マルチノード・マルチGPU上のコレスキー分解に対するデータドリブン型アルゴリズム手法

辻田 裕紀, 遠藤 敏夫
東京工業大学

1 はじめに

近年、アクセラレータによるヘテロジーニアスなスーパーコンピュータを用いた大規模科学計算が注目を集めている。しかし従来の静的なジョブ実行やメモリマネージメントでは、データ転送コストが大きく、計算資源の利用率が低いため、ヘテロジーニアスな計算資源を利用するためには十分ではない。本研究では最適化のターゲットアプリケーションとして重要な線形計算カーネルのひとつであるコレスキー分解をとりあげ、計算資源の利用率を高め、データ転送量を減らすための、スケーラブルなデータドリブン型スケジューリング手法とヘテロジーニアスなメモリマネージメント手法を提案する。NVIDIA GPUによるヘテロジーニアスなスーパーコンピュータであるTSUBAME2.5上での実験を通して提案するタスクスケジューリングと再利用性を考慮したデータ入れ替え戦略による効果を示す。

2 タスクスケジューリングとデータ移動最適化手法の提案

本プログラムではまず入力データをタイルという単位に分割し、ブロックコレスキー分解の各ルーチンをタスクとして、タスクをタイル単位で実行するように実装している。タイルには次の3つの実行状態を持つ変数と実行ステップを表す変数ももっており、これらを見ることで実行すべきタスクを判断することができる。並列実行時、データはブロックサイクリック分割により各MPIプロセスに割り当てられる。各プロセスは割り当てられたタイルのタスクのみを実行するようになっている。本プログラムでは、計算とPCIeおよびMPI通信のオーバーラップを行う為に各MPIプロセスは複数のworkerスレッドとignitionスレッドで構成されている。また各プロセスは実行可能なタスクを管理するために、すべてのスレッドで共有されているタスクキューを持っている。workerスレッドは専らタスクの実行を行い、ignitionスレッドは主に他のノードからのタスクの終了通知を確認している。実行可能なタスクはキューにしまうことで管理している。今回、次のタスクを選ぶ戦略としてFIFO、RAND、GREED、

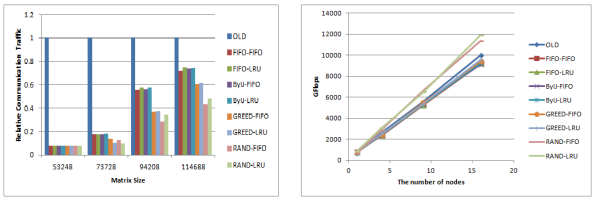

図 1: (a)PCIの相対通信量(4ノード) (b)Weak Scaling

ByIJの4つを用意した。本プログラムでは、GPUメモリに紐付されているタイルはリストにより管理されている。GPUメモリのデータを入れ替える時、このリストを使ってGPUから掃き出されるタイルを選択している。今回、掃き出されるタイルの選び方についてFIFOとLRUの2つを用意した。

3 実験と評価

SDPARA GPU版に対して、データドリブン型アルゴリズムを導入にし、どのように性能が変化したか実験を行った。以降、データドリブン型のアルゴリズムを導入したものをNEW、導入していないものをOLDと呼ぶ。なお今回用いたOLDの実装はTSUBAME2.5の4080GPUを用いて1.7PFlopsを記録したものである[1]。今回、PCIe通信をデバイスメモリより小さいサイズの時、90%以上削減することができた。デバイスメモリを超えるサイズの時でも48から50%の削減を達成できた。また実装はスケーラブルなものであり、16ノード16GPUのとき11.9TFlopsを達成した。

参考文献

[1] Katsuki Fujisawa, Toshio Endo, Yuichiro Yasui, Hitoshi Sato, Naoki Matsuzawa, Satoshi Matsuoka, and Hayato Waki. Peta-scale general solver for semidefinite programming problems with over two million constraints. In *In Proceedings of the International Conference on Parallel and Distributed Processing Symposium 2014 (IPDPS2014)*, p. 10pages, 2014.