Row Buffer Locality Aware Caching Policies for Hybrid Memories

HanBin Yoon
Justin Meza
Rachata Ausavarungnirun
Rachael Harding
Onur Mutlu

SAFARI

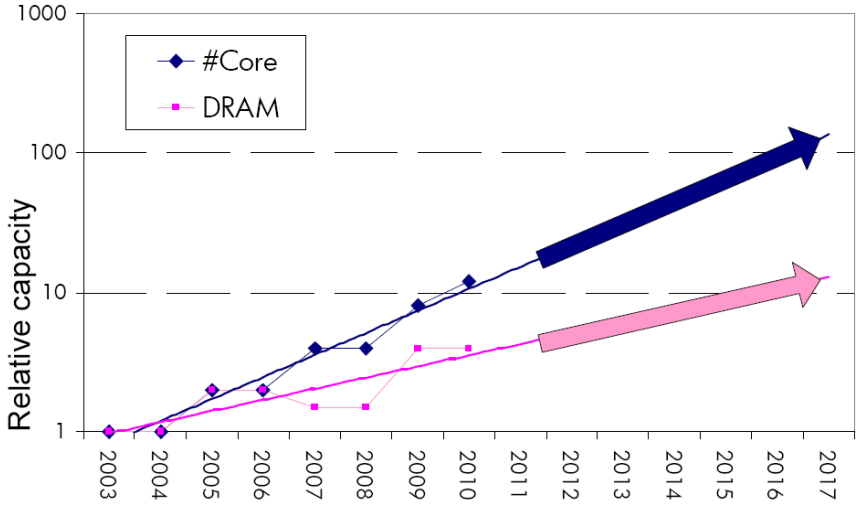
Carnegie Mellon

Executive Summary

- Different memory technologies have different strengths
- A hybrid memory system (DRAM-PCM) aims for best of both
- **Problem:** How to place data between these heterogeneous memory devices?
- **Observation:** PCM array access latency is higher than DRAM's – But peripheral circuit (row buffer) access latencies are similar
- **Key Idea:** Use row buffer locality (RBL) as a key criterion for data placement
- **Solution:** Cache to DRAM rows with low RBL and high reuse
- Improves both performance and energy efficiency over state-of-the-art caching policies

Demand for Memory Capacity

1. Increasing cores and thread contexts
   - Intel Sandy Bridge: 8 cores (16 threads)
   - AMD Abu Dhabi: 16 cores
   - IBM POWER7: 8 cores (32 threads)
   - Sun T4: 8 cores (64 threads)



Source: Lim et al., ISCA 2009.

Demand for Memory Capacity

1. Increasing cores and thread contexts
   - Intel Sandy Bridge: 8 cores (16 threads)
   - AMD Abu Dhabi: 16 cores
   - IBM POWER7: 8 cores (32 threads)
   - Sun T4: 8 cores (64 threads)

2. Modern data-intensive applications operate on increasingly larger datasets
   - Graph, database, scientific workloads

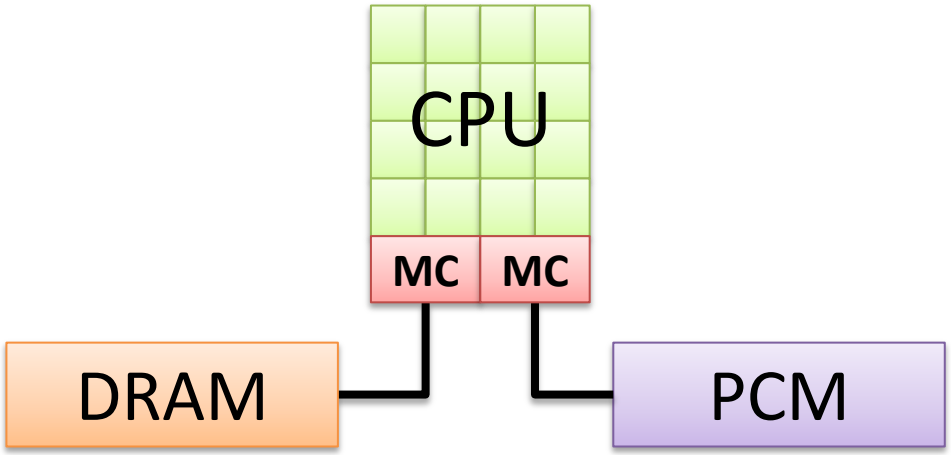
Emerging High Density Memory

- DRAM density scaling becoming costly
- Promising: Phase change memory (PCM)
  - + Projected 3–12× denser than DRAM [Mohan HPTS'09]
  - + Non-volatile data storage
- However, cannot simply replace DRAM
  - – Higher access latency (4–12× DRAM) [Lee+ ISCA'09]
  - – Higher dynamic energy (2–40× DRAM) [Lee+ ISCA'09]
  - – Limited write endurance (~10^8 writes) [Lee+ ISCA'09]

→ Employ both DRAM and PCM

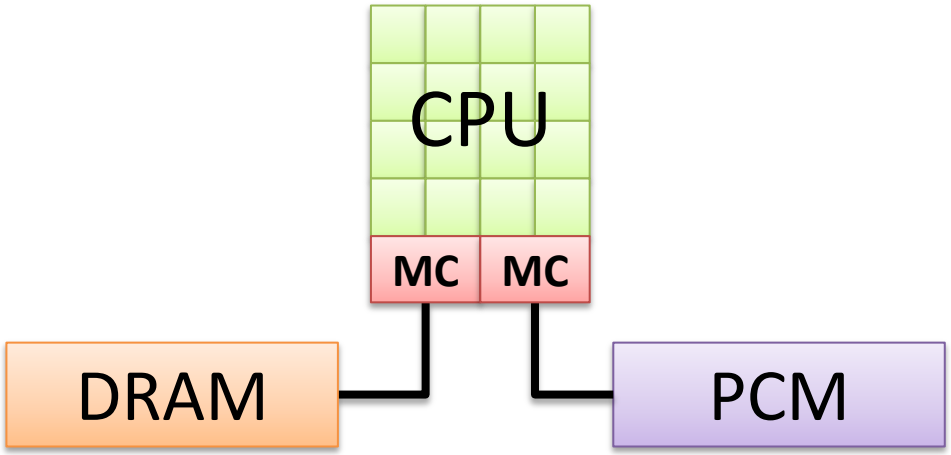
Hybrid Memory

- Benefits from both DRAM and PCM
  - DRAM: Low latency, dyn. energy, high endurance
  - PCM: High capacity, low static energy (no refresh)



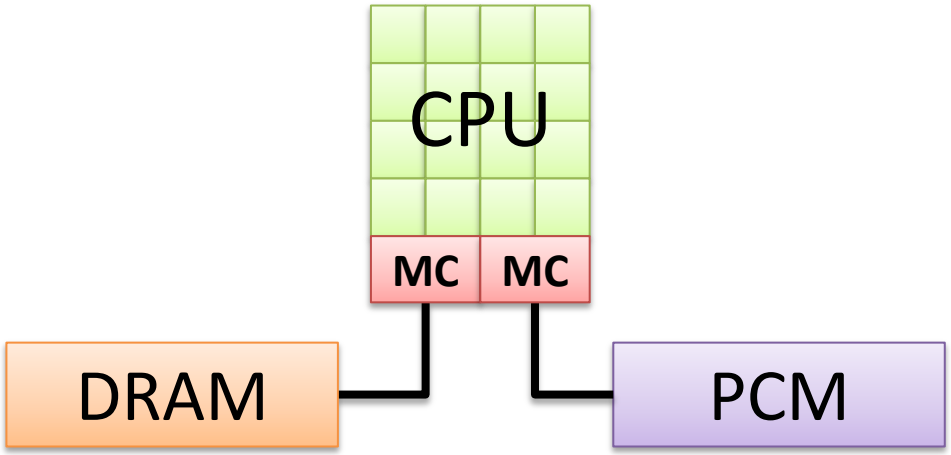
Hybrid Memory

- Design direction: DRAM as a cache to PCM
  [Qureshi+ ISCA'09]
  - Need to avoid excessive data movement
  - Need to efficiently utilize the DRAM cache



