Further Closing the GAP: Improving the Accuracy of gem5's GPU Models

Vishnu Ramadas, Daniel Kouchekinia, Matthew D. Sinclair
University of Wisconsin-Madison
{vramadas, kouchekinia}@wisc.edu sinclair@cs.wisc.edu

I. INTRODUCTION

The breakdown in Moore's Law and Dennard Scaling is leading to drastic changes in the makeup and constitution of computing systems. For example, a single chip integrates 10-100s of cores and has a heterogeneous mix of general-purpose compute engines and highly specialized accelerators. Traditionally, computer architects have relied on tools like architectural simulators (e.g., Accel-Sim [1], gem5 [2], [3], gem5-SALAM [4], GPGPU-Sim [5], MGPUSim [6], Sniper-Sim [7], and ZSim [8]) to *accurately* perform early stage prototyping and optimizations for the proposed research. However, as systems become increasingly complex and heterogeneous, architectural tools are straining to keep up. In particular, publicly available architectural simulators are often not very representative of the industry parts they intend to represent. This leads to a mismatch in expectations; when prototyping new optimizations in gem5 users may draw the wrong conclusions about the efficacy of proposed optimizations if the tool's models do not provide high fidelity.

In this work, we focus on the gem5 simulator, the most popular platform for computer system simulation. In recent years gem5 has been used by ∼20% of simulation-based papers published in top-tier computer architecture conferences per year. Moreover, gem5 can run entire systems, including CPUs, GPUs [9], and accelerators [4], [10], as well as the operating system, runtime, network [11], [12], and other related components (including multiple ISAs). Thus, gem5 has the potential to allow users to study the behavior of the entire heterogeneous systems.

Unfortunately, some of gem5's models do not always provide high accuracy relative to their "real" counterparts. In particular, although gem5's GPU model provides high accuracy internally at AMD [9], the publicly available gem5 GPU model is often inaccurate, especially for the memory subsystem. To understand this, we designed a series of microbenchmarks designed to expose the latencies, bandwidths, and sizes of a variety of GPU components on real AMD GPUs. Our results showed that while gem5's GPU microarchitecture was relatively accurate (within 5-10% in most cases), gem5's memory subsytem was off by an average of 272% (645% max) for latency and 70% (693% max) for bandwidth. Accordingly, to help bridge this divide, we propose to design and use a new tool, GPU Accuracy Profiler (GAP), to compare and improve the behavior of gem5's simulated GPUs relative to real GPUs. By iteratively applying fixes and improvements to gem's GPU model via GAP, we will significantly improved its fidelity relative to real AMD GPUs. Although this work is still ongoing, our preliminary results (Section V) show significant promise: on average 25% error for latency and 16% error for bandwidth, respectively. Overall, by completing this work we hope to enable more widespread adoption of gem5 as an accurate platform for heterogeneous architecture research.

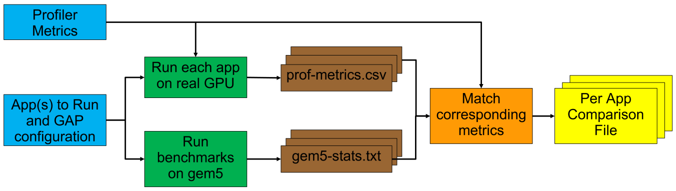

Fig. 1: gem5 GPU Accuracy Profiler (GAP)

II. BACKGROUND

The gem5 simulator is a widely used, open-source, cycle-level computer system simulator with around 700,000 lines of core code and around 250,000 additional lines of code in locally maintained external libraries. The gem5 simulator is used in computer system research to evaluate *novel* hardware designs. To support this use case, gem5 provides a robust API for researchers to modify and extend current models and to create new models in the gem5 infrastructure. At its core, gem5 contains an event-driven simulation engine. On top of this simulation engine, gem5 contains hundreds of cycle-level models for system components from accelerators [4], [10], coherent caches, CPUs (out-of-order designs, in-order designs, and others), I/O devices, GPUs, memories (including DDR3/4, GDDR5, HBM, HBM2, and HMC), on-chip interconnects, and many others. Using a Python scripting interface, users can configure systems and control the simulation to perform full-system performance analysis. Thus, gem5 provides a way to quickly prototype hardware-software co-design. The gem5 models are designed to have enough fidelity to boot Linux, run unmodified applications, and investigate cross-layer design proposals. As a result, it is used frequently in both academia and industrial research labs including AMD Research, ARM Research, and Google.

