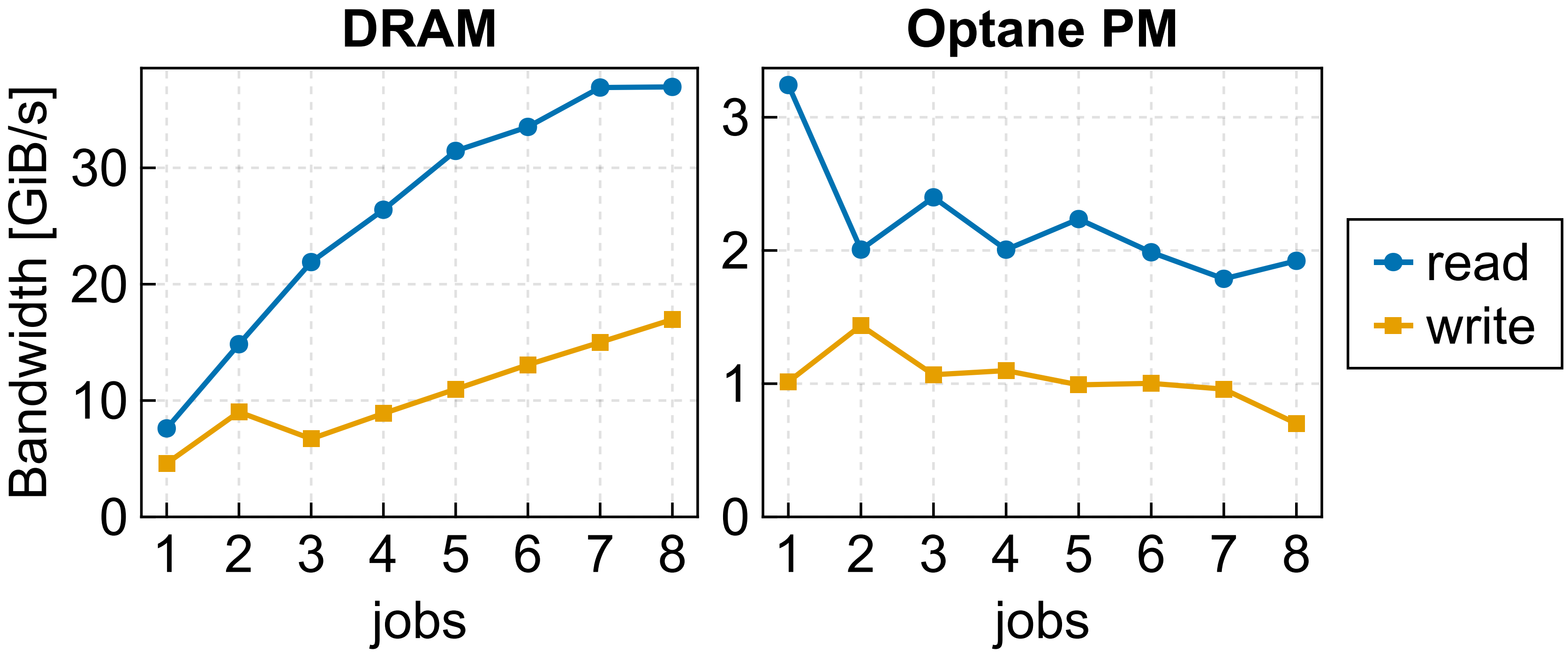
Karlsruher Institut für Technologie

Per-Process Memory Bandwidth Management for Heterogeneous Memory Systems

Lukas Werling, Daniel Habicht, Frank Bellosa

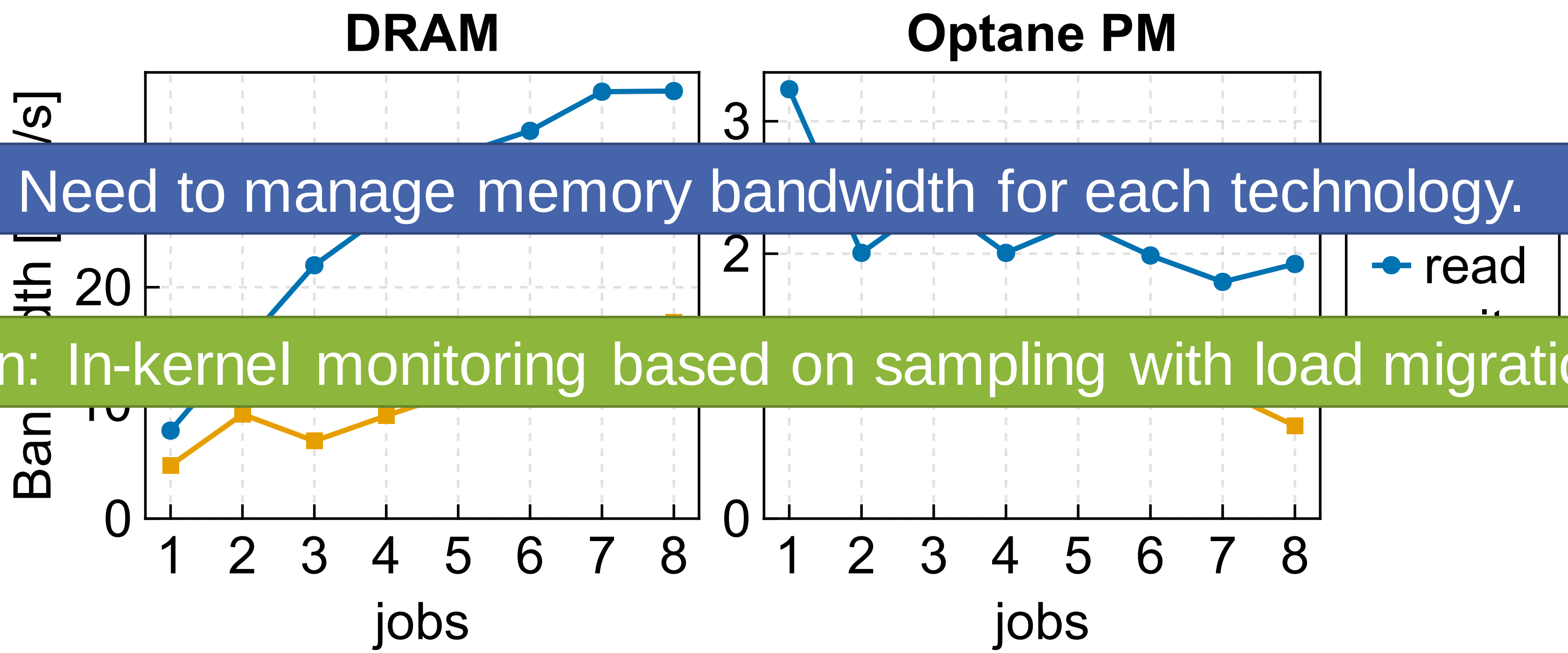
Motivation

- DAX with PM and CXL: Direct mappings not just to DRAM
  - Very different behavior
  - PM: parallel writes hurt overall performance



Motivation

- DAX with PM and CXL: Direct mappings not just to DRAM
  - Very different behavior
  - PM: parallel writes hurt overall performance



Need to manage memory bandwidth for each technology.

Our solution: In-kernel monitoring based on sampling with load migration policies.

