Speculative Multiprocessor Cache Line Actions Using Instruction and Line History

David M. Koppelman
Department of Electrical & Computer Engineering
Louisiana State University, Baton Rouge

koppel@ee.lsu.edu http://www.ee.lsu.edu/koppel

2

Tradeoffs of Shared Memory Over Message Passing

Typical Shared Memory System

- Multiple processors access single address space.
- Physical memory distributed throughout system.
- Caches provide fast access.

Programmer concentrates on reading and writing data . . .

. . . but ignores data location.

Advantage:

Less work for programmer.

Disadvantage:

Data movement less efficient.

Movement initiated by time-consuming cache misses . . .

. . . when the data is already needed.

3

Second-Cache Miss

Some misses are more time consuming than others.

Second-Cache Miss:
A miss to one cache that initiates messages to a second cache.

Second-Cache Read Miss
Read miss to first cache, data exclusive in second cache.

Second-Cache Write Miss
Write miss to first cache, data to be invalidated in second cache(s).

4

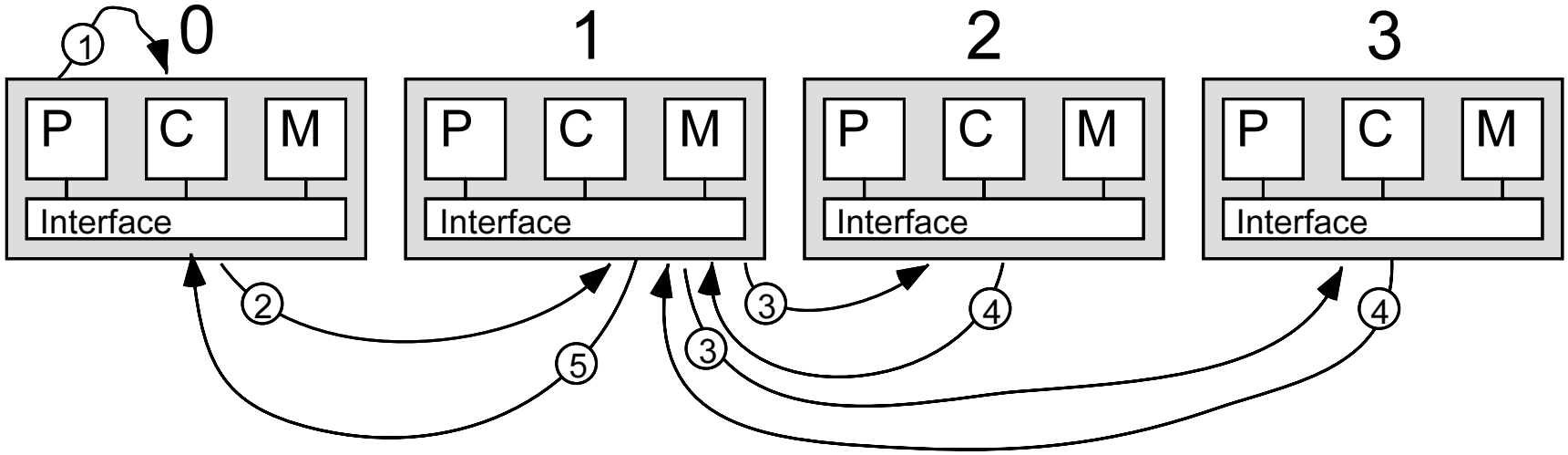
Second-Cache Write Miss Example

Consider a system with directory-based cache coherence.

Initially, block (data) shared in caches 2 and 3; memory 1 is home.

1 Processor 0 issues write which misses.
2 Cache 0 sends a write-request message to memory 1.
3 Memory 1 sends invalidate messages to caches 2 and 3.
4 Caches 2 and 3 acknowledge invalidation.
5 Memory 1 grants exclusive access to proc. 0

At finish, block exclusive in cache 0, not cached elsewhere.



Processor 0 must wait four message transit times.

PDCS-97 Formatted 15:49, 1 October 1997 from pdcs.

4

5

Reducing Second-Cache Miss Delays (Previous Work)

Software Prefetching

Requires programmer or compiler identification of prefetchable addresses.

Automatic Hardware Prefetching

$E.g.$, Dahlgren, Dubois, and Stenström.

Limited to constant-stride accesses.

Subject to learning delays.

Manual Hardware Prefetching and Stream Buffers

Limited to constant-stride accesses.

Requires programmer or compiler identification of prefetchable addresses.

6

Reducing Second-Cache Miss Delays (Previous Work)

Identification of Sharing Patterns (*Migratory* Data)

Cox and Fowler; Stenström, Brorsson, and Sandberg.

On read miss get exclusive copy if, based on history, write expected.

