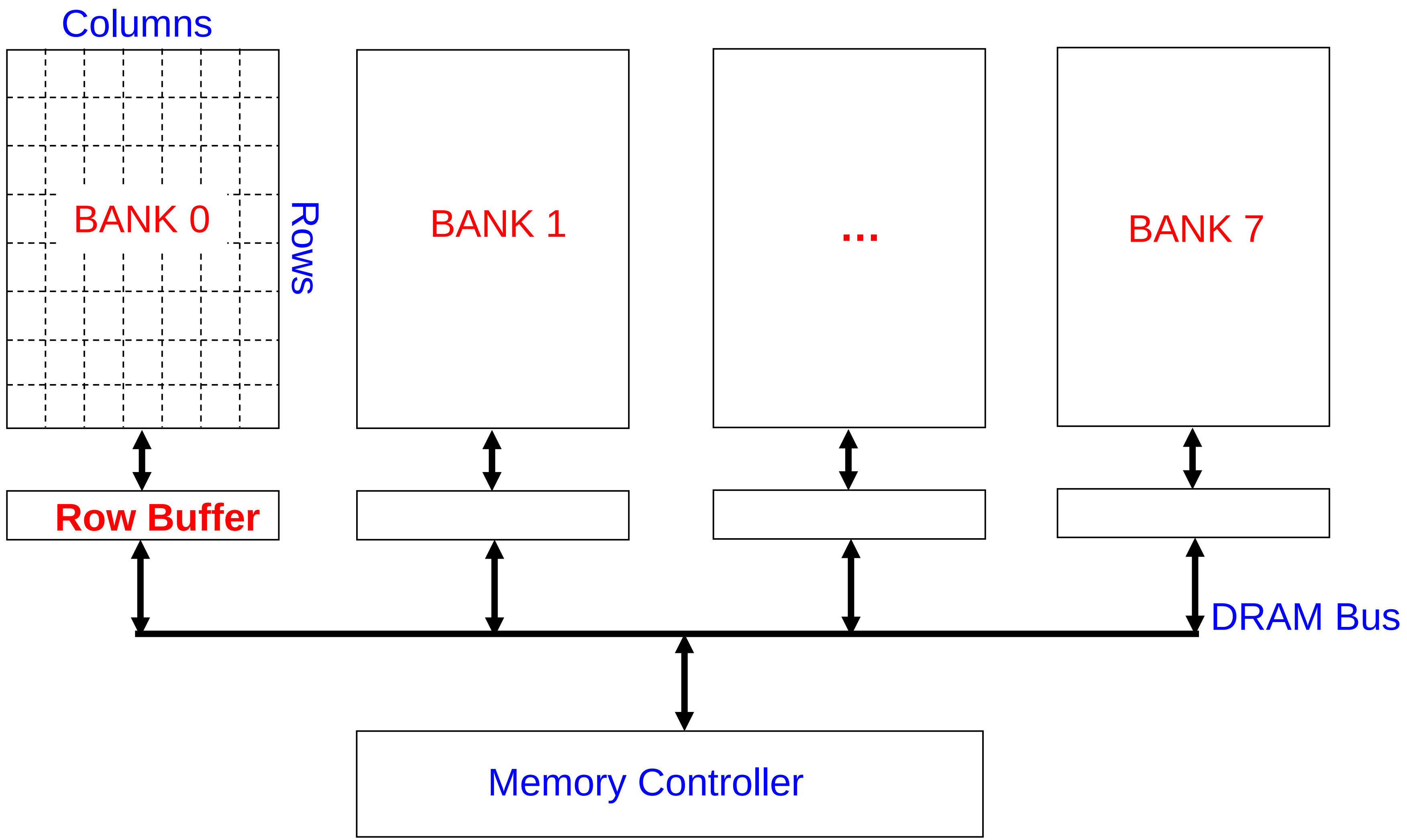
Locality-Aware Bank Partitioning for Shared DRAM MPSoCs

