HeMem:
Scalable Tiered Memory Management for Big Data Applications and Real NVM

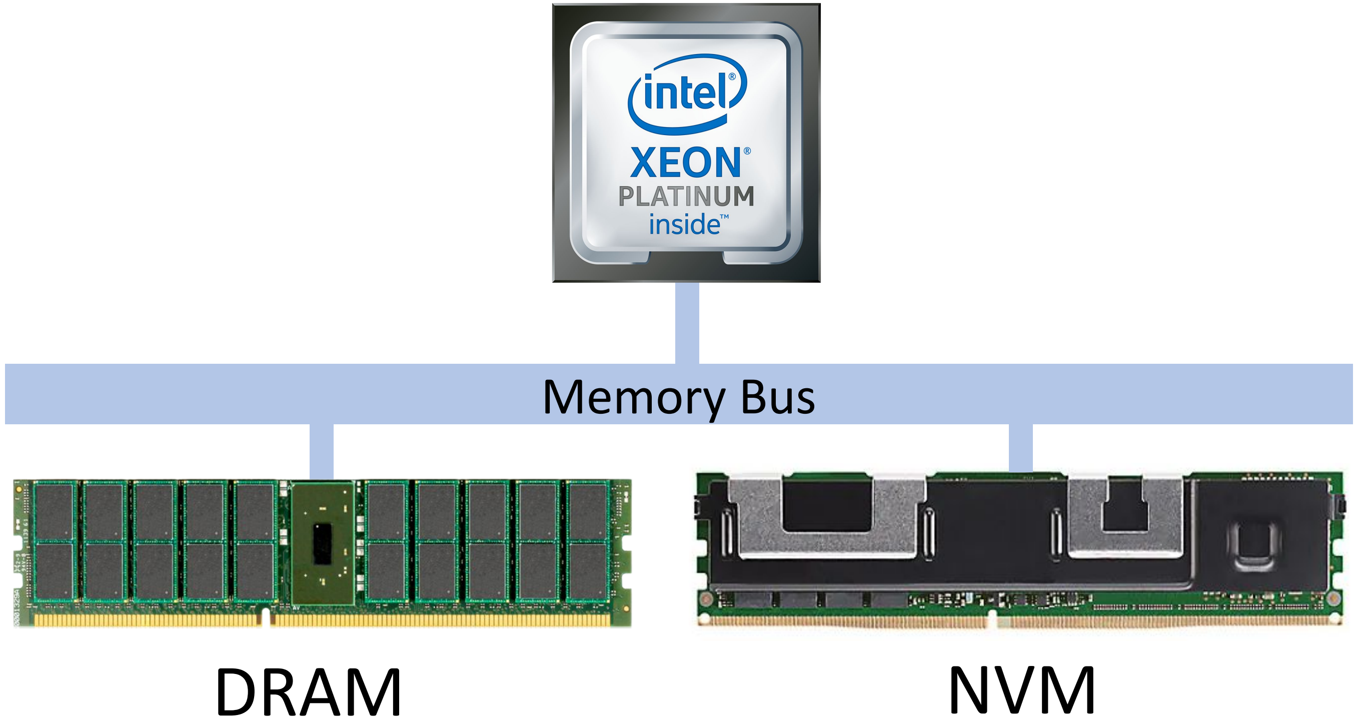
Amanda Raybuck, Tim Stamler, Wei Zhang, Mattan Erez, and Simon Peter

