Interactions Between Compression and Prefetching in Chip Multiprocessors

Alaa R. Alameldeen*
Intel Corporation

David A. Wood
University of Wisconsin-Madison

*Work largely done while at UW-Madison

Overview

- Addressing memory wall in CMPs
- Compression
  - Cache Compression: Increases effective cache size
    - Increases latency, uses additional cache tags
  - Link Compression: Increases effective pin bandwidth
- Hardware Prefetching
  - \+ Decreases average cache latency?
  - – Increases bandwidth demand ⇒ contention
  - – Increases working set size ⇒ cache pollution
- Contribution #1: Adaptive Prefetching
  - Uses additional tags (like cache compression)
  - Avoids useless and harmful prefetches
- Contribution #2: Prefetching&compression interact positively
  - e.g., Link compression helps prefetching's bandwidth demand

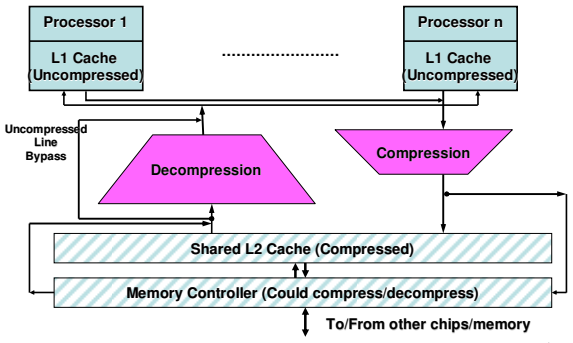
2

Outline

- Overview
- Compressed CMP Design
  - – Compressed Cache Hierarchy
  - – Link Compression
- Adaptive Prefetching
- Evaluation
- Conclusions

3

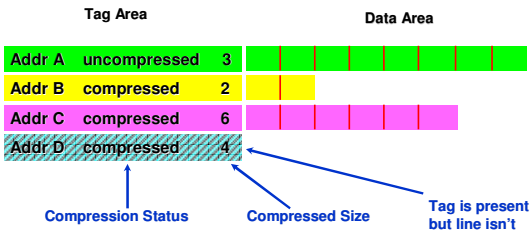
Compressed Cache Hierarchy



4

Cache Compression: Decoupled Variable-Segment Cache (ISCA 2004)

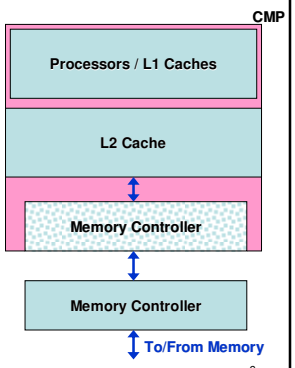
- Example cache set



5

Link Compression

- On-chip Memory Controller transfers compressed messages
- Data Messages
  - – 1-8 sub-messages (flits), 8-bytes each
- Off-chip memory controller combines flits and stores to memory



6

Outline

- Overview
- Compressed CMP Design
- Adaptive Prefetching
- Evaluation
- Conclusions

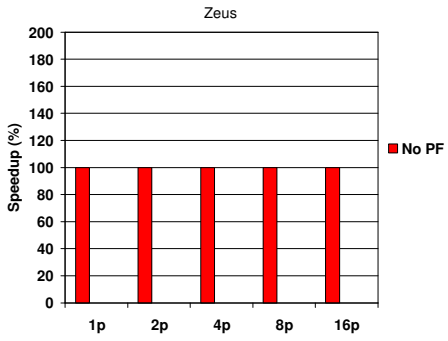
7

Hardware Stride-Based Prefetching

- Evaluation based on IBM Power 4 implementation
- Stride-based prefetching
  - + L1 prefetching hides L2 latency
  - + L2 prefetching hides memory latency
  - - Increases working set size ⇒ cache pollution
  - - Increases on-chip & off-chip bandwidth demand ⇒ contention
- Prefetching Taxonomy [Srinivasan et al. 2004]:
  - Useless prefetches increase traffic not misses
  - Harmful prefetches replace useful cache lines

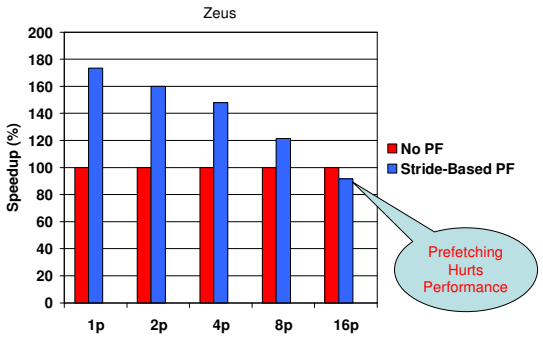
8

Prefetching in CMPs



9

Prefetching in CMPs



➢ Can we avoid useless & harmful prefetches?