Yangguo Liu, Junlin Lu, Dong Tong and Xu Cheng
Microprocessor Research and Development Center
Peking University

Outline

- Background and Prior Work
- Motivation
  - Bank Level Parallelism
- Locality-aware Bank Partitioning
  - Partitioning
  - Integrating Bandwidth and Bank Partitioning
- Evaluation
- Summary

DRAM Memory System

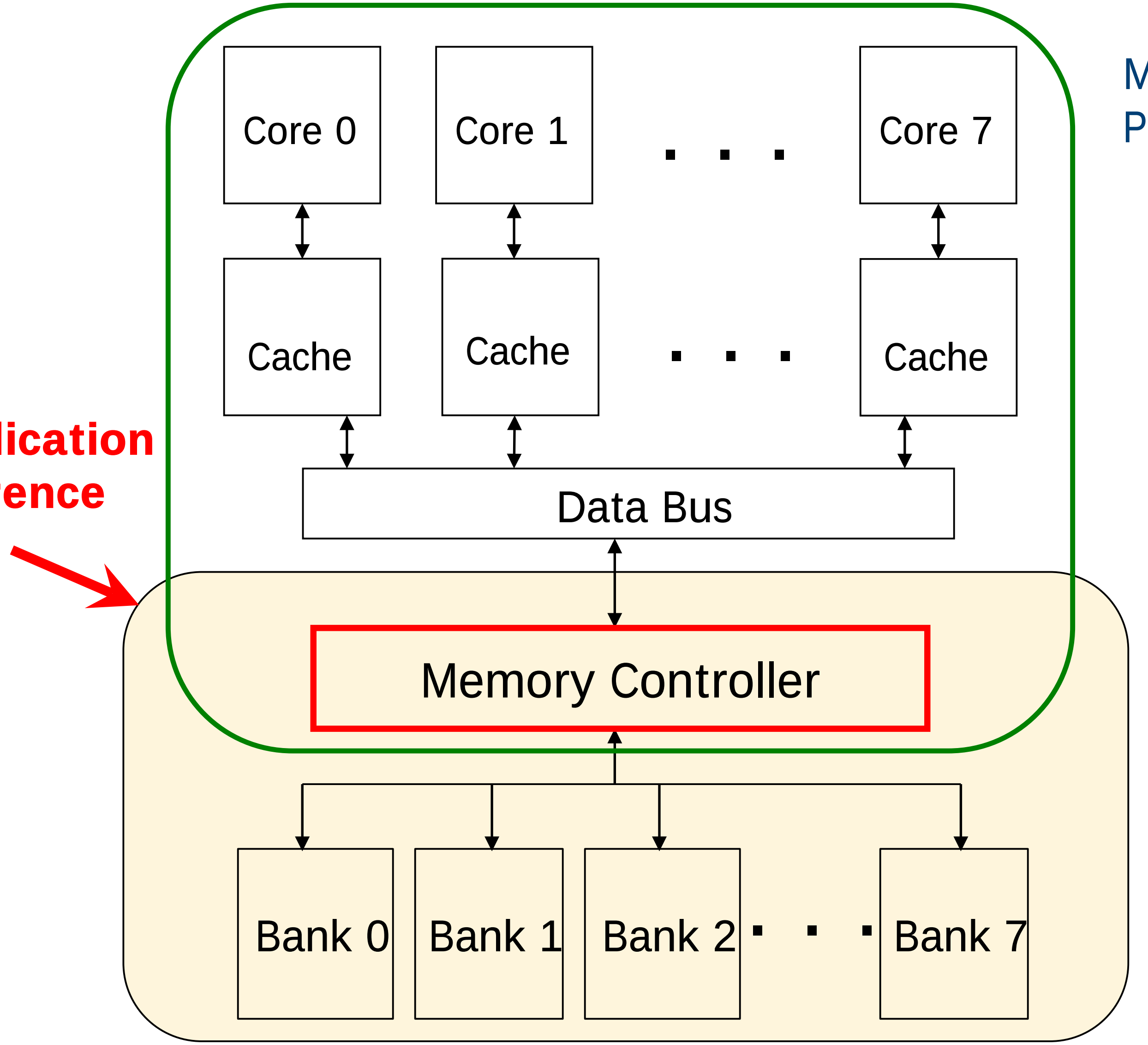


DRAM Memory System

- Row Buffer
  - Provides temporary data storage of a DRAM row
  - Row-buffer hit/conflict
    - Hit: Column access
    - Conflict: Precharge → Activate → Column access
  - Row buffer locality