TEXAS
The University of Texas at Austin

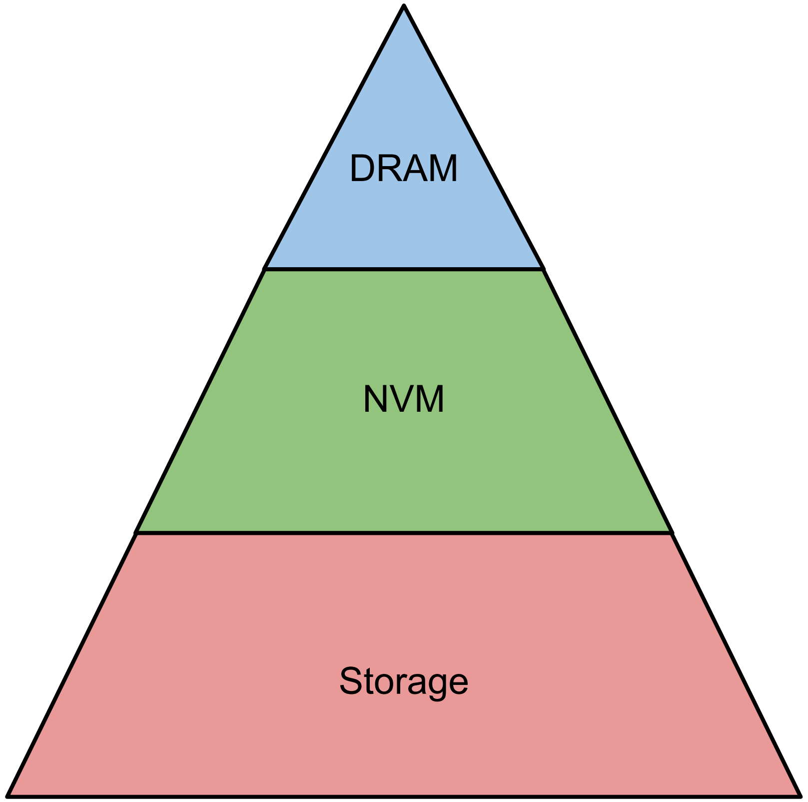
Microsoft

W
UNIVERSITY of WASHINGTON

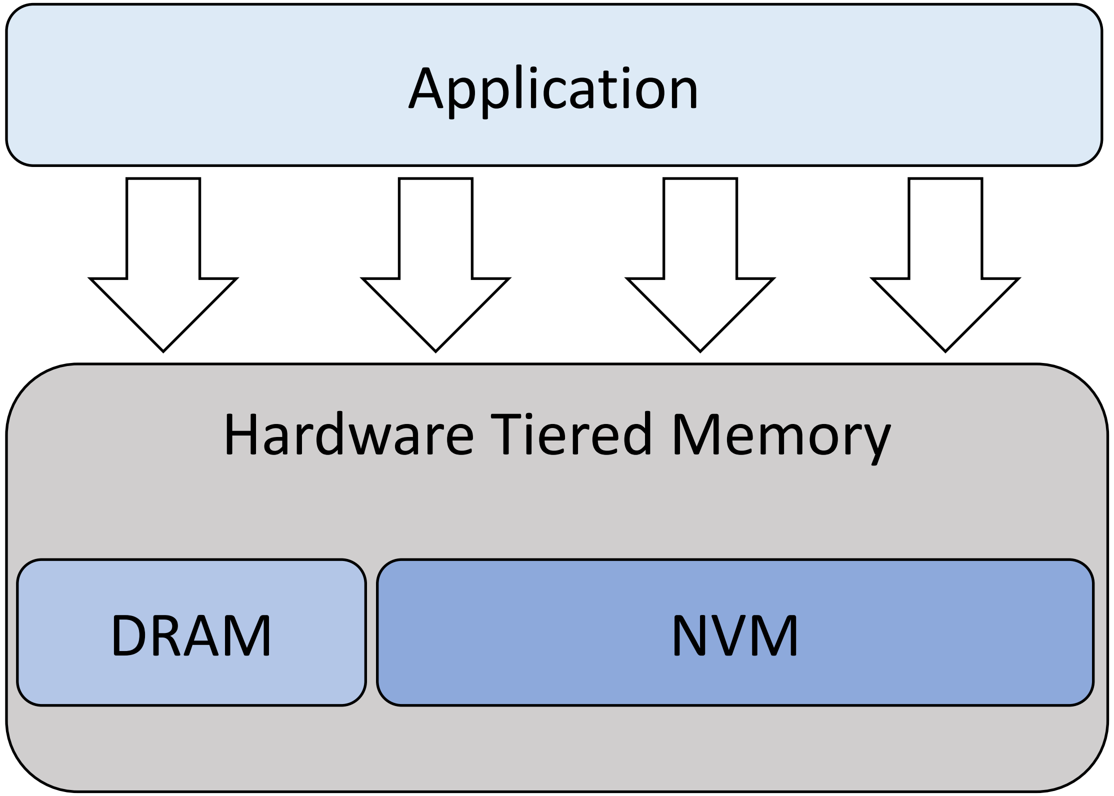
DRAM + NVM tiered memory



- 8x capacity
- 2x latency
- Asymmetric read/write bandwidth
- High overhead for small accesses



