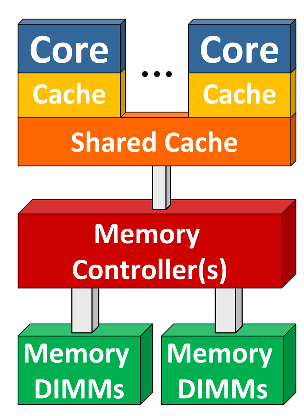
FHM: Fair and High-Performance Memory Scheduling

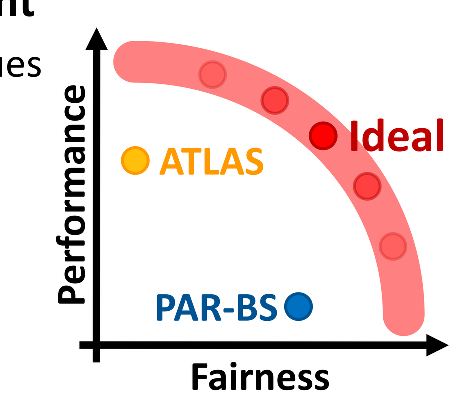
Yoongu Kim, Yu Cai
Department of Electrical and Computer Engineering
Carnegie Mellon University, Pittsburgh, PA 15213
{yoonguk, yucai}@ece.cmu.edu

Michael K. Papamichael
Department of Computer Science
Carnegie Mellon University, Pittsburgh, PA 15213
papamix@cs.cmu.edu

The Memory Scheduling Problem

- **Memory Scheduling:** Translation of memory requests to sequence of DRAM commands
  - Sequencing of commands determined by memory controllers
  - Have to obey many DRAM constraints → **hard problem**
- **Typical memory scheduling algorithms are suboptimal**
  - Traditionally use simple fixed policies: **FR-FCFS**
  - Optimized for **single processor**
  - Cannot adapt to workload's **dynamic memory behavior**
- **Even more complex problem in a CMP environment**
  - Higher contention for memory resources & fairness issues
  - Concurrently running threads destroy locality
- **Current state-of-the-art solutions still suboptimal**
  - Optimize either for **performance** or **fairness**
  - **PAR-BS** sacrifices performance for fairness
  - **ATLAS** sacrifices fairness for performance





Memory Scheduling Issues

1 Memory Request Ranking
- Rank threads based on memory behavior
- Many metrics to choose from
  - **MPKI** (Misses Per Kilo-Instructions)
  - **BLP** (Bank-Level Parallelism)
  - **RBL** (Row-Buffer Locality)

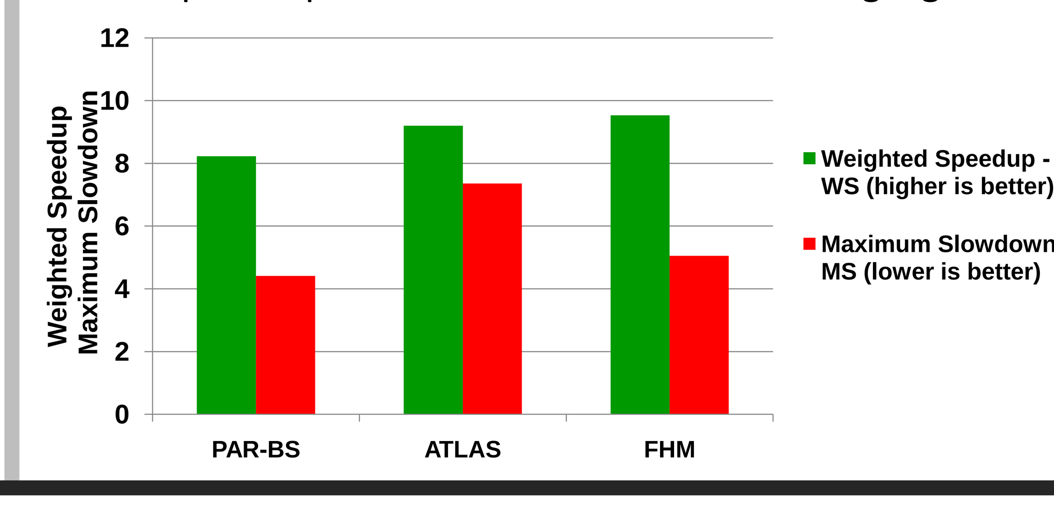
2 Clustering and Shuffling
- Static ranking unfair to lowest-ranked threads
- Cluster lowest-ranked threads & shuffle them
  - Controls performance/fairness trade-off

