

Multicore Resource Management

Kai Shen

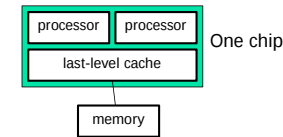
3/21/2011

CSC 258/458 - Spring 2011

1

Multicore Architecture

- Last-level cache becomes a significant part of the chip
- ⇒ Multicore: add another processor core on the chip (sharing a single last-level cache)
 - Low cost (manufacturing, power)



- Additional benefit: faster processor-to-processor sharing

3/21/2011

CSC 258/458 - Spring 2011

2

Contention on Shared Resources

- Shared resources: cache space, memory bandwidth
- Problem due to resource contention:
 - An example: one streaming application and another that significantly reuses a working set
- Performance/efficiency
- Detail of service attacks
- Fairness
 - What does it mean to be fair?
 - Equal resource usage or equal slowdown factor

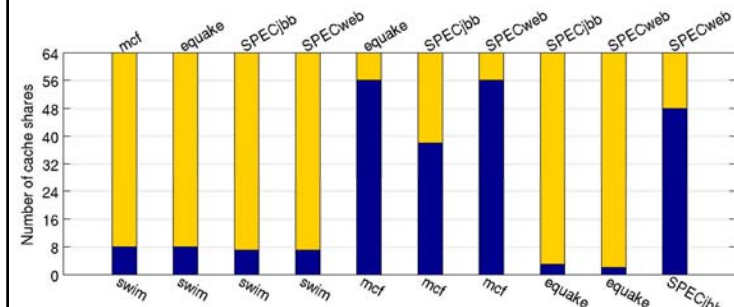
3/21/2011

CSC 258/458 - Spring 2011

3

An Example Resource Allocation

- Cache space partitioning to achieve equal slowdown factor



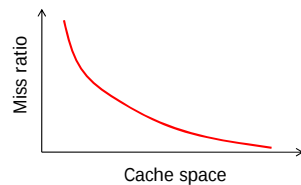
3/21/2011

CSC 258/458 - Spring 2011

4

Cache Management Policy

- How to decide how much cache space given to each application?
- Challenge is how to estimate performance with given space allocation to an application.
 - Miss-ratio curve



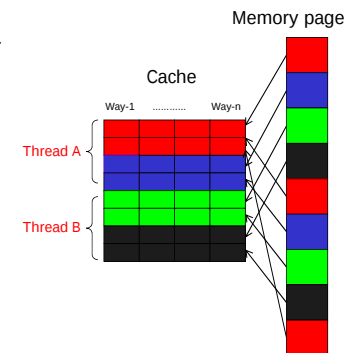
3/21/2011

CSC 258/458 - Spring 2011

5

Cache Management Mechanism

- How to enforce a cache space allocation between co-running applications?
- Page coloring
 - Static cache/mem mapping
 - Allocate app to certain memory pages according to cache allocation



3/21/2011

CSC 258/458 - Spring 2011

6

Challenges for Page Coloring

- Expensive allocation adjustment in a dynamic system
 - Changing a page color means copying the page - 3 microsecs
 - When allocation needs to be adjusted?
- Artificial memory pressure
 - Coloring scheme restricts memory usage $\Rightarrow$ may force one application out of memory while another has ton of free pages

3/21/2011

CSC 258/458 - Spring 2011

7

Hot Page Coloring [Zhang et al., EuroSys 2009]

- Hot pages - frequently accessed memory pages
- Limit page coloring to a small number of hot pages
 - Less adjustment cost; little constraint on memory allocation
 - Substantially achieve the goal of cache space management
- How to identify hot pages online?
 - Periodically check page access bits
 - Scanning the full page table takes too much time? $\Rightarrow$ locality-based jumping

