高効率な資源利用を実現するGPUプログラム自動チューニング手法

小野 和馬*，古橋 一輝，津邑 公暁（名古屋工業大学）

Automatic Code-tuning for efficiently utilizing GPU Resources

Kazuma Ono, Kazuki Furuhashi, Tomoaki Tsumura (Nagoya Institute of Technology)

1. はじめに

画像処理専用のプロセッサである GPU の演算能力を汎用的に用いる GPGPU が発展してきており，GPU プログラミングの重要性が増してきている．現在 GPU 向けの開発環境として最も普及しているのが，NVIDIA 社が提供している並列計算アーキテクチャモデル **CUDA** の開発環境である．しかし，CUDA を用いて GPU の演算能力を十分引き出すためには，GPU のハードウェアリソースの特徴を熟知し，これらを適切に用いるようにプログラムを記述する必要がある．そこで本稿では，GPU 上の演算コアの利用率向上と，適切なメモリへのデータ配置という 2 つの観点から自動的にプログラムにチューニングを施すことで，GPU プログラミングの難易度を緩和することを目指す．

2. GPU のハードウェアリソース

GPU は多数の演算コアを搭載し，これらに処理単位を割り当てて同時に命令を実行することで，処理の高速化を図っている．CUDA では GPU に実行させる処理を**カーネル関数**と定義している．カーネル関数の実行には，GPU 上での処理単位を割り当て，使用するデータを GPU 上のメモリに配置する必要がある．まず処理単位の割り当てに関して，CUDA では最小の処理単位を **Thread** と定義しており，Thread は階層的に管理することができる．Thread の集合は **Block** と呼ばれ，カーネル関数を実行するためには，Block 内の Thread 数と Block 数という 2 つのパラメータから成る**実行構成**を設定する必要がある．この実行構成やカーネル関数が利用する GPU の共有リソース量に左右される，GPU の演算コアの使用率を示す指標を**占有率**と呼び，この占有率ができるだけ大きな値となることが望ましい．しかし，占有率の算出には複雑な計算が必要であり，一般のプログラマが占有率を考慮して CUDA プログラムをチューニングすることは困難である．さらに，カーネル関数の実行に必要なデータの配置先をプログラマが選択する必要があることも，CUDA プログラムのチューニングをより難しくしている．なぜなら，GPU 内には様々なメモリやキャッシュが搭載されており，それぞれアクセス速度や容量，Read・Write アクセスの可否が異なるため，プログラマはデータの特徴を考慮して配置先のメモリを適切に選択しなくてはならないからである．

3. 自動チューニング手法

本稿では，2 つのチューニング手法を提案する．1 つ目は，GPU の共有リソース量や GPU のハードウェア構成を考慮することで，カーネル関数実行時に演算コアの占有率が最大になるよう，実行構成を自動的に設定する手法である．なお，占有率は NVIDIA 社が公開している算出法[1]を用いて算出する．2 つ目は，プログラム中で使用されるデータの特徴量を算出し，特徴量を機械学習により生成した分類器に入力することで，データの適切な配置先メモリを選択する手法である．この手法は，分類器を再生成することで，GPU のハードウェアリソースの変更にも柔軟に対応できる．

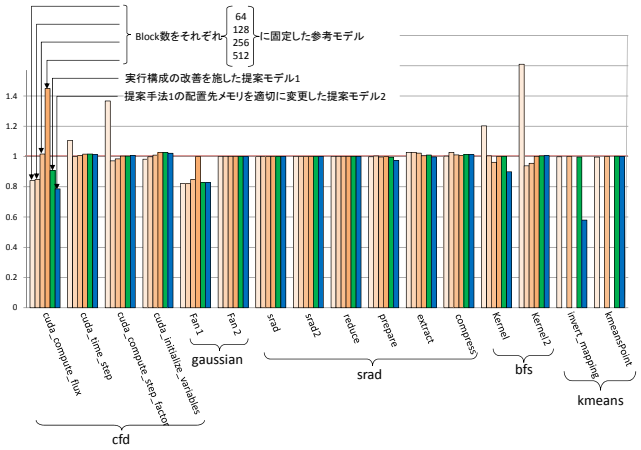

Fig.1　Execution Time

4. 評価

提案手法の有用性を確かめるため，Rodinia Benchmark suite から，実行構成を変更可能な 6 つのベンチマークを用いて評価した．評価結果を Fig.1 に示す．結果はそれぞれ，Block 数を固定した参考モデル，実行構成を自動でチューニングする提案モデル 1，実行構成に加えデータの配置先メモリも適切に選択する提案モデル 2 を用いた場合の，各ベンチマークにおけるカーネル関数ごとの実行時間を表している．また，結果はデフォルトで指定されている実行構成でベンチマークを実行した際の実行時間を 1 として正規化している．この結果から，実行構成の自動チューニング手法を用いることで多くのカーネル関数を高速化することが可能であり，データを適切なメモリへ自動配置する手法を用いることで今回評価した全てのカーネル関数を高速化することが可能であることが確認できた．一部のカーネル関数に対して実行構成の自動チューニング手法が高速化を達成できなかった理由としては，算出した占有率と演算コアの実際の使用率に差が生じている可能性が考えられる．

文　献

(1) NVIDIA Corp.: CUDA Warps and Occupancy, *GPU Computing Webinar 2011*(2011)