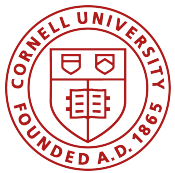
5th Open-Source Computer Architecture Research (OSCAR)

Toward Scalable GPU Modeling: Autonomous Micro-benchmarking with LLM Agents

Xilai Dai Mohamed S. Abdelfattah

Computer Systems Laboratory
Department of Electrical and Computer Engineering
Cornell University



CSL

Contents

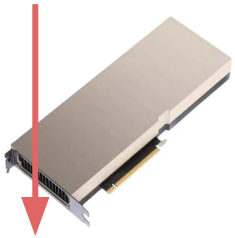
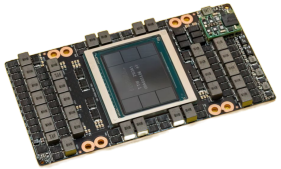
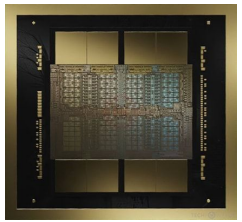
Contents

Limitation on current GPU simulators and modeling

Simulators are only as good as their calibration

- Open-source GPU simulators (Accel-Sim, MGPUSim) underpin architecture research
- Fidelity depends on real-silicon calibration data: throughput, latency, cache behavior, scheduling
- Every new GPU generation adds novel features
- Hand-built micro-benchmarks: deep expertise + weeks to months
- Calibration chronically lags hardware releases

Accel-Sim (GPGPU-sim) is here

V100 (2017) A100 (2020) H100 (2022) B200 (2025)

Challenges in micro-benchmarking

micro-benchmarking is a different task compare to kernel optimization and other coding tasks.

Tasks	Kernel optimization	Micro-benchmarking
Goal	Faster kernel	Trustworthy HW numbers
Ground truth	Correctness check	Almost None
Knowledge	High-level tuning, know pattern	Low-level ISA and HW architecture knowledge
Output	Optimized kernel (program)	HW parameters or build guide

Challenges in micro-benchmarking

1 No ground-truth oracle

A flawed benchmark silently reports plausible-but-wrong numbers — e.g. L1-hit bandwidth mistaken for L2.

→ **Multi-stage validation**

2 Specialized knowledge

Low-level ISA encodings, memory-hierarchy details, and hardware architecture knowledge — less trained by model vendors

