VIRGINIA TECH

Performance Evaluation Methods

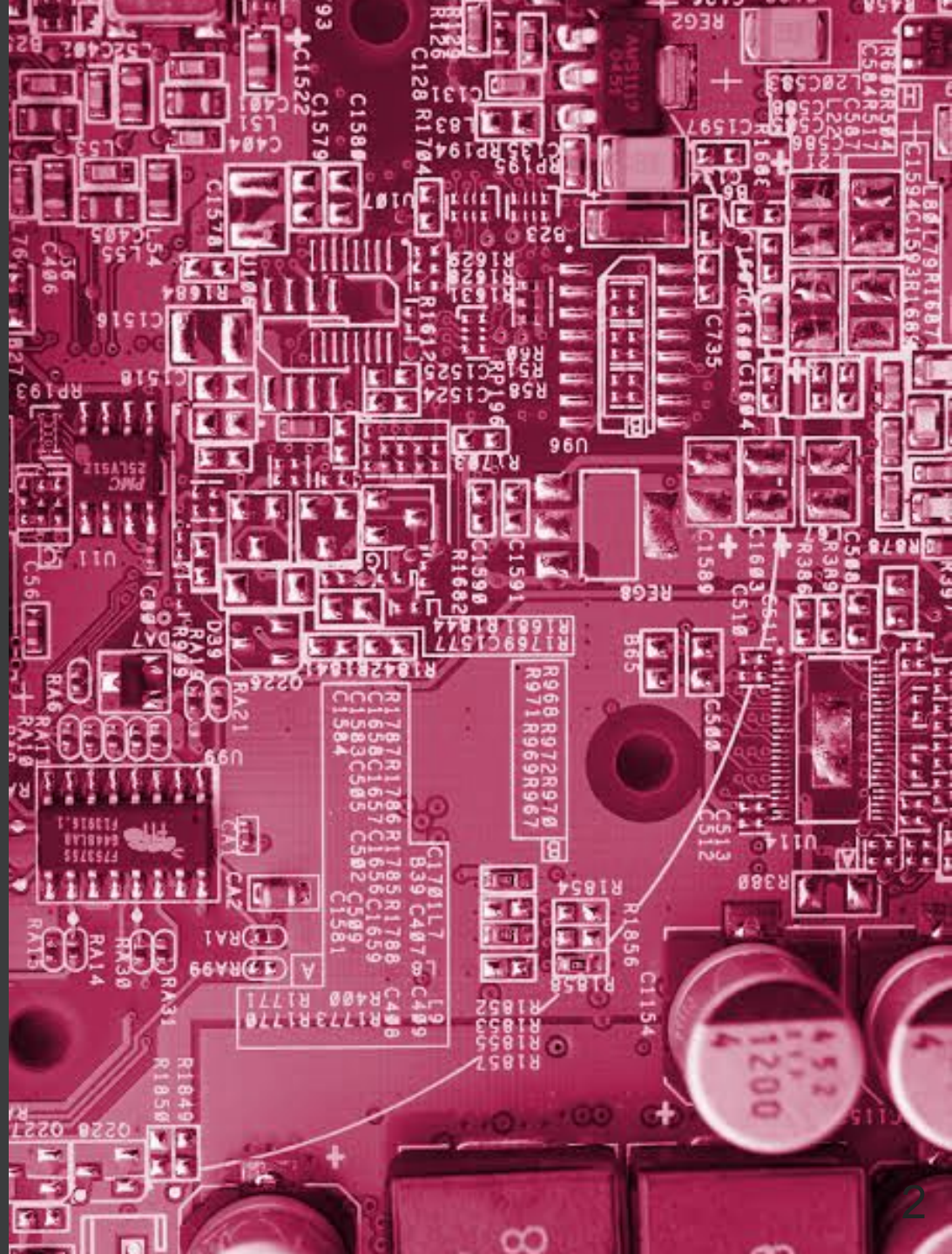
Workgroup #1 Presentation

ECE5504/CS5504 Computer Architecture

September 8th 2026

Group Members

- Aiden Wiehn
- Alexander Kim
- Aubtin Rasouli
- Ayia Ismael
- Gabe Jackson
- Jason Huang
- Rufus Hinton IV
- Timothy Scholtz