- Multi-Bank
  - Multiple banks allow multiple memory requests to proceed in parallel
  - Bank level parallelism: overlap/hide memory access latency

Shared DRAM MPSoC Systems



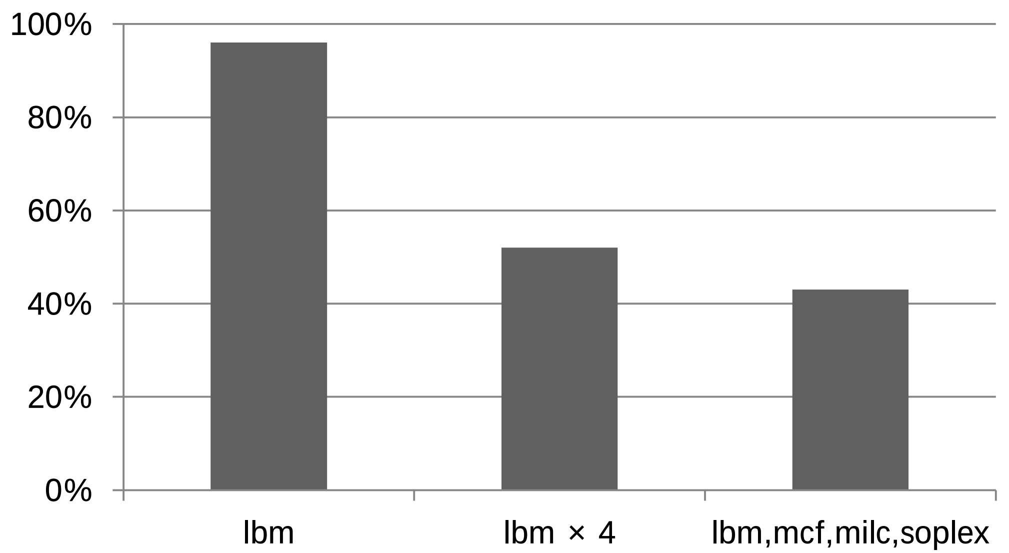
Multi-core Processor

inter-application interference

Shared DRAM Memory System

Inter-application Interference in DRAM System

- Inter-application Interference
  - Memory streams of different applications are interleaved and interfere with each other at DRAM memory
  - Destroy original row-buffer locality of individual applications
  - High row-buffer conflict rate
  - High memory access latency



Prior Work

- Out-of-order Memory scheduling
  - Reorder memory requests to recover original row-buffer locality
  - Recovering ability is restricted
    - Arrival interval between memory requests of individual applications
    - Scheduling buffer size
  - Cannot eliminate inter-application interference

- Bank partitioning
  - Divide memory banks among cores
  - Isolate memory access streams of different applications
  - Eliminate inter-application interference

Prior Work

- Bank partitioning
  - OS-based page coloring
  - Assign different page colors to different cores
  - Eliminate inter-application interference

  - Drawback
    - Previous bank partitioning only addresses the interference among memory-intensive applications but ignores the interference caused by memory non-intensive applications.