Recent work has enhanced and updated gem5's GPU support [9], including adding support for multi-chiplet setups [13] and running ML workloads [14], [15]. Moreover, contributors validated and released a Docker image with the proper software and libraries needed to run AMD's GCN3 and Vega GPU models in gem5. With this container, users can run the gem5 GPU model and build the desired ROCm applications out of the box without needing to properly install the appropriate ROCm software and libraries [14], [15]. However, as discussed in Section I, the publicly available GPU model is not always accurate.

III. PROPOSAL

To overcome the fidelity issues with gem5's GPU models, we propose to develop a tool called GPU Accuracy Profiler (GAP). As shown in Figure 1, GAP compares the performance of a given application (or microbenchmark) between real hardware and gem5. More specifically, given a set of applications, GAP runs each of them in isolation on both the real GPU and the corresponding gem5 GPU model (green, Figure 1). Before running on gem5, it configures the simulator to resemble the real GPU. Then, GAP measures the application's performance on both the real GPU and gem5. On the real GPU, it uses uses AMD's ROCm profiler [16] (rocProf) to collect statistics about the application's behavior (brown, Figure 1). For example, a user may want to compare cache hits/misses and runtime, although the desired profiler metrics can be configured by the user before running (blue, Figure 1). In gem5, GAP uses the simulator's corresponding generated statistics. Given these outputs, GAP then compares and matches[1] the real GPU and gem5's collected stats to generate a *per-application comparison file* (yellow, Figure 1). We propose to use this information to iteratively apply fixes and improve gem5's GPU model.

IV. METHODOLOGY

To measure the accuracy of gem5's GPU models, we will use existing benchmarks in gem5-resources [15]. However, since large benchmarks often make it difficult to isolate the behavior of specific GPU components in larger benchmarks. For example, in an initial prototype, we observed that running square (a simple GPU vector addition program) with GAP showed that the vector ALU utilization is within 1% between the real GPU and gem5, but the L2 cache misses differ by 821%, likely indicating that further tuning of the memory sub-system is required. Thus, we will also either develop or port a variety of GPU microbenchmarks [1], [17]–[19] to HIP (AMD's GPGPU programming language). We will feed these microbenchmarks into GAP to help isolate gem5's inaccuracies such as access latencies and bandwidths of L1, L2 caches, LDS, atomic operations, and global memory.

gem5's current GPU support currently focuses on Carrizo- and Vega-class GPUs. Thus, we propose to initially utilize an AMD Vega 20 (Radeon VII) as our baseline system in our experiments [20]. However, since GAP is highly configurable, we also plan to conduct similar experiments on more modern GPUs, such as AMD's MI200 GPU which is being added as a supported model in gem5 v24.0. In terms of metrics, our main goal will be to minimize Mean Absolute Error (MAE) – first for each microbenchmark in isolation, then for the larger benchmarks from a variety of GPGPU, graph analytics, HPC, and ML suites [21]–[26]. Since we are optimizing the existing gem5 GPU support, our main comparison will be its previous published version [9]. As part of this work, we plan to provide ”known good” configurations for a variety of modern AMD GPUs once the models are accurate. Moreover, we will integrate our tests into existing regression flows to help parties contributing to gem5's source code to ensure their additions do not hurt the behavior of gem5's GPU simulations.

Metric	Old Error	New Error
L1 Latency	2.18%	0.4%
L1 Bandwidth	41.75%	9.83%
L1 Scalar Latency	41.39%	0.98%
L2 Latency	0.08%	0.07%
L2 Bandwidth	52.15%	7.81%
Atomics Latency	51.79%	0.13%
Atomics Bandwidth	47.77%	7.7%

TABLE I: gem5 GPU component errors before and after our improvements, relative to a Vega 20 GPU.

