RETROSPECTIVE: Accel-Sim: An Extensible Simulation Framework for Validated GPU Modeling

Mahmoud Khairy* *Purdue University*
Zhesheng Shen *Intel Cooperation*
Tor M. Aamodt *University of British Columbia*
Timothy G. Rogers *Purdue University*

I. Background and Context

Accel-Sim emerged onto an existing landscape of GPU simulation tools. As a PHD student Tor was recruited to NVIDIA in 2004 and found himself on weekends torn between going to sunny Monterey or writing up his thesis on "helper threads" while otherwise spending every waking hour working as part of the product team developing G80. G80 was a radical new graphics processor unit (GPU) architecture and the first with native support for compute (i.e., CUDA). Amdahl's Law seemed like it might greatly limit the impact of GPU computing. However, conversations with John Edmondson at NVIDIA convinced Tor this "field of dreams" could work out in the long term. After G80 taped out, having earlier tried and failed to convince his NVIDIA manager, John Danskin, to allow him to join the $\sim$3 full time architects then working on GPU compute because (to paraphrase) "we have enough people doing that", Tor decided to take up the question of how best to design these architectures from a new perch as an assistant professor at UBC. However, as GPU Compute was an entirely new class of architecture, tools for studying them were not available. This lead in late 2006 to the decision to begin a team effort developing "GPGPU-Sim" [O5][1], the predecessor to Accel-Sim. Having worked on "helper threads" was useful when trying to emulate thousands of threads. Tim joined Tor's group in September 2010 after leaving his game developer job at Electronic Arts where he had been exposed to GPU shader programming. Over the next several years Tim largely took over GPGPU-Sim development at UBC. During this time Tim also was exposed to industry GPU architecture research via internships at AMD and NVIDIA. By then chip companies had greatly increased their GPU Computing development efforts as compelling software use cases emerged organically in the market. Soon enough Tim found himself as an assistant professor at Purdue.

The Accel-Sim paper's journey proper began in the Winter of 2017 when Mahmoud joined Tim's newly formed research group at Purdue. The GPU simulation landscape by that time was workable but aging. GPGPU-Sim, while significantly updated in 2011 [O1], lacked support for emerging workloads and needed updates to model newer microarchitectures. On the application side, machine learning was exploding as a driver for industry, with specialized instructions and closed-source proprietary libraries featuring hand-tuned machine code becoming the standard. A reliance on emulation-based functional execution of the well-documented virtual ISA PTX made supporting new applications time-consuming at best and impossible at worst. The latter resulted from the fact that many of the most important kernels in machine learning were starting to be written in NVIDIA's sparingly documented, machine ISA known as SASS. From a hardware perspective, ten years of core and memory system development from Fermi to Volta needed to be understood and modeled to produce a credible baseline architecture.

The paper started as an update effort to address the issues in GPGPU-Sim's performance model. In 2018, Rogers' group worked on two papers quantifying and analyzing the virtual ISA and older performance model's effect on correlation and research ideas [O18, O22]. In retrospect, demonstrating and analyzing the problem proved much easier than fixing it. A sizable reverse-engineering and modeling effort pitched in a paper as an update to GPGPU-Sim was rejected by conferences four times in various stages of development, posted on arXiv in 2018, and accepted as a poster in ISPASS 2019 [1]. It became clear during this process that the modeling effort alone was unlikely to have the exposure and impact we hoped for. However, these intermediate versions of the paper and reviewer feedback throughout the process helped improve the quality of what ultimately ended up in Accel-Sim and forced us to focus on the most important factors where the tool would have a long-term impact: ease of supporting new programs and architectures, backed by credible validation.

II. Input from industry and trace-based simulation

More reliably supporting an ever-evolving SASS hardware ISA was an important aspect of the paper that has allowed it to have a sustained impact. Support for SASS had a relatively long history in GPGPU-Sim, dating back to a multi-year development from Tor's group in the early 2010s that produced a functional model for the G80 machine ISA (SM 1.x), released as PTXPlus. Rogers' group spent additional time later in the decade, updating the PTXPlus model to support Pascal (SM 6.x), which was ultimately never finished nor released. A joint effort between Purdue, UBC, and Matt Sinclair's group at Wisconsin attempted to side-step the machine ISA problem for workloads with SASS-only kernels like cuDNN, by identifying equivalent PTX kernels from massive library binaries [O31].

*Currently at AMD

[1]O* denotes a citation from the original paper

These monumental efforts made it apparent that supporting a functional model of the machine ISA was unsustainable and began conversations around a workable alternative, which ended up being trace-based.

