Memory Systems and Memory-Centric Computing

Lecture 4.1: Memory-Centric Computing III

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu

18 July 2024

HiPEAC ACACES Summer School 2024

SAFARI

ETH zürich

Agenda For Today

- Memory Systems and Memory-Centric Computing
  - July 15-19, 2024

- Topic 1: Memory Trends, Challenges, Opportunities, Basics
- Topic 2: Memory-Centric Computing
- Topic 3: Memory Robustness: RowHammer, RowPress & Beyond
- Topic 4: Machine Learning Driven Memory Systems
- Topic 5 (another course): Architectures for Genomics and ML
- Topic 6 (unlikely): Non-Volatile Memories and Storage
- Topic 7 (unlikely): Memory Latency, Predictability & QoS

- Major Overview Reading:
  - Mutlu et al., "A Modern Primer on Processing in Memory," Book Chapter on Emerging Computing and Devices, 2022.

Processing in Memory: Two Approaches

1. Processing **using** Memory
2. Processing **near** Memory

Processing in Memory: Two Approaches

1. Processing using Memory
2. **Processing near Memory**

PIM Review and Open Problems

A Modern Primer on Processing in Memory

Onur Mutlu[a,b], Saugata Ghose[b,c], Juan Gómez-Luna[a], Rachata Ausavarungnirun[d]

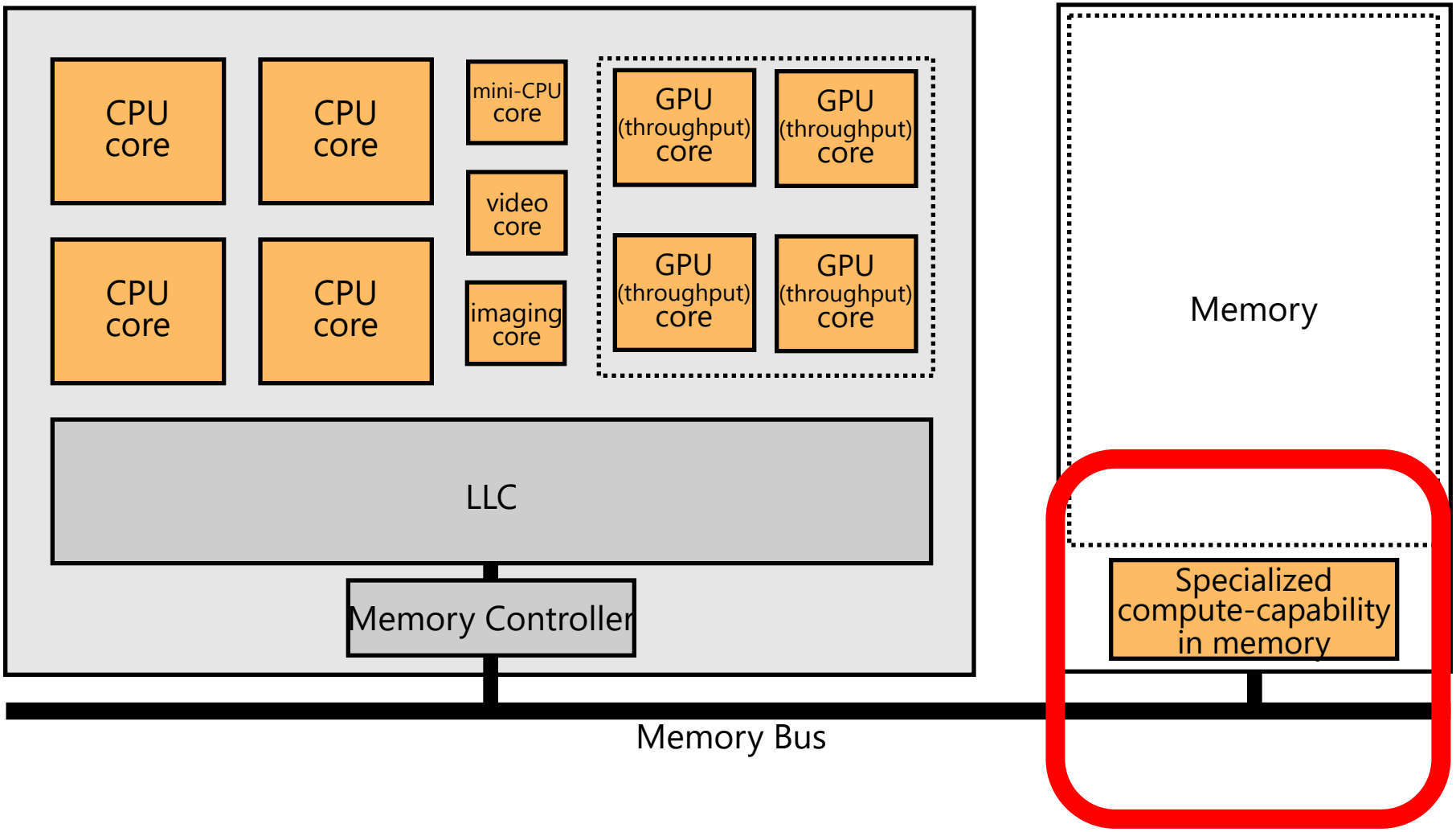
SAFARI Research Group

[a] *ETH Zürich*
[b] *Carnegie Mellon University*
[c] *University of Illinois at Urbana-Champaign*
[d] *King Mongkut's University of Technology North Bangkok*

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun,
"A Modern Primer on Processing in Memory"
Invited Book Chapter in ***Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann***, Springer, to be published in 2021.

https://arxiv.org/pdf/2012.03112.pdf

Mindset: Memory as an Accelerator



Memory similar to a "conventional" accelerator

In-Storage Acceleration of Genome Analytics

In-Storage Genomic Data Filtering [ASPLOS 2022]

- Nika Mansouri Ghiasi, Jisung Park, Harun Mustafa, Jeremie Kim, Ataberk Olgun, Arvid Gollwitzer, Damla Senol Cali, Can Firtina, Haiyu Mao, Nour Almadhoun Alserr, Rachata Ausavarungnirun, Nandita Vijaykumar, Mohammed Alser, and Onur Mutlu,
  "GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis"
  *Proceedings of the 27th International Conference on Architectural Support for Programming Languages and Operating Systems (**ASPLOS**)*, Virtual, February-March 2022.
  [Lightning Talk Slides (pptx) (pdf)]
  [Lightning Talk Video (90 seconds)]

GenStore: A High-Performance In-Storage Processing System for Genome Sequence Analysis

Nika Mansouri Ghiasi[1] Jisung Park[1] Harun Mustafa[1] Jeremie Kim[1] Ataberk Olgun[1]
Arvid Gollwitzer[1] Damla Senol Cali[2] Can Firtina[1] Haiyu Mao[1] Nour Almadhoun Alserr[1]
Rachata Ausavarungnirun[3] Nandita Vijaykumar[4] Mohammed Alser[1] Onur Mutlu[1]

[1]ETH Zürich [2]Bionano Genomics [3]KMUTNB [4]University of Toronto

In-Storage Metagenomics [ISCA 2024]

- Nika Mansouri Ghiasi, Mohammad Sadrosadati, Harun Mustafa, Arvid Gollwitzer, Can Firtina, Julien Eudine, Haiyu Mao, Joel Lindegger, Meryem Banu Cavlak, Mohammed Alser, Jisung Park, and Onur Mutlu,
  "MegIS: High-Performance and Low-Cost Metagenomic Analysis with In-Storage Processing"
  *Proceedings of the 51st Annual International Symposium on Computer Architecture (**ISCA**)*, Buenos Aires, Argentina, July 2024.
  [Slides (pptx) (pdf)]
  [arXiv version]

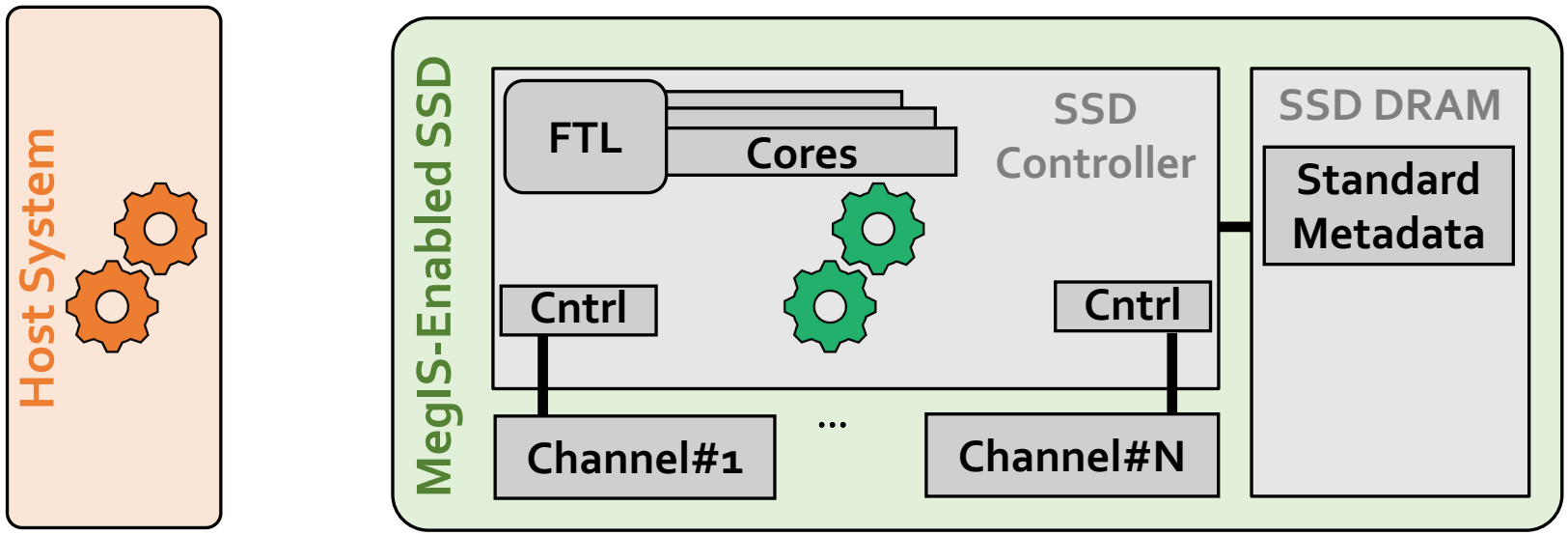
MegIS: High-Performance, Energy-Efficient, and Low-Cost Metagenomic Analysis with In-Storage Processing

Nika Mansouri Ghiasi[1] Mohammad Sadrosadati[1] Harun Mustafa[1] Arvid Gollwitzer[1]
Can Firtina[1] Julien Eudine[1] Haiyu Mao[1] Joël Lindegger[1] Meryem Banu Cavlak[1]
Mohammed Alser[1] Jisung Park[2] Onur Mutlu[1]
[1]ETH Zürich [2]POSTECH

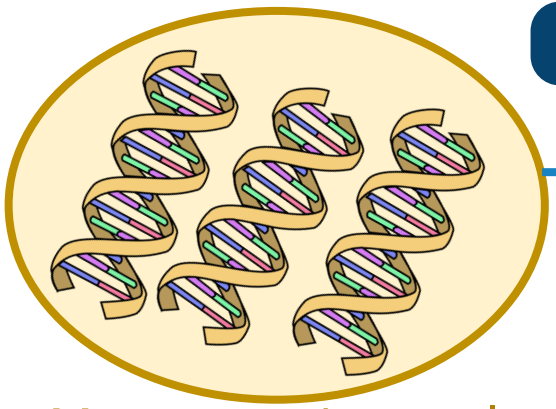
https://arxiv.org/pdf/2406.19113

MegIS: Metagenomics In-Storage

- First in-storage system for *end-to-end* metagenomic analysis
- **Idea:** Cooperative in-storage processing for metagenomic analysis
  - Hardware/software co-design between



MegIS's Steps



Metagenomic sample with species that are **not known** in advance

A large database containing information on **many species**

Step 1

Preparation of Input Queries

Query K-mers: GCTCA, CTCAT, TCATG, ...

Step 2

Presence/Absence Identification

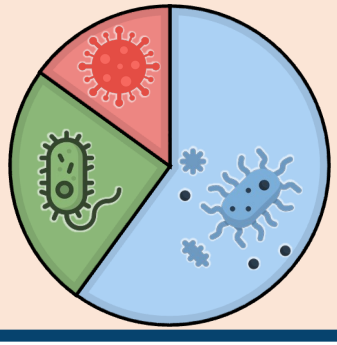
V. cholerae

SARS-CoV-2

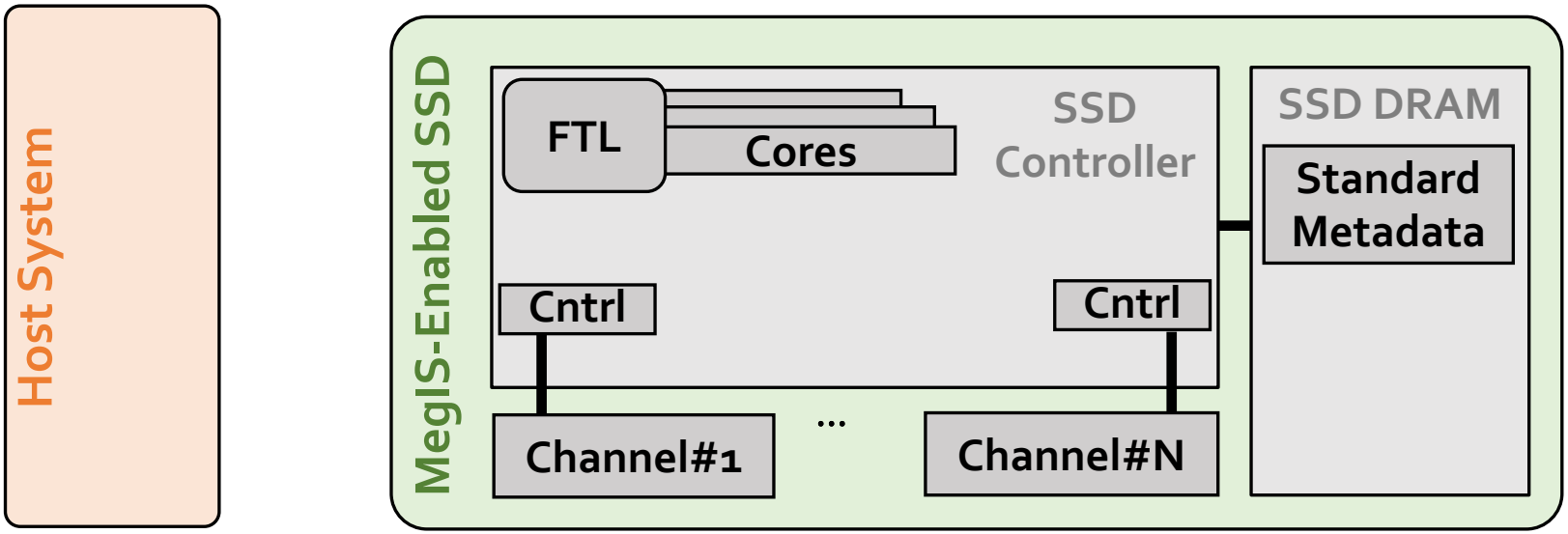
E. coli

Step 3

Abundance Estimation



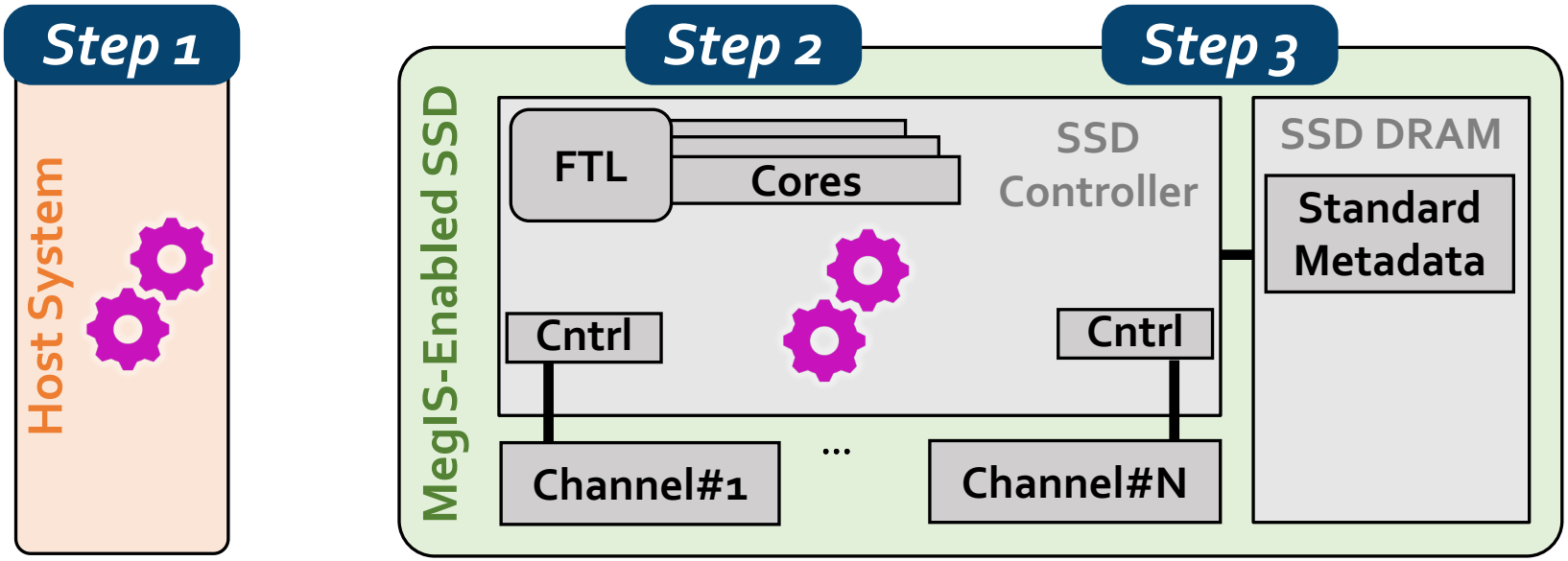
MegIS Hardware-Software Co-Design



MegIS Hardware-Software Co-Design

Task partitioning and mapping

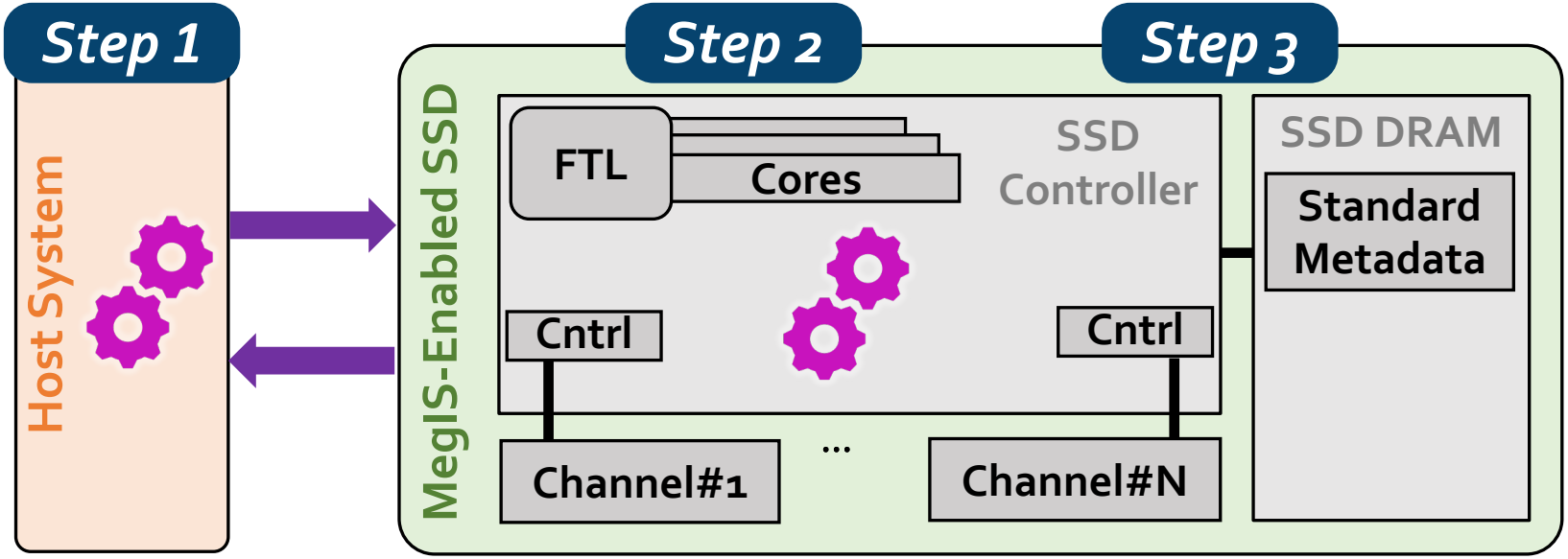
- *Each step executes in its most suitable system*



MegIS Hardware-Software Co-Design

Task partitioning and mapping
- *Each step executes in its most suitable system*

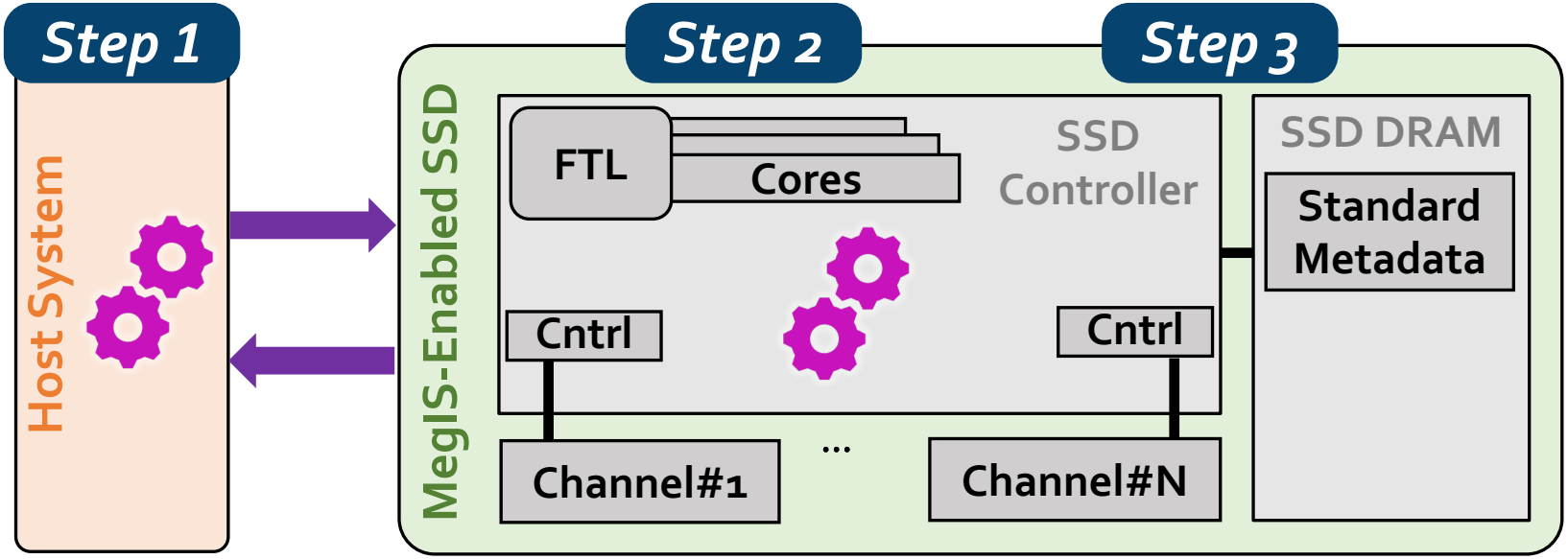
Data/computation flow coordination
- *Reduce communication overhead*
- *Reduce #writes to flash chips*



MegIS Hardware-Software Co-Design

Task partitioning and mapping
- *Each step executes in its most suitable system*

Data/computation flow coordination
- *Reduce communication overhead*
- *Reduce #writes to flash chips*

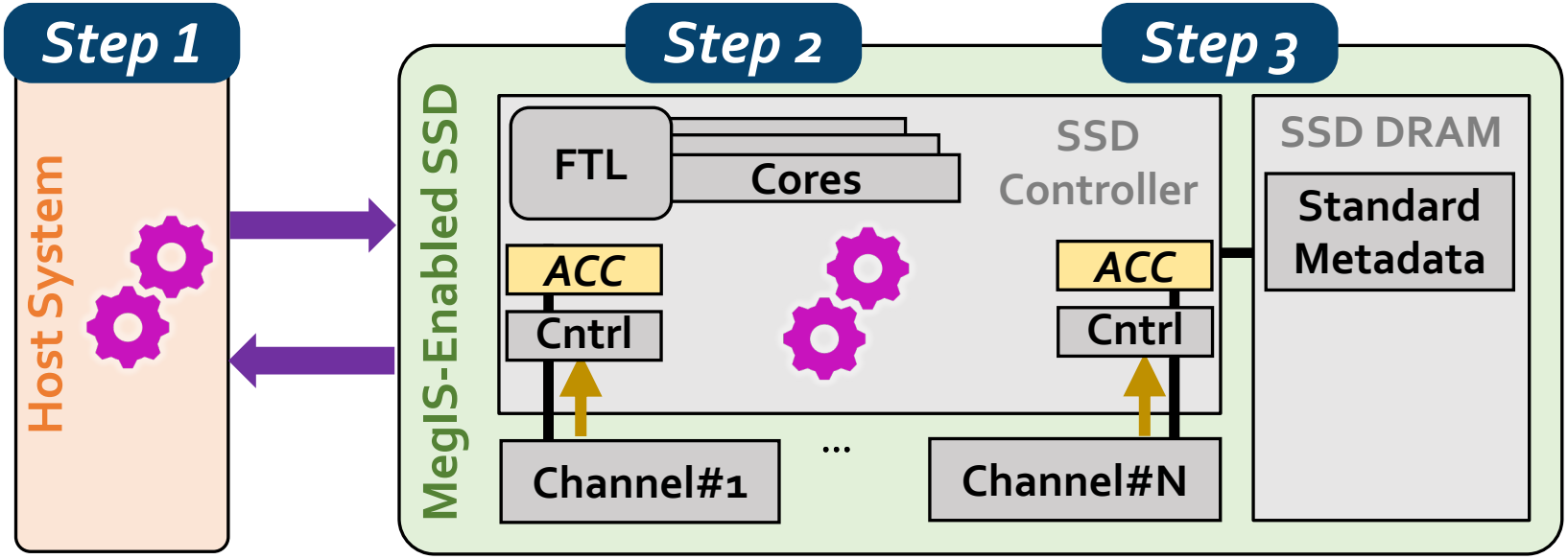


Storage-aware algorithms
- *Enable efficient access patterns to the SSD*

MegIS Hardware-Software Co-Design

Task partitioning and mapping
- *Each step executes in its most suitable system*

Data/computation flow coordination
- *Reduce communication overhead*
- *Reduce #writes to flash chips*



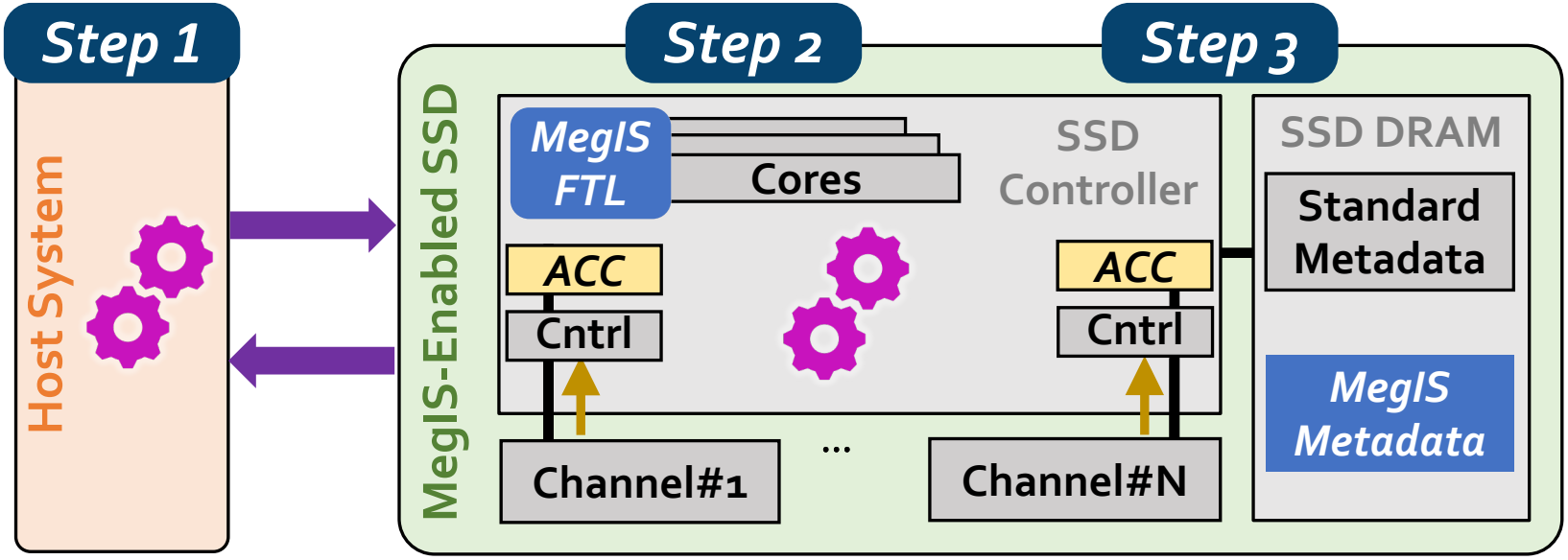
Storage-aware algorithms
- *Enable efficient access patterns to the SSD*

Lightweight in-storage accelerators
- *Minimize SRAM/DRAM buffer spaces needed inside the SSD*

MegIS Hardware-Software Co-Design

Task partitioning and mapping
- *Each step executes in its most suitable system*

Data/computation flow coordination
- *Reduce communication overhead*
- *Reduce #writes to flash chips*



Storage-aware algorithms
- *Enable efficient access patterns to the SSD*

Lightweight in-storage accelerators
- *Minimize SRAM/DRAM buffer spaces needed inside the SSD*

Data mapping scheme and Flash Translation Layer (FTL)
- *Specialize to the characteristics of metagenomic analysis*
- *Leverage the SSD's full internal bandwidth*

Evaluation: Methodology Overview

Performance, Energy, and Power Analysis

Hardware Components

- Synthesized Verilog model for the in-storage accelerators
- MQSim *[Tavakkol+, FAST'18]* for SSD's internal operations
- Ramulator *[Kim+, CAL'15]* for SSD's internal DRAM

Software Components

Measure on a real system:

- AMD® EPYC® CPU with 128 physical cores
- 1-TB DRAM

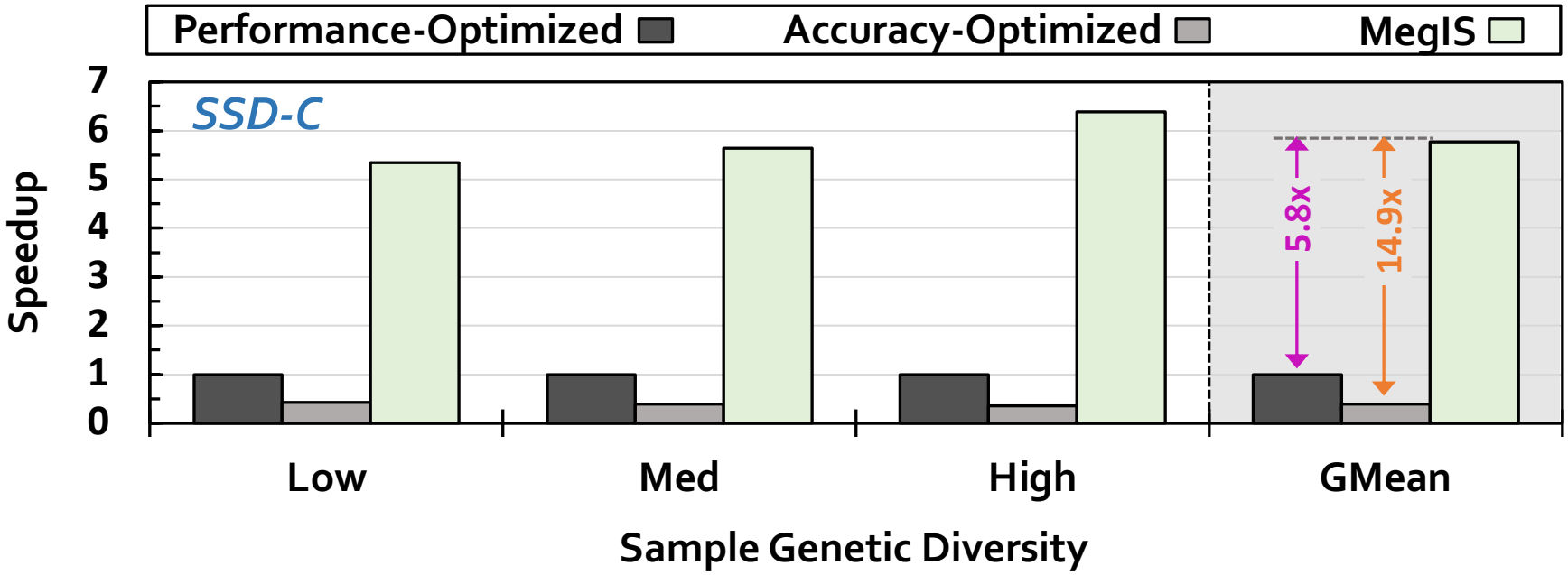
Baseline Comparison Points

- **Performance-optimized software**, Kraken2 *[Genome Biology'19]*
- **Accuracy-optimized software**, Metalign *[Genome Biology'20]*
- **PIM hardware-accelerated tool** (using processing-in-memory), Sieve *[ISCA'21]*

SSD Configurations

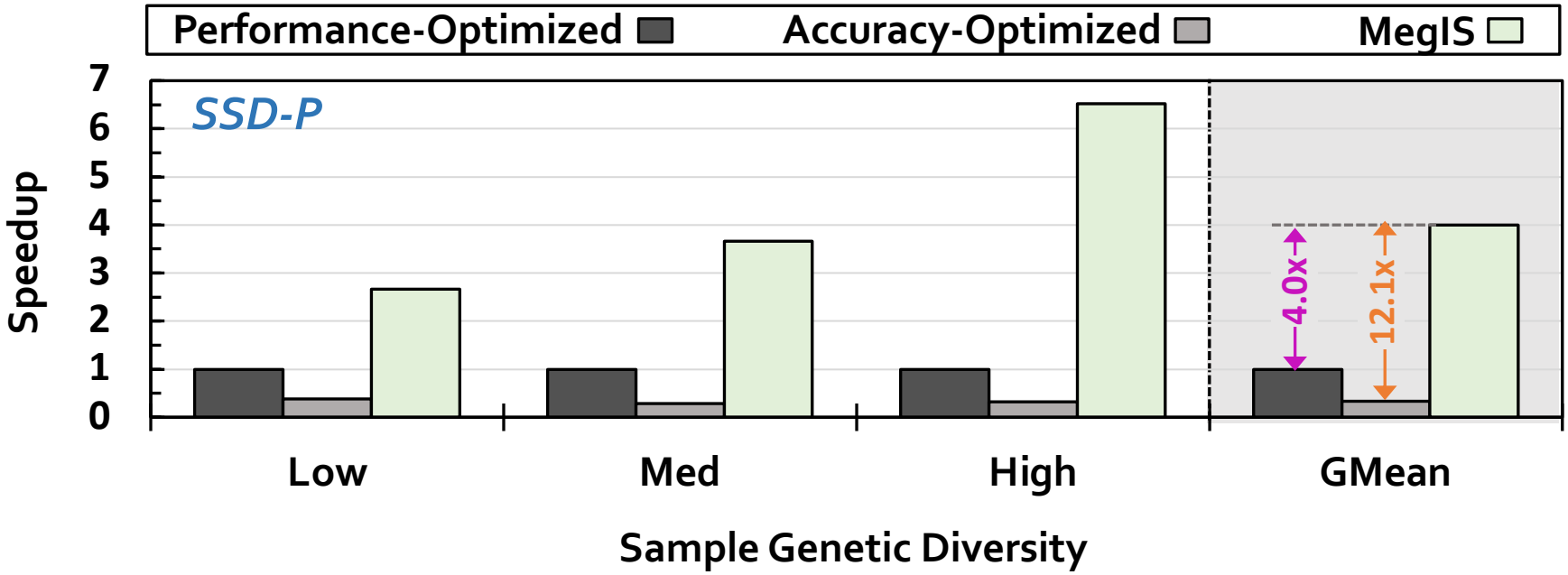
- **SSD-C:** with SATA3 interface (0.5 GB/s sequential read bandwidth)
- **SSD-P:** with PCIe Gen4 interface (7 GB/s sequential read bandwidth)

Evaluation: Speedup over the Software Baselines



MegIS provides significant speedup over both Performance-Optimized and Accuracy-Optimized baselines

