CPUとGPUを用いる高速数値計算ライブラリ

ー最新PCが持つ真の性能を活用するためのソフトウェアー

1. 背景

近年、高度な画像処理の要求に伴いGPU(Graphics Processing Unit)の性能が著しく向上している。現在の GPU は、処理速度が向上しているだけではなく、複雑な画像処理アルゴリズムを実装できるだけの演算の自由度を備えている。また、並列処理やベクトル処理に適したハードウェア構成をしており、CPU(Central Processing Unit)とは異なるタイプの性能向上が進んでいる。今日では、GPU を画像処理だけではなく汎用的な用途に利用する研究が行われており、GPGPU(General-Purpose computation on GPUs)と呼ばれている。

GPUを用いた数値計算は、近年活発に研究が進められているテーマである。行列積やFFTなど、基本的な数値計算問題をGPUで解いた例がいくつかの論文やwebページなどで散見されている。結果として、CPU よりも優れた性能を得たと結論づけているものもある。

真に高い演算性能を得るためには、CPU と GPU 両方の演算性能を得る必要がある。しかしながらこれまでの GPU を用いた数値計算では、CPU を用いて数値計算を行った際の性能と、GPUを用いた数値計算を行った際の性能との比較に主眼が置かれてきた。また、GPGPUプログラミングにはGPUを扱うためのプログラミングスキルが必要であり、他の分野のプログラマにとってGPUを利用することは容易ではない。現在では複数のCPU(複数コアのCPU)を搭載したPC環境や複数の GPU を搭載した PC 環境、複数台の PC を用いた計算環境が容易に構築可能となっているが、これらの先進的な計算環境の性能を引き出すのは更に困難である。

2. 目的

本テーマの目的は、CPU と GPU を同時に用いて高い演算性能を得ることのできる数値計算ライブラリを作成することである。複数のプロセッサや複数の PC を用いる複雑な計算環境への適用も考慮し、アプリケーションプログラマが先進的な計算環境を容易に利用できるようにする。また、プログラムの作成技術を公開し、先進的な計算環境の利用を促進する。

3. 開発の内容

対象とする実行環境の異なる2つの行列積和計算ライブラリを作成した。1つは単数の CPU と単数の GPU を搭載した PC 向けのライブラリ「gpupcgemm_library」であり、もう1つは複数の CPU と複数の GPU を搭載した PC 向けのライブラリ「gpupcgemm_mgc_library」である。なお、gpupcgemm_library を複数の CPU や複数の GPU を搭載した環境で利用することは不可能ではないが、その場合は1つのCPUと1つのGPUのみが計算に用いられる。

いずれのライブラリも GPU の制御に OpenGL を利用しているため、OpenGL および OpenGL の拡張機能（GLEW）が動作する環境が必要である。更に、利用する GPU は ShaderModel3.0 に対応している必要がある。ただし、全ての計算を CPU のみで行わせることも可能であり、この場合は GPU の制限はない。また複数台の PC を用いる計算環境（以下、

複数 PC 環境と呼ぶ)での利用を考慮して作られているため、既存の複数 PC 環境向けプログラミング手法(unix_socket、MPI など)と併用することができる。対応 OS はライブラリごとに差がある。gpupcgemm_library は Windows および Linux での動作を確認しており、FreeBSD や MacOS などでも利用できる可能性が高い。一方で gpupcgemm_mgc_library は Linux でのみ動作を確認しており、Windows での動作は不可能である。これは、複数の GPU を扱う実装上の都合によるものである。

以下、各ライブラリの動作概要について図を用いて説明する。

gpupcgemm_library は行列積和計算を行う関数及び補助的な終了処理を行う関数の2種類の関数を提供する。後者については動的なメモリの開放や実行ログの出力といった処理を行うものであるため、詳細な説明は割愛する。なお、この関数を実行しないことによって致命的な問題が生じることはない。

gpupcgemm_library が提供する行列積和計算関数の流れを図 1 に示す。関数インターフェイスは、線形計算ライブラリ BLAS の行列積和計算関数(GEMM)インターフェイスを採用している。これは、既存の CPU 向けアプリケーションへの適用を容易にするためである。