Subject to per-block learning delays.

Actions Based on Instruction History

Lilja.

Store instructions invalidate or update based upon execution history.

Uses compiler analysis.

7

Speculative Actions Using Instruction History

Speculative Actions

Speculatively invalidate line or revert line to shared . . .

. . . if that line might be accessed elsewhere.

Eliminates second-cache misses[1] . . .

. . . potentially halving miss latency.

Basis for Initiating Speculative Action

Assume those lines last-accessed by an instruction . . .
. . . will share the same fate.

E.g., suppose lines 100, 225, and 312 were last accessed by `st %r1,[%r2]`.

If line 100 gets invalidated assume 225 and 312 will soon be invalidated too.

[1] when done properly

8

Implementation of Speculative Actions

Each cache constructs a linked list for each instruction.

List holds lines accessed by the instruction.

Line moved from list to list each time it's accessed.

After a line is invalidated the corresponding linked list is traversed . . .

. . . and each member is invalidated . . .

. . . with a delay between invalidations (to avoid congestion).

Traversal stops when count maximum reached or list exhausted.

Exclusive lines that are reverted to shared are handled analogously.

9

Implementation Details

Instruction History Table (IHT)

Entry holds data about a memory-access instruction.

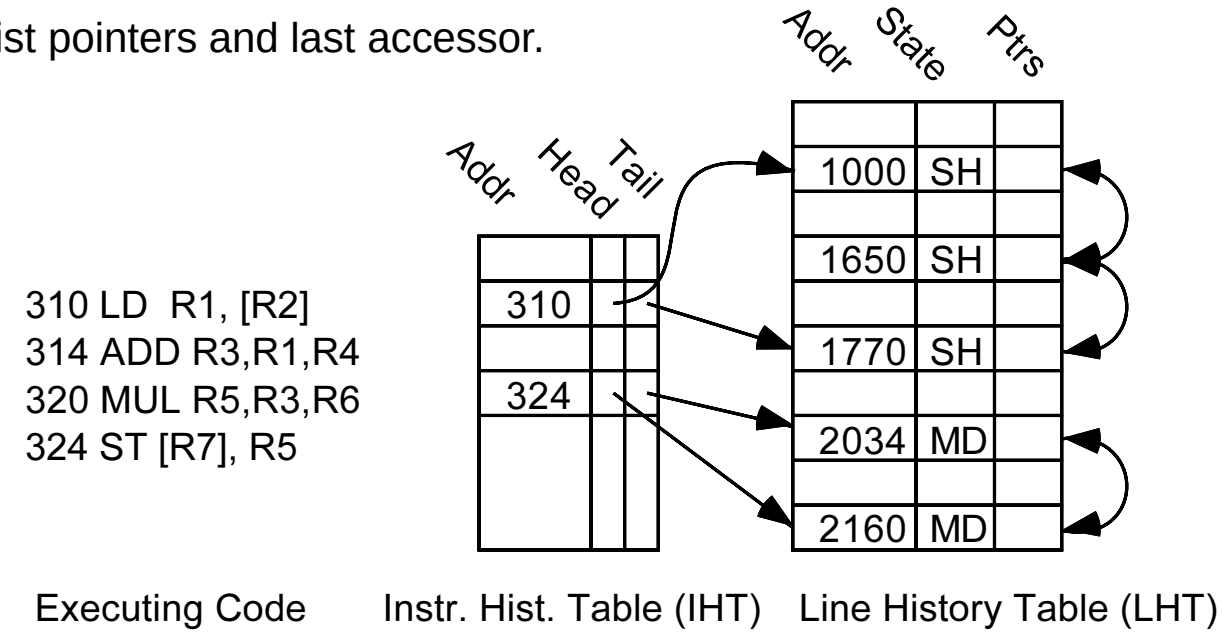
Holds list head and table.

Holds outcome data used for performance adjustment.

Line History Table

Entry holds data about a line.

Holds list pointers and last accessor.



PDCS-97 Formatted 15:49, 1 October 1997 from pdcs.

9

10

Cost and Hit Latency Impact

Cost

Assume cost dominated by storage.

Consider system using 16-byte lines.

Line history table adds 30% to cache size.

IHT entries are about 12 bytes each.

Number of entries independent of cache size.

For base configuration, 1000-entry IHT 5% of cache size.

Cost less than 50% cost of cache.

Impact on Cache Hit Latency

List changes need not use same storage as data.

Changes can be completed after access (don’t affect hit time).

11

Evaluation

Evaluated using execution-driven simulation

Simulated System (Basic)

- Cached virtual shared memory system, full-map directory-based cache coherence.
- Nonblocking writes, reads out of order with respect to writes.
- Eight-way, 2^{13} set, 16-byte line, physically mapped caches.
- One-cycle cache hit latency, ten-cycle memory latency.
- Two instruction-per-cycle effective CPU issue rate.
- Sixteen processors, mesh interconnect.

