大規模並列粒子シミュレーションコード開発用フレームワーク FDPS 上で演算加速装置を効率的に利用するためのアルゴリズムとそれを用いたアプリケーション性能

Algorithms to use accelerators effectively with FDPS and performance of applications

*行方 大輔[1]、岩澤 全規[1]、似鳥 啓吾[1]、野村 昴太郎[1]、Long Wang[3,1]、坪内 美幸[1]、牧野 淳一郎[2,1,4]
*Daisuke Namekata[1], Masaki Iwasawa[1], Keigo Nitadori[1], Kentaro Nomura[1], Long Wang[3,1], Miyuki Tsubouchi[1], Junichiro Makino[2,1,4]

1. 理化学研究所 計算科学研究センター、2. 神戸大学 大学院理学研究科 惑星学専攻、3. Helmholtz Institut fu ¨r Strahlen und Kernphysik、4. 東京工業大学 地球生命研究所
1. RIKEN Center for Computational Science, 2. Department of Planetology, Graduate School of Science, Kobe University, 3. Helmholtz Institut fur Strahlen und Kernphysik, 4. Earth-Life Science Institute, Tokyo Institute of Technology

粒子法に基づく数値シミュレーションは、惑星と惑星系の形成・進化を理解する上で重要な役割を果たしてきた。コンピュータの計算能力の向上に伴い、現実をより詳細に理解するため、より大規模なシミュレーションや様々な物理過程を考慮したシミュレーションが行われるようになってきている。

しかしながら、現代のHPCシステム上で効率的に動作するシミュレーションコードを開発することはますます困難になっている。この主な理由は、現代のHPCプラットフォームが非常に複雑であり、このようなプラットフォームを効率的に使用できる複雑なプログラムの開発に多くの努力を費やす必要があるためである。典型的なHPCシステムは、多数の計算ノードを限られた通信帯域幅のネットワークで接続したクラスターである。最も大きなシステムでは、計算ノード数は約10万にも達する。このような大規模システム上でシミュレーションコードを効率的に動作させるためには、極めて良いロードバランスを示し、かつ、必要最小限の通信しか要求しないようなコードの開発が必要となる。近年、大規模HPCシステムのアーキテクチャは、経済的な理由から、同種のマルチコアプロセッサで構成されるシステムから演算加速装置ベースのシステム、或いは、異種のマルチコアプロセッサで構成されるシステムへと移行しつつある。このことは、コード開発をより困難にさせる要因になっている。理由の1つは、多くのアプリケーションにとって、CPUと演算加速装置間の通信帯域幅がボトルネックになることである。コード開発はこのことを考慮して行う必要がある。もうひとつの理由は、CPUと演算加速装置は通常別々のメモリ空間を持っており、より複雑なプログラミングが要求され、また既存のプログラムも利用できないことである。このように、コード開発に必要な時間がますます増えており、研究の停滞を招いている。

このような状況を改善するため、我々はFDPS (Framework for Developing Particle Simulators)と呼ばれるフレームワークを開発してきた。FDPSは、研究者が容易に独自の並列粒子シミュレーションコードを開発することを可能にするものである。 FDPSの基本的な考え方は、研究者が独自の粒子データ構造と粒子間相互作用を定義することができるように、粒子シミュレーション用の並列アルゴリズムの(高度に最適化された)実装を「一般的な」形で提供することである。ユーザーから提供される粒子データ型と相互作用関数とともにコンパイルされたFDPSは、並列計算に必要とされるすべての機能を提供する。これら機能を使うことで、研究者はあたかもラップトップコンピュータで動くような単純な非並列プログラムを書くように並列プログラムを書くことができる。FDPSは、京コンピュータやx86プロセッサを搭載したCrayシステムなどのような、同種のマルチコアプロセッサで構成されるシステムで高い性能を示すことがわかっている。バージョン2において、我々は演算加速装置を使う相互作用関数を書くことによって、FDPSを演算加速装置と組み合わせて使えるように拡張した。しかしながら、効率は、CPUと演算加速装置間の通信のレイテンシと帯域幅、及び、相互

©2019. Japan Geoscience Union. All Right Reserved.

作用関数と演算加速装置の並列度との間の不一致によって制限され、かなり不十分なものであった。

そこで、我々は、FDPSからGPGPUs (General-purpose computing on graphics processing units)のような演算加速装置をより効率的に使用できるようにするため、ユーザから提供される相互作用関数のインターフェース仕様を見直した。また、CPUと演算加速装置間の通信量、及び、CPU側でのタスクを減らす新しい技術の実装を行った。これらの効果を検証するため、NVIDIA Volta GPGPUを搭載したシステム上で、FDPSを使ったN体シミュレーションコードを作成し、性能測定を実施した。得られた性能は理論ピーク性能の約27%であった。

キーワード：計算科学、ソフトウェア開発、演算加速装置
Keywords: Computational Science, Software development, Accelerators

©2019. Japan Geoscience Union. All Right Reserved.