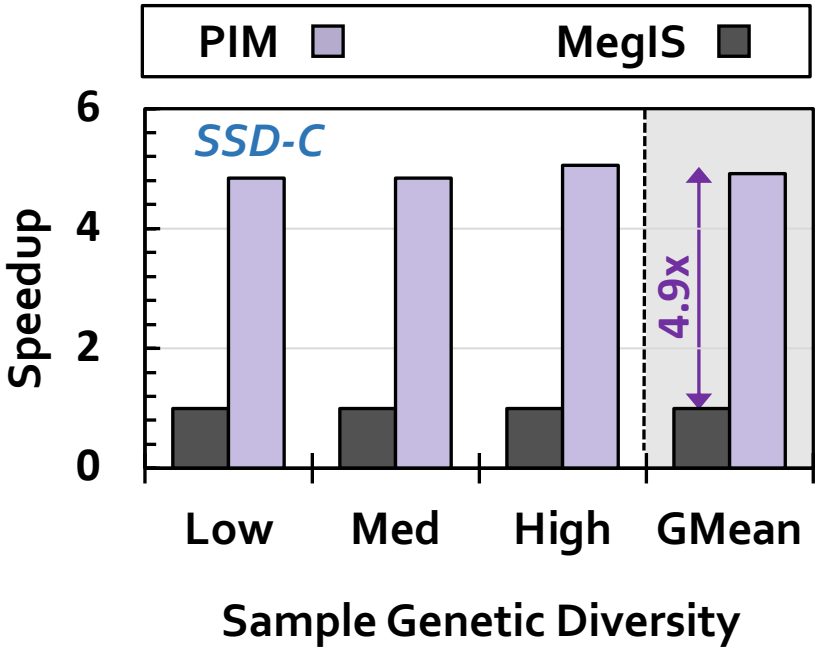
Evaluation: Speedup over the Software Baselines

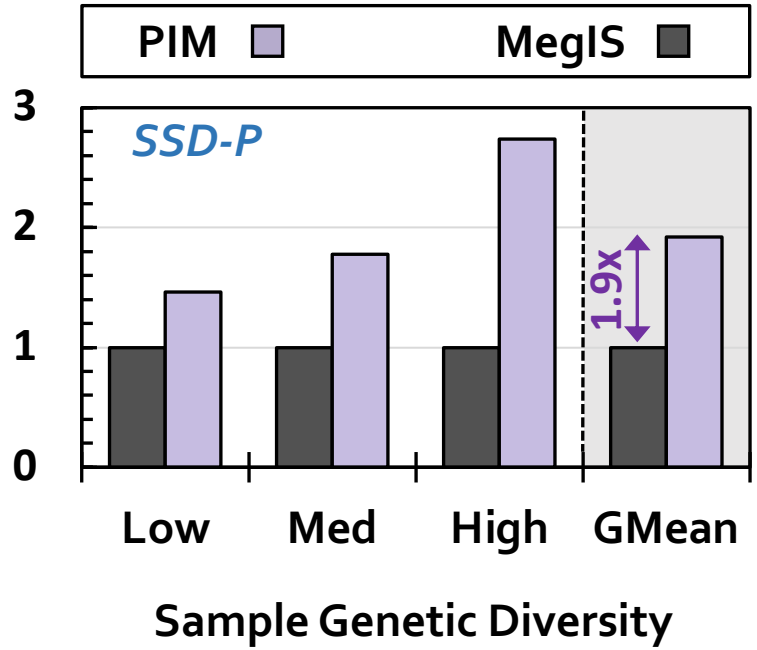


MegIS provides significant speedup over both Performance-Optimized and Accuracy-Optimized baselines

MegIS improves performance on both cost-optimized and performance-optimized SSDs

Evaluation: Speedup over the PIM Baseline

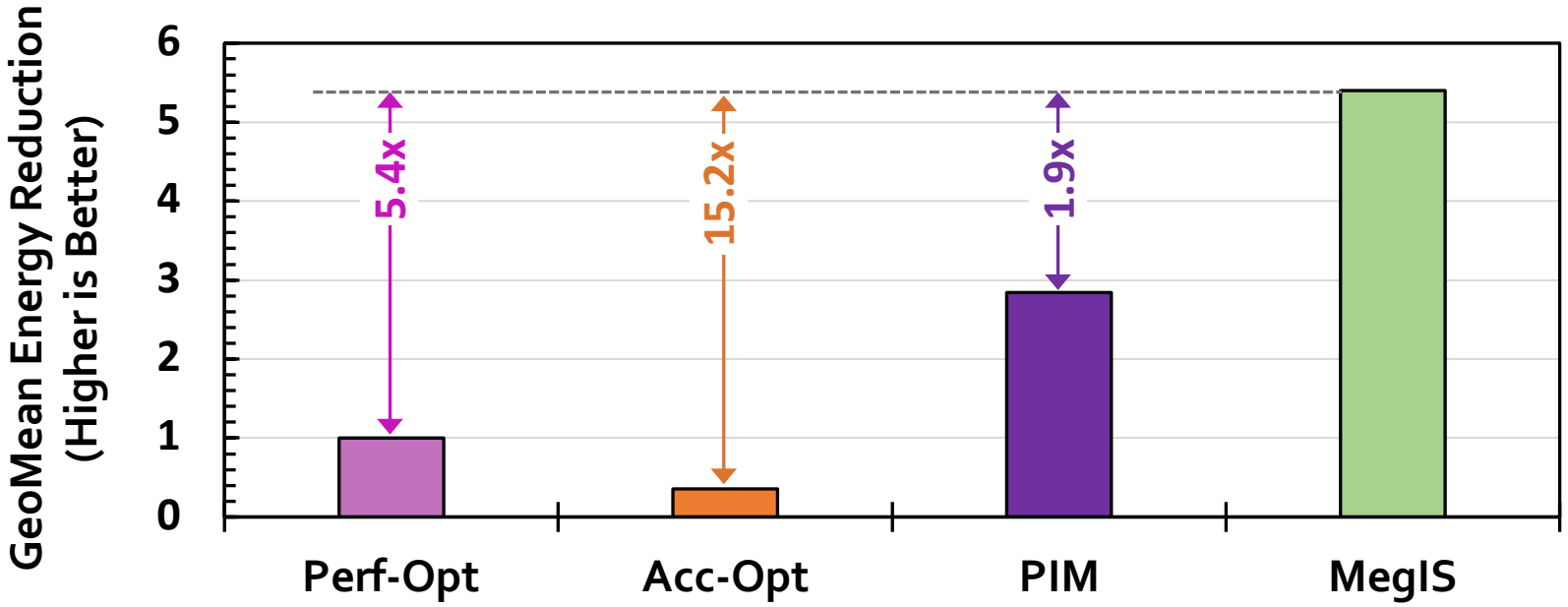




MegIS provides significant speedup over the PIM baseline

Evaluation: Reduction in Energy Consumption

- On average across different input sets and SSDs



MegIS provides significant energy reduction over the Performance-Optimized, Accuracy-Optimized, and PIM baselines

Evaluation: Accuracy, Area, and Power

Accuracy

- **Same accuracy** as the **accuracy-optimized** baseline
- **Significantly higher accuracy** than the **performance-optimized** and **PIM** baselines
  - 4.6 – 5.2× higher F1 scores
  - 3 – 24% lower L1 norm error

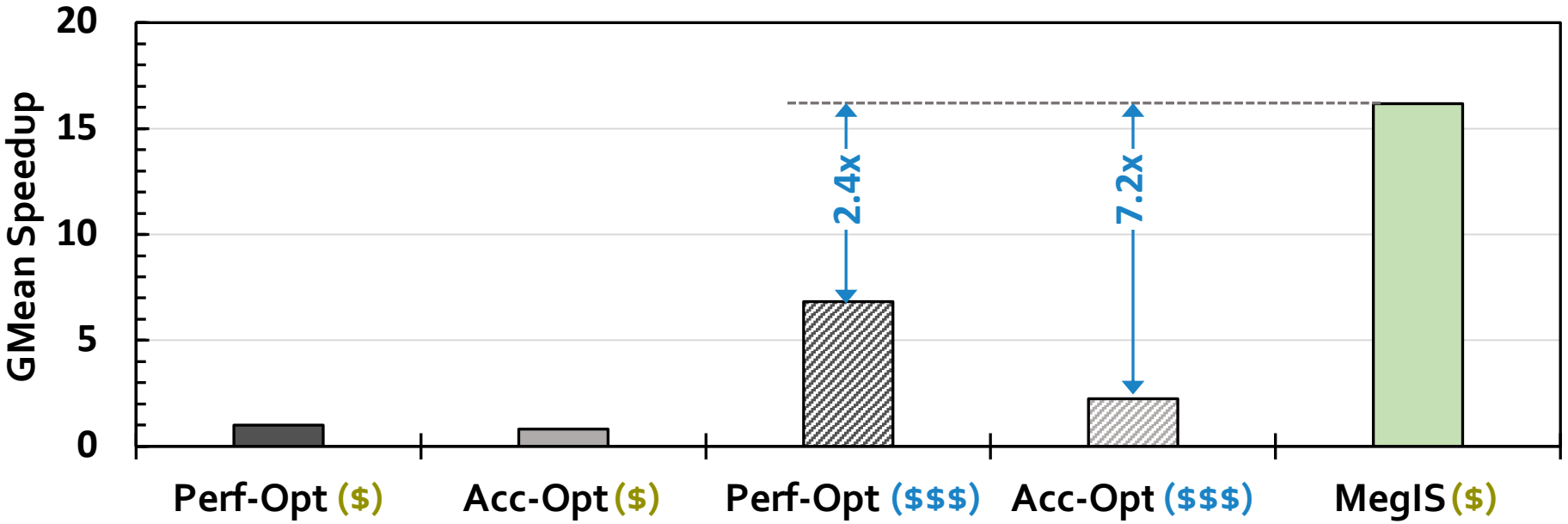
Area and Power

Total for an 8-channel SSD:

- **Area:** 0.04 mm^2 *(Only **1.7%** of the area of three ARM Cortex R4 cores in an SSD controller)*
- **Power:** 7.658 mW

Evaluation: System Cost-Efficiency

- **Cost-optimized system ($):** With SSD-C and 64-GB DRAM
- **Performance-optimized system ($$$):** With SSD-P and 1-TB DRAM

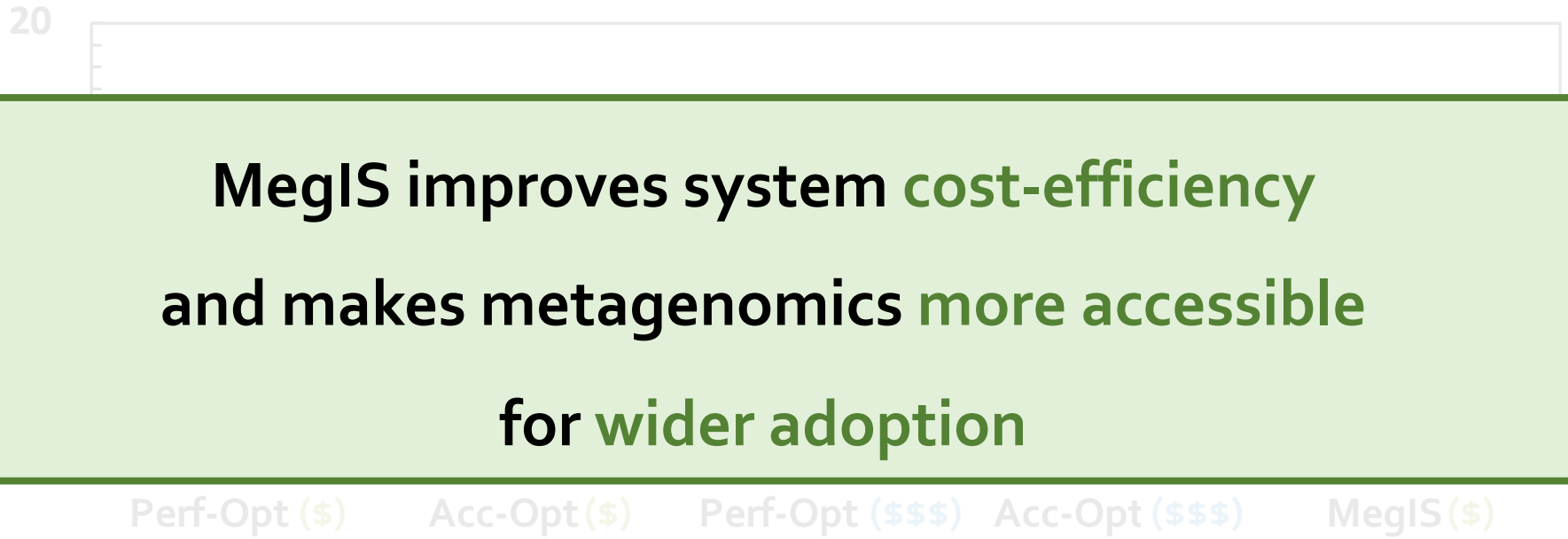


MegIS outperforms the baselines

***even* when running on a much less costly system**

Evaluation: System Cost-Efficiency

- **Cost-optimized system ($):** With SSD-C and 64-GB DRAM
- **Performance-optimized system ($$$):** With SSD-P and 1-TB DRAM



MegIS improves system cost-efficiency

and makes metagenomics more accessible

for wider adoption

MegIS outperforms the baselines

even when running on a much less costly system

More in the Paper

- MegIS's performance when running in-storage processing operations on the **cores existing in the SSD controller**
- MegIS's performance when using the same accelerators **outside SSD**
- **Sensitivity analysis with varying**
  - Database sizes
  - Memory capacities
  - #SSDs
  - #Channels
  - #Samples
- MegIS's performance for **abundance estimation**

More in the Paper

MegIS: High-Performance, Energy-Efficient, and Low-Cost Metagenomic Analysis with In-Storage Processing

Nika Mansouri Ghiasi[1] Mohammad Sadrosadati[1] Harun Mustafa[1] Arvid Gollwitzer[1]
Can Firtina[1] Julien Eudine[1] Haiyu Mao[1] Joël Lindegger[1] Meryem Banu Cavlak[1]
Mohammed Alser[1] Jisung Park[2] Onur Mutlu[1]
[1]ETH Zürich [2]POSTECH

- Database sizes
- Memory capacities
- #SSDs
- #Channels
- #Samples

• MegIS's performance for abundance estimation

https://arxiv.org/abs/2406.19113

MegIS: Summary

Metagenomic analysis suffers from
significant storage I/O data movement overhead

MegIS

The *first* **in-storage processing** system for *end-to-end* metagenomic analysis
Leverages and orchestrates **processing inside** and **outside** the storage system

Improves performance

2.7×–37.2× over **performance-optimized** software
6.9×–100.2× over **accuracy-optimized** software
1.5×–5.1× over **hardware-accelerated PIM** baseline

High accuracy

Same as accuracy-optimized
4.8× higher F1 scores
over **performance-optimized/PIM**

Reduces energy consumption

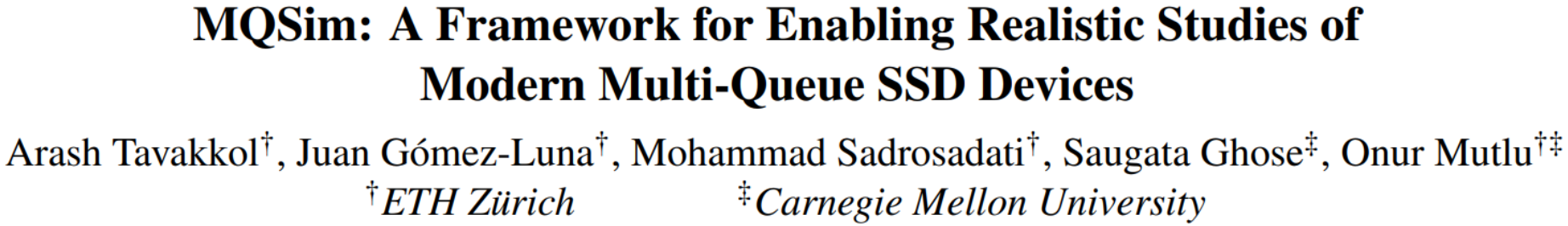
5.4× over **performance-optimized** software
15.2× over **accuracy-optimized** software
1.9× over **hardware-accelerated PIM** baseline

Low area overhead

1.7% of the three cores
in an SSD controller

MQSim: Simulating Storage [FAST 2018]

- Arash Tavakkol, Juan Gomez-Luna, Mohammad Sadrosadati, Saugata Ghose, and Onur Mutlu,
  "MQSim: A Framework for Enabling Realistic Studies of Modern Multi-Queue SSD Devices"
  *Proceedings of the 16th USENIX Conference on File and Storage Technologies (**FAST**)*, Oakland, CA, USA, February 2018.
  [Slides (pptx) (pdf)]
  [Source Code]



https://github.com/CMU-SAFARI/MQSim

Tools to Decide What to Execute Where

DAMOV Analysis Methodology & Workloads

DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

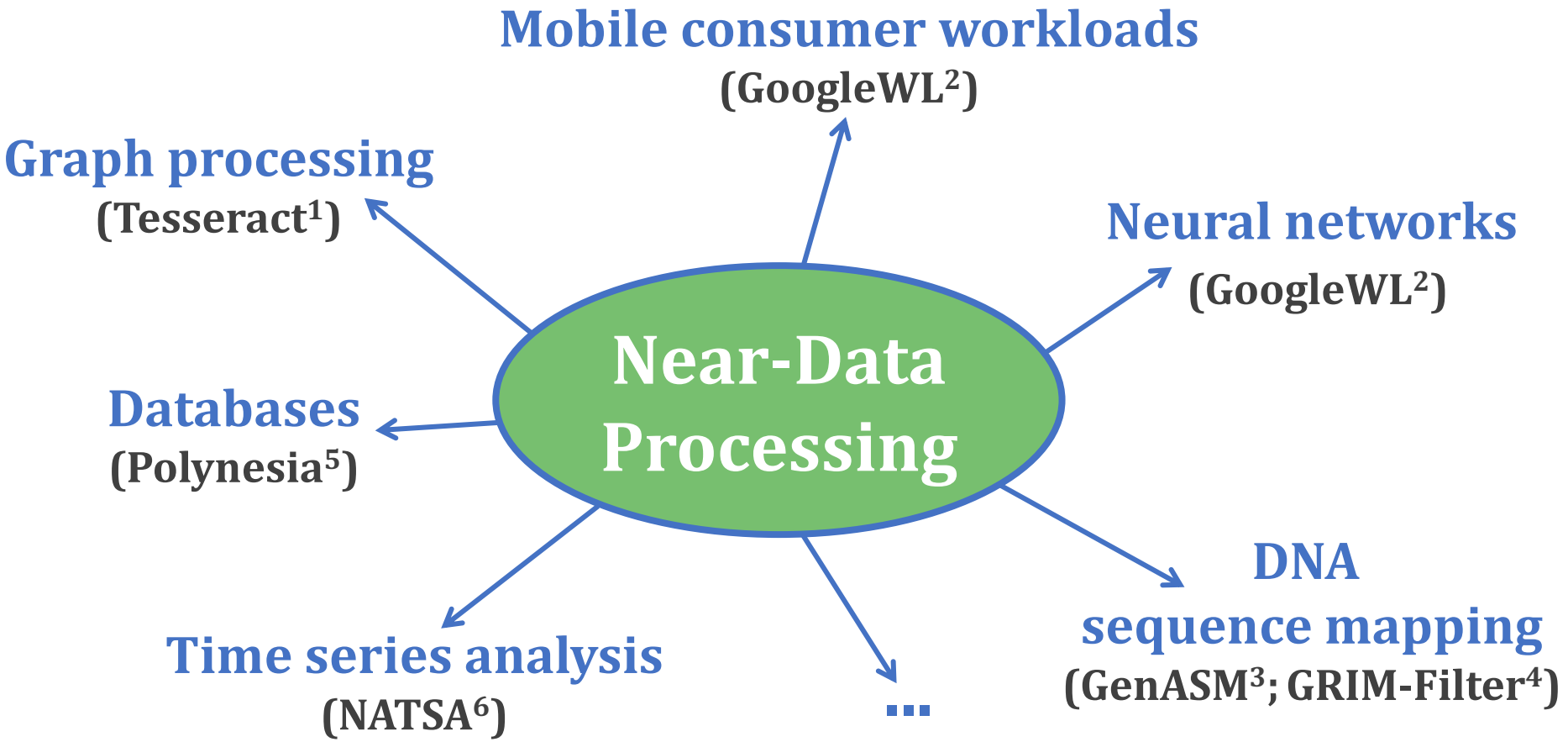
GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
LOIS OROSA, ETH Zürich, Switzerland
SAUGATA GHOSE, University of Illinois at Urbana–Champaign, USA
NANDITA VIJAYKUMAR, University of Toronto, Canada
IVAN FERNANDEZ, University of Malaga, Spain & ETH Zürich, Switzerland
MOHAMMAD SADROSADATI, Institute for Research in Fundamental Sciences (IPM), Iran & ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

Data movement between the CPU and main memory is a first-order obstacle against improving performance, scalability, and energy efficiency in modern systems. Computer systems employ a range of techniques to reduce overheads tied to data movement, spanning from traditional mechanisms (e.g., deep multi-level cache hierarchies, aggressive hardware prefetchers) to emerging techniques such as Near-Data Processing (NDP), where some computation is moved close to memory. Prior NDP works investigate the root causes of data movement bottlenecks using different profiling methodologies and tools. However, there is still a lack of understanding about the key metrics that can identify different data movement bottlenecks and their relation to traditional and emerging data movement mitigation mechanisms. Our goal is to methodically identify potential sources of data movement over a broad set of applications and to comprehensively compare traditional compute-centric data movement mitigation techniques (e.g., caching and prefetching) to more memory-centric techniques (e.g., NDP), thereby developing a rigorous understanding of the best techniques to mitigate each source of data movement.

With this goal in mind, we perform the first large-scale characterization of a wide variety of applications, across a wide range of application domains, to identify fundamental program properties that lead to data movement to/from main memory. We develop the first systematic methodology to classify applications based on the sources contributing to data movement bottlenecks. From our large-scale characterization of 77K functions across 345 applications, we select 144 functions to form the first open-source benchmark suite (DAMOV) for main memory data movement studies. We select a diverse range of functions that (1) represent different types of data movement bottlenecks, and (2) come from a wide range of application domains. Using NDP as a case study, we identify new insights about the different data movement bottlenecks and use these insights to determine the most suitable data movement mitigation mechanism for a particular application. We open-source DAMOV and the complete source code for our new characterization methodology at https://github.com/CMU-SAFARI/DAMOV.

https://arxiv.org/pdf/2105.03725.pdf

When to Employ Near-Data Processing?



[1] Ahn+, "A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing," ISCA, 2015

[2] Boroumand+, "Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks," ASPLOS, 2018

[3] Cali+, "GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis," MICRO, 2020

[4] Kim+, "GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies," BMC Genomics, 2018

[5] Boroumand+, "Polynesia: Enabling Effective Hybrid Transactional/Analytical Databases with Specialized Hardware/Software Co-Design," arXiv:2103.00798 [cs.AR], 2021

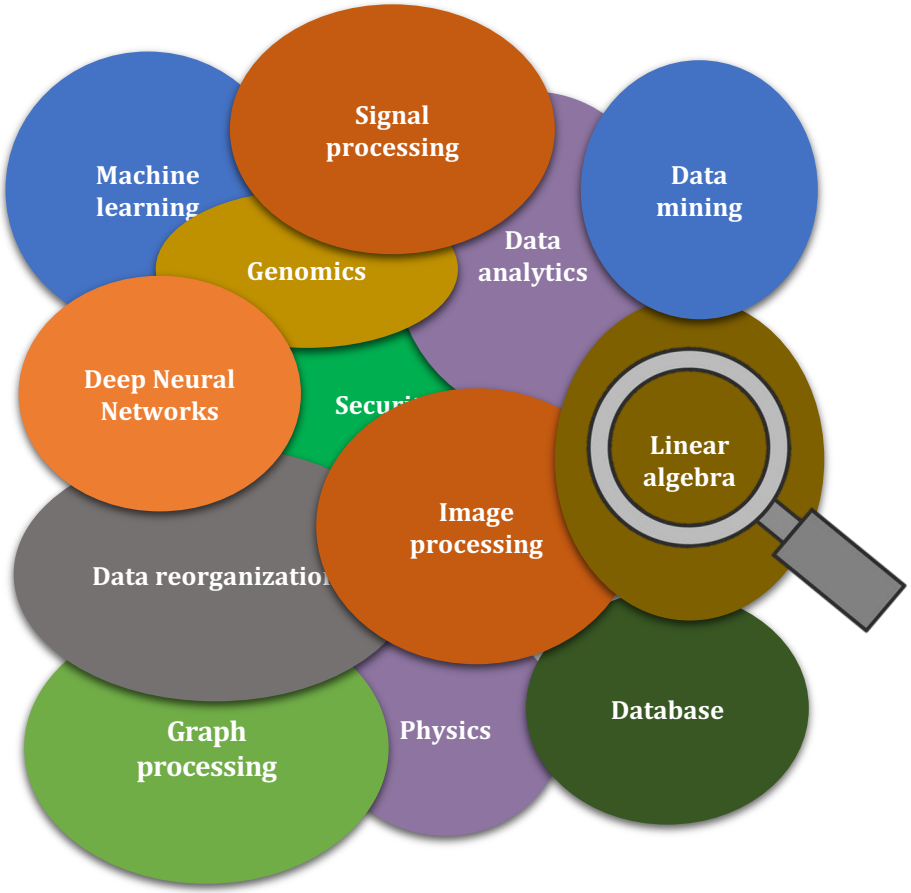
[6] Fernandez+, "NATSA: A Near-Data Processing Accelerator for Time Series Analysis," ICCD, 2020

Step 1: Application Profiling

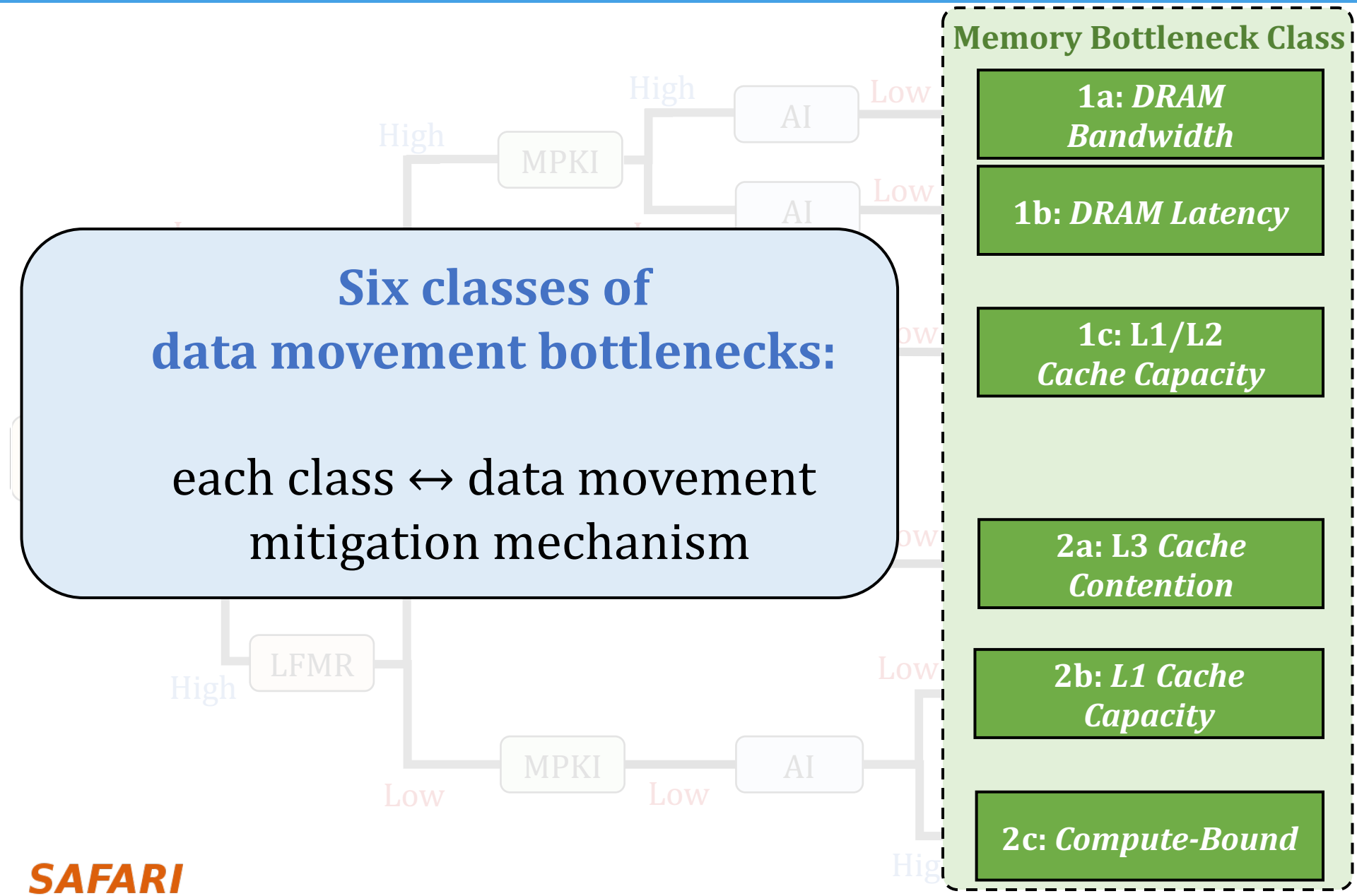
- We analyze 345 applications from distinct domains:

- Graph Processing
- Deep Neural Networks
- Physics
- High-Performance Computing
- Genomics
- Machine Learning
- Databases
- Data Reorganization
- Image Processing
- Map-Reduce
- Benchmarking
- Linear Algebra

...



Step 3: Memory Bottleneck Analysis



Six classes of data movement bottlenecks:

each class ↔ data movement mitigation mechanism

DAMOV is Open Source

- We open-source our benchmark suite and our toolchain

DAMOV-SIM

DAMOV Benchmarks

CMU-SAFARI / **DAMOV**

<> Code | Issues | Pull requests | Actions | Projects | Security | Insights | Settings

main | 1 branch | 0 tags | Go to file | Add file | Code

omutlu Update README.md — ce1b4ea 17 days ago — 5 commits

simulator	Cleaning	19 days ago
README.md	Update README.md	17 days ago
get_workloads.sh	DAMOV -- first commit	19 days ago

README.md

DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processing.

The DAMOV benchmark suite is the first open-source benchmark suite for main memory data movement-related studies, based on our systematic characterization methodology. This suite consists of 144 functions representing different sources of data movement bottlenecks and can be used as a baseline benchmark set for future data-movement mitigation research. The applications in the DAMOV benchmark suite belong to popular benchmark suites, including BWA, Chai, Darknet, GASE, Hardware Effects, Hashjoin, HPCC, HPCG, Ligra, PARSEC, Parboil, PolyBench, Phoenix, Rodinia, SPLASH-2, STREAM.

About

DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processing. Described by Oliveira et al. (preliminary version at https://arxiv.org/pdf/2105.03725.pdf)

Readme

Releases

No releases published
Create a new release

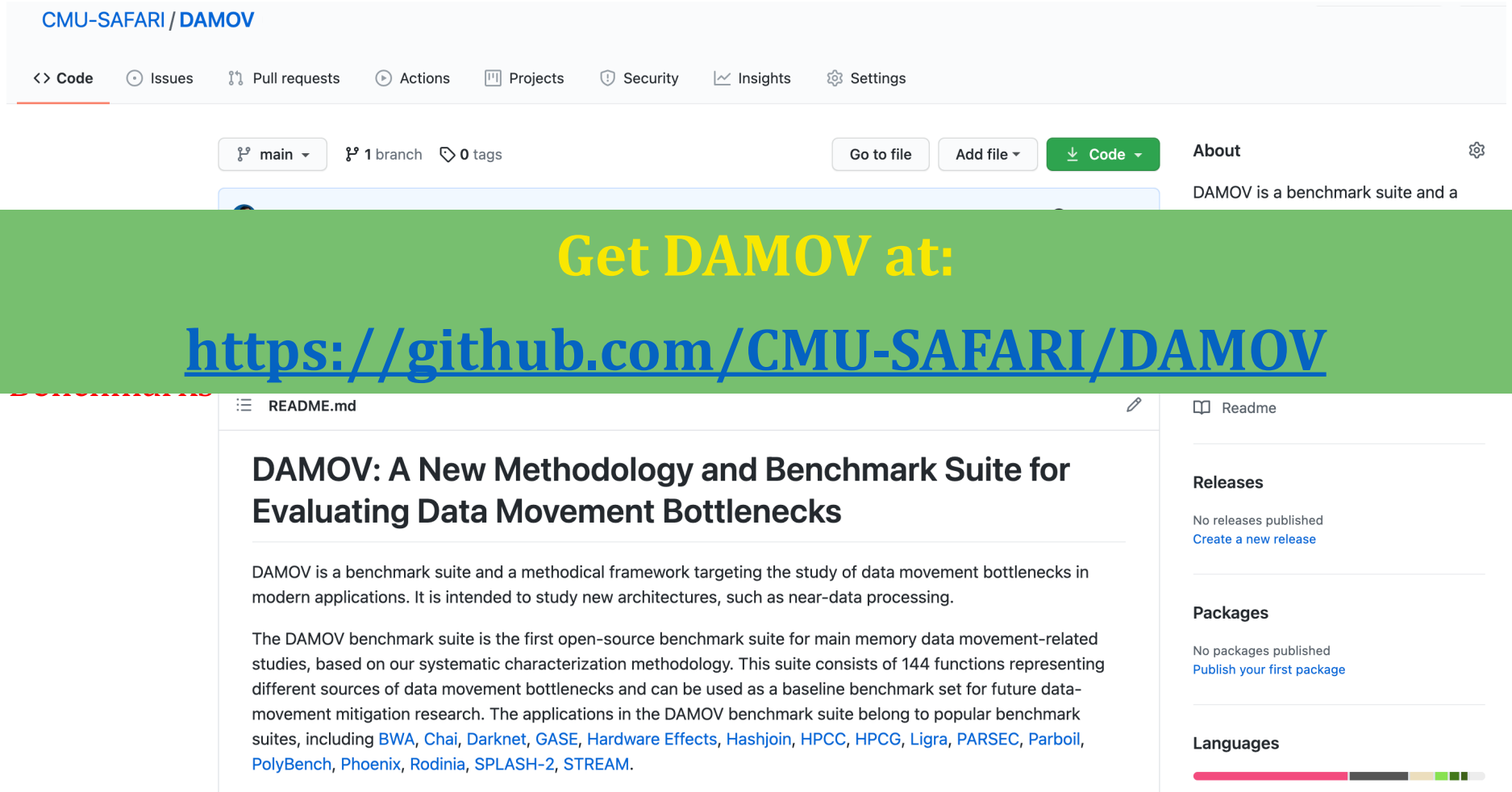
Packages

No packages published
Publish your first package

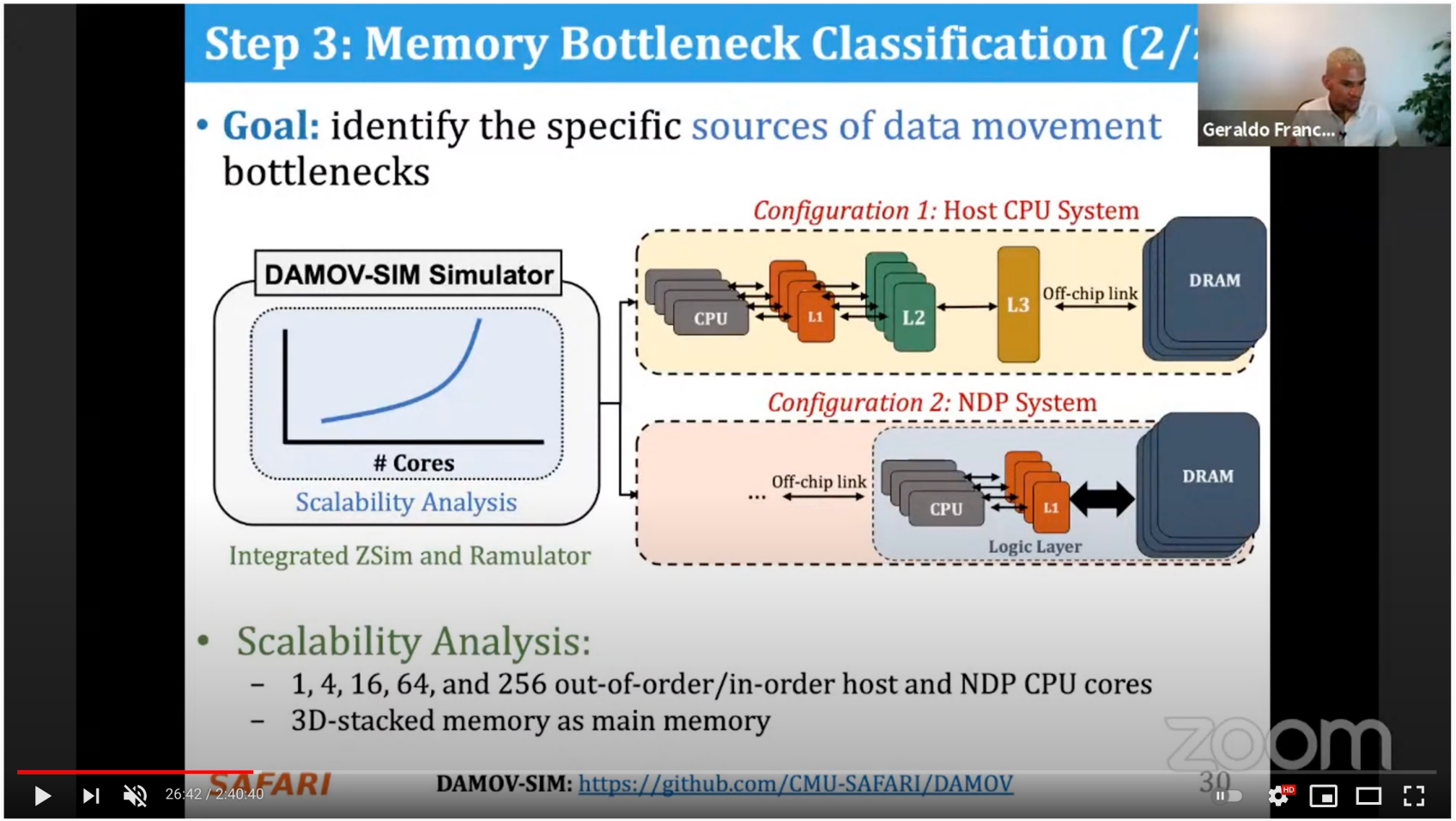
Languages

DAMOV is Open Source

- Benchmark suite and toolchain are open sourced



More on DAMOV Analysis Methodology & Workloads



SAFARI Live Seminar: DAMOV: A New Methodology & Benchmark Suite for Data Movement Bottlenecks

352 views • Streamed live on Jul 22, 2021

18 0 SHARE SAVE ...