Simulator

- Proteus, version L3.11(.5).
- Execution-driven, on SPARC (Solaris 3.5) host.
- Simulation via assembly-level code augmentation.

12

Experiments

Workload

Four Splash 2 kernels: Radix, LU, FFT, Cholesky.

Four Splash 2 applications: FMM, Barnes, Ocean, Water N^2.

Goals

Determine absolute performance.

Determine sensitivity to:

- Cache Size
- Line Size
- Network Latency
- Memory Throughput (paper only)

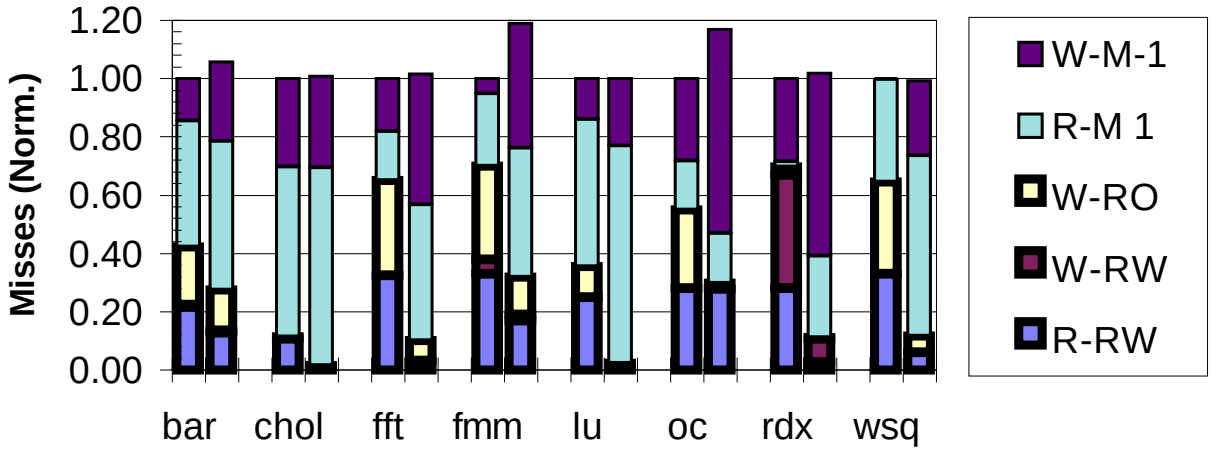
13

Results

Misses by Type Without (left) and With (right) Speculative Actions

Second cache misses in **bold**.

Misses normalized to conventional system (left bar).



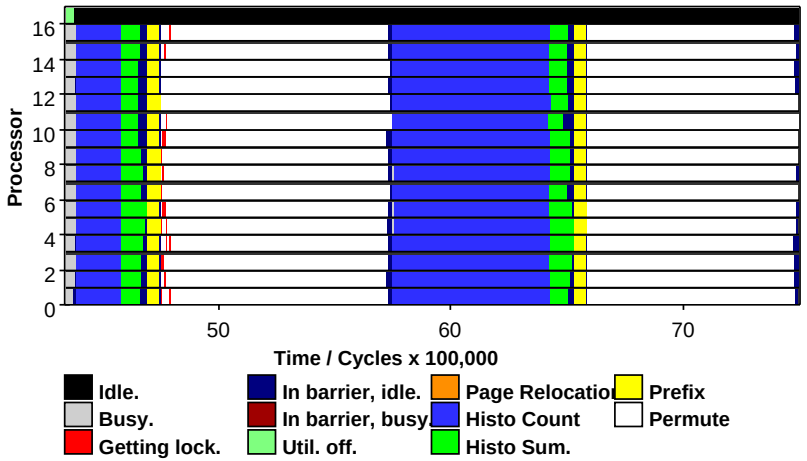
- Substantial reduction in second cache misses.
- Small increase in total number of misses.

PDCS-97 Formatted 15:49, 1 October 1997 from pdcs.