Outline

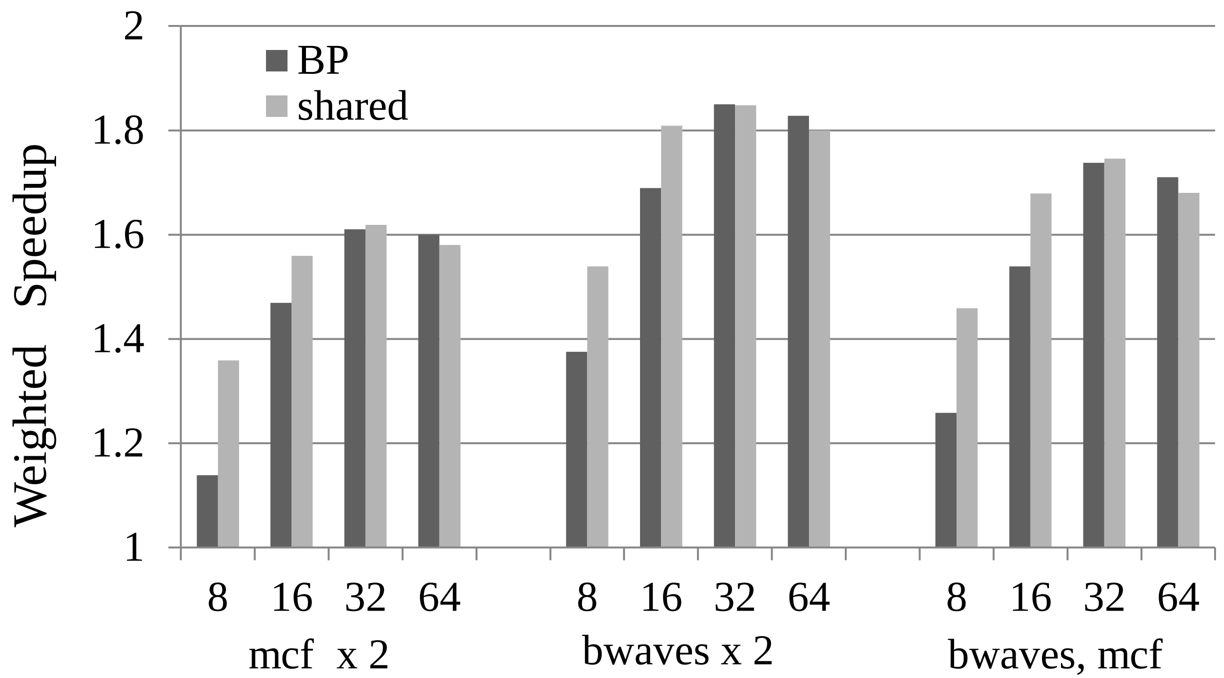
- Background and Prior Work
- Motivation
  - Bank Level Parallelism
- Dynamic Bank Partitioning
  - Partitioning
  - Integrating Bandwidth and Bank Partitioning
- Evaluation
- Summary

Bank Level Parallelism

- Memory non-intensive applications are not sensitive to bank amounts
- Memory intensive applications are sensitive to bank amounts
  - High row-buffer locality
  - Low row-buffer locality
  - Most applications give peak performance at 8 or 16 banks

BLP-sensitive

Trade Off between Parallelism and Locality



- Bank Partitioning
  - Reserve row-buffer locality
- shared
  - Improve bank level parallelism at the cost of interference
- Low row-buffer locality applications: Improving BLP brings more benefits than eliminating interference

Outline

- Background and Prior Work
- Motivation
  - Bank Level Parallelism
- Locality-aware Bank Partitioning
  - Partitioning
  - Integrating Bandwidth and Bank Partitioning
- Evaluation
- Summary

Locality-aware Bank Partitioning

- Step1: Profiling applications’ memory characteristics
  - Memory intensity
    - MAPI — memory access per interval
  - Row-buffer locality
    - RBH — row-buffer hit rate

- Step2: Grouping applications
  - Memory non-intensive cluster (NI cluster)
  - Memory intensive cluster (MI cluster)
    - Low row-buffer locality (L-RBL cluster)
    - High row-buffer locality (H-RBL cluster)