Memory Bandwidth Monitoring Goals

process association

arbitrary devices

low latency

low overhead

high accuracy

Challenge: Insufficient HW Control

process association

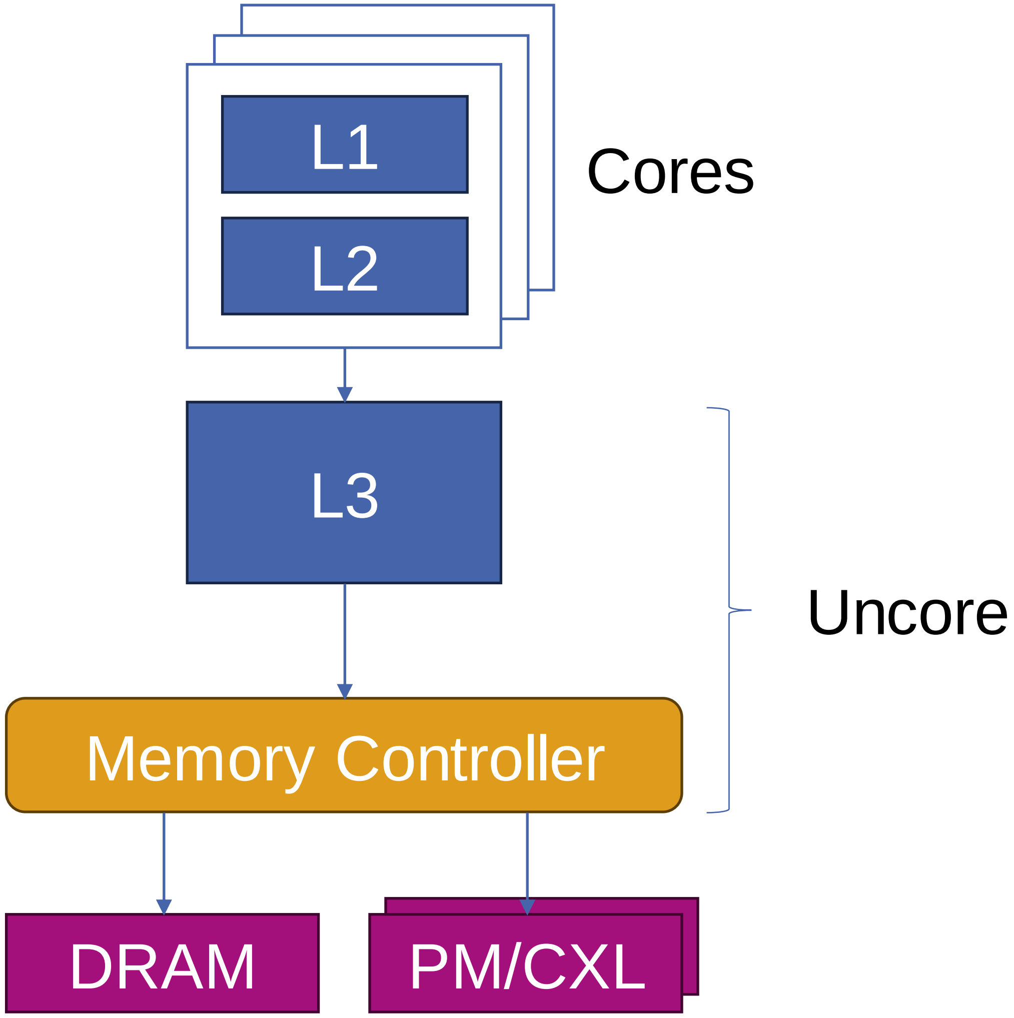
arbitrary devices

low latency

low overhead

high accuracy

- Performance counters
  - Missing store events (core)
  - Limited process association (uncore)
  - Fixed memory classes (DRAM, PM)
- Intel Memory Bandwidth Monitoring
  - Only for overall bandwidth



Challenge: Insufficient HW Control

process association

arbitrary devices

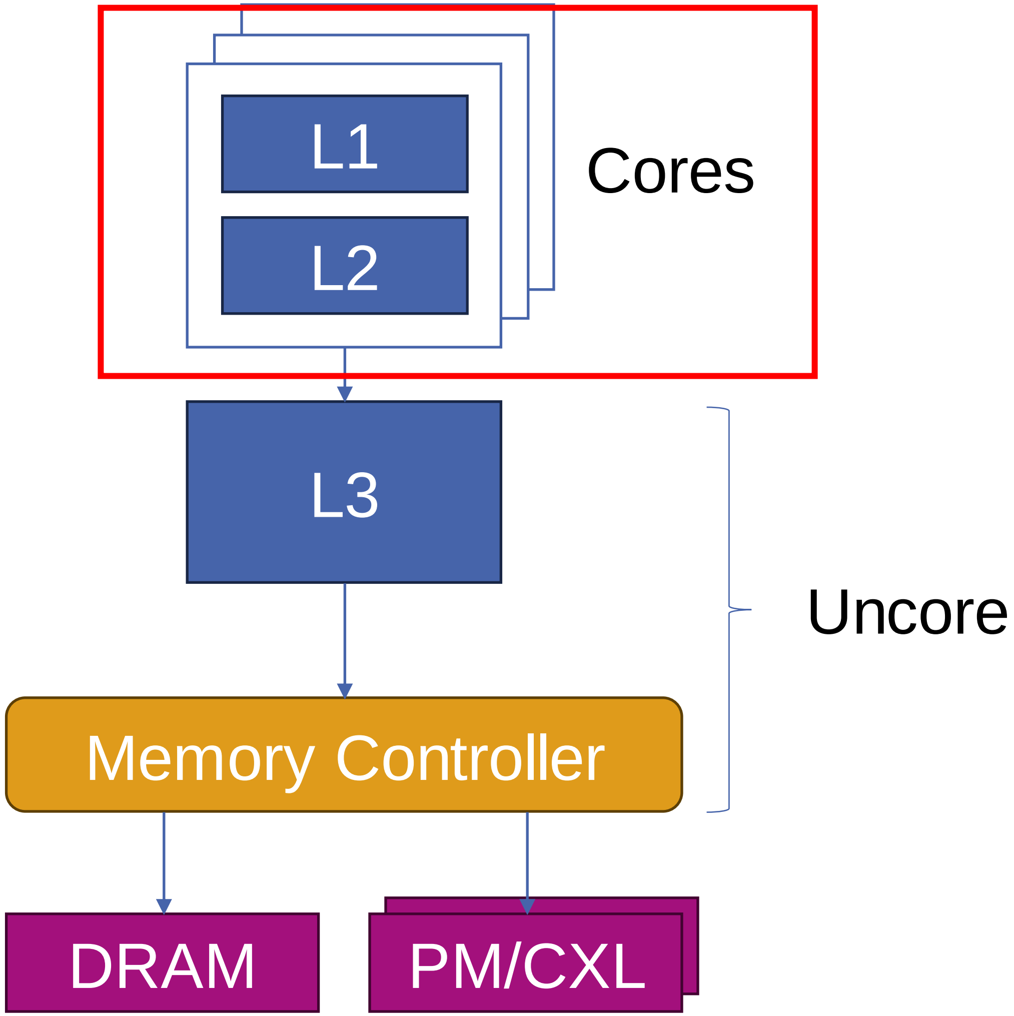
low latency

low overhead

high accuracy

- Performance counters
  - Missing store events (core)
  - Limited process association (uncore)
  - Fixed memory classes (DRAM, PM)