Contributions

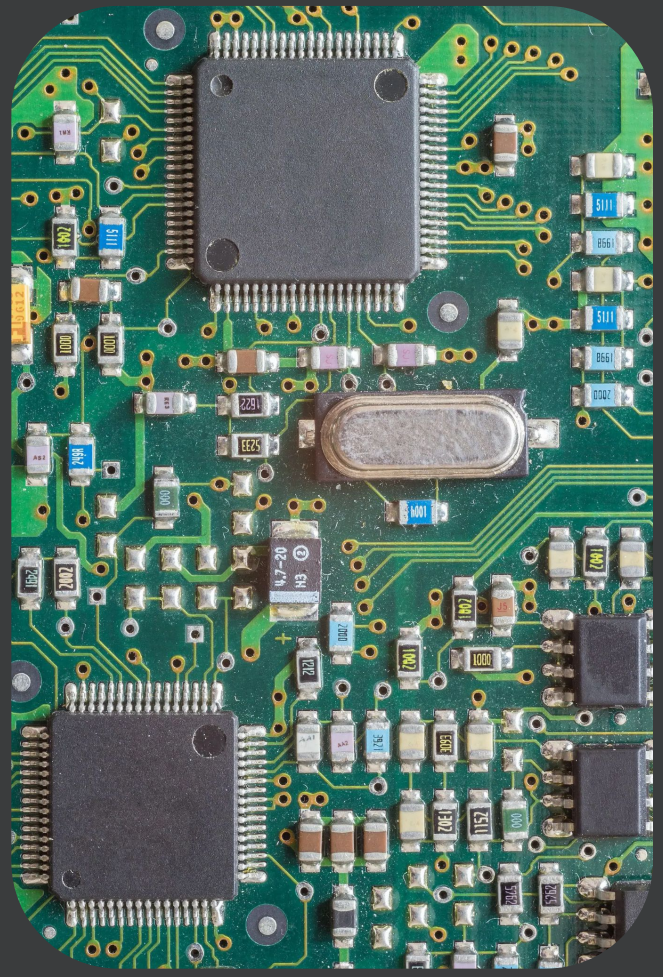
- Paper Categorizations: All
- Phase 1: Aubtin and Aiden
- Phase 2: Alexander and Jason
- Phase 3: Rufus, Ayia
- Phase 4: Gabe, Timothy
- Making/Presenting Slides: Contributions are labeled in the bottom left corner of future slides.

VIRGINIA TECH

Piazza Activity

- A total of 16 papers discussed across 17 piazza posts
  - 13 were papers proposed by us
  - 3 were new papers discussed by the class
    - SPEC CPU: The Next Generation: @35
      - Used in our presentation
    - Photon: A Fine-grained Sampled Simulation Methodology for GPU Workloads: @41
    - GPGPU: Dissecting and Modeling the Architecture of Modern GPU Cores: @43
      - Used in our presentation

Motivation and Background



- To understand complex systems, we must evaluate their performance
- But what you measure, and how you measure, shapes the design choices and overall product

Hanoi Rat Problem - What you measure matters

- Hanoi, 1902: French colonial government wanted fewer rats in the sewers[1]
- Bounty paid per rat tail turned in
- Result: tailless rats in the streets, rat farms on the outskirts
- The metric was optimized. The problem was not solved.

Timothy Scholtz

How this related to Comp. Arch. Intel NetBurst - Pentium

Megahertz Myth (gigahertz myth): refers to the misconception of only using clock rate to compare the performance of different processors.

- **Clock rate** - one number for "speed/performance"
- **Easy to measure, Easy to market**
- **Intel designed for it**. NetBurst stretched the pipeline to 20 stages and later 31:
- Less work per stage, shorter cycle time, but more cycles per second
- **The metric went up, performance didn't**. Deeper pipelines mean costlier mispredictions

The Goal: What is our motivation for evaluating performance?

- We need a meaningful way to benchmark computer performance, and do so consistently.
- Benchmarking has evolved as computer systems and workloads have become more complex:
  - Operating systems
  - Hardware
  - Varying metrics
- We need to determine how we select our benchmarks to meet the above criteria.

A brief history of benchmarking [2]

- Early benchmarks in the 1970s, such as Whetstone and Dhrystone, focused on a single metric--Whetstones/Dhrystones per second.
  - However, benchmarks become outdated as architectures change and workloads differ
- To fix this, new benchmarks were created over time:
  - LINPACK, SPEC CPU, MLPerf
- Instead of focusing on just one metric like CPI, modern benchmarks now focus on multiple metrics:
  - Energy efficiency, reproducibility, performance
- Zablah et al., 2025

Table 1. Historical evolution of classical benchmarks (1970–1990).

Benchmark	Year	Primary Purpose	Strengths	Limitations
Whetstone	1976	Measure floating-point performance	First widely used synthetic benchmark; reproducible	Poor representativeness; compiler dependent
Dhrystone	1984	Measure integer performance	Simplicity; portability; massive popularity (DMIPS)	Easily optimized; not reflective of real workloads
LINPACK	1979	Solve dense systems of linear equations	Representative of scientific workloads; precursor of TOP500	Very specific to linear algebra; limited generality
SPEC	1988	Suite of real-world applications (C, Fortran)	Greater realism; industrial standardization	Costly to execute; administratively complex

Aiden Wiehn

State of the Art [3]

- Same compiler, three benchmarks, & three different verdicts[3]
- Embench 2.0, a consistent benchmark for embedded workloads.

