Managing Memory Tiers with CXL in Virtualized Environments

Yuhong Zhong ♔ ⊞ Daniel S. Berger ⊞ W Carl Waldspurger* Ryan Wee ♔

Ishwar Agarwal i Rajat Agarwal i Frank Hady i Karthik Kumar i

Mark D. Hill W Mosharaf Chowdhury M Asaf Cidon ♔

♔ *Columbia University* ⊞ *Microsoft Azure* W *University of Washington*

**Carl Waldspurger Consulting* i *Intel* W *University of Wisconsin-Madison* M *University of Michigan*

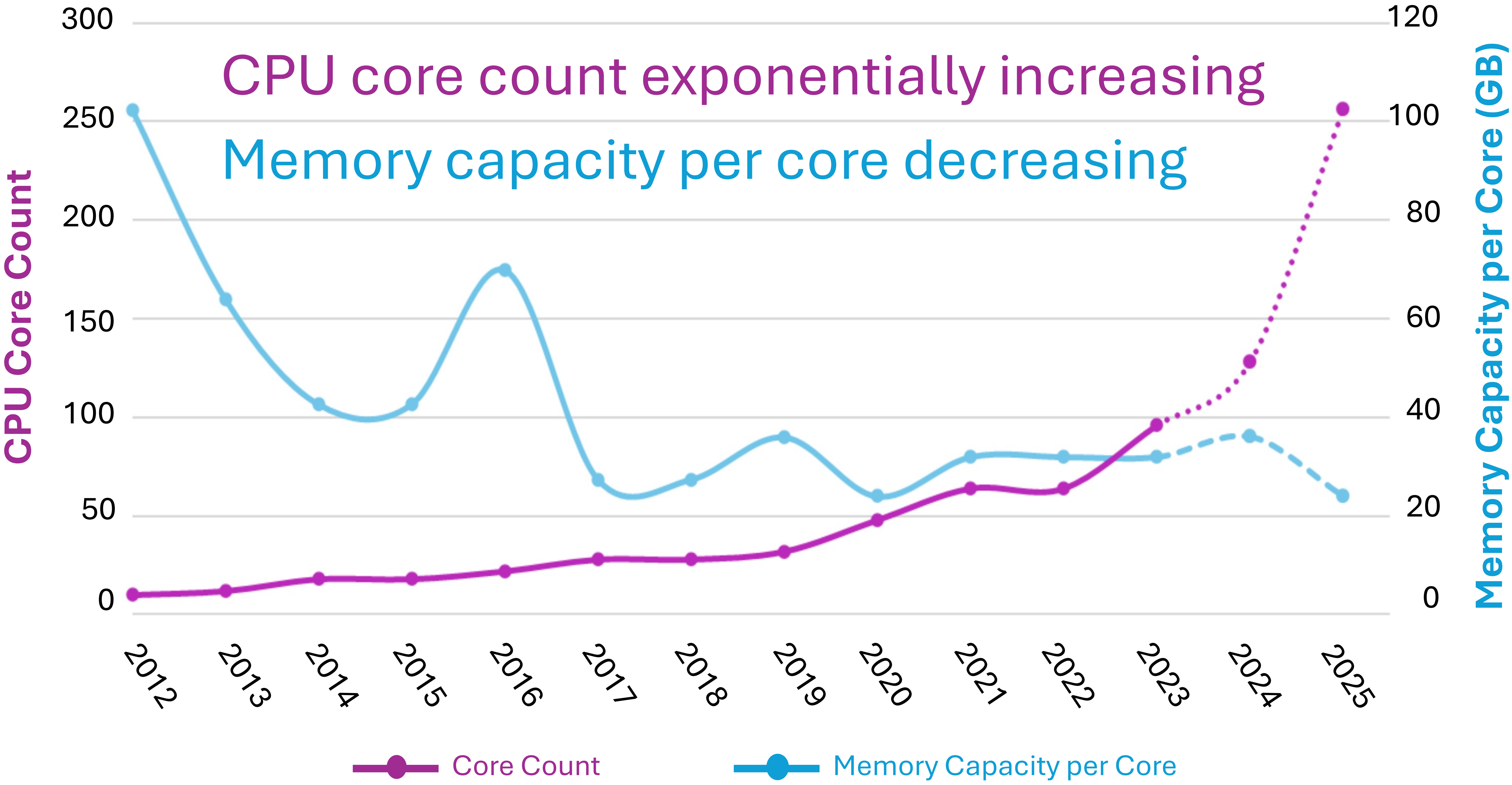
Executive Summary

Background:

- CPU **core counts scaling faster** than memory capacity
- CXL enables **second-tier memory** to facilitate core scaling
- But CXL adds latency that hurts performance if not mitigated
- Software tiering helps some but **is not well suited for public clouds**

Contributions:

- Intel Flat Memory Mode: First **hardware-managed** memory tiering for CXL
    - But still has **limitations** that degrade workloads
- Memstrata: Memory allocator for hardware tiering to **mitigate outliers**
- Slowdown reduces to ~5% vs. unattainable one-tier memory