Although not a new idea, both generally and in the context of GPUs (a never-released, early version of GPGPU-Sim, predating the development of the CUDA-based PTX functional model, performed trace-based simulation), simulating machine ISA traces accomplished four things: (1) new instructions and architectures did not require a new functional implementation effort, (2) closed-source programs that only contained SASS code could be seamlessly simulated, (3) modeling errors introduced by the high abstraction level of the virtual ISA were eliminated, and (4) the simulator's speed was significantly increased by not emulating the work of tens of thousands of concurrent threads every cycle. However, there were practical roadblocks to developing a trace-based frontend. In particular, contemporary GPUs lacked the dynamic binary instrumentation needed to trace applications at scale. During our modeling effort, we began conversations with Steve Keckler and David Nellans at NVIDIA research about the possibility of NVIDIA developing and releasing such a tool, which ultimately took the form of NVBit [O70]. It is probably safe to say that without NVBit, there would be no Accel-Sim. Developing a binary instrumentation tool ourselves was intractable without access to internal information. Likewise, because NVIDIA has access to internal information, they could have never published something like Accel-Sim. In retrospect, we believe Accel-Sim is a good example of industry taking the ball as far as it can reasonably go, empowering academia to create something that industry would never be allowed to release or publish.

III. Architectural modeling effort

In addition to the frontend changes that make the framework faster and more adaptable, there was also a significant modeling effort to reverse engineer and implement changes to the timing model that reflected contemporary microarchitecture. Some of these changes were obvious from white papers and other public information, while others needed extensive microbenchmarking to understand what had changed. Although they were challenging to publish independently, these careful modeling efforts were still a key contribution of the Accel-Sim paper, which included updates to GPGPU-Sim's performance model, and took significant effort to implement. When drawing the line between GPGPU-Sim and Accel-Sim, we decided to keep GPGPU-Sim what it always has been: a detailed GPU performance model based on functional PTX execution. The paper's changes to GPGPU-Sim updated the model and made the code amenable to operating as a component of the Accel-Sim framework.

To help make it easier to adapt to variants on known parameters, a streamlined tuner was introduced to evaluate the space of unknowns. The tuner makes configuring the existing knobs of the simulator easier. Still, if you need a new knob (as is often the case when significant changes are made), nothing can replace the development effort needed for a completely new or significantly different machine feature. We argue that Accel-Sim makes it easier to focus on this important task.

IV. Extensibility and Validation

One of the key claims of the paper is that the framework makes it much easier to model and validate new baselines as the state of the industry evolves. With a few years of perspective, we have been able to see how well this can be put into practice. Since the official code release of Accel-Sim in the Summer of 2020, two major generations of NVIDIA compute-focused GPUs were released, Ampere and Hopper. Once we got access to the hardware (a difficult task given the current market), we were able to model Ampere with reasonable accuracy with just two weeks of engineering effort.

V. Continuing Legacy and Challenges

Three years after the publication and release of Accel-Sim, we can start taking stock of its impact and evaluate its effectiveness. Accel-Sim has received significant adoption from the GPU architecture community and opened new microarchitecture, and memory systems research directions for the most important contemporary workloads. Despite a rapidly changing hardware and software landscape, the validation and extensibility elements of Accel-Sim have enabled it to adapt and live on. The infrastructure continues to evolve with ongoing efforts from the groups at Purdue, UBC, and others. However, as with any large piece of infrastructure, supporting its continued development has challenges.

The tremendous implementation effort led by Mahmoud and supported by Zhesheng represented a significant portion of their respective Ph.D. and Masters degrees. As both former students have moved on to positions in industry (Mahmoud at AMD Research and Zhesheng at Intel), their ability to contribute to an open-source GPU simulator has limits. As with many academic tools, the continued success of Accel-Sim, and its associated impact on the community, depends on new groups of students and researchers who will never get credit for the original paper.

For Tor and Tim, who are still in their academic positions at UBC and Purdue (Tim was granted tenure in 2022), finding an appropriate way to credit the highly impactful work of maintaining and modernizing useful tools is an ongoing challenge. However, we believe that tooling efforts are a great way for graduate students to learn how things work, make a meaningful contribution to the community, and can help generate new and interesting research problems. Infrastructure work is hard and risky, but Accel-Sim, GPGPU-Sim, and other academic projects have proven the impact can be well worth it in the end.

References

[1] M. Khairy, A. Jain, T. M. Aamodt, and T. G. Rogers, "A detailed model for contemporary gpu memory systems," in *2019 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2019, pp. 141–142.