- Intel Memory Bandwidth Monitoring
  - Only for overall bandwidth



Challenge: Insufficient HW Control

process association

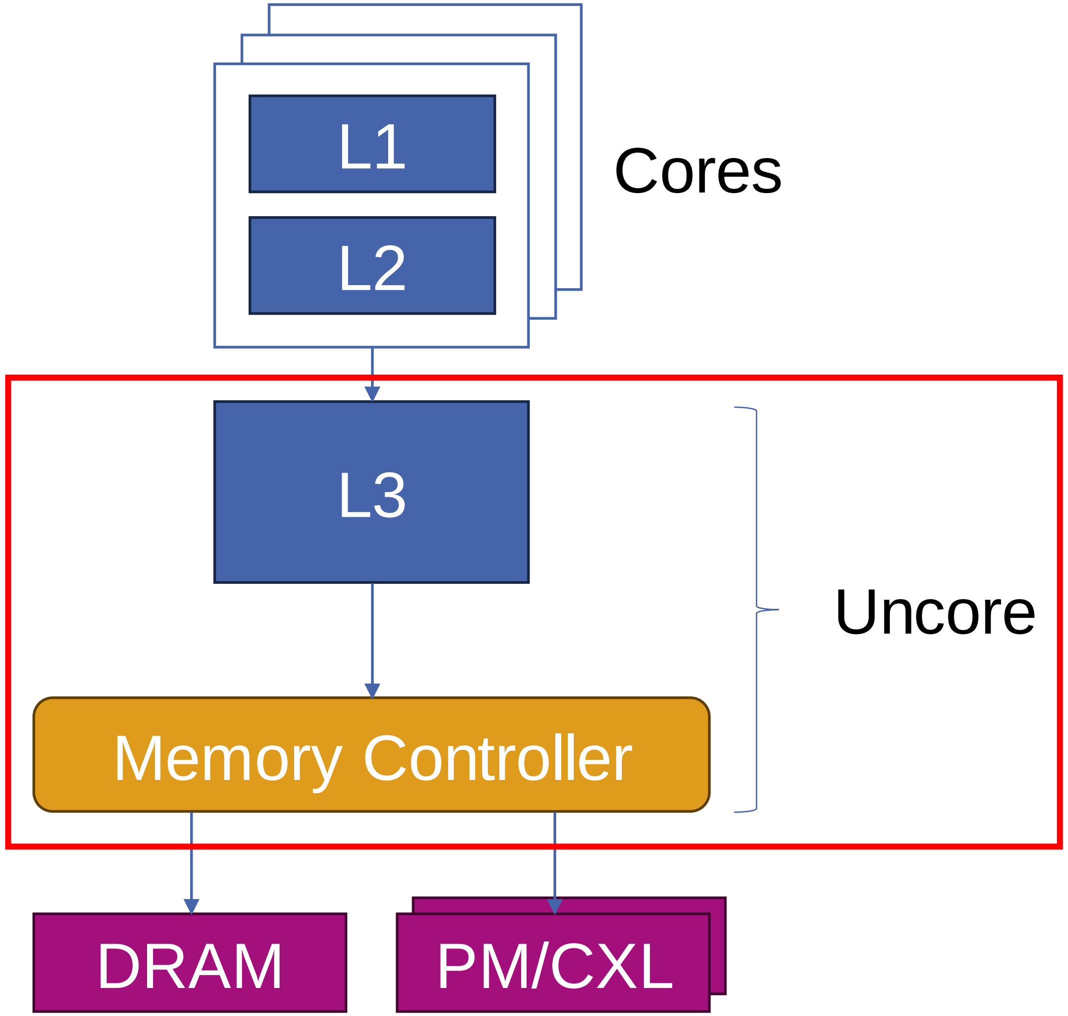
arbitrary devices

low latency

low overhead

high accuracy

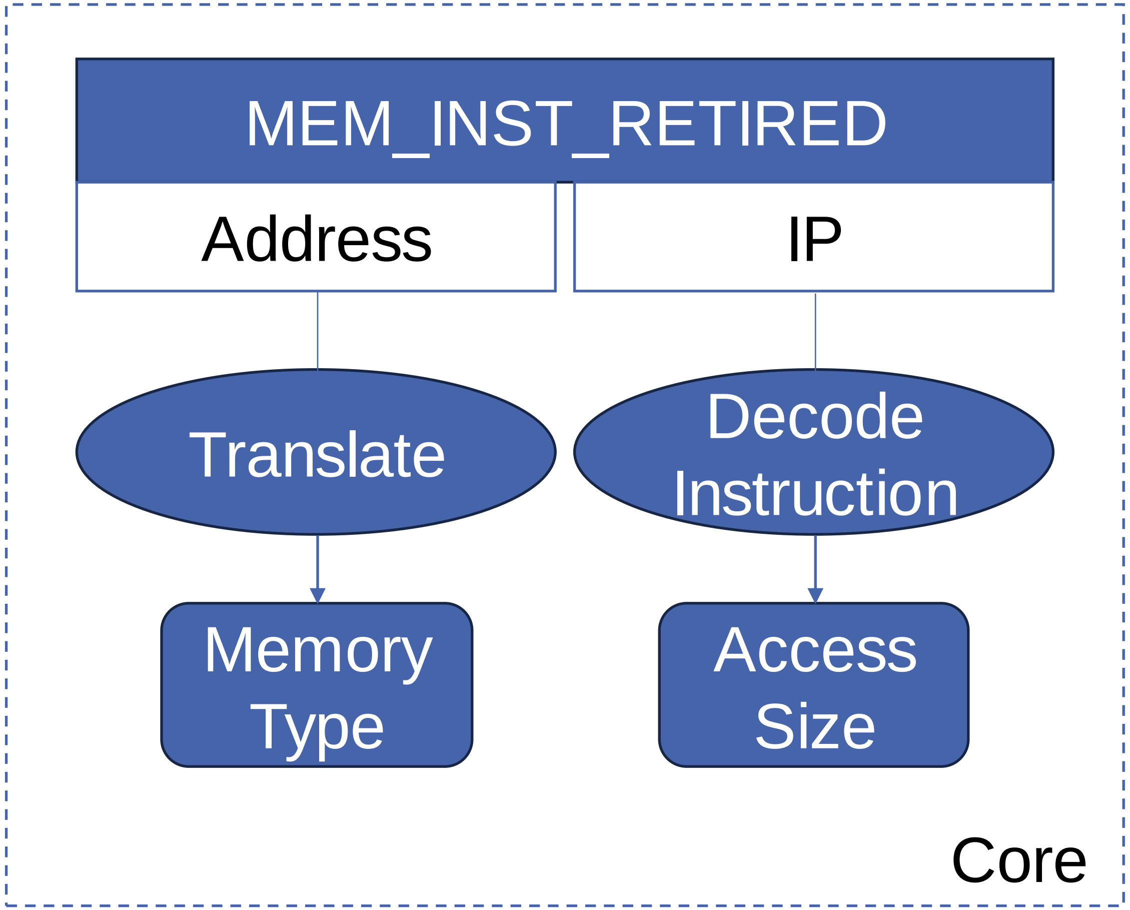
- Performance counters
  - Missing store events (core)
  - Limited process association (uncore)
  - Fixed memory classes (DRAM, PM)
- Intel Memory Bandwidth Monitoring
  - Only for overall bandwidth



Approach: Sampling Stores

- Stores: Intel PEBS
  - Per-core counters
  - Capture additional information every Nth instruction
  - Process sampling buffer regularly (1-6 samples)