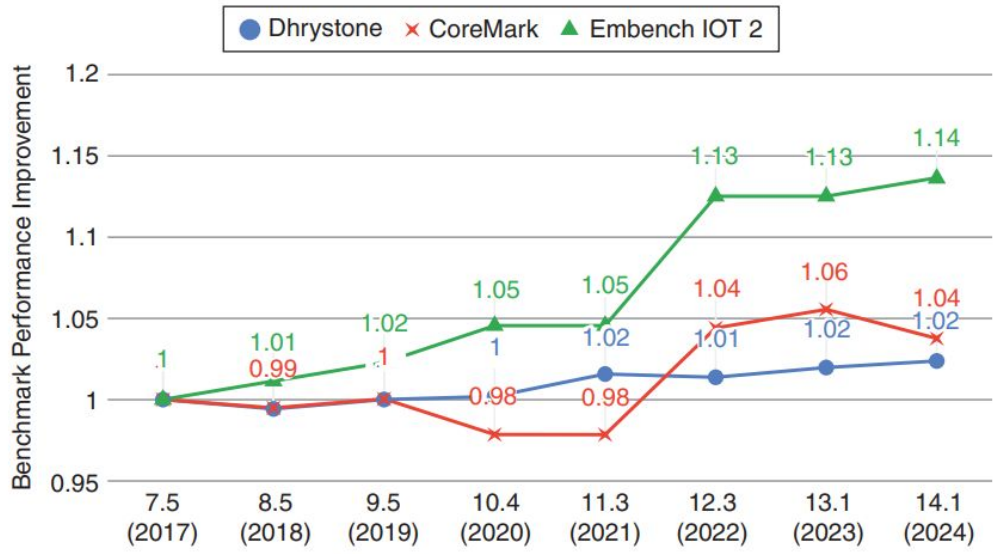


FIGURE 1. Performance for GCC versions on ARM from 2017 to 2024.

Takeaway: Embench 2.0 tracks real progress by using a broad benchmark suite, legacy benchmarks don't.

State of the Art and Future Work [4]

- Same sampling method, very different accuracy per metric.
- Temperature: a dimension the standard method misses

- Takeaway: Sampling gets IPC right, but is ineffective in other metrics.

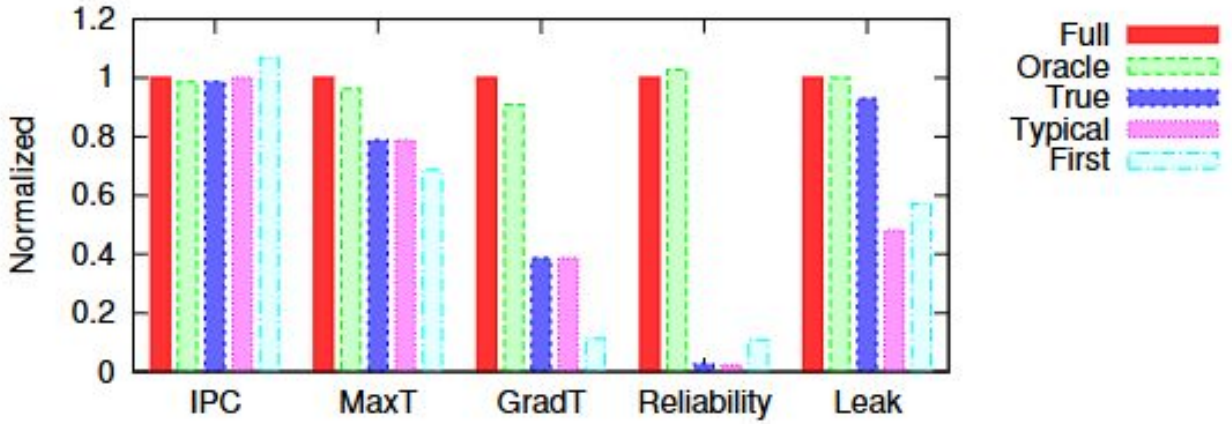


Figure 10. Different Simpoint approaches average results normalized against the full execution.

The Rigor: Methodology & Statistics

- **Motivation:** How can we measure these metrics?
- **History:**
  - Simulators ⟷ ML
  - **Relation to Course Material:** Amdahl's Law
- **State of the Art:**
  - Load Tests
  - Minimizing simulator errors
  - ML

Jason Huang

VIRGINIA TECH

Measuring Experimental Error in MicroProcessors Simulation[5]

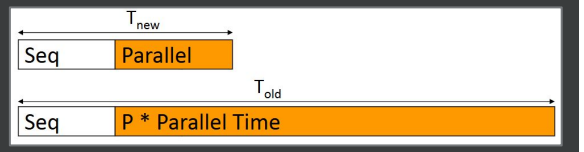
- **Problem:** Unvalidated sims avg. *74.7% initial error* (Desikan et al., ISCA '01)
- **Root Causes:** Bugs, faulty specs, and missing structural details
- **Reality Check:** Even at 2% microbenchmark error, complex workloads hit 18% error