Onur Mutlu Lectures
17.7K subscribers

ANALYTICS EDIT VIDEO

https://www.youtube.com/watch?v=GWideVyo0nM&list=PL5Q2soXY2Zi_tOTAYm--dYByNPL7JhwR9&index=3

More on DAMOV Methods & Benchmarks

- Geraldo F. Oliveira, Juan Gomez-Luna, Lois Orosa, Saugata Ghose, Nandita Vijaykumar, Ivan fernandez, Mohammad Sadrosadati, and Onur Mutlu, **"DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks"**
  IEEE Access, 8 September 2021.
  Preprint in ***arXiv***, 8 May 2021.
  [arXiv preprint]
  [IEEE Access version]
  [DAMOV Suite and Simulator Source Code]
  [SAFARI Live Seminar Video (2 hrs 40 mins)]
  [Short Talk Video (21 minutes)]

DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
LOIS OROSA, ETH Zürich, Switzerland
SAUGATA GHOSE, University of Illinois at Urbana–Champaign, USA
NANDITA VIJAYKUMAR, University of Toronto, Canada
IVAN FERNANDEZ, University of Malaga, Spain & ETH Zürich, Switzerland
MOHAMMAD SADROSADATI, ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

Ramulator 2.0 for PIM Systems

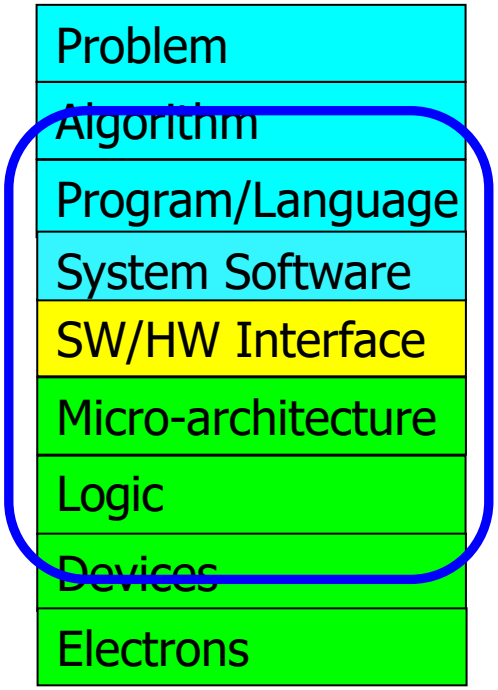
- Haocong Luo, Yahya Can Tugrul, F. Nisa Bostanci, Ataberk Olgun, A. Giray Yaglikci, and Onur Mutlu,
  "Ramulator 2.0: A Modern, Modular, and Extensible DRAM Simulator"
  Preprint on ***arxiv***, August 2023.
  [arXiv version]
  [Ramulator 2.0 Source Code]

https://arxiv.org/pdf/2308.11030.pdf

https://github.com/CMU-SAFARI/ramulator2

We Need to Revisit the Entire Stack

- With a **memory-centric mindset**



We can get there step by step

PIM Review and Open Problems

A Modern Primer on Processing in Memory

Onur Mutlu[a,b], Saugata Ghose[b,c], Juan Gómez-Luna[a], Rachata Ausavarungnirun[d]

SAFARI Research Group

[a] *ETH Zürich*
[b] *Carnegie Mellon University*
[c] *University of Illinois at Urbana-Champaign*
[d] *King Mongkut's University of Technology North Bangkok*

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun,
"A Modern Primer on Processing in Memory"
Invited Book Chapter in ***Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann***, Springer, to be published in 2021.

https://arxiv.org/pdf/2012.03112.pdf

PIM Review and Open Problems (II)

A Workload and Programming Ease Driven Perspective of Processing-in-Memory

Saugata Ghose[†] Amirali Boroumand[†] Jeremie S. Kim[†§] Juan Gómez-Luna[§] Onur Mutlu[§†]

[†]*Carnegie Mellon University* [§]*ETH Zürich*

Saugata Ghose, Amirali Boroumand, Jeremie S. Kim, Juan Gomez-Luna, and Onur Mutlu,
"Processing-in-Memory: A Workload-Driven Perspective"
Invited Article in IBM Journal of Research & Development, Special Issue on Hardware for Artificial Intelligence, to appear in November 2019.
[Preliminary arXiv version]

https://arxiv.org/pdf/1907.12947.pdf

Processing in Memory: Adoption Challenges

1. Processing **using** Memory
2. Processing **near** Memory

Eliminating the Adoption Barriers

How to Enable Adoption of Processing in Memory

Potential Barriers to Adoption of PIM

1. **Applications** & **software** for PIM

2. Ease of **programming** (interfaces and compiler/HW support)

3. **System** and **security** support: coherence, synchronization, virtual memory, isolation, communication interfaces, …

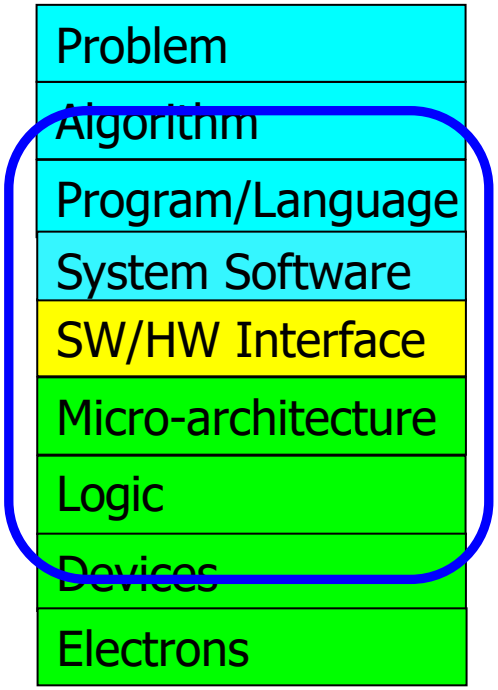
4. **Runtime** and **compilation** systems for adaptive scheduling, data mapping, access/sharing control, …

5. **Infrastructures** to assess benefits and feasibility

All can be solved with change of mindset

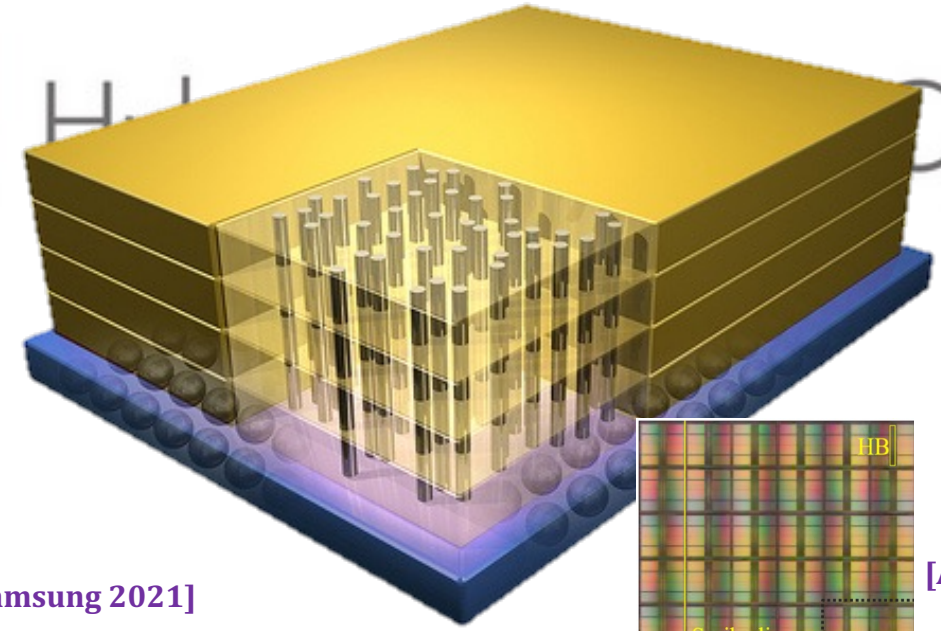
We Need to Revisit the Entire Stack

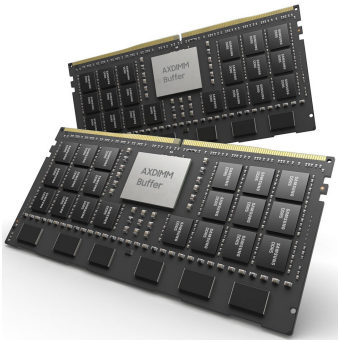
- With a **memory-centric mindset**



We can get there step by step

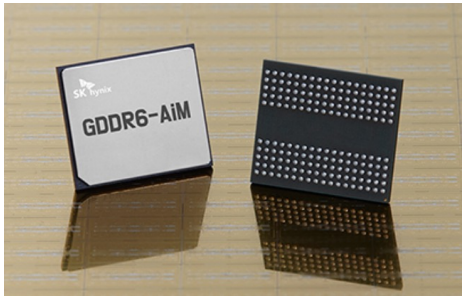
Processing-in-Memory Landscape Today



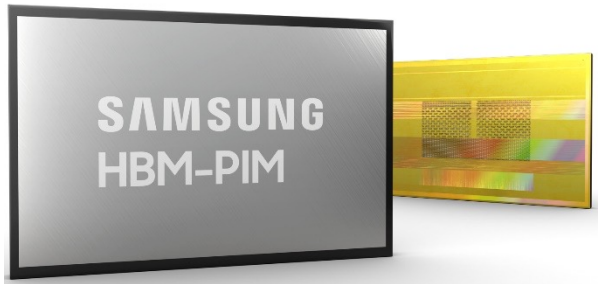


[Samsung 2021]

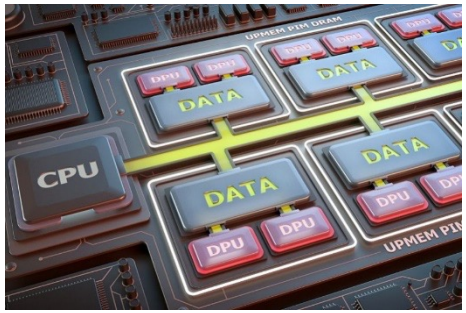
[Alibaba 2022]



[SK Hynix 2022]



[Samsung 2021]



[UPMEM 2019]

And, many other experimental chips and startups

Adoption: How to Keep It **Simple**?

- Junwhan Ahn, Sungjoo Yoo, Onur Mutlu, and Kiyoung Choi, **"PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture"** *Proceedings of the 42nd International Symposium on Computer Architecture (**ISCA**)*, Portland, OR, June 2015. [Slides (pdf)] [Lightning Session Slides (pdf)]

PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture

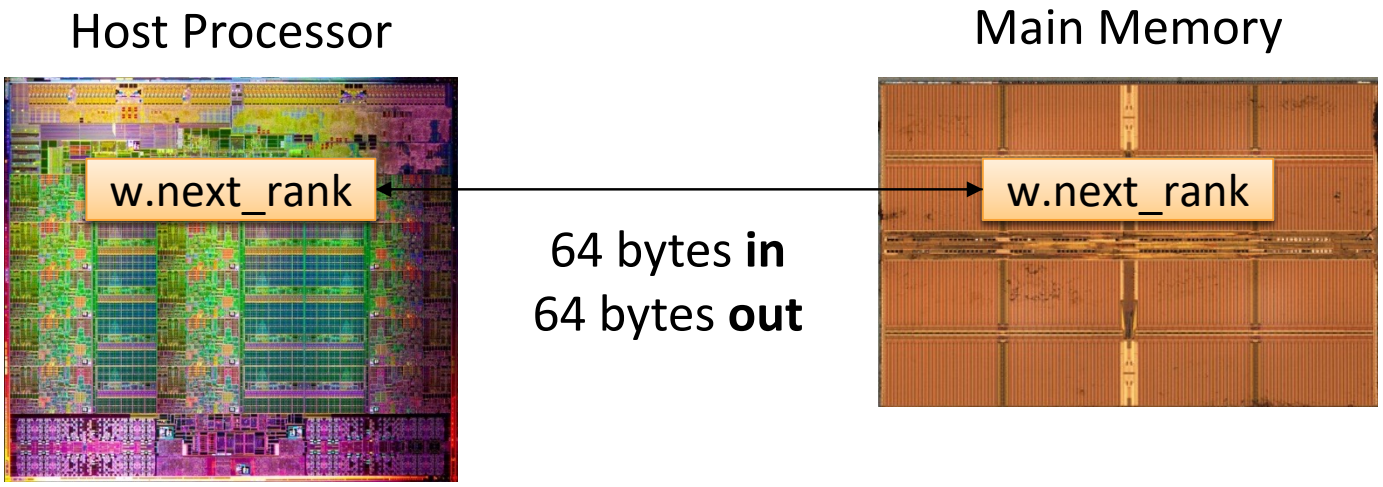
Junwhan Ahn Sungjoo Yoo Onur Mutlu† Kiyoung Choi
junwhan@snu.ac.kr, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr
Seoul National University †Carnegie Mellon University

PEI: PIM-Enabled Instructions (Ideas)

- Goal: Develop mechanisms to get the most out of near-data processing with minimal cost, minimal changes to the system, no changes to the programming model
- Key Idea 1: Expose each PIM operation as a cache-coherent, virtually-addressed host processor instruction (called PEI) that operates on only a single cache block
  - e.g., __pim_add(&w.next_rank, value) → pim.add r1, (r2)
  - No changes sequential execution/programming model
  - No changes to virtual memory
  - Minimal changes to cache coherence
  - No need for data mapping: Each PEI restricted to a single memory module
- Key Idea 2: Dynamically decide where to execute a PEI (i.e., the host processor or PIM accelerator) based on simple locality characteristics and simple hardware predictors
  - Execute each operation at the location that provides the best performance

Simple PIM Operations as ISA Extensions (II)

```
for (v: graph.vertices) {
    value = weight * v.rank;
    for (w: v.successors) {
        w.next_rank += value;
    }
}
```

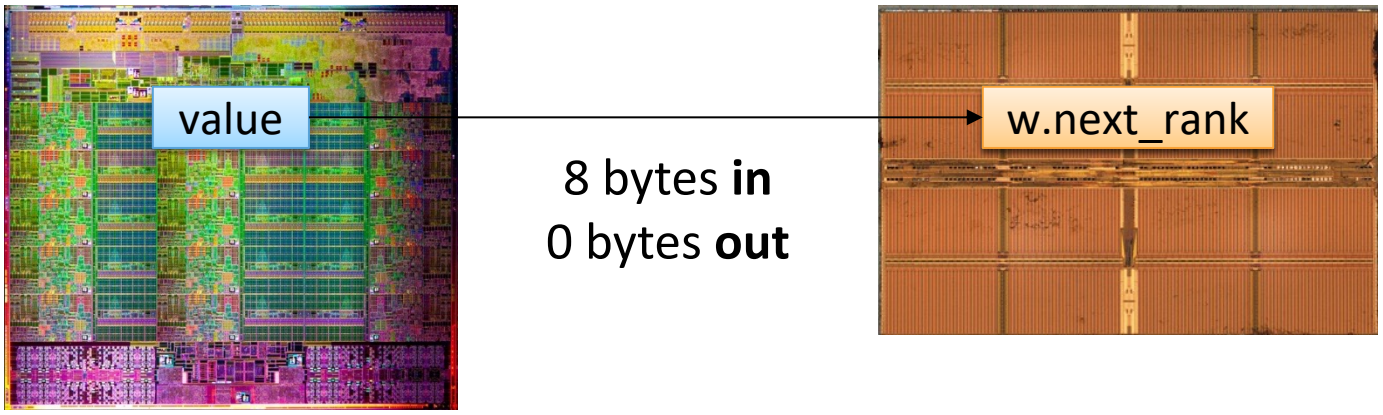


Conventional Architecture

Simple PIM Operations as ISA Extensions (III)

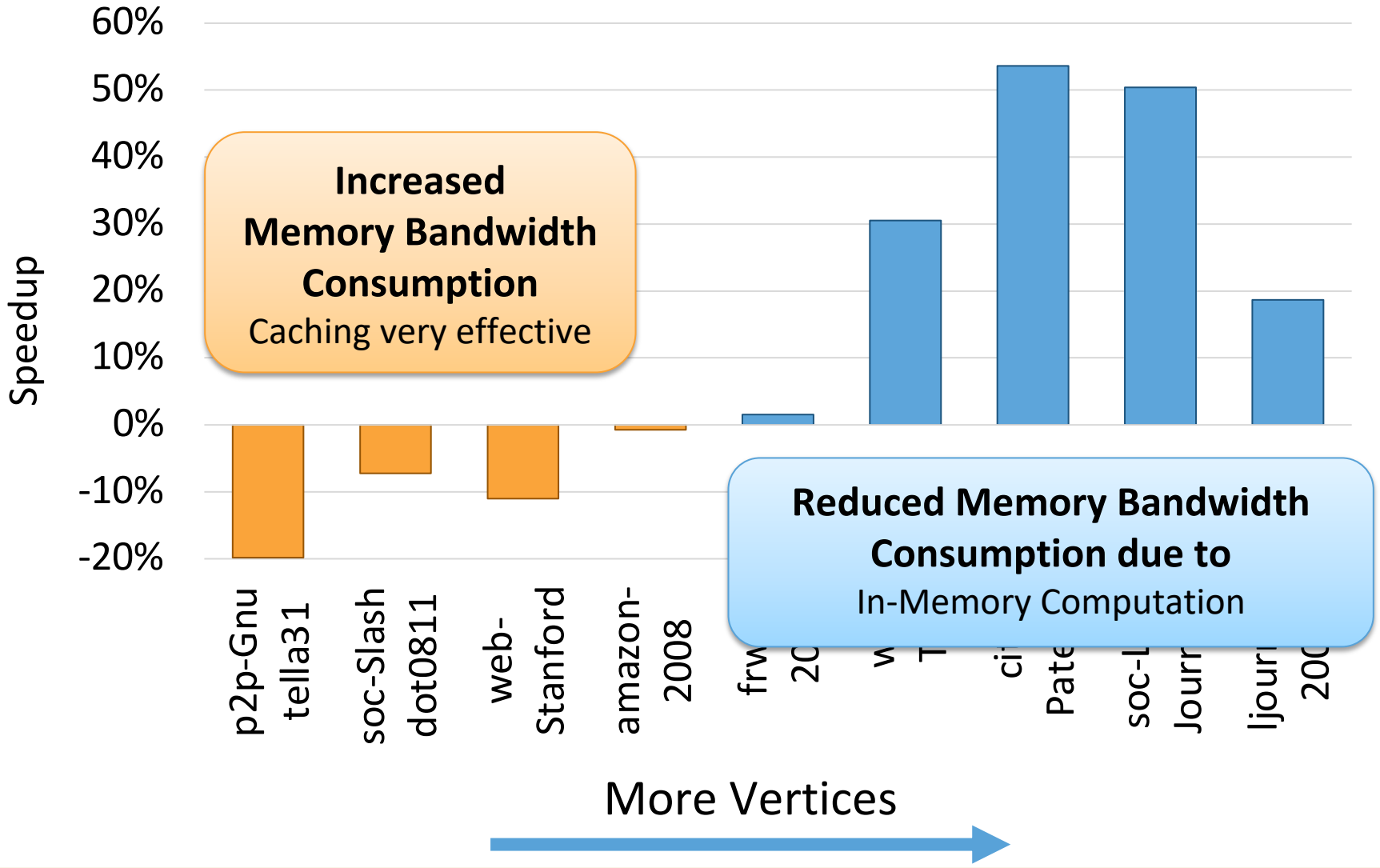
```
for (v: graph.vertices) {
    value = weight * v.rank;
    for (w: v.successors) {
        __pim_add(&w.next_rank, value);
    }
}
```

pim.add r1, (r2)



In-Memory Addition

Always Executing in Memory? Not A Good Idea



PEI: PIM-Enabled Instructions (Example)

```
for (v: graph.vertices) {
  value = weight * v.rank;
  for (w: v.successors) {
    __pim_add(&w.next_rank, value);
  }
}
pfence();
```

pim.add r1, (r2)

pfence

Table 1: Summary of Supported PIM Operations

Operation	R	W	Input	Output	Applications
8-byte integer increment	O	O	0 bytes	0 bytes	AT
8-byte integer min	O	O	8 bytes	0 bytes	BFS, SP, WCC
Floating-point add	O	O	8 bytes	0 bytes	PR
Hash table probing	O	X	8 bytes	9 bytes	HJ
Histogram bin index	O	X	1 byte	16 bytes	HG, RP
Euclidean distance	O	X	64 bytes	4 bytes	SC
Dot product	O	X	32 bytes	8 bytes	SVM

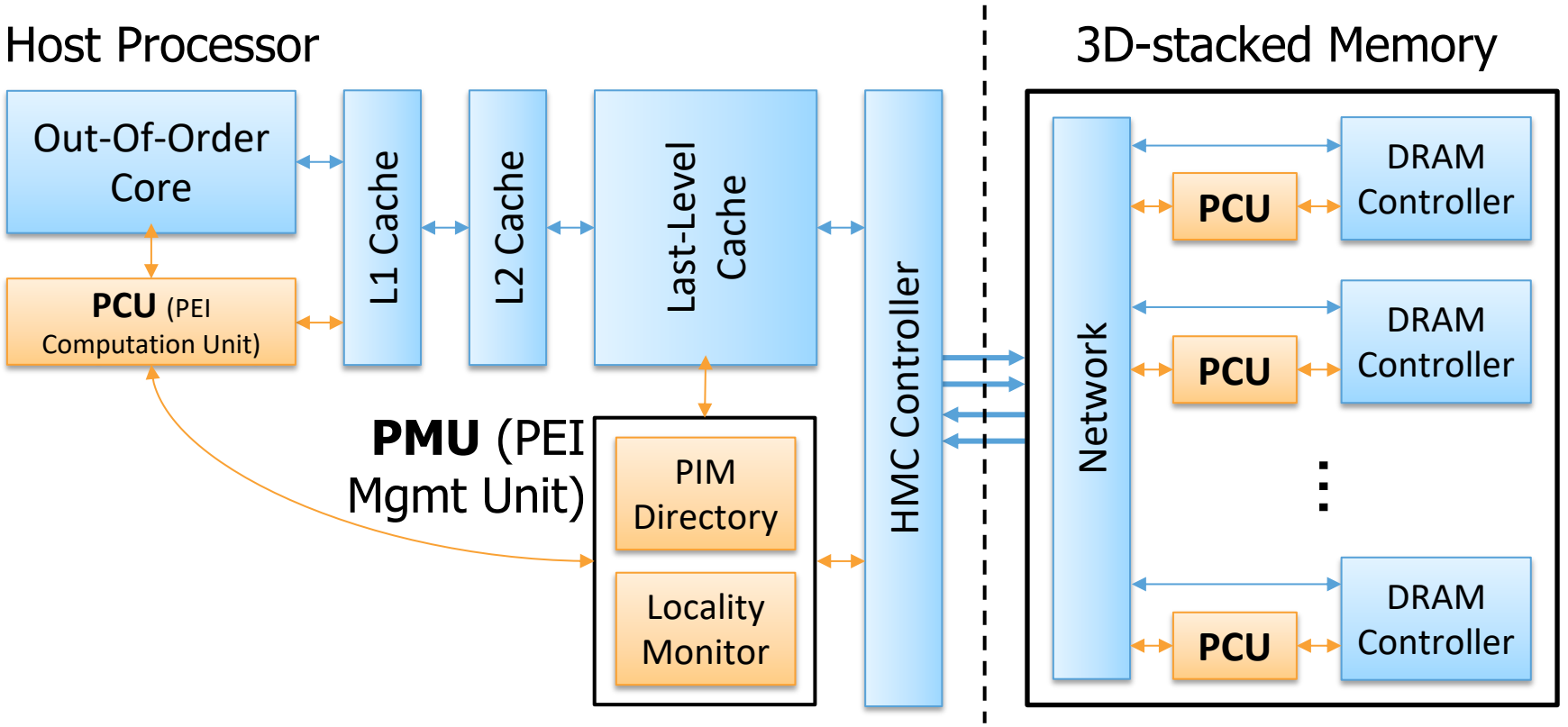
- Executed either in memory or in the processor: dynamic decision
  - Low-cost locality monitoring for a single instruction
- Cache-coherent, virtually-addressed, single cache block only
- Atomic between different PEIs
- *Not* atomic with normal instructions (use *pfence* for ordering)

PIM-Enabled Instructions

- Key to practicality: single-cache-block restriction
  - Each PEI can access *at most one last-level cache block*
  - Similar restrictions exist in atomic instructions

- Benefits
  - **Localization**: each PEI is bounded to one memory module
  - **Interoperability**: easier support for cache coherence and virtual memory
  - **Simplified locality monitoring**: data locality of PEIs can be identified simply by the cache control logic

Example (Abstract) PEI uArchitecture



Example PEI uArchitecture

PEI: Initial Evaluation Results

- Initial evaluations with 10 emerging data-intensive workloads
  - Large-scale graph processing
  - In-memory data analytics
  - Machine learning and data mining
  - Three input sets (small, medium, large) for each workload to analyze the impact of data locality

Table 2: Baseline Simulation Configuration

Component	Configuration
Core	16 out-of-order cores, 4 GHz, 4-issue
L1 I/D-Cache	Private, 32 KB, 4/8-way, 64 B blocks, 16 MSHRs
L2 Cache	Private, 256 KB, 8-way, 64 B blocks, 16 MSHRs
L3 Cache	Shared, 16 MB, 16-way, 64 B blocks, 64 MSHRs
On-Chip Network	Crossbar, 2 GHz, 144-bit links
Main Memory	32 GB, 8 HMCs, daisy-chain (80 GB/s full-duplex)
HMC	4 GB, 16 vaults, 256 DRAM banks [20]
– DRAM	FR-FCFS, tCL = tRCD = tRP = 13.75 ns [27]
– Vertical Links	64 TSVs per vault with 2 Gb/s signaling rate [23]

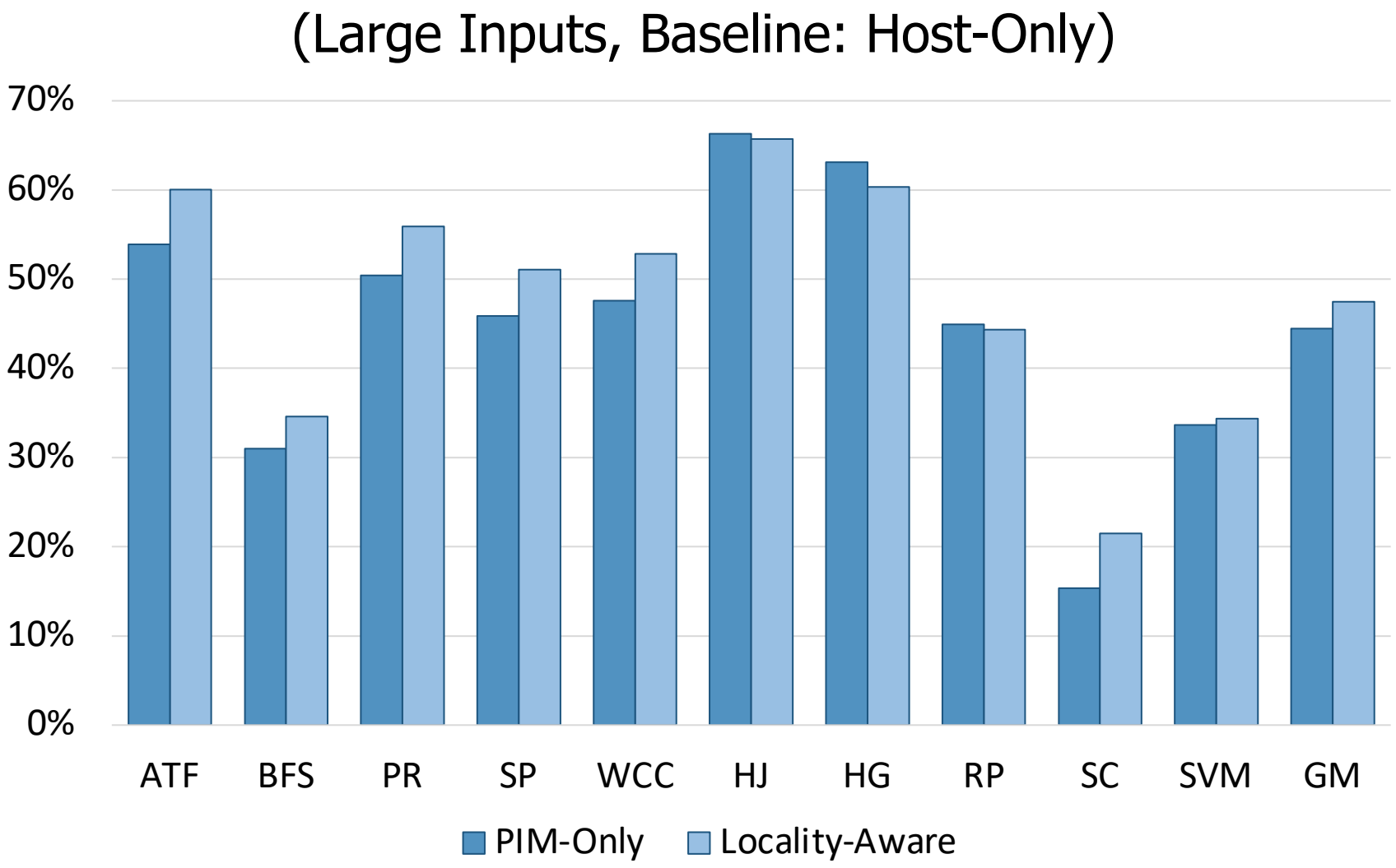
- Pin-based cycle-level x86-64 simulation

- **Performance Improvement and Energy Reduction:**
  - 47% average speedup with large input data sets
  - 32% speedup with small input data sets
  - 25% avg. energy reduction in a single node with large input data sets

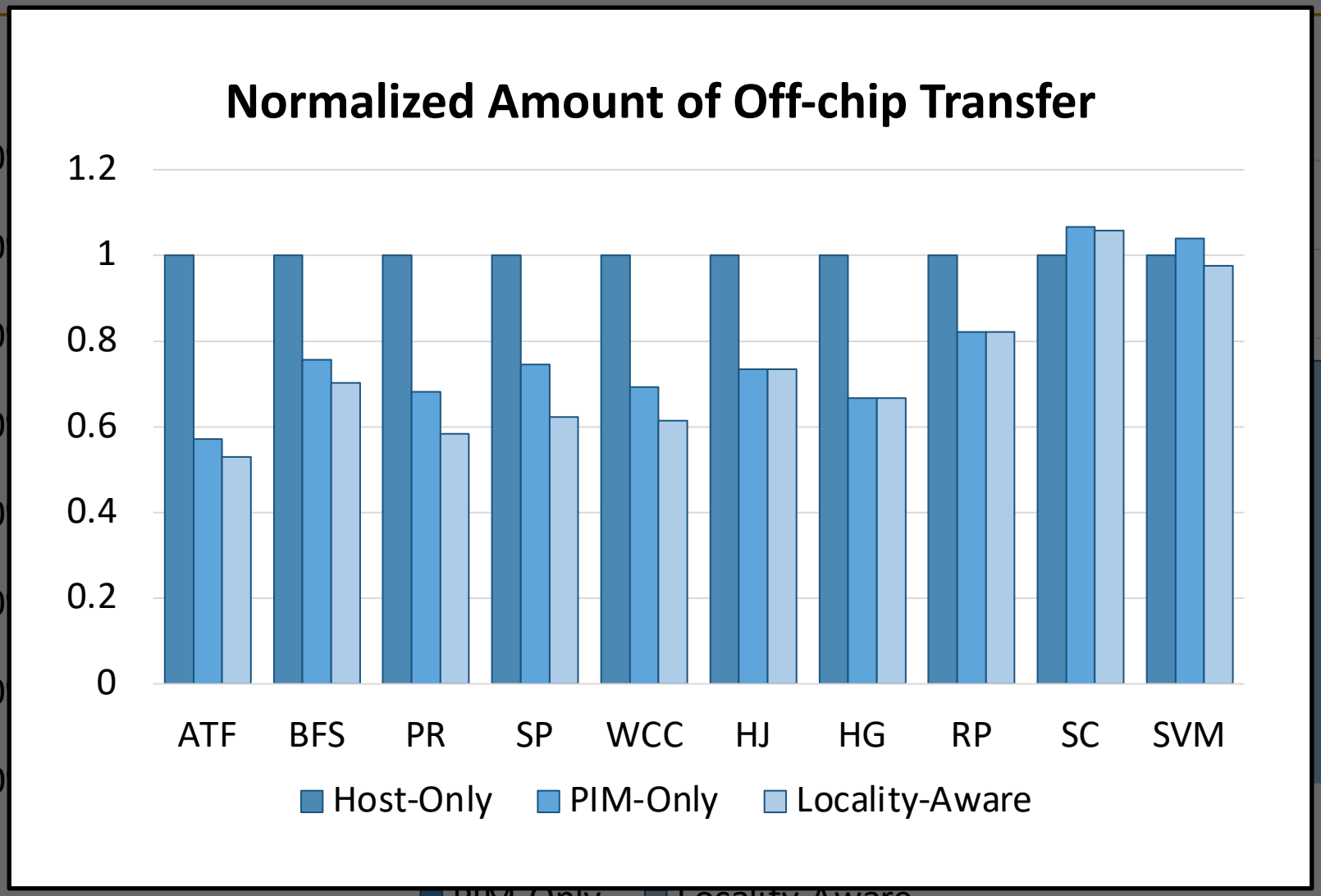
Evaluated Data-Intensive Applications

- Ten emerging data-intensive workloads
  - Large-scale graph processing
    - Average teenage follower, BFS, PageRank, single-source shortest path, weakly connected components
  - In-memory data analytics
    - Hash join, histogram, radix partitioning
  - Machine learning and data mining
    - Streamcluster, SVM-RFE
- Three input sets (small, medium, large) for each workload to show the impact of data locality