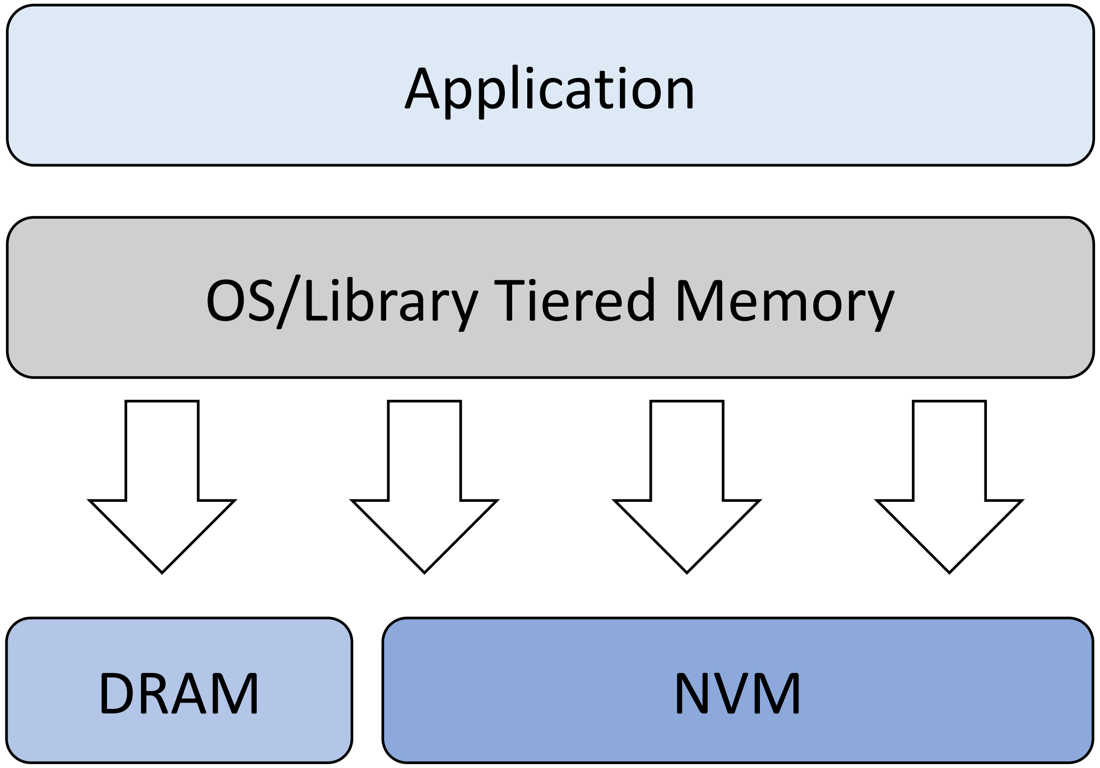
Hardware tiered memory



Example: Intel Optane Memory Mode

- ✓ No OS support needed
- ✓ Low overhead
- ✗ No visibility into apps
- ✗ Limited to simple management techniques

Existing software tiered memory



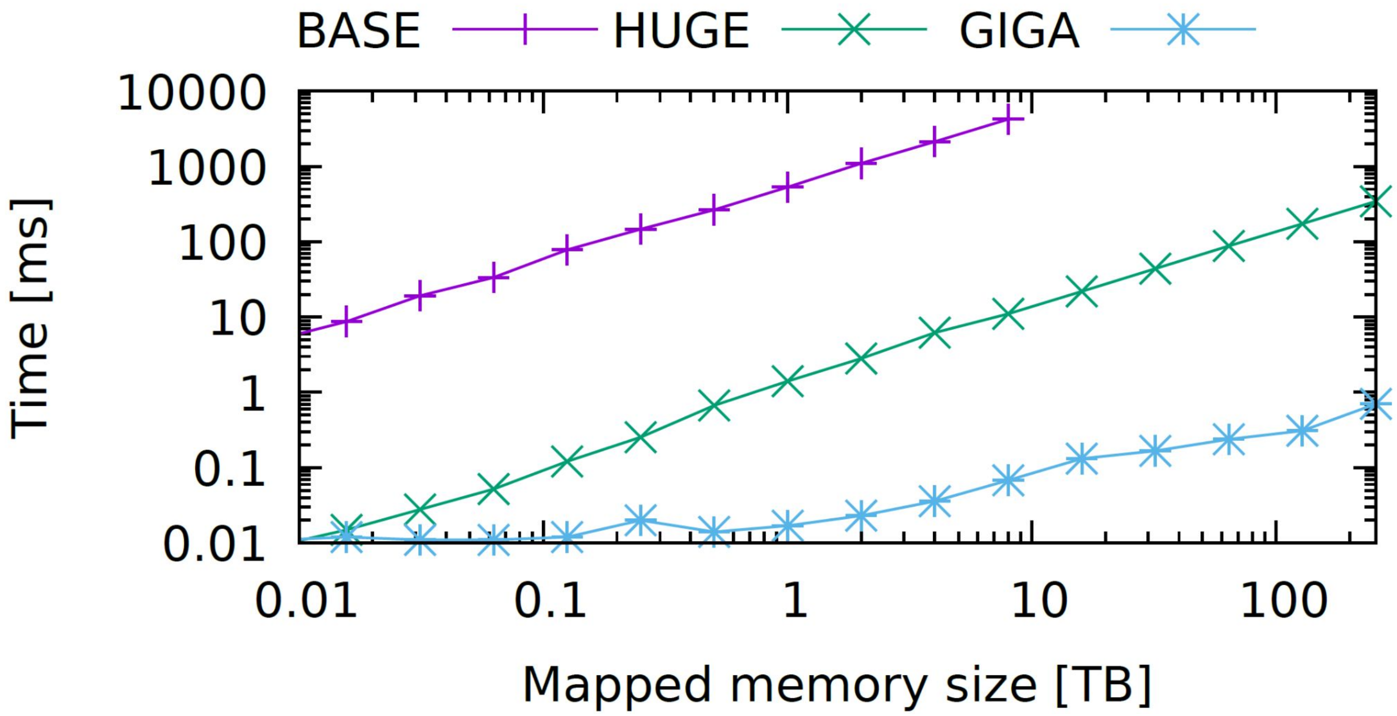
Examples: HeteroOS [ISCA ‘17], Nimble Page Management [ASPLOS ‘19]

- ✓ Insights into applications
- ✓ Supports complex policies

Evaluated only on emulated NVM:

- ✗ Does not scale to NVM capacity
- ✗ No support for asymmetric read/write bandwidth
- ✗ Limited flexibility

Why not access/dirty bits?