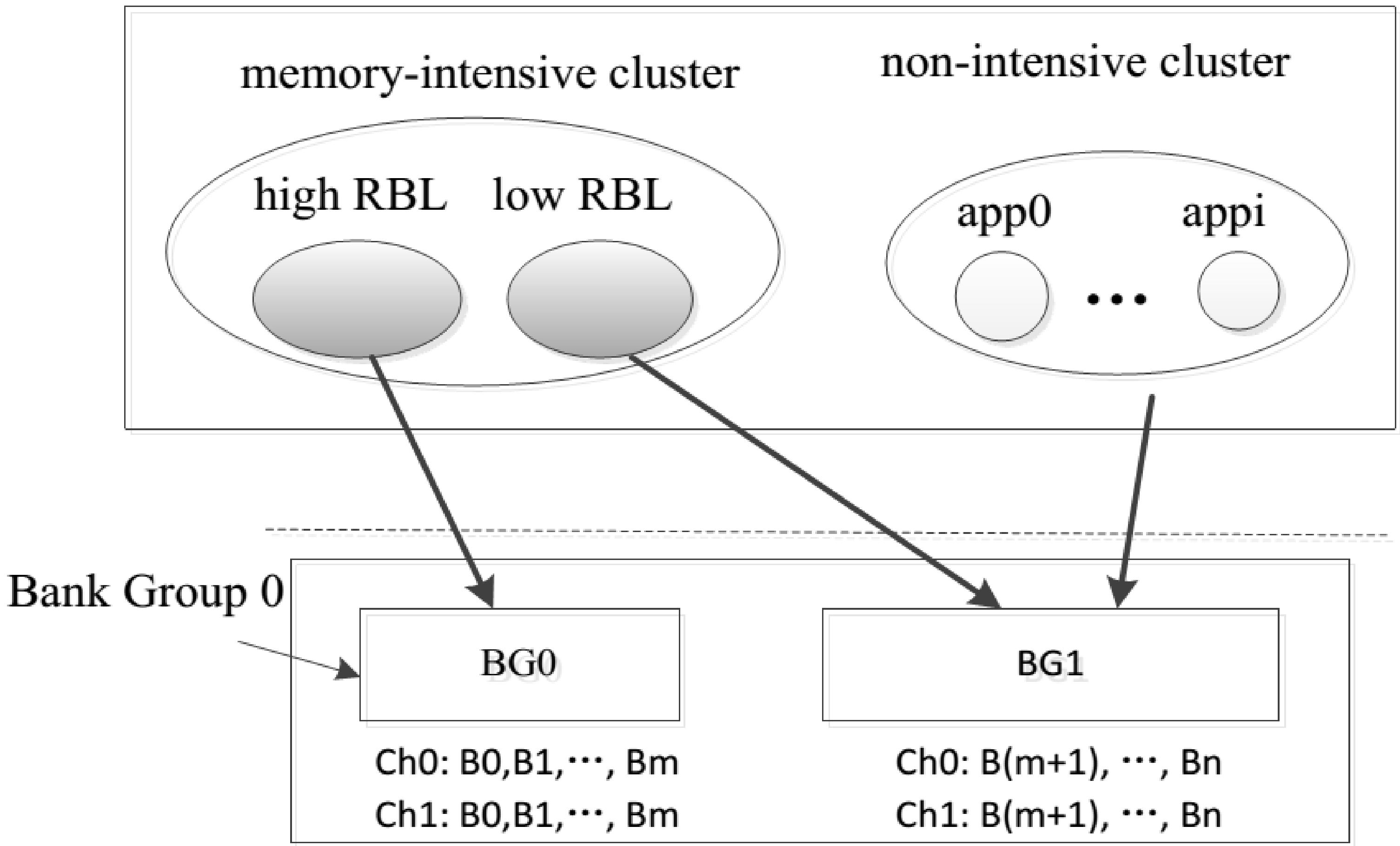
- Step3: Locality-aware bank partitioning