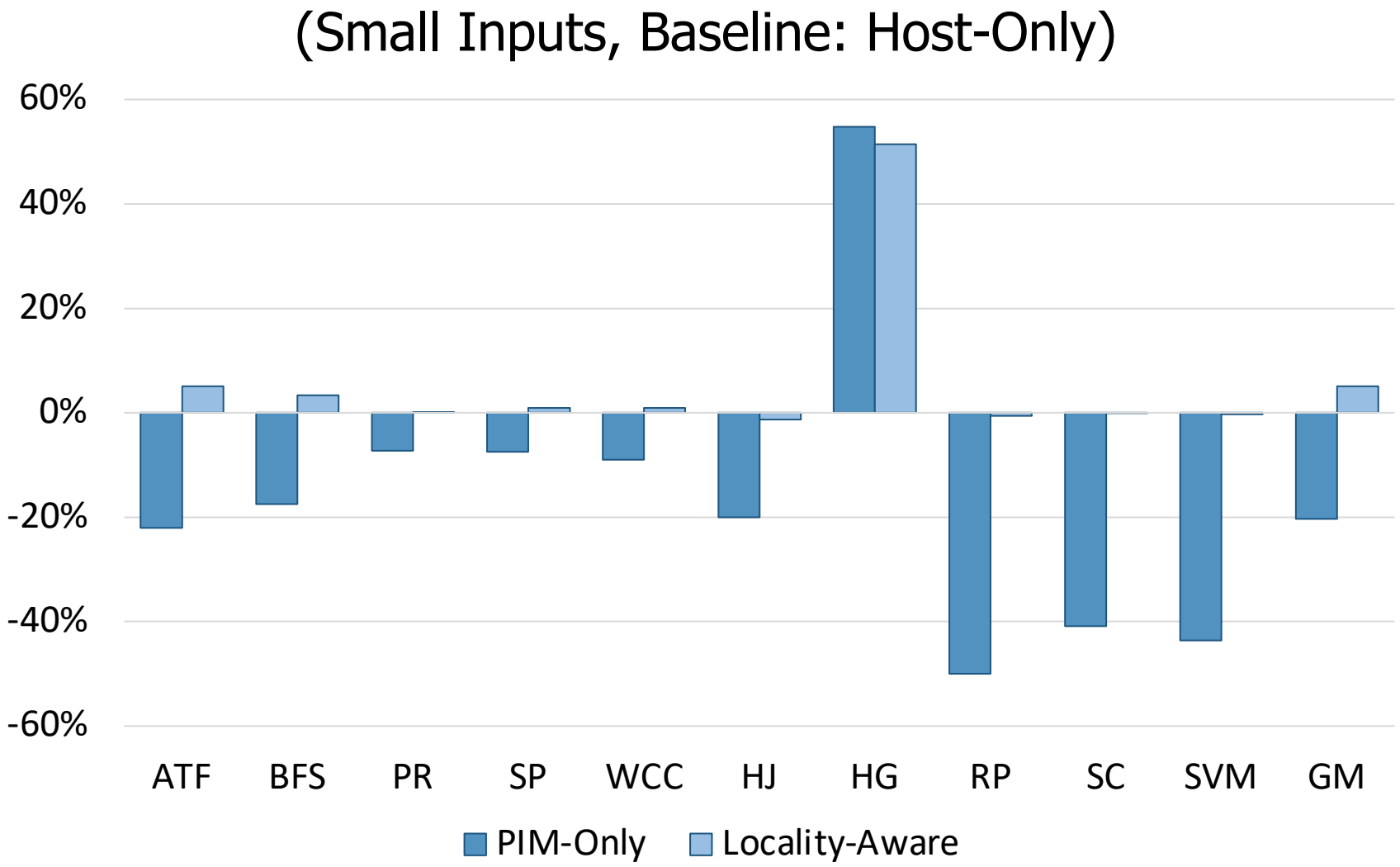
PEI Performance Delta: Large Data Sets



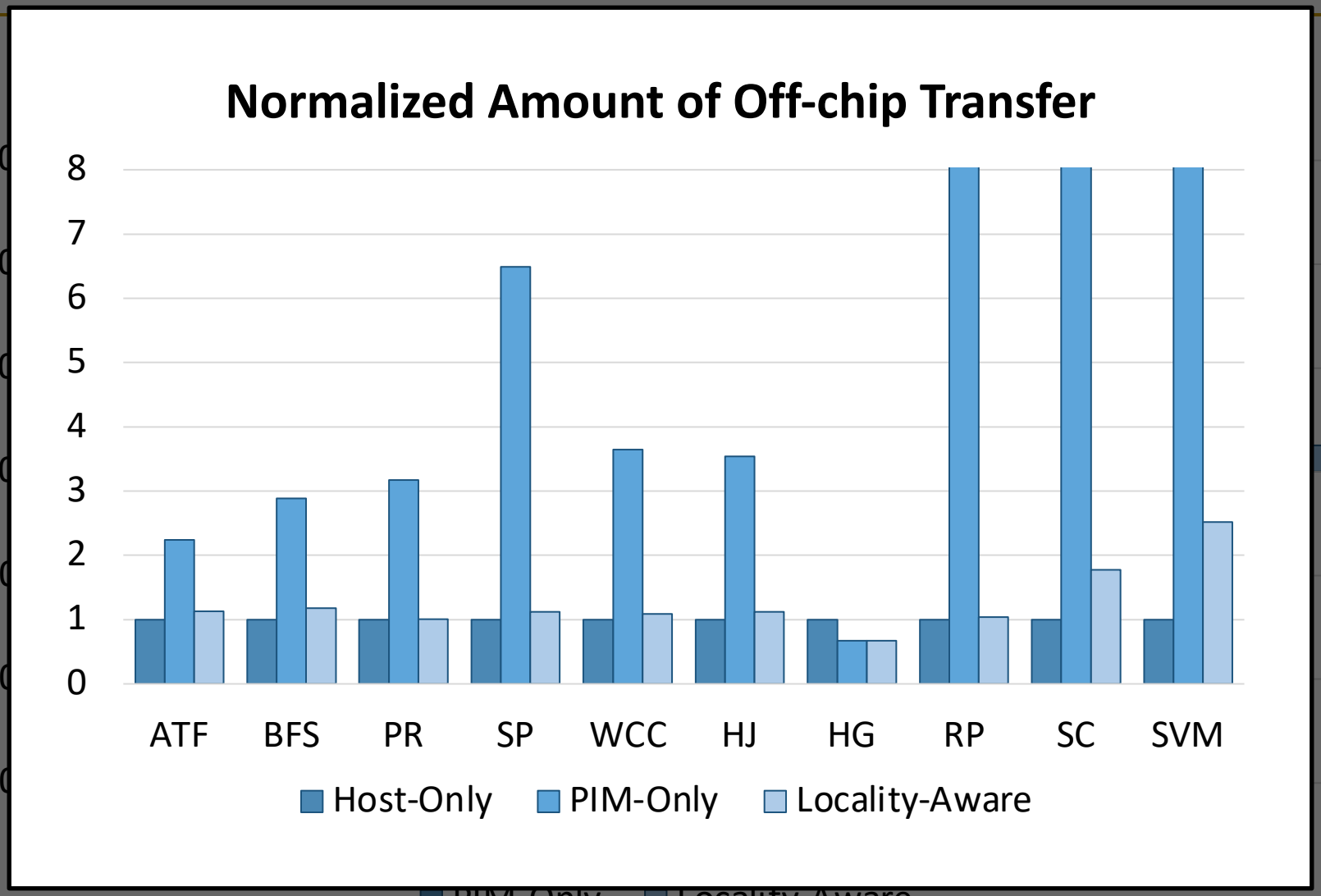
PEI Performance: Large Data Sets



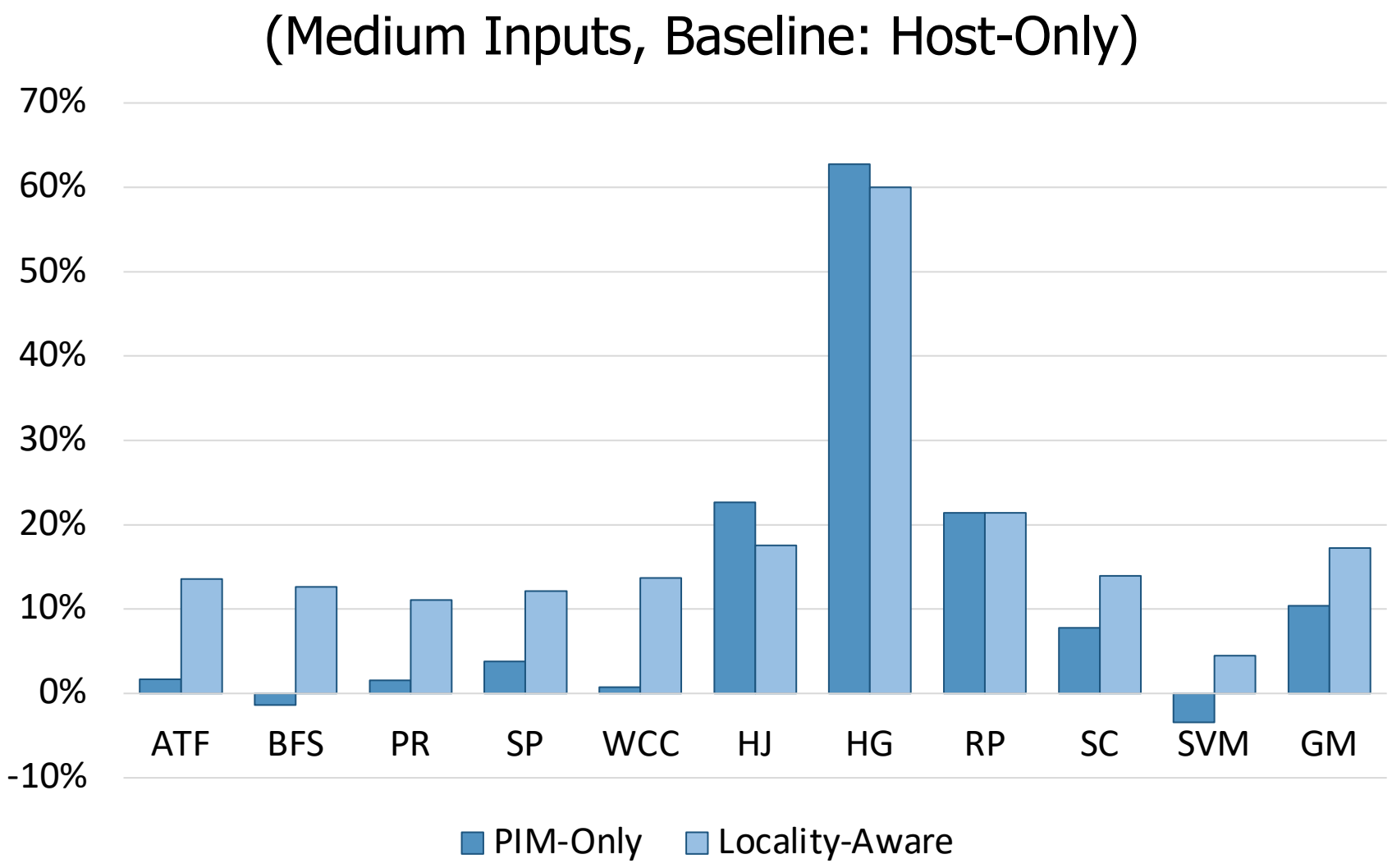
PEI Performance Delta: Small Data Sets



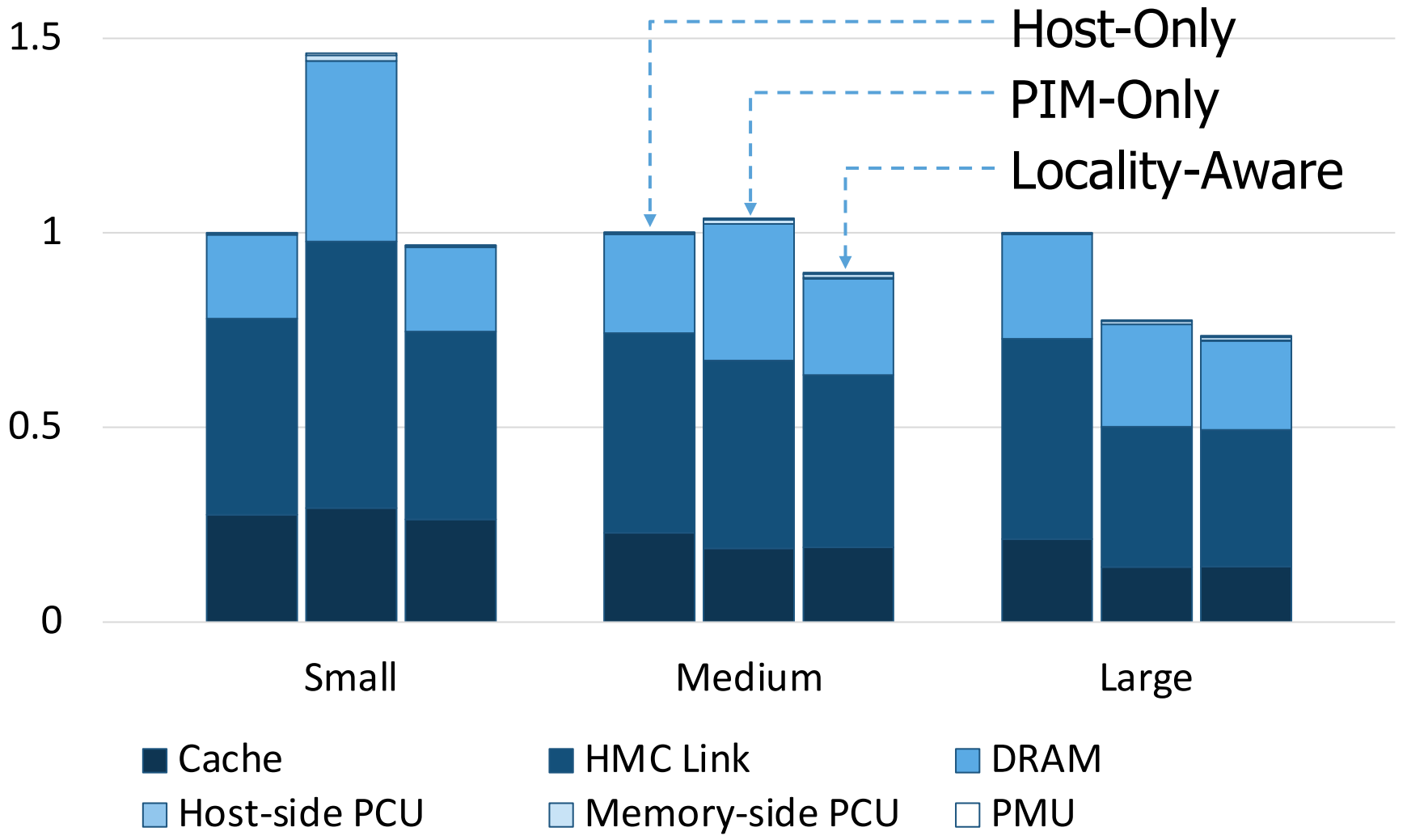
PEI Performance: Small Data Sets



PEI Performance Delta: Medium Data Sets



PEI Energy Consumption



PEI: Advantages & Disadvantages

- Advantages
  + Simple and low-cost approach to PIM
  + No changes to programming model, virtual memory
  + Dynamically decides where to execute an instruction

- Disadvantages
  - Does not take full advantage of PIM potential
    - Single cache block restriction is limiting

Adoption: How to Keep It **Simple**?

- Junwhan Ahn, Sungjoo Yoo, Onur Mutlu, and Kiyoung Choi, **"PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture"** *Proceedings of the 42nd International Symposium on Computer Architecture (**ISCA**)*, Portland, OR, June 2015. [Slides (pdf)] [Lightning Session Slides (pdf)]

PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture

Junwhan Ahn Sungjoo Yoo Onur Mutlu† Kiyoung Choi
junwhan@snu.ac.kr, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr
Seoul National University †Carnegie Mellon University

Adoption: How to Ease **Programmability**? (I)

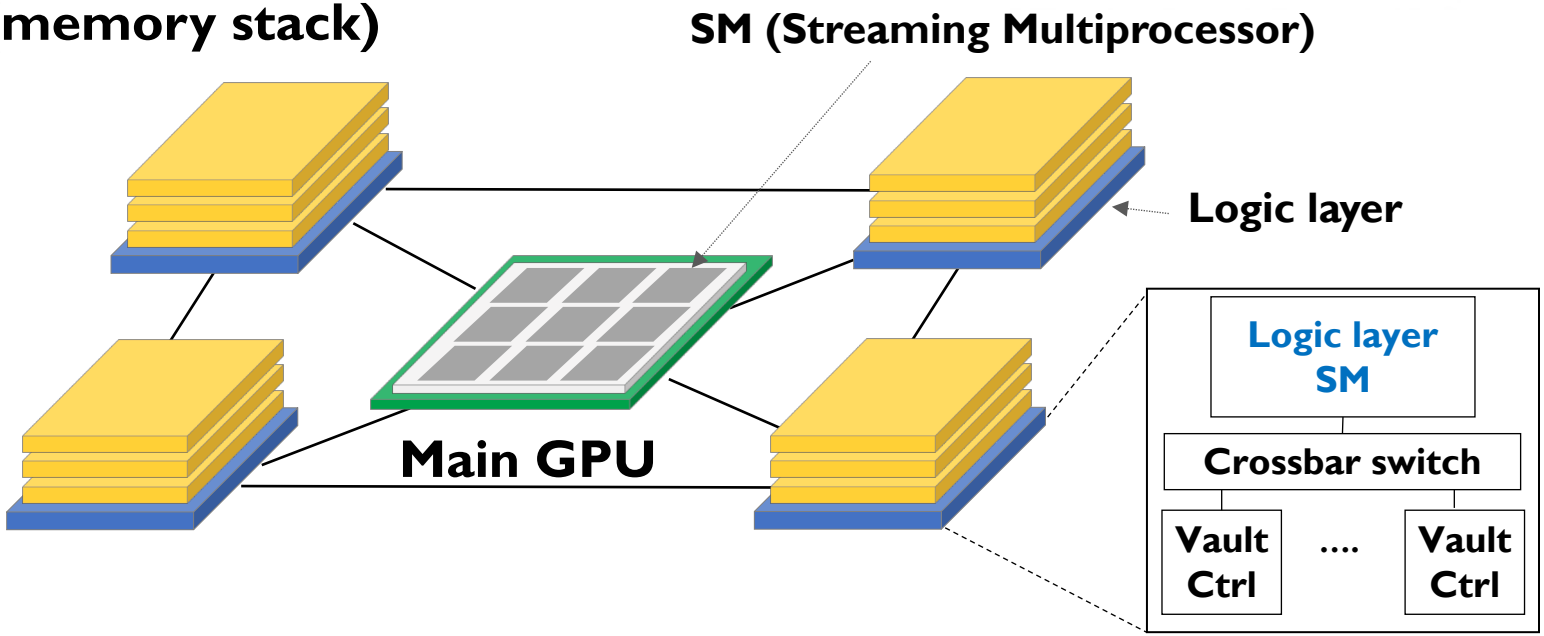
- Kevin Hsieh, Eiman Ebrahimi, Gwangsun Kim, Niladrish Chatterjee, Mike O'Connor, Nandita Vijaykumar, Onur Mutlu, and Stephen W. Keckler, **"Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems"**
  *Proceedings of the 43rd International Symposium on Computer Architecture (**ISCA**)*, Seoul, South Korea, June 2016.
  [Slides (pptx) (pdf)]
  [Lightning Session Slides (pptx) (pdf)]

Transparent Offloading and Mapping (TOM):
Enabling Programmer-Transparent Near-Data Processing in GPU Systems

Kevin Hsieh[‡] Eiman Ebrahimi[†] Gwangsun Kim[*] Niladrish Chatterjee[†] Mike O'Connor[†]
Nandita Vijaykumar[‡] Onur Mutlu[§‡] Stephen W. Keckler[†]
[‡]Carnegie Mellon University [†]NVIDIA [*]KAIST [§]ETH Zürich

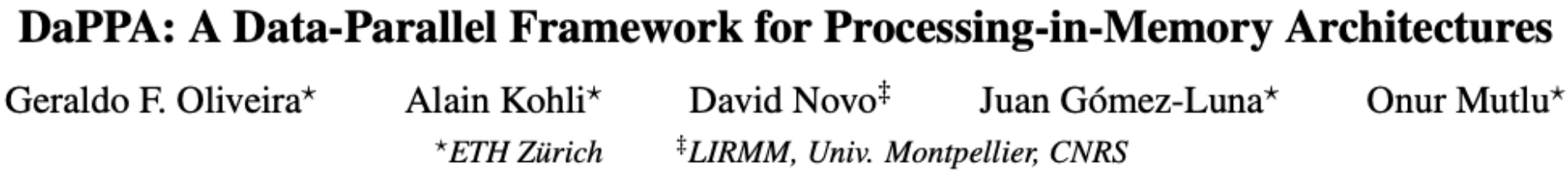
Truly Distributed GPU Processing with PIM

```
__global__
void applyScaleFactorsKernel( uint8_T * const out,
    uint8_T const * const in, const double *factor,
    size_t const numRows, size_t const numCols )
{
    // Work out which pixel we are working on.
    const int rowIdx = blockIdx.x * blockDim.x + threadIdx.x;
    const int colIdx = blockIdx.y;
    const int sliceIdx = threadIdx.z;

    // Check this thread isn't off the image
    if( rowIdx >= numRows ) return;

    // Compute the index of my element
    size_t linearIdx = rowIdx + colIdx*numRows +
        sliceIdx*numRows*numCols;
```



Adoption: How to Ease **Programmability**? (II)

- Geraldo F. Oliveira, Alain Kohli, David Novo, Juan Gómez-Luna, Onur Mutlu, **"DaPPA: A Data-Parallel Framework for Processing-in-Memory Architectures,"** in *PACT SRC Student Competition,* Vienna, Austria, October 2023.



DaPPA: A Data-Parallel Framework for Processing-in-Memory Architectures

Geraldo F. Oliveira⋆ Alain Kohli⋆ David Novo‡ Juan Gómez-Luna⋆ Onur Mutlu⋆

⋆*ETH Zürich* ‡*LIRMM, Univ. Montpellier, CNRS*

Adoption: How to Ease **Programmability**? (III)

- Jinfan Chen, Juan Gómez-Luna, Izzat El Hajj, YuXin Guo, and Onur Mutlu,
  "SimplePIM: A Software Framework for Productive and Efficient Processing in Memory"
  *Proceedings of the 32nd International Conference on Parallel Architectures and Compilation Techniques (**PACT**)*, Vienna, Austria, October 2023.

SimplePIM: A Software Framework for Productive and Efficient Processing-in-Memory

Jinfan Chen[1] Juan Gómez-Luna[1] Izzat El Hajj[2] Yuxin Guo[1] Onur Mutlu[1]
[1]ETH Zürich [2]American University of Beirut

Adoption: How to Ease **Programmability**? (IV)

- Geraldo F. Oliveira, Juan Gomez-Luna, Lois Orosa, Saugata Ghose, Nandita Vijaykumar, Ivan fernandez, Mohammad Sadrosadati, and Onur Mutlu, **"DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks"**
  IEEE Access, 8 September 2021.
  Preprint in ***arXiv***, 8 May 2021.
  [arXiv preprint]
  [IEEE Access version]
  [DAMOV Suite and Simulator Source Code]
  [SAFARI Live Seminar Video (2 hrs 40 mins)]
  [Short Talk Video (21 minutes)]

DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
LOIS OROSA, ETH Zürich, Switzerland
SAUGATA GHOSE, University of Illinois at Urbana–Champaign, USA
NANDITA VIJAYKUMAR, University of Toronto, Canada
IVAN FERNANDEZ, University of Malaga, Spain & ETH Zürich, Switzerland
MOHAMMAD SADROSADATI, ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

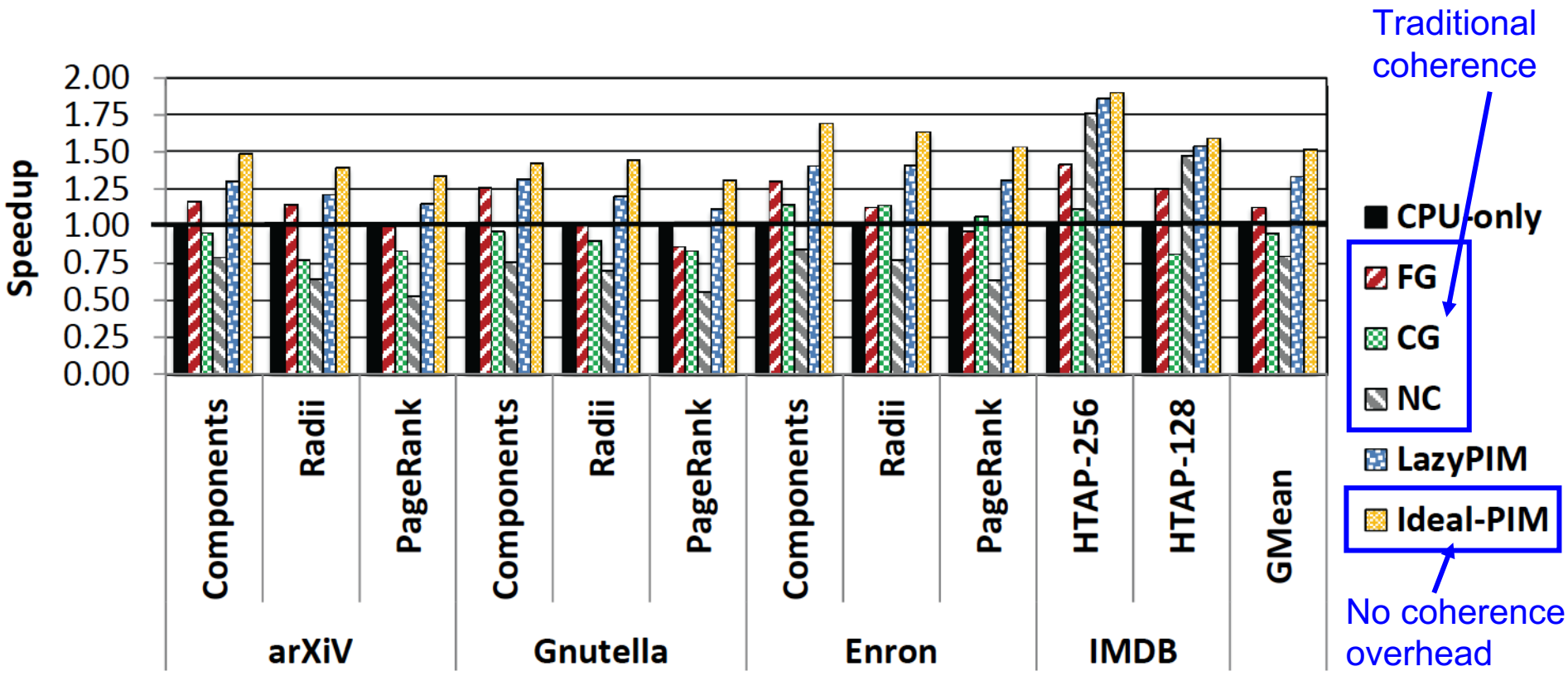
Adoption: How to Maintain **Coherence**? (I)

- Amirali Boroumand, Saugata Ghose, Minesh Patel, Hasan Hassan, Brandon Lucia, Kevin Hsieh, Krishna T. Malladi, Hongzhong Zheng, and Onur Mutlu,
  "LazyPIM: An Efficient Cache Coherence Mechanism for Processing-in-Memory"
  IEEE Computer Architecture Letters (***CAL***), June 2016.

LazyPIM: An Efficient Cache Coherence Mechanism for Processing-in-Memory

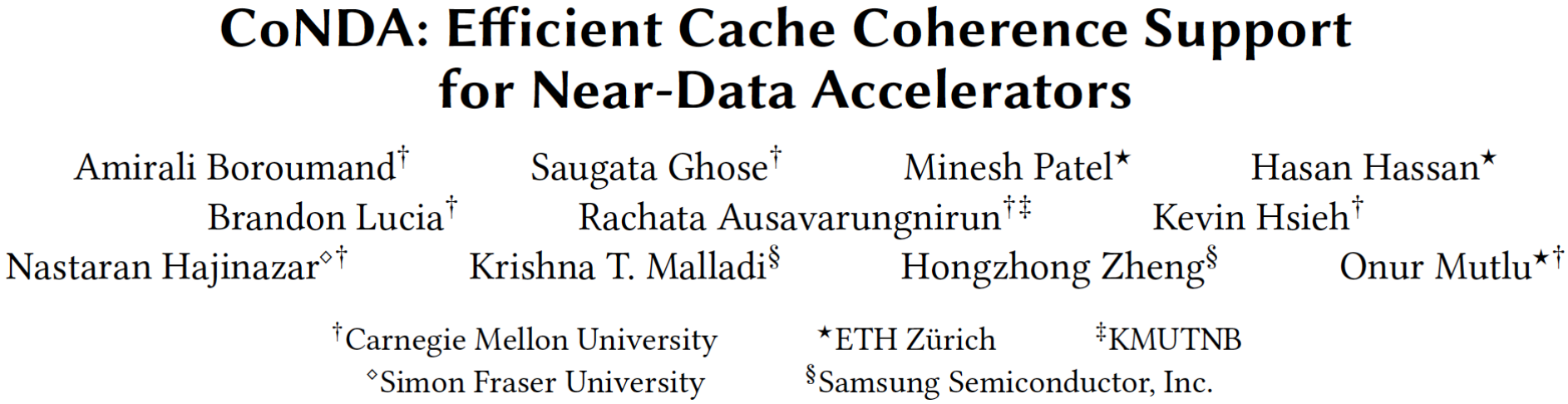
Amirali Boroumand[†], Saugata Ghose[†], Minesh Patel[†], Hasan Hassan[†§], Brandon Lucia[†],
Kevin Hsieh[†], Krishna T. Malladi[⋆], Hongzhong Zheng[⋆], and Onur Mutlu[‡†]
[†]*Carnegie Mellon University* [⋆]*Samsung Semiconductor, Inc.* [§]*TOBB ETÜ* [‡]*ETH Zürich*

Challenge: Coherence for Hybrid CPU-PIM Apps



Adoption: How to Maintain **Coherence**? (II)

- Amirali Boroumand, Saugata Ghose, Minesh Patel, Hasan Hassan, Brandon Lucia, Kevin Hsieh, Krishna T. Malladi, Hongzhong Zheng, and Onur Mutlu,
  "CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators"
  *Proceedings of the 46th International Symposium on Computer Architecture (**ISCA**)*, Phoenix, AZ, USA, June 2019.



CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators

Amirali Boroumand[†] Saugata Ghose[†] Minesh Patel[⋆] Hasan Hassan[⋆]
Brandon Lucia[†] Rachata Ausavarungnirun[†‡] Kevin Hsieh[†]
Nastaran Hajinazar[◇†] Krishna T. Malladi[§] Hongzhong Zheng[§] Onur Mutlu[⋆†]

[†]Carnegie Mellon University [⋆]ETH Zürich [‡]KMUTNB
[◇]Simon Fraser University [§]Samsung Semiconductor, Inc.

Adoption: How to Support **Synchronization**?

- Christina Giannoula, Nandita Vijaykumar, Nikela Papadopoulou, Vasileios Karakostas, Ivan Fernandez, Juan Gómez-Luna, Lois Orosa, Nectarios Koziris, Georgios Goumas, Onur Mutlu, **"SynCron: Efficient Synchronization Support for Near-Data-Processing Architectures"**
  *Proceedings of the 27th International Symposium on High-Performance Computer Architecture (**HPCA**)*, Virtual, February-March 2021.
  [Slides (pptx) (pdf)]
  [Short Talk Slides (pptx) (pdf)]
  [Talk Video (21 minutes)]
  [Short Talk Video (7 minutes)]

SynCron: Efficient Synchronization Support for Near-Data-Processing Architectures

Christina Giannoula[†‡] Nandita Vijaykumar[*‡] Nikela Papadopoulou[†] Vasileios Karakostas[†] Ivan Fernandez[§‡]
Juan Gómez-Luna[‡] Lois Orosa[‡] Nectarios Koziris[†] Georgios Goumas[†] Onur Mutlu[‡]
[†]*National Technical University of Athens* [‡]*ETH Zürich* [*]*University of Toronto* [§]*University of Malaga*

Adoption: How to Support **Virtual Memory**?

- Kevin Hsieh, Samira Khan, Nandita Vijaykumar, Kevin K. Chang, Amirali Boroumand, Saugata Ghose, and Onur Mutlu,
  "Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation"
  *Proceedings of the 34th IEEE International Conference on Computer Design (**ICCD**)*, Phoenix, AZ, USA, October 2016.

Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation

Kevin Hsieh[†] Samira Khan[‡] Nandita Vijaykumar[†]
Kevin K. Chang[†] Amirali Boroumand[†] Saugata Ghose[†] Onur Mutlu[§†]
[†]*Carnegie Mellon University* [‡]*University of Virginia* [§]*ETH Zürich*

Adoption: Evaluation Infrastructures

- Haocong Luo, Yahya Can Tugrul, F. Nisa Bostanci, Ataberk Olgun, A. Giray Yaglikci, and Onur Mutlu,
  "Ramulator 2.0: A Modern, Modular, and Extensible DRAM Simulator"
  Preprint on ***arxiv***, August 2023.
  [arXiv version]
  [Ramulator 2.0 Source Code]

https://arxiv.org/pdf/2308.11030.pdf

https://github.com/CMU-SAFARI/ramulator2

Methodologies, Workloads, and Tools for Processing-in-Memory: Enabling the Adoption of Data-Centric Architectures

Geraldo F. Oliveira and Onur Mutlu

geraldofojunior@gmail.com

https://geraldofojunior.github.io/

SAFARI

ETH zürich

Processing-in-Memory: Challenges

To fully support PIM systems, we need to develop:

1. **Workload characterization methodologies and benchmark suites targeting PIM architectures**
2. **Frameworks that can facilitate the implementation of complex operations and algorithms using PIM primitives**
3. **Compiler support and compiler optimizations targeting PIM architectures**
4. **Operating system support for PIM-aware virtual memory, memory management, data allocation and mapping**
5. **End-to-End System-on-Chip Design Beyond DRAM**

The lack of tools and system support for PIM architectures limit the adoption of PIM systems

An Example: SimplePIM Framework

- Jinfan Chen, Juan Gómez-Luna, Izzat El Hajj, YuXin Guo, and Onur Mutlu,
  "SimplePIM: A Software Framework for Productive and Efficient Processing in Memory"
  *Proceedings of the 32nd International Conference on Parallel Architectures and Compilation Techniques (**PACT**)*, Vienna, Austria, October 2023.