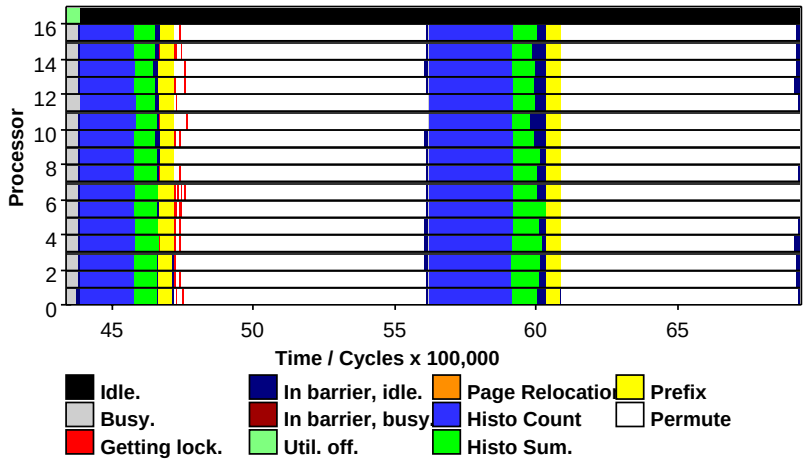
13

Effect on Memory Access Latency and Traffic in Radix

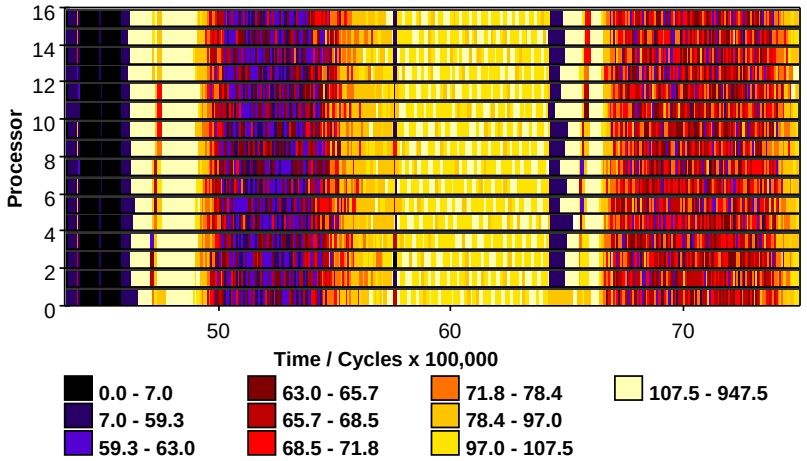
Conventional, States



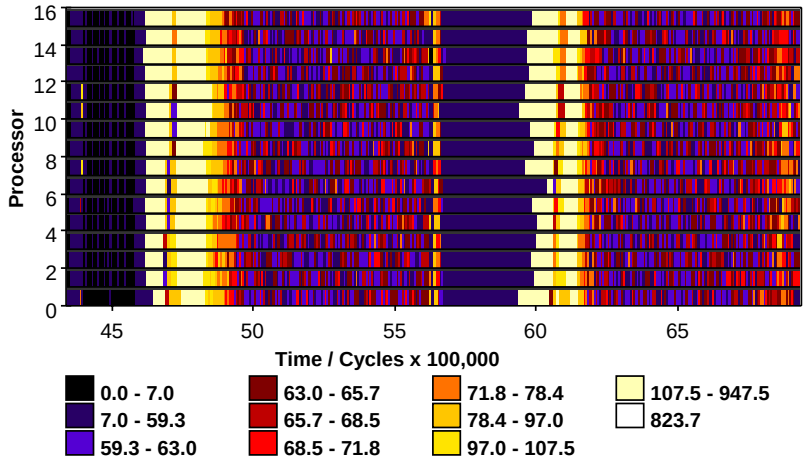
Speculative, States



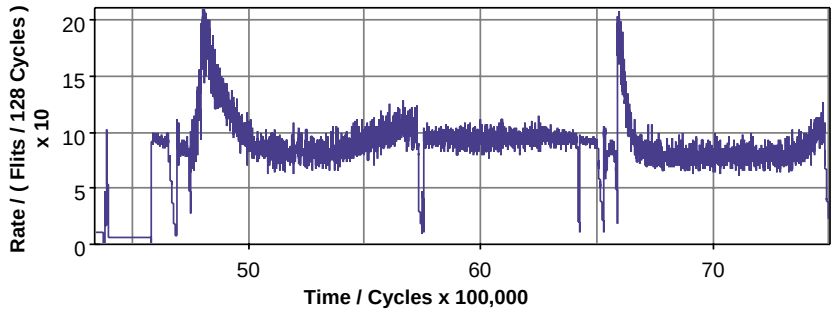
Memory Access Latency



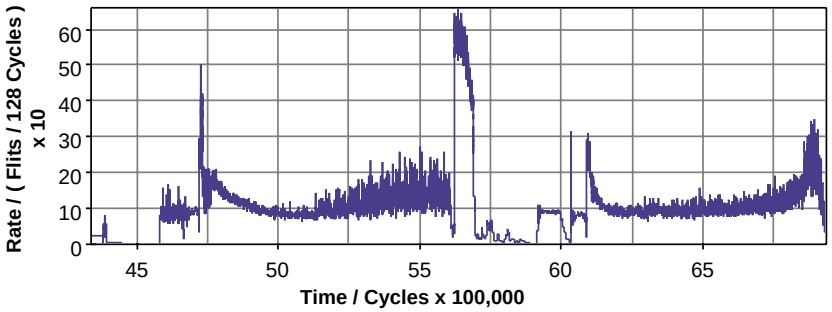