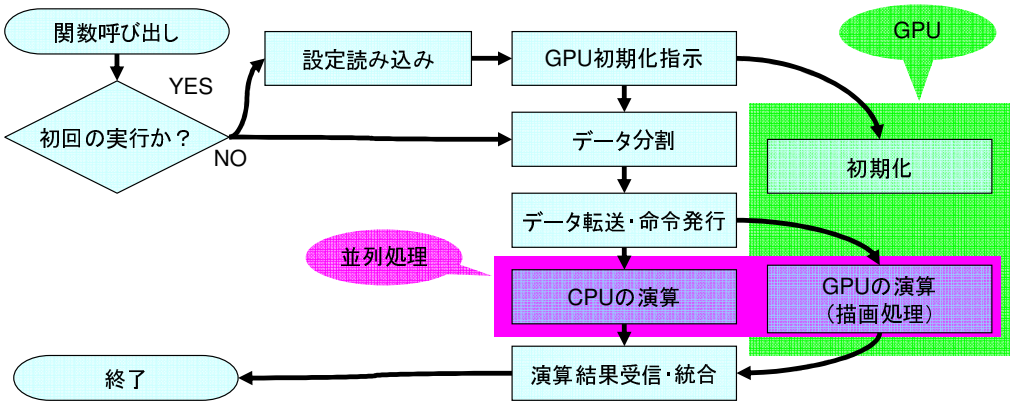

図 1 gpupcgemm_library における行列積和関数の処理の流れ

CPU と GPU で並列に計算を行う機構の概要は以下の通りである。OpenGL の描画命令によって GPU が演算をする際、CPU への負荷は描画命令の発行のみであり、また描画命令は描画処理が完全に終了する前に関数が返る実装になっている。そのため描画命令に続いて CPU の処理を記述することで、マルチスレッドやマルチプロセスといった並列化プログラミングの技法を用いずに CPU と GPU による並列処理を実装することができる。ただし、GPU の動作を制御するための OpenGL 関数呼び出しや CPU-GPU 間のデータ転送、初回実行時に必要な GPU の初期化処理が並列処理のオーバヘッドとなる。これらの処理は、対象問題が著しく小さいときには実行時間に占める割合が大きくなり、速度低下を招く。そのため gpupcgemm_library では、対象問題の問題サイズが規定のサイズより小さい場合には CPU のみで演算を行う機構を備えている。この際の問題サイズの閾値や、その他いくつかのチューニングパラメタは、外部の設定ファイルから読み込んで利用する。これらの基本的な実装方法やオーバヘッドの問題は、gpupcgemm_mgc_library でも同様である。

gpupcgemm_mgc_library についても、提供する関数は gpupcgemm_library と同様である。行列積和関数の処理の流れを図 2 に示す。本関数は、GPU（ビデオカード）1つにつき1つの子プロセスを生成し、親プロセスが子プロセスの管理およびマルチスレッドを用いた CPU による演算の管理を行う構造となっている。プロセス間のデータ共有にはメモリマップファイルを用いた共有メモリ（shm_open/shm_unlink と mmap を用いる、Posix 共有メモリ）を利用し、プロセス間の同期及び問題設定の通知には unix_socket を利用した。gpupcgemm_library と比べて実行環境の制限が厳しいのは、複数の GPU を制御するために GLUT（The OpenGL Utility Toolkit）の X Window system 向け実装依存な機能を利用しているためである。

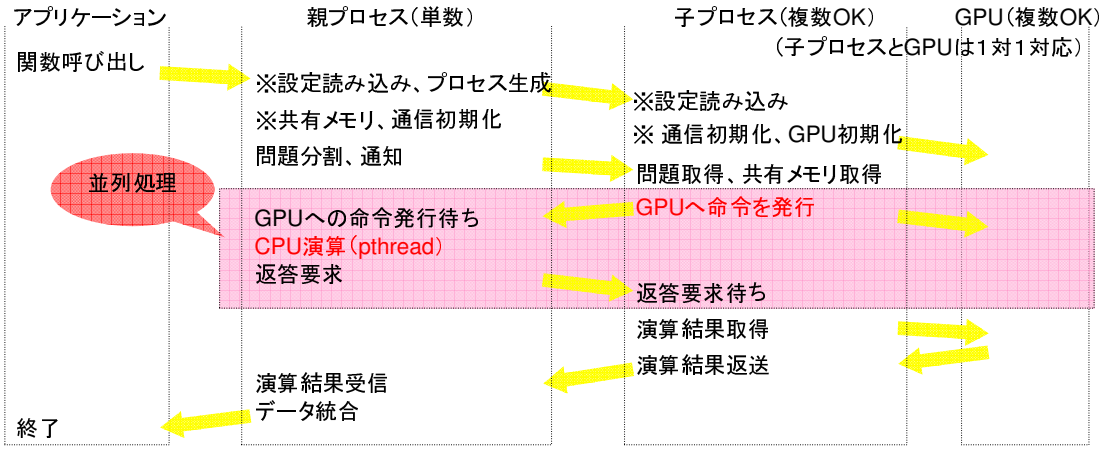

図 2 gpupcgemm_mgc_library における行列積和関数の処理の流れ

いずれのライブラリも、ライブラリ内で実行している CPU による行列積和計算には ATLAS の行列積和関数を用いている。ATLAS は BLAS のインターフェイスを持つ CPU 向けの高速な計算ライブラリの1つである。