SimplePIM: A Software Framework for Productive and Efficient Processing-in-Memory

Jinfan Chen[1] Juan Gómez-Luna[1] Izzat El Hajj[2] Yuxin Guo[1] Onur Mutlu[1]
[1]ETH Zürich [2]American University of Beirut

Executive Summary

- Real PIM hardware is now available, e.g., UPMEM PIM
- However, programming real PIM hardware is challenging, e.g., need to:
  - Distribute data across PIM memory banks,
  - Manage data transfers between host cores and PIM cores, between PIM cores, and between DRAM bank and PIM scratchpad
  - Launch PIM kernels on the PIM cores, etc.
  - Synchronize properly between threads
- SimplePIM is a high-level programming framework for real PIM hardware
  - Iterators such as `map`, `reduce`, and `zip`
  - Collective communication with `broadcast`, `scatter`, and `gather`
- Implementation on UPMEM and evaluation with six different workloads
  - Reduction, vector add, histogram, linear/logistic regression, K-means
  - **4.4x fewer lines of code compared to hand-optimized code**
  - Between **15% and 43% faster than hand-optimized code** for three workloads

- Source code: https://github.com/CMU-SAFARI/SimplePIM

The SimplePIM Programming Framework

Our Goal

Design a high-level programming framework that abstracts hardware-specific complexities and provides a clean yet powerful interface for ease of use and high program performance

- SimplePIM provides standard abstractions to build and deploy applications on PIM systems
  - Management interface
    - Metadata management for PIM-resident arrays
  - Communication interface
    - Abstractions for host-PIM and PIM-PIM communication
    - Collective communication with `broadcast`, `scatter`, and `gather`
  - Processing interface
    - Iterators (`map`, `reduce`, `zip`) to implement workloads

SimpePIM: General Code Optimizations

- Strength reduction
- Loop unrolling
- Avoiding boundary checks
- Function inlining
- Adjustment of data transfer sizes

More in the Paper

SimplePIM: A Software Framework for Productive and Efficient Processing-in-Memory

Jinfan Chen[1] Juan Gómez-Luna[1] Izzat El Hajj[2] Yuxin Guo[1] Onur Mutlu[1]
[1]ETH Zürich [2]American University of Beirut

https://arxiv.org/pdf/2310.01893.pdf

Productivity Improvement (I)

- Example: Hand-optimized histogram with UPMEM SDK

```
... // Initialize global variables and functions for histogram
int main_kernel() {
  if (tasklet_id == 0)
    mem_reset(); // Reset the heap
  ... // Initialize variables and the histogram
  T *input_buff_A = (T*)mem_alloc(2048); // Allocate buffer in scratchpad memory

  for (unsigned int byte_index = base_tasklet; byte_index < input_size; byte_index += stride) {
    // Boundary checking
    uint32_t l_size_bytes = (byte_index + 2048 >= input_size) ? (input_size - byte_index) : 2048;
    // Load scratchpad with a DRAM block
    mram_read((const __mram_ptr void*)(mram_base_addr_A + byte_index), input_buff_A, l_size_bytes);
    // Histogram calculation
    histogram(hist, bins, input_buff_A, l_size_bytes/sizeof(uint32_t));
  }
  ...
  barrier_wait(&my_barrier); // Barrier to synchronize PIM threads
  ... // Merging histograms from different tasklets into one histo_dpu
  // Write result from scratchpad to DRAM
  if (tasklet_id == 0)
    if (bins * sizeof(uint32_t) <= 2048)
      mram_write(histo_dpu, (__mram_ptr void*)mram_base_addr_histo, bins * sizeof(uint32_t));
    else
      for (unsigned int offset = 0; offset < ((bins * sizeof(uint32_t)) >> 11); offset++) {
        mram_write(histo_dpu + (offset << 9), (__mram_ptr void*)(mram_base_addr_histo +
                   (offset << 11)), 2048);
      }
  return 0;
}
```

Productivity Improvement (II)

- Example: SimplePIM histogram

```
// Programmer-defined functions in the file "histo_filepath"
void init_func (uint32_t size, void* ptr) {
  char* casted_value_ptr = (char*) ptr;
  for (int i = 0; i < size; i++)
    casted_value_ptr[i] = 0;
}

void acc_func (void* dest, void* src) {
  *(uint32_t*)dest += *(uint32_t*)src;
}

void map_to_val_func (void* input, void* output, uint32_t* key) {
  uint32_t d = *((uint32_t*)input);
  *(uint32_t*)output = 1;
  *key = d * bins >> 12;
}

// Host side handle creation and iterator call
handle_t* handle = simple_pim_create_handle("histo_filepath", REDUCE, NULL, 0);

// Transfer (scatter) data to PIM, register as "t1"
simple_pim_array_scatter("t1", src, bins, sizeof(T), management);

// Run histogram on "t1" and produce "t2"
simple_pim_array_red("t1", "t2", sizeof(T), bins, handle, management);
```

Productivity Improvement (III)

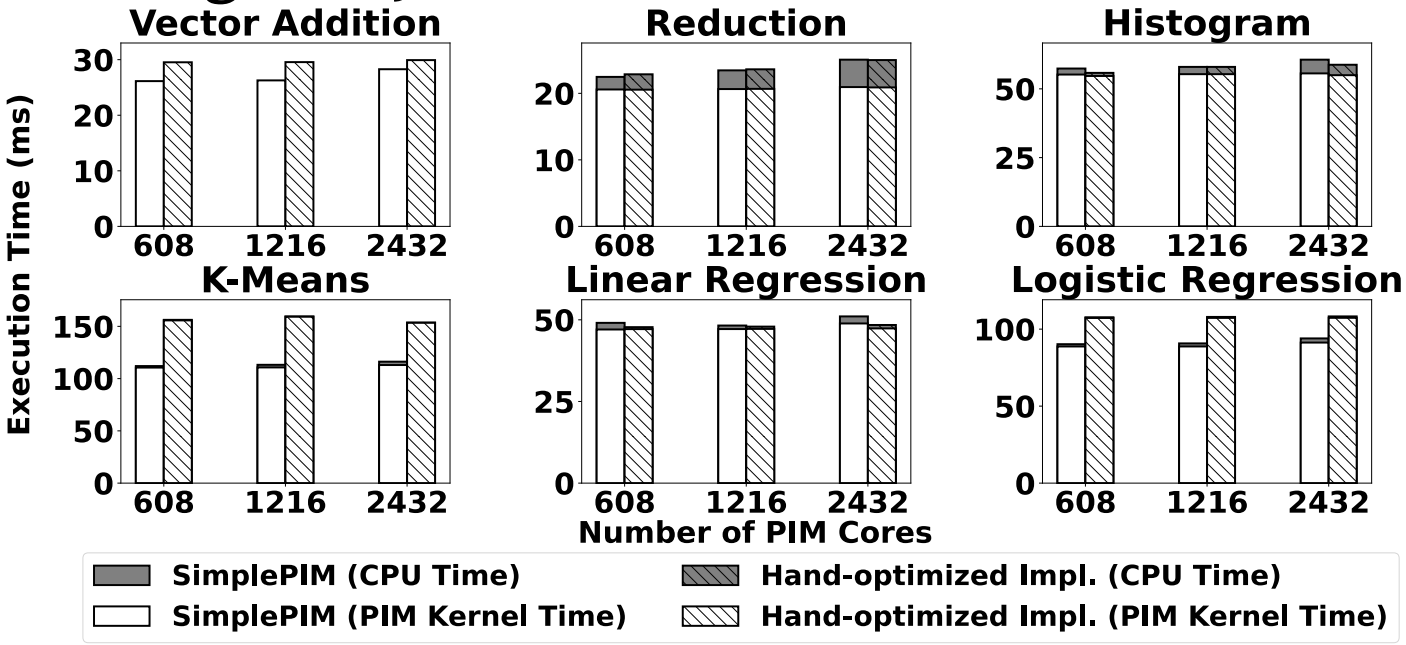
- Lines of code (LoC) reduction

	SimplePIM	Hand-optimized	LoC Reduction
Reduction	14	83	5.93×
Vector Addition	14	82	5.86×
Histogram	21	114	5.43×
Linear Regression	48	157	3.27×
Logistic Regression	59	176	2.98×
K-Means	68	206	3.03×

SimplePIM reduces the number of lines of effective code by a factor of 2.98× to 5.93×

Performance Evaluation

- Weak scaling analysis

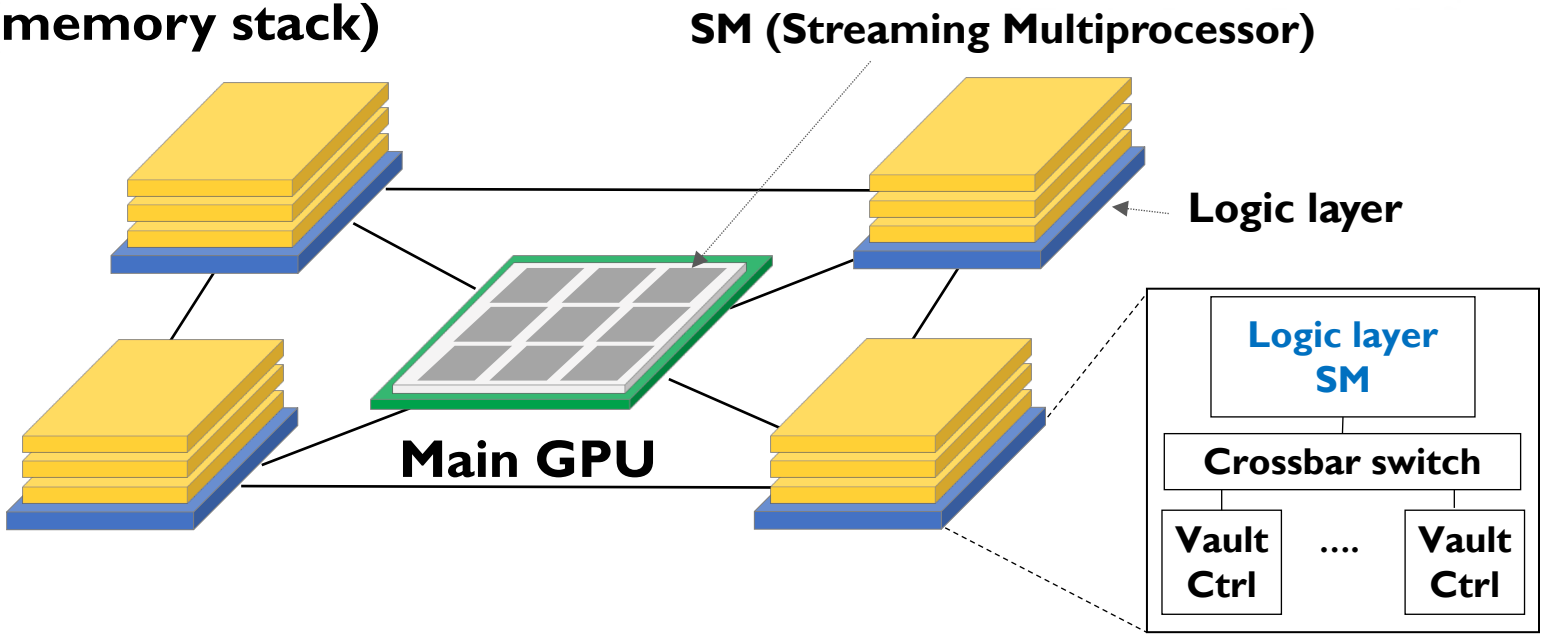


SimplePIM achieves comparable performance for `reduction`, `histogram`, and `linear regression`

SimplePIM outperforms hand-optimized implementations for `vector addition`, `logistic regression`, and `k-means` by 10%-37%

Truly Distributed GPU Processing with PIM

```
__global__
void applyScaleFactorsKernel( uint8_T * const out,
    uint8_T const * const in, const double *factor,
    size_t const numRows, size_t const numCols )
{
    // Work out which pixel we are working on.
    const int rowIdx = blockIdx.x * blockDim.x + threadIdx.x;
    const int colIdx = blockIdx.y;
    const int sliceIdx = threadIdx.z;

    // Check this thread isn't off the image
    if( rowIdx >= numRows ) return;

    // Compute the index of my element
    size_t linearIdx = rowIdx + colIdx*numRows +
        sliceIdx*numRows*numCols;
```



Accelerating GPU Execution with PIM (I)

- Kevin Hsieh, Eiman Ebrahimi, Gwangsun Kim, Niladrish Chatterjee, Mike O'Connor, Nandita Vijaykumar, Onur Mutlu, and Stephen W. Keckler, **"Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems"**
  *Proceedings of the 43rd International Symposium on Computer Architecture (**ISCA**)*, Seoul, South Korea, June 2016.
  [Slides (pptx) (pdf)]
  [Lightning Session Slides (pptx) (pdf)]

Transparent Offloading and Mapping (TOM):
Enabling Programmer-Transparent Near-Data Processing in GPU Systems

Kevin Hsieh[‡] Eiman Ebrahimi[†] Gwangsun Kim[*] Niladrish Chatterjee[†] Mike O'Connor[†]
Nandita Vijaykumar[‡] Onur Mutlu[§‡] Stephen W. Keckler[†]
[‡]Carnegie Mellon University [†]NVIDIA [*]KAIST [§]ETH Zürich

Accelerating GPU Execution with PIM (II)

- Ashutosh Pattnaik, Xulong Tang, Adwait Jog, Onur Kayiran, Asit K. Mishra, Mahmut T. Kandemir, Onur Mutlu, and Chita R. Das, **"Scheduling Techniques for GPU Architectures with Processing-In-Memory Capabilities"**
  *Proceedings of the 25th International Conference on Parallel Architectures and Compilation Techniques (**PACT**)*, Haifa, Israel, September 2016.

Scheduling Techniques for GPU Architectures with Processing-In-Memory Capabilities

Ashutosh Pattnaik[1] Xulong Tang[1] Adwait Jog[2] Onur Kayıran[3]
Asit K. Mishra[4] Mahmut T. Kandemir[1] Onur Mutlu[5,6] Chita R. Das[1]
[1]Pennsylvania State University [2]College of William and Mary
[3]Advanced Micro Devices, Inc. [4]Intel Labs [5]ETH Zürich [6]Carnegie Mellon University

Accelerating Linked Data Structures

- Kevin Hsieh, Samira Khan, Nandita Vijaykumar, Kevin K. Chang, Amirali Boroumand, Saugata Ghose, and Onur Mutlu,
  "Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation"
  *Proceedings of the 34th IEEE International Conference on Computer Design (**ICCD**)*, Phoenix, AZ, USA, October 2016.


Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation

Kevin Hsieh[†] Samira Khan[‡] Nandita Vijaykumar[†]
Kevin K. Chang[†] Amirali Boroumand[†] Saugata Ghose[†] Onur Mutlu[§†]
[†]*Carnegie Mellon University* [‡]*University of Virginia* [§]*ETH Zürich*

Accelerating Dependent Cache Misses

- Milad Hashemi, Khubaib, Eiman Ebrahimi, Onur Mutlu, and Yale N. Patt, **"Accelerating Dependent Cache Misses with an Enhanced Memory Controller"**
  *Proceedings of the 43rd International Symposium on Computer Architecture (**ISCA**)*, Seoul, South Korea, June 2016.
  [Slides (pptx) (pdf)]
  [Lightning Session Slides (pptx) (pdf)]

Accelerating Dependent Cache Misses with an Enhanced Memory Controller

Milad Hashemi*, Khubaib†, Eiman Ebrahimi‡, Onur Mutlu§, Yale N. Patt*

*The University of Texas at Austin †Apple ‡NVIDIA §ETH Zürich & Carnegie Mellon University

Accelerating Runahead Execution

- Milad Hashemi, Onur Mutlu, and Yale N. Patt,
  "Continuous Runahead: Transparent Hardware Acceleration for Memory Intensive Workloads"
  *Proceedings of the 49th International Symposium on Microarchitecture (**MICRO**)*, Taipei, Taiwan, October 2016.
  [Slides (pptx) (pdf)] [Lightning Session Slides (pdf)] [Poster (pptx) (pdf)]
  Best paper session.

Continuous Runahead: Transparent Hardware Acceleration for Memory Intensive Workloads

Milad Hashemi*, Onur Mutlu§, Yale N. Patt*

*The University of Texas at Austin §ETH Zürich

Accelerating Climate Modeling

- Gagandeep Singh, Dionysios Diamantopoulos, Christoph Hagleitner, Juan Gómez-Luna, Sander Stuijk, Onur Mutlu, and Henk Corporaal,
  "NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling"
  *Proceedings of the 30th International Conference on Field-Programmable Logic and Applications (**FPL**)*, Gothenburg, Sweden, September 2020.
  [Slides (pptx) (pdf)]
  [Lightning Talk Slides (pptx) (pdf)]
  [Talk Video (23 minutes)]
  Nominated for the Stamatis Vassiliadis Memorial Award.

NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling

Gagandeep Singh[a,b,c] Dionysios Diamantopoulos[c] Christoph Hagleitner[c] Juan Gómez-Luna[b]
Sander Stuijk[a] Onur Mutlu[b] Henk Corporaal[a]
[a]Eindhoven University of Technology [b]ETH Zürich [c]IBM Research Europe, Zurich

Accelerating DNA Read Mapping

- Jeremie S. Kim, Damla Senol Cali, Hongyi Xin, Donghyuk Lee, Saugata Ghose, Mohammed Alser, Hasan Hassan, Oguz Ergin, Can Alkan, and Onur Mutlu,
  "GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies"
  BMC Genomics, 2018.
  *Proceedings of the 16th Asia Pacific Bioinformatics Conference (**APBC**)*, Yokohama, Japan, January 2018.
  [Slides (pptx) (pdf)]
  [Source Code]
  [arxiv.org Version (pdf)]
  [Talk Video at AACBB 2019]

GRIM-Filter: Fast seed location filtering in DNA read mapping using processing-in-memory technologies

Jeremie S. Kim[1,6*], Damla Senol Cali[1], Hongyi Xin[2], Donghyuk Lee[3], Saugata Ghose[1], Mohammed Alser[4], Hasan Hassan[6], Oguz Ergin[5], Can Alkan[4*] and Onur Mutlu[6,1*]

From The Sixteenth Asia Pacific Bioinformatics Conference 2018
Yokohama, Japan. 15-17 January 2018

Accelerating Approximate String Matching

- Damla Senol Cali, Gurpreet S. Kalsi, Zulal Bingol, Can Firtina, Lavanya Subramanian, Jeremie S. Kim, Rachata Ausavarungnirun, Mohammed Alser, Juan Gomez-Luna, Amirali Boroumand, Anant Nori, Allison Scibisz, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu, **"GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis"**
  *Proceedings of the 53rd International Symposium on Microarchitecture (**MICRO**)*, Virtual, October 2020.
  [Lighting Talk Video (1.5 minutes)]
  [Lightning Talk Slides (pptx) (pdf)]
  [Talk Video (18 minutes)]
  [Slides (pptx) (pdf)]

GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis

Damla Senol Cali[†⋈] Gurpreet S. Kalsi[⋈] Zülal Bingöl[▽] Can Firtina[◇] Lavanya Subramanian[‡] Jeremie S. Kim[◇†]
Rachata Ausavarungnirun[⊙] Mohammed Alser[◇] Juan Gomez-Luna[◇] Amirali Boroumand[†] Anant Nori[⋈]
Allison Scibisz[†] Sreenivas Subramoney[⋈] Can Alkan[▽] Saugata Ghose[⋆†] Onur Mutlu[◇†▽]

[†]*Carnegie Mellon University* [⋈]*Processor Architecture Research Lab, Intel Labs* [▽]*Bilkent University* [◇]*ETH Zürich*
[‡]*Facebook* [⊙]*King Mongkut's University of Technology North Bangkok* [⋆]*University of Illinois at Urbana–Champaign*

Accelerating Sequence-to-Graph Mapping

- Damla Senol Cali, Konstantinos Kanellopoulos, Joel Lindegger, Zulal Bingol, Gurpreet S. Kalsi, Ziyi Zuo, Can Firtina, Meryem Banu Cavlak, Jeremie Kim, Nika MansouriGhiasi, Gagandeep Singh, Juan Gomez-Luna, Nour Almadhoun Alserr, Mohammed Alser, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu,
  "SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping"
  *Proceedings of the 49th International Symposium on Computer Architecture (**ISCA**)*, New York, June 2022.
  [arXiv version]

SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping

Damla Senol Cali[1] Konstantinos Kanellopoulos[2] Joël Lindegger[2] Zülal Bingöl[3]
Gurpreet S. Kalsi[4] Ziyi Zuo[5] Can Firtina[2] Meryem Banu Cavlak[2] Jeremie Kim[2]
Nika Mansouri Ghiasi[2] Gagandeep Singh[2] Juan Gómez-Luna[2] Nour Almadhoun Alserr[2]
Mohammed Alser[2] Sreenivas Subramoney[4] Can Alkan[3] Saugata Ghose[6] Onur Mutlu[2]

[1]Bionano Genomics [2]ETH Zürich [3]Bilkent University [4]Intel Labs
[5]Carnegie Mellon University [6]University of Illinois Urbana-Champaign

Accelerating Basecalling + Read Mapping

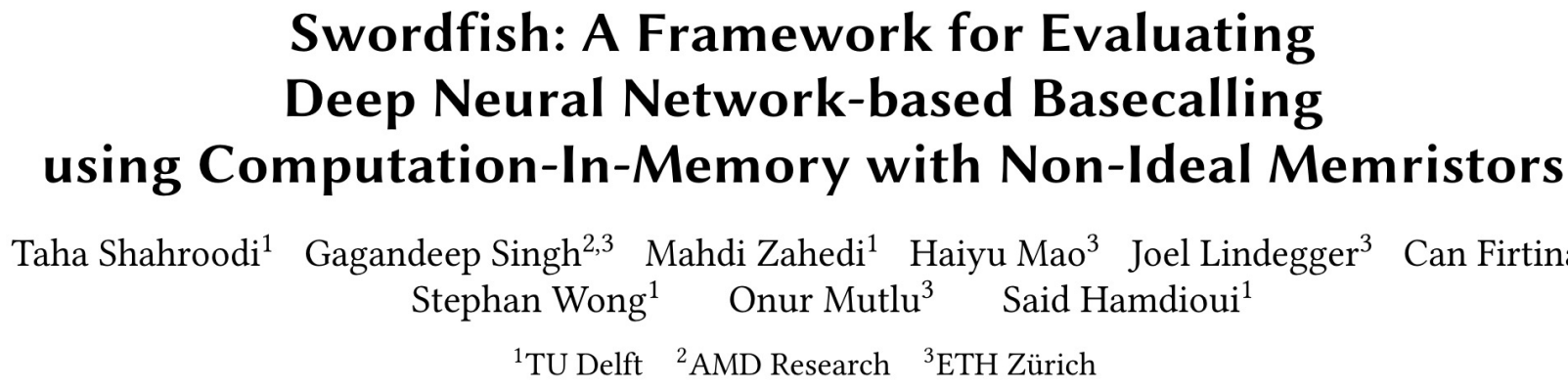
- Haiyu Mao, Mohammed Alser, Mohammad Sadrosadati, Can Firtina, Akanksha Baranwal, Damla Senol Cali, Aditya Manglik, Nour Almadhoun Alserr, and Onur Mutlu,
  "GenPIP: In-Memory Acceleration of Genome Analysis via Tight Integration of Basecalling and Read Mapping"
  Proceedings of the 55th International Symposium on Microarchitecture (***MICRO***),
  Chicago, IL, USA, October 2022.
  [Slides (pptx) (pdf)]
  [Longer Lecture Slides (pptx) (pdf)]
  [Lecture Video (25 minutes)]
  [arXiv version]

GenPIP: In-Memory Acceleration of Genome Analysis via Tight Integration of Basecalling and Read Mapping

Haiyu Mao[1] Mohammed Alser[1] Mohammad Sadrosadati[1] Can Firtina[1] Akanksha Baranwal[1]
Damla Senol Cali[2] Aditya Manglik[1] Nour Almadhoun Alserr[1] Onur Mutlu[1]
[1]*ETH Zürich* [2]*Bionano Genomics*

Accelerating Basecalling

- Taha Shahroodi, Gagandeep Singh, Mahdi Zahedi, Haiyu Mao, Joel Lindegger, Can Firtina, Stephan Wong, Onur Mutlu, and Said Hamdioui,
  "Swordfish: A Framework for Evaluating Deep Neural Network-based Basecalling using Computation-In-Memory with Non-Ideal Memristors"
  *Proceedings of the 56th International Symposium on Microarchitecture (**MICRO**)*, Toronto, ON, Canada, November 2023.
  [Slides (pptx) (pdf)]
  [arXiv version]



Swordfish: A Framework for Evaluating Deep Neural Network-based Basecalling using Computation-In-Memory with Non-Ideal Memristors

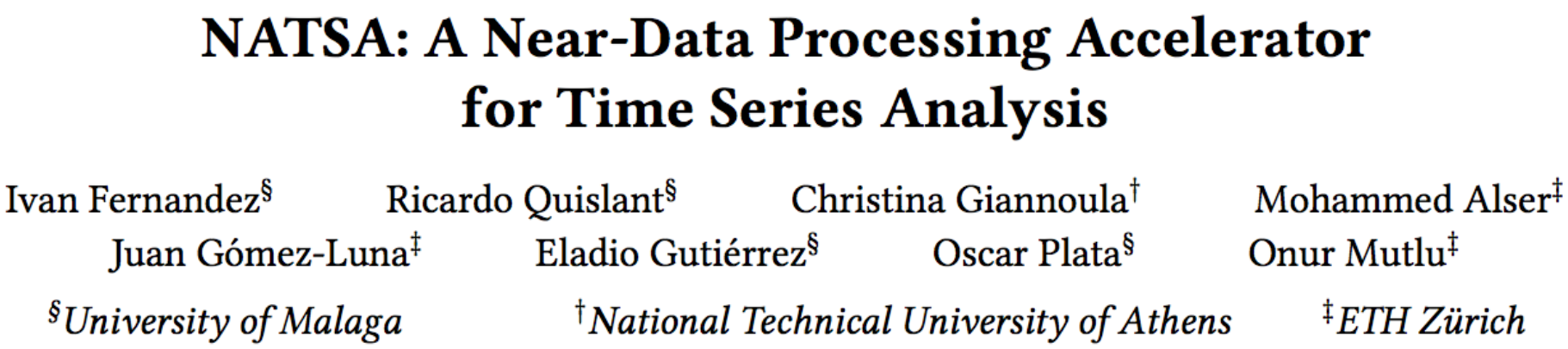
Taha Shahroodi[1] Gagandeep Singh[2,3] Mahdi Zahedi[1] Haiyu Mao[3] Joel Lindegger[3] Can Firtina[3]
Stephan Wong[1] Onur Mutlu[3] Said Hamdioui[1]

[1]TU Delft [2]AMD Research [3]ETH Zürich

https://arxiv.org/pdf/2310.04366

Accelerating Time Series Analysis (I)

- Ivan Fernandez, Ricardo Quislant, Christina Giannoula, Mohammed Alser, Juan Gómez-Luna, Eladio Gutiérrez, Oscar Plata, and Onur Mutlu,
  "NATSA: A Near-Data Processing Accelerator for Time Series Analysis"
  *Proceedings of the 38th IEEE International Conference on Computer Design (**ICCD**)*, Virtual, October 2020.
  [Slides (pptx) (pdf)]
  [Talk Video (10 minutes)]
  [Source Code]



NATSA: A Near-Data Processing Accelerator for Time Series Analysis

Ivan Fernandez[§] Ricardo Quislant[§] Christina Giannoula[†] Mohammed Alser[‡]
Juan Gómez-Luna[‡] Eladio Gutiérrez[§] Oscar Plata[§] Onur Mutlu[‡]

[§]*University of Malaga* [†]*National Technical University of Athens* [‡]*ETH Zürich*

https://arxiv.org/pdf/2010.02079

Accelerating Time Series Analysis (II)

- Ivan Fernandez, Christina Giannoula, Aditya Manglik, Ricardo Quislant, Nika Mansouri Ghiasi, Juan Gomez Luna, Eladio Gutierrez, Oscar Plata and Onur Mutlu,
  "MATSA: An MRAM-Based Energy-Efficient Accelerator for Time Series Analysis"
  IEEE Access, March 2024.
  [arXiv version]
  [IEEE Access version]

Accelerating Time Series Analysis via Processing using Non-Volatile Memories

Ivan Fernandez[§†¶] *Christina Giannoula[†‡] *Aditya Manglik[†] Ricardo Quislant[§] Nika Mansouri Ghiasi[†]
Juan Gómez-Luna[†] Eladio Gutierrez[§] Oscar Plata[§] Onur Mutlu[†]
[§]*University of Malaga* [†]*ETH Zürich* [¶]*Barcelona Supercomputing Center* [‡]*National Technical University of Athens*

https://arxiv.org/pdf/2211.04369

Accelerating Graph Pattern Mining

- Maciej Besta, Raghavendra Kanakagiri, Grzegorz Kwasniewski, Rachata Ausavarungnirun, Jakub Beránek, Konstantinos Kanellopoulos, Kacper Janda, Zur Vonarburg-Shmaria, Lukas Gianinazzi, Ioana Stefan, Juan Gómez-Luna, Marcin Copik, Lukas Kapp-Schwoerer, Salvatore Di Girolamo, Nils Blach, Marek Konieczny, Onur Mutlu, and Torsten Hoefler,
  "SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems"
  Proceedings of the 54th International Symposium on Microarchitecture (***MICRO***), Virtual, October 2021.
  [Slides (pdf)]
  [Talk Video (22 minutes)]
  [Lightning Talk Video (1.5 minutes)]
  [Full arXiv version]

SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems

Maciej Besta[1], Raghavendra Kanakagiri[2], Grzegorz Kwasniewski[1], Rachata Ausavarungnirun[3], Jakub Beránek[4], Konstantinos Kanellopoulos[1], Kacper Janda[5], Zur Vonarburg-Shmaria[1], Lukas Gianinazzi[1], Ioana Stefan[1], Juan Gómez-Luna[1], Marcin Copik[1], Lukas Kapp-Schwoerer[1], Salvatore Di Girolamo[1], Nils Blach[1], Marek Konieczny[5], Onur Mutlu[1], Torsten Hoefler[1]

[1]ETH Zurich, Switzerland [2]IIT Tirupati, India [3]King Mongkut's University of Technology North Bangkok, Thailand [4]Technical University of Ostrava, Czech Republic [5]AGH-UST, Poland

https://arxiv.org/pdf/2104.07582

Accelerating HTAP Database Systems

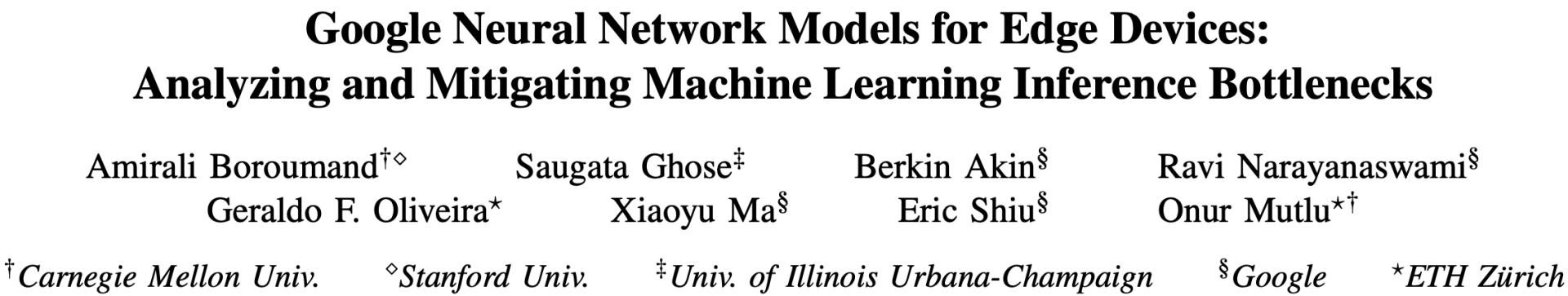
- Amirali Boroumand, Saugata Ghose, Geraldo F. Oliveira, and Onur Mutlu, **"Polynesia: Enabling High-Performance and Energy-Efficient Hybrid Transactional/Analytical Databases with Hardware/Software Co-Design"** *Proceedings of the 38th International Conference on Data Engineering* (***ICDE***), Virtual, May 2022.
  [arXiv version]
  [Slides (pptx) (pdf)]
  [Short Talk Slides (pptx) (pdf)]

Polynesia: Enabling High-Performance and Energy-Efficient Hybrid Transactional/Analytical Databases with Hardware/Software Co-Design

Amirali Boroumand[†] Saugata Ghose[◇] Geraldo F. Oliveira[‡] Onur Mutlu[‡]
[†]*Google* [◇]*Univ. of Illinois Urbana-Champaign* [‡]*ETH Zürich*

Accelerating ML Inference

- Amirali Boroumand, Saugata Ghose, Berkin Akin, Ravi Narayanaswami, Geraldo F. Oliveira, Xiaoyu Ma, Eric Shiu, and Onur Mutlu,
  "Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks"
  *Proceedings of the 30th International Conference on Parallel Architectures and Compilation Techniques (**PACT**)*, Virtual, September 2021.
  [Slides (pptx) (pdf)]
  [Talk Video (14 minutes)]



Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand[†◇] Saugata Ghose[‡] Berkin Akin[§] Ravi Narayanaswami[§]
Geraldo F. Oliveira[⋆] Xiaoyu Ma[§] Eric Shiu[§] Onur Mutlu[⋆†]

[†]*Carnegie Mellon Univ.* [◇]*Stanford Univ.* [‡]*Univ. of Illinois Urbana-Champaign* [§]*Google* [⋆]*ETH Zürich*

Accelerating Data-Intensive Workloads

- Junwhan Ahn, Sungjoo Yoo, Onur Mutlu, and Kiyoung Choi, **"PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture"** *Proceedings of the 42nd International Symposium on Computer Architecture (**ISCA**)*, Portland, OR, June 2015. [Slides (pdf)] [Lightning Session Slides (pdf)]

PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture

Junwhan Ahn Sungjoo Yoo Onur Mutlu† Kiyoung Choi
junwhan@snu.ac.kr, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr
Seoul National University †Carnegie Mellon University

FPGA-based Processing Near Memory

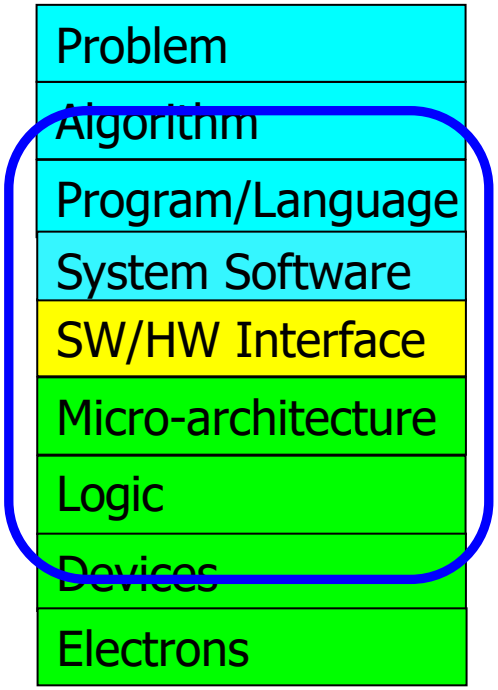
- Gagandeep Singh, Mohammed Alser, Damla Senol Cali, Dionysios Diamantopoulos, Juan Gómez-Luna, Henk Corporaal, and Onur Mutlu, **"FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications"**
  IEEE Micro (***IEEE MICRO***), 2021.

FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications

Gagandeep Singh[◇] **Mohammed Alser**[◇] **Damla Senol Cali**[⋈]
Dionysios Diamantopoulos[▽] **Juan Gómez-Luna**[◇]
Henk Corporaal[⋆] **Onur Mutlu**[◇⋈]

[◇]*ETH Zürich* [⋈]*Carnegie Mellon University*
[⋆]*Eindhoven University of Technology* [▽]*IBM Research Europe*

We Need to Revisit the Entire Stack



We can get there step by step

Security Issues in Processing in Memory

- Does PIM make security better or easier?
- Does PIM make security worse?
- Many interesting questions here
- Some recent papers:
  - Evaluating Homomorphic Operations on a Real-World Processing-In-Memory System **[IISWC 2023]**
  - Amplifying Main Memory-Based Timing Covert and Side Channels using Processing-in-Memory Operations **[arxiv 2024]**

Homomorphic Operations on Real PIM Systems

- Harshita Gupta, Mayank Kabra, Juan Gómez-Luna, Konstantinos Kanellopoulos, and Onur Mutlu,
  "Evaluating Homomorphic Operations on a Real-World Processing-In-Memory System"
  *Proceedings of the 2023 IEEE International Symposium on Workload Characterization Poster Session (**IISWC**)*, Ghent, Belgium, October 2023.
  [arXiv version]
  [Lightning Talk Slides (pptx) (pdf)]
  [Poster (pptx) (pdf)]

Evaluating Homomorphic Operations on a Real-World Processing-In-Memory System

Harshita Gupta* Mayank Kabra* Juan Gómez-Luna Konstantinos Kanellopoulos Onur Mutlu
ETH Zürich

PIM Amplifies Covert & Side Channels

Amplifying Main Memory-Based Timing Covert and Side Channels using Processing-in-Memory Operations

Konstantinos Kanellopoulos[†*] F. Nisa Bostancı[†*] Ataberk Olgun[†]
A. Giray Yağlıkçı[†] İsmail Emir Yüksel[†] Nika Mansouri Ghiasi[†]
Zülal Bingöl[†‡] Mohammad Sadrosadati[†] Onur Mutlu[†]

[†]ETH Zürich [‡]Bilkent University

https://arxiv.org/pdf/2404.11284

A Talk on Security of PIM Systems

https://www.youtube.com/watch?v=UjE9hygFXEM

Eliminating the Adoption Barriers

Processing-in-Memory in the Real World

PIM Tutorial at ISCA 2024

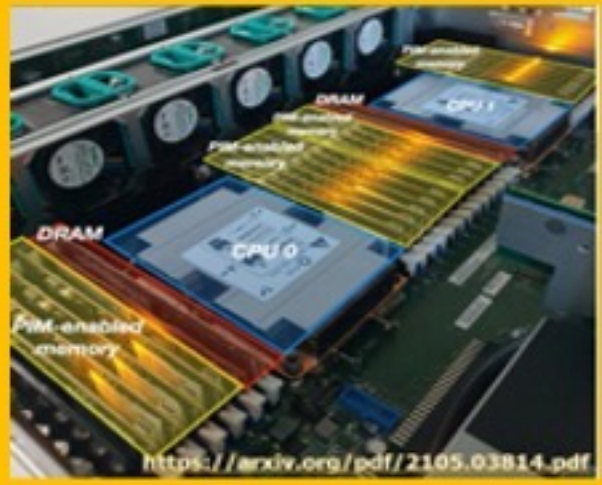
ISCA 2024 Memory-Centric Computing Systems Tutorial

Saturday, June 29, Buenos Aires, Argentina

Organizers: Geraldo F. Oliveira, Dr. Mohammad Sadrosadati, Ataberk Olgun, Professor Onur Mutlu

Program: https://events.safari.ethz.ch/isca24-memorycentric-tutorial/

- Overview of PIM | PIM taxonomy
- PIM in memory & storage
- Real-world PNM systems
- PUM for bulk bitwise operations
- Programming techniques & tools
- Infrastructures for PIM Research
- Research challenges & opportunities

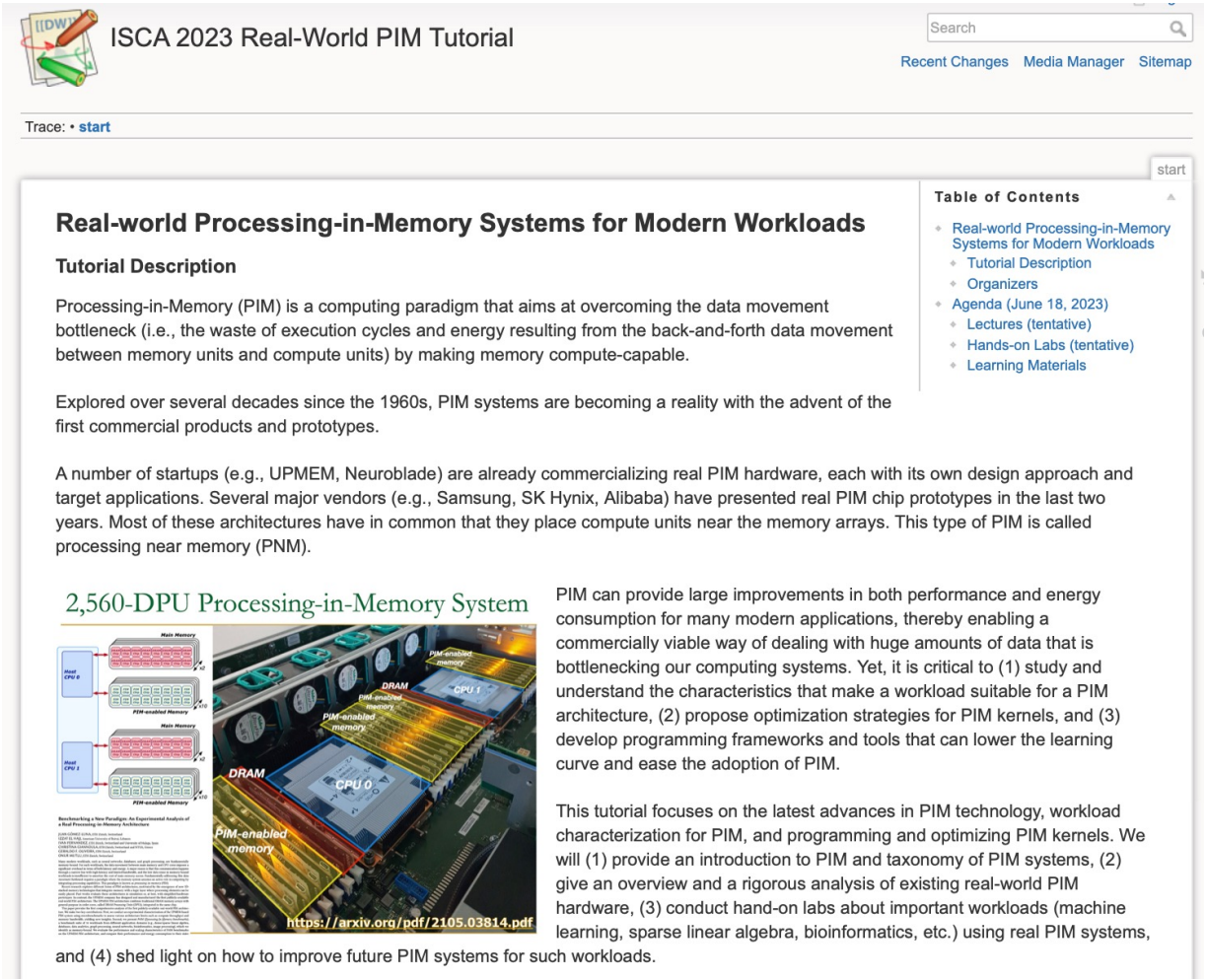


https://www.youtube.com/watch?v=KV2MXvcBgb0

https://events.safari.ethz.ch/isca24-memorycentric-tutorial

PIM Tutorials [MICRO'23, ISCA'23, ASPLOS'23, HPCA'23, ISCA'24]