V. PRELIMINARY RESULTS

Although much work remains in this project, as a proof of concept we analyzed how GAP shows gem5's GPU model compares against an AMD Vega 20 (Radeon VII) [20] for a subset of our proposed microbenchmarks. We configured gem5 to use resemble a Vega 20 and ran the same binaries on gem5 and the Vega 20. As shown in Table I, GAP has helped us significantly improve the fidelity by identifying components with significant errors. For example, we found that the public gem5 GPU support assumed all atomics were system-scope [27], [28], even when cheaper scopes (e.g., device scope) were specified by the program. Similarly, AMD GPUs have ISA extensions that allow loads and stores to bypass one or more levels of cache, which gem5 did not previously support. Overall, we reduced the error from an average of 272% for latency and 70% for bandwidth to 25% error for latency and 16% error for bandwidth, respectively, for the microbenchmarks. However, much work remains. For example, gem5's GPU main memory model seemingly does not model HBM2 well (which Vega 20's use), resulting in significant error: 125% for latency, 85% for bandwidth.

VI. RELATED WORK

Although some tools [1], [29] have validation specific GPU components, our work differs in that gem5 models the entire computing stack, including the drivers, runtime, and Command Processor interface between the host and device. Thus, we must design a much broader set of tests to ensure accuracy for the GPU in the context of the entire system. Nevertheless, we plan to use these works as inspiration, especially when designing tests for specific components. Similarly, we hope to leverage prior work on designing GPU microbenchmarks for testing features like bandwidth, latency, and size [1], [19]. However, unlike these prior works, since AMD GPUs do

---

[1] Our scripts do matching by checking all cases since rocProf has few metrics. We plan to use machine learning to automate this process.

not have an intermediate ISA like NVIDIA's PTX we must often write tests in hand-tuned assembly for specific to the architecture we are testing [9].

ACKNOWLEDGMENTS

This work is supported in by the Semiconductor Research Corporation grant 3151.001 and National Science Foundation grant ENS-1925485.

REFERENCES

[1] M. Khairy, Z. Shen, T. M. Aamodt, and T. G. Rogers, "Accel-Sim: An Extensible Simulation Framework for Validated GPU Modeling," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture*, ser. ISCA, 2020, pp. 473–486.

[2] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, 2011.

[3] J. Lowe-Power, A. M. Ahmad, A. Akram, M. Alian, R. Amslinger, M. Andreozzi, A. Armejach, N. Asmussen, S. Bharadwaj, G. Black, G. Bloom, B. R. Bruce, D. R. Carvalho, J. Castrillon, L. Chen, N. Derumigny, S. Diestelhorst, W. Elsasser, M. Fariborz, A. Farmahini-Farahani, P. Fotouhi, R. Gambord, J. Gandhi, D. Gope, T. Grass, B. Hanindhito, A. Hansson, S. Haria, A. Harris, T. Hayes, A. Herrera, M. Horsnell, S. A. R. Jafri, R. Jagtap, H. Jang, R. Jeyapaul, T. M. Jones, M. Jung, S. Kannoth, H. Khaleghzadeh, Y. Kodama, T. Krishna, T. Marinelli, C. Menard, A. Mondelli, T. Mück, O. Naji, K. Nathella, H. Nguyen, N. Nikoleris, L. E. Olson, M. Orr, B. Pham, P. Prieto, T. Reddy, A. Roelke, M. Samani, A. Sandberg, J. Setoain, B. Shingarov, M. D. Sinclair, T. Ta, R. Thakur, G. Travaglini, M. Upton, N. Vaish, I. Vougioukas, Z. Wang, N. Wehn, C. Weis, D. A. Wood, H. Yoon, and Éder F. Zulian, "The gem5 simulator: Version 20.0+," 2020.

[4] S. Rogers, J. Slycord, M. Baharani, and H. Tabkhi, "gem5-SALAM: A System Architecture for LLVM-based Accelerator Modeling," in *53rd Annual IEEE/ACM International Symposium on Microarchitecture*, 2020, pp. 471–482.

[5] 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*, 2009.