Hybrid Memory

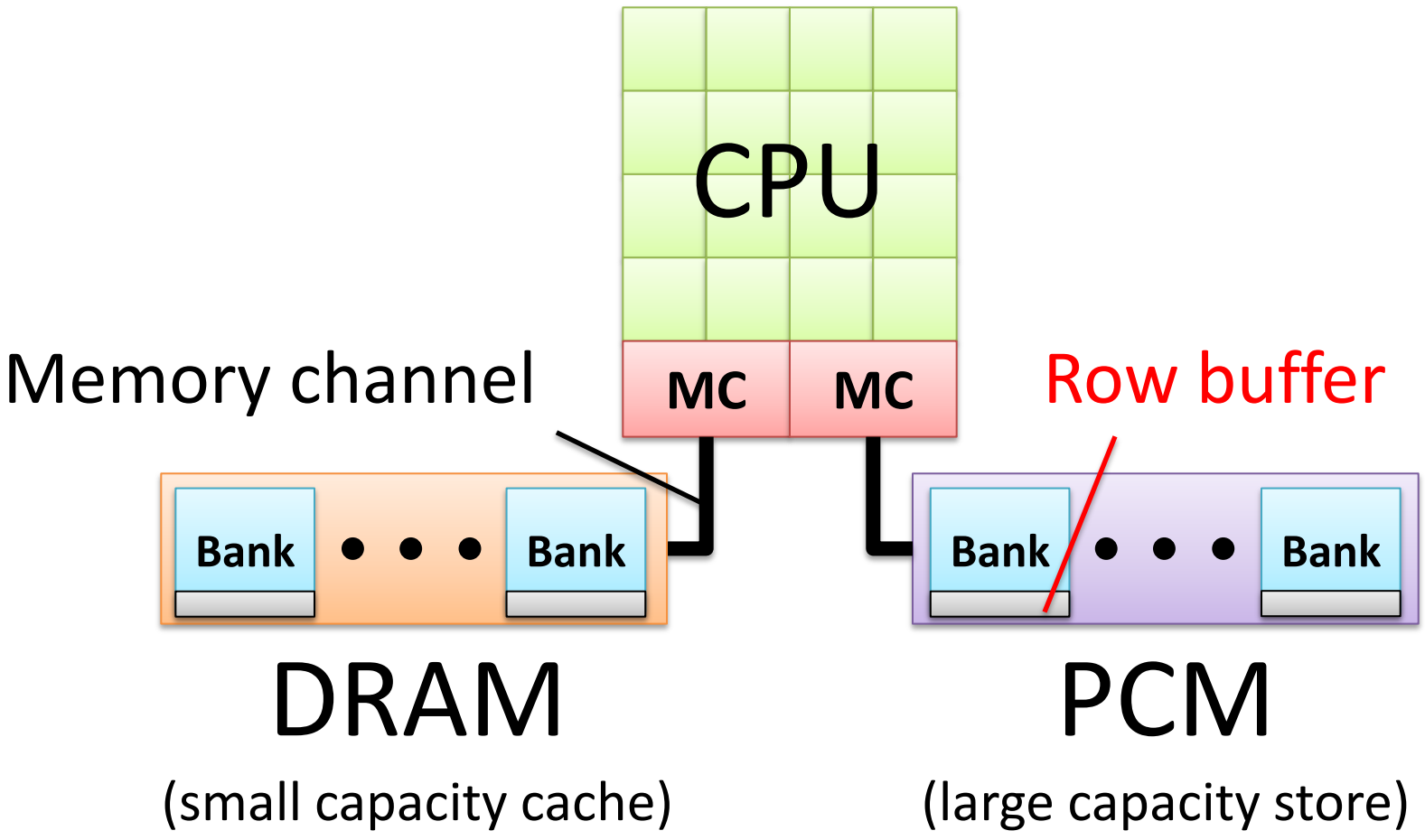
- Key question: How to place data between the heterogeneous memory devices?



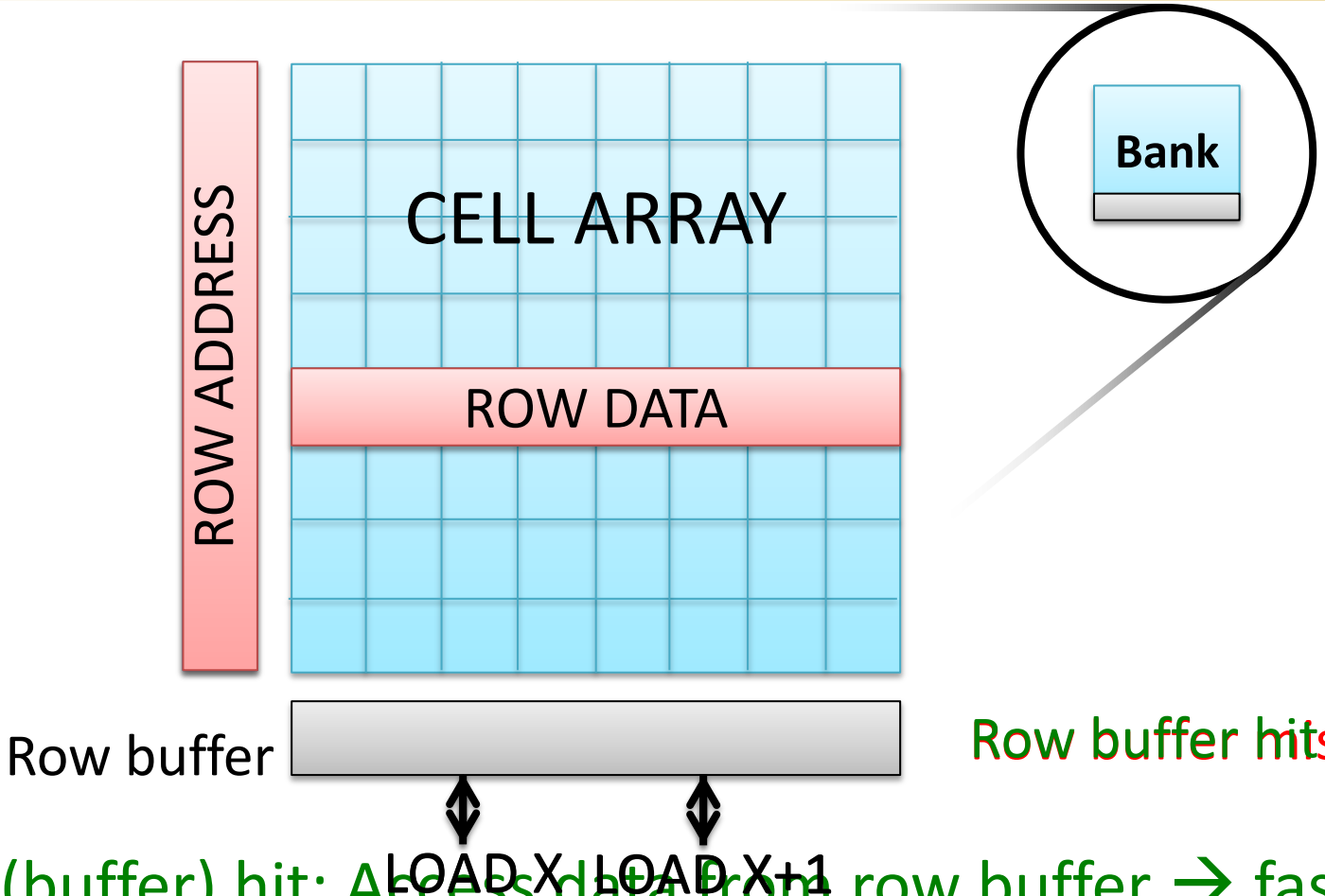
Outline

- Background: Hybrid Memory Systems
- Motivation: Row Buffers and Implications on Data Placement
- Mechanisms: Row Buffer Locality-Aware Caching Policies
- Evaluation and Results
- Conclusion

Hybrid Memory: A Closer Look



Row Buffers and Latency



Row (buffer) hit: Access data from row buffer → fast

Row (buffer) miss: Access data from cell array → slow

Key Observation

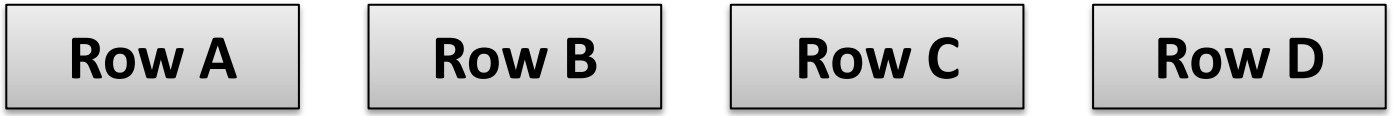
- Row buffers exist in both DRAM and PCM
  - Row hit latency **similar** in DRAM & PCM [Lee+ ISCA'09]
  - Row miss latency **small** in DRAM, **large** in PCM

- Place data in DRAM which
  - is likely to miss in the row buffer (low row buffer locality)→ miss penalty is smaller in DRAM