- Lectures + Hands-on labs + Invited talks



https://www.youtube.com/live/GIb5EgSrWk0

https://events.safari.ethz.ch/isca-pim-tutorial/

Real PIM Tutorial [ISCA 2023]

- June 18: Lectures + Hands-on labs + Invited talks

Tutorial Materials

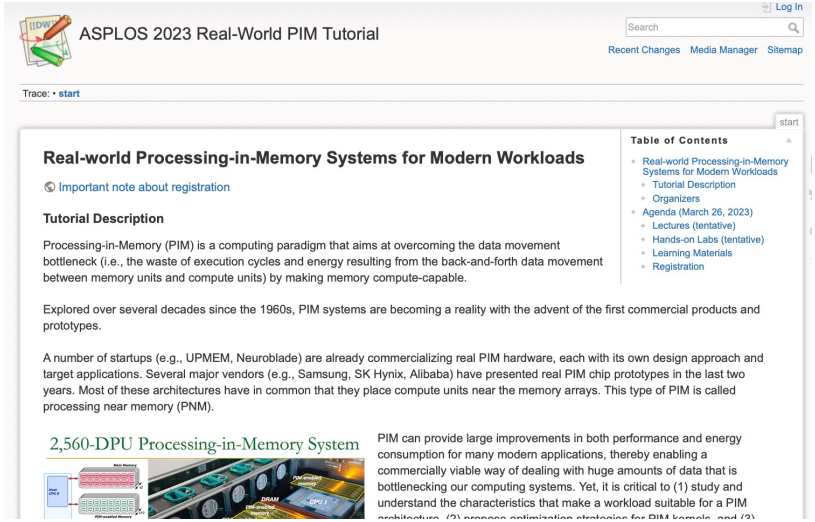
Time	Speaker	Title	Materials
8:55am-9:00am	Dr. Juan Gómez Luna	Welcome & Agenda	(PDF) (PPT)
9:00am-10:20am	Prof. Onur Mutlu	Memory-Centric Computing	(PDF) (PPT)
10:20am-11:00am	Dr. Juan Gómez Luna	Processing-Near-Memory: Real PNM Architectures / Programming General-purpose PIM	(PDF) (PPT)
11:20am-11:50am	Prof. Izzat El Hajj	High-throughput Sequence Alignment using Real Processing-in-Memory Systems	(PDF) (PPT)
11:50am-12:30pm	Dr. Christina Giannoula	SparseP: Towards Efficient Sparse Matrix Vector Multiplication for Real Processing-In-Memory Systems	(PDF) (PPT)
2:00pm-2:45pm	Dr. Sukhan Lee	Introducing Real-world HBM-PIM Powered System for Memory-bound Applications	(PDF) (PPT)
2:45pm-3:30pm	Dr. Juan Gómez Luna / Ataberk Olgun	Processing-Using-Memory: Exploiting the Analog Operational Properties of Memory Components / PUM Prototypes: PiDRAM	(PDF) (PPT) (PDF) (PPT)
4:00pm-4:40pm	Dr. Juan Gómez Luna	Accelerating Modern Workloads on a General-purpose PIM System	(PDF) (PPT)
4:40pm-5:20pm	Dr. Juan Gómez Luna	Adoption Issues: How to Enable PIM?	(PDF) (PPT)
5:20pm-5:30pm	Dr. Juan Gómez Luna	Hands-on Lab: Programming and Understanding a Real Processing-in-Memory Architecture	(Handout) (PDF) (PPT)

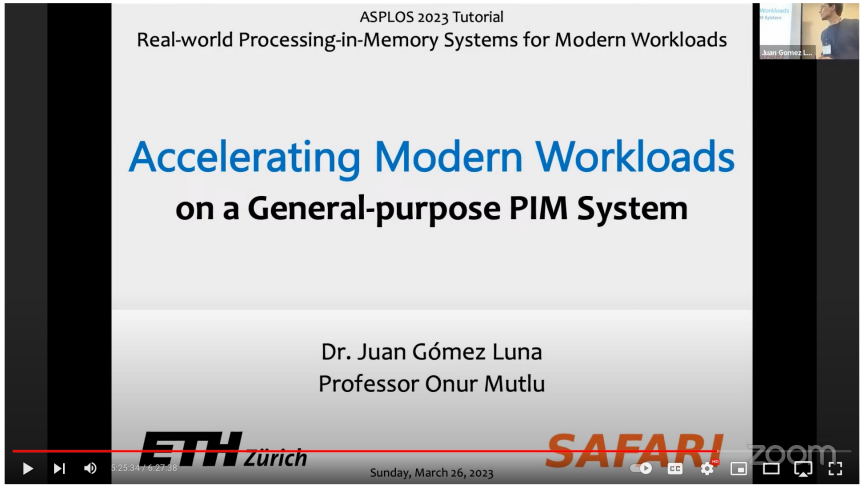
https://www.youtube.com/live/GIb5EgSrWk0

https://events.safari.ethz.ch/isca-pim-tutorial/

Real PIM Tutorial [ASPLOS 2023]

- March 26: Lectures + Hands-on labs + Invited talks





ASPLOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads

Onur Mutlu Lectures
32.1K subscribers
Subscribed
33
Share
Clip
Save
views Streamed 7 days ago Livestream - Data-Centric Architectures: Fundamentally Improving Performance and Energy (Spring 2023)
LOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads
://events.safari.ethz.ch/asplos-...

Tutorial Materials

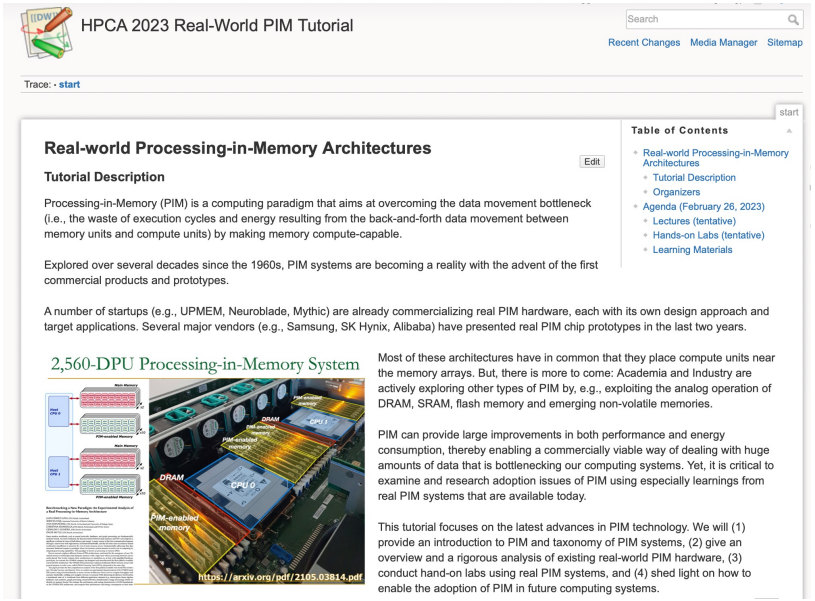
Time	Speaker	Title	Materials
9:00am-10:20am	Prof. Onur Mutlu	Memory-Centric Computing	(PDF) (PPT)
10:40am-12:00pm	Dr. Juan Gómez Luna	Processing-Near-Memory: Real PNM Architectures Programming General-purpose PIM	(PDF) (PPT)
1:40pm-2:20pm	Prof. Alexandra (Sasha) Fedorova (UBC)	Processing in Memory in the Wild	(PDF) (PPT)
2:20pm-3:20pm	Dr. Juan Gómez Luna & Ataberk Olgun	Processing-Using-Memory: Exploiting the Analog Operational Properties of Memory Components	(PDF) (PPT) (PDF) (PPT)
3:40pm-4:10pm	Dr. Juan Gómez Luna	Adoption issues: How to enable PIM? Accelerating Modern Workloads on a General-purpose PIM System	(PDF) (PPT) (PDF) (PPT)
4:10pm-4:50pm	Dr. Yongkee Kwon & Eddy (Chanwook) Park (SK Hynix)	System Architecture and Software Stack for GDDR6-AiM	(PDF) (PPT)
4:50pm-5:00pm	Dr. Juan Gómez Luna	Hands-on Lab: Programming and Understanding a Real Processing-in-Memory Architecture	(Handout) (PDF) (PPT)

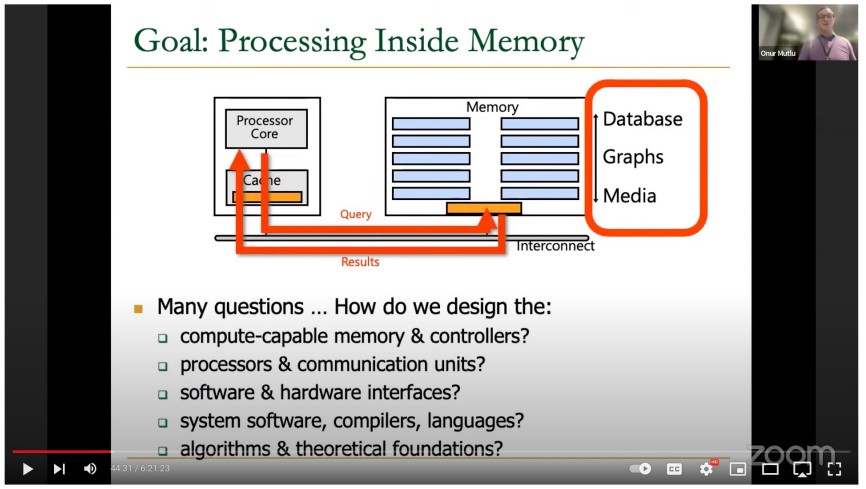
https://www.youtube.com/watch?v=oYCaLcT0Kmo

https://events.safari.ethz.ch/asplos-pim-tutorial/

Real PIM Tutorial [HPCA 2023]

- February 26: Lectures + Hands-on labs + Invited Talks





HPCA 2023 Tutorial: Real-World Processing-in-Memory Architectures

Onur Mutlu Lectures
32.1K subscribers
Subscribed
50
Share
Clip
Save

1.8K views Streamed 1 month ago Livestream - P&S Data-Centric Architectures: Fundamentally Improving Performance and Energy (Fall 2022)
HPCA 2023 Tutorial: Real-World Processing-in-Memory Architectures
https://events.safari.ethz.ch/real-pi...

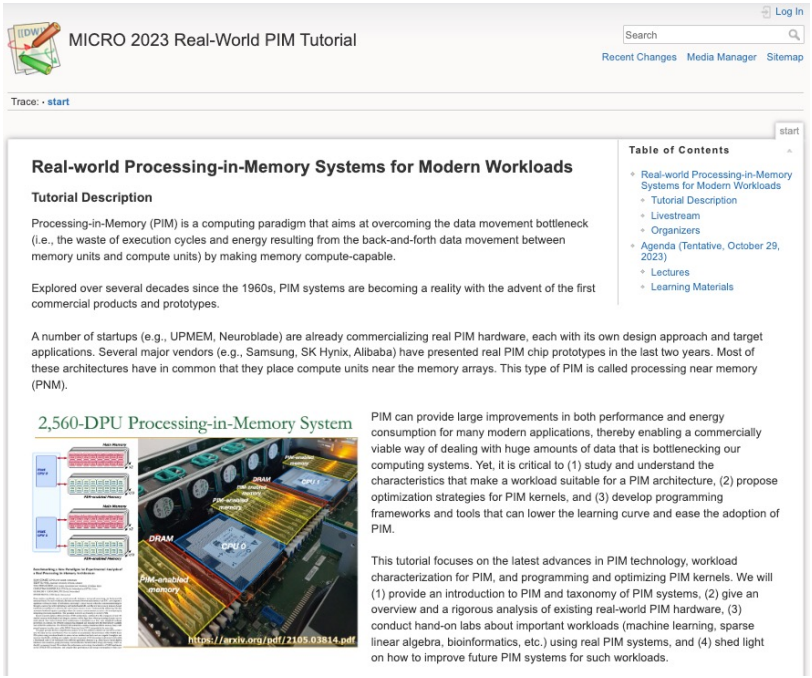
Time	Speaker	Title	Materials
8:00am-8:40am	Prof. Onur Mutlu	Memory-Centric Computing	(PDF) (PPT)
8:40am-10:00am	Dr. Juan Gómez Luna	Processing-Near-Memory: Real PNM Architectures Programming General-purpose PIM	(PDF) (PPT)
10:20am-11:00am	Dr. Dimin Niu	A 3D Logic-to-DRAM Hybrid Bonding Process-Near-Memory Chip for Recommendation System	
11:00am-11:40am	Dr. Christina Giannoula	SparseP: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures	(PDF) (PPT)
1:30pm-2:10pm	Dr. Juan Gómez Luna	Processing-Using-Memory: Exploiting the Analog Operational Properties of Memory Components	(PDF) (PPT)
2:10pm-2:50pm	Dr. Manuel Le Gallo	Deep Learning Inference Using Computational Phase-Change Memory	
2:50pm-3:30pm	Dr. Juan Gómez Luna	PIM Adoption Issues: How to Enable PIM Adoption?	(PDF) (PPT)
3:40pm-5:40pm	Dr. Juan Gómez Luna	Hands-on Lab: Programming and Understanding a Real Processing-in-Memory Architecture	(Handout) (PDF) (PPT)

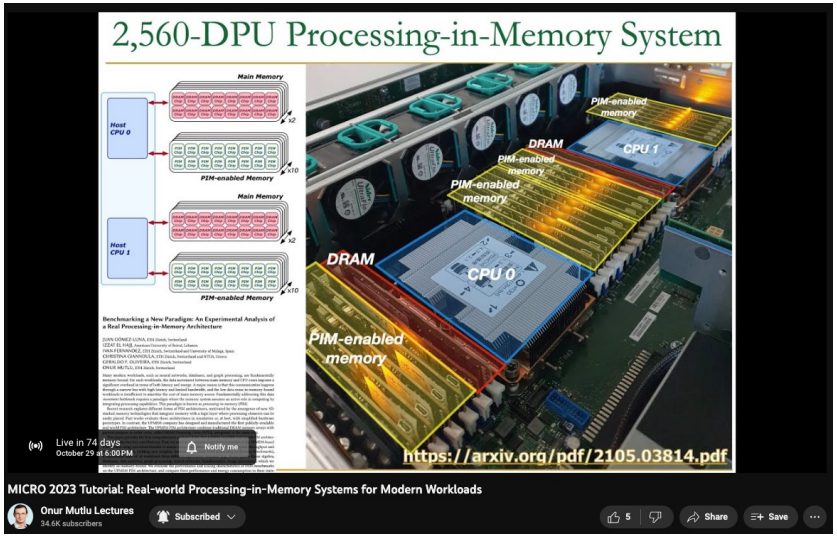
https://www.youtube.com/watch?v=f5-nT1tbz5w

https://events.safari.ethz.ch/real-pim-tutorial/

Real PIM Tutorial [MICRO 2023]

- **October 29**: Lectures + Hands-on labs + Invited talks





https://www.youtube.com/watch?v=ohUooNSIxOI

Agenda (Tentative, October 29, 2023)

Lectures

1. Introduction: PIM as a paradigm to overcome the data movement bottleneck.
2. PIM taxonomy: PNM (processing near memory) and PUM (processing using memory).
3. General-purpose PNM: UPMEM PIM.
4. PNM for neural networks: Samsung HBM-PIM, SK Hynix AiM.
5. PNM for recommender systems: Samsung AxDIMM, Alibaba PNM.
6. PUM prototypes: PiDRAM, SRAM-based PUM, Flash-based PUM.
7. Other approaches: Neuroblade, Mythic.
8. Adoption issues: How to enable PIM?
9. Hands-on labs: Programming a real PIM system.

https://events.safari.ethz.ch/micro-pim-tutorial

FPGA-based Processing Near Memory

- Gagandeep Singh, Mohammed Alser, Damla Senol Cali, Dionysios Diamantopoulos, Juan Gómez-Luna, Henk Corporaal, and Onur Mutlu, **"FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications"**
  IEEE Micro (***IEEE MICRO***), 2021.

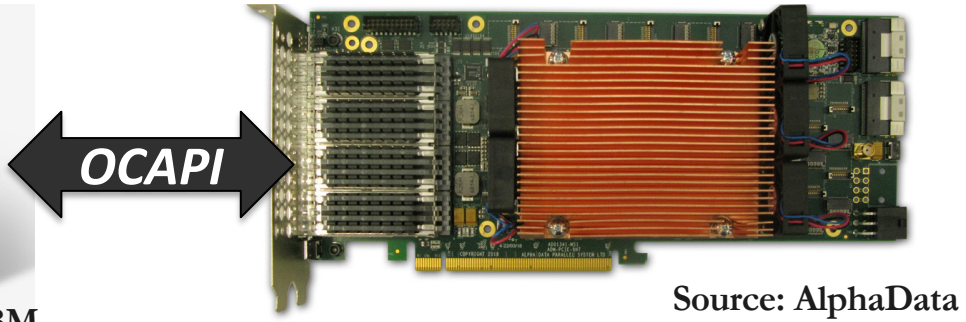
FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications

Gagandeep Singh[◇] **Mohammed Alser**[◇] **Damla Senol Cali**[⋈]
Dionysios Diamantopoulos[▽] **Juan Gómez-Luna**[◇]
Henk Corporaal[⋆] **Onur Mutlu**[◇⋈]

[◇]*ETH Zürich* [⋈]*Carnegie Mellon University*
[⋆]*Eindhoven University of Technology* [▽]*IBM Research Europe*

Near-Memory Acceleration using FPGAs

IBM POWER9 CPU

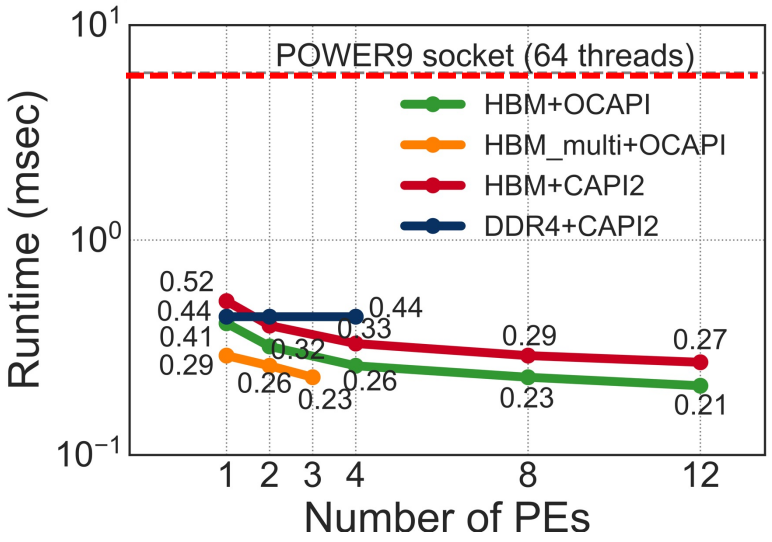


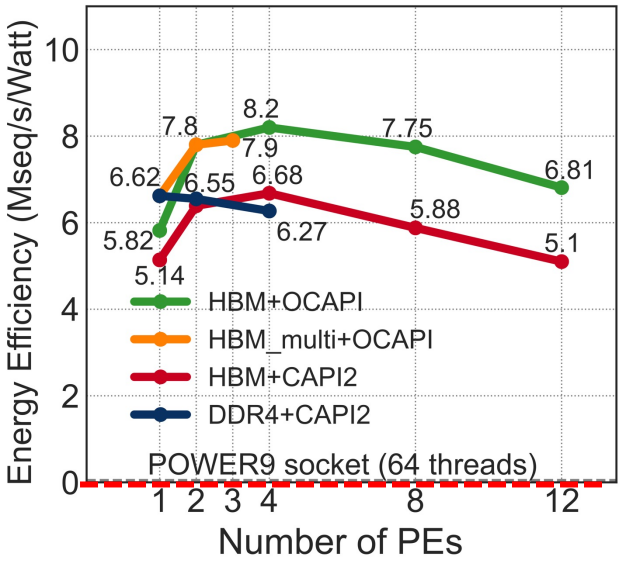
HBM-based FPGA board

Near-HBM FPGA-based accelerator

Two communication technologies: **CAPI2** and **OCAPI**
Two memory technologies: **DDR4** and **HBM**
Two workloads: **Weather Modeling** and **Genome Analysis**

Performance & Energy Greatly Improve





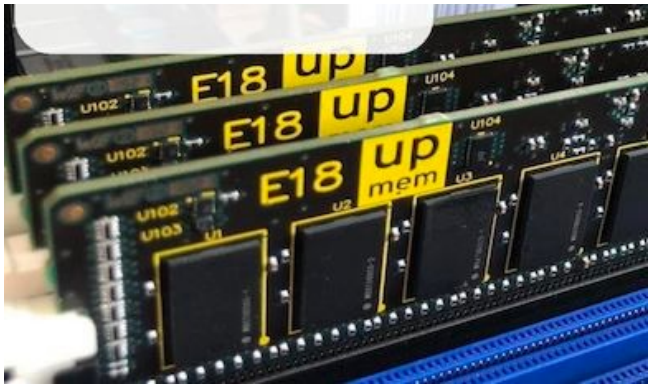
5-27× performance vs. a 16-core (64-thread) IBM POWER9 CPU

12-133× energy efficiency vs. a 16-core (64-thread) IBM POWER9 CPU

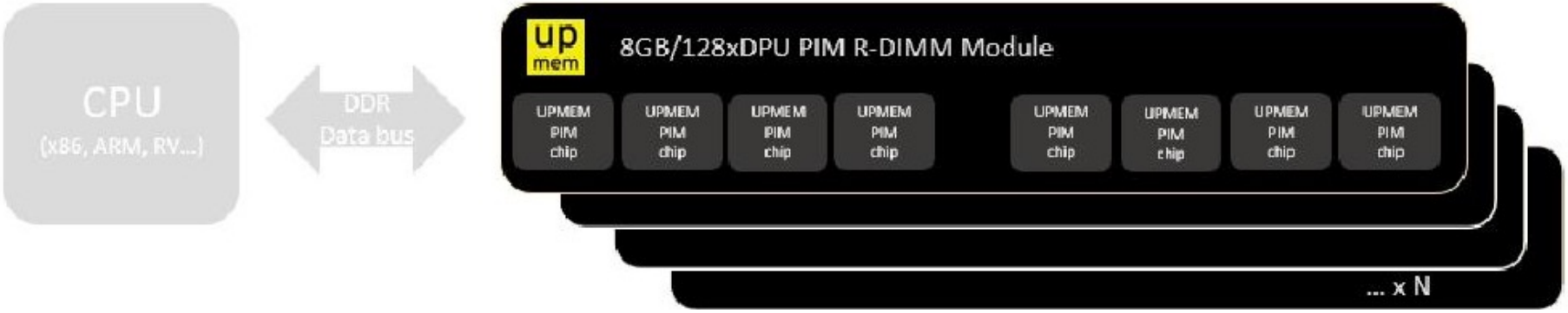
HBM alleviates memory bandwidth contention vs. DDR4

UPMEM Processing-in-DRAM Engine (2019)

- Processing in DRAM Engine
- Includes **standard DIMM modules**, with a **large number of DPU processors** combined with DRAM chips.



- Replaces **standard** DIMMs
  - DDR4 R-DIMM modules
    - 8GB+128 DPUs (16 PIM chips)
    - Standard 2x-nm DRAM process
  - **Large amounts of** compute & memory bandwidth



https://www.anandtech.com/show/14750/hot-chips-31-analysis-inmemory-processing-by-upmem
https://www.upmem.com/video-upmem-presenting-its-true-processing-in-memory-solution-hot-chips-2019/

Experimental Analysis of the UPMEM PIM Engine

Benchmarking a New Paradigm: An Experimental Analysis of a Real Processing-in-Memory Architecture

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
IZZAT EL HAJJ, American University of Beirut, Lebanon
IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain
CHRISTINA GIANNOULA, ETH Zürich, Switzerland and NTUA, Greece
GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

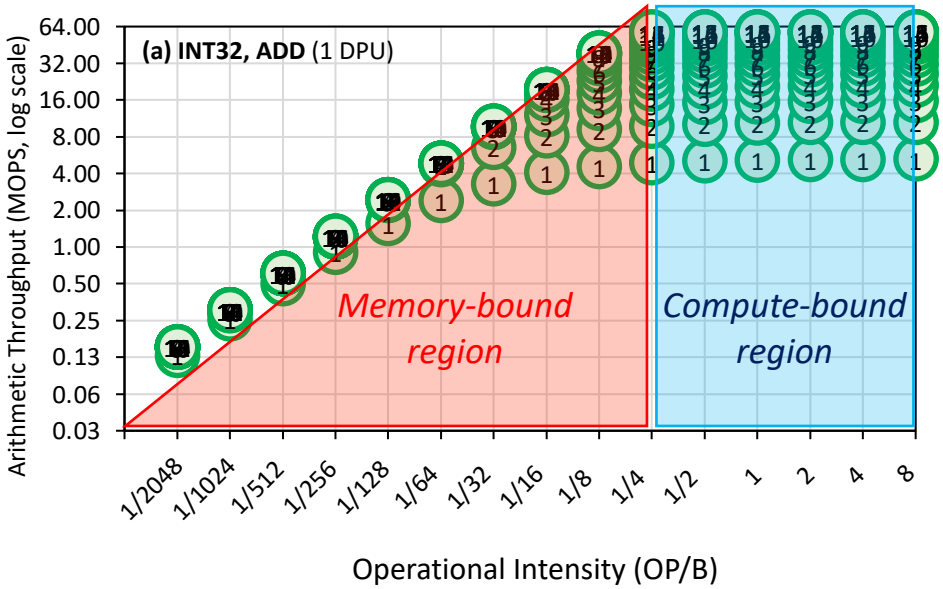
Many modern workloads, such as neural networks, databases, and graph processing, are fundamentally memory-bound. For such workloads, the data movement between main memory and CPU cores imposes a significant overhead in terms of both latency and energy. A major reason is that this communication happens through a narrow bus with high latency and limited bandwidth, and the low data reuse in memory-bound workloads is insufficient to amortize the cost of main memory access. Fundamentally addressing this *data movement bottleneck* requires a paradigm where the memory system assumes an active role in computing by integrating processing capabilities. This paradigm is known as *processing-in-memory* (*PIM*).

Recent research explores different forms of PIM architectures, motivated by the emergence of new 3D-stacked memory technologies that integrate memory with a logic layer where processing elements can be easily placed. Past works evaluate these architectures in simulation or, at best, with simplified hardware prototypes. In contrast, the UPMEM company has designed and manufactured the first publicly-available real-world PIM architecture. The UPMEM PIM architecture combines traditional DRAM memory arrays with general-purpose in-order cores, called *DRAM Processing Units* (*DPUs*), integrated in the same chip.

This paper provides the first comprehensive analysis of the first publicly-available real-world PIM architecture. We make two key contributions. First, we conduct an experimental characterization of the UPMEM-based PIM system using microbenchmarks to assess various architecture limits such as compute throughput and memory bandwidth, yielding new insights. Second, we present *PrIM* (*Processing-In-Memory benchmarks*), a benchmark suite of 16 workloads from different application domains (e.g., dense/sparse linear algebra, databases, data analytics, graph processing, neural networks, bioinformatics, image processing), which we identify as memory-bound. We evaluate the performance and scaling characteristics of PrIM benchmarks on the UPMEM PIM architecture, and compare their performance and energy consumption to their state-of-the-art CPU and GPU counterparts. Our extensive evaluation conducted on two real UPMEM-based PIM systems with 640 and 2,556 DPUs provides new insights about suitability of different workloads to the PIM system, programming recommendations for software designers, and suggestions and hints for hardware and architecture designers of future PIM systems.

https://arxiv.org/pdf/2105.03814.pdf

Key Takeaway 1



The throughput saturation point is as low as ¼ OP/B, i.e., 1 integer addition per every 32-bit element fetched

KEY TAKEAWAY 1

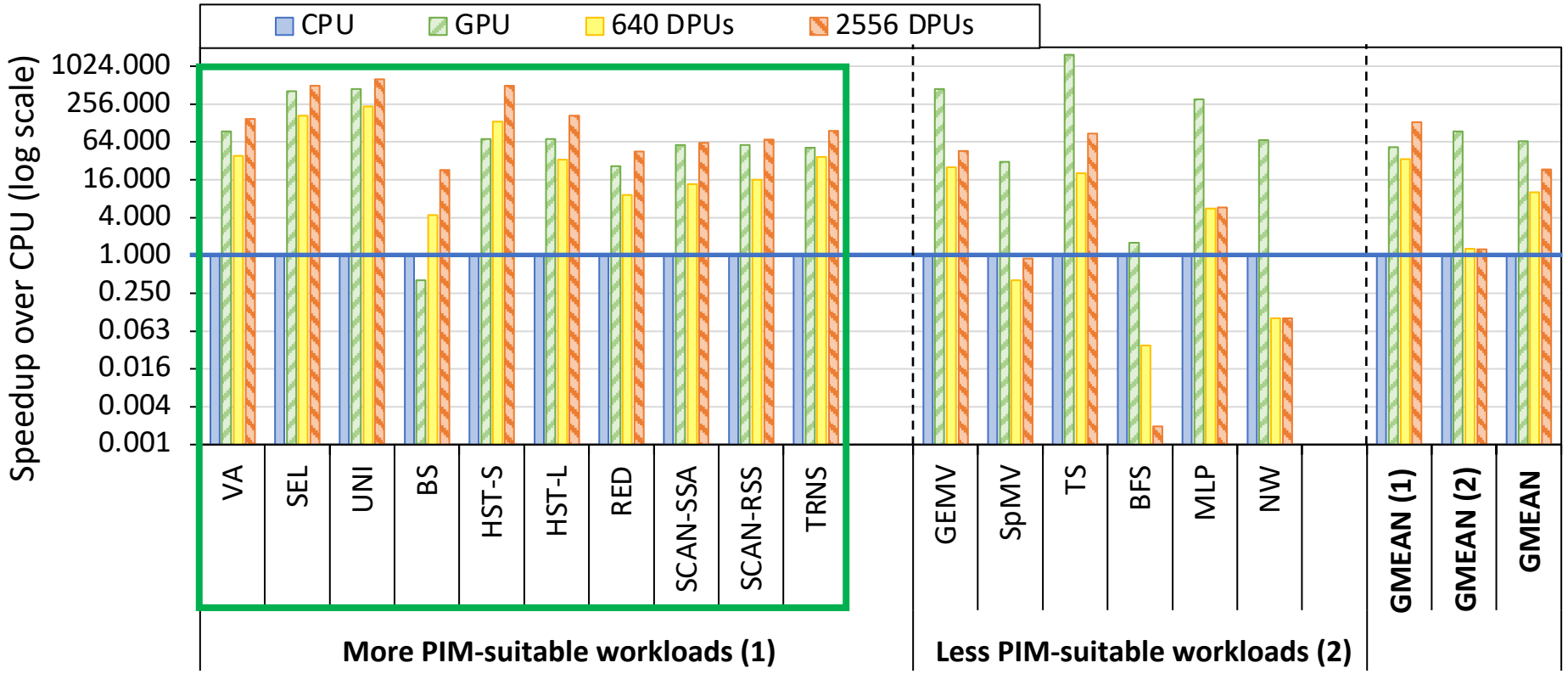
The UPMEM PIM architecture is fundamentally compute bound.
As a result, **the most suitable workloads are memory-bound.**

Key Takeaway 2

Table 4: Evaluated CPU, GPU, and UPMEM-based PIM Systems.

System	Process Node	Processor Cores			Memory		TDP
		Total Cores	Frequency	Peak Performance	Capacity	Total Bandwidth	
Intel Xeon E3-1225 v6 CPU [241]	14 nm	4 (8 threads)	3.3 GHz	26.4 GFLOPS*	32 GB	37.5 GB/s	73 W
NVIDIA Titan V GPU [277]	14 nm	80 (5,120 SIMD lanes)	1.2 GHz	12,288.0 GFLOPS	12 GB	652.8 GB/s	250 W
2,556-DPU PIM System	2x nm	2,556[9]	350 MHz	894.6 GOPS	159.75 GB	1.7 TB/s	383 W†
640-DPU PIM System	2x nm	640	267 MHz	170.9 GOPS	40 GB	333.75 GB/s	96 W†

*Estimated GFLOPS = 3.3 GHz × 4 cores × 2 instructions per cycle.
†Estimated TDP = $\frac{\text{Total DPUs}}{\text{DPUs/chip}} \times 1.2$ W/chip [199].



KEY TAKEAWAY 2

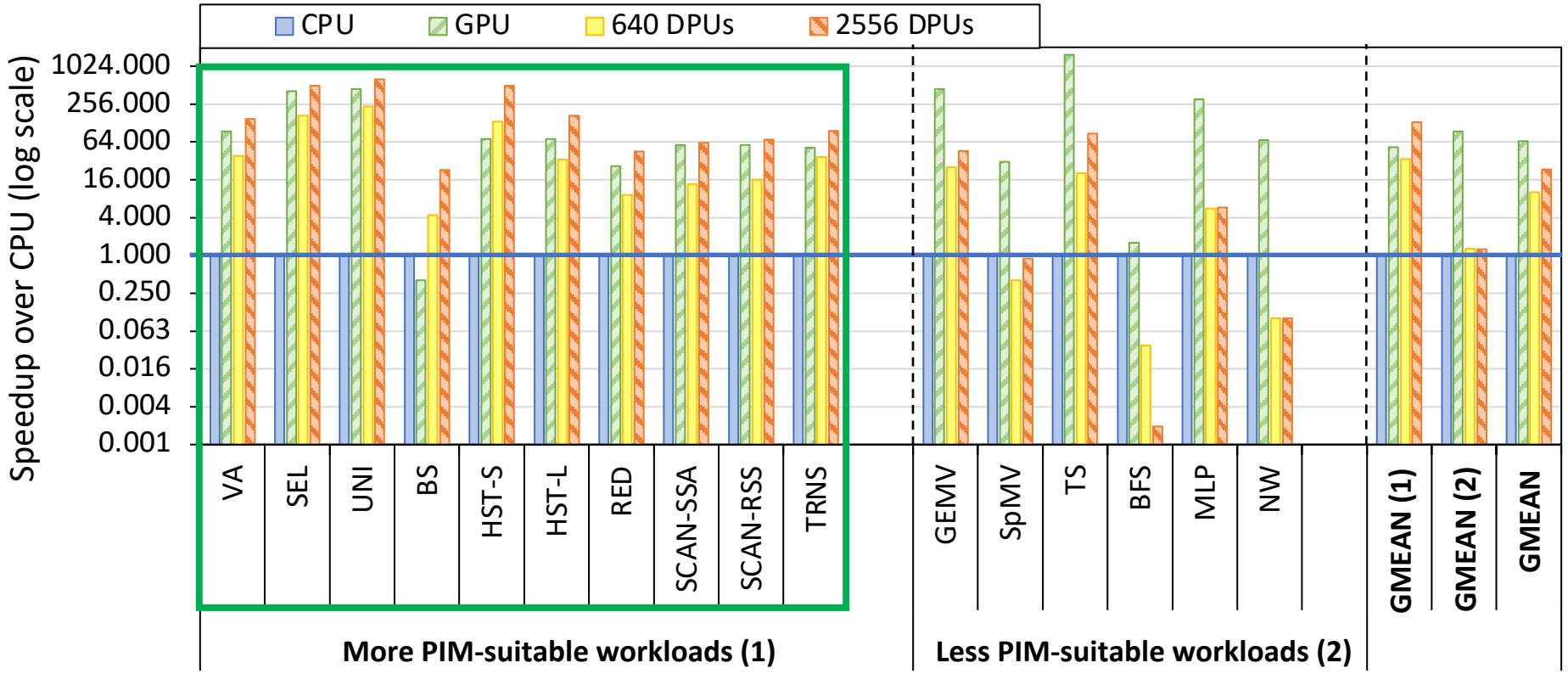
The most well-suited workloads for the UPMEM PIM architecture use no arithmetic operations or use only simple operations (e.g., bitwise operations and integer addition/subtraction).

