Analisando o Impacto do DVFS no Desempenho e Energia de Aplicações Paralelas em GPUs

Thiago dos S. Gonçalves[1] **e Arthur F. Lorenzon**[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul – Brasil

{thiago.goncalves, aflorenzon}@inf.ufrgs.br

Resumo. *A capacidade de processamento paralelo de unidades de processamento gráfico (GPUs) tornou essencial seu uso em aceleração de aplicações de inteligência artificial. Com forte presença de multiplicação de matrizes nessas aplicações, novas estratégias são necessárias para obter melhor eficiência energética. Dessa maneira, analisamos o impacto de métricas da cache em uma GPU e mostramos uma diferença de 19,87% em gasto de energia com pequenos ganhos no desempenho.*

1. Introdução

A popularização de aplicações que utilizam inteligência artificial (IA) nos últimos anos tornou o uso de unidades de processamento gráfico (GPUs) essencial para processamentos em larga escala dessas aplicações [Sharma et al. 2016]. Utilizar a GPU nessas aplicações resulta em ganhos significativos de desempenho devido a sua especialidade em realizar cálculos em paralelo, com melhoras no tempo de execução dessas aplicações [Baji 2017]. Porém, essa melhora no desempenho também pode resultar em um consumo maior de energia, sendo necessário estudar estratégias para balancear um melhor tempo de execução com um menor consumo energético.

Neste contexto, diferentes estratégias podem ser utilizadas para melhorar a eficiência energética das GPUs durante a execução de programas. Exemplos incluem a otimização do código a ser executado (e.g., sobrepor etapas de comunicação com computação, selecionar o melhor número de blocos e *threads* por bloco para executar o *kernel*, entre outros), aprimorar a paralelização entre múltiplas GPUs, ou ainda, encontrar a frequência ideal de operação dos componentes de *hardware* da GPU. No entanto, um fator em comum entre essas estratégias é a necessidade de se compreender o comportamento da aplicação a ser executada através de métricas de *hardware* e *software* e o impacto da frequência de operação da GPU nas aplicações paralelas.

Deste modo, este trabalho objetiva analisar o comportamento do consumo de energia e desempenho durante a execução de quatro aplicações paralelizadas com CUDA em uma GPU da NVIDIA ao alterar a frequência de operação da GPU. Para tanto, serão considerados diferentes frequências de operação das unidades computacionais para relacionar o consumo energético e tempo de execução com o comportamento da aplicação. Com isso, mostramos que características de uso da memória cache L1 e L2 em aplicações podem significar um aumento de 20% no gasto de energia com uma melhora de apenas 1,78% em seu tempo de execução.

2. Fundamentação Teórica e Trabalhos Relacionados

Unidades de processamento gráfico surgiram como uma parte do computador focada na rasterização e processamento geral de vídeo, com um foco em realizar vários cálculos

simples em paralelo. E assim, o uso de GPUs na computação de alto desempenho leva a ganhos no desempenho de programas paralelizáveis, mais especificamente, os de cálculos com matrizes. Para realizar melhores otimizações, é preciso conhecer melhor as demandas do programa através de métricas como uso de memória e dos *Streaming Multiprocessors* (SMs) para otimizar o uso dessas partes específicas da GPU. Uma dessas formas é a variação da frequência de operação, que define a velocidade em que operações serão realizadas, e dependendo da aplicação, pode resultar em ganhos significativos de eficiência energética.

Diversos trabalhos analisaram o impacto de alterar a frequência considerando diferentes características de aplicação na execução de problemas em GPU. *GPU-NEST* é uma proposta para caracterizar a eficiência energética em sistemas de inferência de IA com múltiplas GPUs [Jahanshahi et al. 2020]. Li et al. compara a eficiência energética entre CPU e GPU em redes neurais convolucionais, aprofundando-se nos efeitos de tecnologias diferentes na eficiência energética [Li et al. 2016]. sBEET é um escalonador que promete otimizar o consumo de energia de GPUs [Wang et al. 2021] usando métodos de multi-tarefa espacial. Anzt et al. demonstra otimizações de implementações em CUDA de bibliotecas de multiplicação entre matrizes esparsas [Anzt et al. 2015]. Fet et al. propõe um método de criptografia paralelo que executa em plataformas heterogêneas de processador e GPU [Fei et al. 2020]. Diferentemente dos trabalhos mencionados, nosso trabalho utiliza métricas específicas da GPU que está sendo utilizada em cada programa, e relaciona essas métricas com fatores no tempo de execução e gasto energético das aplicações.