    AND
  - is reused many times → cache only the data worth the movement cost and DRAM space

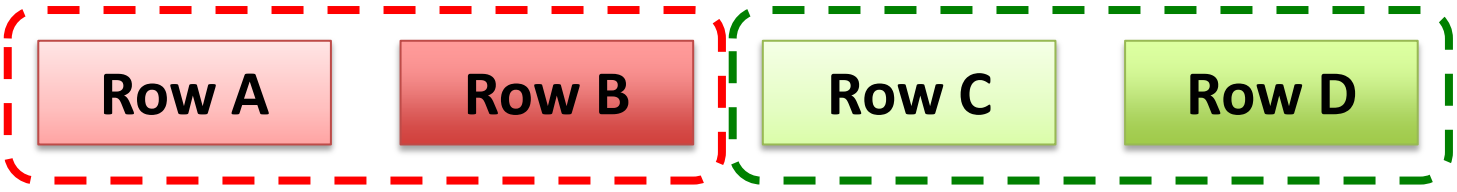
RBL-Awareness: An Example

Let’s say a processor accesses four rows



RBL-Awareness: An Example

Let's say a processor accesses four rows with different row buffer localities (RBL)



Low RBL
(Frequently miss in row buffer)

High RBL
(Frequently hit in row buffer)

Case 1: RBL-*Unaware* Policy (state-of-the-art)

Case 2: RBL-Aware Policy (RBLA)

Case 1: RBL-*Unaware* Policy

A **row buffer locality-*unaware*** policy could place these rows in the following manner

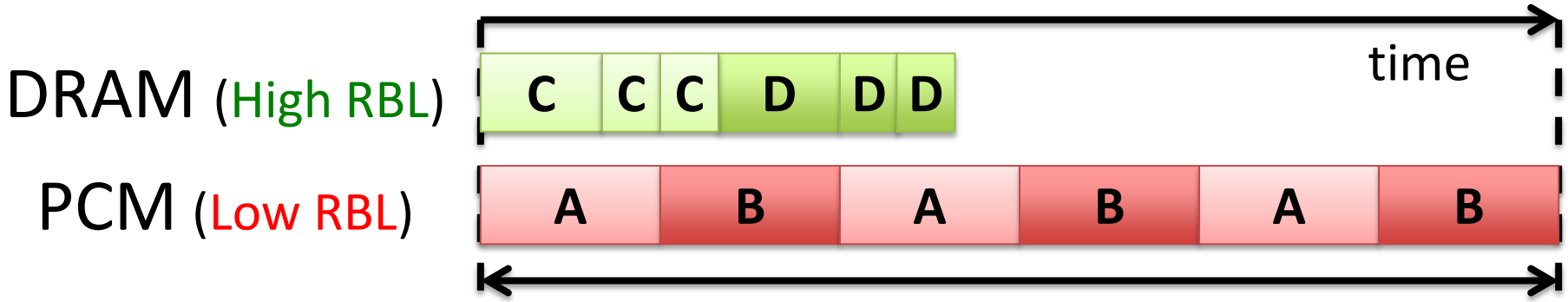
DRAM (High RBL)

PCM (Low RBL)

Case 1: RBL-*Unaware* Policy

Access pattern to main memory:

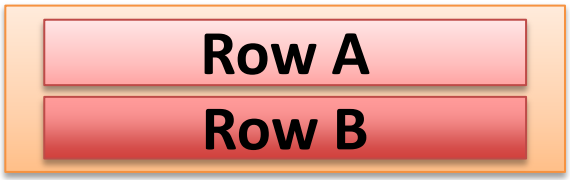
A (oldest), B, C, C, C, A, B, D, D, D, A, B (youngest)



RBL-*Unaware*: Stall time is 6 PCM device accesses

Case 2: RBL-Aware Policy (RBLA)

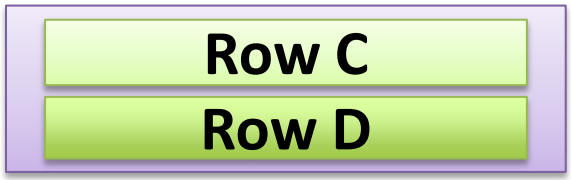
A **row buffer locality-aware** policy would place these rows in the **opposite** manner



DRAM

(Low RBL)

→ Access data at lower row buffer miss latency of DRAM



PCM

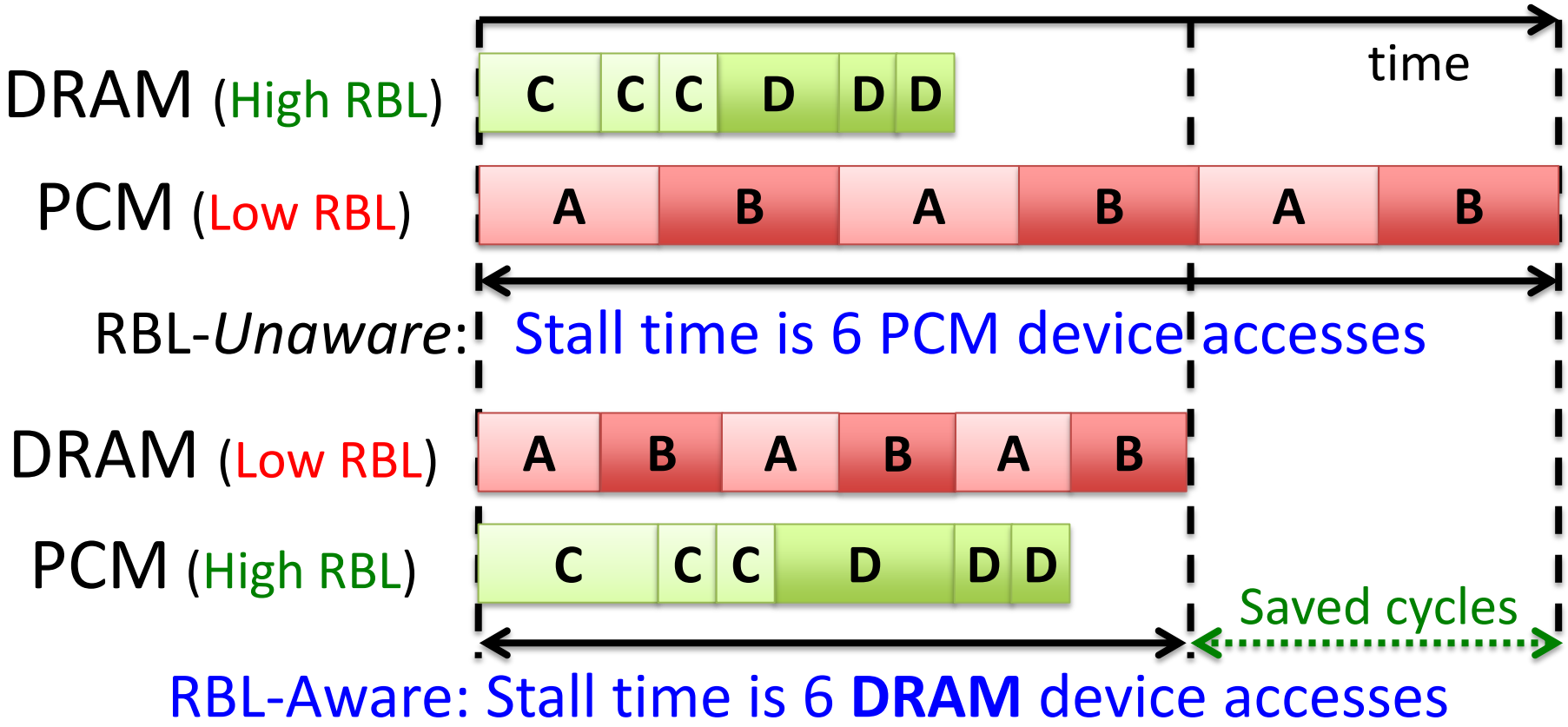
(High RBL)

→ Access data at low row buffer hit latency of PCM

Case 2: RBL-Aware Policy (RBLA)

Access pattern to main memory:

A (oldest), B, C, C, C, A, B, D, D, D, A, B (youngest)



Outline

- Background: Hybrid Memory Systems
- Motivation: Row Buffers and Implications on Data Placement
- Mechanisms: Row Buffer Locality-Aware Caching Policies
- Evaluation and Results
- Conclusion

Our Mechanism: RBLA

1. For recently used rows in PCM:
   – Count row buffer misses as indicator of row buffer locality (RBL)

2. Cache to DRAM rows with misses $\geq$ threshold
   – Row buffer miss counts are periodically reset (only cache rows with high reuse)