Key Takeaway 3

Table 4: Evaluated CPU, GPU, and UPMEM-based PIM Systems.

System	Process Node	Processor Cores			Memory		TDP
		Total Cores	Frequency	Peak Performance	Capacity	Total Bandwidth	
Intel Xeon E3-1225 v6 CPU [241]	14 nm	4 (8 threads)	3.3 GHz	26.4 GFLOPS*	32 GB	37.5 GB/s	73 W
NVIDIA Titan V GPU [277]	14 nm	80 (5,120 SIMD lanes)	1.2 GHz	12,288.0 GFLOPS	12 GB	652.8 GB/s	250 W
2,556-DPU PIM System	2x nm	2,556[9]	350 MHz	894.6 GOPS	159.75 GB	1.7 TB/s	383 W†
640-DPU PIM System	2x nm	640	267 MHz	170.9 GOPS	40 GB	333.75 GB/s	96 W†

*Estimated GFLOPS = 3.3 GHz × 4 cores × 2 instructions per cycle.
†Estimated TDP = $\frac{Total\ DPUs}{DPUs/chip}$ × 1.2 W/chip [199].



KEY TAKEAWAY 3

The most well-suited workloads for the UPMEM PIM architecture require little or no communication across DPUs (inter-DPU communication).

UPMEM PIM System Summary & Analysis

- Juan Gomez-Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F. Oliveira, and Onur Mutlu,
  "Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-in-Memory Hardware"
  *Invited Paper at Workshop on Computing with Unconventional Technologies (**CUT**)*, Virtual, October 2021.
  [arXiv version]
  [PrIM Benchmarks Source Code]
  [Slides (pptx) (pdf)]
  [Talk Video (37 minutes)]
  [Lightning Talk Video (3 minutes)]

Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-in-Memory Hardware

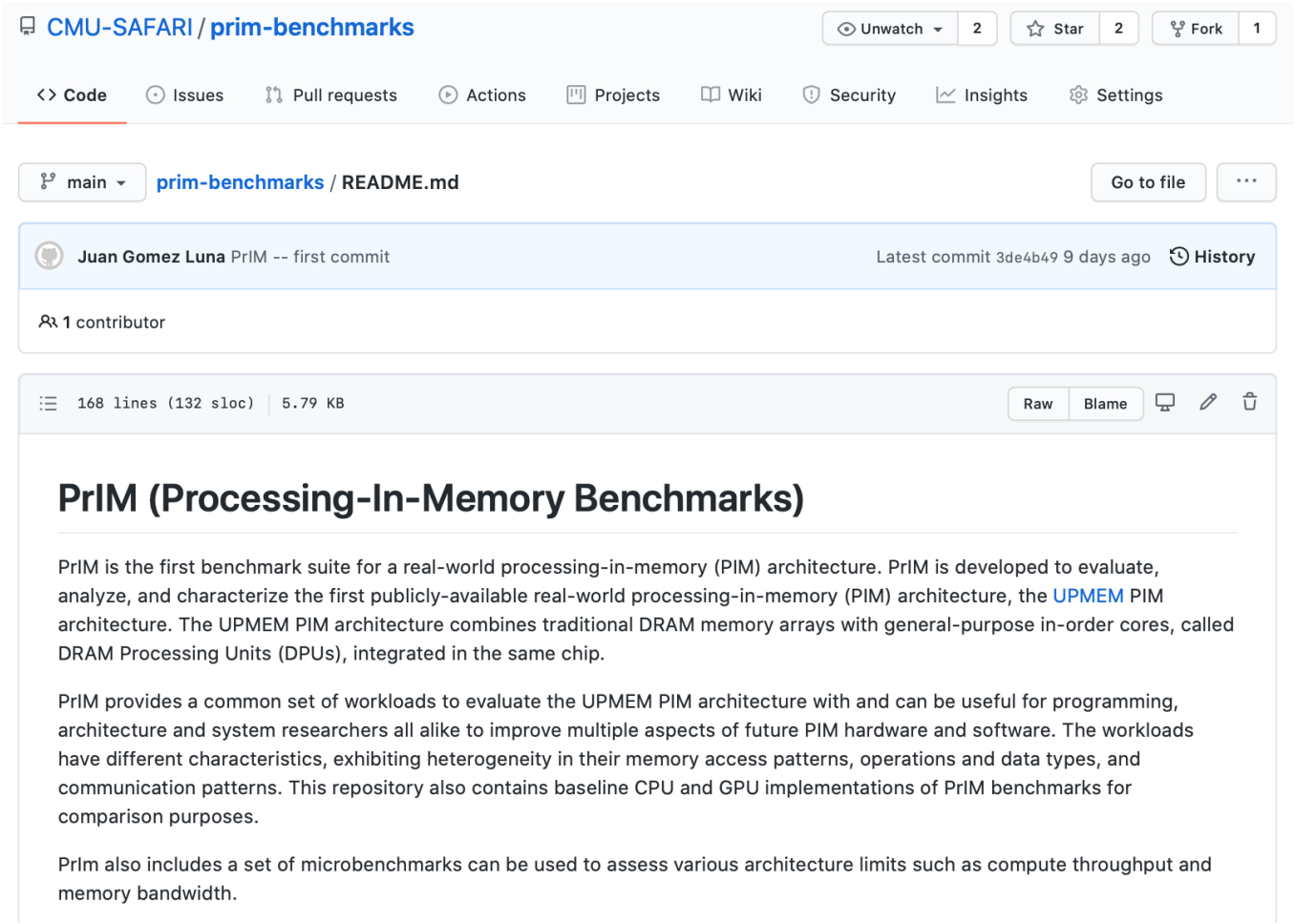
Juan Gómez-Luna, *ETH Zürich*
Izzat El Hajj, *American University of Beirut*
Ivan Fernandez, *University of Malaga*
Christina Giannoula, *National Technical University of Athens*
Geraldo F. Oliveira, *ETH Zürich*
Onur Mutlu, *ETH Zürich*

PrIM Benchmarks: Application Domains

Domain	Benchmark	Short name
Dense linear algebra	Vector Addition	VA
	Matrix-Vector Multiply	GEMV
Sparse linear algebra	Sparse Matrix-Vector Multiply	SpMV
Databases	Select	SEL
	Unique	UNI
Data analytics	Binary Search	BS
	Time Series Analysis	TS
Graph processing	Breadth-First Search	BFS
Neural networks	Multilayer Perceptron	MLP
Bioinformatics	Needleman-Wunsch	NW
Image processing	Image histogram (short)	HST-S
	Image histogram (large)	HST-L
Parallel primitives	Reduction	RED
	Prefix sum (scan-scan-add)	SCAN-SSA
	Prefix sum (reduce-scan-scan)	SCAN-RSS
	Matrix transposition	TRNS

PrIM Benchmarks are Open Source

- All microbenchmarks, benchmarks, and scripts
- https://github.com/CMU-SAFARI/prim-benchmarks



Understanding a Modern PIM Architecture

Benchmarking a New Paradigm: Experimental Analysis and Characterization of a Real Processing-in-Memory System

JUAN GÓMEZ-LUNA[1], IZZAT EL HAJJ[2], IVAN FERNANDEZ[1,3], CHRISTINA GIANNOULA[1,4], GERALDO F. OLIVEIRA[1], AND ONUR MUTLU[1]

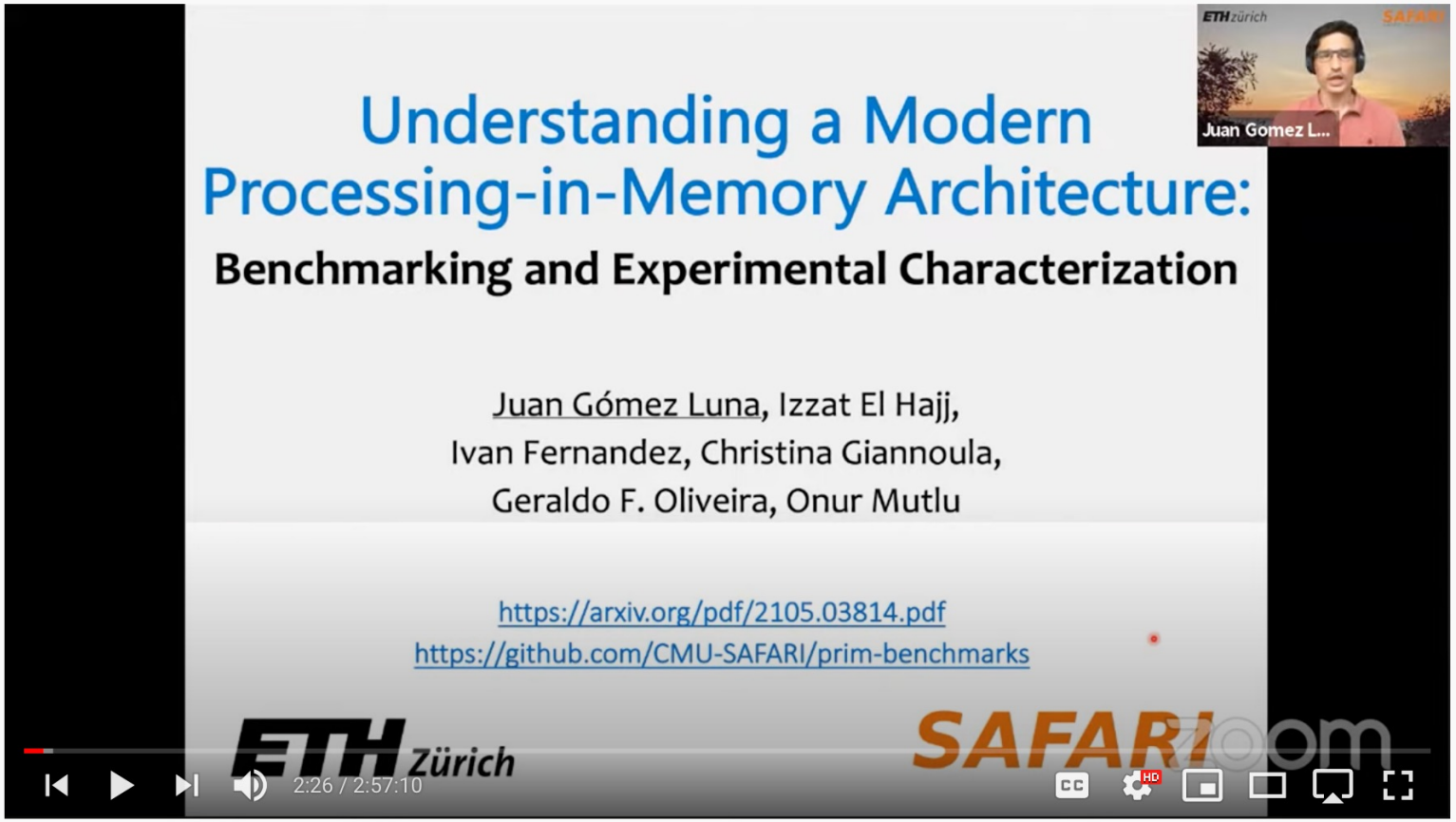
[1]ETH Zürich
[2]American University of Beirut
[3]University of Malaga
[4]National Technical University of Athens

Corresponding author: Juan Gómez-Luna (e-mail: juang@ethz.ch).

https://arxiv.org/pdf/2105.03814.pdf

https://github.com/CMU-SAFARI/prim-benchmarks

Understanding a Modern PIM Architecture



SAFARI Live Seminar: Understanding a Modern Processing-in-Memory Architecture

2,579 views • Streamed live on Jul 12, 2021 93 0 SHARE SAVE ...

Onur Mutlu Lectures
18.7K subscribers
SUBSCRIBED

https://www.youtube.com/watch?v=D8Hjy2iU9l4&list=PL5Q2soXY2Zi_tOTAYm--dYByNPL7JhwR9

ML Training on a Real PIM System

Machine Learning Training on a Real Processing-in-Memory System

Juan Gómez-Luna[1] Yuxin Guo[1] Sylvan Brocard[2] Julien Legriel[2]
Remy Cimadomo[2] Geraldo F. Oliveira[1] Gagandeep Singh[1] Onur Mutlu[1]
[1]ETH Zürich [2]UPMEM

An Experimental Evaluation of Machine Learning Training on a Real Processing-in-Memory System

Juan Gómez-Luna[1] Yuxin Guo[1] Sylvan Brocard[2] Julien Legriel[2]
Remy Cimadomo[2] Geraldo F. Oliveira[1] Gagandeep Singh[1] Onur Mutlu[1]
[1]ETH Zürich [2]UPMEM

Short version: https://arxiv.org/pdf/2206.06022.pdf

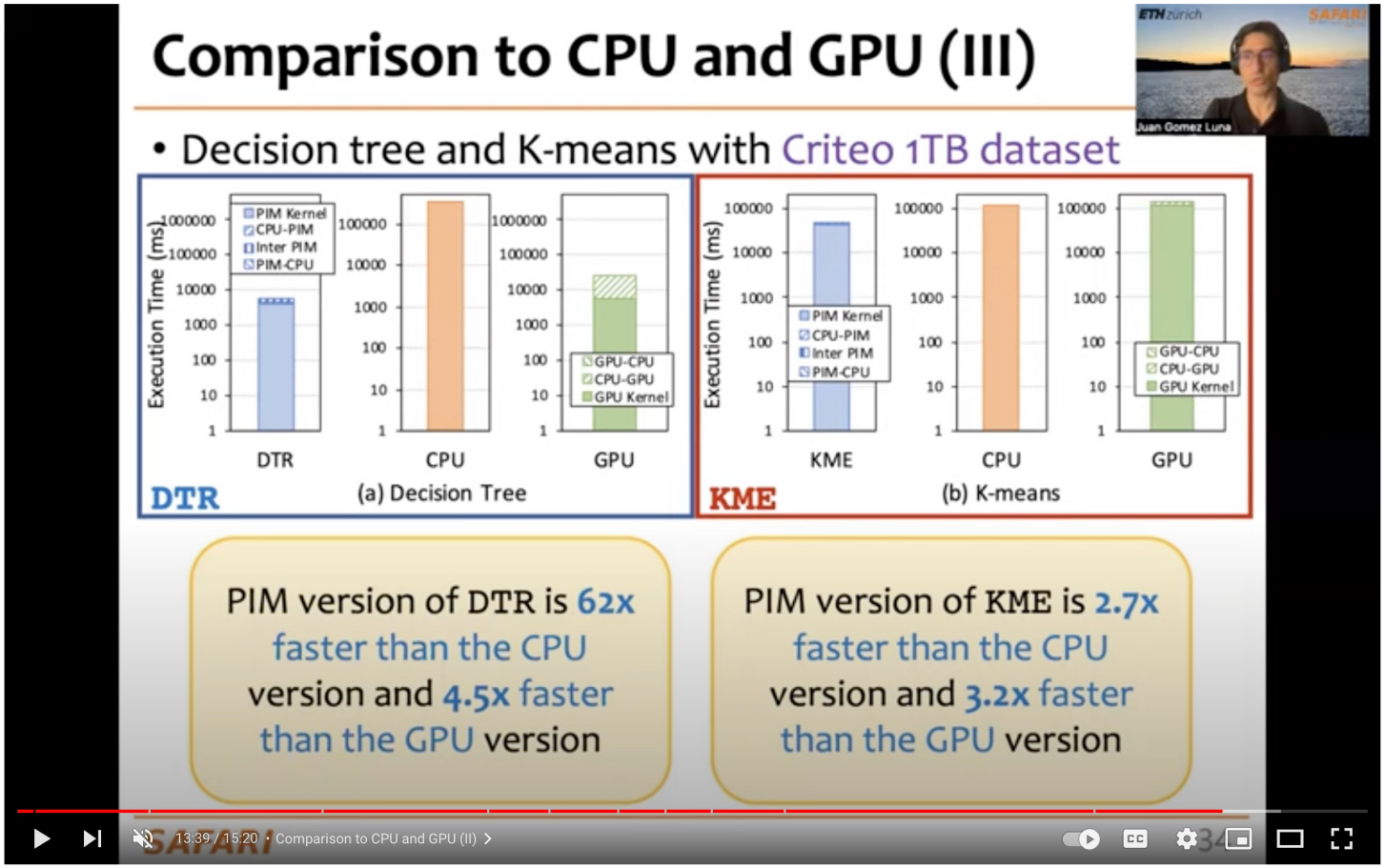
Long version: https://arxiv.org/pdf/2207.07886.pdf

https://www.youtube.com/watch?v=qeukNs5XI3g&t=11226s

ML Training on a Real PIM System

- **Need to optimize data representation**
  - (1) fixed-point
  - (2) quantization
  - (3) hybrid precision
- **Use lookup tables (LUTs) to implement complex functions (e.g., sigmoid)**
- **Optimize data placement & layout for streaming**
- **Large speedups:** 2.8X/27X vs. CPU, 1.3x/3.2x vs. GPU

ML Training on Real PIM Talk Video



Machine Learning Training on Memory-centric Computing Systems, Juan Gómez-Luna for ISPASS 2023

Onur Mutlu Lectures
32.9K subscribers
Analytics | Edit video | 9 | Share | Download | Clip | Save

242 views 11 days ago Livestream - Data-Centric Architectures: Fundamentally Improving Performance and Energy (Spring 2023)
Evaluating Machine Learning Workloads on Memory-centric Computing Systems

https://www.youtube.com/watch?v=60pkaI5AeM4

ML Training on Real PIM Systems

- Juan Gómez Luna, Yuxin Guo, Sylvan Brocard, Julien Legriel, Remy Cimadomo, Geraldo F. Oliveira, Gagandeep Singh, and Onur Mutlu,
  "Evaluating Machine Learning Workloads on Memory-Centric Computing Systems"
  *Proceedings of the 2023 IEEE International Symposium on Performance Analysis of Systems and Software (**ISPASS**)*, Raleigh, North Carolina, USA, April 2023.
  [arXiv version, 16 July 2022.]
  [PIM-ML Source Code]
  Best paper session.

An Experimental Evaluation of Machine Learning Training on a Real Processing-in-Memory System

Juan Gómez-Luna[1] Yuxin Guo[1] Sylvan Brocard[2] Julien Legriel[2]
Remy Cimadomo[2] Geraldo F. Oliveira[1] Gagandeep Singh[1] Onur Mutlu[1]
[1]ETH Zürich [2]UPMEM

https://github.com/CMU-SAFARI/pim-ml

https://arxiv.org/pdf/2207.07886.pdf

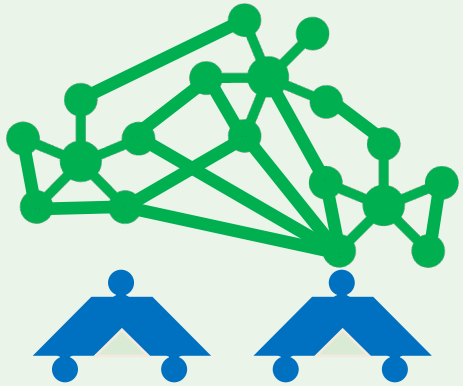
SpMV Multiplication on Real PIM Systems

- Appears at SIGMETRICS 2022

SparseP: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Systems

CHRISTINA GIANNOULA, ETH Zürich, Switzerland and National Technical University of Athens, Greece
IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain
JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
NECTARIOS KOZIRIS, National Technical University of Athens, Greece
GEORGIOS GOUMAS, National Technical University of Athens, Greece
ONUR MUTLU, ETH Zürich, Switzerland

https://arxiv.org/pdf/2201.05072.pdf
https://github.com/CMU-SAFARI/SparseP



SparseP

Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures

Christina Giannoula

Ivan Fernandez, Juan Gomez-Luna,

Nectarios Koziris, Georgios Goumas, Onur Mutlu

SAFARI ETH zürich National Technical University of Athens CSLab UNIVERSIDAD DE MÁLAGA

SparseP: Key Contributions

1. Efficient SpMV kernels for current & future PIM systems
   - SparseP library = 25 SpMV kernels
     - Compression, data types, data partitioning, synchronization, load balancing

SparseP is Open-Source

SparseP: https://github.com/CMU-SAFARI/SparseP

2. Comprehensive analysis of SpMV on the first commercially-available real PIM system

   - 26 sparse matrices
   - Comparisons to state-of-the-art CPU and GPU systems
   - Recommendations for software, system and hardware designers

Recommendations for Architects and Programmers

Full Paper: https://arxiv.org/pdf/2201.05072.pdf

SparseP Talk Video

Processing-in-Memory Course: Lecture 11: SpMV on a Real PIM Architecture - Spring 2022

149 views • Streamed live on May 19, 2022

12 DISLIKE SHARE DOWNLOAD CLIP SAVE ...

Onur Mutlu Lectures
25K subscribers

ANALYTICS EDIT VIDEO

https://www.youtube.com/watch?v=5kaOsJKlGrE

More on SparseP

Christina Giannoula, Ivan Fernandez, Juan Gomez-Luna, Nectarios Koziris, Georgios Goumas, and Onur Mutlu,
"SparseP: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures"
*Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (**SIGMETRICS**)*, Mumbai, India, June 2022.
[Extended arXiv Version]
[Abstract]
[Slides (pptx) (pdf)]
[Long Talk Slides (pptx) (pdf)]
[SparseP Source Code]
[Talk Video (16 minutes)]
[Long Talk Video (55 minutes)]

***SparseP*: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Systems**

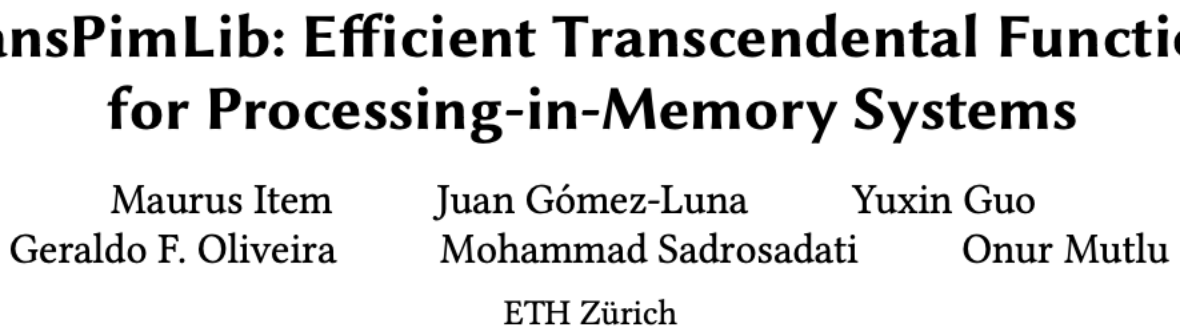
CHRISTINA GIANNOULA, ETH Zürich, Switzerland and National Technical University of Athens, Greece
IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain
JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
NECTARIOS KOZIRIS, National Technical University of Athens, Greece
GEORGIOS GOUMAS, National Technical University of Athens, Greece
ONUR MUTLU, ETH Zürich, Switzerland

https://github.com/CMU-SAFARI/SparseP

https://arxiv.org/pdf/2201.05072.pdf

Transcendental Functions on Real PIM Systems

- Maurus Item, Juan Gómez Luna, Yuxin Guo, Geraldo F. Oliveira, Mohammad Sadrosadati, and Onur Mutlu,
  "TransPimLib: Efficient Transcendental Functions for Processing-in-Memory Systems"
  *Proceedings of the 2023 IEEE International Symposium on Performance Analysis of Systems and Software (**ISPASS**)*, Raleigh, North Carolina, USA, April 2023.
  [arXiv version]
  [Slides (pptx) (pdf)]
  [TransPimLib Source Code]
  [Talk Video (17 minutes)]



https://github.com/CMU-SAFARI/transpimlib

https://arxiv.org/pdf/2304.01951.pdf

Sequence Alignment on Real PIM Systems

- Safaa Diab, Amir Nassereldine, Mohammed Alser, Juan Gómez Luna, Onur Mutlu, and Izzat El Hajj,
  "A Framework for High-throughput Sequence Alignment using Real Processing-in-Memory Systems"
  Bioinformatics, [published online on] 27 March 2023.
  [Online link at Bioinformatics Journal]
  [arXiv preprint]
  [AiM Source Code]

A Framework for High-throughput Sequence Alignment using Real Processing-in-Memory Systems

Safaa Diab [1] Amir Nassereldine [1] Mohammed Alser [2] Juan Gómez Luna [2]
Onur Mutlu [2] Izzat El Hajj [1]

[1]American University of Beirut [2]ETH Zürich

https://github.com/CMU-SAFARI/alignment-in-memory

https://arxiv.org/pdf/2208.01243.pdf

Summary

- Sequence alignment on traditional systems is limited by the **memory bandwidth bottleneck**
- **Processing-in-memory (PIM)** overcomes this bottleneck by placing cores near the memory
- Our framework, **Alignment-in-Memory (AIM)**, is a PIM framework that supports multiple alignment algorithms (NW, SWG, GenASM, WFA)
  - Implemented on UPMEM, the first real PIM system
- Results show **substantial speedups over both CPUs (1.8X-28X) and GPUs (1.2X-2.7X)**
- AIM is available at:
  - https://github.com/CMU-SAFARI/alignment-in-memory

Homomorphic Operations on Real PIM Systems

- Harshita Gupta, Mayank Kabra, Juan Gómez-Luna, Konstantinos Kanellopoulos, and Onur Mutlu,
  "Evaluating Homomorphic Operations on a Real-World Processing-In-Memory System"
  *Proceedings of the 2023 IEEE International Symposium on Workload Characterization Poster Session (**IISWC**)*, Ghent, Belgium, October 2023.
  [arXiv version]
  [Lightning Talk Slides (pptx) (pdf)]
  [Poster (pptx) (pdf)]

Evaluating Homomorphic Operations on a Real-World Processing-In-Memory System

Harshita Gupta* Mayank Kabra* Juan Gómez-Luna Konstantinos Kanellopoulos Onur Mutlu
ETH Zürich

Accelerating Reinforcement Learning

- Kailash Gogineni, Sai Santosh Dayapule, Juan Gomez-Luna, Karthikeya Gogineni, Peng Wei, Tian Lan, Mohammad Sadrosadati, Onur Mutlu, Guru Venkataramani, **"SwiftRL: Towards Efficient Reinforcement Learning on Real Processing-In-Memory Systems"**
  *Proceedings of the 2024 IEEE International Symposium on Performance Analysis of Systems and Software (**ISPASS**)*, Indianapolis, Indiana, May 2024.
  [Slides (pptx) (pdf)]
  [arXiv version]

SwiftRL: Towards Efficient Reinforcement Learning on Real Processing-In-Memory Systems

Kailash Gogineni[1] Sai Santosh Dayapule[1] Juan Gómez-Luna[2] Karthikeya Gogineni[3]
Peng Wei[1] Tian Lan[1] Mohammad Sadrosadati[2] Onur Mutlu[2] Guru Venkataramani[1]

[1]George Washington University, USA [2]ETH Zürich, Switzerland [3]Independent

https://arxiv.org/pdf/2405.03967

Accelerating ML Training on Real PIM Systems

- **https://arxiv.org/pdf/2404.07164**

Analysis of Distributed Optimization Algorithms on a Real Processing-In-Memory System

Steve Rhyner[1] Haocong Luo[1] Juan Gómez-Luna[2] Mohammad Sadrosadati[1]
Jiawei Jiang[3] Ataberk Olgun[1] Harshita Gupta[1] Ce Zhang[4] Onur Mutlu[1]

[1]ETH Zurich [2]NVIDIA [3]Wuhan University [4]University of Chicago

Accelerating ML Training on Real PIM Systems

- **https://arxiv.org/pdf/2404.07164**

8. Conclusion

We evaluate and train ML models on large-scale datasets with centralized parallel optimization algorithms on a *real-world* PIM architecture. We show the importance of carefully *choosing* the distributed optimization algorithm that fits PIM and analyze tradeoffs. We demonstrate that *commercial* general-purpose PIM systems can be a viable alternative for many ML training workloads on large-scale datasets to processor-centric architectures. Our results demonstrate the necessity of adapting PIM architectures to enable inter-DPU communication to overcome scalability challenges for many ML training workloads and discuss decentralized parallel SGD optimization algorithms as a potential solution.

Accelerating GNNs on Real PIM Systems

- **https://arxiv.org/pdf/2402.16731**

Accelerating Graph Neural Networks on Real Processing-In-Memory Systems

Christina Giannoula*†, Peiming Yang*, Ivan Fernandez Vega§†, Jiacheng Yang*, Yu Xin Li*,
Juan Gomez Luna¶†, Mohammad Sadrosadati†, Onur Mutlu†‡, Gennady Pekhimenko*‖
*University of Toronto †ETH Zürich §Barcelona Supercomputing Center
¶NVIDIA ‡Stanford ‖CentML

Accelerating GNNs on Real PIM Systems

- **https://arxiv.org/pdf/2402.16731**

***Abstract*—Graph Neural Networks (GNNs) are emerging ML models to analyze graph-structure data. Graph Neural Network (GNN) execution involves both compute-intensive and memory-intensive kernels, the latter dominates the total time, being significantly bottlenecked by data movement between memory and processors. Processing-In-Memory (PIM) systems can alleviate this data movement bottleneck by placing simple processors near or inside to memory arrays. In this work, we introduce PyGim, an efficient ML framework that accelerates GNNs on real PIM systems. We propose intelligent parallelization techniques for memory-intensive kernels of GNNs tailored for real PIM systems, and develop handy Python API for them. We provide hybrid GNN execution, in which the compute-intensive and memory-intensive kernels are executed in processor-centric and memory-centric computing systems, respectively, to match their algorithmic nature. We extensively evaluate PyGim on a real-world PIM system with 1992 PIM cores using emerging GNN models, and demonstrate that it outperforms its state-of-the-art CPU counterpart on Intel Xeon by on average 3.04×, and achieves higher resource utilization than CPU and GPU systems. Our work provides useful recommendations for software, system and hardware designers. PyGim will be open-sourced to enable the widespread use of PIM systems in GNNs.**

Samsung Function-in-Memory DRAM (2021)

Samsung Newsroom

CORPORATE | PRODUCTS | PRESS RESOURCES | VIEWS | ABOUT US

Samsung Develops Industry's First High Bandwidth Memory with AI Processing Power

Korea on February 17, 2021 Audio Share

The new architecture will deliver over twice the system performance and reduce energy consumption by more than 70%

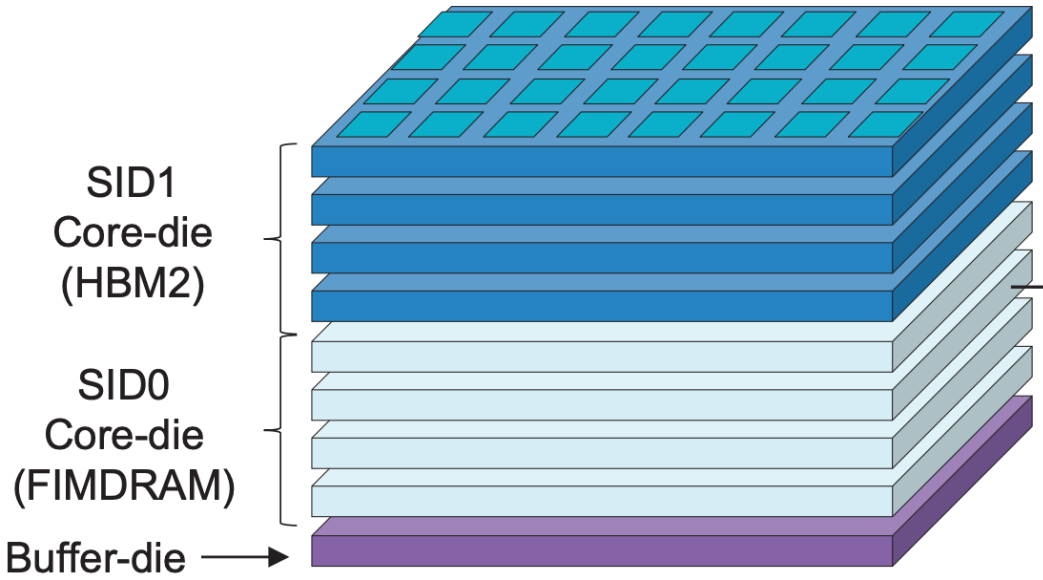
Samsung Electronics, the world leader in advanced memory technology, today announced that it has developed the industry's first High Bandwidth Memory (HBM) integrated with artificial intelligence (AI) processing power — the HBM-PIM. The new processing-in-memory (PIM) architecture brings powerful AI computing capabilities inside high-performance memory, to accelerate large-scale processing in data centers, high performance computing (HPC) systems and AI-enabled mobile applications.