Memory Access Latency



Network Traffic Volume

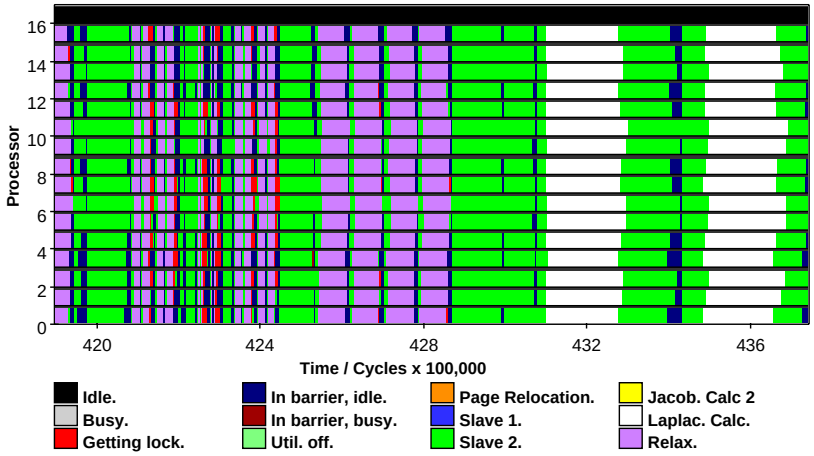


Network Traffic Volume

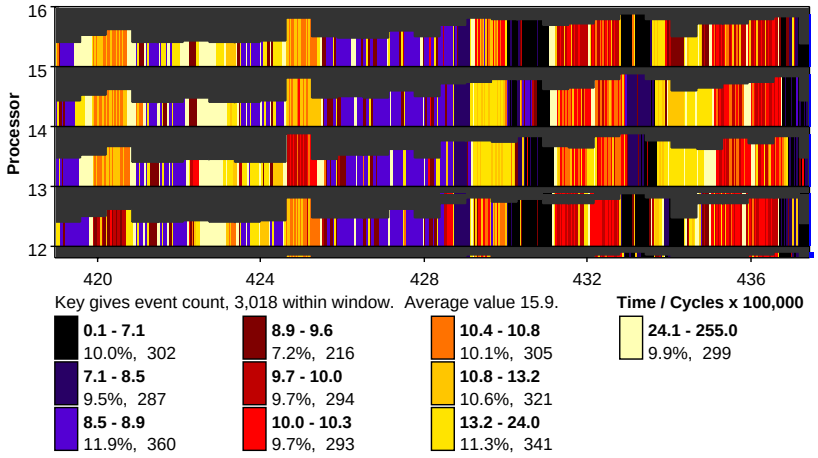


Effect on Memory Access Latency in Ocean

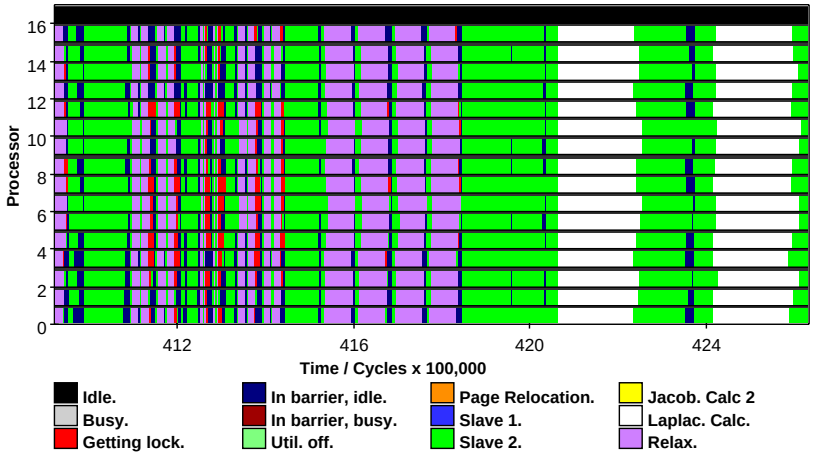
Conventional, States



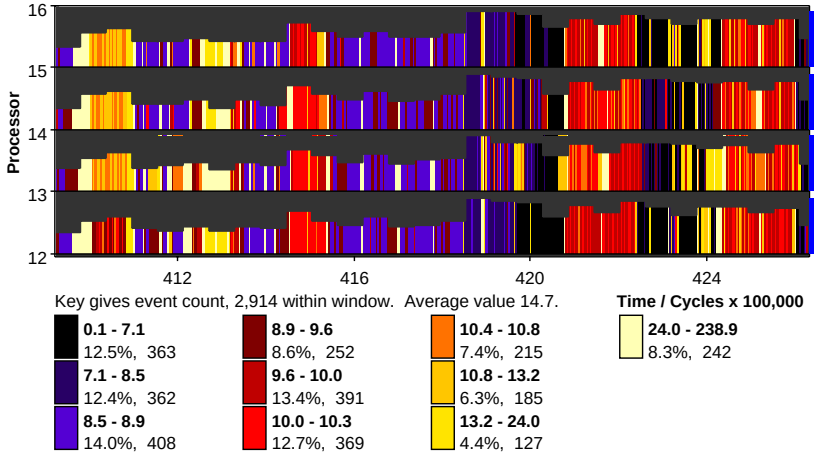
Conventional, Memory Access Lat.