	Alpha 21264	Initial simulator (sim-initial)		Validated simulator (sim-alpha)		SimpleScalar 3.0b (sim-outorder)	
benchmark	IPC	IPC	% error	IPC	% error	IPC	% difference
C-Ca	1.80	0.38	-498.1%	1.87	4.3%	3.17	28.2%
C-Cb	1.87	0.52	-260.4%	1.87	0.6%	3.00	37.8%
C-R	2.65	0.89	-198.4%	2.66	0.3%	3.54	25.2%
C-S1	0.56	0.81	31.2%	0.60	6.4%	0.88	36.1%
C-S2	0.85	0.82	-3.6%	0.86	2.1%	1.33	36.5%
C-S3	0.95	0.87	-8.5%	0.95	0.5%	1.64	42.2%
C-CO	1.75	0.53	-273.6%	1.74	-0.6%	2.05	3.0%
E-I	4.00	3.31	-20.9%	3.99	-0.4%	3.99	-0.4%
E-F	1.01	1.01	-0.1%	1.01	0.2%	1.01	0.2%
E-D1	1.03	1.04	0.3%	1.04	0.4%	1.04	0.4%
E-D2	2.16	2.15	-0.0%	2.15	0.0%	2.21	2.6%
E-D3	2.72	2.99	9.3%	3.07	11.5%	3.19	14.8%
E-D4	2.79	2.89	3.6%	2.80	0.3%	4.00	30.2%
E-D5	3.30	3.23	-2.1%	3.50	5.8%	4.00	17.6%
E-D6	3.11	3.31	6.1%	3.15	1.3%	4.00	22.2%
E-DM1	0.15	1.04	85.7%	0.15	-0.3%	0.15	-0.3%
M-I	2.98	2.39	-24.2%	2.99	0.6%	3.00	0.7%
M-D	1.66	1.25	-32.9%	1.66	0.4%	1.26	-31.1%
M-L2	0.36	0.34	-4.0%	0.35	-0.9%	0.55	35.6%
M-M	0.07	0.07	-8.2%	0.08	4.2%	0.07	-0.3%
M-IP	1.75	0.89	-97.9%	1.76	0.5%	1.22	-43.1%
Mean			74.7%		2.0%		19.5%

Source: Desikan, Burger, & Keckler, ISCA '01 (Comparison of microbenchmark vs. macrobenchmark error rates)

MicroLib: Designing Reliable Performance Evaluations [6]

Source: Gustafson's Law: Lecture 3, Slide 10



- **Problem:** Inconsistent setups/cherry-picked benchmarks skew results
  (Gracia Perez et al., MICRO '04)
- **Reality Check:** SDRAM model drops speedup by *57.9%* (Global History Buffer)
- **Solution:** MicroLib, an open source library model

	Base	TP	VC	SP	Markov	FVC	DBCP	TKVC	TK	CDP	CDPSP	TCP	GHB
1		√	√	√	√	√	√	√	√	√	√		√
2		√	√	√	√	√	√	√	√	√	√		√
3		√	√	√	√	√	√	√	√		√		√
4		√	√	√	√	√	√	√	√		√		√
5		√	√	√	√	√	√	√	√		√		√
6		√	√	√	√	√		√	√		√		√
7		√	√	√	√	√		√	√		√		√
8		√	√	√	√	√		√	√		√		√
9		√	√	√	√	√		√	√		√		√
10		√	√	√		√		√	√		√		√
11		√	√	√		√		√	√		√		√
12		√	√	√		√		√	√		√		√
13		√	√	√				√	√		√		√
14		√	√	√				√	√		√		√
15		√	√	√				√	√		√		√
16		√	√	√				√	√		√		√
17		√		√				√	√		√		√
18		√		√				√	√		√		√
19		√		√				√	√		√		√
20				√					√		√		√
21				√					√		√		√
22				√					√		√		√
23				√									√
24													√
25													√
26													√

Table 6. Which mechanism can be the best with N benchmarks?

Source: Gracia Perez, Mouchard, & Temam, MICRO-37 (Table 6: Impact of benchmark selection and memory latency on performance rankings).

VIRGINIA TECH

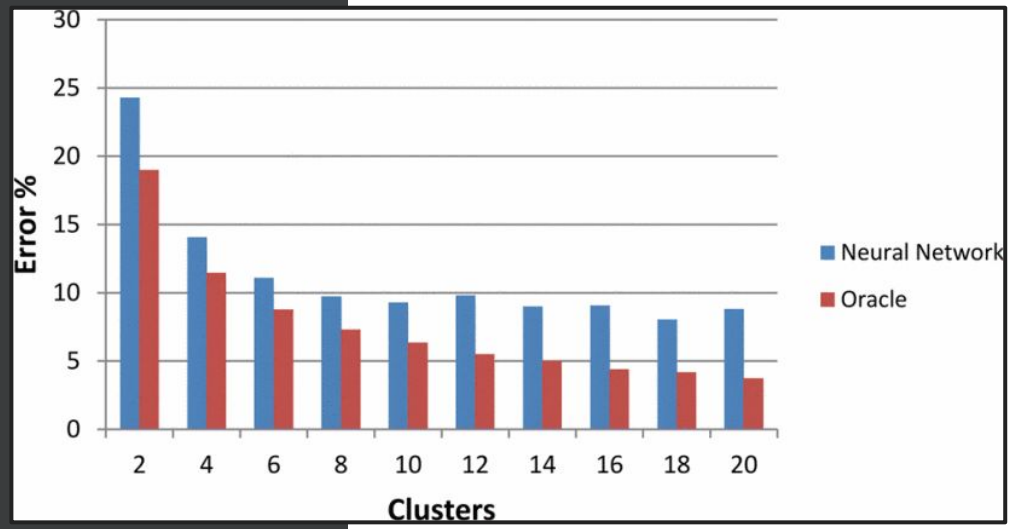
Alexander Kim

GPGPU Performance and Power Estimation Using Machine Learning [7]