Locality-aware Bank Partitioning

- Step1: Profiling applications' memory characteristics
- Step2: Grouping applications
- Step3: Locality-aware bank partitioning
  - Objectives
    - Isolate the memory access streams of H-RBL applications from other MI applications
    - Improve BLP for BLP-sensitive applications
  - Definition: Minimum Partitioning Unit

$$MPU = \begin{cases} (\mathrm{N}_{\mathrm{rank}} \cdot \mathrm{N}_{\mathrm{bank}})/(\mathrm{N}_{\mathrm{core}} \cdot \mathrm{MI}), & \mathrm{MI} \leq 100\% \\ (\mathrm{N}_{\mathrm{rank}} \cdot \mathrm{N}_{\mathrm{bank}})/\mathrm{N}_{\mathrm{core}}, & \mathrm{MI} = 0 \end{cases}$$

Locality-aware Bank Partitioning Algorithm



Locality-aware Bank Partitioning Algorithm

- NI group
  - Do not allocate dedicate colors, but share banks with L-RBL group
  - Save memory banks for BLP-sensitive applications

- H-RBL group
  - Allocate *MPU* banks to each core
  - Isolate H-RBL applications from other memory intensive applications

- L-RBL group
  - MPU < 16
    - Each two core shares *MPU* · 2 colors
    - Improve BLP for low row-buffer locality applications
  - MPU >= 16
    - Allocate *MPU* colors to each core

Integrating Bandwidth and Bank Partitioning

- Bank Partitioning
  - Divide memory banks among applications
  - Resolve inter-application interference
  - Don’t consider system throughput and fairness

- Bandwidth Throttling
  - Consider applications’ memory access behavior and system fairness
    - Prioritizing light applications can improve system throughput

      [Kim et al., HPCA 2010, Muralidhara et al., MICRO 2011, Zheng et al., ICPP 2008]
  - Improve system throughput and fairness
  - Can’t eliminate inter-application interference

Integrating Bandwidth and Bank Partitioning

- Locality-aware Bank Partitioning
  - Dynamically divide memory banks among applications
  - Mitigate inter-application interference
  - Improve bank level parallelism

- Bandwidth Throttling
  - Proportionally throttle MI applications
  - Applications in NI cluster are relatively prioritized

IBBP(Integrated Bandwith throttling & Bank Partitioning)
Mitigate interference, improve system performance

Outline

- Background and Prior Work
- Motivation
  - Bank Level Parallelism
- Locality-aware Bank Partitioning
  - Partitioning
  - Integrating Bandwidth and Bank Partitioning
- Evaluation
- Summary

Evaluation Methodology

- Simulation Model
  - 8 cores, 2 channels, 2 ranks/channel, 8 banks/rank
  - Core Model
    - Out-of-order
    - 32KB Inst/32KB Data, 4-way, 64B line, LRU
    - 8MB shared L2 cache, 16-way, 64B line
  - DRAMSim2 – DDR3 [P. Rosenfeld et al., CAL 2011]
- Workloads
  - SPEC CPU 2006 multi-programmed workloads (categorized based on memory intensity)
- Metrics

$$Weighted\ Speedup = \sum_i \frac{IPC_i^{shared}}{IPC_i^{alone}} \qquad MaximumSlowdown = \max_i \frac{IPC_i^{alone}}{IPC_i^{shared}}$$

