

GPGPU Performance Estimation with Core and Memory Frequency Scaling

Qiang Wang

Department of Computer Science
Hong Kong Baptist University, Hong Kong

Xiaowen Chu

Department of Computer Science
Hong Kong Baptist University, Hong Kong
HKBU Institute of Research and Continuing Education,
Shenzhen, China
chxw@comp.hkbu.edu.hk

ABSTRACT

Graphics processing units (GPUs) support dynamic voltage and frequency scaling to balance computational performance and energy consumption. However, simple and accurate performance estimation for a given GPU kernel under different frequency settings is still lacking for real hardware, which is important to decide the best frequency configuration for energy saving. We reveal a fine-grained analytical model to estimate the execution time of GPU kernels with both core and memory frequency scaling. We validate our model on two GPU platforms, Maxwell and Pascal. Over a wide scaling range of both core and memory frequencies among 20 GPU kernels, our model achieves high accuracy of 4.83% average error on the Maxwell GPU. Compared to the cycle-level simulators, our model only needs simple micro-benchmarks to extract a set of hardware parameters and kernel performance counters to produce such high accuracy.

KEYWORDS

Graphics Processing Units, Dynamic Voltage and Frequency Scaling, GPU Performance Modeling

1 INTRODUCTION

Energy conservation techniques in modern computers are generally based on dynamic voltage and frequency scaling (DVFS). GPUs usually support simple automatic voltage and frequency adjustment to save power and protect the hardware. Nevertheless, GPUs hardly gain the best energy efficiency under the default voltage and frequency settings [1, 6] and still have potentials of energy conservation. To find the most energy-efficient DVFS configurations, energy consumption under different DVFS settings should be predicted, which requires modeling the performance of GPUs under various voltage and frequency settings.

Some state-of-the-art work has revealed the analytical pipe-line GPU performance model [3, 5, 7, 8], which emphasizes the relationship between compute cycles and memory latency. However, most of the previous models only work under default GPU frequency settings. Kernel behavior may change significantly when the core and memory frequencies are adjusted.

We believe that a fast and accurate GPU performance model is a key ingredient in energy conservation with the DVFS technique and that it should be applicable to real hardware. We first attempt to model the memory system of a GPU with a simplified first-come-first-served queue in which the service rate depends on the memory

frequency. Based on this, we propose a GPGPU performance estimation model that considers both core and memory frequency scaling. Notably, this is the first performance modeling paper that explicitly considers both core and memory frequency scaling. We make the following contributions:

- (1) We model the memory system of a GPU with a simplified queuing model related to frequency.
- (2) We establish an analytical GPU performance model with both core and memory frequency scaling.
- (3) On Maxwell GPU hardware, our performance model achieves a mean absolute percentage error (MAPE) of 4.83% across 36 frequency settings with up to 2× scaling among 20 kernels. We also achieve MAPEs of 0.73% to 11.2% for each single kernel, indicating the great accuracy and low variance of our performance model.

2 METHODOLOGY

2.1 Memory Modeling with Frequency Scaling

We first estimate the total execution time of pure DRAM access (T^{Gm}) with a simplified queuing model. The minimum latency happens when the memory system is idle and only contains the overhead of path traveling between SMs and DRAM, and data access in DRAM. T^{Gm} can be calculated using Eq. (1). λ denotes the arrival rate of the coming memory requests. N^W denotes the total warp number. L^{Dm} denotes the minimum memory latency with no memory contention. N^{GT} denotes the number of global memory transactions of one warp.

$$T^{Gm} = \frac{1}{\lambda} \times (N^W - 1) + L^{Dm} \times N^{GT} \quad (1)$$

If the memory system is saturated due to intensive memory requests, most of the requests must wait in the queue until the previous ones have been finished. T^{Gm} can be calculated using Eq. (2). D^{Dm} denotes the service time of one memory transaction.

$$T^{Gm} = L^{Dm} + D^{Dm} \times N^{GT} \times N^W \quad (2)$$

We measure the L^{Dm} and D^{Dm} under different memory frequencies. We find that they can be fitted by Eq. (3a) and Eq. (3b) with 0.9959 R-squared.