- **Summary:**
  - Speed up finding optimal GPU configuration
  - Utilize ML to estimate GPU performance

- **Limitations:**
  - Estimations vs. Accuracy
  - Introducing new hardware generations

- **Interesting Point(s) in Discussion:**
  - Md Tanvir Mahmud Prince: Dynamic Clustering



Source: Wu, G., Greathouse, J. L., Lyashevsky, A., Nuwan Jayasena, & Chiou, D. (2015). GPGPU performance and power estimation using machine learning. IEEE Xplore. https://doi.org/10.1109/hpca.2015.7056063 (Figure 15)

The Tools: *Modeling, Simulation, Emulation & Real Hardware*

- Computer architects use different tools to test and measure system performance
- Modeling and simulation allow for early exploration of designs before hardware is available
- Emulation and real hardware provide more realistic testing
- Each method considers accuracy, flexibility, cost, and evaluation speed

Simulation-Based Performance Evaluation

- Motivation
  - Measure how architecture changes can affect performance before implementation
- History
  - Simulators are used to compare ideas before implementation
  - These tools continue to improve as R&D progresses
- State of the Art
  - gem5[8]
    - Provides configurable software models of various computer parts and systems for comparing performance between design choices
  - TraceRTL[9]
    - Applies trace-driven performance evaluation to RTL, allows for earlier testing before unrelated functional details are complete/verified
- Future Directions
  - Continue to advance model validation, simulation speed, trace accuracy, and processor coverage

ASim [49]	x86	Alpha, x86	UM/MOD/TIM (decoupled timing and functional models)	out-of-order	yes	
Augmint [73]	x86/Unix, x86/Windows	x86	EDr/TD	—	yes	based on MINT and Tango Lite
CMP$im [35]						or functional simu-
COTSon [111]						
Dinero IV [85]						
DRAMSim [84]						with many other like Sim-Alpha, ASE, MARSSx86
ESESC [68]						and thermal mod- ts CPU-GPU sim-
Flexus [43]						cs for full system
gem5 [2]						M5 and GEMS sim-
GEMS [38]	x86/Linux, AMD64-linux,	SPARC, x86	FSys/TIM (decou-pled functional and timing models)	out-of-order	yes	uses Simics for functional simulation, not maintained now
			UM cycle-level (decoupled timing			supports Nvidia's Fermi and

Metrics for Evaluation [11]

1. Accuracy
2. Performance
3. Level of details
4. Ease of development
5. Flexibility
6. User friendliness

Simulation vs Real Hardware

The Benefits and Challenges

PTLsim

gem5 sniper

Software Simulators [11]

gem5, MARSSx86, Multi2Sim, PTLsim, Sniper, and ZSim

FPGA Acceleration [12]

Xilinx XUP FPGA board

Hardware [11]

Intel's Haswell Microarchitecture (Core i7-4770)

Ayia Ismael

The Frontier:
Modern Challenges

Comparing new vs old challenges

- *A survey of cache simulators* (2020)
- *Challenges in Computer Architecture Evaluation* (2003)

Modern research

- *SPEC CPU: The Next Generation* (2026)
- *MLPerf Inference (2020)*

VIRGINIA TECH

Gabe Jackson

A Survey of Cache Simulators (2020) [11]

- Brais et. al. surveyed 28 CPU cache simulators
- Four core findings:
  - Simulator selection matters
  - Modern hardware is difficult to fully model
  - Simulator validators are inconsistent
  - Even hardware isn't a perfect ground truth

What does accuracy even mean when your reference measurement has its own abstractions and uncertainty?

But, how modern are these challenges?

Challenges in Computer Architecture Evaluation (2003)

A Survey of Cache Simulators (2020)

Challenges in Computer Architecture Evaluation (2003) [12]

- Skadron et. al. created a perspective on challenges:
  - Growing hardware complexity
  - Validation of simulators
  - Tools limit researchers; the ‘spotlight effect’

- Core problems persisted; others evolved as hardware complexity increased.

Discussion:

- David's contribution
- Bryan's contribution

Conclusion:

- Hardware is much more complex
- Software validation

Up next: Modern Benchmarks

David Utsis 2 weeks ago

The biggest takeaway I had from this paper was that the need for better simulation environments, benchmarks, and metrics is not a new idea. Even with a focus geared towards designing new tools and measurements, this is a problem that will continue to persist. This is a good problem to have, but as these systems become even more complex, the work needed to validate and create methods to validate these systems will increase. In essence, while the challenges described in this paper may be better defined in the future, they will always hold true.