3 Sub-Row Interleaving
- Memory mapping biased towards BLP or RBL
  - Sub-Row Interleaving can expose both

4 Validating DRAM Timing
- Simulator uses abstract memory timing model
  - Validate against cycle-accurate DRAM model

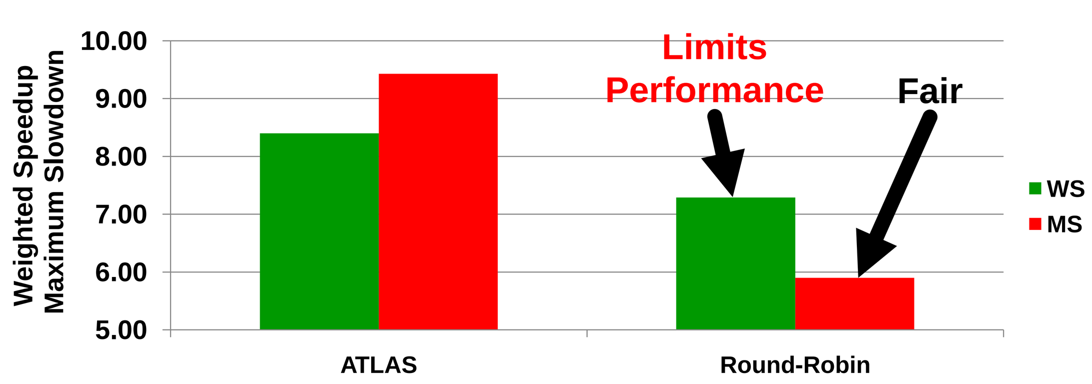
Contributions & Results

- **Novel memory scheduling algorithm (FHM)**
  - Outperforms current state-of-the-art algorithms
  - Considers thread BLP and RBL in addition to MPKI
  - 4% higher weighted speedup
  - 32% lower maximum slowdown
- **Sub-Row Interleaving memory mapping scheme**
  - Exposes Bank-Level Parallelism & Row-Buffer Locality
  - Improves performance across all scheduling algorithms



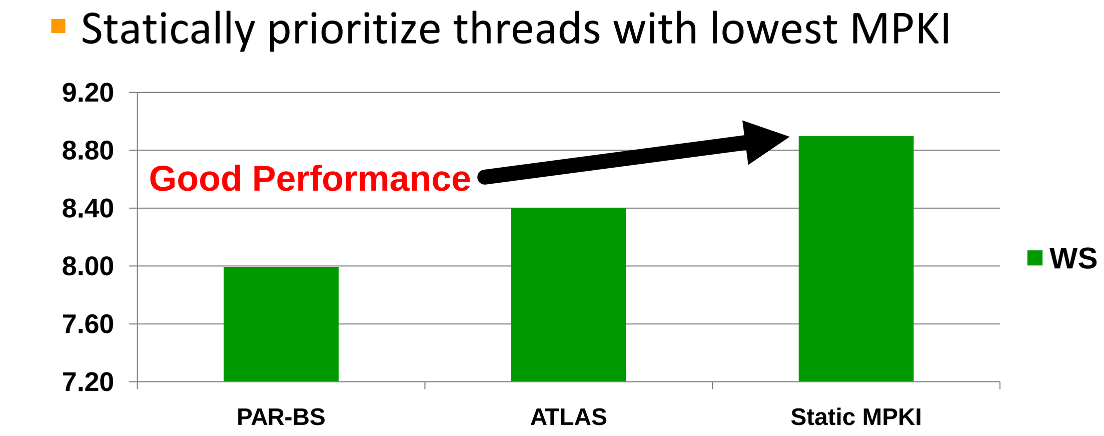
1 Memory Request Ranking

- **Naïve round-robin scheduling**
  - Very fair, **BUT**
  - Limits overall system performance
  - "Light" threads stuck behind memory-bound threads



