Impacto do Subsistema de Memória da Arquitetura Kepler no Desempenho de uma Aplicação de Propagação de Onda *

Ricardo K. Lorenzoni[1], Matheus S. Serpa[2], Edson L. Padoin[1,2], Philippe O. A. Navaux[2], Jean-François Méhaut[3]

[1]Universidade Reg. do Noroeste do Estado do Rio G. do Sul (UNIJUI) – Ijuí – RS – Brasil

{ricardo.lorenzoni, padoin}@unijui.edu.br

[2]Universidade Federal do Rio Grande do Sul (UFRGS) – Porto Alegre – RS – Brasil

{msserpa, navaux}@inf.ufrgs.br

[3]Laboratoire d'Informatique de Grenoble (LIG), Universite Grenoble Alpes – França

jean-francois.mehaut@imag.fr

Resumo. *Arquiteturas heterogêneas foram introduzidas com intuito de melhorar o desempenho de simulações numéricas. Entretanto, essas arquiteturas possuem hierarquias de memória complexas as quais são controladas pelo programador. Nossa proposta é investigar o impacto do uso das diferentes memórias de uma GPU em uma aplicação de propagação de onda. Possibilitando assim, a otimização do desempenho da aplicação e descrever o comportamento de cada memória.*

1. Introdução

Simulações numéricas vêm sendo utilizadas com a intenção de prever o comportamento de diferentes fenômenos. No entanto, a precisão e acurácia dos modelos estão associadas aos recursos computacionais disponíveis. Avanços tecnológicos em processadores de propósito geral tem reduzido o tempo de execução destas simulações mas seu desempenho é limitado pelo *Thermal Design Power* (TDP). Diante disso, arquiteturas heterogêneas tem-se tornado uma alternativa para a execução deste tipo de aplicações [Barak et al. 2010].

Neste contexto, o objetivo deste trabalho é analisar o desempenho de uma aplicação que simula a propagação de ondas executada em uma arquitetura heterogênea do tipo GPU. Será utilizada a ferramenta **nvprof** [Nvidia 2016] para obter medidas precisas do impacto de diferentes otimizações aplicadas no *kernel*. Desta forma, será possível analisar como o uso de diferentes memórias da arquitetura impactam no desempenho dessa aplicação.

2. Trabalhos Relacionados

Diversos trabalhos têm sido desenvolvidos sobre a otimização de aplicações de propagação de ondas. Em [Nasciutti and Panetta 2016], os autores aplicam otimizações na computação de estênceis 3D e analisam o desempenho de GPUs focando no uso adequado da hierarquia de memória e concluem que a codificação mais indicada é baseada na combinação do uso do cache somente leitura, internalização do laço em *Z* e o reuso de registradores.

*Trabalho parcialmente apoiado por CNPq, CAPES, FAPERGS e FINEP. Pesquisa realizada no contexto do Laboratório Internacional Associado LICIA e tem recebido recursos do programa EU H2020 e do MCTI/RNP-Brasil sob o projeto HPC4E de número 689772.

Em [Jang et al. 2011] os autores apresentam técnicas de direcionamento de vetorização e seleção de memoria algorítmica para aprimorar a eficiência de memória em GPUs, obtendo ganhos de desempenho entre 11,4 e 13,5 vezes. Um método proposto por [Ikuzawa et al. 2016] obteve ganho de desempenho de 3,9 vezes na transformação de ondas discretas baseadas em elevação, ao unificar as regiões de alocação de memória de entrada e saída.

Neste sentido, o foco do presente artigo é analisar detalhadamente a relação entre diversos aspectos de utilização da hierarquia de memória da GPU e o desempenho de uma aplicação real de propagação de onda.

3. Metodologia

Os experimentos estão sendo realizados em uma GPU Tesla k20m que possui 2496 núcleos CUDA. Os resultados serão a média de 30 execuções aleatórias. Além de tempo de execução, outras métricas como utilização, largura de banda e taxa de acerto do subsistema de memória serão avaliadas com auxílio da ferramenta **nvprof**.

A microarquitetura da GPU Kepler possui três memórias *cache* de primeiro nível que podem ser utilizadas para melhorar o desempenho de suas aplicações. A *cache L1* é usada automaticamente para dados locais e *register spill*. As *caches shared* e *read-only* são ambas controladas pelo programador. A diretiva *shared* indica dados que serão alocados na *cache shared* para leitura e escrita. A intrínsica *lgd()* indica dados de apenas leitura que serão armazenados na *cache read-only*. Com intuito de investigar o impacto dessas memórias no desempenho da aplicação, três versões da aplicação de propagação de onda foram desenvolvidas.

4. Resultados Esperados

Ao final da pesquisa, espera-se compreender o comportamento de cada memória e caracterizar o impacto do uso de cada uma no desempenho da aplicação. Com o auxílio de contadores de *hardware* da placa, serão discutidos os motivos para a redução ou o aumento do desempenho da aplicação.

Trabalhos futuros, incluem a utilização de *benchmarks* sintéticas e outras aplicações reais focando no estudo dos impactos do uso de cada memória no desempenho das aplicações.

References

[Barak et al. 2010] Barak, A., Ben-Nun, T., Levy, E., and Shiloh, A. (2010). A package for opencl based heterogeneous computing on clusters with many gpu devices. In *2010 IEEE International Conference On Cluster Computing Workshops and Posters (CLUSTER WORKSHOPS)*, pages 1–7.

[Ikuzawa et al. 2016] Ikuzawa, T., Ino, F., and Hagihara, K. (2016). Reducing memory usage by the lifting-based discrete wavelet transform with a unified buffer on a gpu. *Journal of Parallel and Distributed Computing*, 93:44–55.

[Jang et al. 2011] Jang, B., Schaa, D., Mistry, P., and Kaeli, D. (2011). Exploiting memory access patterns to improve memory performance in data-parallel architectures. *IEEE Transactions on Parallel and Distributed Systems*, 22(1):105–118.

[Nasciutti and Panetta 2016] Nasciutti, T. C. and Panetta, J. (2016). Impacto da arquitetura de memória de gpgpus na velocidade da computação de estênceis.

[Nvidia 2016] Nvidia (2016). Developer Zone - CUDA Toolkit Documentation.