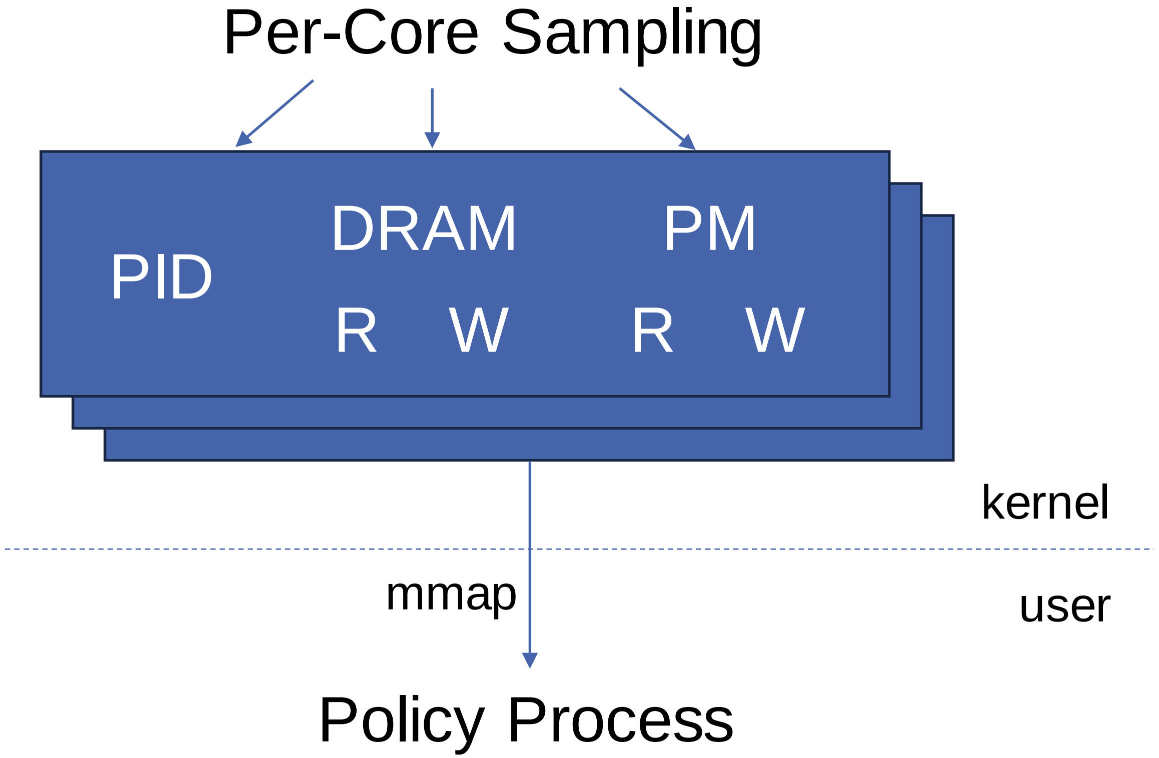
- Loads: normal L3 miss counters
  - Distinguish DRAM/PM



```
writes[type] += access_size * sampling_interval
```

Approach: Interface for Policies

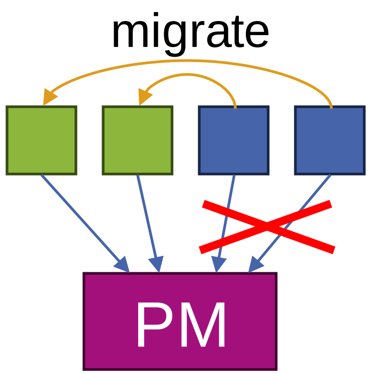
- Challenge: Access current data at low latency
- Solution: Shared buffers
  - Data for running process
  - Lock-free access
  - Enables user-space policies
- Additional access via /proc