3. Metodologia

Foram consideradas 4 aplicações de operações com matrizes implementadas em CUDA do repositório *HeCBench* [Jin and Vetter 2023]: *spgemm* (Multiplicação entre matrizes esparsas e densas), *blas-gemmBatched* (Multiplicação entre matrizes em lotes), *simpleSpmv* (Multiplicação entre matrizes esparsas e vetores), *sps2d* (Conversão de matriz densa para esparsa). A Tabela 1 mostra a taxa média de acerto da cache L1 e L2, coletados pelo NVIDIA *Nsight Compute* para cada aplicação. Considerando a Tabela 1, as 4 aplicações possuem características diferentes em relação ao uso de memória. As aplicações foram executadas com o conjunto de entrada padrão. Os experimentos foram realizados em uma máquina com uma GPU NVIDIA *P100* utilizando a arquitetura NVIDIA *Pascal* com 3584 CUDA *Cores* e 16GB de memória RAM HBM2. Essa GPU executa em frequências de $544MHz$ a $1480MHz$ nos núcleos de processamento gráfico, e uma frequência fixa de $715MHz$ em sua memória VRAM. A máquina também possui duas CPUs Intel *Xeon E5-2699 v4*, com 256GB de memória RAM DDR4.

Métricas	**Cache L1**	**Cache L2**
spgemm	11,01%	72,88%
blas-gemmBatched	35,87%	22,67%
simpleSpmv	74,09%	15,49%
sps2d	13,33%	52,82%

Tabela 1. Taxa média de acerto de leituras e escritas na memória cache L1 e L2

Para os experimentos, foram consideradas as execuções das aplicações nas

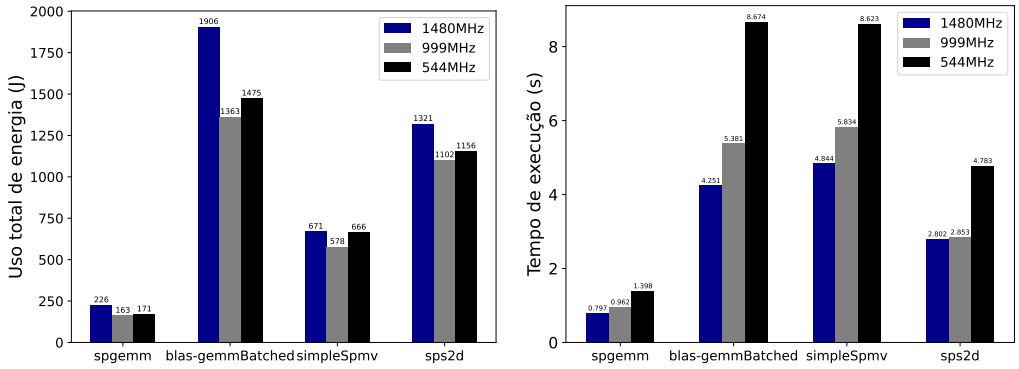

Figura 1. Da esquerda para direita, gráficos de uso de energia em joules e de tempo de execução em segundos das aplicações *spgemm*, *blas-gemmBatched*, *simpleSpmv*, e *sps2d*

.

frequências $1480MHz$ (maior possível), $544MHz$ (menor possível), e $999MHz$ (mediana), coletando dados de tempo de execução das aplicações e de consumo de energia. Os dados de tempo de execução foram coletados a partir da saída de cada aplicação. A potência da GPU foi coletada de 20 em 20 milissegundos utilizando a ferramenta *nvidia-smi*, e o consumo total de energia calculado como a integral da potência pelo tempo total de execução. As métricas em nível de GPU de cada aplicação foram coletadas com o NVIDIA NCU, sendo elas a taxa de acerto das memórias cache, executando na frequência mais alta da máquina. Como o NCU re-executa operações para coletar diversas métricas, os resultados acabam saindo como uma média da métrica encontrada ao longo de várias execuções. Os resultados apresentados na próxima seção consideram a média de 20 execuções, com desvio padrão inferior à 1.0%.

