高可用性のための仮想 GPU レプリケーション

関野泰宏[†]　鈴木勇介[†]　河野健二[†]

1. はじめに

近年の GPU 仮想化技術[1)2)] の登場により，クラウド環境での GPU の利用が現実的になってきている．GPU の仮想化技術とは，一台の物理 GPU を複数台の仮想 GPU として管理できる技術である．これにより，複数ユーザが GPU を共有して使用することが可能になる．より安価に GPU を利用できるようになるため，より広く一般のアプリケーションが GPU の計算資源を利用できるようになる．

クラウド環境で広く GPU が利用されるに伴い，高い可用性を要求するアプリケーションによる GPU の利用も普及していくと考えられる．実際に，Web サーバ[3)] や SSL リバースプロキシ[4)]，キャッシュサーバ[5)] といった用途で，GPU を利用する研究が行なわれている．これら Web サービスの基幹となるアプリケーションは，短時間でも停止すると大きな損害を出すことが知られている．例えば，Amazon では 1 分間に平均約 700 万円の売り上げがあり[6)]，Amazon が停止することでこれらの商取引の機会は失われてしまう．

現状では，GPU を可用化するためには，GPU のロックステップ実行を可能にする特殊なハードウェアが必要になるために，コストがかかる[7)]．汎用 GPU に対して高可用性を提供できれば，GPU を使用するシステムで安価に高可用を達成できる．

そこで本発表では，GPU の複製を安価に作成する方法を提案する．仮想マシンのメモリイメージの複製技術と組み合わせることで，GPU を使用している仮想マシン全体の可用性を向上させることが可能となる．

2. GPU

GPU は，CPU から書き込まれるコマンドをベースに実行を進めていく．はじめに，GPU で動作するプログラムとデータをシステムメモリ上に用意する．次に，DMA の開始を指示するコマンドを送り，GPU 上で実行させるプログラムとデータをグラフィックボード上のメモリ (VRAM) に送る．その後，計算の実行を指示するコマンドを送ることで，GPU 上でプログラムが実行され，計算結果が VRAM に書き込まれる．最後に，計算結果をシステムメモリに書き戻すためのコマンドを送り，計算結果を得ることができる．

そのため，GPU のコンテキストは，GPU に書き込まれるコマンドと送られるデータとなる．これらと同一のものを別の GPU に書き込むことで，GPU のコンテキストが複製できる．

ここで，GPU にデータを送る経路は二通り存在することに注意が必要となる．一つは，システムコールによって GPU ドライバを呼び出して DMA を発行する場合であり，もう一つは，ユーザスペースに VRAM をマップして MMIO で書き込む場合である．コンテキストを複製する際には，いずれの経路から書き込まれるデータも複製する必要がある．

また，クラウド環境での利用を想定して仮想 GPU の複製を考えるが，GPU の仮想化にはいくつかの手段が存在する．仮想化に利用するレイヤで大別すると，ライブラリコールのレイヤで仮想化する API リモーティング，システムコールのレイヤで仮想化する準仮想化，GPU ドライバの下のレイヤで仮想化する完全仮想化の 3 つが挙げられる．柔軟性や効率性を考慮し，本研究では，GPU 仮想化の手法として準仮想化を対象に議論をすすめる．

3. 提　　案

GPU のコンテキストを複製するための手法として，仮想 GPU に対する操作を複製するという手法を提案する．図 1 に，提案システムの全体図を示す．仮想 GPU に対する操作を，ハイパーバイザのレイヤで待機系のマシンに送り，障害発生時には自動で待機系のマシンに切り替えるというものである．

複製が必要な操作には，システムコールとユーザスペース MMIO の 2 つがある．システムコールに関しては，準仮想化ドライバを呼び出したときのパラメータを複製する．このとき，GPU のメモリを管理して

