Reducing Memory Interference in Multicore Systems via Application-Aware Memory Channel Partitioning
Brief Paper Recap
Applications’ requests interfere at the main memory

This inter-application interference degrades system performance

Problem is further exacerbated by
- Increasing number of cores
- Limited off-chip pin bandwidth
Data Mapping in Current Systems

Causes interference between applications’ requests
Partitioning Channels Between Applications

Eliminates interference between applications’ requests
Overview: Memory Channel Partitioning (MCP)

- **Goal**
  - Eliminate harmful interference between applications

- **Basic Idea**
  - Map the data of **badly-interfering applications** to different channels

- **Key Principles**
  - Separate **low and high memory-intensity applications**
  - Separate **low and high row-buffer locality applications**
# Memory Channel Partitioning (MCP) Mechanism

1. **Profile** applications
2. **Classify** applications into groups
3. **Partition channels** between application groups
4. **Assign a preferred channel** to each application
5. **Allocate application pages** to preferred channel

---

**Hardware**

**System Software**
Interval Based Operation

Current Interval

1. Profile applications
2. Classify applications into groups
3. Partition channels between groups
4. Assign preferred channel to applications

Next Interval

5. Enforce channel preferences

time
Integrating Partitioning and Scheduling

Goal:
Mitigate Inter-Application Interference

Previous Approach:
Application-Aware Memory Request Scheduling

Our First Approach:
Application-Aware Memory Channel Partitioning

Our Second Approach:
Integrated Memory Partitioning and Scheduling
Hardware Costs of the Two Approaches

1. Memory Channel Partitioning (MCP)
   - Only profiling counters in hardware
   - No modifications to memory scheduling logic
   - 1.5 KB storage cost for a 24-core, 4-channel system

2. Integrated Memory Partitioning and Scheduling (IMPS)
   - A single bit per request
   - Scheduler prioritizes based on this single bit
Comparison to Previous Scheduling Policies

Averaged over 240 workloads

Normalized System Performance

Better system performance than the best previous scheduler at lower hardware cost

Significant performance improvement over baseline FRFCFS
Interaction with Memory Scheduling

IMPS improves performance regardless of scheduling policy
Highest improvement over FRFCFS as IMPS designed for FRFCFS
Summary

- Uncontrolled inter-application interference in main memory degrades system performance

- **Application-aware memory channel partitioning (MCP)**
  - Separates the data of badly-interfering applications to different channels, eliminating interference

- **Integrated memory partitioning and scheduling (IMPS)**
  - Prioritizes very low memory-intensity applications in scheduler
  - Handles other applications’ interference by partitioning

- MCP/IMPS provide better performance than application-aware memory request scheduling at lower hardware cost
Strengths
Strengths (1/2)

- Novel solution to a key problem in multi-core systems
  - Memory interference
  - The importance of problem will increase over time

- Keeps the memory scheduling hardware simple

- Enables HW and SW components to work cooperatively where each works best

- Combines multiple interference reduction techniques
Strengths (2/2)

- Can provide performance isolation across applications mapped to different channels

- General idea of partitioning can be extended to smaller granularities in the memory hierarchy: banks, subarrays, etc.

- Well-written paper

- Thorough simulation-based evaluation
Weaknesses
Weaknesses

- Overhead of moving pages between channels restricts mechanism’s benefits
- Load imbalance across channels can reduce performance
  - The paper addresses this and compares to another mechanism
- Software-hardware cooperative solution might not always be easy to adopt
- Evaluation is done solely in simulation
- Evaluation does not consider multi-chip systems
- Are these the best workloads to evaluate?
Recall: Try to Avoid Rat Holes

Limitations of the Mechanism

- Mechanism may not work effectively if workload changes behavior after profiling
- Small number of memory channels reduces the scope of partitioning
- Adds restrictions on physical-to-DRAM address mapping
Thoughts and Ideas
Extensions

- Can this idea be extended to different granularities in memory?
  - Partition banks, subarrays, mats across workloads

- Can this idea be extended to provide performance predictability and performance isolation? How?

- How can MCP be combined effectively with other interference reduction techniques?
  - E.g., source throttling methods [Ebrahimi+, ASPLOS 2010]
  - E.g., thread scheduling methods

- Can this idea be evaluated on a real system? How?
Fairness via Source Throttling

- Eiman Ebrahimi, Chang Joo Lee, Onur Mutlu, Yale N. Patt, "Fairness via Source Throttling: A Configurable and High-Performance Fairness Substrate for Multi-Core Memory Systems"


Fairness via Source Throttling: A Configurable and High-Performance Fairness Substrate for Multi-Core Memory Systems

Eiman Ebrahimi†  Chang Joo Lee†  Onur Mutlu§  Yale N. Patt†

†Department of Electrical and Computer Engineering
The University of Texas at Austin
{ebrahimi, cjlee, patt}@ece.utexas.edu

§Computer Architecture Laboratory (CALCM)
Carnegie Mellon University
onur@cmu.edu
Bank Partitioning

- Liu Liu, Zehan Cui, Mingjie Xing, Yungang Bao, Mingyu Chen, Chengyong Wu,


A Software Memory Partition Approach for Eliminating Bank-level Interference in Multicore Systems

Lei Liu$^{1,2}$, Zehan Cui$^{81,2}$, Mingjie Xing$^1$, Yungang Bao$^1$, Mingyu Chen$^1$, Chengyong Wu$^1$

$^1$State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences

$^2$Graduate School of Chinese Academy of Sciences

Beijing, China

{liulei2010, cuizehan, baoyg, cmy, cwu}@ict.ac.cn
Bank Partitioning

- Min Kyu Jeong, Doe Hyun Yoon, Dam Sunwoo, Mike Sullivan, Ikhwan Lee, Mattan Erez,

Balancing DRAM Locality and Parallelism in Shared Memory CMP Systems

Min Kyu Jeong*, Doe Hyun Yoon†, Dam Sunwoo‡, Mike Sullivan*, Ikhwan Lee*, and Mattan Erez*

* Dept. of Electrical and Computer Engineering, The University of Texas at Austin
† Intelligent Infrastructure Lab, Hewlett-Packard Labs
‡ ARM Inc.
Takeaways
Key Takeaways

- A novel method to reduce memory interference
- Simple and effective
- Hardware/software cooperative
- Good potential for work building on it to extend it
  - To different structures
  - To different metrics
  - Multiple works have already built on the paper (see bank partitioning works in PACT 2012, HPCA 2012)
- Easy to read and understand paper
Open Discussion
Discussion Starters

- Thoughts on the previous ideas?
- How practical is this?
- Will the problem become bigger and more important over time?
- Will the solution become more important over time?
- Are other solutions better?
- Is this solution clearly advantageous in some cases?