Our Mechanism: RBLA-Dyn

1. For recently used rows in PCM:
   – Count row buffer misses as indicator of row buffer locality (RBL)

2. Cache to DRAM rows with misses $\geq$ threshold
   – Row buffer miss counts are periodically reset (only cache rows with high reuse)

3. Dynamically adjust threshold to adapt to workload/system characteristics
   – Interval-based cost-benefit analysis

Implementation: "Statistics Store"

- Goal: To keep count of row buffer misses to recently used rows in PCM

- Hardware structure in memory controller
  - Operation is similar to a cache
    - Input: row address
    - Output: row buffer miss count
  - 128-set 16-way statistics store (9.25KB) achieves system performance within 0.3% of an unlimited-sized statistics store

Outline

- Background: Hybrid Memory Systems
- Motivation: Row Buffers and Implications on Data Placement
- Mechanisms: Row Buffer Locality-Aware Caching Policies
- Evaluation and Results
- Conclusion

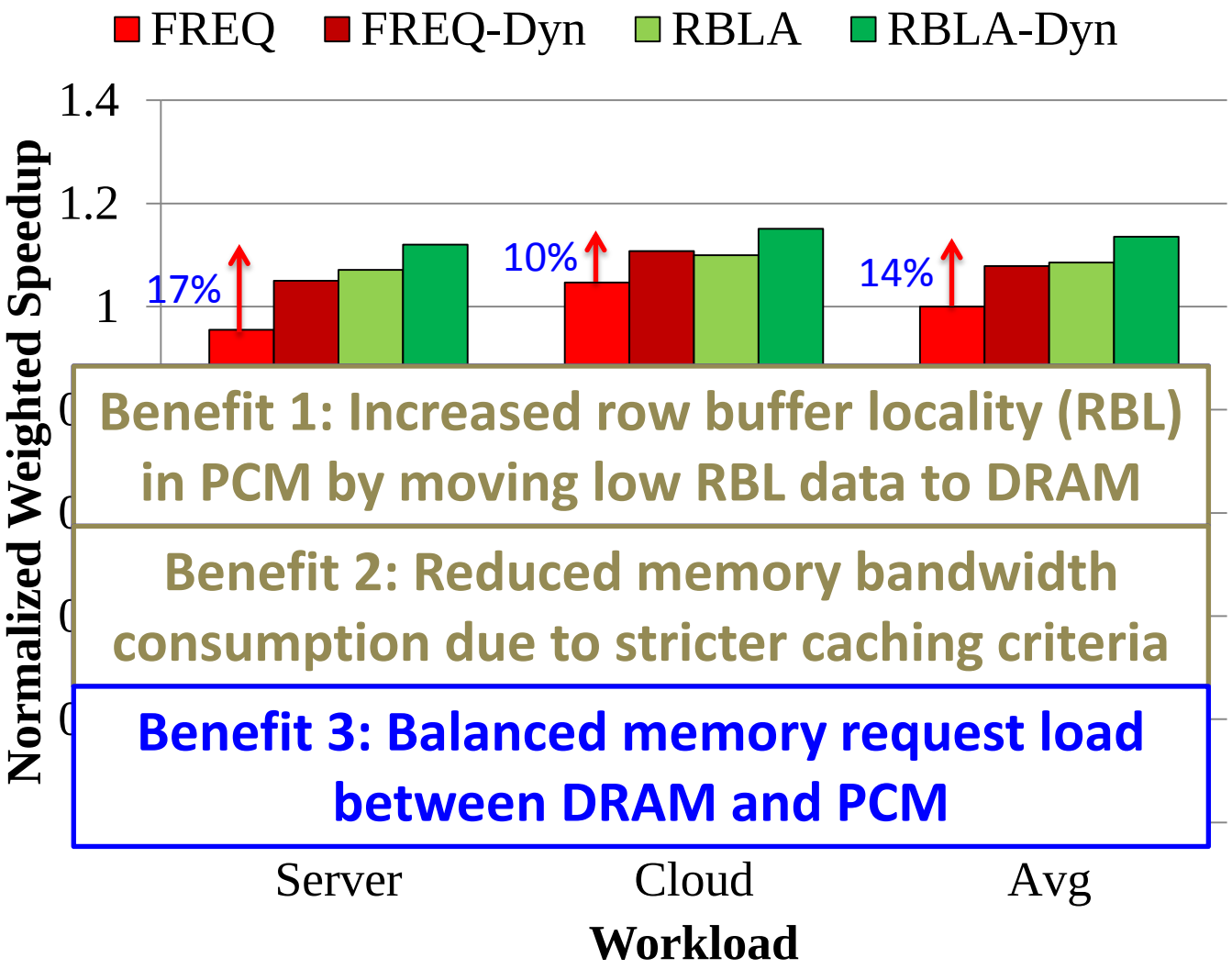
Evaluation Methodology

- Cycle-level x86 CPU-memory simulator
  - **CPU**: 16 out-of-order cores, 32KB private L1 per core, 512KB shared L2 per core
  - **Memory**: 1GB DRAM (8 banks), 16GB PCM (8 banks), 4KB migration granularity
- 36 multi-programmed server, cloud workloads
  - Server: TPC-C (OLTP), TPC-H (Decision Support)
  - Cloud: Apache (Webserv.), H.264 (Video), TPC-C/H
- Metrics: Weighted speedup (perf.), perf./Watt (energy eff.), Maximum slowdown (fairness)

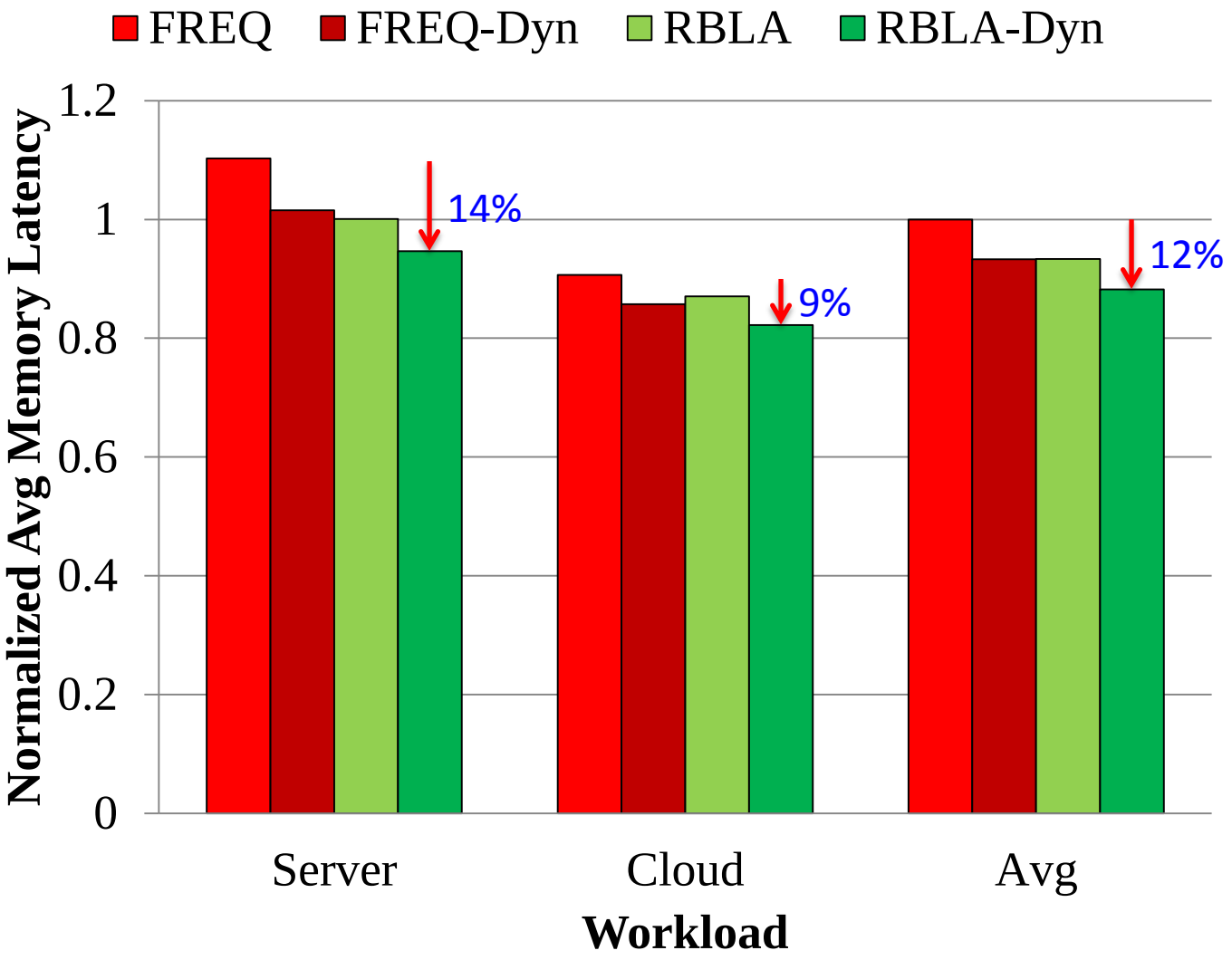
Comparison Points

- **Conventional LRU Caching**
- **FREQ**: Access-frequency-based caching
  - Places "hot data" in cache [Jiang+ HPCA'10]
  - Cache to DRAM rows with accesses ≥ threshold
  - Row buffer locality-*unaware*
- **FREQ-Dyn**: Adaptive Freq.-based caching
  - **FREQ** + our dynamic threshold adjustment
  - Row buffer locality-*unaware*
- **RBLA**: Row buffer locality-aware caching
- **RBLA-Dyn**: Adaptive RBL-aware caching