Work in Progress: Policies

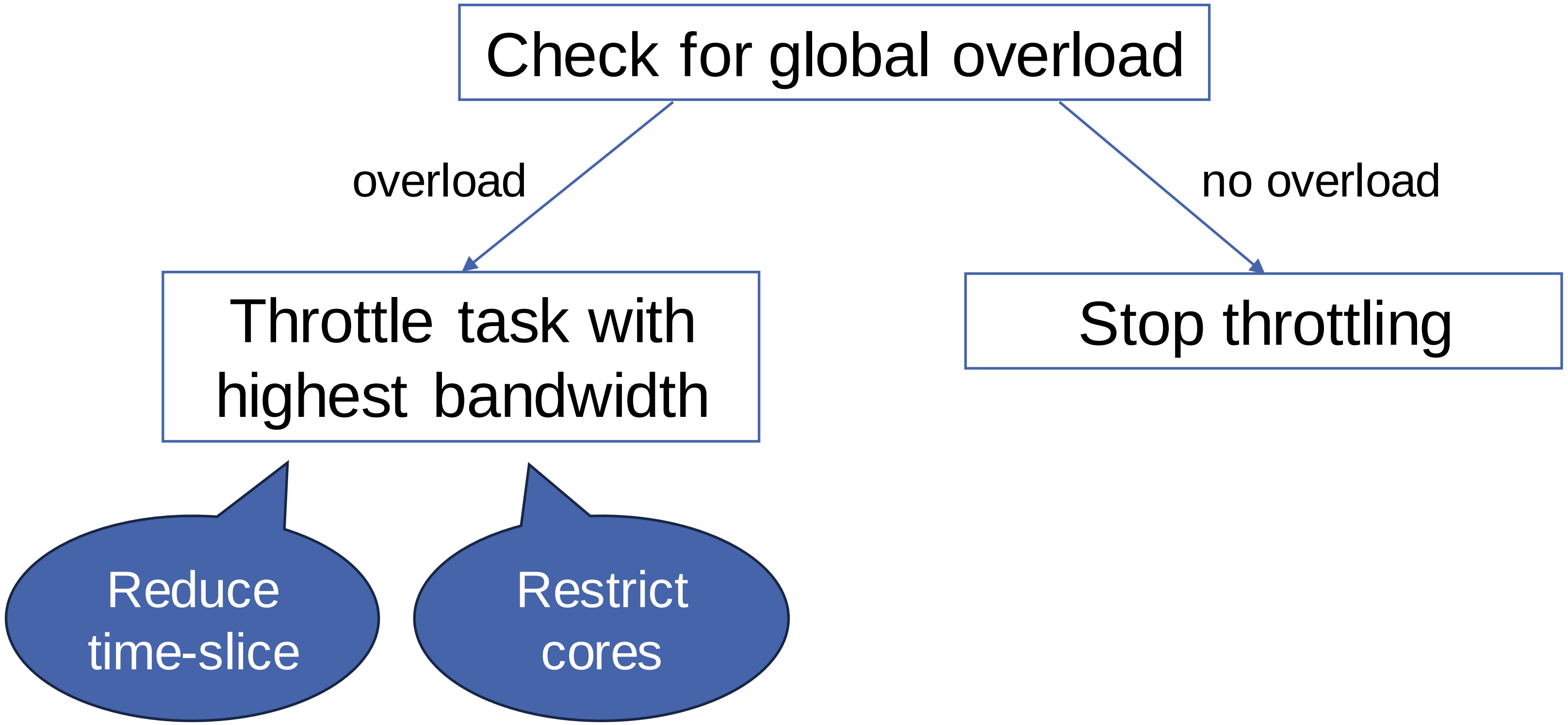
Avoid PM overload

Core Specialization on overload



Fair bandwidth allocation

High priority / best effort



Check for global overload

overload

no overload

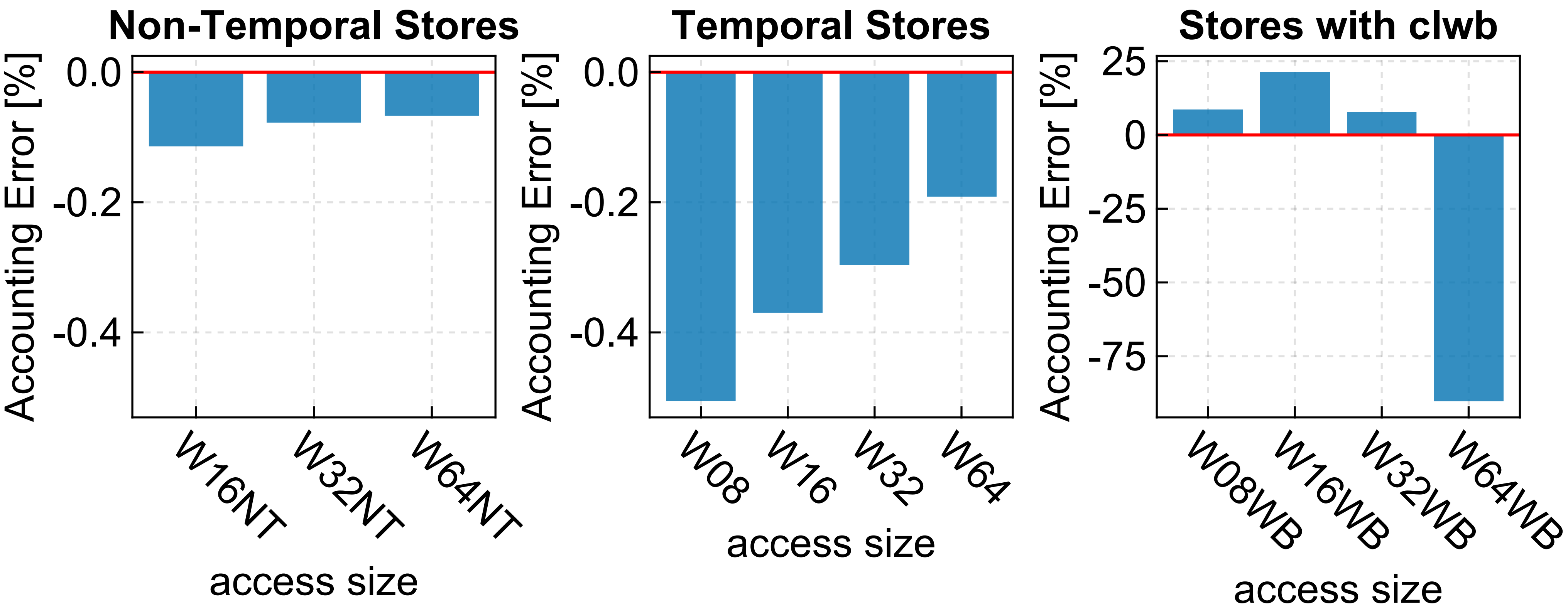
Throttle task with highest bandwidth

Stop throttling