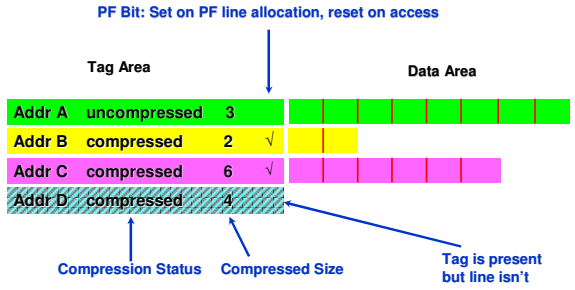
10

Prefetching in CMPs

➢Adaptive prefetching avoids hurting performance

11

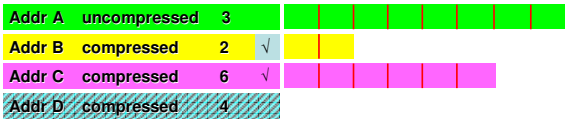
Adaptive Prefetching



➢ How do we identify useful, useless, harmful prefetches?

12

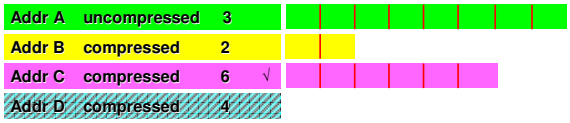
Useful Prefetch



- Read/Write Address B
  - PF bit set ⇒ PF used before replacement
  - Access resets PF bit

13

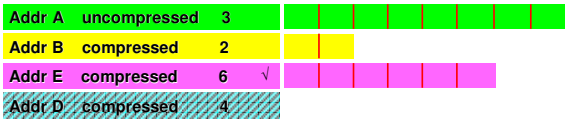
Useless Prefetch



- Replace Address C with Address E
  - PF bit set ⇒ PF unused before replacement
  - PF did not change #misses, only increased traffic

14

Harmful Prefetch



- Read/Write Address D
  - Tag found ⇒ Could have been a hit
  - Heuristic: If PF bit of other line set ⇒ Avoidable miss
  - Prefetch(es) incurred an extra miss

15

Adaptive Prefetching : Summary

- Uses extra compression tags & PF bit per tag
  - PF bit set when prefetched line is allocated, reset on access
  - Extra tags still maintain address tag, status
- Use counter for #startup prefetches per cache: 0 – MAX
  - Increment for useful prefetches
  - Decrement for useless & harmful prefetches
- Classification of Prefetches:
  - Useful Prefetch: PF bit set on a cache hit
  - Useless Prefetch: Replacing a line whose PF bit is set
  - Harmful Prefetch: Cache miss matches invalid line's tag, valid line tag has PF bit set

16

Outline

- Overview
- Compressed CMP Design
- Adaptive Prefetching
- Evaluation
  - Workloads and System Configuration
  - Performance
  - Compression and Prefetching Interactions
- Conclusions

17

Simulation Setup

- Workloads:
  - Commercial workloads [Computer'03, CAECW'02] :
    - OLTP: IBM DB2 running a TPC-C like workload
    - SPECJBB
    - Static Web serving: Apache and Zeus
  - SPEComp benchmarks:
    - apsi, art, fma3d, mgrid
- Simulator:
  - Simics full system simulator; augmented with:
  - Multifacet General Execution-driven Multiprocessor Simulator (GEMS) [Martin, et al., 2005, http://www.cs.wisc.edu/gems/]

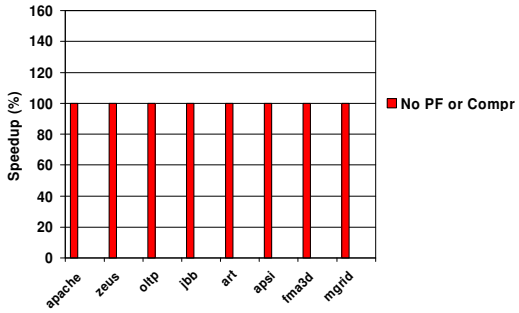
18

System Configuration

- *8-core* CMP
- **Cores**: single-threaded, out-of-order superscalar with an 11-stage pipeline, 64 entry IW, 128 entry ROB, 5 GHz clock frequency
- **L1 Caches**: 64K instruction, 64K data, 4-way SA, 320 GB/sec total on-chip bandwidth (to/from L1), 3-cycle latency
- **Shared L2 Cache**: 4 MB, 8-way SA (uncompressed), 15-cycle uncompressed latency
- **Memory**: 400 cycles access latency, 20 GB/sec memory bandwidth
- Prefetching
  - 8 unit/negative/non-unit stride streams for L1 and L2 for each processor
  - Issue 6 L1 prefetches (max) on L1 miss
  - Issue 25 L2 prefetches (max) on L2 miss