$$L^{Dm} = 222.78 \times f^{SM} / f^{MEM} + 277.32 \quad (3a)$$

$$D^{Dm} = 805.03 / f^{MEM} + 8.1762 \quad (3b)$$

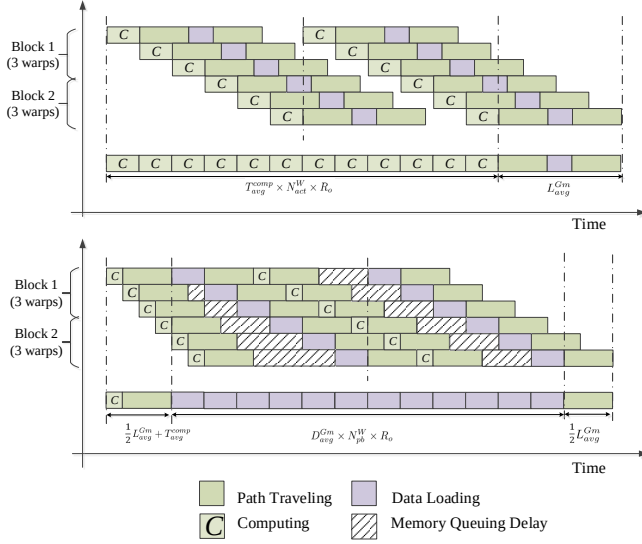


Figure 1: The upper figure shows execution time pipeline of a compute-dominated kernel. The bottom figure shows execution time pipeline of a memory-intensive kernel.

2.2 Graphics Processing Unit Performance Modeling with Frequency Scaling

Once we obtain the time consumption of one round of active warps (T^{act}), the total execution time of a kernel can be estimated using Eq. (4). N^B denotes the total number of thread blocks. N_{pb}^W denotes the number of warps per block. N^{SM} denotes the number of SMs. N_{act}^W denotes the number of active warps per SM.

$$T^{exec} = T^{act} \times (N_{pb}^W \times N^B / (N_{act}^W \times N^{SM})) \quad (4)$$

Compute-dominated kernel: When there are enough compute instructions to be issued and the memory requests are not too intensive due to long computation periods, the global memory latency can be hidden, as illustrated by the upper part of Fig. 1. We can estimate T^{act} using Eq. (5):

$$T^{act} = T_{avg}^{comp} \times N_{act}^W \times R_o + L^{Gm}, \quad (5)$$

where $T_{avg}^{comp} \geq D^{Gm}$ and $T_{avg}^{comp} \times (N_{act}^W - 1) \geq L^{Gm}$. R_o denotes the repeat times of a pair of computation period and global memory transaction period.

Memory-dominated kernel: When the memory bandwidth is saturated or there are not enough warps to issue compute instructions, one memory request must wait until all of the outstanding requests have been finished. The bottom part of Fig. 1 demonstrates this case. We can regard the compute cycles as the inter-arrival time of two consequent memory requests. We can estimate T^{act} using Eq. (6) by focusing on the D_{avg}^{Gm} of each warp:

$$T^{act} = L^{Gm} + T_{avg}^{comp} + D_{avg}^{Gm} \times N_{pb}^W \times R_o, \quad (6)$$

where $T_{avg}^{comp} \leq D^{Gm}$ and $T_{avg}^{comp} + L^{Gm} \geq D^{Gm} \times (N_{act}^W - 1)$.

3 EXPERIMENTS

3.1 Experimental Methodology

On our tested NVIDIA GTX 980, we cover both core frequency and memory frequency at a $2\times$ range of scaling from 500 MHz to 1,000 MHz with a step size of 100 MHz. We validate our model among 20 realistic GPU kernels from CUDA SDK 8.0 [9] and Rodinia [2]. We repeat each experiment for 1,000 times and report the MAPEs.