Speculative, States



Speculative, Memory Access Lat.



Effect of Cache Size

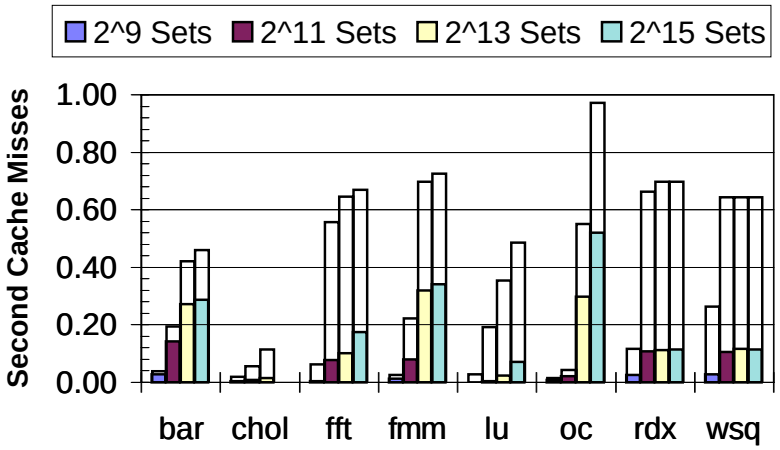
- *Reduction* in SC misses at all sizes . . .

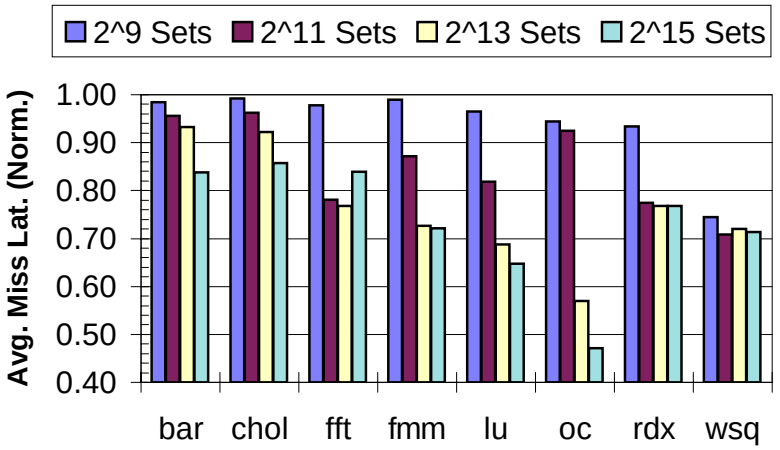
  . . . fewer SC misses when caches small . . .

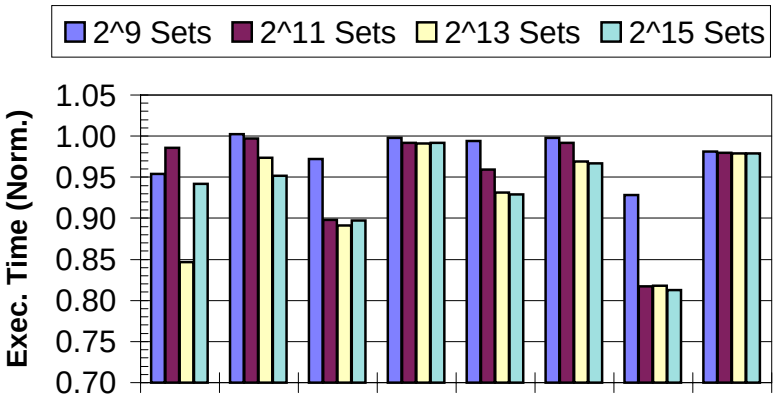
  ∴ little benefit with undersized caches.