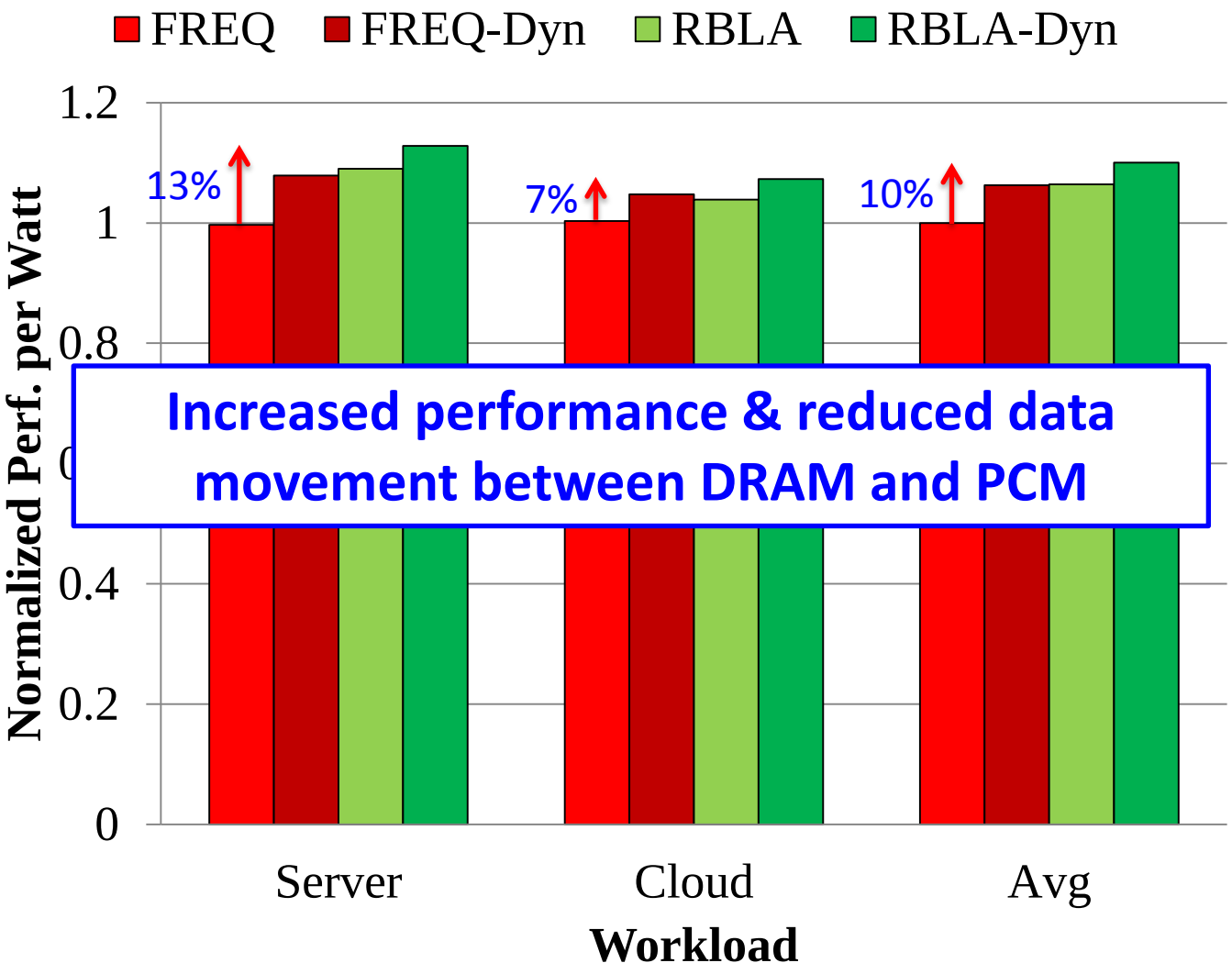
System Performance



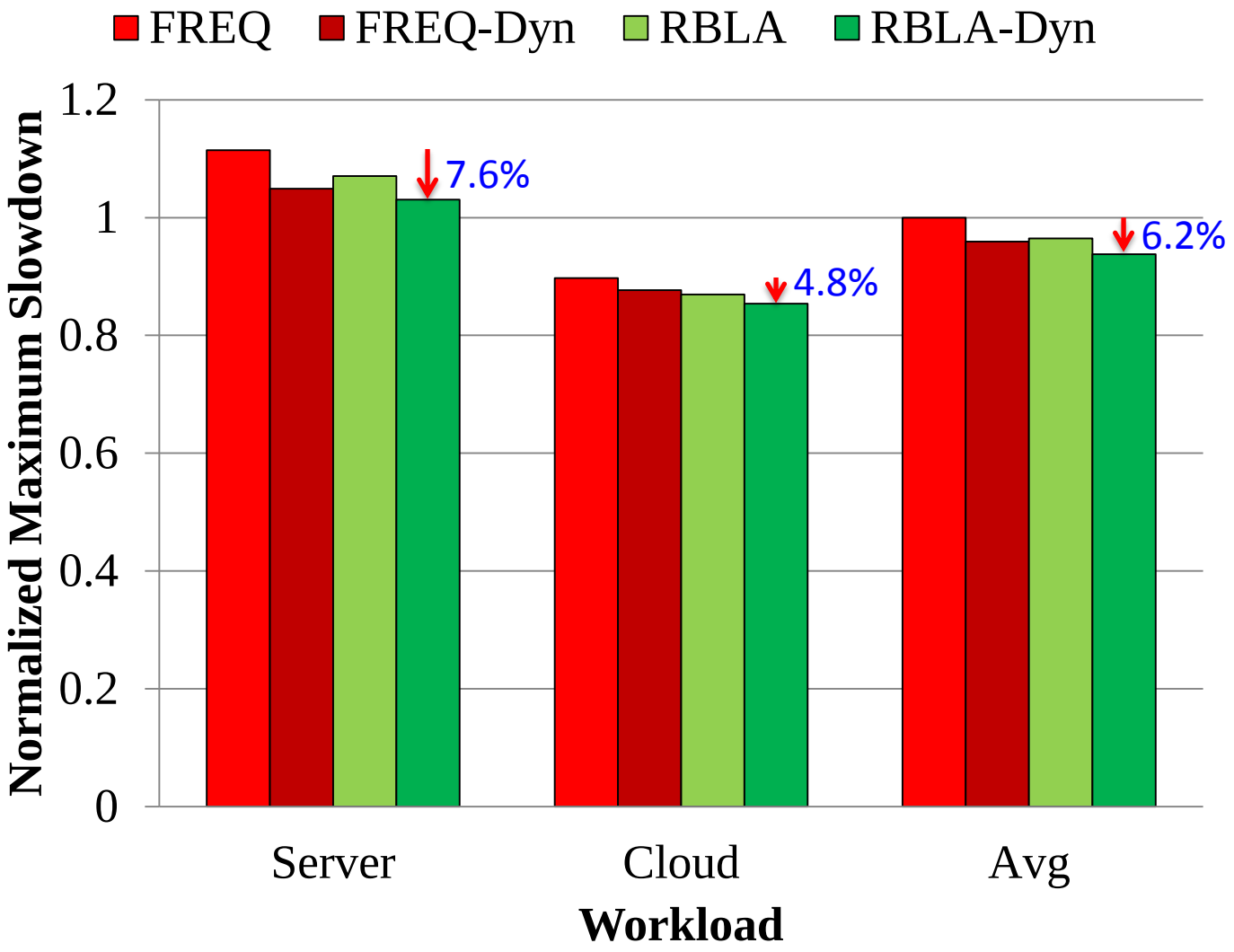
Average Memory Latency



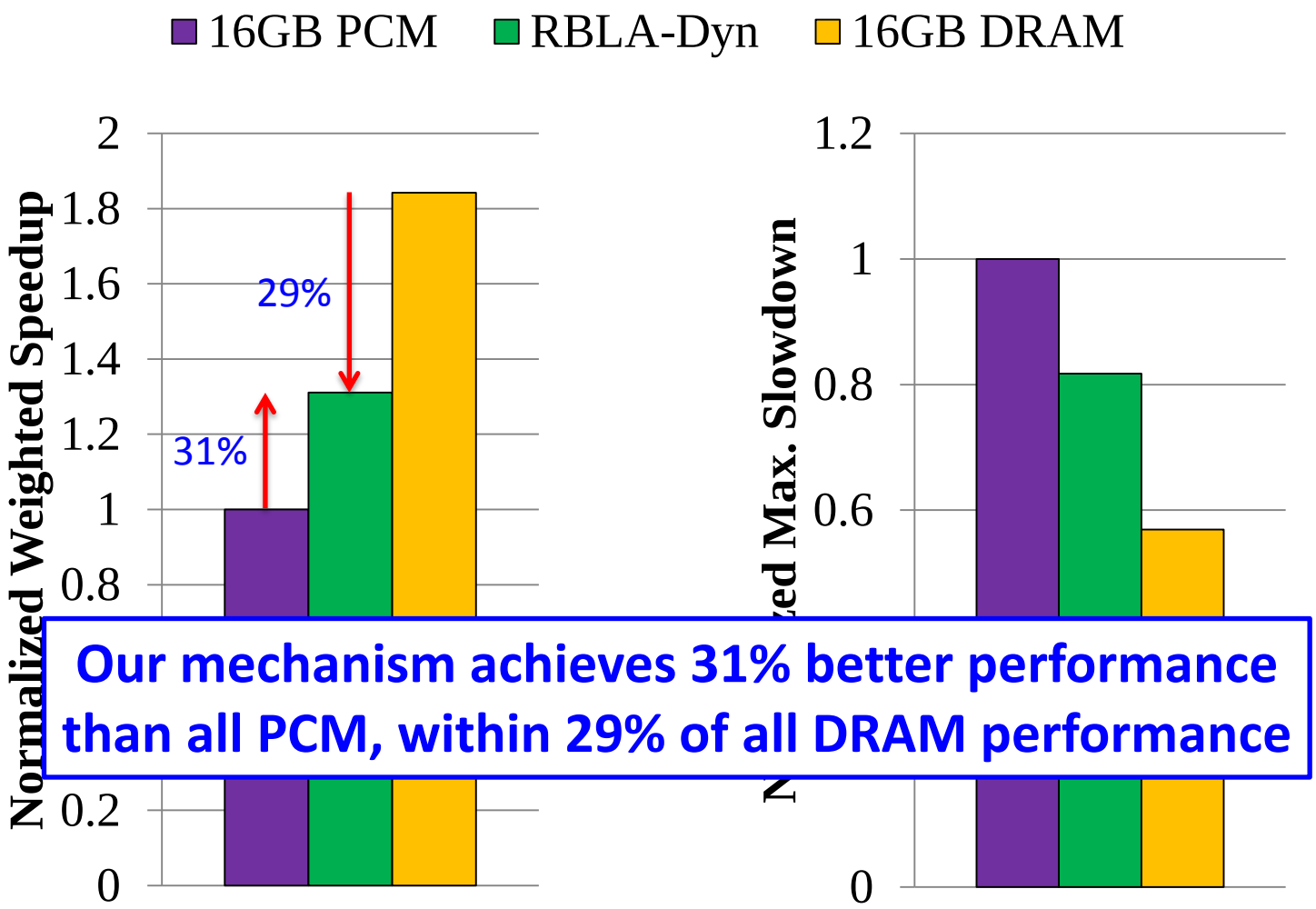
Memory Energy Efficiency



Thread Fairness



Compared to All-PCM/DRAM



Other Results in Paper

- RBLA-Dyn increases the portion of PCM row buffer hit by 6.6 times
- RBLA-Dyn has the effect of balancing memory request load between DRAM and PCM
  - PCM channel utilization increases by 60%.

Summary

- Different memory technologies have different strengths
- A hybrid memory system (DRAM-PCM) aims for best of both
- **Problem:** How to place data between these heterogeneous memory devices?
- **Observation:** PCM array access latency is higher than DRAM's – But peripheral circuit (row buffer) access latencies are similar
- **Key Idea:** Use row buffer locality (RBL) as a key criterion for data placement
- **Solution:** Cache to DRAM rows with low RBL and high reuse
- Improves both performance and energy efficiency over state-of-the-art caching policies

Thank you! Questions?