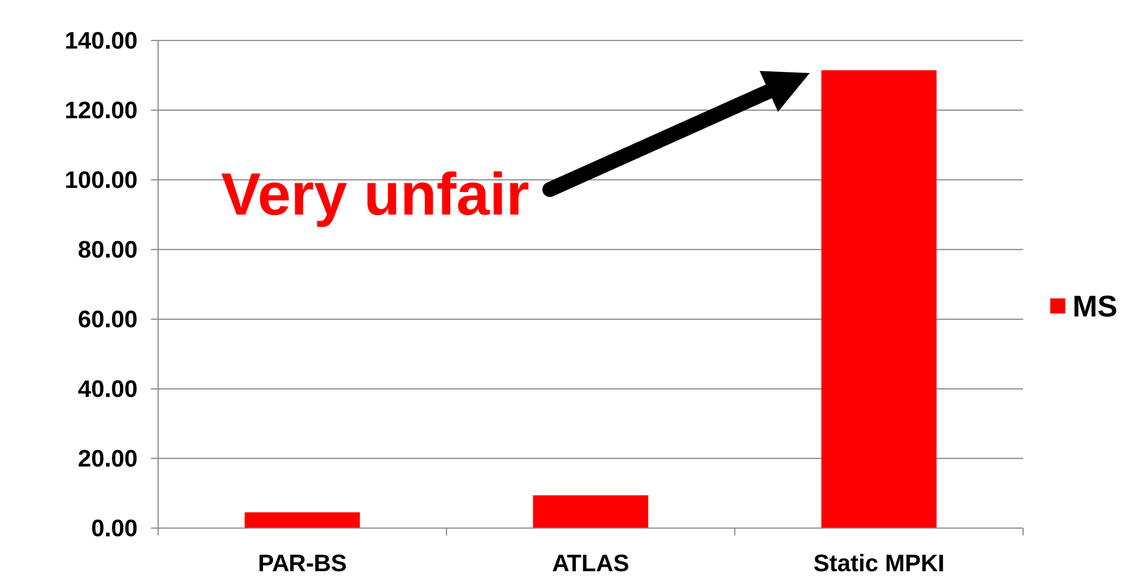
- **Need to be more clever about scheduling**
  - Compute-bound threads are latency-sensitive
  - Memory-bound threads saturate memory bandwidth

- **Schedule threads based on their ranking**
  - Rank threads according to their memory behavior
- **MPKI Ranking**
  - Statically prioritize threads with lowest MPKI



- **"Niceness" Ranking**
  - Incorporates low-level memory behavior
  - Ranks threads based on the interference they cause

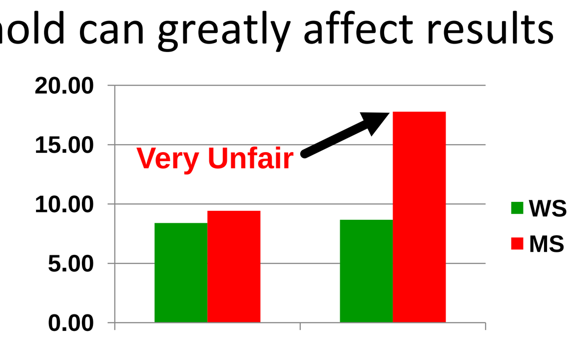
- **Static Ranking improves system throughput, BUT**
  - can severely hurt fairness of lowest-priority threads



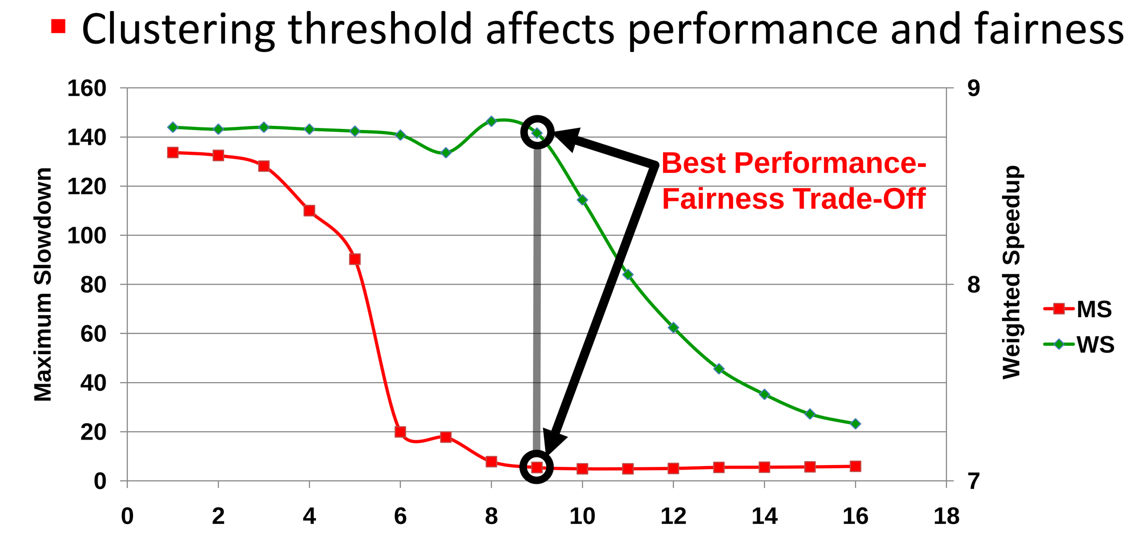
- **Highest-intensity threads are scheduled very rarely**
  - **Idea**: Cluster threads into groups & treat separately