4. Resultados

Nesta Seção, é discutido o impacto que a frequência de operação da GPU tem no desempenho e consumo de energia das quatro aplicações alvo. Como pode ser observado na Figura 1, todas as aplicações obtiveram um aumento em seu tempo de execução ao diminuir a frequência de operação. Por exemplo, na aplicação *sps2d*, o tempo de execução diminui em 40,35% aumentando a frequência de operação de $544MHz$ para $999MHz$, porém apenas diminui em 1,78% aumentando a frequência de $999MHz$ para $1480MHz$, e piora seu consumo energético em 19,87%. Isso pode ser relacionado com a taxa de acerto da memória cache L1 durante o programa, pois a pequena taxa de acerto de 13,33% indica maior perda de ciclos com a busca de informação na cache de nível mais alto e na DRAM, e como a taxa de acerto na memória L2 é de apenas 52,82%, o programa é limitado pela latência da memória DRAM. Na aplicação *blas-gemmBatched*, vemos uma redução de tempo de execução de 38% entre as frequências mínimas e medianas de operação, porém, seu consumo energético aumenta em 40% entre a frequência mediana e a máxima, com a redução do tempo de execução sendo de apenas 21%. A baixa taxa de acerto da cache L1 e L2 indica maior uso da memória RAM da GPU, o que pode acabar ocasionando em vários ciclos perdidos devido a alta latência do acesso a memória RAM, e esses ciclos perdidos em frequências mais altas ocasionam em maior gasto energético. As aplicações *spgemm* e *simpleSpmv* possuíram diferenças no tempo de execução de 43% comparando

a maior e a menor frequência de operação, e diferenças de 27,9% e 13,9% no consumo total de energia entre o melhor e pior caso. Como ambas aplicações possuem taxas de acerto altas na cache L1 ou L2, é possível concluir que não acabam sendo limitadas pela latência de memória, mas a execução em frequências mais moderadas acaba trazendo ganhos energéticos.

5. Conclusão

Este trabalho avaliou o impacto da frequência de operação da GPU no desempenho e consumo de energia de aplicações matriciais, considerando métricas de acesso a cache. Dessa forma, mostramos que altas taxas de erros na cache causam pior eficiência energética executando aplicações em frequências maiores, e no pior caso, pode aumentar o consumo energético da aplicação em até 40% com ganhos reduzidos. Como futuro trabalho, planejamos expandir a quantidade de aplicações e frequências testadas, como também aumentar a quantidade de métricas da execução de cada aplicação.

Agradecimentos

Essa pesquisa foi parcialmente financiada pelos órgãos CAPES, FAPERGS, e CNPq.

Referências

Anzt, H., Tomov, S., and Dongarra, J. (2015). Energy efficiency and performance frontiers for sparse computations on gpu supercomputers. In *Proceedings of the Sixth International Workshop on Programming Models and Applications for Multicores and Manycores*, PMAM '15, page 1–10, New York, NY, USA. Association for Computing Machinery.

Baji, T. (2017). GPU: the biggest key processor for AI and parallel processing. In Takehisa, K., editor, *Symposium on Photomask and Next-Generation Lithography Mask Technology*, volume 10454, page 1045406. International Society for Optics and Photonics, SPIE.

Fei, X., Li, K., Yang, W., and Li, K. (2020). Analysis of energy efficiency of a parallel aes algorithm for cpu-gpu heterogeneous platforms. *Parallel Computing*, 94-95:102621.

Jahanshahi, A., Sabzi, H. Z., Lau, C., and Wong, D. (2020). Gpu-nest: Characterizing energy efficiency of multi-gpu inference servers. *IEEE Computer Architecture Letters*, 19(2):139–142.

Jin, Z. and Vetter, J. S. (2023). A benchmark suite for improving performance portability of the sycl programming model. In *2023 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 325–327.

Li, D., Chen, X., Becchi, M., and Zong, Z. (2016). Evaluating the energy efficiency of deep convolutional neural networks on cpus and gpus. In *IEEE international conferences on big data and cloud computing*, pages 477–484. IEEE.

Sharma, R., M, V., and Moharir, M. (2016). Revolutionizing machine learning algorithms using gpus. In *CSITSS*, pages 318–323.

Wang, Y., Karimi, M., Xiang, Y., and Kim, H. (2021). Balancing energy efficiency and real-time performance in gpu scheduling. In *2021 IEEE Real-Time Systems Symposium (RTSS)*, pages 110–122.