Row Buffer Locality Aware Caching Policies for Hybrid Memories

HanBin Yoon
Justin Meza
Rachata Ausavarungnirun
Rachael Harding
Onur Mutlu

Carnegie Mellon University

Appendix

Cost-Benefit Analysis (1/2)

- Each quantum, we measure the *first-order* costs and benefits under the current threshold
  - Cost = cycles expended for data movement
  - Benefit = cycles saved servicing requests in DRAM versus PCM
- Cost = Migrations $\times$ $t_{migration}$
- Benefit = $Reads_{DRAM} \times (t_{read,PCM} - t_{read,DRAM})$
  $+ Writes_{DRAM} \times (t_{write,PCM} - t_{write,DRAM})$

Cost-Benefit Analysis (2/2)

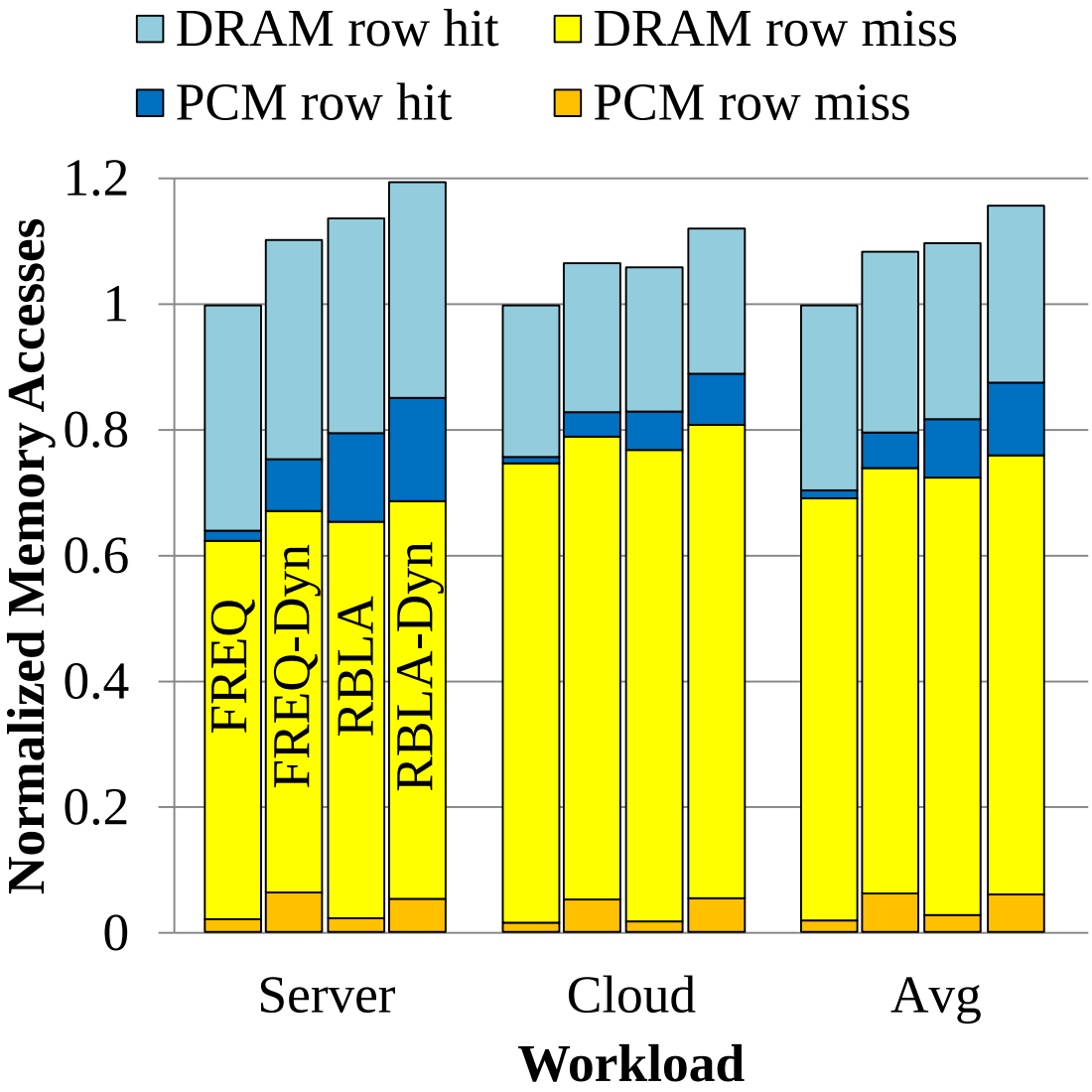
- Dynamic Threshold Adjustment Algorithm

```
NetBenefit = Benefit - Cost
if (NetBenefit < 0)
        MissThresh++
else if (NetBenefit > PreviousNetBenefit)
        if (MissThresh was previously incremented)
                MissThresh++
        else
                MissThresh--
else
        if (MissThresh was previously incremented)
                MissThresh--
        else
                MissThresh++
PreviousNetBenefit = NetBenefit
```

