための高水準インターフェースとして、ディレクティブを含む Fortran のダミーコードで変換を記述する xevtgen を実装・評価し、その有効性と限界についても議論した。