→ **Knowledge injection**

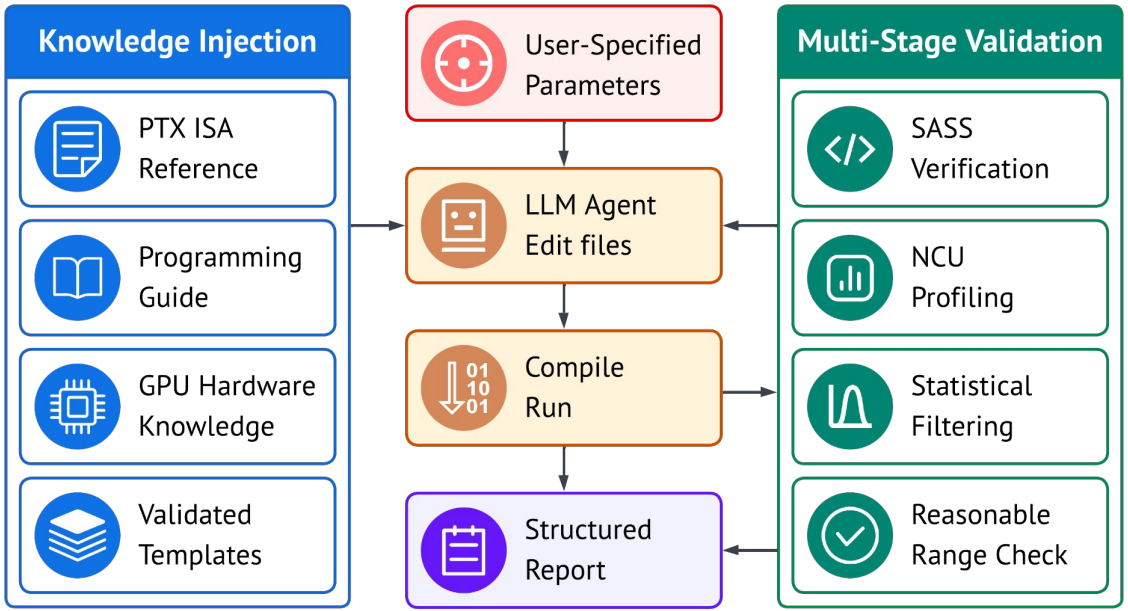
3 Structured output format

A validated set of parameters in a format simulators can directly ingest. OR a building guide of new GPU features for simulator.

→ **Orchestration**

Contents

Auto-Microbench Framework



Agentic framework design for fully autonomous workflow and high-fidelity benchmarking

Knowledge Injection

We provide these four major knowledge sources to LLM agents to help build better benchmarks.

Bridging the gap between general coding knowledge and hardware micro benchmark requirements

PTX ISA/AMD ISA reference

Instruction encodings, semantics, and intrinsics

Validated benchmark templates

Throughput, cache & memory BW + latency, scheduling

General GPU hardware cookbook

Human expertise distilled from prior research

Vendor white paper

Official NVIDIA/AMD whitepaper/blogs, programming guides.

Multi-Stage Validation

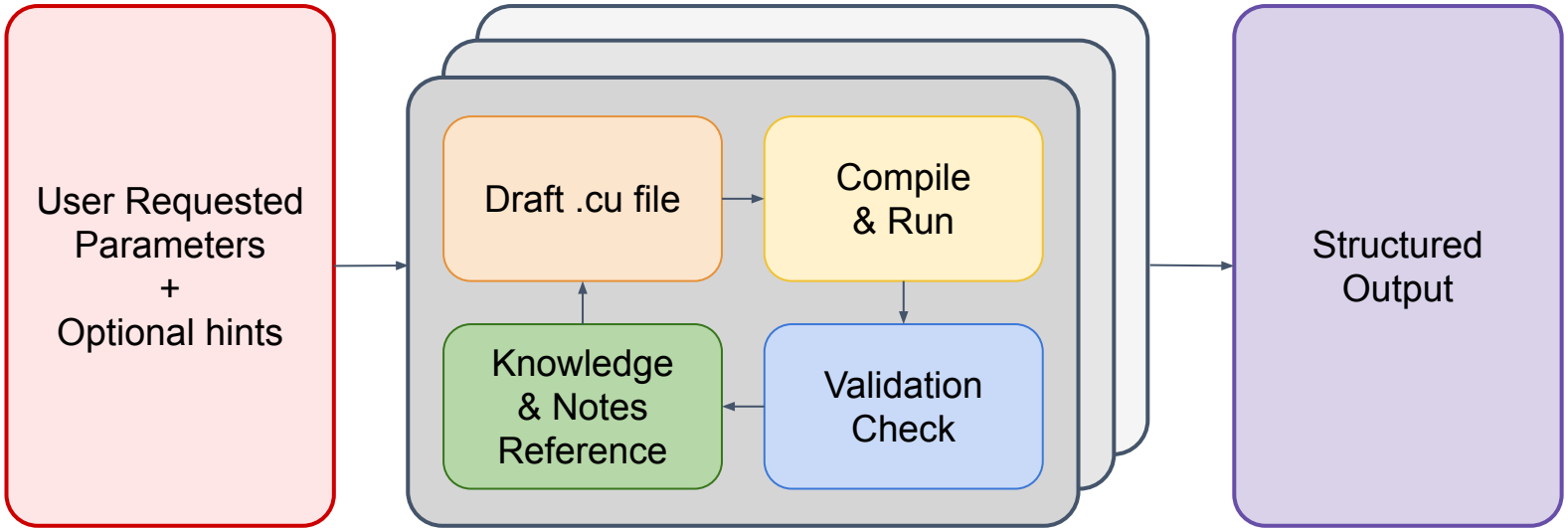
We apply a multi-stage validation steps to ensure the derived benchmarks and measurements as accurate as possible.