- Not scalable
- Takes seconds to scan large memories with base pages
- Overhead of TLB shootdowns to clear bits

HeMem:

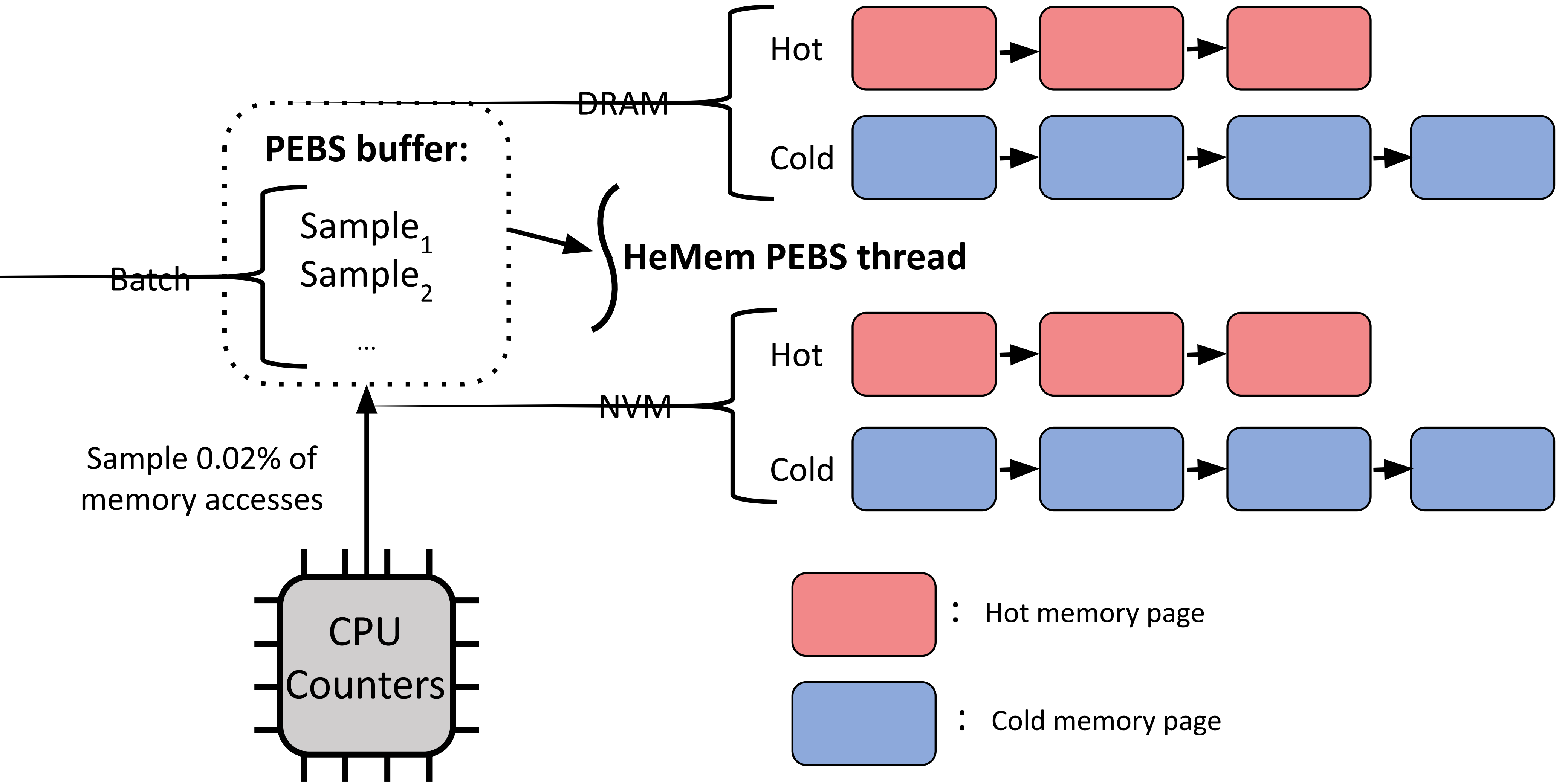
Scalable software tiered memory management system designed for real NVM

- **Design principles:**
  - Asynchronous memory access sampling with CPU performance counters
  - Asynchronous memory migration with DMA offload
  - Focus on asymmetric NVM bandwidth
  - Data scalability awareness
  - Flexibility