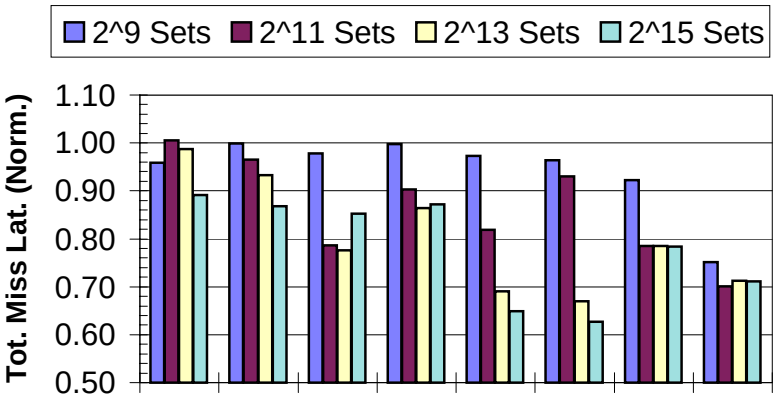
- Or, improved performance . . .

  . . . when increasing cache size ineffective.



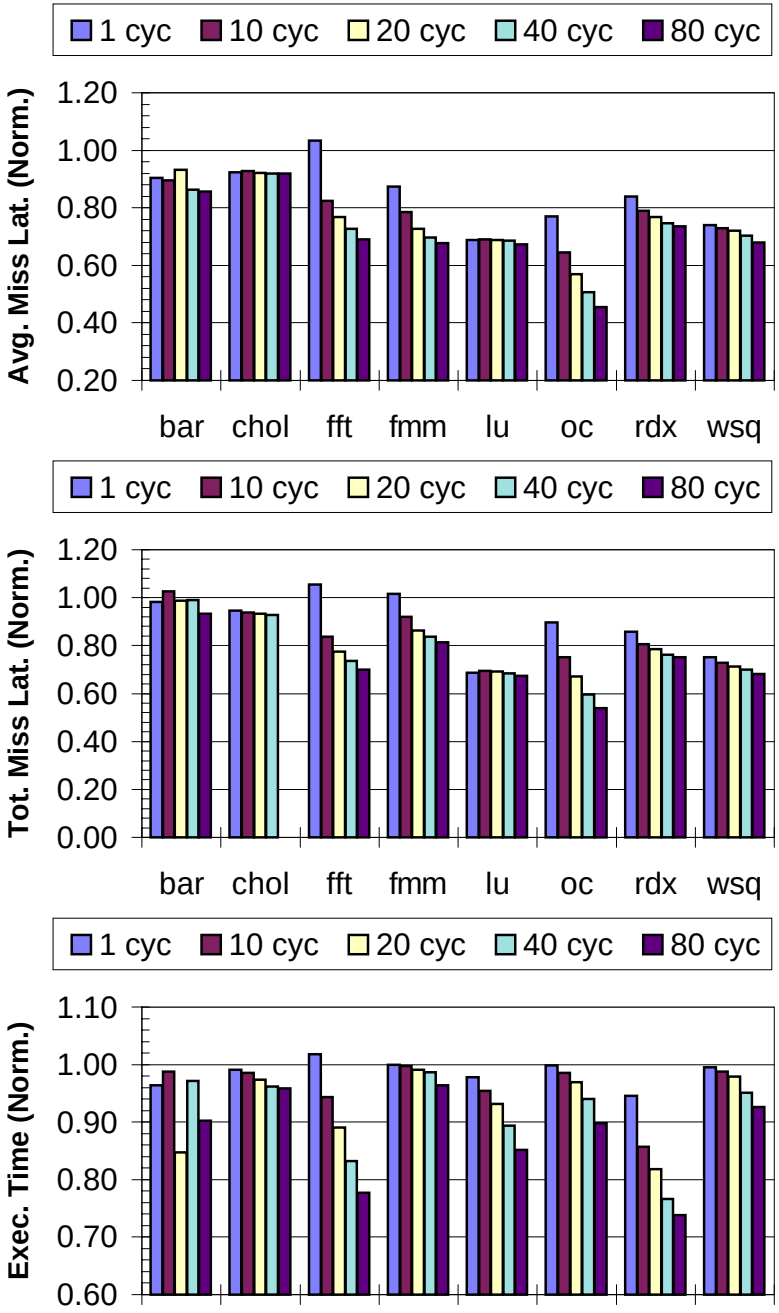






Effect of Network Latency

- Greater benefit when latency higher.