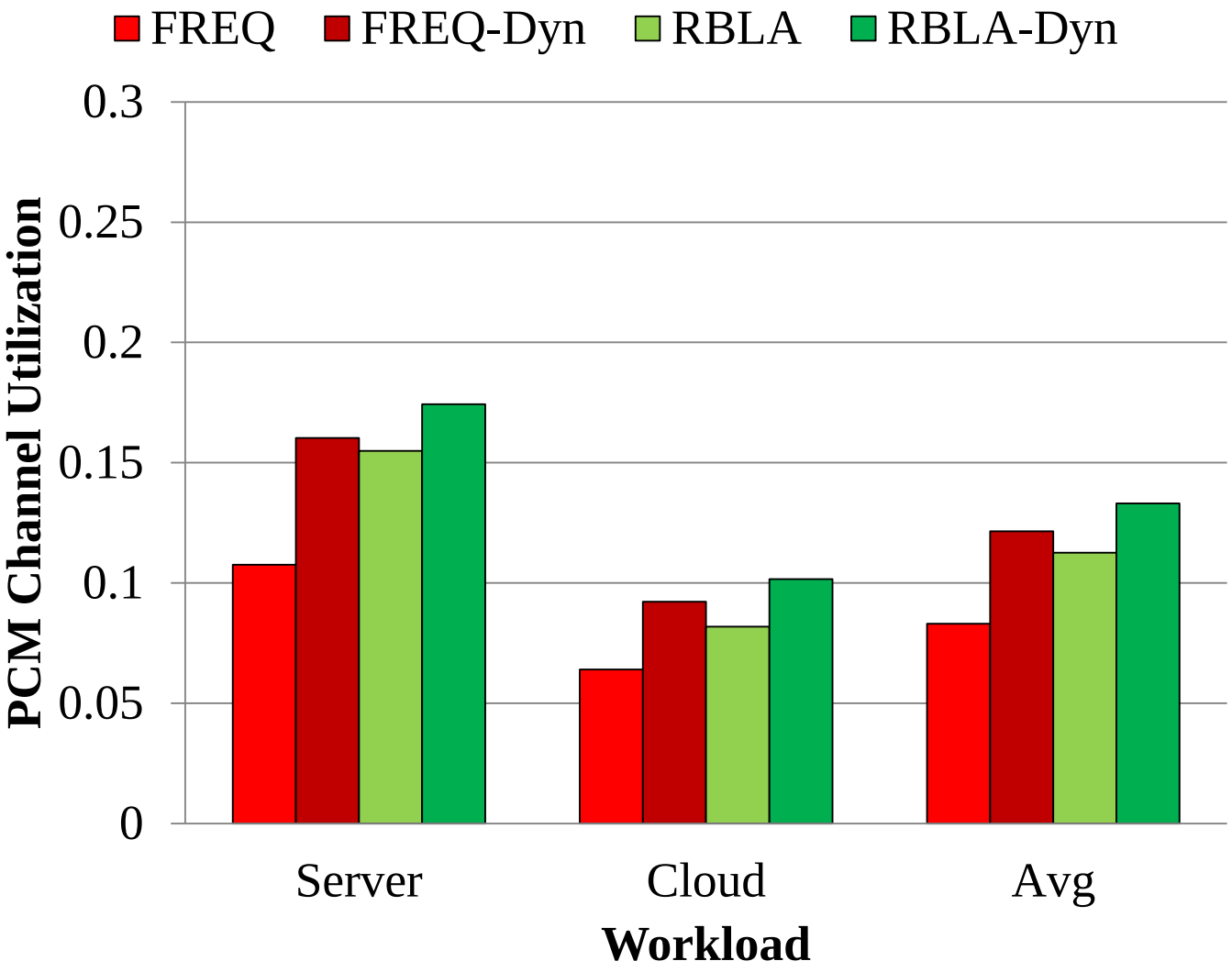
Simulator Parameters

- Core model
  - 3-wide issue with 128-entry instruction window
  - Private 32 KB per core L1 cache
  - Shared 512 KB per core L2 cache
- Memory model
  - 1 GB DRAM (1 rank), 16 GB PCM (1 rank)
  - Separate memory controllers, 8 banks per device
  - Row buffer hit: 40 ns
  - Row buffer miss: 80 ns (DRAM); 128, 368 ns (PCM)
  - Migrate data at 4 KB granularity