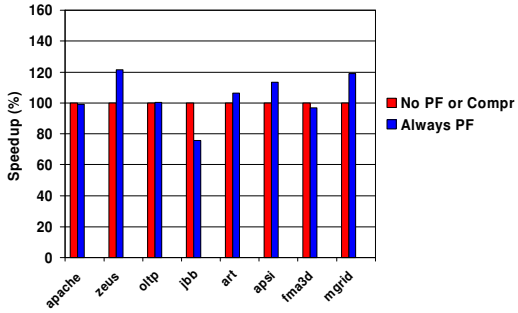
19

Performance



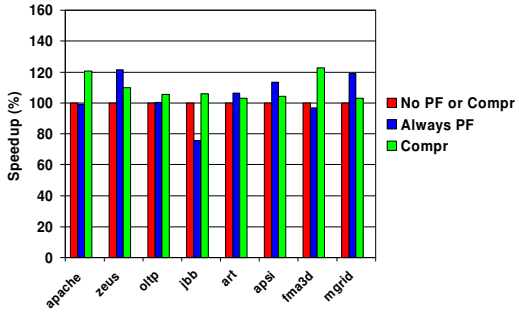
20

Performance



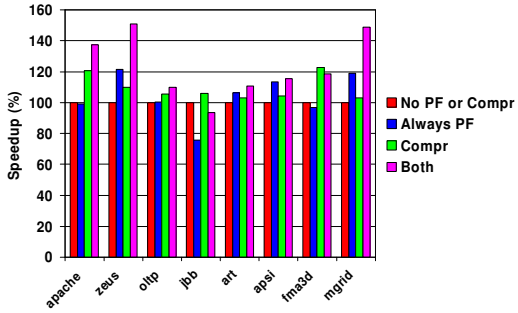
21

Performance



22

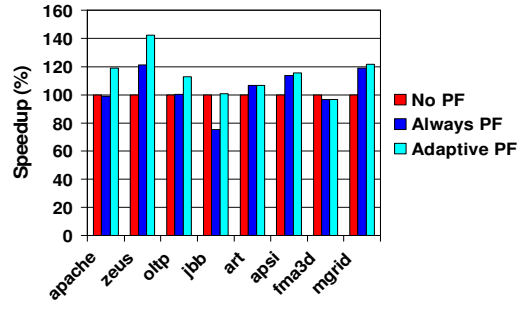
Performance



➢Significant Improvements when Combining Prefetching and Compression

23

Performance: Adaptive Prefetching



➢Adaptive PF avoids significant performance losses

24

Compression and Prefetching Interactions

- **Positive Interactions:**
  - L1 prefetching hides part of decompression overhead
  - Link compression reduces increased bandwidth demand because of prefetching
  - Cache compression increases effective L2 size, L2 prefetching increases working set size
- **Negative Interactions:**
  - L2 prefetching and L2 compression can eliminate the same misses

➲ Is the overall interaction positive or negative?

25

Interactions Terminology

- Assume a base system S with two architectural enhancements A and B, All systems run program P

Speedup(A) = Runtime(P, S) / Runtime(P, A)

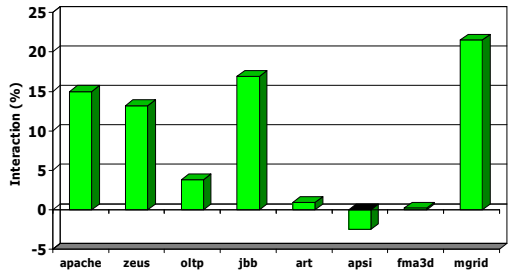
Speedup(B) = Runtime(P, S) / Runtime (P, B)

Speedup(A, B) = Speedup(A) x Speedup(B)

x (1 + Interaction(A,B))

26

Interaction Between Prefetching and Compression



➲ Interaction is positive for all benchmarks except apsi

27

Conclusions

- Compression can help CMPs
  - Cache compression improves CMP performance
  - Link Compression reduces pin bandwidth demand
- Hardware prefetching can hurt performance
  - Increases cache pollution
  - Increases bandwidth demand
- Contribution #1: Adaptive prefetching reduces useless & harmful prefetches
  - Gets most of benefit from prefetching
  - Avoids large performance losses
- Contribution #2: Compression&prefetching interact positively

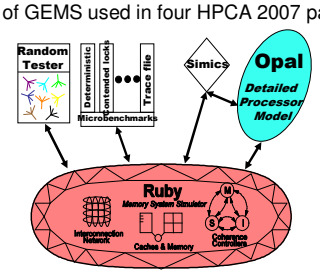
28

Google "Wisconsin GEMS"