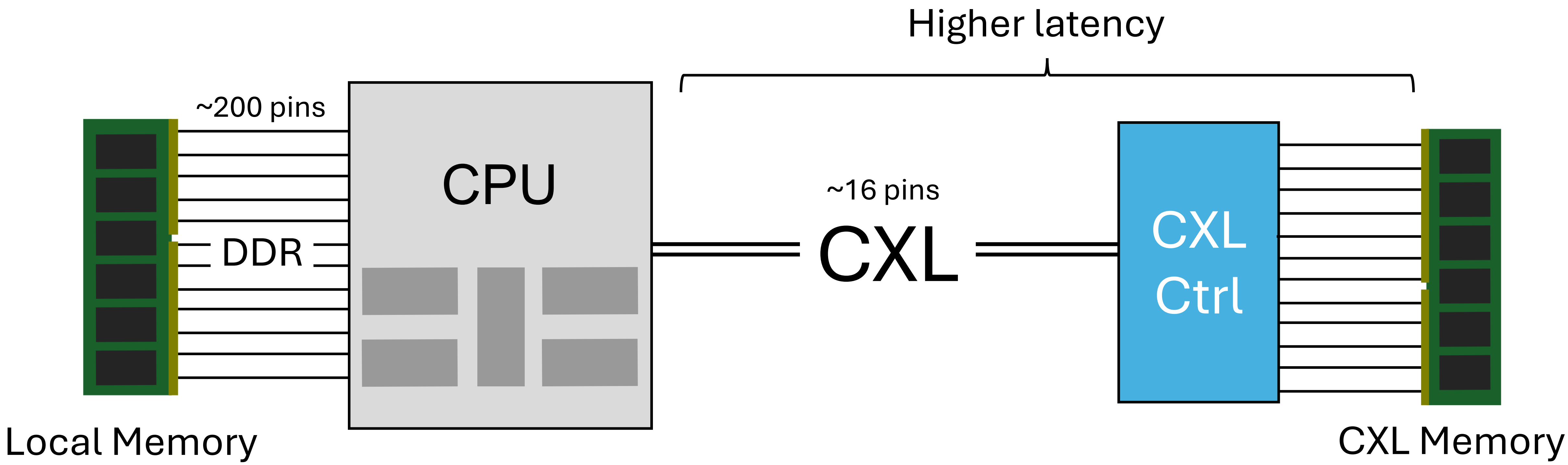


CPU core count exponentially increasing

Memory capacity per core decreasing

Source: Micron’s Perspective on Impact of CXL on DRAM Bit Growth Rate

CXL Enables Memory Capacity Scaling



Higher CXL Latency Can Degrade Workloads

- CXL latency (220 ns) ≈ 2x local memory latency (100 ns)
- CXL slowdowns workloads by up to 62%
- Memory tiering: place data between local and CXL memory

Cloud requirements for CXL include:

- **Minimal slowdown**
- **Low CPU overhead**
- **Huge page friendly**

Das Sharma et al., An Introduction to the Compute Express Link (CXL) Interconnect, CSUR 2024

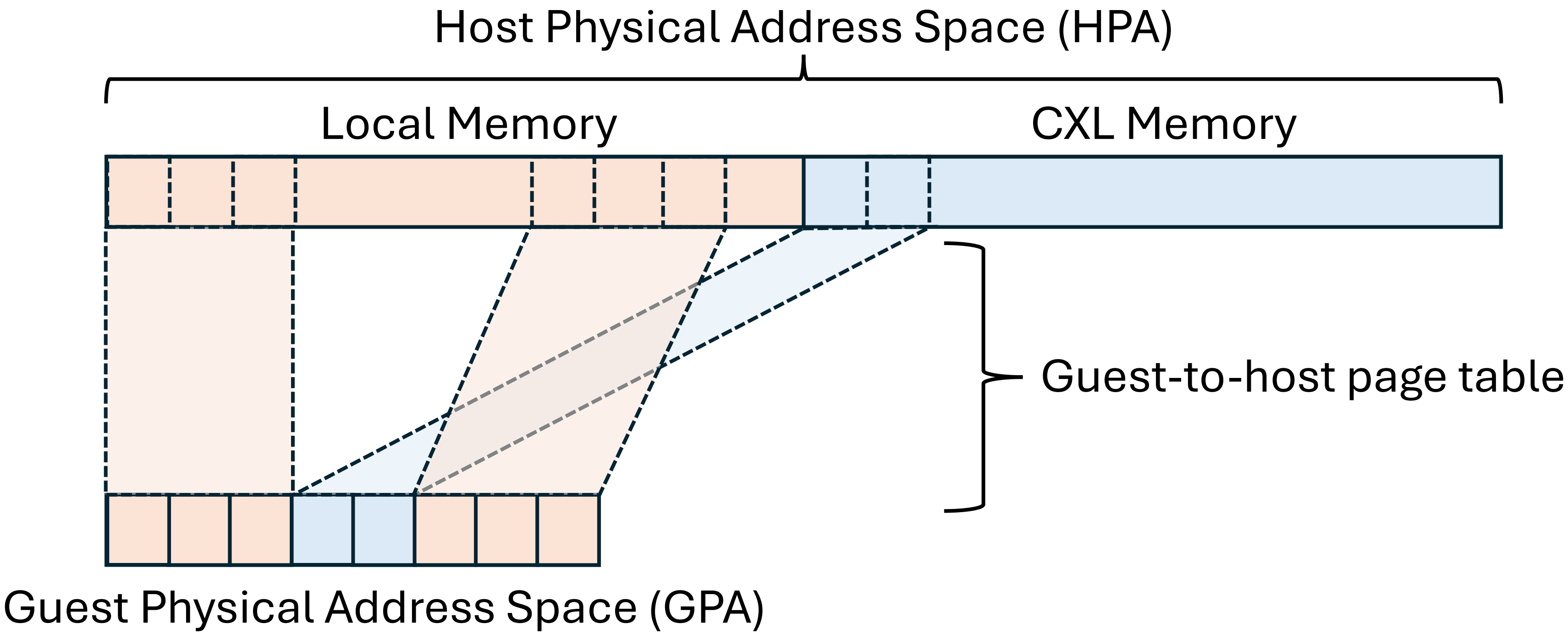
Combining Software and Hardware for Memory Tiering

	Software Tiering	Hardware Tiering	Software + Hardware Tiering
	HeMem (SOSP ’21) TPP (ASPLOS ’23) MEMTIS (SOSP ’23)	**Intel Flat Memory Mode**	Intel Flat Memory Mode and **Memstrata**
Minimal slowdown	⚠ High tail slowdown	⚠ High tail slowdown	✅ Minimal slowdown
Low CPU overhead	❌ High overhead	✅ Low overhead	✅ Low overhead
Huge page friendly	❌ Unfriendly	✅ Friendly	✅ Friendly

Introduced in this work

Prior Work: Software-Managed Memory Tiering

Use **hypervisor/OS** to **identify popular pages** and **decide page placement**


Host Physical Address Space (HPA)
Local Memory
CXL Memory
Guest-to-host page table
Guest Physical Address Space (GPA)

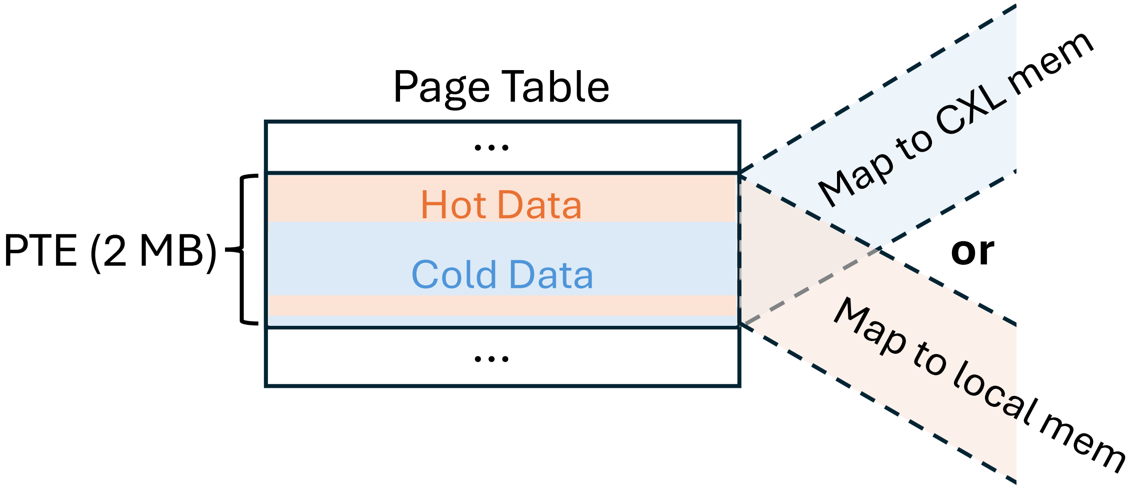
Software Tiering at Odds With Virtualization