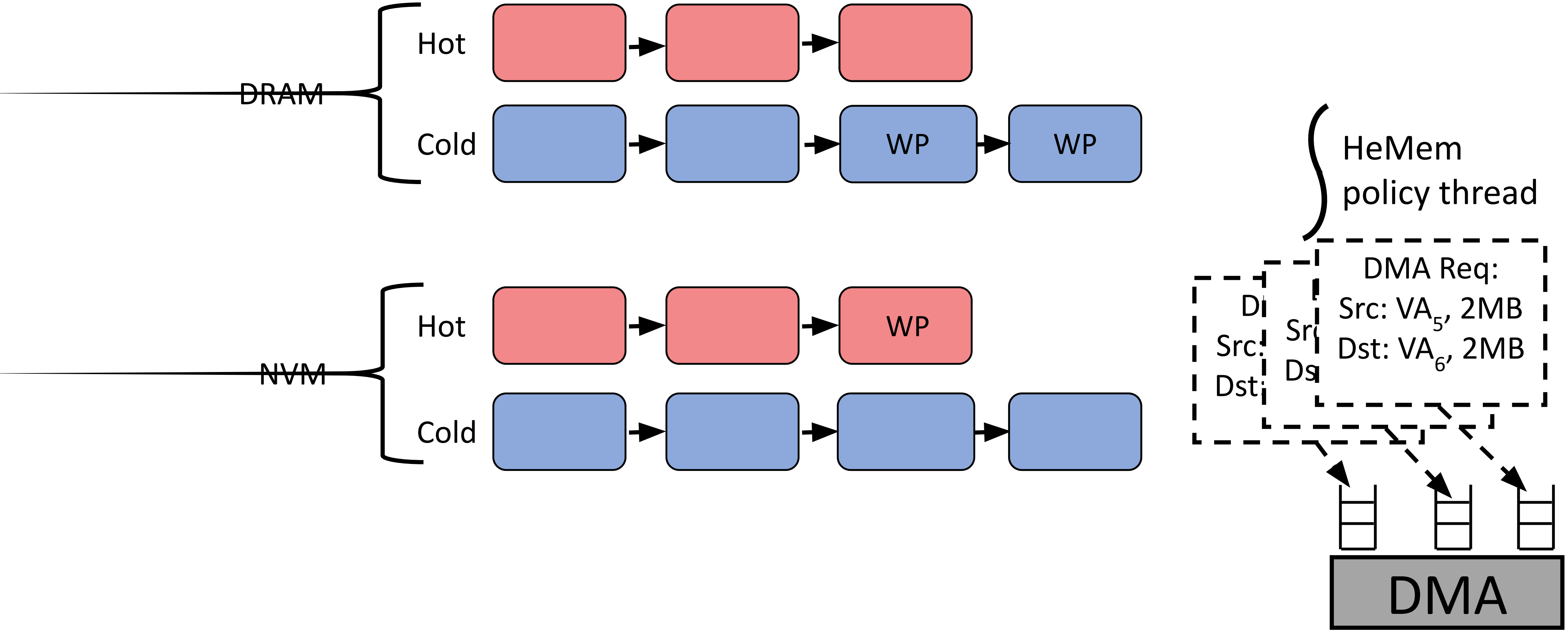
PEBS memory access sampling

- PEBS: processor event-based sampling
  - Supported in modern Intel processors
- Processor records samples of load/store virtual memory address
  - Records are stored in a memory buffer
- We measure DRAM loads, NVM loads, and all stores
  - Instead of using page table access/dirty bits
- Sampling 0.02% of all memory accesses provides sufficient fidelity

Asynchronous hot/cold classification

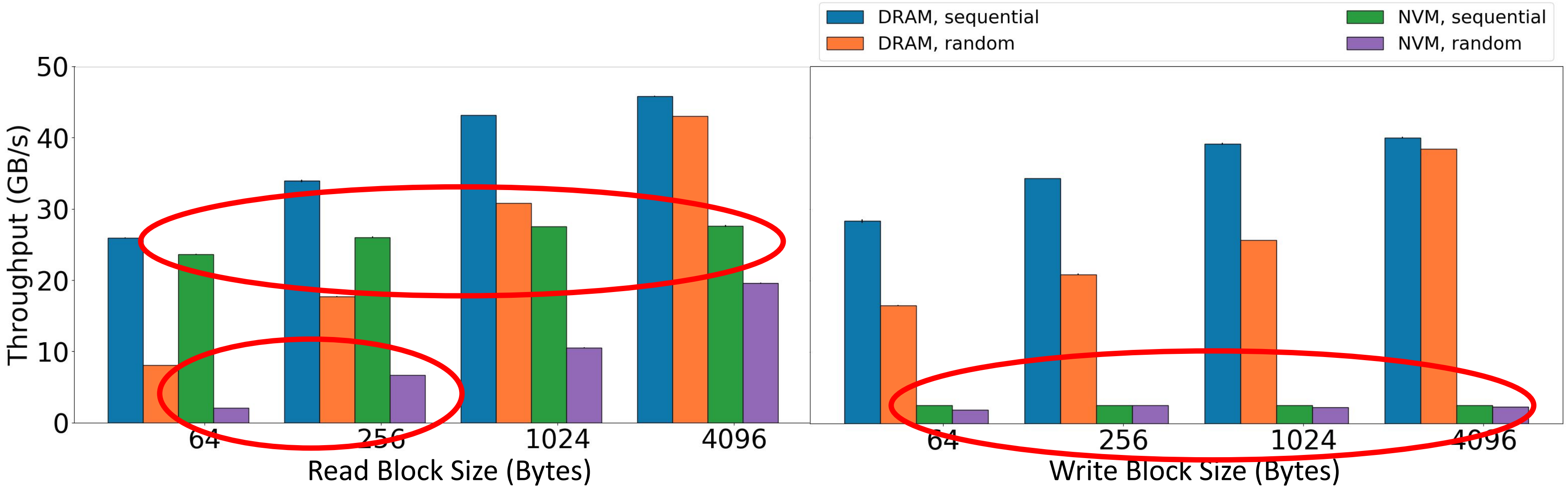


Asynchronous memory migration



Optimize for real NVM

- Keep small objects in DRAM
  - Avoid the small random reads from NVM that suffer overheads
  - Small, ephemeral objects remain in DRAM
- Limit writes to NVM to avoid write bandwidth bottleneck
  - Migrate and keep frequently written pages to DRAM