Bryan Torres 5 days ago

The biggest takeaway is that there isn't a singular best cache simulator. The cache simulator itself could have several different purposes based on the researcher's task or independent goals. A simple functional simulator could just be enough for a researcher who only cares about caches and misses, whereas if they were trying to understand cache access latency, this could require a much more sophisticated cycle level simulator. That additional sophistication not only requires expensive resources and harder development, but it couldn't necessarily result in better experimentation or more accurate results. The authors make a fair point that a simulator that is too simple can produce incorrect conclusions, and to primarily use metrics and parts of the system that are relevant to the idea that they're trying to test. Another insight that I noticed is the inconsistency between the simulators. Different simulators had different benchmarking and validation methods. Some were only validated using cache miss rates, while others had not been validated against another simulator at all. Therefore, validation should not be treated commonly when stating whether or not the simulator is reliable or not.

Gabe Jackson

A Look at Two Modern Benchmarks

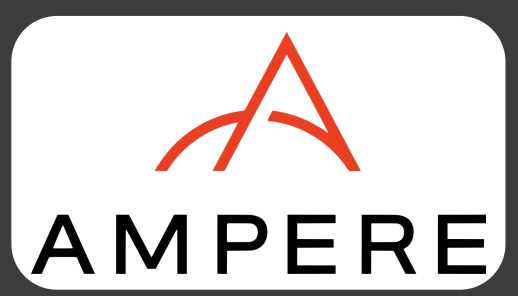
- General Computring : SPEC CPU 2026
- Specialty Benchmark: MLPerf Inference

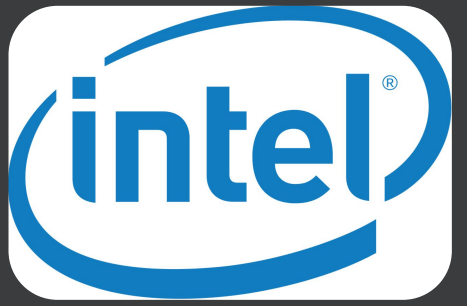
SPEC CPU: THE NEXT GENERATION

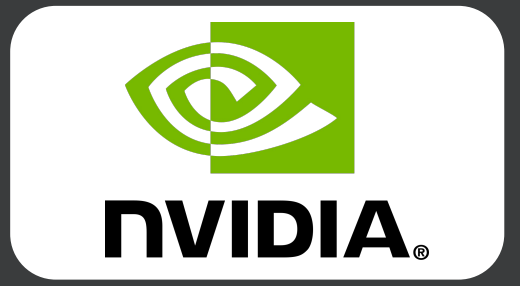
Suggested By: Alexander Burkholder

Mahesh Madhav[1], Allen Lee[2], Andres Mejia[3], Branden Moore[4], Charan Soppadandi[5], Chris Cambly[6], Christoph Müllner[7], Daniel Bowers[8], David Reiner[4], Denis Bakhvalov[9], Di Zhao[1], Duane Voth[4], Feng Xue[1], Frédérique Silber-Chaussumier[10], James Bucek[8], James Southern[11], Jiangning Liu[1], Jim Himer[8], John Henning[8], Kevin Smith[1], Kristen Yang[4], Kunal Kashyap[4], Mason Guy[3], Mat Colgrove[12], Michael Berg[13], Prasad Battini[3], Prasad Joshi[3], Rohit Prasad[3], Shay Bhattacharya[4], Sriyash Caculo[1], Stefan Reimbold[6], Sundar Iyengar[3], Van Smith[4], and Zarko Todorovski[6]

- **Problem**
  - **Comparing Computer Performance**
  - **Performance is a property of the whole stack** - microarchitecture, compiler, OS, memory hierarchy - not of the CPU alone.
  - **Benchmarks decay**. A suite frozen in 2017 slowly stops resembling modern use cases
  - Long-lived suites also get over-optimized: vendors and compiler teams tune specifically for known workloads