Kwangil Park, senior vice president of Memory Product Planning at Samsung Electronics stated, "Our groundbreaking HBM-PIM is the industry's first programmable PIM solution tailored for diverse AI-driven workloads such as HPC, training and inference. We plan to build upon this breakthrough by further collaborating with AI solution providers for even more advanced PIM-powered applications."

https://news.samsung.com/global/samsung-develops-industrys-first-high-bandwidth-memory-with-ai-processing-power

Samsung Function-in-Memory DRAM (2021)

- FIMDRAM based on HBM2



[3D Chip Structure of HBM with FIMDRAM]

Chip Specification
128DQ / 8CH / 16 banks / BL4
32 PCU blocks (1 FIM block/2 banks)
1.2 TFLOPS (4H)
FP16 ADD / Multiply (MUL) / Multiply-Accumulate (MAC) / Multiply-and- Add (MAD)

ISSCC 2021 / SESSION 25 / DRAM / 25.4

25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications

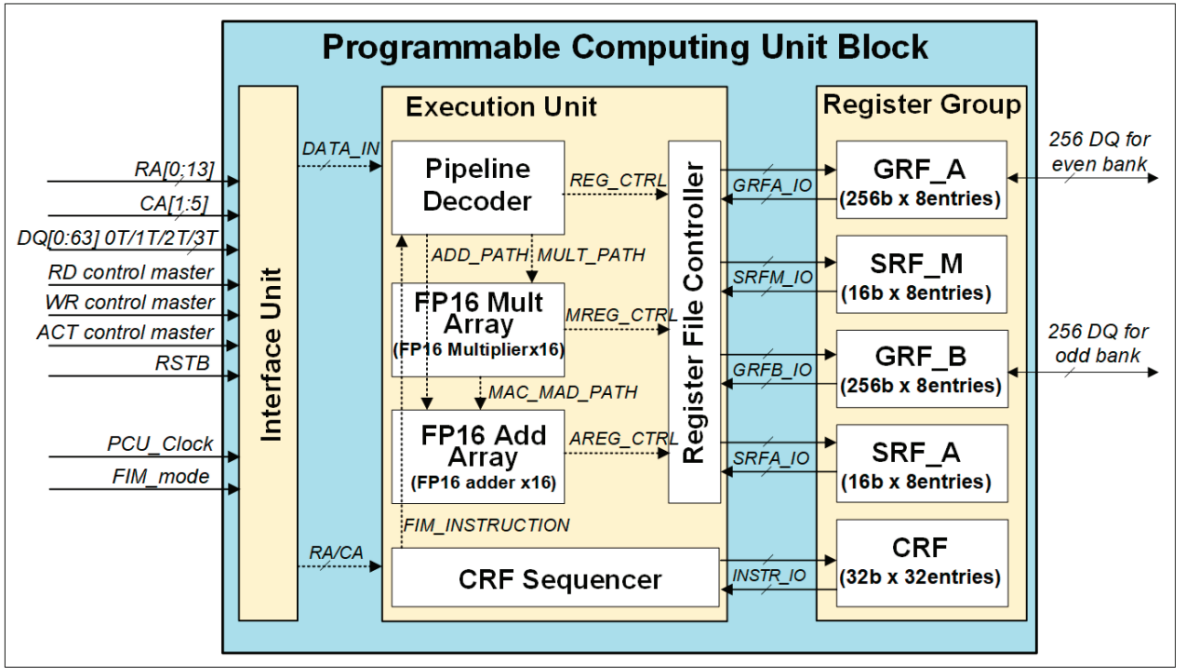
Young-Cheon Kwon[1], Suk Han Lee[1], Jaehoon Lee[1], Sang-Hyuk Kwon[1], Je Min Ryu[1], Jong-Pil Son[1], Seongil O[1], Hak-Soo Yu[1], Haesuk Lee[1], Soo Young Kim[1], Youngmin Cho[1], Jin Guk Kim[1], Jongyoon Choi[1], Hyun-Sung Shin[1], Jin Kim[1], BengSeng Phuah[1], HyoungMin Kim[1], Myeong Jun Song[1], Ahn Choi[1], Daeho Kim[1], SooYoung Kim[1], Eun-Bong Kim[1], David Wang[2], Shinhaeng Kang[1], Yuhwan Ro[3], Seungwoo Seo[3], JoonHo Song[3], Jaeyoun Youn[1], Kyomin Sohn[1], Nam Sung Kim[1]

[1]Samsung Electronics, Hwaseong, Korea
[2]Samsung Electronics, San Jose, CA
[3]Samsung Electronics, Suwon, Korea

Samsung Function-in-Memory DRAM (2021)

Programmable Computing Unit

- Configuration of PCU block
  - Interface unit to control data flow
  - Execution unit to perform operations
  - Register group
    - 32 entries of CRF for instruction memory
    - 16 GRF for weight and accumulation
    - 16 SRF to store constants for MAC operations



[Block diagram of PCU in FIMDRAM]

ISSCC 2021 / SESSION 25 / DRAM / 25.4

25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications

Young-Cheon Kwon[1], Suk Han Lee[1], Jaehoon Lee[1], Sang-Hyuk Kwon[1], Je Min Ryu[1], Jong-Pil Son[1], Seongil O[1], Hak-Soo Yu[1], Haesuk Lee[1], Soo Young Kim[1], Youngmin Cho[1], Jin Guk Kim[1], Jongyoon Choi[1], Hyun-Sung Shin[1], Jin Kim[1], BengSeng Phuah[1], HyoungMin Kim[1], Myeong Jun Song[1], Ahn Choi[1], Daeho Kim[1], SooYoung Kim[1], Eun-Bong Kim[1], David Wang[2], Shinhaeng Kang[1], Yuhwan Ro[3], Seungwoo Seo[3], JoonHo Song[3], Jaeyoun Youn[1], Kyomin Sohn[1], Nam Sung Kim[1]

[1]Samsung Electronics, Hwaseong, Korea
[2]Samsung Electronics, San Jose, CA
[3]Samsung Electronics, Suwon, Korea

Samsung Function-in-Memory DRAM (2021)

[Available instruction list for FIM operation]

Type	CMD	Description
Floating Point	ADD	FP16 addition
	MUL	FP16 multiplication
	MAC	FP16 multiply-accumulate
	MAD	FP16 multiply and add
Data Path	MOVE	Load or store data
	FILL	Copy data from bank to GRFs
Control Path	NOP	Do nothing
	JUMP	Jump instruction
	EXIT	Exit instruction

ISSCC 2021 / SESSION 25 / DRAM / 25.4

25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications

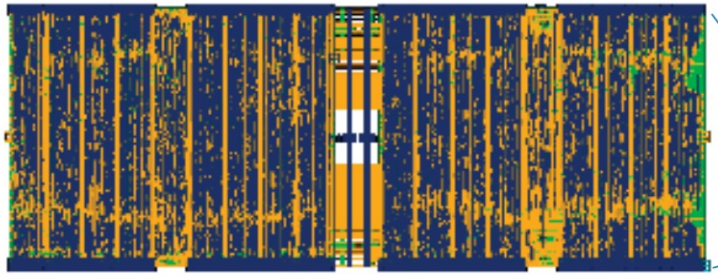
Young-Cheon Kwon[1], Suk Han Lee[1], Jaehoon Lee[1], Sang-Hyuk Kwon[1], Je Min Ryu[1], Jong-Pil Son[1], Seongil O[1], Hak-Soo Yu[1], Haesuk Lee[1], Soo Young Kim[1], Youngmin Cho[1], Jin Guk Kim[1], Jongyoon Choi[1], Hyun-Sung Shin[1], Jin Kim[1], BengSeng Phuah[1], HyoungMin Kim[1], Myeong Jun Song[1], Ahn Choi[1], Daeho Kim[1], SooYoung Kim[1], Eun-Bong Kim[1], David Wang[2], Shinhaeng Kang[1], Yuhwan Ro[3], Seungwoo Seo[3], JoonHo Song[3], Jaeyoun Youn[1], Kyomin Sohn[1], Nam Sung Kim[1]

[1]Samsung Electronics, Hwaseong, Korea
[2]Samsung Electronics, San Jose, CA
[3]Samsung Electronics, Suwon, Korea

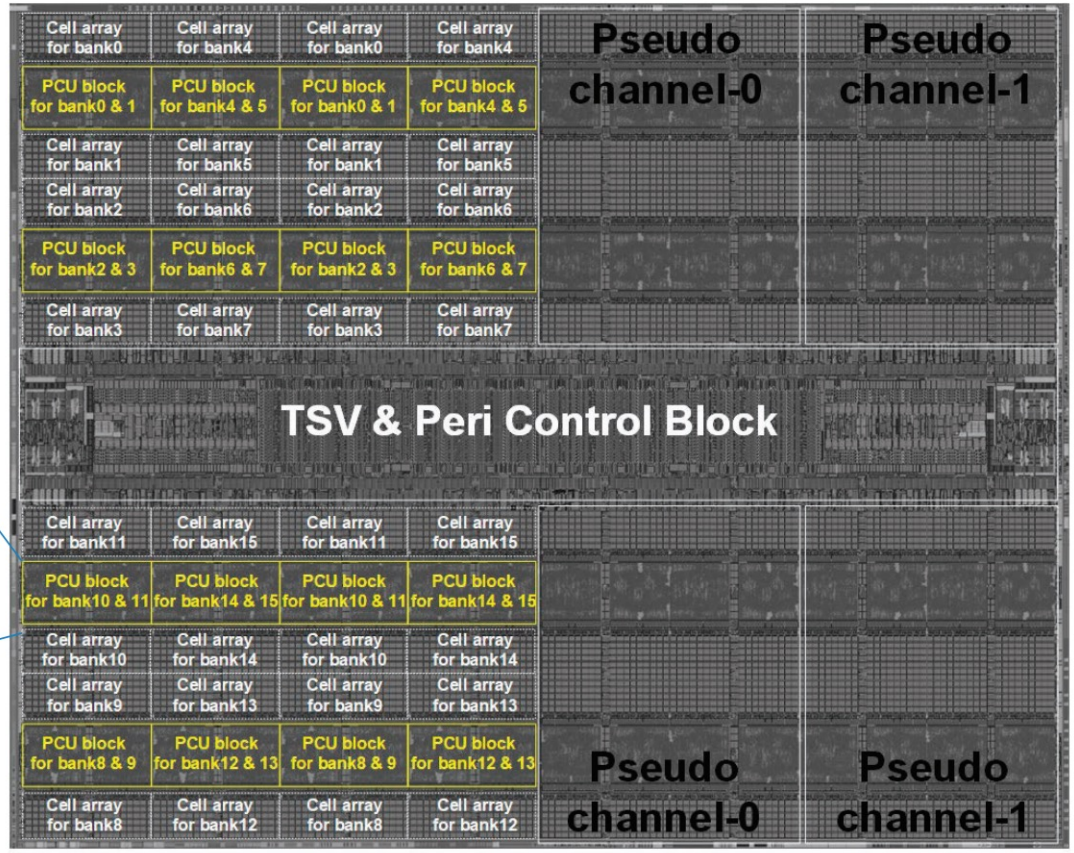
Samsung Function-in-Memory DRAM (2021)

Chip Implementation

- Mixed design methodology to implement FIMDRAM
  - Full-custom + Digital RTL



[Digital RTL design for PCU block]



ISSCC 2021 / SESSION 25 / DRAM / 25.4

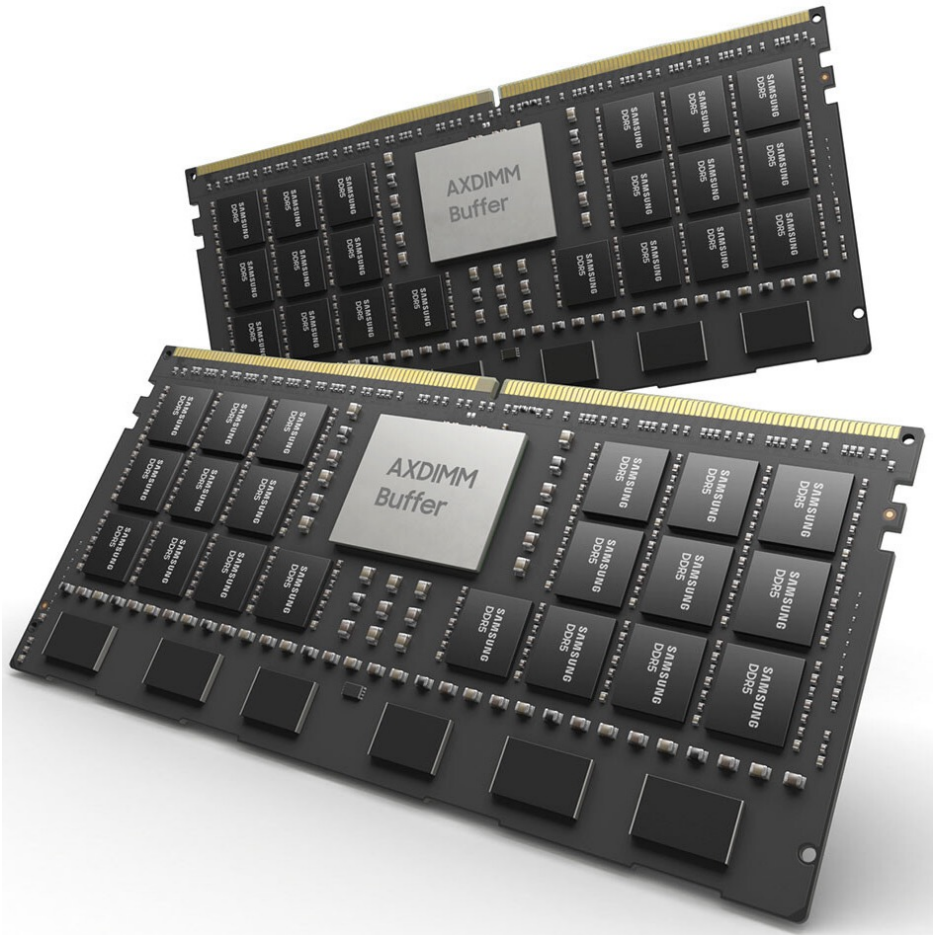
25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications

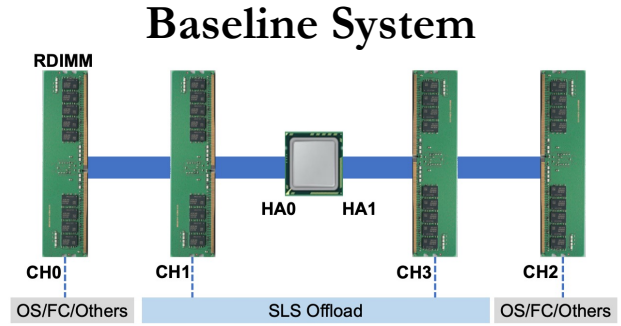
Young-Cheon Kwon[1], Suk Han Lee[1], Jaehoon Lee[1], Sang-Hyuk Kwon[1], Je Min Ryu[1], Jong-Pil Son[1], Seongil O[1], Hak-Soo Yu[1], Haesuk Lee[1], Soo Young Kim[1], Youngmin Cho[1], Jin Guk Kim[1], Jongyoon Choi[1], Hyun-Sung Shin[1], Jin Kim[1], BengSeng Phuah[1], HyoungMin Kim[1], Myeong Jun Song[1], Ahn Choi[1], Daeho Kim[1], SooYoung Kim[1], Eun-Bong Kim[1], David Wang[2], Shinhaeng Kang[1], Yuhwan Ro[3], Seungwoo Seo[3], JoonHo Song[3], Jaeyoun Youn[1], Kyomin Sohn[1], Nam Sung Kim[1]

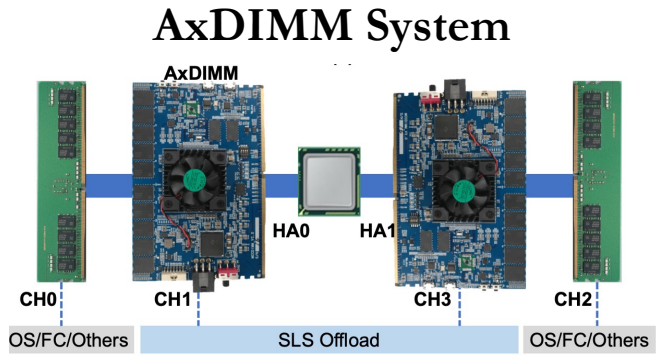
[1]Samsung Electronics, Hwaseong, Korea
[2]Samsung Electronics, San Jose, CA
[3]Samsung Electronics, Suwon, Korea

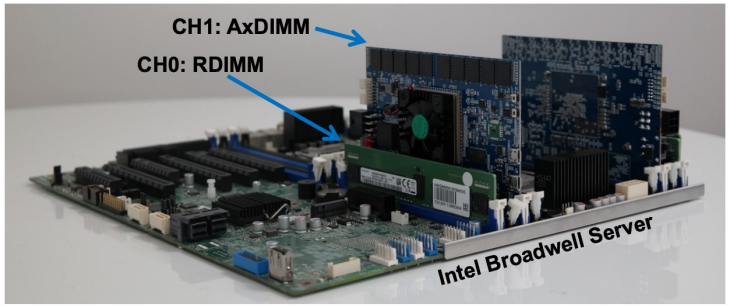
Samsung AxDIMM (2021)

- DDRx-PIM
  - DLRM recommendation system









Ke et al. "Near-Memory Processing in Action: Accelerating Personalized Recommendation with AxDIMM", IEEE Micro (2021)

SK Hynix Accelerator-in-Memory (2022)

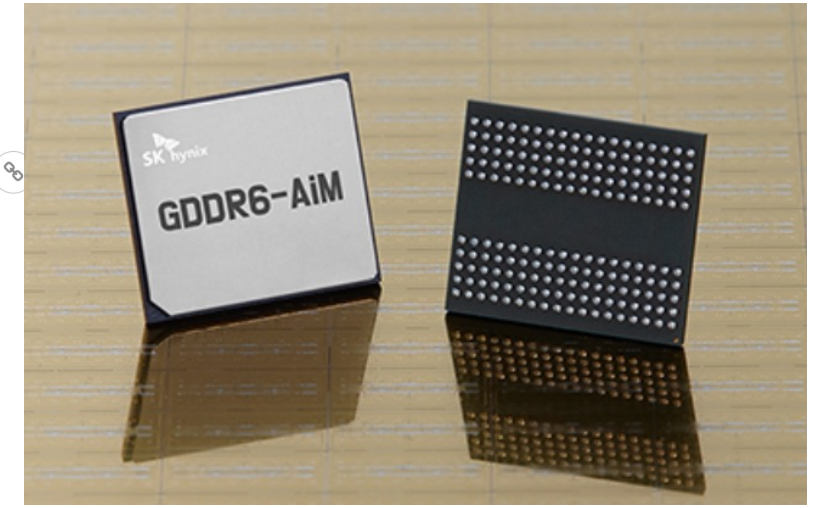
SKhynix NEWSROOM

ENG

INSIGHT | SK hynix STORY | PRESS CENTER | MULTIMEDIA | Search

SK hynix Develops PIM, Next-Generation AI Accelerator

February 16, 2022



Seoul, February 16, 2022

SK hynix (or "the Company", www.skhynix.com) announced on February 16 that it has developed PIM*, a next-generation memory chip with computing capabilities.

*PIM(Processing In Memory): A next-generation technology that provides a solution for data congestion issues for AI and big data by adding computational functions to semiconductor memory

It has been generally accepted that memory chips store data and CPU or GPU, like human brain, process data. SK hynix, following its challenge to such notion and efforts to pursue innovation in the next-generation smart memory, has found a breakthrough solution with the development of the latest technology.

SK hynix plans to showcase its PIM development at the world's most prestigious semiconductor conference, 2022 ISSCC*, in San Francisco at the end of this month. The company expects continued efforts for innovation of this technology to bring the memory-centric computing, in which semiconductor memory plays a central role, a step closer to the reality in devices such as smartphones.

*ISSCC: The International Solid-State Circuits Conference will be held virtually from Feb. 20 to Feb. 24 this year with a theme of "Intelligent Silicon for a Sustainable World"

For the first product that adopts the PIM technology, SK hynix has developed a sample of GDDR6-AiM (Accelerator* in memory). The GDDR6-AiM adds computational functions to GDDR6* memory chips, which process data at 16Gbps. A combination of GDDR6-AiM with CPU or GPU instead of a typical DRAM makes certain computation speed 16 times faster. GDDR6-AiM is widely expected to be adopted for machine learning, high-performance computing, and big data computation and storage.

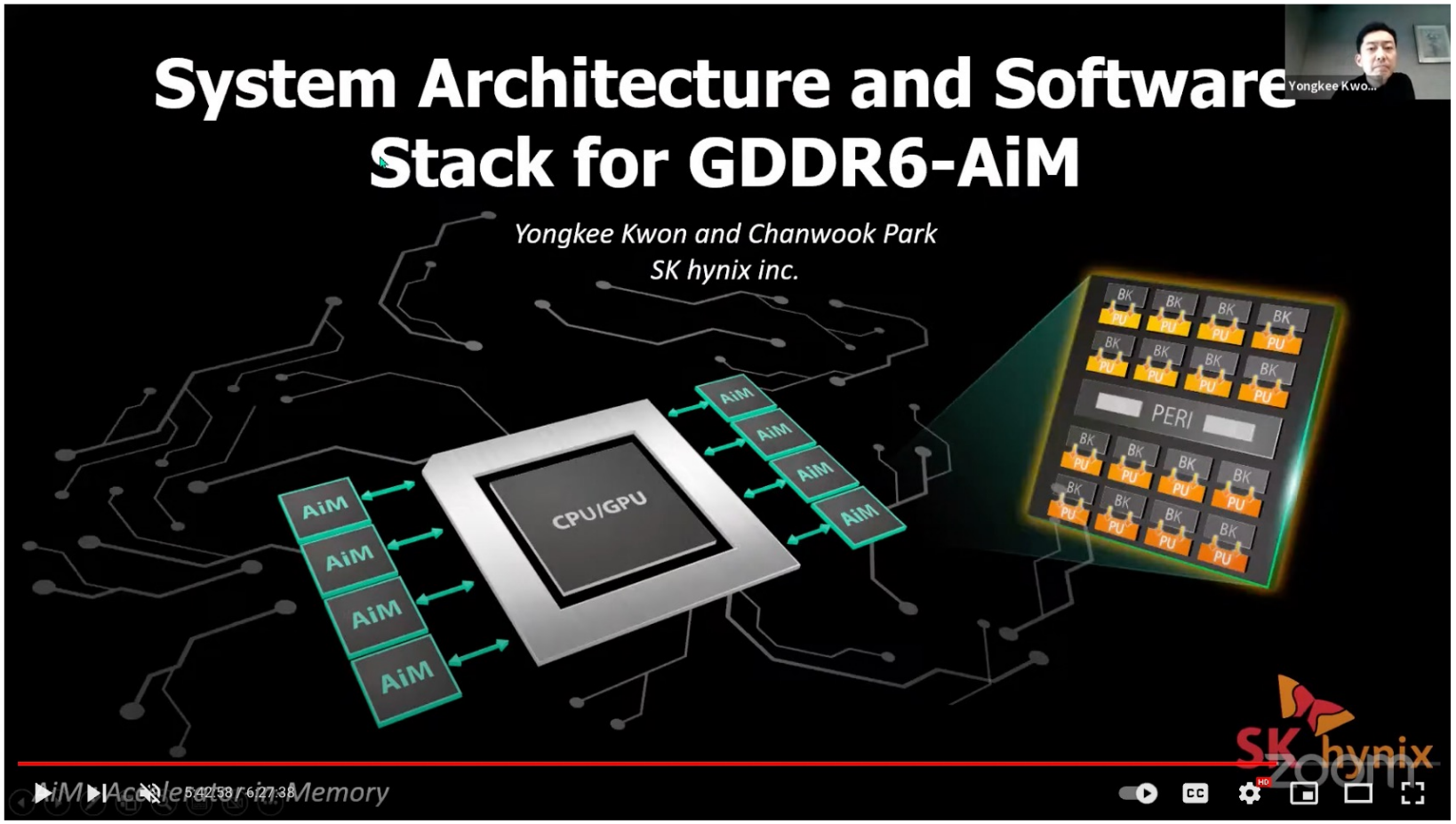
11.1 A 1ynm 1.25V 8Gb, 16Gb/s/pin GDDR6-based Accelerator-in-Memory supporting 1TFLOPS MAC Operation and Various Activation Functions for Deep-Learning Applications

Seongju Lee, SK hynix, Icheon, Korea

In Paper 11.1, SK Hynix describes an 1ynm, GDDR6-based accelerator-in-memory with a command set for deep-learning operation. The 8Gb design achieves a peak throughput of 1TFLOPS with 1GHz MAC operations and supports major activation functions to improve accuracy.

https://news.skhynix.com/sk-hynix-develops-pim-next-generation-ai-accelerator/

SK Hynix Accelerator-in-Memory (2022)



ASPLOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads

Onur Mutlu Lectures
32.1K subscribers

Analytics | Edit video | 33 | Share | Download | Clip | Save | ...

1,146 views Streamed live on Mar 26, 2023 Livestream - Data-Centric Architectures: Fundamentally Improving Performance and Energy (Spring 2023)
ASPLOS 2023 Tutorial: Real-world Processing-in-Memory Systems for Modern Workloads
https://events.safari.ethz.ch/asplos-...

https://www.youtube.com/watch?v=oYCaLcT0Kmo

AliBaba PIM Recommendation System (2022)

ISSCC 2022 / February 24, 2022 / 8:30 AM

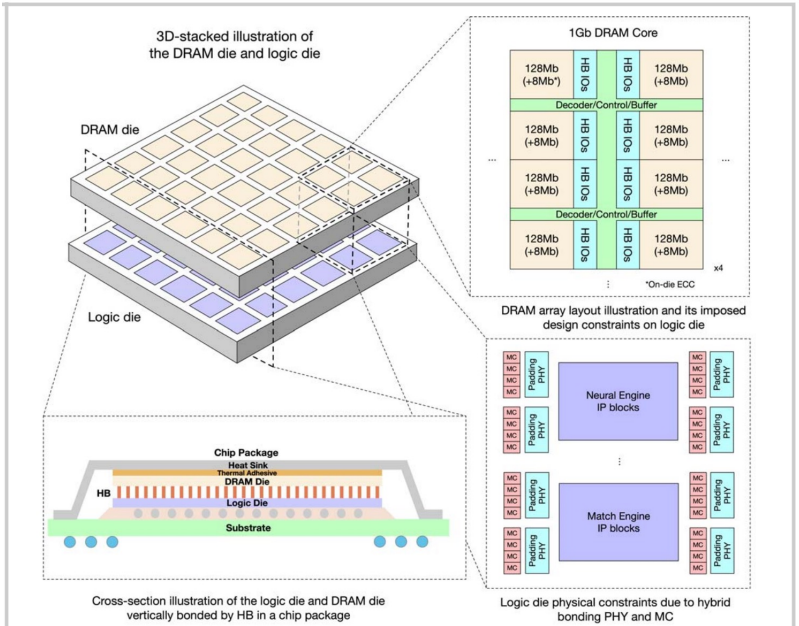
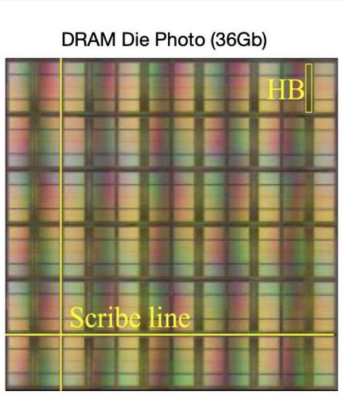
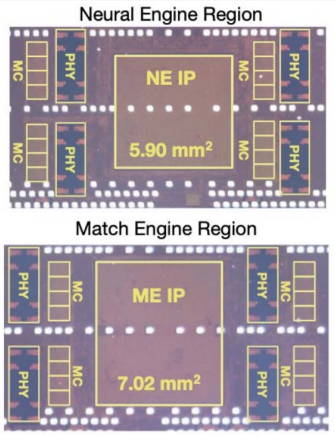


Figure 29.1.2: Illustration of 3D-stacked chip, cross-illustration of package, DRAM array layout and design blocks on logic die.





DRAM Die		
Technology	25nm	
Area	Total*	602.22 mm²
	Neural Engine	32 mm²
	Match Engine	32 mm²
Voltage	1.1 V	
Frequency (max)	150 MHz	
Power	300 mW per 1Gb	
Bandwidth**	153.60 GB/s / 1.38 TB/s	

* Total die size of test chip
** Memory bandwidth of NE and ME / Total bandwidth of DRAM die

Logic Die		
Technology	55nm	
Area	Total*	602.22 mm²
	Neural Engine	5.90 mm²
	Match Engine	7.02 mm²
# of MC	16 per IP	
Voltage	1.2 V	
Frequency	300 MHz	
Power	977.70 mW	
Precision	INT8	

Figure 29.1.7: Die micrographs of DRAM die, NE and ME. Detailed specifications of DRAM die and logic die.

29.1 184QPS/W 64Mb/mm² 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System

Dimin Niu[1], Shuangchen Li[1], Yuhao Wang[1], Wei Han[1], Zhe Zhang[2], Yijin Guan[2], Tianchan Guan[3], Fei Sun[1], Fei Xue[1], Lide Duan[1], Yuanwei Fang[1], Hongzhong Zheng[1], Xiping Jiang[4], Song Wang[4], Fengguo Zuo[4], Yubing Wang[4], Bing Yu[4], Qiwei Ren[4], Yuan Xie[1]

SK Hynix CXL Processing Near Memory (2023)

IEEE COMPUTER ARCHITECTURE LETTERS, VOL. 22, NO. 1, JANUARY-JUNE

Computational CXL-Memory Solution for Accelerating Memory-Intensive Applications

Joonseop Sim, Soohong Ahn, Taeyoung Ahn, Seungyong Lee, Myunghyun Rhee, Jooyoung Kim, Kwangsik Shin, Donguk Moon, Euiseok Kim, and Kyoung Park

Abstract—CXL interface is the up-to-date technology that enables effective memory expansion by providing a memory-sharing protocol in configuring heterogeneous devices. However, its limited physical bandwidth can be a significant bottleneck for emerging data-intensive applications. In this work, we propose a novel CXL-based memory disaggregation architecture with a real-world prototype demonstration, which overcomes the bandwidth limitation of the CXL interface using near-data processing. The experimental results demonstrate that our design achieves up to 1.9× better performance/power efficiency than the existing CPU system.

Index Terms—Compute express link (CXL), near-data-processing (NDP)

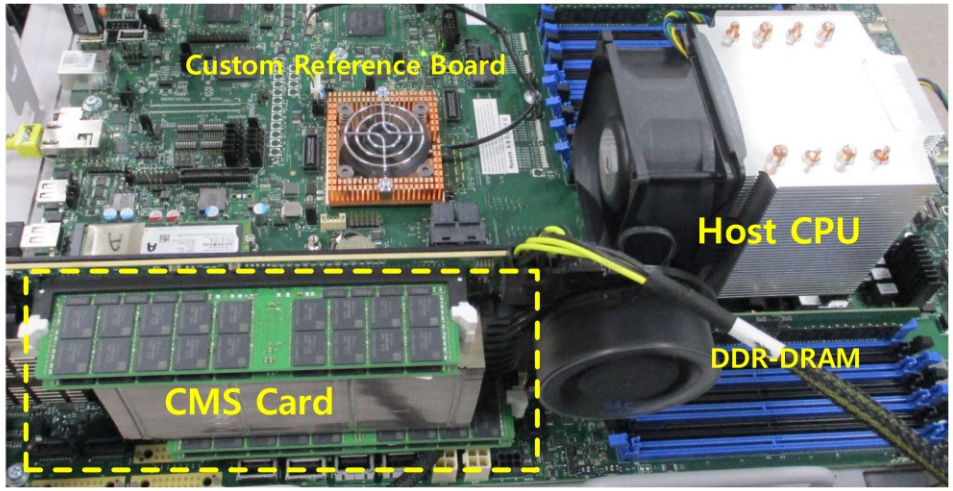


Fig. 6. FPGA prototype of proposed CMS card.

Samsung CXL Processing Near Memory (2023)

Samsung Processing in Memory Technology at Hot Chips 2023

By **Patrick Kennedy** - August 28, 2023

Samsung PIM PNM For Transformer Based AI HC35_Page_24

https://www.servethehome.com/samsung-processing-in-memory-technology-at-hot-chips-2023/

Concluding Remarks

Challenge and Opportunity for Future

Fundamentally
Energy-Efficient
(Data-Centric)
Computing Architectures

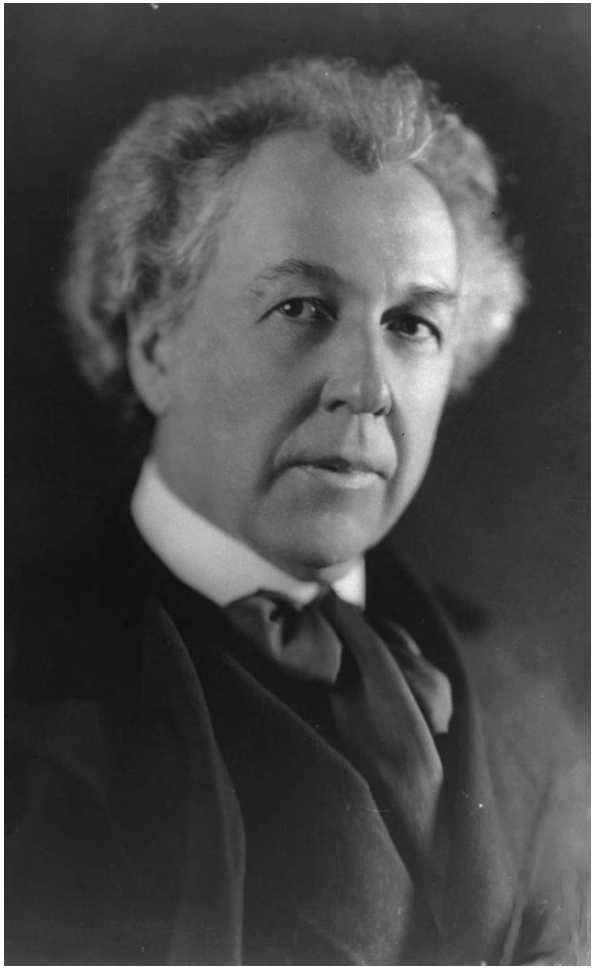
Challenge and Opportunity for Future

Fundamentally
High-Performance
(Data-Centric)
Computing Architectures

Computing Architectures with Minimal Data Movement

A Quote from A Famous Architect

- "architecture [...] based upon principle, and not upon precedent"



Precedent-Based Design?

- "architecture […] based upon principle, and not upon precedent"

Principled Design

- "architecture [...] based upon principle, and not upon precedent"


www.GreatBuildings.com

The Overarching Principle

Organic architecture

From Wikipedia, the free encyclopedia

Organic architecture is a philosophy of architecture which promotes harmony between human habitation and the natural world through design approaches so sympathetic and well integrated with its site, that buildings, furnishings, and surroundings become part of a unified, interrelated composition.

A well-known example of organic architecture is Fallingwater, the residence Frank Lloyd Wright designed for the Kaufmann family in rural Pennsylvania. Wright had many choices to locate a home on this large site, but chose to place the home directly over the waterfall and creek creating a close, yet noisy dialog with the rushing water and the steep site. The horizontal striations of stone masonry with daring cantilevers of colored beige concrete blend with native rock outcroppings and the wooded environment.

Another Example: Precedent-Based Design

Source: http://cookiemagik.deviantart.com/art/Train-station-207266944

Principled Design

Source: By Toni_V, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=4087256

Another Principled Design

Source: By Martín Gómez Tagle - Lisbon, Portugal, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=13764903

Source: http://www.arcspace.com/exhibitions/unsorted/santiago-calatrava/

Another Principled Design

Principle Applied to Another Structure

Source: By 準建築人手札網站 Forgemind ArchiMedia - Flickr: IMG_2489.JPG, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=51493556, https://en.wikipedia.org/wiki/Santiago_Calatrava

Source: https://www.dezeen.com/2016/08/29/santiago-calatrava-oculus-world-trade-center-transportation-hub-new-york-photographs-hufton-crow/

The Overarching Principle

Zoomorphic architecture

From Wikipedia, the free encyclopedia

Zoomorphic architecture is the practice of using animal forms as the inspirational basis and blueprint for architectural design. "While animal forms have always played a role adding some of the deepest layers of meaning in architecture, it is now becoming evident that a new strand of biomorphism is emerging where the meaning derives not from any specific representation but from a more general allusion to biological processes."[1]

Some well-known examples of Zoomorphic architecture can be found in the TWA Flight Center building in New York City, by Eero Saarinen, or the Milwaukee Art Museum by Santiago Calatrava, both inspired by the form of a bird's wings.[3]

Overarching Principles for Computing?

Source: http://spectrum.ieee.org/image/MjYzMzAyMg.jpeg

Concluding Remarks

- It is time to design principled system architectures to solve the memory problem
- We must design systems to be **balanced, high-performance, and energy-efficient** → **memory-centric**
  - Enable computation capabilities in memory
- This can
  - Lead to **orders-of-magnitude** improvements
  - **Enable new applications & computing platforms**
  - **Enable better understanding of nature**
  - **…**
- Future of **truly memory-centric computing** is bright
  - We need to do research & design across the computing stack

Fundamentally Better Architectures

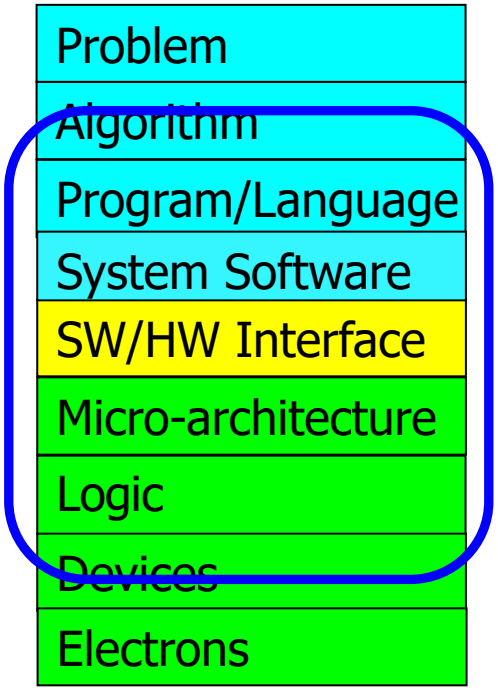
Data-centric

Data-driven

Data-aware

We Need to Revisit the Entire Stack

- With a **memory-centric mindset**



We can get there step by step

PIM Review and Open Problems

A Modern Primer on Processing in Memory

Onur Mutlu[a,b], Saugata Ghose[b,c], Juan Gómez-Luna[a], Rachata Ausavarungnirun[d]

SAFARI Research Group

[a] *ETH Zürich*
[b] *Carnegie Mellon University*
[c] *University of Illinois at Urbana-Champaign*
[d] *King Mongkut's University of Technology North Bangkok*

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun,
"A Modern Primer on Processing in Memory"
Invited Book Chapter in ***Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann***, Springer, to be published in 2021.

https://arxiv.org/pdf/2012.03112.pdf

Open Source Tools: SAFARI GitHub

SAFARI Research Group at ETH Zurich and Carnegie Mellon University

Site for source code and tools distribution from SAFARI Research Group at ETH Zurich and Carnegie Mellon University.

440 followers · ETH Zurich and Carnegie Mellon U... · https://safari.ethz.ch/ · omutlu@gmail.com

Overview | Repositories 98 | Projects | Packages | People 13

ramulator Public

A Fast and Extensible DRAM Simulator, with built-in support for modeling many different DRAM technologies including DDRx, LPDDRx, GDDRx, WIOx, HBMx, and various academic proposals. Described in the...

C++ ☆ 532 ⑂ 206

prim-benchmarks Public

PrIM (Processing-In-Memory benchmarks) is the first benchmark suite for a real-world processing-in-memory (PIM) architecture. PrIM is developed to evaluate, analyze, and characterize the first publ...

C ☆ 126 ⑂ 47

MQSim Public

MQSim is a fast and accurate simulator modeling the performance of modern multi-queue (MQ) SSDs as well as traditional SATA based SSDs. MQSim faithfully models new high-bandwidth protocol implement...

C++ ☆ 268 ⑂ 143

rowhammer Public

Source code for testing the Row Hammer error mechanism in DRAM devices. Described in the ISCA 2014 paper by Kim et al. at http://users.ece.cmu.edu/~omutlu/pub/dram-row-hammer_isca14.pdf.

C ☆ 211 ⑂ 42

SoftMC Public

SoftMC is an experimental FPGA-based memory controller design that can be used to develop tests for DDR3 SODIMMs using a C++ based API. The design, the interface, and its capabilities and limitatio...

Verilog ☆ 120 ⑂ 27

Pythia Public

A customizable hardware prefetching framework using online reinforcement learning as described in the MICRO 2021 paper by Bera et al. (https://arxiv.org/pdf/2109.12021.pdf).

C++ ☆ 109 ⑂ 34

https://github.com/CMU-SAFARI/

Referenced Papers, Talks, Artifacts

- All are available at

  https://people.inf.ethz.ch/omutlu/projects.htm

  https://www.youtube.com/onurmutlulectures

  https://github.com/CMU-SAFARI/

Funding Acknowledgments

- Alibaba, AMD, ASML, Google, Facebook, Hi-Silicon, HP Labs, Huawei, IBM, Intel, Microsoft, Nvidia, Oracle, Qualcomm, Rambus, Samsung, Seagate, VMware, Xilinx
- NSF
- NIH
- GSRC
- SRC
- CyLab
- EFCL
- SNSF
- ACCESS

Thank you!

Memory Systems and Memory-Centric Computing

Lecture 4.1: Memory-Centric Computing III

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu

18 July 2024

HiPEAC ACACES Summer School 2024

SAFARI

ETH zürich