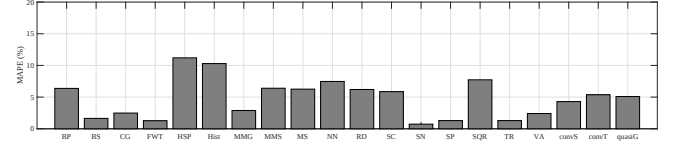


Figure 2: Mean absolute percentage error for each GPU kernel, GTX 980.

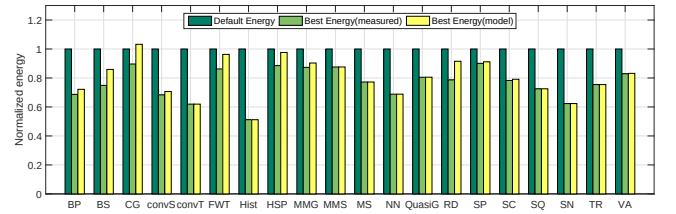


Figure 3: Active Energy Conservation of GPU

3.2 Experimental Results

3.2.1 Model Accuracy. Fig. 2 shows the mean absolute precision error (MAPE) of each tested GPU kernel across 36 frequency configurations. Our performance model estimates achieve high accuracy with an average MAPE of 4.83% across 720 unique experiments. For each kernel, the MAPEs range from 0.73% to 11.2% as shown in Fig. 2.

3.2.2 Energy Conservation. Combining our performance estimation model and the power model proposed by [10], we predict the energy consumption of each kernel under different frequency settings by multiplying the time and average power consumption and choose the one saving most energy. Fig. 3 illustrates the results. Our measured results illustrate that proper frequency settings may save an average 23.5% of energy consumption among the 20 tested kernels. With the predictions of our model, we finally achieve up to 48.8% energy saving and an average of 20% among 20 tested GPU kernels.

4 CONCLUSION

We demonstrate a GPGPU performance predictor for a wide range of core and memory frequencies. Our model can quickly and accurately predict the execution time of a GPU kernel on real hardware, which is important to deriving real-time energy conservation suggestions with DVFS techniques. We show that our performance estimation method can achieve a MAPE of 4.83% across up to $2\times$ both core and memory frequencies scaling among 20 realistic GPU kernels.

REFERENCES

- [1] Robert A Bridges, Neena Imam, and Tiffany M Mintz. 2016. Understanding GPU Power: A Survey of Profiling, Modeling, and Simulation Methods. *ACM CSUR* 49, 3 (2016), 41.
- [2] Shuai Che et al. 2009. Rodinia: A benchmark suite for heterogeneous computing. In *Workload Characterization (IISWC) 2009. IEEE International Symposium on*. IEEE, 44–54.
- [3] Xiaoming Chen et al. 2014. Run-time technique for simultaneous aging and power optimization in GPGPUs. In *Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE*. IEEE, 1–6.
- [4] John L Hennessy and David A Patterson. 2011. *Computer architecture: a quantitative approach*. Elsevier.
- [5] Sunpyo Hong and Hyesoon Kim. 2009. An Analytical Model for a GPU Architecture with Memory-level and Thread-level Parallelism Awareness. In *Proceedings of the 36th ISCA (ISCA '09)*.
- [6] Xinxin Mei, Qiang Wang, and Xiaowen Chu. 2017. A survey and measurement study of GPU DVFS on energy conservation. *Digital Communications and Networks* 3, 2 (2017), 89 – 100.
- [7] Rustam Miftakhutdinov, Eiman Ebrahimi, and Yale N Patt. 2012. Predicting performance impact of DVFS for realistic memory systems. In *Proceedings of the 45th MICRO, 2012*.
- [8] Rajib Nath and Dean Tullsen. 2015. The CRISP performance model for dynamic voltage and frequency scaling in a GPGPU.
- [9] NVIDIA. 2016. GPU Computing SDK. [Online] <https://developer.nvidia.com/gpu-computing-sdk>. (2016).
- [10] Qiang Wang and Xiaowen Chu. 2017. GPGPU Power Estimation with Core and Memory Frequency Scaling. *SIGMETRICS Perform. Eval. Rev.* 45, 2 (Oct. 2017), 73–78.