2 Clustering and Shuffling

- **Cluster threads into two groups & treat separately**
  - Low memory-intensity or **compute-bound**
    - Statically prioritize for **performance**
  - High memory-intensity or **memory-bound**
    - Periodically shuffle to maintain **fairness**
- **Thread clustering**
  - Statically partition based on workload knowledge
  - **Caution:** Partition threshold can greatly affect results



- **What if you don't have any workload knowledge?**
  - Dynamically determine clustering threshold
  - Clustering threshold affects performance and fairness

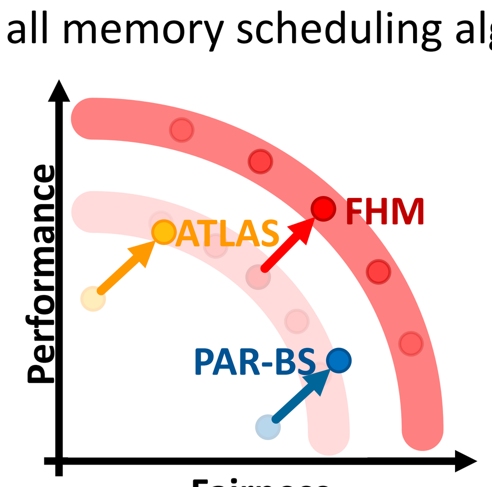


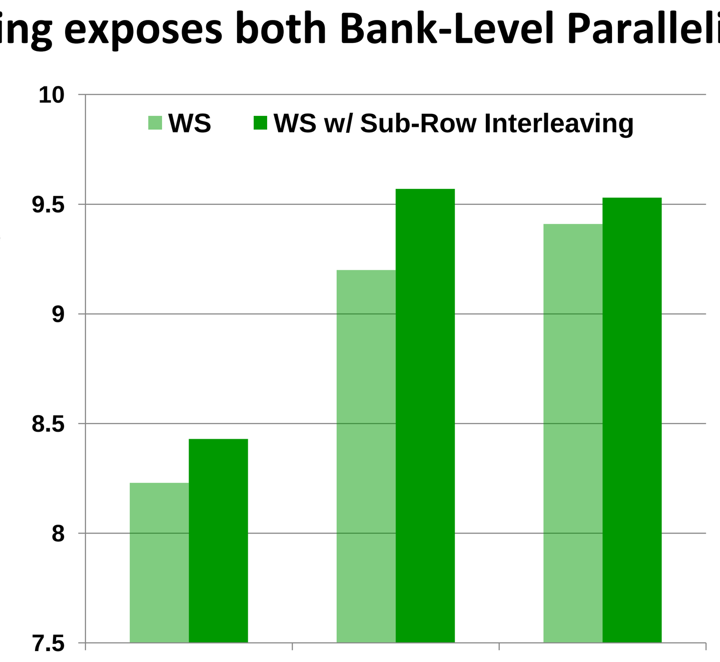
- **Try to find the knee of the curve**
  - Provides best performance fairness trade-off