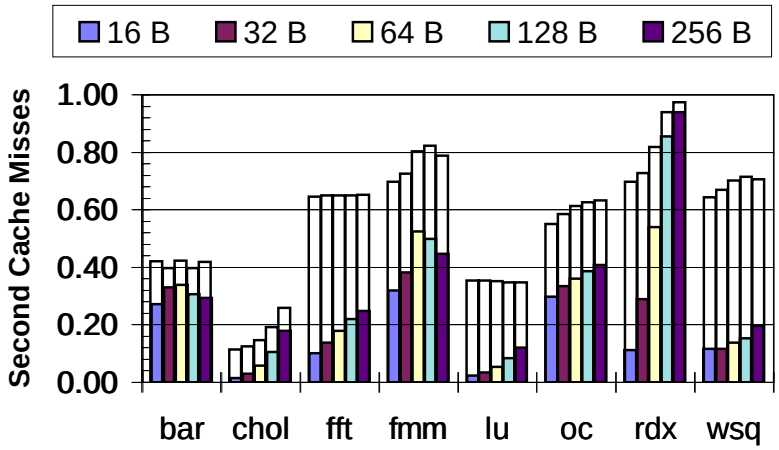
Effect Of Line Size

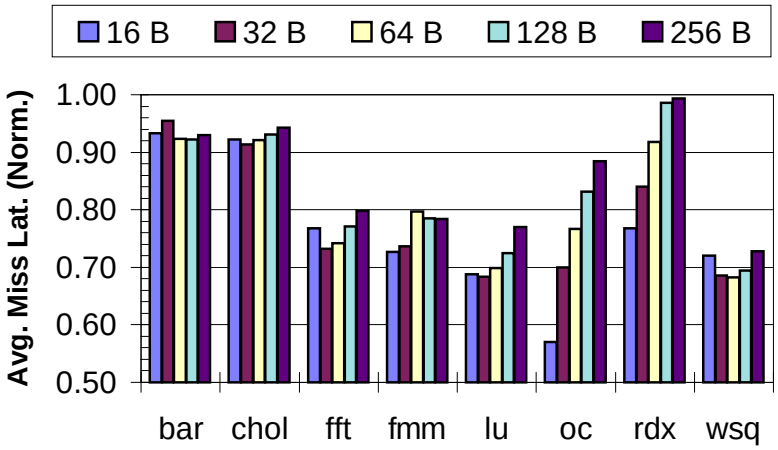
- Lower benefit with longer lines.

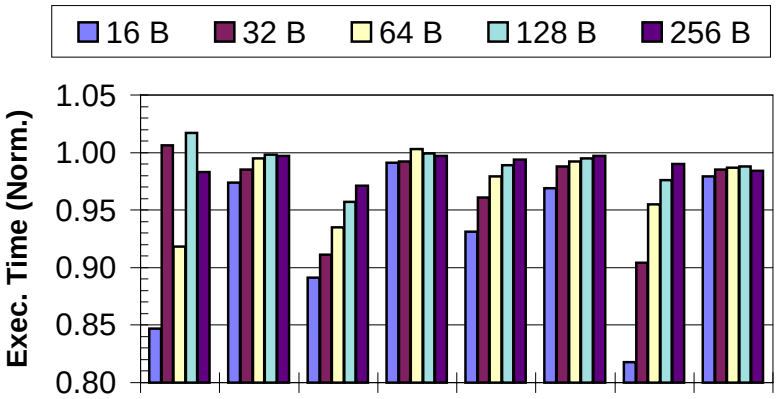
  Effect of line size app. dependent.

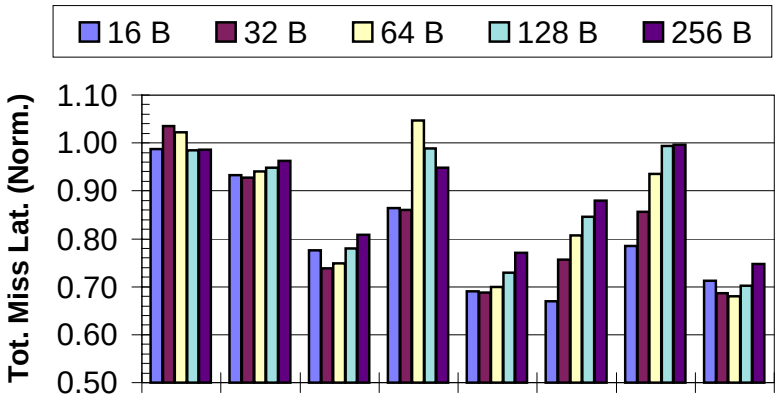
  LU, FMM, Cholesky faster with long lines.

  Apps and Radix slower with longer lines.









19

Summary

Second cache misses effectively reduced.

Benefit best with large cache and latency.

Benefit depends on application, small for some.

Cost comparable to cache.

19