BLAS の行列積和計算関数は、演算対象のデータ型に合わせて複数の関数が存在する。主な対象データ型としては単精度浮動小数や倍精度浮動小数がある。GPU は単精度浮動小数の演算に対応しているが、倍精度浮動小数の演算に対応していない。そのため gpupcgemm_library および gpupcgemm_mgc_library が正式に対応しているのは単精度浮動小数向けの計算のみである。ただし、CPU の担当する演算のみを倍精度で実行し、GPU の担当する演算を単精度で実行する、倍精度の行列積和関数インターフェイスを持つ関数も実装した。現時点では精度評価の際や、既存の倍精度計算を行うプログラムに実験的に適用する際に用いることができる程度である。将来的に倍精度演算に対応した GPU がリリースされた際に有用となることが考えられる。

本ライブラリを用いた際の性能については、CPU と GPU の性能バランスによって差が生じる。CPU と GPU それぞれ単体での実行時間の差が小さい場合に、CPU と GPU での並列処理によって実行時間を半分近くまで短縮させた例や、更に 2CPU+2GPU や複数台の PC を利用することで 3 割程度まで短縮した例がある。一方で、仮に CPU と GPU それぞれ単体での実行時間に 100 倍の差があれば、ライブラリを利用してもたいした効果が期待できない。また、複数の

CPU と複数の GPU を搭載した複数台の PC では確かに実行時間は短縮できるが、各プロセッサの担当する問題サイズが小さくなりすぎてしまい、並列処理のオーバヘッドが目立ちやすくなるという問題がある。こうした問題点は、本ライブラリを大規模な計算問題の中で利用する場合や、本プロジェクトの根幹をなす技術である CPU と GPU による並列処理を他の計算問題やアプリケーションへ適用していく際に考慮することで、より良い性能を得られるようになると考えられる。

4. 従来の技術（または機能）との相違

本プロジェクトの開発成果であるライブラリソフトウェアおよび得られたプログラミング技術の特徴は、実行環境の持つ演算性能をこれまで以上に有効に引き出すことが出来る点である。本成果を用いることで、CPU 単体よりも GPU 単体よりも高速な演算を行うことができ、また複数の CPU と複数の GPU を搭載した PC を複数台用いた計算環境のような先進的な計算環境を有効に活用することができる。従来このような先進的な計算環境を利用するためには複雑なプログラミング技術が必要とされてきたが、数値計算の基本問題をライブラリ形式で提供することにより、アプリケーションプログラマの労力削減が期待できる。

5. 期待される効果

新しい計算環境の登場はプログラミングコストの上昇という問題を抱える可能性がある。近年は複数の CPU を搭載した環境や GPU を搭載した計算環境の普及が始まっているが、まさにこの問題を抱えていると言える。本テーマで作成したライブラリソフトウェアは、こうした計算環境を容易かつ有効に活用できる１つの方法を示すものである。ゆえに、本テーマの成果によって新しい計算環境の利用が促進されることが期待できる。また本テーマを踏まえたうえで、異なるより良い利用方法が提案される可能性も考えられるが、これも本テーマに期待される効果の１つであると言える。これらの効果は特定の分野に限られたものではなく、一般的な PC コンピューティング全般に言えることである。

6. 普及（または活用）の見通し

作成したライブラリは既存の CPU 向け数値計算ライブラリのインターフェイスを利用しているため、既存の多くのアプリケーションに対して本プロジェクトの成果を適用することが出来る。

一方、複数の CPU や複数の GPU および複数台の PC を用いるプログラミング技術が今後様々な分野で活用されるようになる可能性を持つこと、それが本プロジェクトに期待される効果の１つであることは、前章に記述したとおりである。

7. 開発者名（所属）

大島聡史（電気通信大学　大学院情報システム学研究科　博士後期課程）

（参考）開発者 URL

http://aaa.jspeed.jp/~ohshima/