Issue 1: High CPU overhead

- Instruction sampling (PEBS, IBS) is disabled in clouds
- Frequent page table scans incur excessive CPU overhead

Issue 2: Huge page penalty[1]

- Virtualization uses larger page sizes (2 MB, 1 GB) to reduce TLB cost



Larger Pages

- ✅ **Lower TLB cost**
- ❌ **Coarse data placement**

[1] Calciu et al., Rethinking Software Runtimes for Disaggregated Memory, ASPLOS 2021

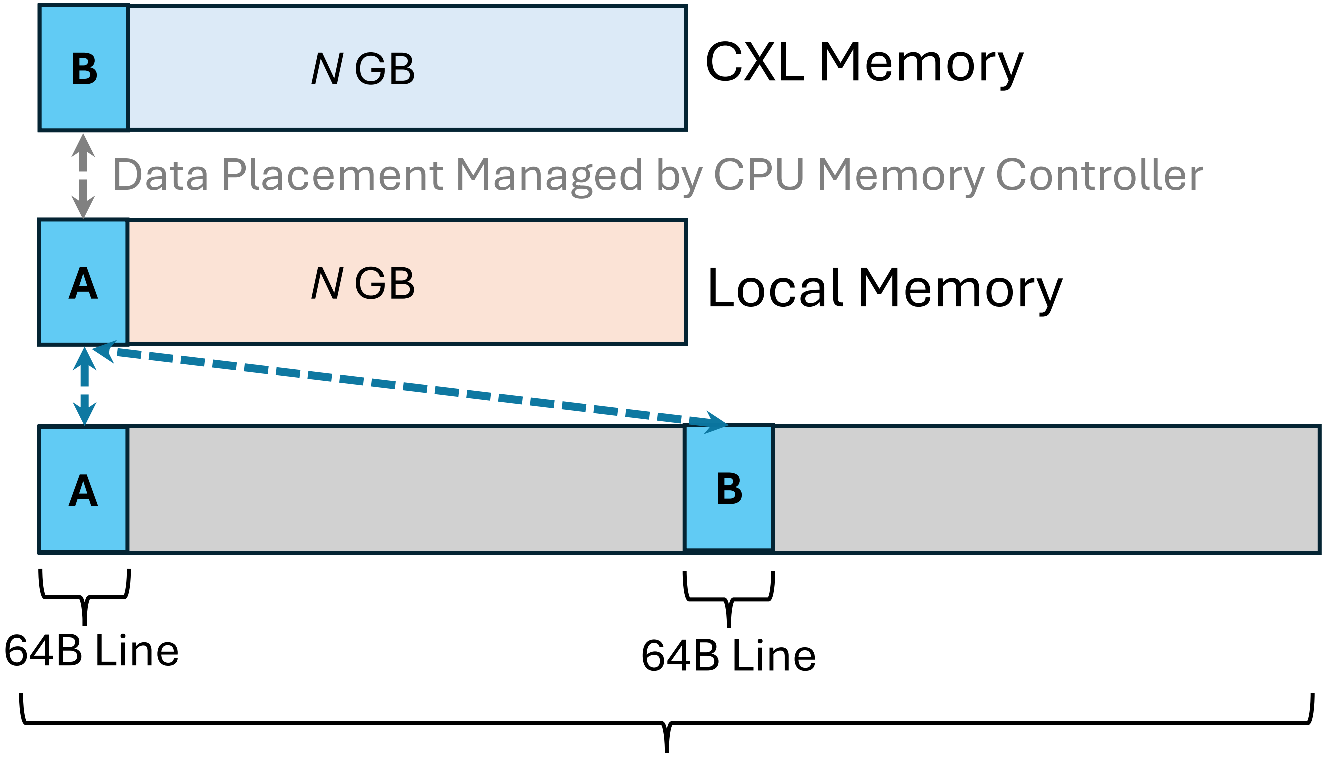
Introducing Hardware Tiering for CXL

We introduce **Intel Flat Memory Mode**:

- First hardware-managed cacheline-granular memory tiering for CXL
- Data placement managed by the CPU memory controller
  - Zero CPU overhead
  - Huge page friendly
- Available in Intel Xeon 6 Processor

Associativity and Mapping of Intel Flat Memory Mode

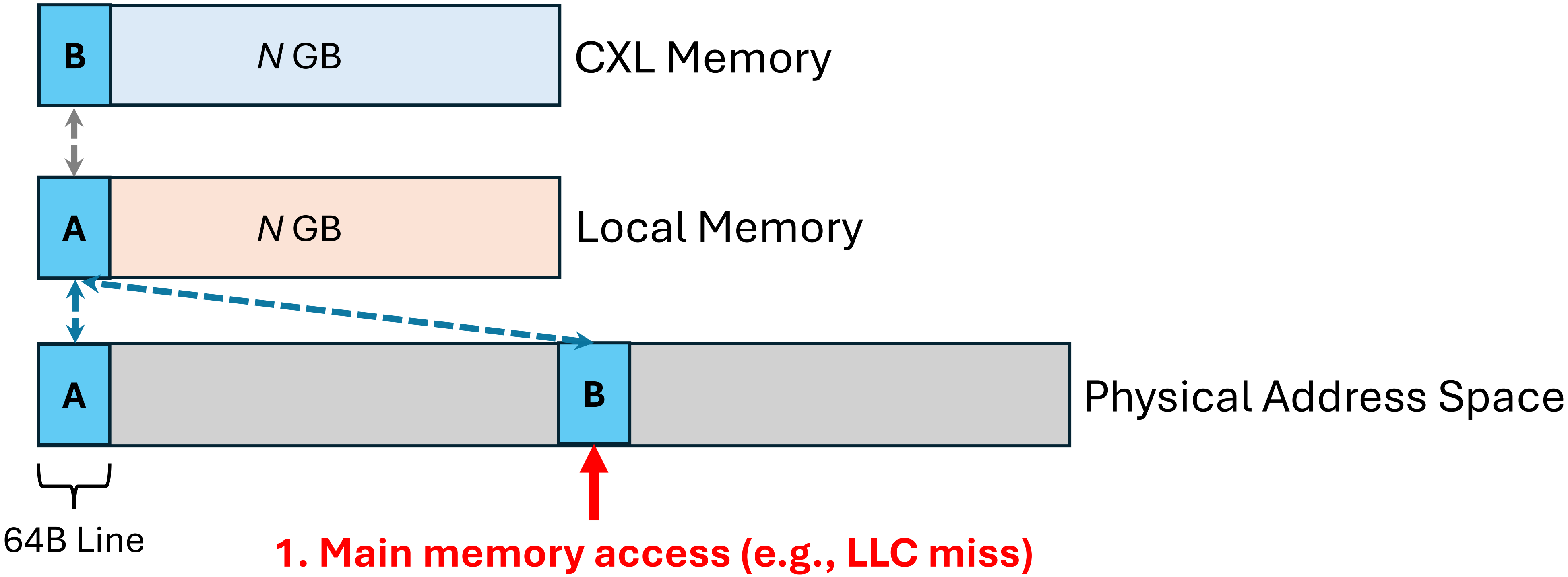
Local memory as a **direct-mapped**, **exclusive** cache of CXL memory