Data Scalability Awareness

- Tracking hot/cold memory is expensive with lots of memory
- Only manage objects that are long-lived and likely to grow
  - Allow Linux to handle everything else (program text, kernel objects...)
- Smaller objects are more likely to be short-lived and can be in DRAM

Flexible user space mechanisms

- HeMem is implemented as a user-level library
  - Can be modified to better suit applications
  - Can more closely integrate with managed runtimes to further optimize
    - Garbage collection
  - Userfaultfd for handling of page and write-protection faults
- Intercepts memory allocation calls to learn size of objects
- Works with unmodified applications

Implementation

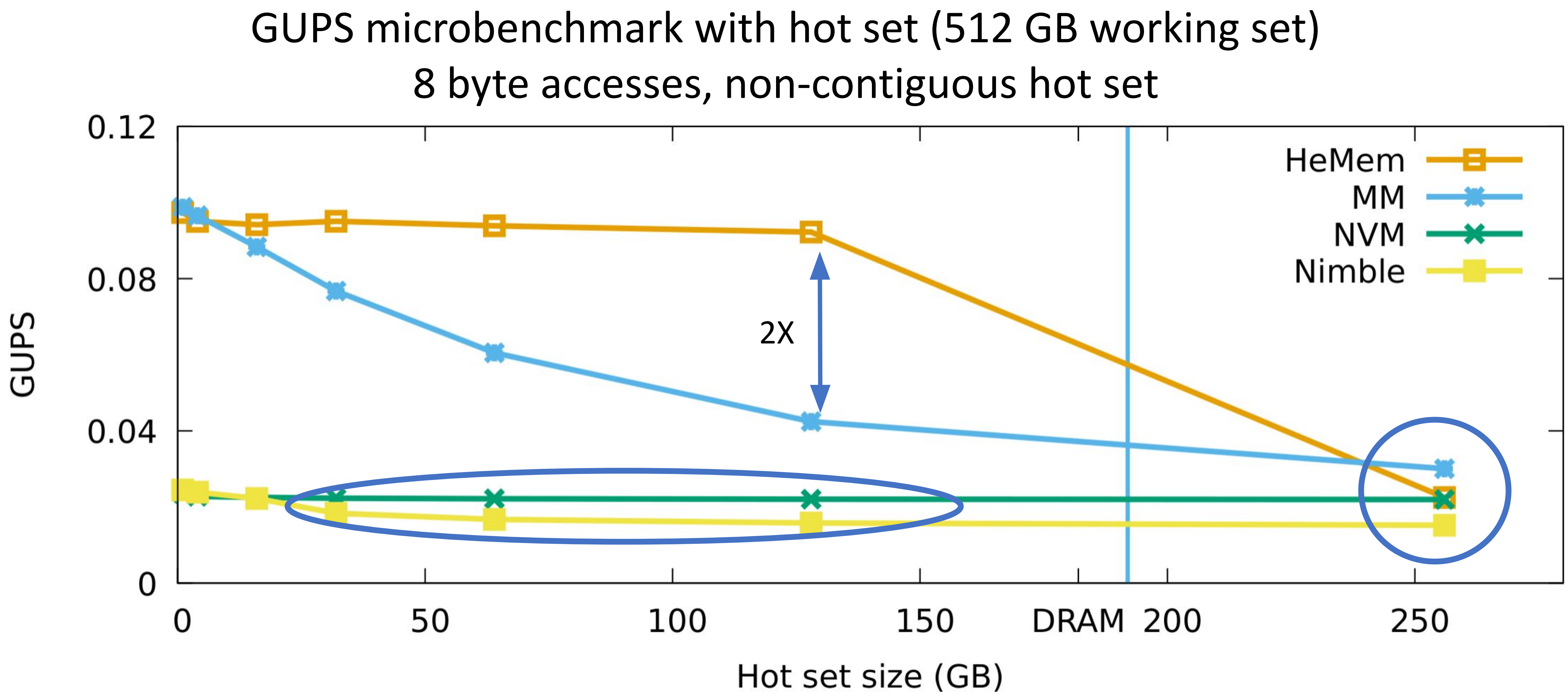
- HeMem library implemented with ~4100 lines of C code
- Relies on a custom Linux kernel with support for /dev/dax
  - Added ~1300 lines to linux kernel
- Both DRAM and NVM exposed as /dev/dax files
  - DRAM /dev/dax reserved at startup with `memmap` command line argument
- Uses PEBS via the Linux perf interface
  - `MEM_LOAD_L3_MISS_RETIRED.LOCAL_DRAM` for DAM loads
  - `MEM_LOAD_RETIRED.LOCAL_PMM` for NVM loads
  - `MEM_INST_RETIRED.ALL_STORES` for stores

Evaluation

Evaluation setup

- Cascade Lake-SP w/ 24 cores, 192 GB DRAM, 768 GB NVM
    - All DIMMs populated, leveraging all 6 memory channels
- Comparisons:
    - Intel Memory Mode
    - Linux nimble tiered memory management [ASPLOS ‘19]