[6] Y. Sun, T. Baruah, S. A. Mojumder, S. Dong, X. Gong, S. Treadway, Y. Bao, S. Hance, C. McCardwell, V. Zhao, H. Barclay, A. K. Ziabari, Z. Chen, R. Ubal, J. L. Abellán, J. Kim, A. Joshi, and D. Kaeli, "MGPUSim: Enabling Multi-GPU Performance Modeling and Optimization," in *Proceedings of the 46th International Symposium on Computer Architecture*, ser. ISCA '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 197–209. [Online]. Available: https://doi.org/10.1145/3307650.3322230

[7] T. E. Carlson, W. Heirman, and L. Eeckhout, "Sniper: Exploring the Level of Abstraction for Scalable and Accurate Parallel Multi-core Simulation," in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC, 2011, pp. 1–12.

[8] D. Sanchez and C. Kozyrakis, "ZSim: Fast and Accurate Microarchitectural Simulation of Thousand-core Systems," in *Proceedings of the 40th Annual International Symposium on Computer Architecture*, ser. ISCA '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 475–486. [Online]. Available: https://doi.org/10.1145/2485922.2485963

[9] A. Gutierrez, B. M. Beckmann, A. Dutu, J. Gross, M. LeBeane, J. Kalamatianos, O. Kayiran, M. Poremba, B. Potter, S. Puthoor, M. D. Sinclair, M. Wyse, J. Yin, X. Zhang, A. Jain, and T. Rogers, "Lost in Abstraction: Pitfalls of Analyzing GPUs at the Intermediate Language Level," in *2018 IEEE International Symposium on High Performance Computer Architecture*, ser. HPCA, Feb 2018, pp. 608–619.

[10] Y. S. Shao, S. L. Xi, V. Srinivasan, G.-Y. Wei, and D. Brooks, "Co-designing accelerators and SoC interfaces using gem5-Aladdin," in *49th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO, 2016, pp. 1–12.

[11] N. Agarwal, T. Krishna, L.-S. Peh, and N. K. Jha, "GARNET: A Detailed On-chip Network Model Inside a Full-system Simulator," in *IEEE International Symposium on Performance Analysis of Systems and Software*, ser. ISPASS, 2009, pp. 33–42.

[12] S. Bharadwaj, J. Yin, B. Beckmann, and T. Krishna, "Kite: A Family of Heterogeneous Interposer Topologies Enabled via Accurate Interconnect Modeling," in *57th ACM/IEEE Design Automation Conference*, ser. DAC, 2020, pp. 1–6.

[13] B. W. Yogatama, M. D. Sinclair, and M. M. Swift, "Enabling Multi-GPU Support in gem5," in *3rd gem5 Users' Workshop*, June 2020.

[14] K. Roarty and M. D. Sinclair, "Modeling Modern GPU Applications in gem5," in *3rd gem5 Users' Workshop*, June 2020.

[15] B. R. Bruce, A. Akram, H. Nguyen, K. Roarty, M. Samani, M. Fariborz, T. Reddy, M. D. Sinclair, and J. Lowe-Power, "Enabling Reproducible and Agile Full-System Simulation," in *IEEE International Symposium on Performance Analysis of Systems and Software*, ser. ISPASS, 2021.

[16] AMD, "AMD ROCm Profiler," https://rocmdocs.amd.com/en/latest/ROCm_Tools/ROCm-Tools.html, 2021.

[17] T. Deakin, J. Price, M. Martineau, and S. McIntosh-Smith, "GPU-STREAM v2.0: Benchmarking the Achievable Memory Bandwidth of Many-Core Processors Across Diverse Parallel Programming Models," in *High Performance Computing*, M. Taufer, B. Mohr, and J. M. Kunkel, Eds. Cham: Springer International Publishing, 2016, pp. 489–507.

[18] M. Khairy, A. Jain, T. M. Aamodt, and T. G. Rogers, "Exploring modern GPU memory system design challenges through accurate modeling," *CoRR*, vol. abs/1810.07269, 2018. [Online]. Available: http://arxiv.org/abs/1810.07269

[19] H. Wong, M.-M. Papadopoulou, M. Sadooghi-Alvandi, and A. Moshovos, "Demystifying GPU Microarchitecture Through Microbenchmarking," in *IEEE International Symposium on Performance Analysis of Systems Software*, ser. ISPASS, 2010, pp. 235–246.