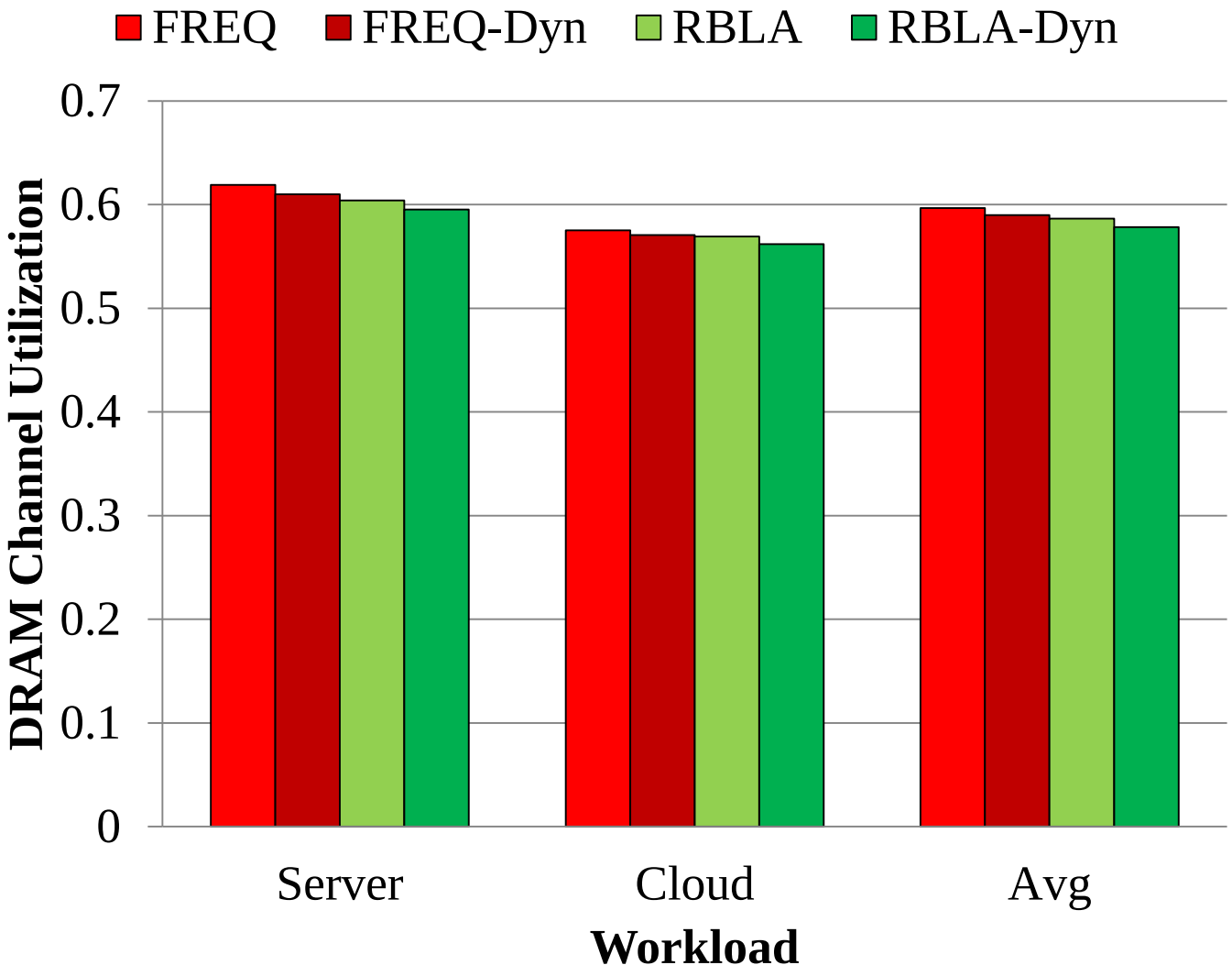
Row Buffer Locality



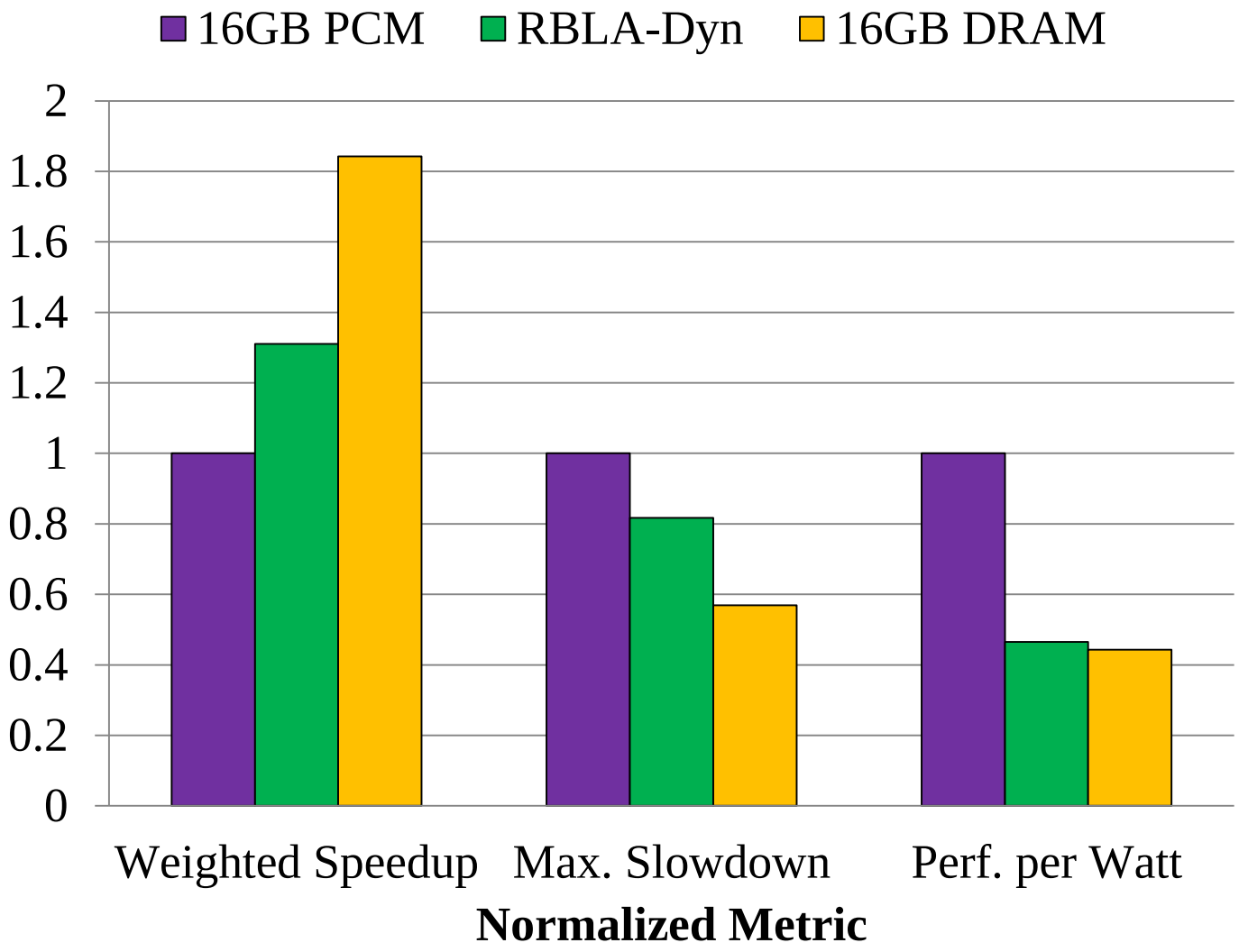
PCM Channel Utilization



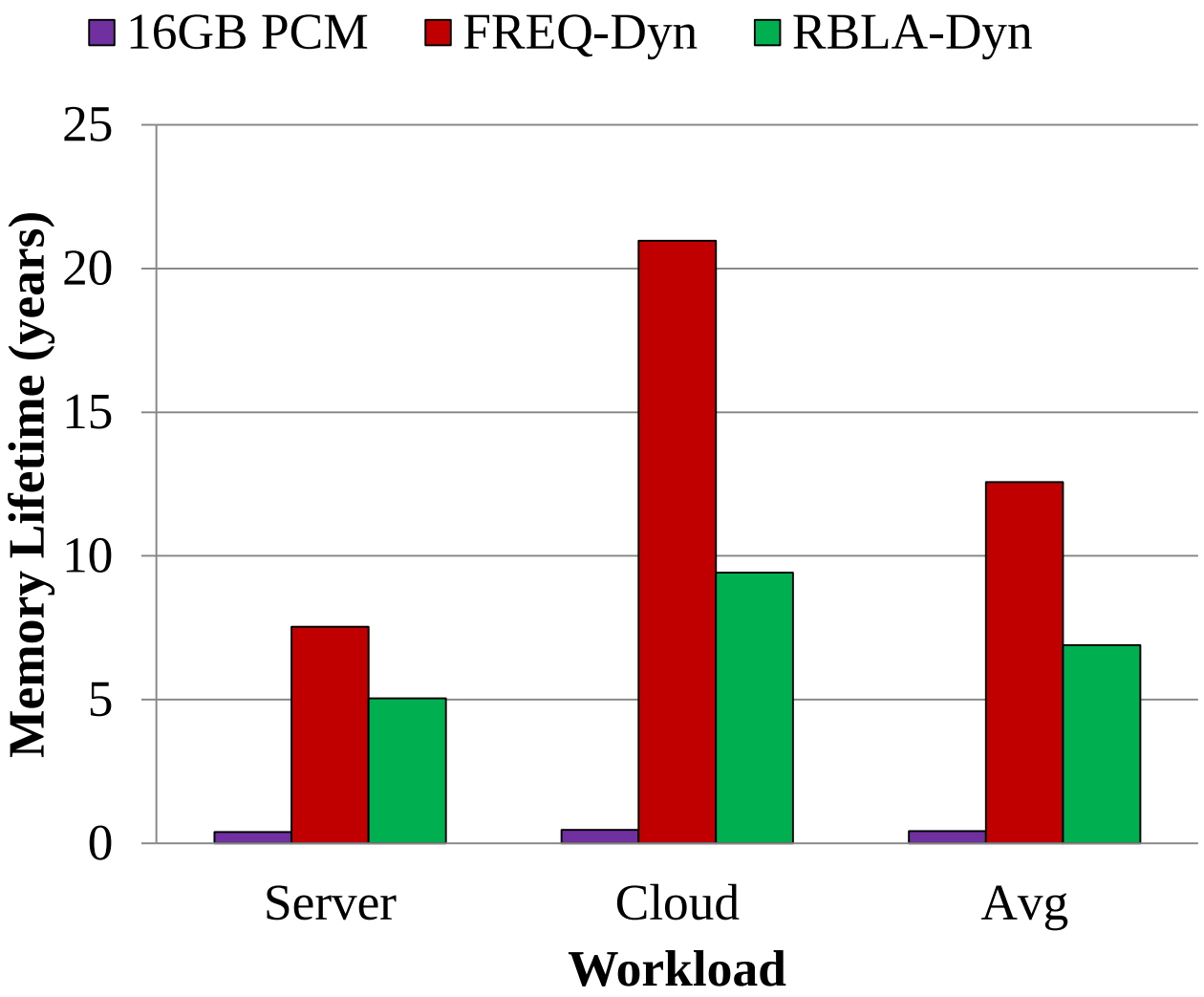
DRAM Channel Utilization



Compared to All-PCM/DRAM



Memory Lifetime



DRAM Cache Hit Rate