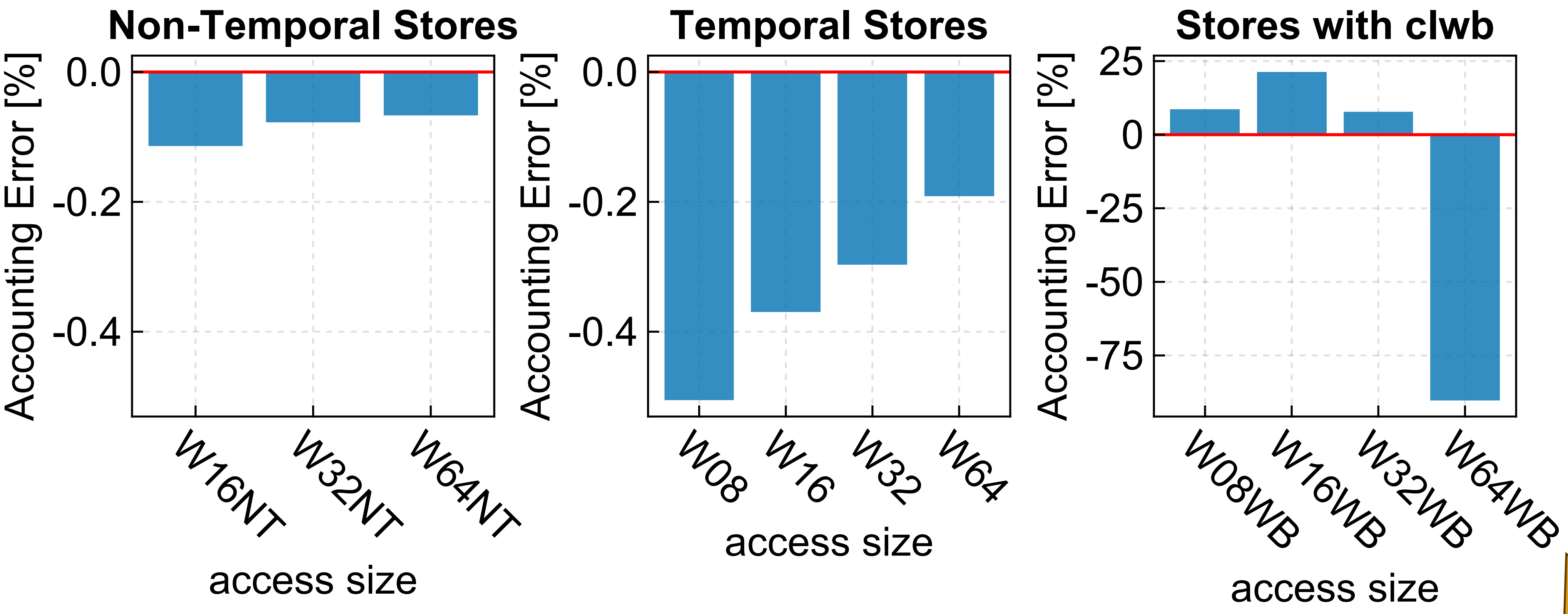
Reduce time-slice

Restrict cores

Evaluation: Accuracy



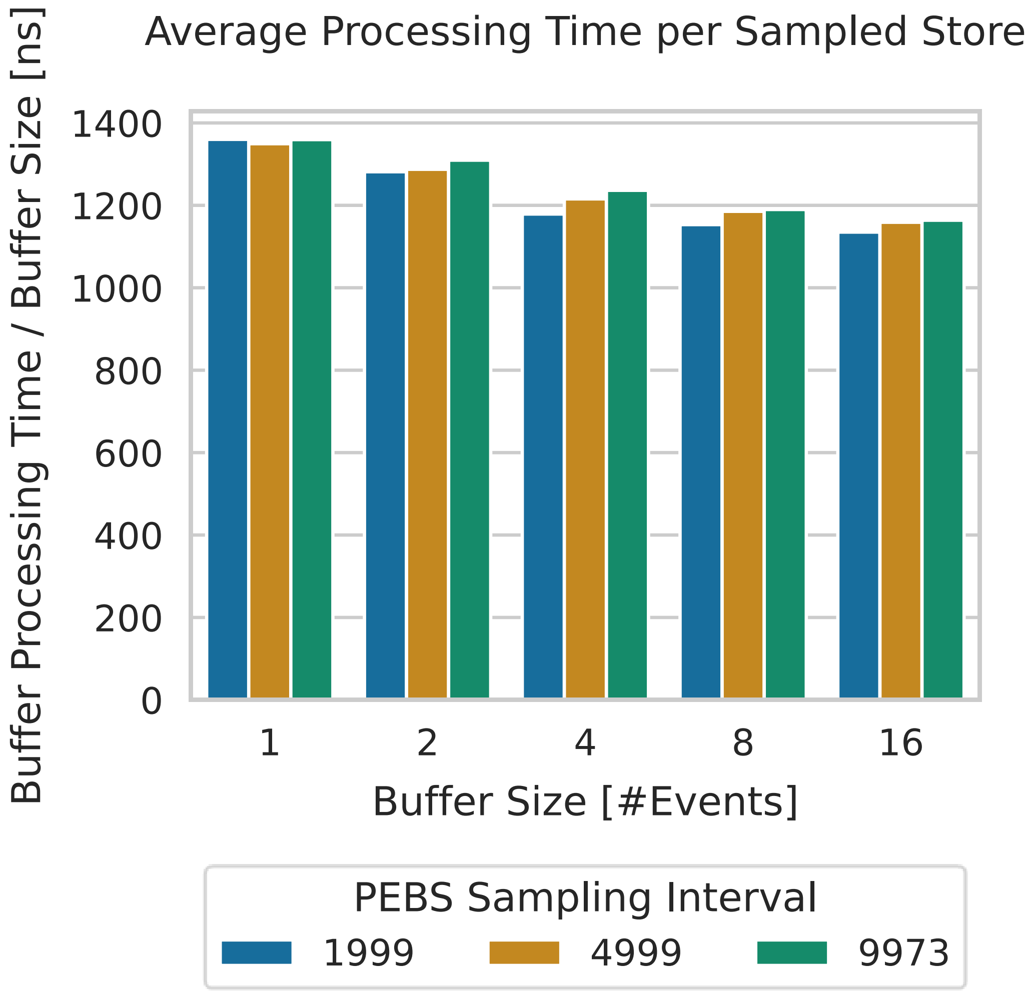
Evaluation: Accuracy



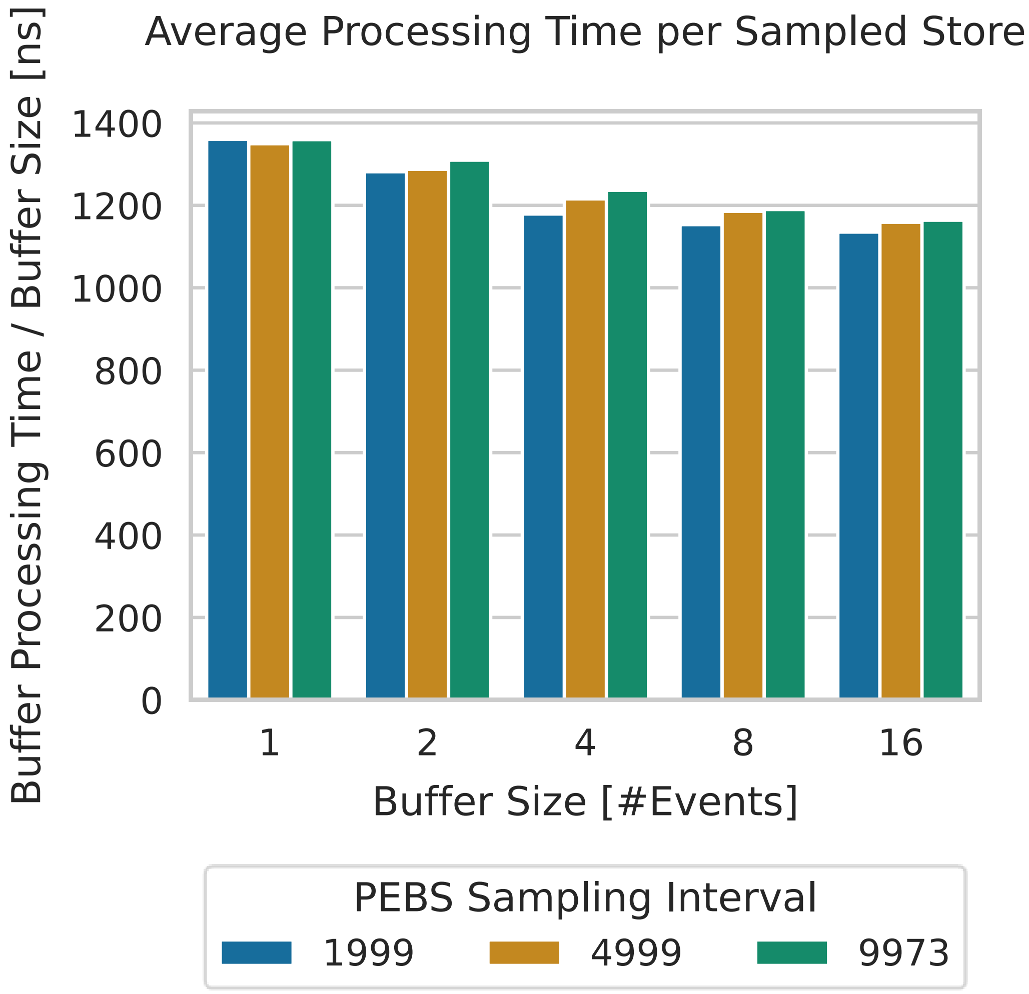
High accuracy for pure stores

Large amount of clwb samples

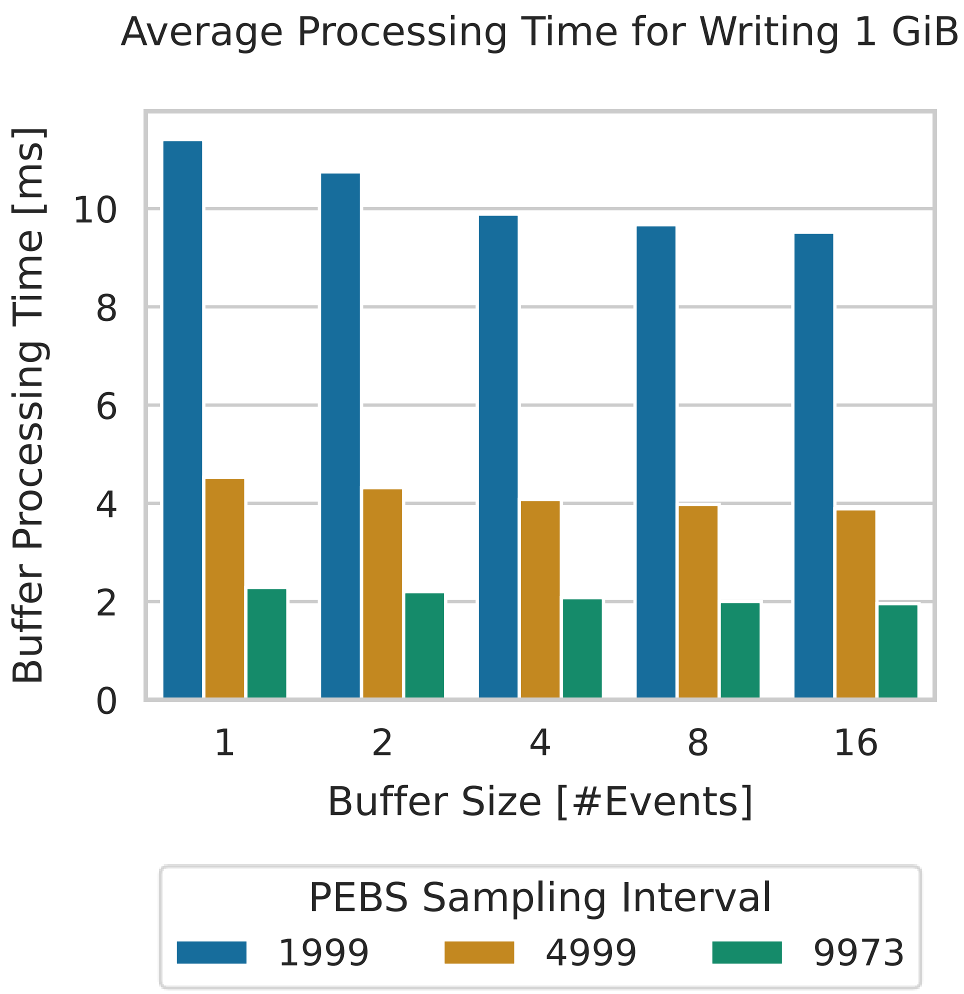
Evaluation: Overhead



Evaluation: Overhead