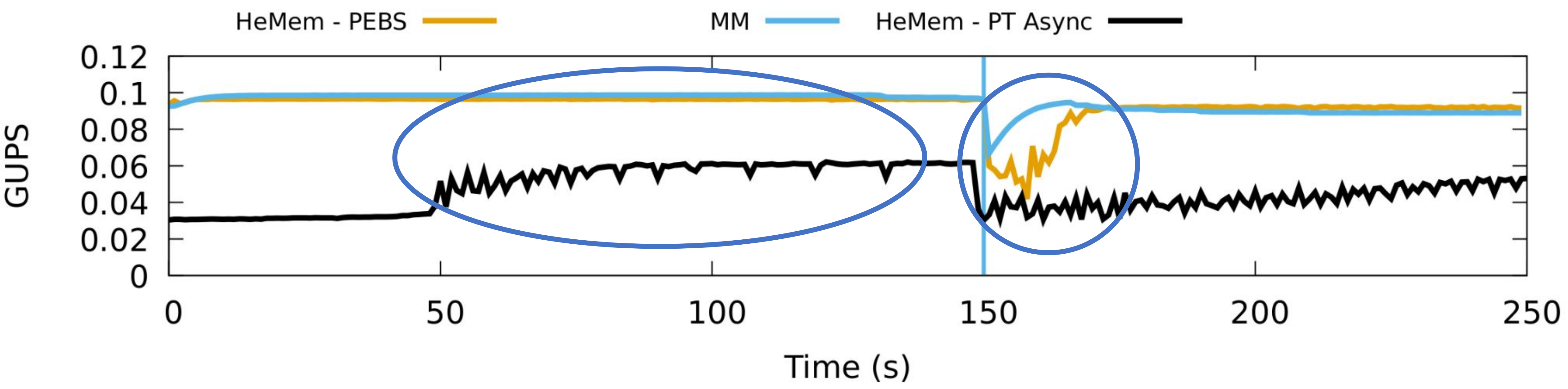
Hot set identification



Dynamic hot set identification

GUPS with a 512 GB working set and a 16 GB hot set

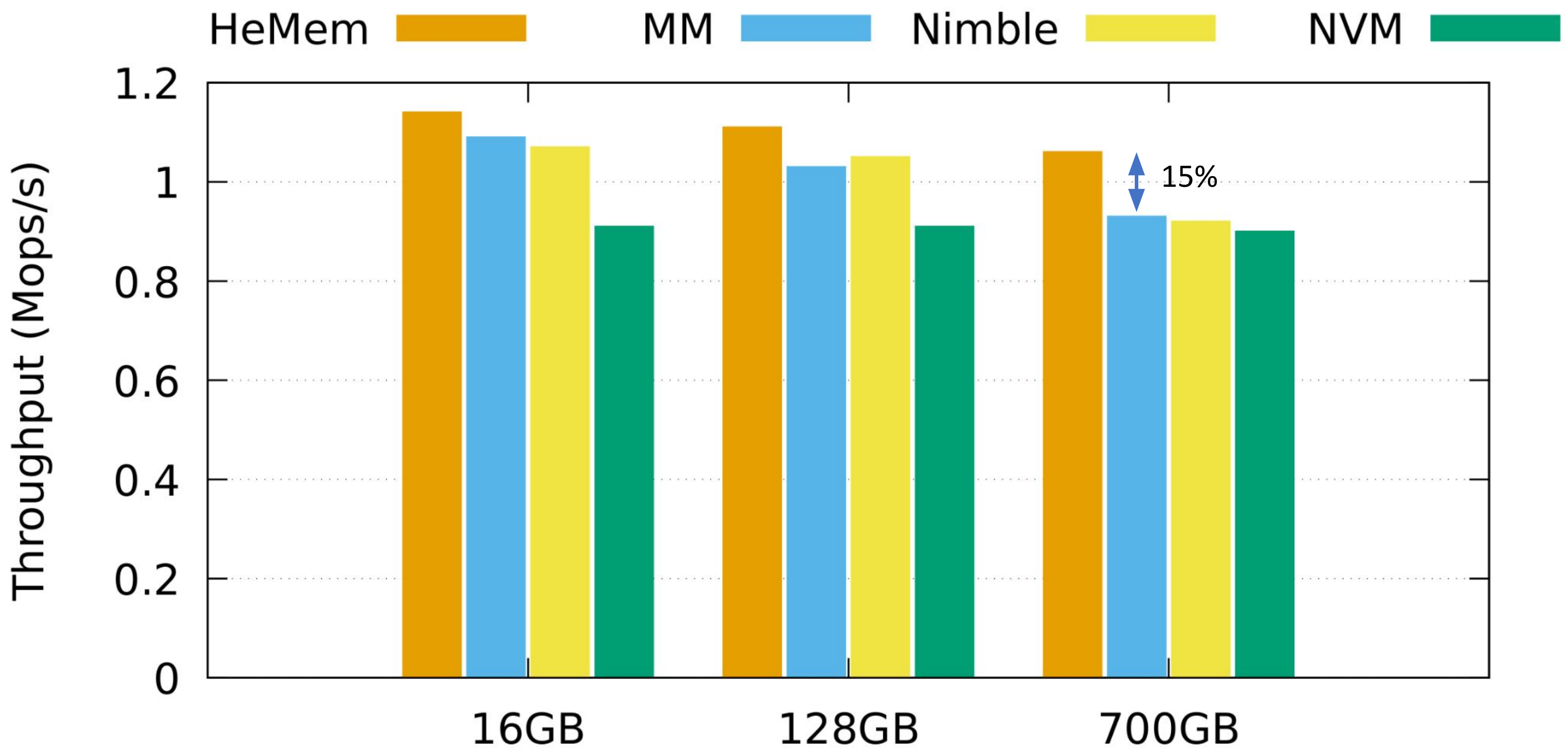
At time t=150, shift hot set over by 4 GB