Physical Memory Address Space (2*N* GB)

Local Memory Miss in Intel Flat Memory Mode

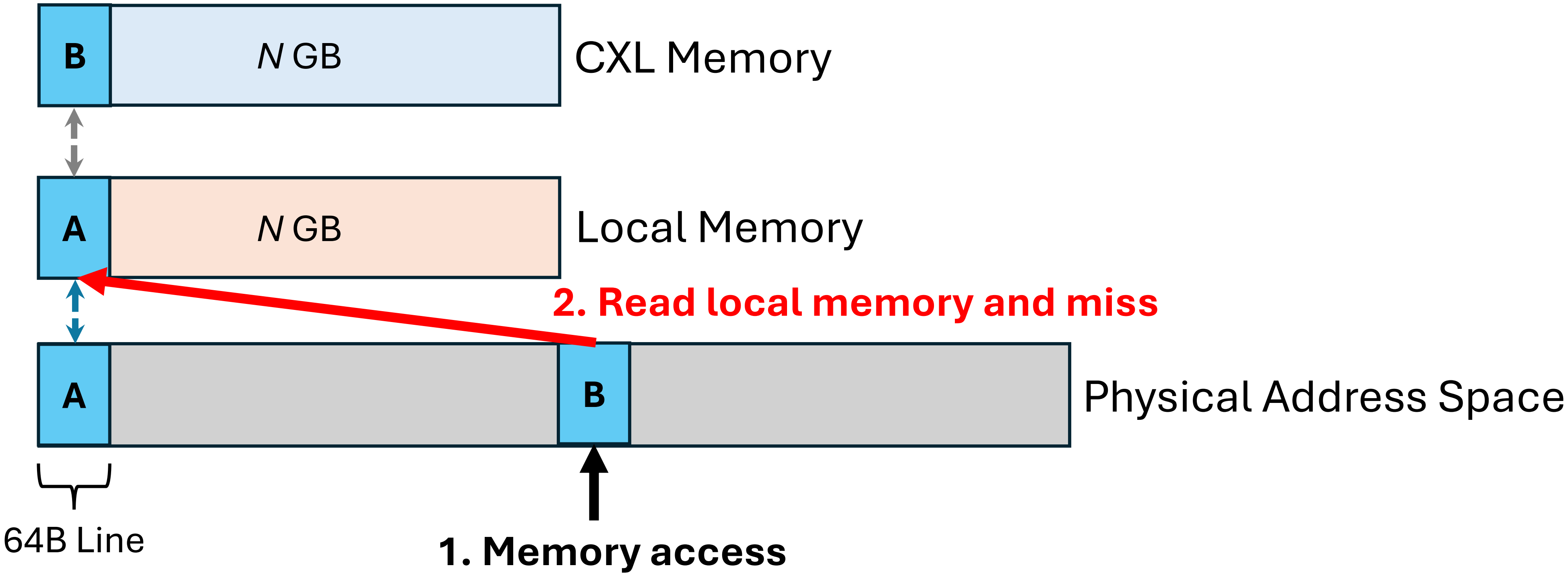
When a main memory access misses in local memory, the hardware will "**swap**" the two cache lines



1. Main memory access (e.g., LLC miss)

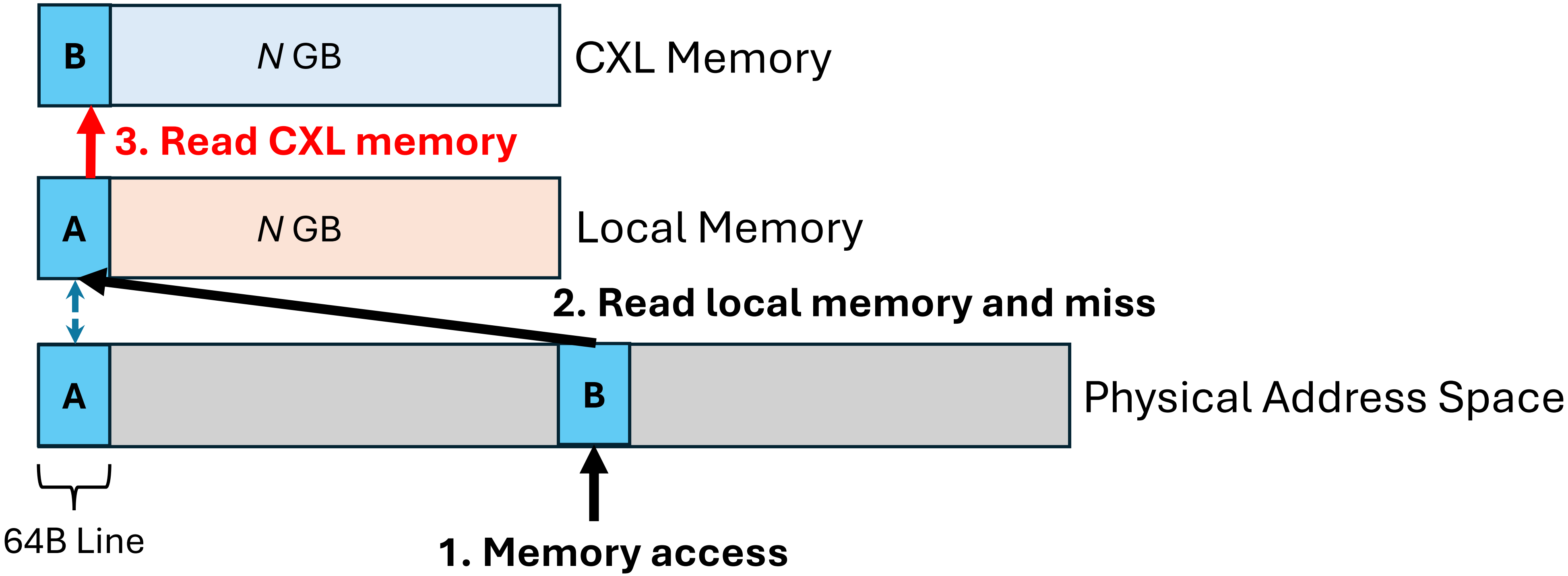
Local Memory Miss in Intel Flat Memory Mode