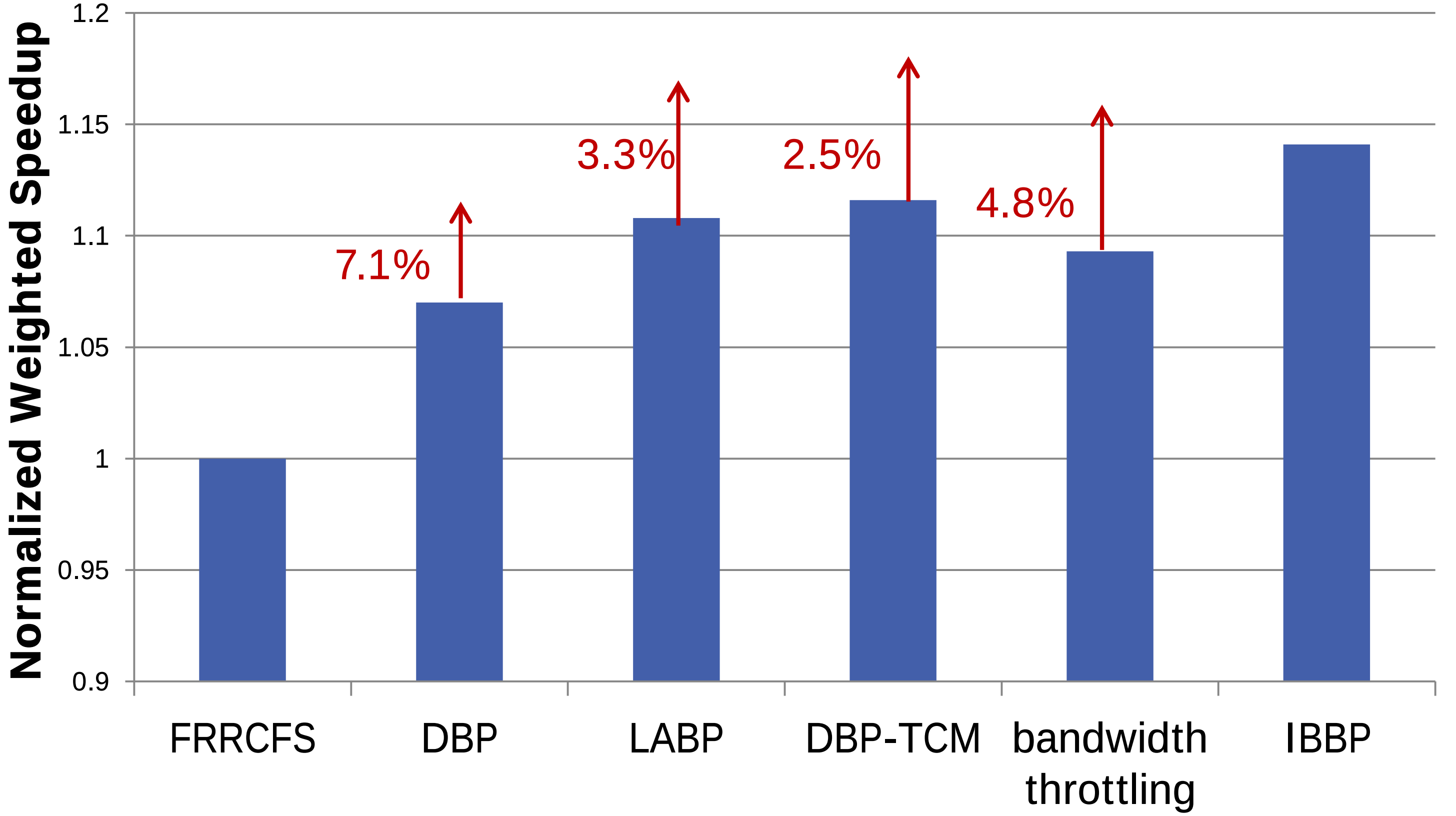
Comparison with Other Methods

- Baseline FR-FCFS [Rixner et al., ISCA 2000]
  - Prioritizes row-hit requests, older requests

- DBP [mingli xie et al., HPCA 2014]