[20] "AMD Radeon VII," https://www.techpowerup.com/gpu-specs/radeon-vii.c3358 , TechPowerUp, 2019.

[21] S. Che, B. M. Beckmann, S. K. Reinhardt, and K. Skadron, "Pannotia: Understanding Irregular GPGPU Graph Applications," in *IEEE International Symposium on Workload Characterization*, ser. IISWC, Sept 2013, pp. 185–195.

[22] S. Dong and D. Kaeli, "DNNMark: A Deep Neural Network Benchmark Suite for GPUs," in *Proceedings of the General Purpose GPUs*, ser. GPGPU. New York, NY, USA: ACM, 2017, pp. 63–72. [Online]. Available: http://doi.acm.org/10.1145/3038228.3038239

[23] S. Narang and G. Diamos, "An update to DeepBench with a focus on deep learning inference," https://svail.github.io/DeepBench-update/, 2017.

[24] Lawrence Livermore National Labs, "CORAL-2 Benchmarks," https://asc.llnl.gov/coral-2-benchmarks, 2020.

[25] V. J. Reddi, C. Cheng, D. Kanter, P. Mattson, G. Schmuelling, C.-J. Wu, B. Anderson, M. Breughe, M. Charlebois, W. Chou, R. Chukka, C. Coleman, S. Davis, P. Deng, G. Diamos, J. Duke, D. Fick, J. S. Gardner, I. Hubara, S. Idgunji, T. B. Jablin, J. Jiao, T. S. John, P. Kanwar, D. Lee, J. Liao, A. Lokhmotov, F. Massa, P. Meng, P. Micikevicius, C. Osborne, G. Pekhimenko, A. T. R. Rajan, D. Sequeira, A. Sirasao, F. Sun, H. Tang, M. Thomson, F. Wei, E. Wu, L. Xu, K. Yamada, B. Yu, G. Yuan, A. Zhong, P. Zhang, and Y. Zhou, "MLPerf Inference Benchmark," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture*, ser. ISCA, 2020, pp. 446–459.

[26] P. Mattson, C. Cheng, C. Coleman, G. Diamos, P. Micikevicius, D. A. Patterson, H. Tang, G. Wei, P. Bailis, V. Bittorf, D. Brooks, D. Chen, D. Dutta, U. Gupta, K. M. Hazelwood, A. Hock, X. Huang, B. Jia, D. Kang, D. Kanter, N. Kumar, J. Liao, G. Ma, D. Narayanan, T. Oguntebi, G. Pekhimenko, L. Pentecost, V. J. Reddi, T. Robie, T. S. John, C. Wu, L. Xu, C. Young, and M. Zaharia, "MLPerf Training Benchmark," *CoRR*, vol. abs/1910.01500, 2019. [Online]. Available: http://arxiv.org/abs/1910.01500

[27] D. R. Hower, B. A. Hechtman, B. M. Beckmann, B. R. Gaster, M. D. Hill, S. K. Reinhardt, and D. A. Wood, "Heterogeneous-race-free Memory Models," in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS. New York, NY, USA: ACM, 2014, pp. 427–440. [Online]. Available: http://doi.acm.org/10.1145/2541940.2541981

[28] B. R. Gaster, D. Hower, and L. Howes, "Hrf-relaxed: Adapting hrf to the complexities of industrial heterogeneous memory models," *ACM Trans. Archit. Code Optim.*, vol. 12, no. 1, pp. 7:1–7:26, Apr. 2015. [Online]. Available: http://doi.acm.org/10.1145/2701618

[29] Y. Bao, Y. Sun, Z. Feric, M. T. Shen, M. Weston, J. L. Abellán, T. Baruah, J. Kim, A. Joshi, and D. Kaeli, “Navisim: A highly accurate gpu simulator for amd rdna gpus,” in *Proceedings of the International Conference on Parallel Architectures and Compilation Techniques*, ser. PACT ’22. New York, NY, USA: Association for Computing Machinery, 2023, p. 333–345. [Online]. Available: https://doi.org/10.1145/3559009.3569666