When a main memory access misses in local memory, the hardware will "**swap**" the two cache lines



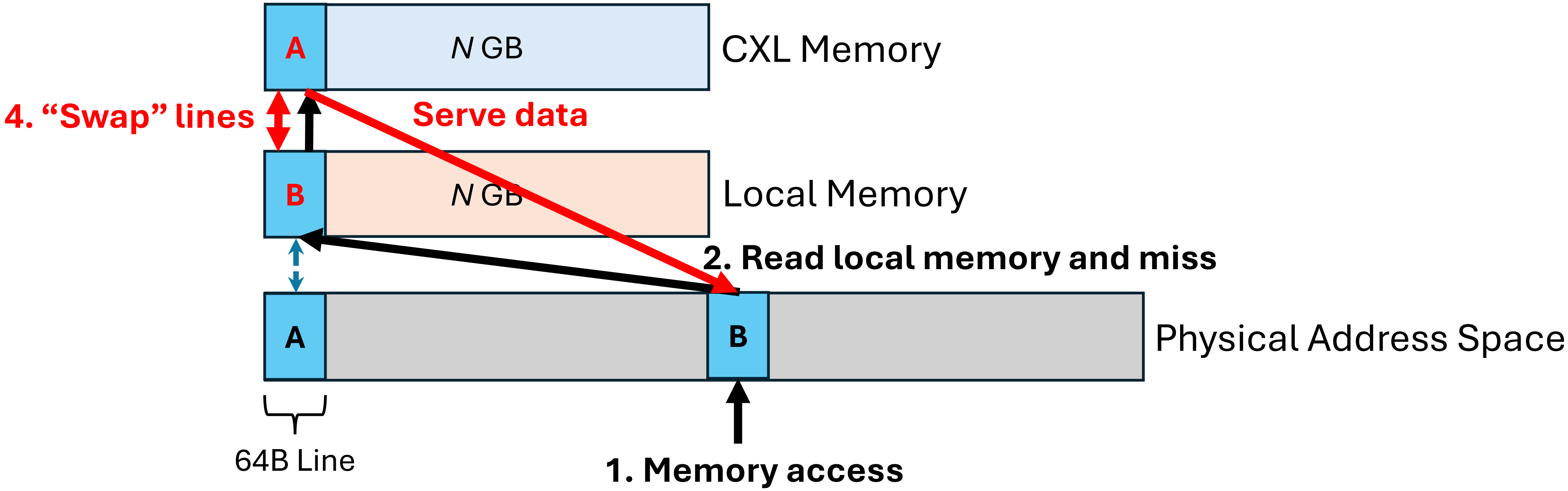
Local Memory Miss in Intel Flat Memory Mode

When a main memory access misses in local memory, the hardware will “**swap**” the two cache lines



Local Memory Miss in Intel Flat Memory Mode

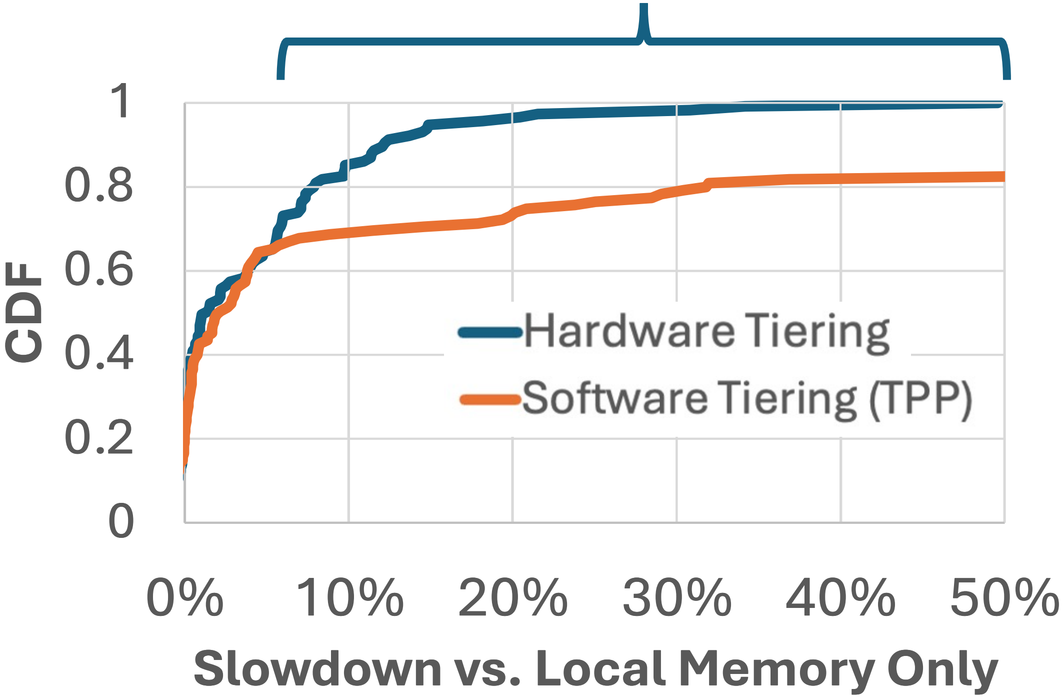
When a main memory access misses in local memory, the hardware will “**swap**” the two cache lines



Hardware Tiering Alone Still Has Limitations

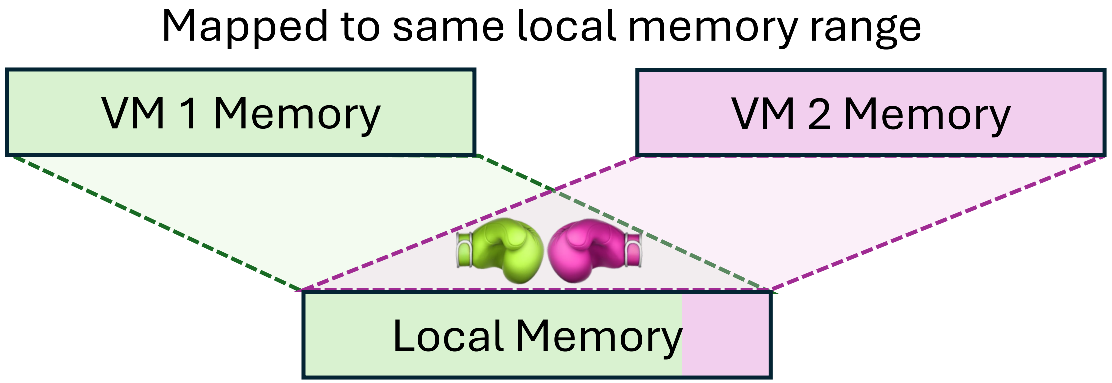
Challenge 1: Some workloads have **heavy local memory misses**