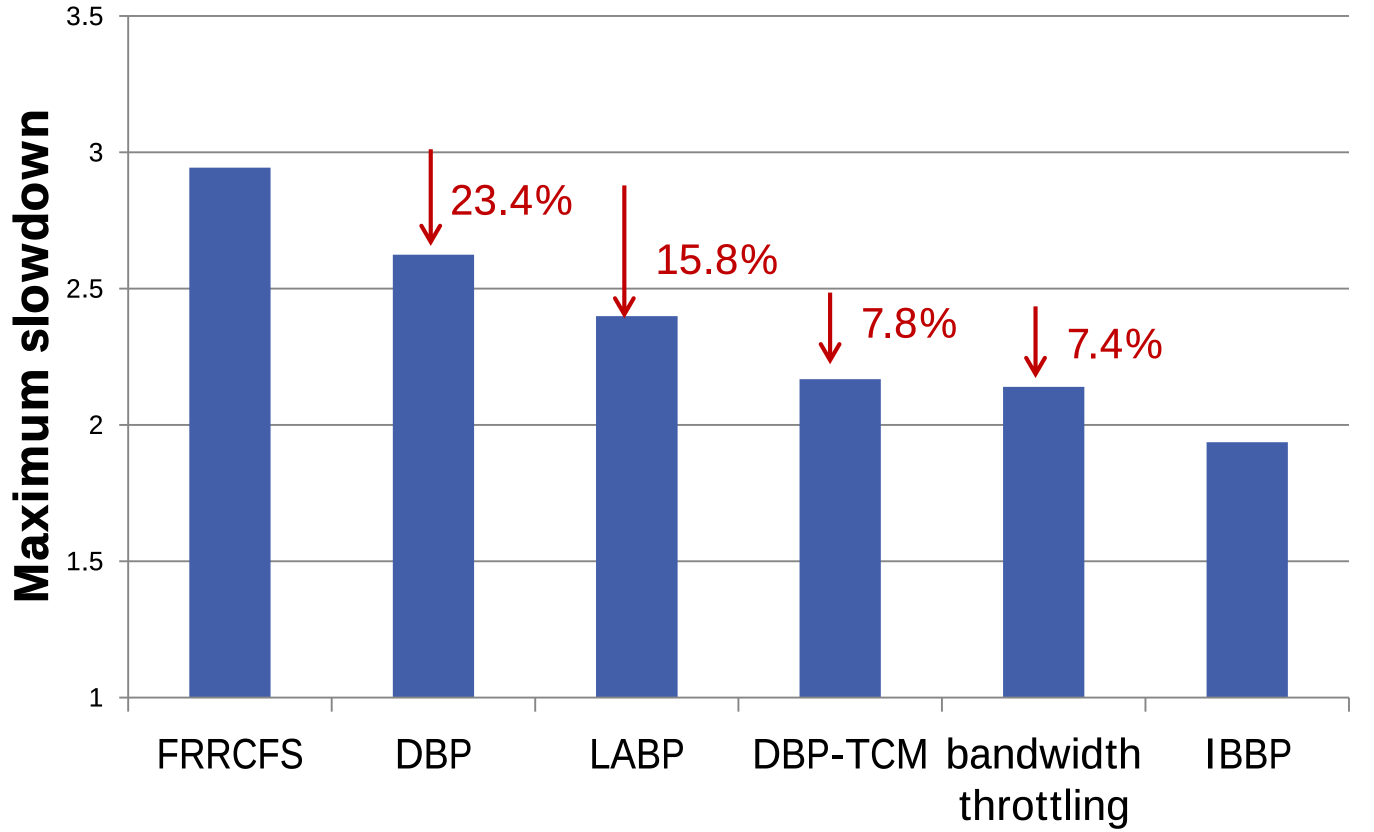
- DBP-TCM
  - Integrating Dynamic Bank Partitioning with Thread Cluster Memory scheduling [mingli xie et al., HPCA 2014]

System Throughput



Integrating LABP and bandwidth throttling provides better system throughput than either LABP or bandwidth throttling alone

System Fairness



Integrating LABP and bandwidth throttling provides better system fairness than either LABP or bandwidth throttling alone

Summary

- Inter-application interference in DRAM memory of MPSoC systems degrades system performance

- Locality-aware Bank Partitioning
  - Resolve interference
  - Improve intra-applications' bank level parallelism

- Integrating Bank partitioning with bandwidth throttling
  - Dynamic Bank partitioning
    - Resolves interference
  - Bandwidth throttling
    - Proportionally throttle MI applications
    - Applications in NI cluster are relatively prioritized

Thank you. Questions?