FlexKVS key-value store throughput

4KB value size, 90% GET, 10% SET,

20% hot keys accessed 90% of time



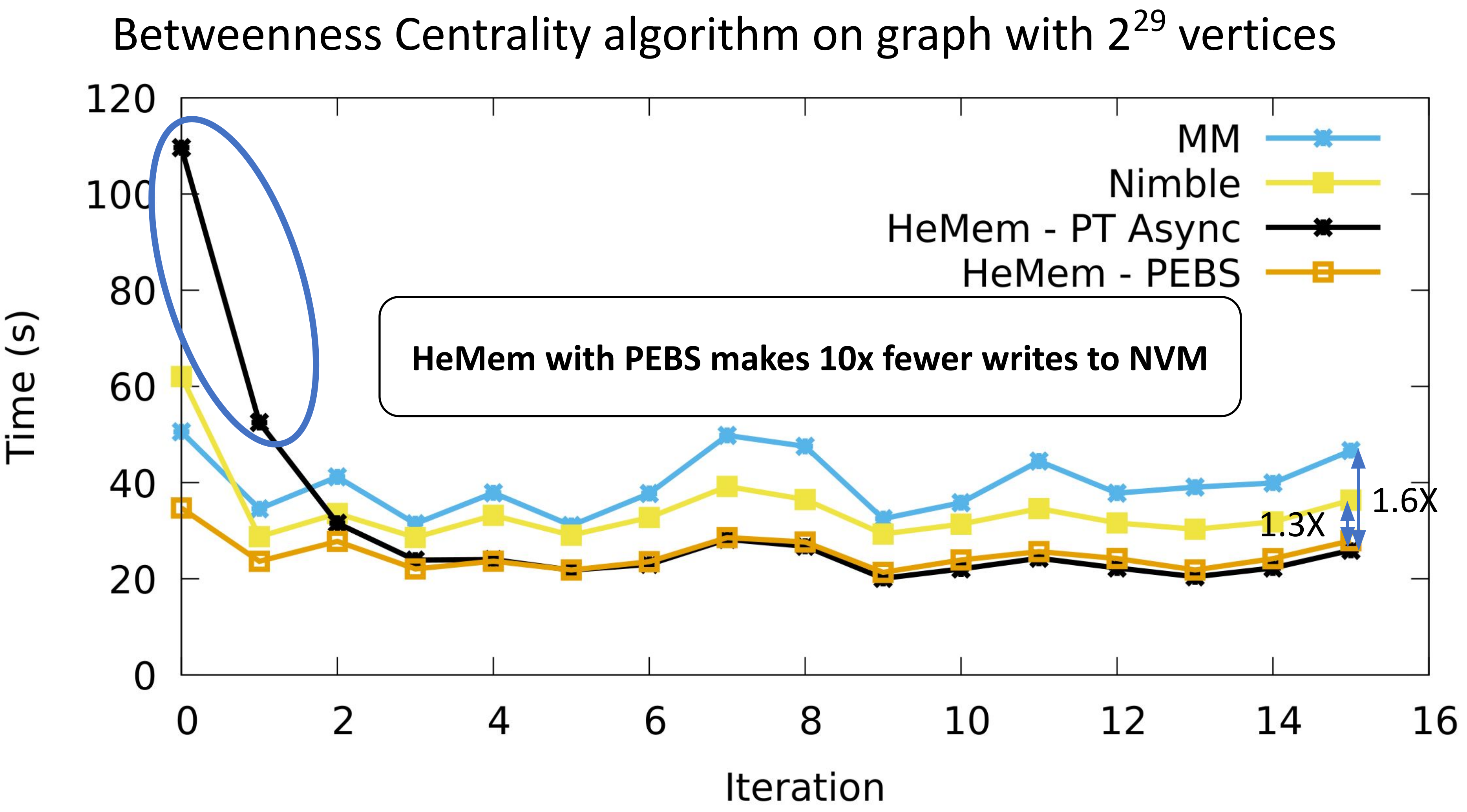
FlexKVS key-value store priority latency

1 prioritized server with 16GB working set

1 non-prioritized server with 500GB working set

	Priority			Regular		
μs	**50p**	**99p**	**99.9p**	**50p**	**99p**	**99.9p**
HeMem	86	239	341	146	318	409
MM	127	278	342	156	310	380
%	47	16	0	6	-2	-8

GAPbs execution time



Summary

- Tiered memory systems need to support real NVM
  - Need to scale to large capacities
  - Need to support unique NVM performance features

- **HeMem:** redesign of tiered memory management with real NVM
  - Sampling-based memory access monitoring without page tables
  - Asynchronous memory migration in batches with DMA offload
  - Accurately distinguishes hot from cold memory

- Up to 1.6x GAPbs speedup, 2x GUPS, 10x fewer NVM writes

Source code: https://bitbucket.org/ajaustin/hemem/src/sosp-submission/