26% workloads have > 5% slowdown (“**outlier**” workloads)



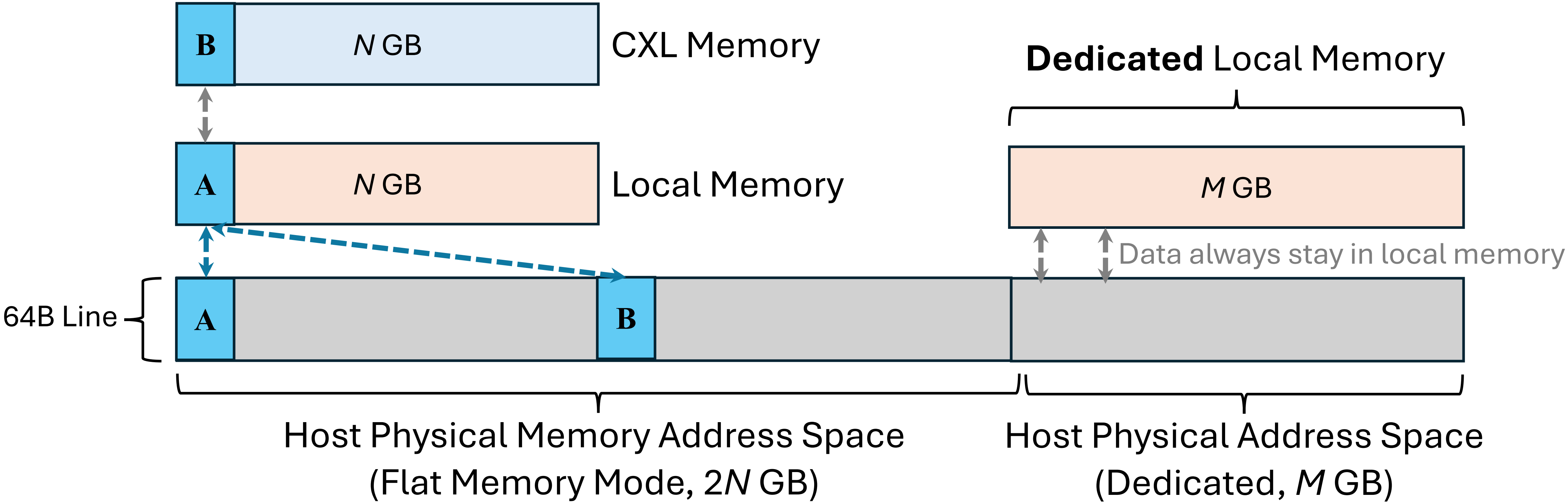
Challenge 2: **No performance isolation** across VMs

Local memory contention across VMs (more than 50% slowdown)



Adding Dedicated Local Memory for Outliers

Question: How to allocate dedicated local memory across VMs?

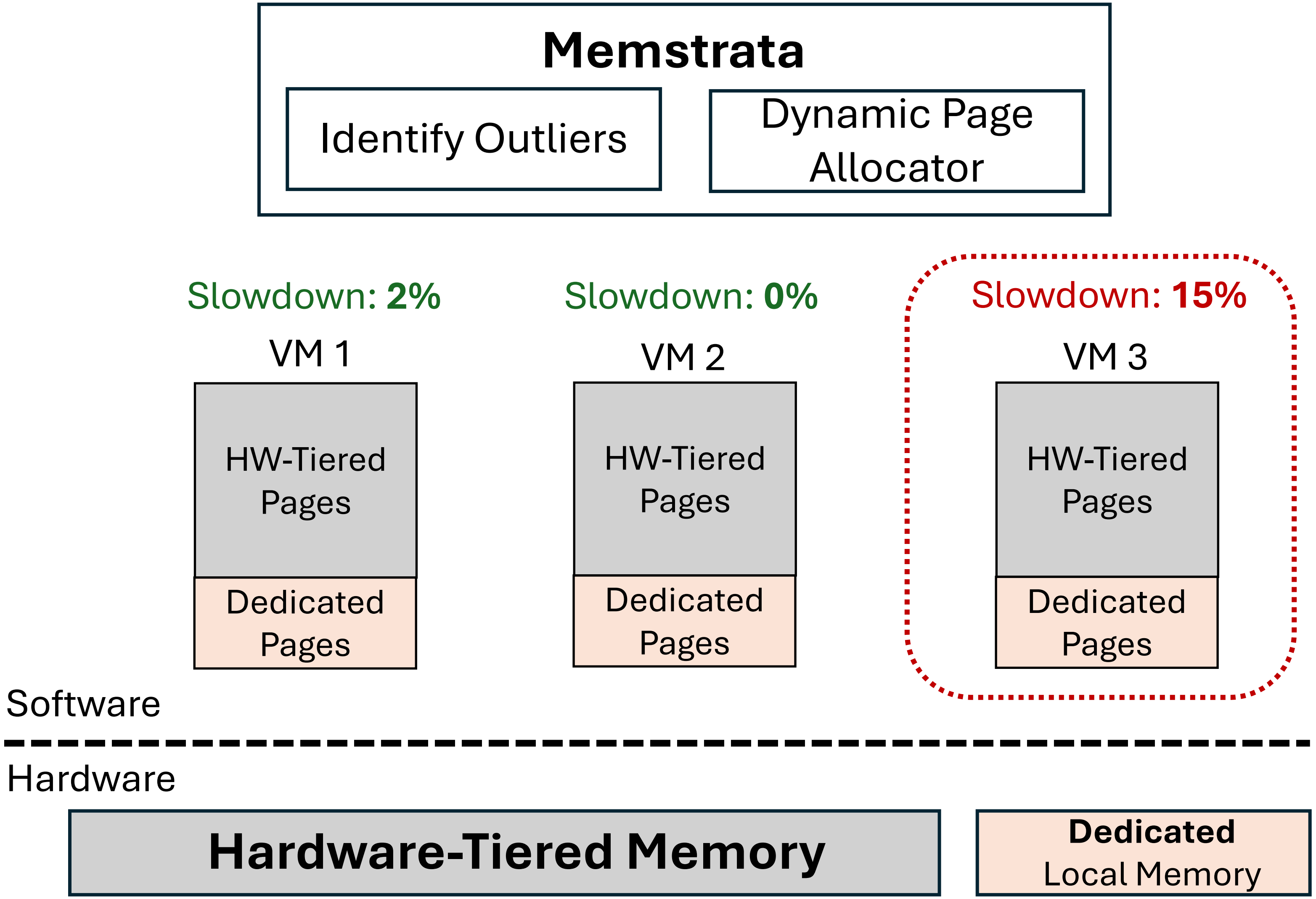

B
N GB
CXL Memory
A
N GB
Local Memory
Dedicated Local Memory
M GB
Data always stay in local memory
64B Line
A
B
Host Physical Memory Address Space
(Flat Memory Mode, 2*N* GB)
Host Physical Address Space
(Dedicated, *M* GB)