Works w/ Simics (free to academics) ➔ commercial OS/apps

SPARC out-of-order, x86 in-order, CMPs, SMPs, & LogTM

GPL release of GEMS used in four HPCA 2007 papers



29

Backup Slides

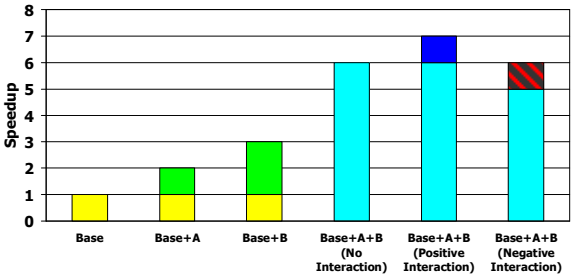
30

Frequent Pattern Compression (FPC)

- A significance-based compression algorithm
  - Compresses each 32-bit word separately
  - Suitable for short (32-256 byte) cache lines
  - Compressible Patterns: zeros, sign-ext. 4,8,16-bits, zero-padded half-word, two SE half-words, repeated byte
  - A 64-byte line is decompressed in a five-stage pipeline
- More details in Alaa's thesis

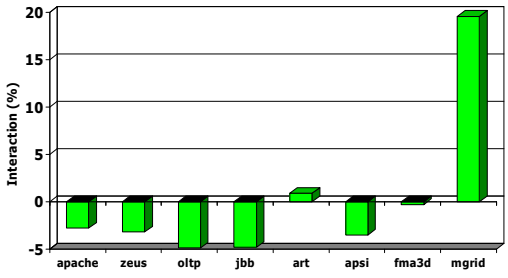
31

Positive and Negative Interactions



32

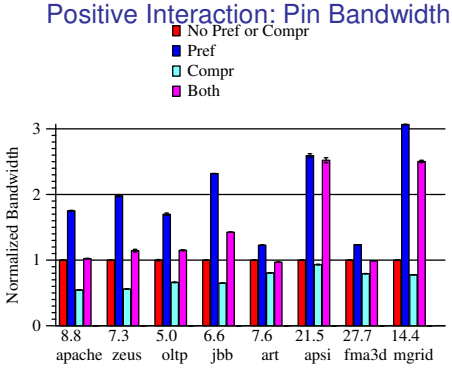
Interaction Between Adaptive Prefetching and Compression



➲ Less BW impact ⇒Interaction slightly -ve (except art, mgrid)

33

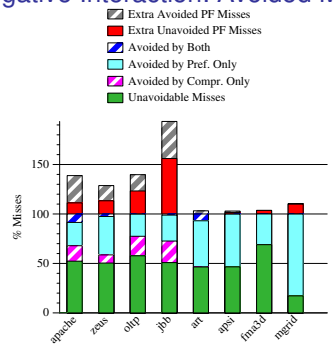
Positive Interaction: Pin Bandwidth



➲ Compression saves bandwidth consumed by prefetching

34

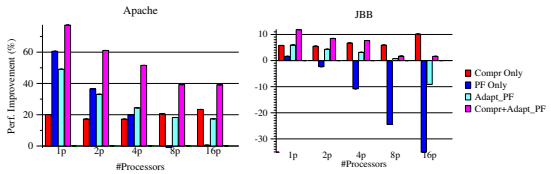
Negative Interaction: Avoided Misses



➲ Small Fraction of misses is avoided by both

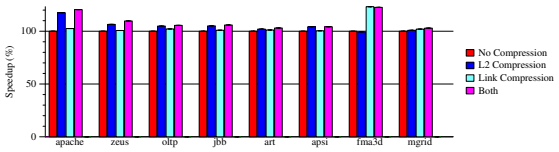
35

Sensitivity to Processor Count



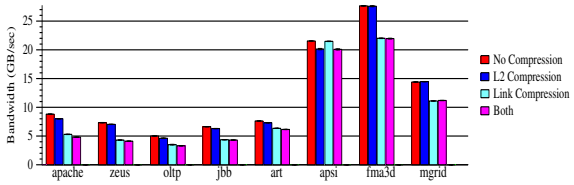
36

Compression Speedup



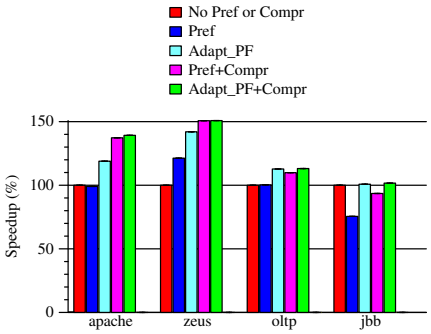
37

Compression Bandwidth Demand



38

Commercial Workloads & Adaptive Prefetching



39