平成 24 年度 9 月 25 日

仮想 GPU による動的負荷分散ツールの開発

電気通信大学　情報工学科　成見研究室

0911096　増澤　沙紀

1.　研究背景

近年、高度な並列計算を得意とする GPU を、グラフィックス描画という本来の目的ではなく、より一般的な計算用途に使用しようという GPGPU の利用が広がりを見せている。

また、ネットワークにより接続された PC の GPU をあたかもローカルに装着されているかのように利用することができる仮想 GPU という技術により、数十台の GPU を 1 台の PC で使用することができるようになった。このため、GPGPU と GPU の仮想化を利用することで、短い時間の中で膨大な計算を行わなければならないようなシミュレーションにおいてもリアルタイムで結果を表示させることが可能となった。

しかし、複数の GPU で計算を行う場合、各 GPU での計算量が均等ではないことにより、時間がそれほど短くならないことがある。

また、仮想 GPU では 1 つの GPU に複数ユーザーからのアクセスが可能であため、1 つの GPU で複数の計算プロセスが実行されることにより負荷が均一にならない場合も考えられる。

そこで本研究では、使用するユーザーが意識をすることなく、システムが各 GPU での負荷を計測し、その負荷に合わせて自動的に計算を分散するようなツールの開発を目的とする。

2.　既存研究

既存研究として、MPI による動的負荷分散システムの開発がある[1]。ハードウェアのリソースを仮想化し、複数の OS を動作させることが可能な仮想計算機 Xen を使用して開発をされている。この研究では動的負荷分散を行うことができるが、マッピングの際に、仮想マシン全体を他の PC に移動を行うため、通信量が多くなってしまう。また、MPI プログラムを用いて複数ノードの計算を行う必要があり、プログラムの手間がかかる。

仮想 GPU を用いた計算では、シリアルなプログラムで並列計算を行うことが可能であるが、自動で負荷分散を行うことは不可能である。本研究では仮想 GPU での自動的負荷分散を行うツールを開発する。このツールにより、簡単に効率の良い並列計算が今後可能となると考えられる。

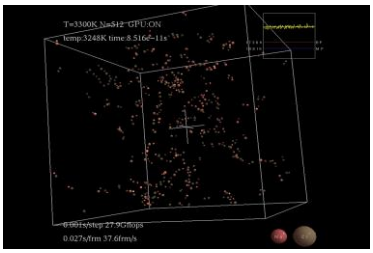


図 1　分子シミュレーション

3.　研究概要

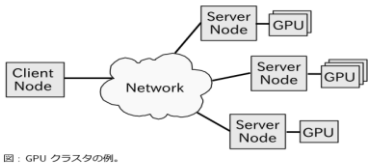
本研究では分子シミュレーションを活用した動的負荷分散ツールの開発を行う。

目標とする仕様は以下の通りである。

i. 仮想 GPU の利用
   GPU を仮想化することで複数台の GPU の利用を可能とする。
ii. 分子シミュレーションの空間分割
   各 GPU の負荷を変えるため、使用する GPU の台数に合わせて空間分割を行い、それぞれの区画に別々の GPU を割り当て計算させる。
iii. 各 GPU における負荷の計測
   使用している全 GPU に簡単な計算をさせ、その計算時間により、各 GPU の負荷を測定する。
iv. 負荷の分散
   各 GPU の負荷に応じて、負荷の高い GPU での計算の一部を負荷の低い GPU に移動をさせる。

4.　使用するソフトウェア

本研究では、GPU仮想化ソフトウェアとして、K&F Computing Researchの川井らが作成したDS-CUDAを用る[2]。



図：GPU クラスタの例。

図 2　DS-CUDAの構造

DS-CUDA とは、ネットワークを介することで、他の PC の GPU をあたかもローカルに接続された GPU かのようにみせることができる。DS-CUDA の構造は図 2 のようになる。DS-CUDA 用のサンプルプログラムである銀河衝突シミュレーションを用いて、GPU の数による計算速度の変化を調べると図 3 のようになる。ネットワークで接続されているため、通信にやや時間がかかるが、8 台の GPU を使用すると計算時間も 8 倍となることがわかる。

5.　進捗状況

現在の進捗状況は以下の通りである。

i. DS-CUDA のテスト。

ii. 2 台の GPU を用いたプログラムの作成。

- 粒子分割
- x 軸で空間分割

5.1　2 台の GPU を用いたときの計算速度の変化

GPU を 2 台使用したときの分子シミュレーションの計算速度を測定した。粒子分割は図 4、空間分割は図 5 である。

粒子分割では、複数台 GPU を使用するとホストからデバイスへのデータの転送をするため、その分の時間が少しかかる。そのため、少ない粒子数の計算では 1 台の GPU での計算の方が速いが、粒子数が多くなるにつれ複数台の GPU で計算を行った方が速くなる。

一方、空間分割では、粒子分割に比べ各 GPU の計算量が均等ではないため若干の速度低下が考えられる。しかし、図 5 では 5500 粒子数あたりから速度が急激に低下してしまっている。原因はまだはっきりとはしていないが、おそらく空間の分割に時間がかかってしまっているのではないかと考

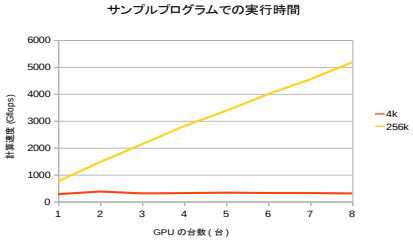


図 3　DS-CUDA の構造

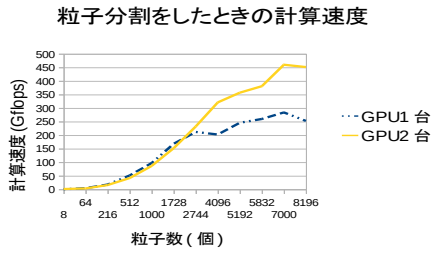


図 4　粒子分割をしたときの計算速度

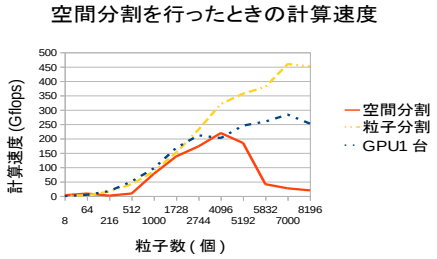


図 5　空間分割を行ったときの計算速度

えられる。

6.　今後の課題

i. 空間分割プログラムの改善

本研究では、より大規模な計算を行うことが前提となっているので、空間分割の処理時間を高速化する必要がある。また、現在は x 軸での分割となっているが、最低でも x,y の 2 次元での分割を目指す。

ii. DS-CUDA の対応

Make ファイルを書き換えるだけで対応できる。

iii. 各 GPU の負荷の測定

各 GPU に簡単な計算を流して、実行時間を調べることで各 GPU の負荷を測定しようと考えている。

iv. 動的分散を行う

iii．で測定した負荷によって、負荷の高い GPU での計算を負荷の低い GPU に自動的に移動させる。

7.　参考文献

[1] 立薗真樹　中田秀基　松岡聡『仮想計算機を用いて負荷分散を行う MPI 実行環境』

[2] Atsushi Kawai, Kenji Yasuoka, Kazuyuki Yoshikawa, and Tetsu Narumi,
Distributed-Shared CUDA：Virtualization of Large-Scale GPU Systems for
Programmability and Reliability, The Fourth International Conference on
Future Computational Technologies and Applications, Nice, France, 2012.