A Detailed Model for Contemporary GPU Memory Systems

Mahmoud Khairy*, Akshay Jain*, Tor M. Aamodt†, Timothy G. Rogers*
*School of Electrical and Computer Engineering, Purdue University
{abdallm,akshayj,timrogers}@purdue.edu
†Electrical and Computer Engineering Department, University of British Columbia
aamodt@ece.ubc.ca

***Abstract*—This paper explores the impact of simulator accuracy on architecture design decisions in the general-purpose graphics processing unit (GPGPU) space. We enhance the most popular publicly available GPU simulator, GPGPU-Sim, by performing a rigorous correlation of the simulator with a contemporary GPU. Our enhanced GPU model is able to describe the NVIDIA Volta architecture in sufficient detail to reduce error in memory system counters by as much as $66\times$. The reduced error in the memory system further reduces execution time error by $2.5\times$. To demonstrate the accuracy of our enhanced model against a real machine, we perform a counter-by-counter validation against an NVIDIA TITAN V Volta GPU, demonstrating the relative accuracy of the new simulator versus the previous model.**

We go on to demonstrate that the simpler model discounts the importance of advanced memory system designs such as out-of-order memory access scheduling. Our results demonstrate that it is important for the academic community to enhance the level of detail in architecture simulators as system complexity continues to grow.

I. Introduction

In contemporary computer architecture research, simulation is commonly used to estimate the effectiveness of a new architectural design idea. High-level simulators enable architects to rapidly evaluate ideas at the expense of less accurate simulation results. Ideas that do not show promise in simulation are discarded while those that do show promise are refined in an iterative process. Our paper focuses on the simulation of massively parallel architectures, in particular GPUs. GPUs have witnessed rapid change and a widespread increase in their adoption with the rise of GPGPU computing and machine learning. In academia, the design of programmable accelerators is mostly carried out through modeling new techniques in high level GPU simulators. Over the past four years, there have been approximately 20 papers per year focusing on GPU-design at the top architecture conferences. 80% of those papers have used today's most popular open-source GPU simulator, GPGPU-Sim [1]. The relative popularity of GPGPU-Sim can be attributed to several factors, but it's most appealing aspect is perhaps the accuracy with which it models modern GPUs (relative to other open-source solutions). Such accuracy should provide a solid basleine for studying important architectural ideas that are relevant to future machines.

Recent work on validating GPGPU-Sim [2] has demonstrated that there are several areas where a lack of detail in the performance model creates a major source of error. However, the bulk of the error comes from the modeling of the memory system. Building on that observation, we perform a module-by-module redesign of the GPU's memory system, demonstrating its improved correlation with real hardware. Over a diverse set of more than 1700 kernels, our improved performance model decreases memory system error on several key metrics by as much as $66\times$, resulting in a $2.5\times$ reduction in execution-cycle error when modeling an NVIDIA TITAN V GPU. Table I presents the average absolute error rates of the previous GPGPU-Sim model versus our new model. The *old model* is a faithful representation of how papers currently scale GPGPU-Sim to model a contemporary GPU, without modifying the code. The *new model* represents all the changes implemented in this paper.

TABLE I: Average absolute error and correlation rates of the previous GPGPU-Sim model versus our enhanced memory system model for an NVIDIA Volta TITAN V.

Statistic	Mean Abs Error		Correlation	
	Old Model	New Model	Old Model	New Model
L1 Reqs	48%	0.5%	92%	100%
L1 Hit Ratio	41%	18%	89%	93%
L2 Reads	66%	1%	49%	94%
L2 Writes	56%	1%	99%	100%
L2 Read Hits	80%	15%	68%	81%
DRAM Reads	89%	11%	60%	95%
Execution Cycles	68%	27%	71%	96%

To demonstrate that our new model more closely matches the hardware, we perform a counter-by-counter validation of the improved memory system versus the previous memory system. During the course of this analysis we uncover a number of interesting insights into how contemporary hardware works. We utilize every publicly available resource to construct the new memory system model, capturing the details of what others have either disclosed or discovered. In the process of further microbenchmarking the memory system, we uncover and model several previously undocumented behaviours. For example, the details of the Volta streaming L1 cache and adaptive L1/shared memory partitioning policy, sectoring of the L1 and L2 caches, and the previously undocumented L2 write policy.

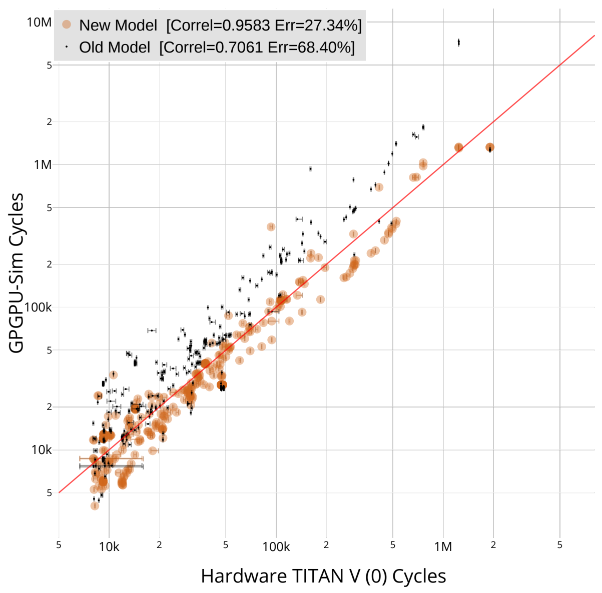

Fig. 1: Execution time correlation for the NVIDIA TITANV.