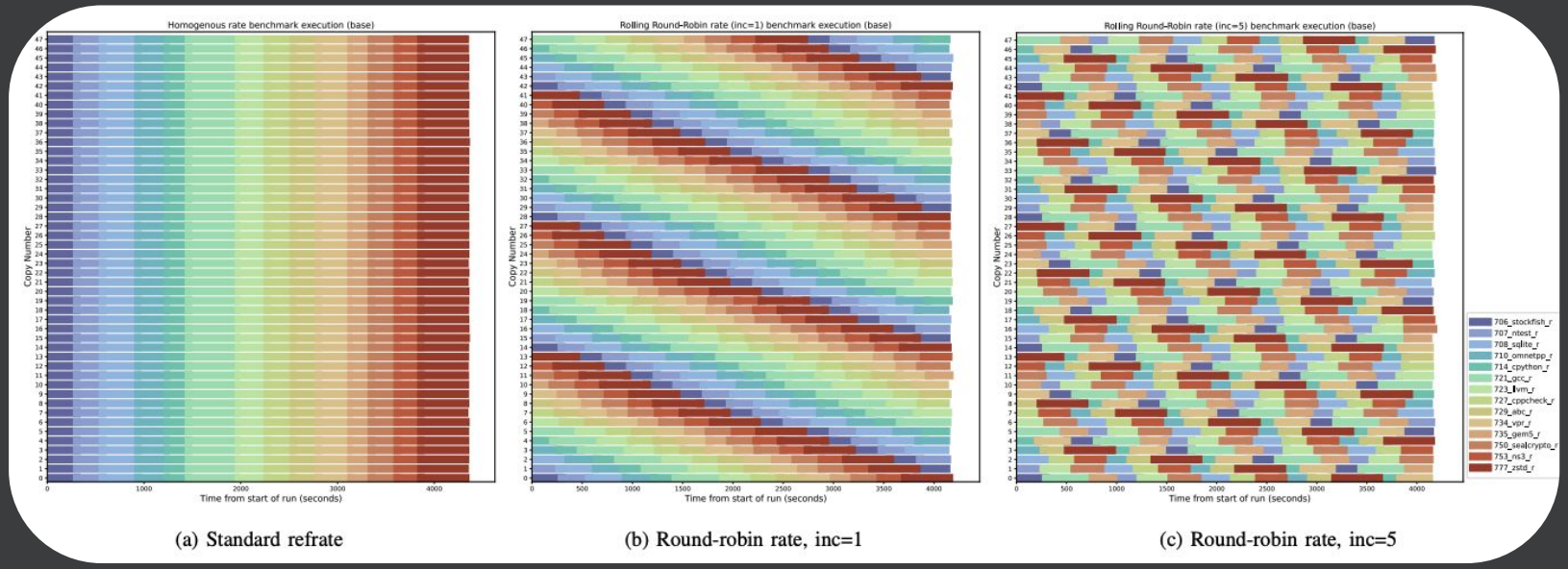




Timothy Scholtz

SPEC CPU: THE NEXT GENERATION

- Summary
  - “Benchmarks shape a field” - David Patterson
  - Began in 2020, run by volunteers from competing organizations
  - Suite **hardened for portability** across architectures, OSes, and compilers
  - **Rolling-Round-Robin Rate**: standardized way to run heterogeneous, multiprogrammed workloads
  - Expanded **multithreaded** and also targets distinct **microarchitectural** profiles
- Limitations
  - Benchmarks can't be developed ahead of the hardware
  - Compute-intensive by design, doesn’t cover I/O, networking, or interactive latency



(a) Standard refrate (b) Round-robin rate, inc=1 (c) Round-robin rate, inc=5

Timothy Scholtz

MLPerf Inference: Standardizing for Modern ML Systems [15]

- Problem
  - Inference runs on a variety of hardware, frameworks, and optimizations
  - Goal - be able to compare real world models on similar metics
- Summary
  - Created an industry standard to compare models
  - Fixed reference models, datasets, and accuracy targets
  - Four scenarios: single-stream, multi-stream, server, offline
  - LoadGen issues queries in realistic arrival patterns
- Limitations
  - Standardization costs realism — no retraining, no custom preprocessing, no caching, though all are common in production
  - Vendor downloads the benchmark suite, runs on own systems, requires vendors to be honest with reported numbers

Timothy Scholtz

MLPerf Inference: Standardizing for Modern ML Systems

Vijay Janapa Reddi,* Christine Cheng,† David Kanter,‡ Peter Mattson,§
Guenther Schmuelling,¶ Carole-Jean Wu,‖ Brian Anderson,§ Maximilien Breughe,**
Mark Charlebois,†† William Chou,†† Ramesh Chukka,† Cody Coleman,‡‡ Sam Davis,[x]
Pan Deng,[xxvii] Greg Diamos,[xi] Jared Duke,§ Dave Fick,[xii] J. Scott Gardner,[xiii] Itay Hubara,[xiv]
Sachin Idgunji,** Thomas B. Jablin,§ Jeff Jiao,[xv] Tom St. John,[xxii] Pankaj Kanwar,§
David Lee,[xvi] Jeffery Liao,[xvii] Anton Lokhmotov,[xviii] Francisco Massa,‖ Peng Meng,[xxvii]
Paulius Micikevicius,** Colin Osborne,[xix] Gennady Pekhimenko,[xx] Arun Tejusve Raghunath Rajan,†
Dilip Sequeira,** Ashish Sirasao,[xxi] Fei Sun,[xxiii] Hanlin Tang,† Michael Thomson,[xxiv]
Frank Wei,[xxv] Ephrem Wu,[xxi] Lingjie Xu,[xxviii] Koichi Yamada,† Bing Yu,[xvi]
George Yuan,** Aaron Zhong,[xv] Peizhao Zhang,‖ Yuchen Zhou[xxvi]

*Harvard University †Intel ‡Real World Insights §Google ¶Microsoft ‖Facebook **NVIDIA ††Qualcomm
‡‡Stanford University [x]Myrtle [xi]Landing AI [xii]Mythic [xiii]Advantage Engineering [xiv]Habana Labs
[xv]Alibaba T-Head [xvi]Facebook (formerly at MediaTek) [xvii]OPPO (formerly at Synopsys) [xviii]dividiti [xix]Arm
[xx]University of Toronto & Vector Institute [xxi]Xilinx [xxii]Tesla [xxiii]Alibaba (formerly at Facebook)
[xxiv]Centaur Technology [xxv]Alibaba Cloud [xxvi]General Motors [xxvii]Tencent [xxviii]Biren Research (formerly at Alibaba)