Memstrata: Memory Allocator for Hardware Tiering

- A lightweight memory allocator in the hypervisor
- Dynamically allocates dedicated memory to **eliminate outliers**
- Provides **performance isolation** between VMs using page coloring

Memstrata + hardware tiering reduces slowdown from 34% to ~5% across all workloads

Memstrata Dynamically Allocates Dedicated Pages



Identifying Outliers in Hypervisor Is Challenging

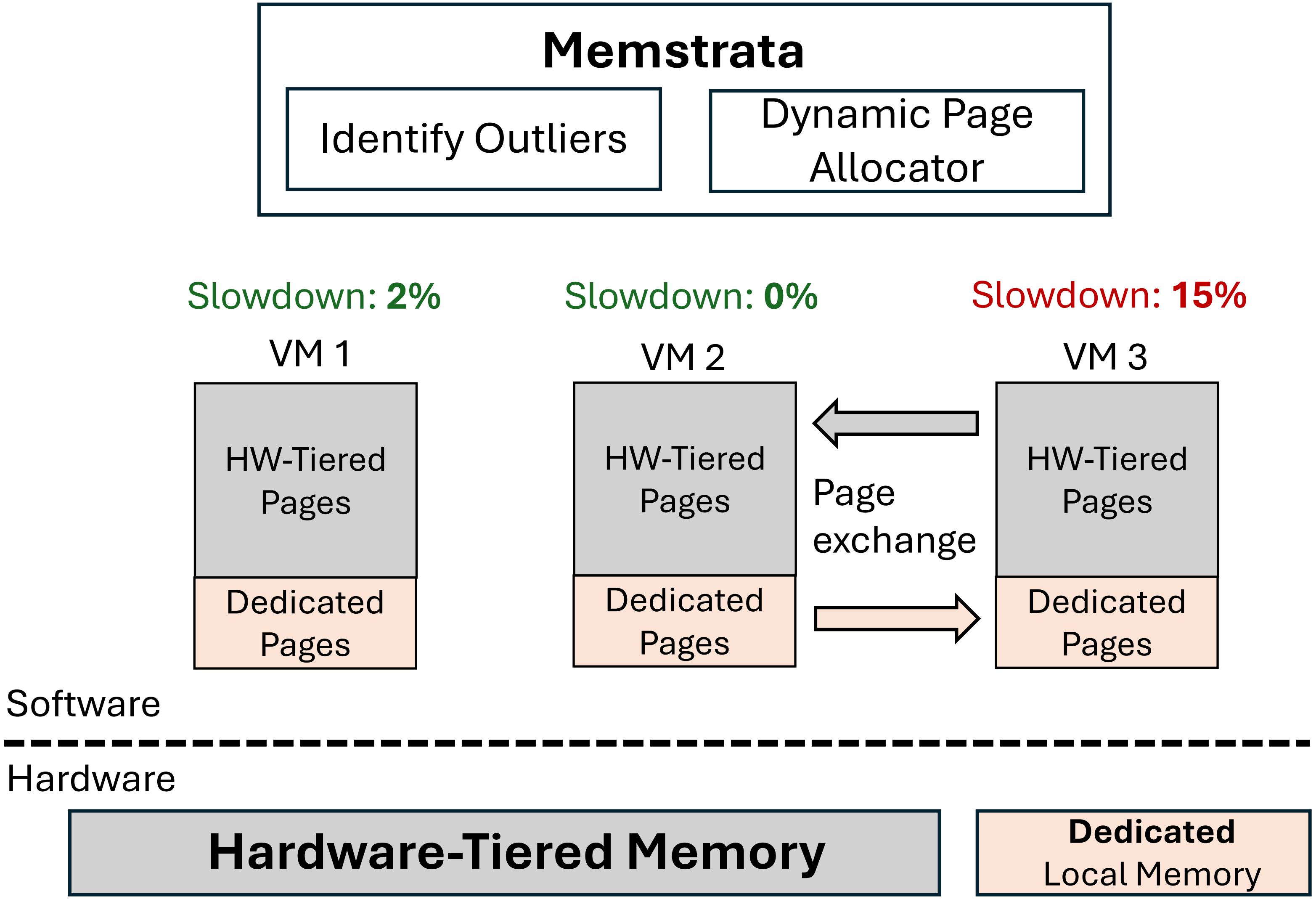
Challenges:

- Hypervisor is unaware of VM workloads
- Hardware tiering only provides system-wide local memory miss rate