† 慶應義塾大学
Keio University

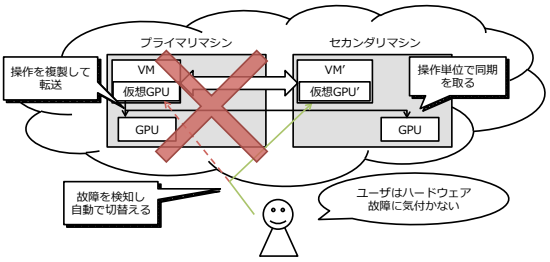

図 1 提案システム

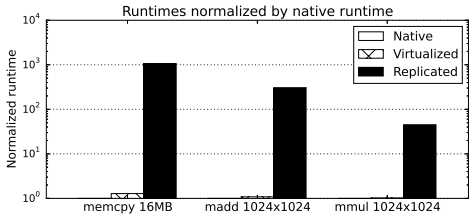

図 2 実行時間の比較

いる オブジェクトには固有のハンドラが割り当てられているため，このハンドラの対応関係を取る必要がある．

ユーザスペースの MMIO に関しては，ページフォルトを利用して補足する．具体的には，次のアルゴリズムで MMIO のトレースを行う．まず，GPU のメモリがマップされた範囲のアドレスのページをページテーブルから外す．すると，MMIO アクセスにより，ページフォルトが発生する．その時のカーネルスタック上の命令ポインタを参照し，命令をデコードしてアクセス幅と値を取り出す．カーネルスタック上のフラグレジスタを書き換え，割り込み復帰後にシングルステップ実行モードに入れるようにする．フォルトしたページを挿入する．プロセスを再開し，1 命令実行後にトラップされるので，ページを再び外す．以上のアルゴリズムによって MMIO アクセスをトレースし，複製先の GPU メモリの対応するアドレスに同じ値を書き込むことで MMIO アクセスの複製を行うことができる．

4. 予 備 実 験

実際に一台の物理マシン内で仮想 GPU を複製するシステムを実装したところ，上記の手法で仮想 GPU が複製されている様子が読み取れた．図 2 に示すように，複製を作るには高いオーバヘッドが生じる．さらに調査したところ，MMIO の複製に高いオーバヘッドがかかることがわかったが，これは MMIO の命令ごとに複製を実行しているためであり，ページ単位での複製にすることで最適化を図ることが可能と考えられる．

5. ま と め

GPU の使途の拡大に伴い，GPU を使用しているシステムの可用性の向上が望まれている．しかし，現状では GPU のロックステップ実行を可能とするハードウェアが必要となり，コストが高い．

そこで本研究では，安価に仮想 GPU を複製するための手法を提案した．実際に一台のマシン内でプロトタイプ実装したところ，MMIO の複製に高いオーバヘッドがかかることが判明した．

参 考 文 献

1) Suzuki, Y., Kato, S., Yamada, H. and Kono, K.: GPUvm: GPU Virtualization at the Hypervisor, *IEEE Transactions on Computers*, Vol. 65, No. 9, pp. 2752–2766 (2016).
2) Tian, K., Dong, Y. and Cowperthwaite, D.: A Full GPU Virtualization Solution with Mediated Pass-through, *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, USENIX ATC'14, Berkeley, CA, USA, USENIX Association, pp. 121–132 (2014).
3) Agrawal, S. R., Pistol, V., Pang, J., Tran, J., Tarjan, D. and Lebeck, A. R.: Rhythm: Harnessing Data Parallel Hardware for Server Workloads, *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '14, New York, NY, USA, ACM, pp. 19–34 (2014).
4) Jang, K., Han, S., Han, S., Moon, S. and Park, K.: SSLShader: Cheap SSL Acceleration with Commodity Processors, *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, Berkeley, CA, USA, USENIX Association, pp. 1–14 (2011).
5) Hetherington, H., T., O'Connor, Mike, Aamodt and M., T.: MemcachedGPU: Scaling-up Scale-out Key-value Stores, *Proceedings of the Sixth ACM Symposium on Cloud Computing*, SoCC '15, New York, NY, USA, ACM, pp. 43–57 (2015).
6) : AMAZON.COM, INC. FORM 10-K, `http://www.sec.gov/Archives/edgar/data/1018724/000119312513028520/d445434d10k.htm`.
7) : HPE Integrity NonStop systems, `http://www8.hp.com/us/en/products/servers/integrity/nonstop/index.html`.