MLPerf

ML Commons

Better Machine Learning for Everyone

MLC

Timothy Scholtz

Conclusion

- Phase 1: Benchmarking tools have evolved over the years to accommodate for changing architecture, operating systems, metrics, and more.
- Phase 2: There's a wide variety of options to measure performance ranging from HW simulators all the way to ML classifiers, but each have tradeoffs that should be considered.
- Phase 3: Computer architects have many tools, like gem5, at their disposal to evaluate performance, but determining which tool to use requires consideration of multiple criterion.
- Phase 4: As technology continues to progress and become more complex and versatile, we need to continuously update what and how we evaluate systems.

Aiden Wiehn

Q & A?

References

1. Vann, M. (n.d.). Feral atlas. Retrieved September 9, 2026, from https://feralatlas.supdigital.org/poster/colonial-sewers-led-to-more-rats
2. Zablah, I., Sosa-Díaz, L., & Garcia-Loureiro, A. (2025). Relevance and evolution of benchmarking in computer systems: A comprehensive historical and conceptual review. Computers, 14(12), 516. https://doi.org/10.3390/computers14120516
3. Patterson, D., Bennett, J., Bennett, M., & Chelin, H. (2025, May). Embench IOT 2.0 and DSP 1.0: Modern Embedded Computing Benchmarks. IEEE. IEEE Xplore. https://ieeexplore.ieee.org/document/10970203
4. Mesa-Martinez, F. J., Ardestani, E. K., & Renau, J. (2010). Characterizing processor thermal behavior. ACM Digital Library, 38(1), 193–204. https://doi.org/10.1145/1735970.1736043
5. Desikan, R., Burger, D., & Keckler, S. W. (2001). Measuring experimental error in microprocessor simulation. Proceedings of the 28th Annual International Symposium on Computer Architecture - ISCA '01, 266–277. https://doi.org/10.1145/379240.565338
6. Gracia, D., Lri, P., Sud, P., Mouchard, G., & Lri, O. (2004). MicroLib: A case for the quantitative comparison of micro-architecture mechanisms. https://www.eecg.utoronto.ca/~moshovos/ACA08/readings/microlib.pdf
7. Wu, G., Greathouse, J. L., Lyashevsky, A., Nuwan Jayasena, & Chiou, D. (2015). GPGPU performance and power estimation using machine learning. IEEE Xplore. https://doi.org/10.1109/hpca.2015.7056063
8. Loew-Power, J., Akram, A., Alian, M., & Chen, L. (2007). The gem5 Simulator: Version 20.0+. arXiv. https://arxiv.org/pdf/2007.03152
9. Zhang, Z., Xu, Y., Wang, S., & Tang, D. (2026). TraceRTL: Agile Performance Evaluation for Microarchitecture Exploration. HPCA. IEEE Xplore. https://ieeexplore.ieee.org/document/11408626
10. Tan, Z. et. al. (2010). A case for FAME: FPGA architecture model execution. ACM Digital Library, 290–301. https://doi.org/10.1145/1815961.1815999
11. Akram, A., & Sawalha, L. (2019). A survey of computer architecture simulation techniques and tools. IEEE Access, 7, 78120–78145. https://doi.org/10.1109/access.2019.2917698
12. Skadron et. al. (2003). Challenges in computer architecture evaluation. Computer, 36(8), 30–36. https://doi.org/10.1109/mc.2003.1220579
13. Brais, H., Kalayappan, R., & Panda, P. R. (2020). A survey of cache simulators. ACM Computing Surveys, 53(1), 1–32. https://doi.org/10.1145/3372393
14. Madhav, M. et. al. (2026). SPEC CPU: THE NEXT GENERATION. Arxiv. https://arxiv.org/pdf/2605.01575
15. Reddi et. al. (2020). MLPerf inference benchmark. https://arxiv.org/pdf/1911.02549

Outline (25 minutes total)

Title page (15 seconds) Jason

Contributions of each individual in the group (required) e.g., leading topic X; paper categorization; making slide #; presenting slide # (15

Count the activities on piazza. How many papers are discussed? How many posts?

(45 seconds) Aiden

Motivation and Background (1 minute) -Why the topic is important - Timothy

Phase 1: The Goal (Metrics & Benchmarks) (5 minutes) - Aubtin, Aiden Wiehn

Phase 2: The Rigor (Methodology & Statistics) (5 minutes) - Alexander Kim, Jason Huang

Connections to other course material (1 minute)- Person

Phase 3: The Tools (Modeling, Simulation, Emulation, Real Hardware) (5 minutes) - Rufus, Ayia

Phase 4: The Frontier (Modern Challenges) (5 minutes) - Gabe, Timothy

Conclusions (45 seconds) - everyone contribute as they go, Aubtin can organize it after the fact

References - Gabe

Q/A (5 minutes) - Alexander Kim, Aiya, Everyone