情報マネジメント

プログラムの CUDA 化と最適化

コンピュータアーキテクチャ研究室　大野班

420837 野村文誉

2023 年 7 月 27 日

プログラム実行手順
gcc -o kadai kadai.c コマンドで kadai.c ファイルをコンパイル
nvcc -o kadaicu kadai.cu コマンドで kadai.cu をファイルをコンパイル
nvcc -o saiteki saitekikadai.cu コマンドで saitekikadai.cu ファイルをコンパイル
上記のコマンドを実行すると
kadai kadaicu saiteki　という実行ファイルができる
その後に実行したいファイルを ./kadai　などのコマンドで実行できる
また実行時間を測る際には　time ./kadai などのコマンドで実行できる

CUDA 化前のプログラム概要
「加藤誠也、須田礼仁、玉田嘉紀.GPU におけるダイバージェンス削減による高速化手法.情報処理学会研究報告.2012.Vol.2012-HPC-134.No5」を参考にした「データ長によって処理数が多くなる」プログラムを作成した。処理は５８個として、処理 A～z（ASCII コードにより名前付けした）とした。処理ごとのデータの長さは 2~10000 の範囲でランダムに設定した。

CUDA 化後プログラム概要
並列化によって、処理を順番にそれぞれのスレッドで処理させることで性能を向上させた。

最適化プログラム概要
処理の数を判定し、処理をリスケジュールし、スレッドへの振り分けを 1 番処理が短いものと 1 番処理が長いものをセットにして各スレットに分けていくことで、プログラムの最適化を行った。

参考文献

加藤誠也、須田礼仁、玉田嘉紀.GPU におけるダイバージェンス削減による高速化手法.情報処理学会研究報告.2012.Vol.2012-HPC-134.No5.2023/07/27 閲覧

測定した実行時間と速度向上率

	CUDA 化前プログラム	CUDA 化後プログラム	最適化プログラム
実行時間	1112.975u	1.530u	1.451u
速度向上率		727.4%	105.4%

考察

データ構造とは違いここに独立したプログラムならすべてのプログラムに適用可能であると考えたので、この最適化を検討し行った、結果としては思ったよりも速度向上がみられなかった、原因としてはリスケジュールをするプログラムの計算量が多くなってしまい計算に時間がかかってしまったことが考えられる、これはより簡略したリスケジュールプログラムを考えることで解決する。また他の原因として私の作ったプログラムは処理数を等分しているので処理ごとの計算量に偏りがあった場合に効率的にスケジューリングできない可能性があるので、処理を等分するのではなくデータ長によって振り分けを変え最適化することで解決することができると考えられる。

参考文献

加藤誠也、須田礼仁、玉田嘉紀.GPU におけるダイバージェンス削減による高速化手法.情報処理学会研究報告.2012.Vol.2012-HPC-134.No5.2023/07/27 閲覧