1	**SASS verification**	Confirms intended instructions are emitted — not skipped or reordered
2	**NCU hardware profiling**	Check hardware profiling: issue rate, pipeline utilization, cache hit rate, bandwidth…
3	**Statistical stability filtering**	Discards runs with outlier variance across repeated executions
4	**Reasonable range checking**	Out-of-bounds measurements trigger automatic re-implementation
5	**Cross reference checking**	Cross check with measured/known numbers to ensure overall consistency.

Orchestration and Workflow

The agent flows operates in a nested loop:

1. Iterate over all parameters.
2. each parameter optimize and measure multiple times.



Contents

Evaluation Setup

We evaluated Auto-Microbench framework on Blackwell GPUs.

Agent Harness:

- Claude Code (Claude Agent SDK)

LLM Models:

- Claude Opus 4.7
- Deepseek V4 Pro (open weight model)

Target GPUs:

- RTX 5090 (Blackwell, SM120)
- RTX Pro 6000 (Blackwell, SM120)
- B200 (Blackwell, SM100)
- B300 (Blackwell, SM103)

Evaluation Results

The average time for execution is approximate 0.5 - 1.5 hours, with zero human intervention.

We report a portion of the results, mainly on throughput numbers.

Units are MACs/cycle/SM for tensor ops and Instructions/cycle/SM for others

Operations	RTX 5090	RTX Pro 6000	B200	B300
mma_tf32_fp32	128	256	2048	2048
mma_fp16_fp16	512	512	4096	4096
mma_fp16_fp32	256	512	4096	4096
mma_bf16_fp32	256	512	4096	4096
mma_fp8_fp16	1024	1024	8192	8192
mma_fp8_fp32	512	1024	8192	8192
mma_mxfp8_fp32	1024	1024	8192	8192
mma_mxfp6_fp32	1024	1024	8192	8192
mma_mxfp4_fp32	2048	2048	16384	24576
mma_nvfp4_fp32	2048	2048	16384	24576
fma_fp64	1.68	1.68	64	1.68
fma_fp32	128	128	128	128
fma_fp16x2	64	64	64	64
sfu_ex2_fp32	16	16	16	32
sfu_ex2_fp16x2	8	8	8	16
sfu_ex2_bf16x2	8	8	8	16
sfu_sin_fp32	16	16	16	16

Evaluation Results

Professional vs Consumer

(RTX Pro 6000 vs RTX 5090):

- FP32 accumulation tensor throughput is 2x on the Pro

Datacenter shifts

(B300 vs B200):

- 1.5x FP4 throughput for LLM inference workload.
- 2x EXP for attention.
- Largely reduced FP64.

All these are called Blackwell GPUs

Operations	RTX 5090	RTX Pro 6000	B200	B300
mma_tf32_fp32	128	256	2048	2048
mma_fp16_fp16	512	512	4096	4096
mma_fp16_fp32	256	512	4096	4096
mma_bf16_fp32	256	512	4096	4096
mma_fp8_fp16	1024	1024	8192	8192
mma_fp8_fp32	512	1024	8192	8192
mma_mxfp8_fp32	1024	1024	8192	8192
mma_mxfp6_fp32	1024	1024	8192	8192
mma_mxfp4_fp32	2048	2048	16384	24576
mma_nvfp4_fp32	2048	2048	16384	24576
fma_fp64	1.68	1.68	64	1.68
fma_fp32	128	128	128	128
fma_fp16x2	64	64	64	64
sfu_ex2_fp32	16	16	16	32
sfu_ex2_fp16x2	8	8	8	16
sfu_ex2_bf16x2	8	8	8	16
sfu_sin_fp32	16	16	16	16

Summary

Auto Microbench Framework

- Keep simulator calibration data up to date
- Largely reduce human efforts
- Easy to extend to future architectures

Know limitations

- Lack-of-oracle is mitigated but not fully solved.
- Lacks agentic framework to continuously build new simulators and implement new features.

Open-Source Link

https://github.com/abdelfattah-lab/Auto-Microbench