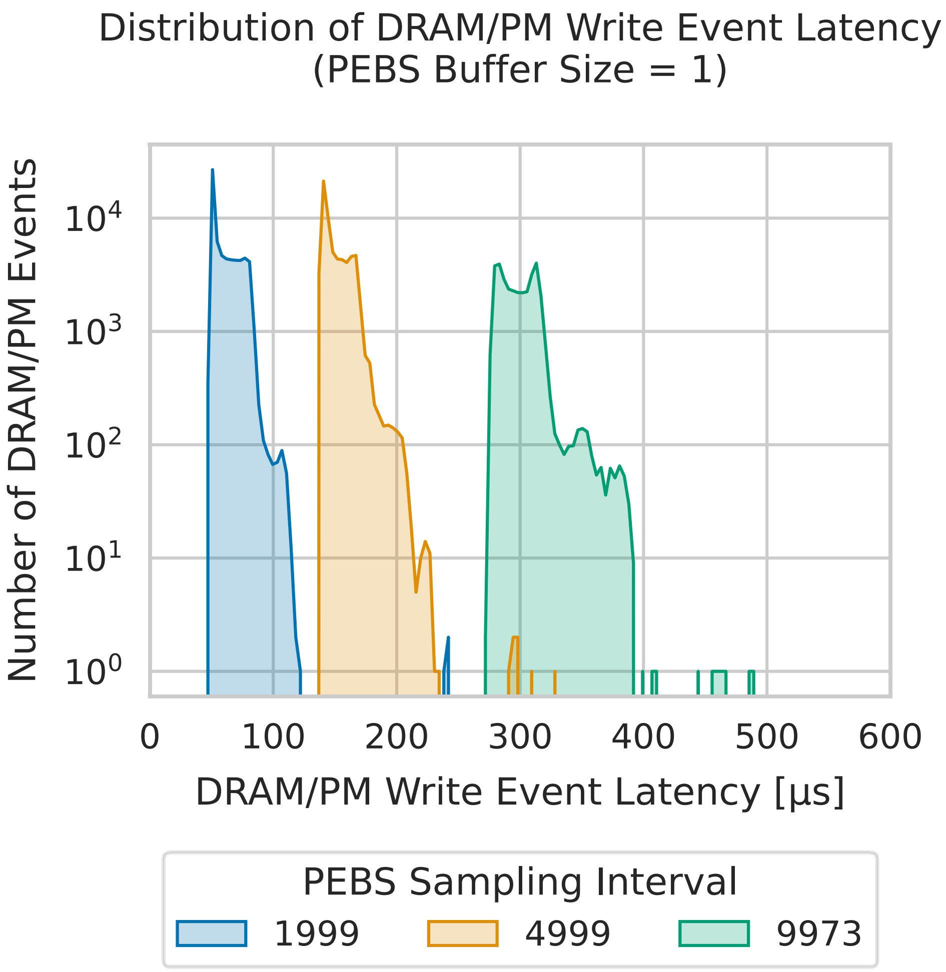


Average Processing Time per Sampled Store



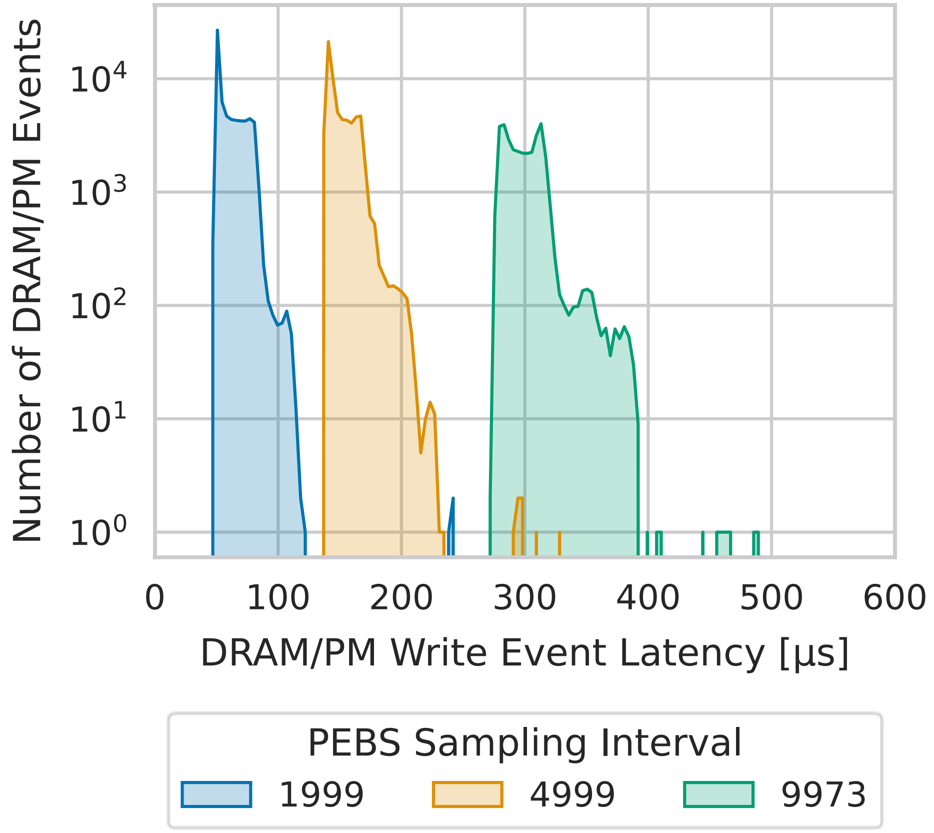
Average Processing Time for Writing 1 GiB

Evaluation: Latency



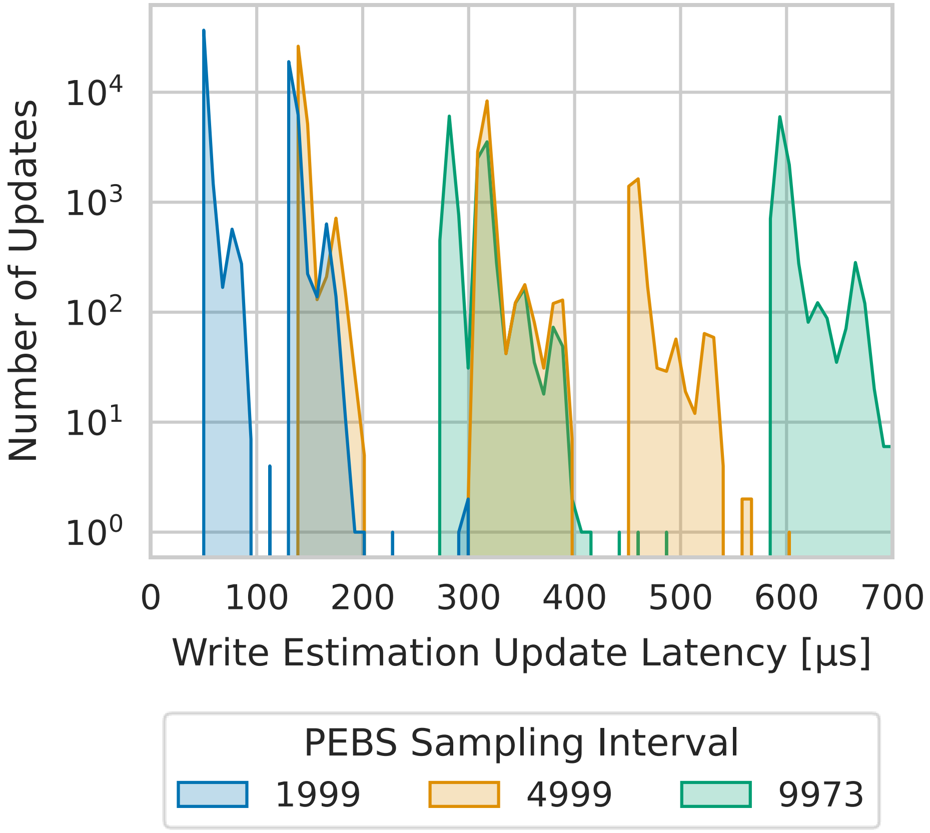
Evaluation: Latency

Distribution of DRAM/PM Write Event Latency
(PEBS Buffer Size = 1)