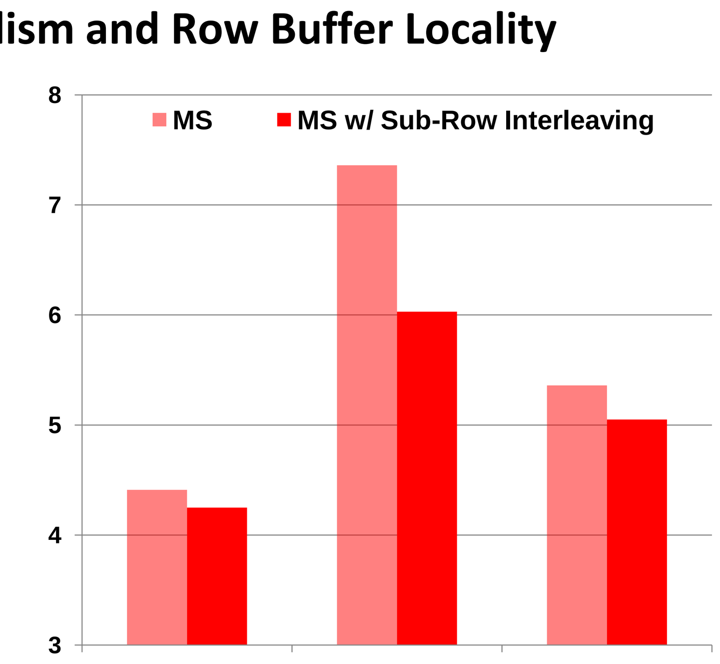
- **Dynamic clustering outperforms previous algorithms**
  - Guarantees sufficient bandwidth for "light" threads
  - Shares remaining bandwidth among "heavy" threads
- **Example workload with 10 memory-intensive threads**
  - FHM offers best performance & good fairness

3 Sub-Row Interleaving

- **"Sub-Row Interleaving" memory mapping exposes both Bank-Level Parallelism and Row Buffer Locality**
  - Switches banks every 4 cache-blocks
- **Improves Performance and Fairness**
  - For all memory scheduling algorithms

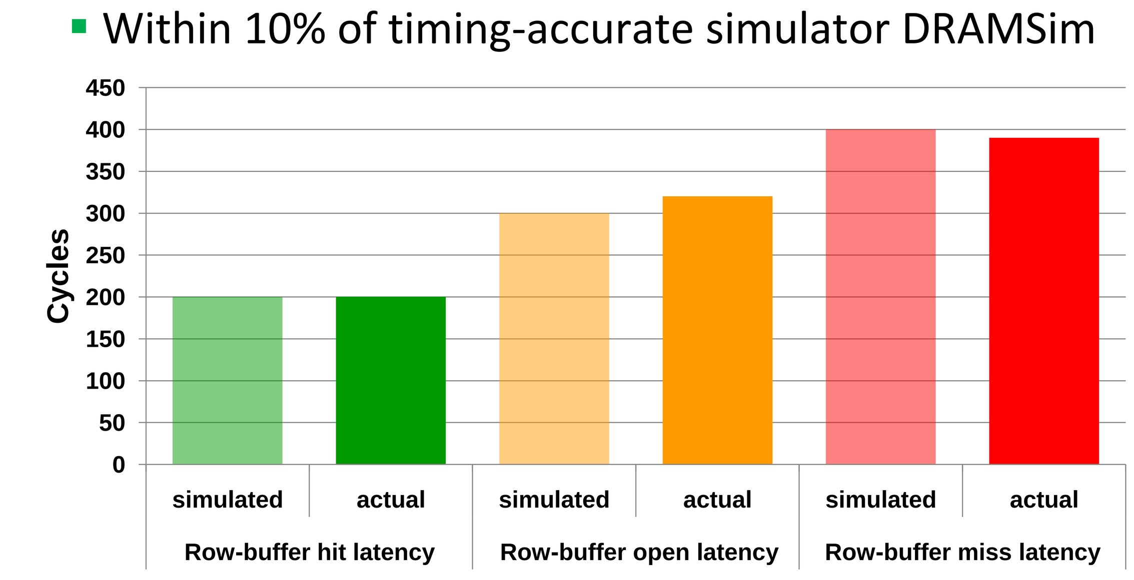






4 Validating DRAM Timing

- **Validated DRAM timing against cycle-accurate model**
  - DRAM: DDR2-667 / CPU: 3.33 GHz
  - Within 10% of timing-accurate simulator DRAMSim



All results are for a 16-core CMP running multi-programmed SPEC-2006 workload mixes. Memory subsystem has 2 channels with 4 banks per channel.