3/21/2011

CSC 258/458 - Spring 2011

8

Indirect Resource Allocation

- Allocate resources indirectly by adjusting application running speed
 - Slow down an application if it is too aggressive in using resources
 - How to do that?
- CPU quantum adjustment [Fedorova et al., PACT 2007]
 - Shorten an application's CPU quantum if it uses too much resources



3/21/2011

CSC 258/458 - Spring 2011

9

Execution Throttling

- CPU quantum adjustment is too coarse-grained
 - ⇒ unstable performance
- Fine-grained CPU execution throttling
 - CPU frequency adjustment [2.0/2.33/2.66/3.0 Ghz] (accompanying voltage adjustment to save power consumption)
 - Duty-cycle modulation [1, 2, ..., 8]
 - Disable certain hardware cache prefetchers [sequential, strided, ...]
- Per-core configurable?

3/21/2011

CSC 258/458 - Spring 2011

10

Multicore Resource Management Using Execution Throttling

- [Zhang et al., USENIX 2009]
 - Using duty-cycle modulation and prefetcher adjustment
- Very simple implementation
 - Dozens of lines of code in Linux
- Low cost configuration
 - Read/write special-purpose processor registers
 - 0.2 microsec for duty-cycle adjustment; 0.8 microsec for prefetcher adjustment
 - Compared to 3 microsecs per page copying in page coloring
- Remaining problem:
 - Too many configuration options to choose from

3/21/2011

CSC 258/458 - Spring 2011

11

Contention Reduction Through Scheduling

- So far assume the given group of co-running applications, but scheduling can adjust the co-running group
- What kinds of applications should be put into one co-running group?

3/21/2011

CSC 258/458 - Spring 2011

12

Learn what is going on

- Hardware counter metrics (Example: Intel Xeon processors)

■ NONHALT_TICKS	Num. of ticks that CPU is in non-halt state
■ INSTRUCTN_RTD	Num. of retired instructions
■ UOPS_RETIRED	Num. of retired micro-ops
■ L1_MISS_RTD	Num. of L1 cache misses due to retired accesses
■ L2_MISS_RTD	Num. of L2 cache misses due to retired accesses
■ L2_MISS	Num. of L2 cache misses
■ L2_REFERENCE	Num. of L2 cache references
■ DTLB_MISS_RTD	Num. of data TLB misses due to retired accesses
■ PGWKMISSTLB	Num. of page walks that page miss handler performs due to data TLB misses
■ DELIVER_MODE	Num. of cycles in trace cache deliver/build modes
■ TRACECACHE_MS	Num. of trace cache lookup misses
■ PGWKMISSTLB	Num. of page walks that page miss handler performs due to instruction TLB misses
■ FSB_DATAREADY	Num. of Data Ready and Data Busy events that occur on the front side bus
■ BUSACCES_CHIP	Num. of transactions on the bus issued by chip
■ BUSACCES_ALL	Num. of transactions on the bus (may be issued by chip or DMA)
■ MACH_CL_COUNT	Num. of entire pipeline being cleared
■ X87_FP_UOP	Num. of X87 float point micro-ops
■ MEM_CANCEL	Num. of canceled requests in the Data Cache Address Control Unit
■ UOPQ_W	Num. of valid micro-ops written to the micro-op queue
■ RES_STALL	Num. of stalls in the Allocator
■ MISPREP_BRANCH	Num. of mis-predicted branches
■ RTD_MISPREP_BRANCH	Num. of retired mis-predicted branches
■ BRANCH	Num. of branches
■ PENDING_EVENT	Num. of load/store micro-ops

3/21/2011 CSC 258/458 - Spring 2011 13

Estimating Cache Space Usage

- Would be nice to know how much cache space an application is using
- Not directly reported by current hardware counters

3/21/2011 CSC 258/458 - Spring 2011 14

Disclaimer

- Materials were derived from collaborative work with Xiao Zhang, Sandhya Dwarkadas, and Rongrong Zhong (all at Rochester).

3/21/2011 CSC 258/458 - Spring 2011 15