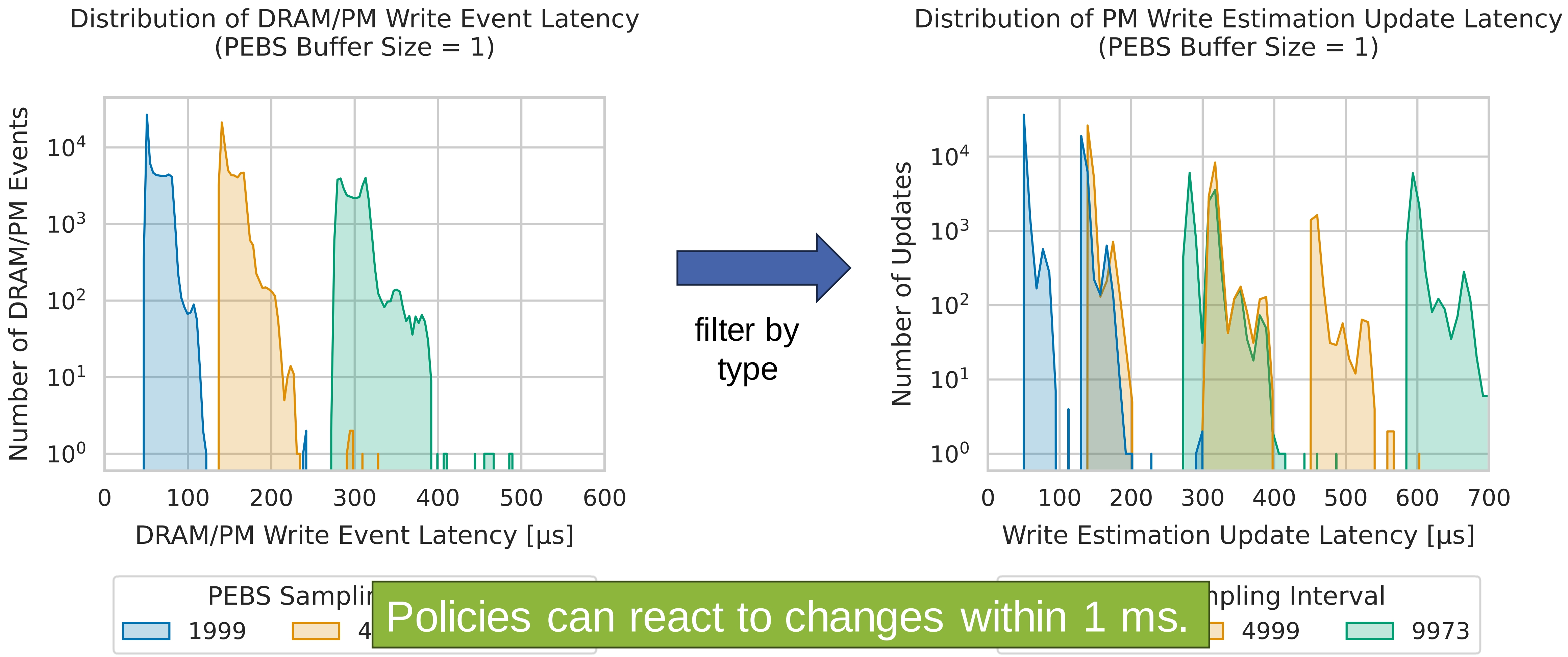


filter by type

Distribution of PM Write Estimation Update Latency
(PEBS Buffer Size = 1)



Evaluation: Latency



Distribution of DRAM/PM Write Event Latency
(PEBS Buffer Size = 1)

Distribution of PM Write Estimation Update Latency
(PEBS Buffer Size = 1)

filter by type

Policies can react to changes within 1 ms.

Discussion

- Accuracy
  - Problem: cache hits
  - Problem: write amplification

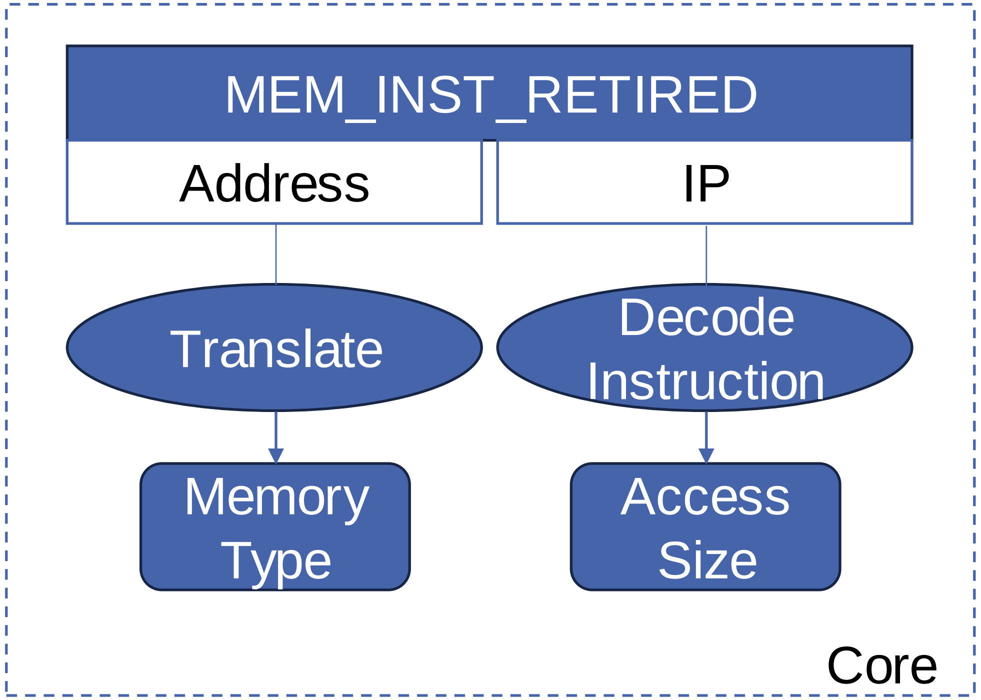
Policies only need relative numbers

- Overhead / latency trade-off
  - Sampling interval
  - Sampling buffer size

- Latency
  - Simultaneous heterogeneous memory use increases latency

Conclusion

- Memory bandwidth is a limited resource
  - Different technologies
  - Monitoring and management important
  - Existing tools insufficient
- Our solution: Sampling with PEBS
  - Process association
  - High accuracy, low overhead
  - Low-latency interface for throttling
- Future work: Throttling policies


MEM_INST_RETIRED
Address
IP
Translate
Decode Instruction
Memory Type
Access Size
Core