Using this more detailed model, we perform a case-study on the effect modeling error in the memory system can have on architectural design decisions. In particular, we demonstrate that out-of-order memory access scheduling, which appears relatively ineffective under the old model, yields a significant performance improvement when the level of detail in the memory system is increased to more closely match the hardware.

II. Execution Time Correlation

Figure 1 plots the cycle correlation of both the old and new models. The error bars on the x-axis represent the range of the silicon GPU execution time for a particular kernel over 10 successive runs of the application. To minimize the effect of kernels where the silicon measurements are noisy because of their short duration, only kernels that ran for at least 8000 GPU cycles in hardware are considered. Both the correlation and mean absolute error are significantly improved using the new memory model. In particular, the mean absolute error is reduced by $2.5\times$. This result confirms our working assumption that increasing the level of detail in the memory system would decrease the overall error in execution time. The old model generally overestimates the execution time of every kernel, where the new model is more balanced in under- versus over-estimation. In addition, the outliers in the new model have been greatly reduced in both number and magnitude.

III. Design decision case study

To demonstrate how modeling a contemporary memory system affects GPU architecture research, we perform a case study using the DRAM access scheduler. DRAM scheduling plays an important role in increasing memory bandwidth utilization and can improve the performance of memory-sensitive workloads. Figure 2 plots the performance gain resulting from

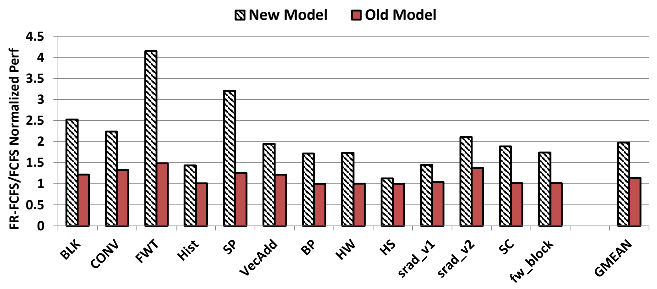

Fig. 2: FR_FCFS performance normalized to the FCFS in both old and new model.

the use of an out-of-order First-Ready, First-Come-First-Serve (FR_FCFS) [3] scheduler over simple FCFS scheduling, using both the new and old model, on a memory-sensitive subset of our workloads. The memory scheduling policy has a more dramatic impact on performance in the new model than in the old model, where some workloads are insensitive or show little difference. On average, there is a 20% performance increase when applying FR_FCFS in the old model, as opposed to a $2\times$ improvement with the new model. A more accurate memory coalescer, sectored caches and an advanced write-allocation policy increase the granularity of accesses seen at the DRAM controller, making its scheduling decisions more critical.

IV. Conclusion

This paper presents the most accurate open-source model of a contemporary GPU to date. Through detailed micro-benchmarking and modeling, we have uncovered a number of important design decisions made in the GPU memory system that no previous work has identified and no open-source simulator has attempted to model. The paper goes on to concretely demonstrate the effect this memory system error has on design decisions. Specifically, we show that error in the less detailed memory system model discounts the performance effects of out-of-order memory access scheduling, hampering potential research on the topic.

We refer the reader to [4] for a complete analysis of our model. The simulator changes described in this paper are available publicly on GPGPU-Sim's Github repo [5].

References

[1] A. Bakhoda, G. L. Yuan, W. W. Fung, H. Wong, and T. M. Aamodt, “Analyzing cuda workloads using a detailed gpu simulator,” in *Performance Analysis of Systems and Software, 2009. ISPASS 2009. IEEE International Symposium on*.

[2] A. Jain, M. Khairy, and T. G. Rogers, “A quantitative evaluation of contemporary gpu simulation methodology,” *Proceedings of the ACM on Measurement and Analysis of Computing Systems - SIGMETRICS*, vol. 2, no. 2, p. 35, 2018.

[3] S. Rixner, W. J. Dally, U. J. Kapasi, P. Mattson, and J. D. Owens, “Memory access scheduling,” in *ACM SIGARCH Computer Architecture News*, vol. 28, pp. 128–138, ACM, 2000.

[4] M. Khairy, A. Jain, T. M. Aamodt, and T. G. Rogers, “Exploring modern GPU memory system design challenges through accurate modeling,” *http://arxiv.org/abs/1810.07269*, 2018.

[5] GPGPU-sim Github, dev branch. https://github.com/gpgpu-sim/gpgpu-sim_distribution/tree/dev.