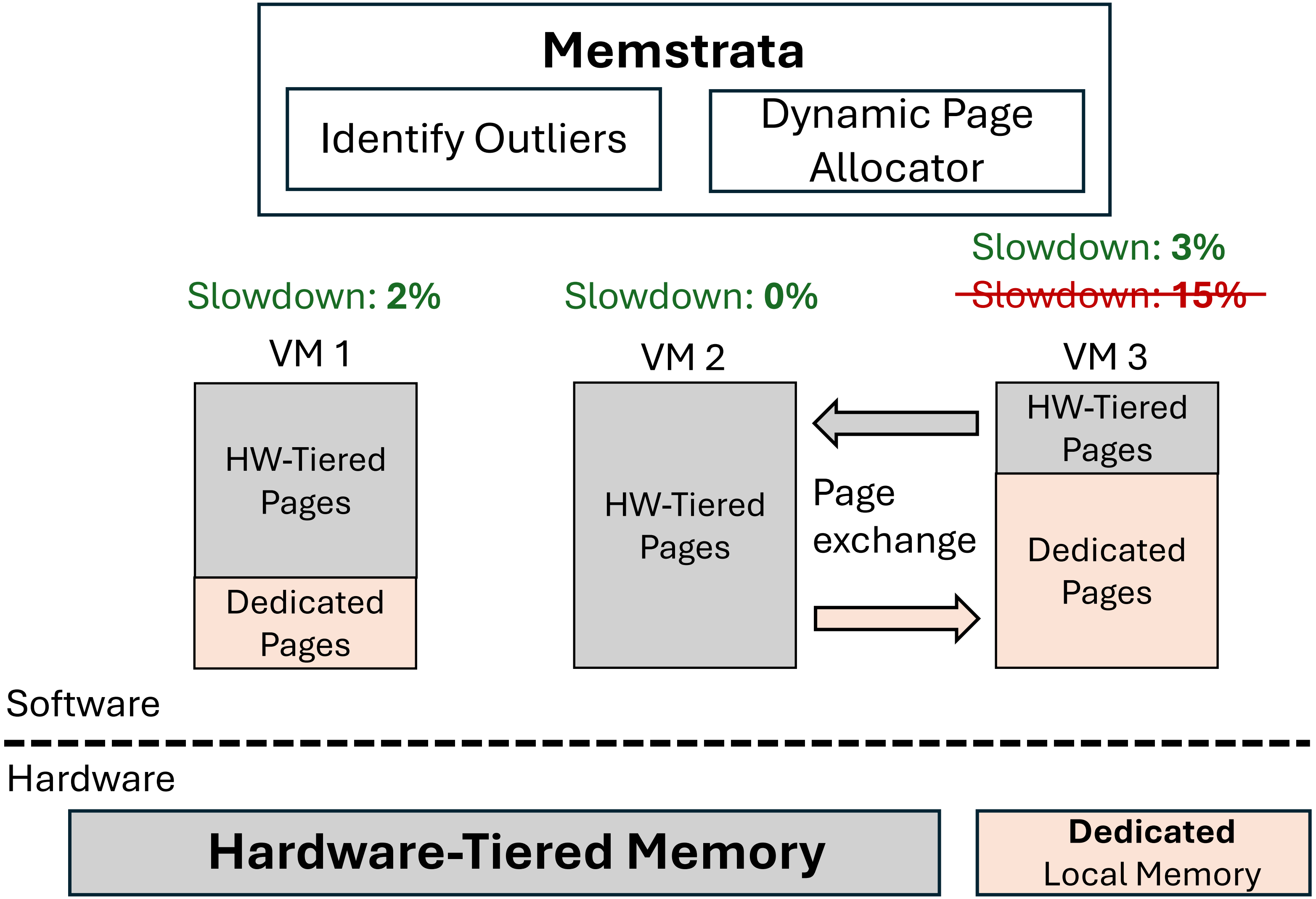
We build a **lightweight prediction model** to identify outliers using low-level performance metrics

- **Per-core** metric: L3 miss latency correlates with miss ratio

Memstrata Dynamically Allocates Dedicated Pages



Memstrata Dynamically Allocates Dedicated Pages



Evaluate 115 Popular Cloud Workloads

Pre-production Intel Xeon 6 CPU with **real CXL cards** from Astera Labs

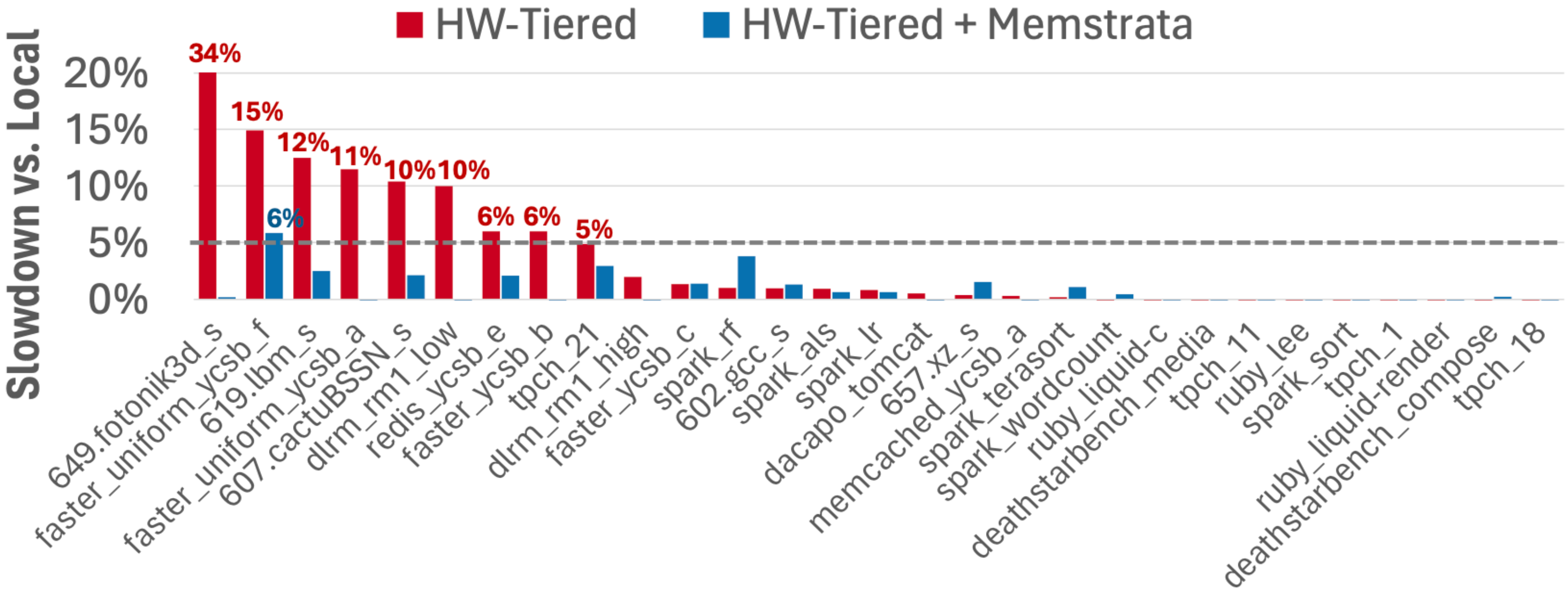
Web Database Machine Learning (ML) Key-Value Store

Spark Graph Processing Scientific Compute

115 workloads in total

Memstrata Eliminates Outliers With Low CPU Overhead

- Sample workloads from representative Azure workload compositions
- Continuous VM arrivals and departures
- Memstrata **mitigates outliers** with **low CPU overhead** (< 3% of a core)



Executive Summary

Source Code

Background:

- CPU **core counts scaling faster** than memory capacity
- CXL enables **second-tier memory** to facilitate core scaling
- But CXL adds latency that hurts performance if not mitigated
- Software tiering helps some but **is not well suited for public clouds**

Contributions:

- Intel Flat Memory Mode: First **hardware-managed** memory tiering for CXL
    - But still has **limitations** that slowdown workloads
- Memstrata: Memory allocator for hardware tiering to **mitigate outliers**
- Slowdown reduces to 5% vs. unattainable one-tier memory

✉ yz@cs.columbia.edu