OS CONTROLLED PAGE COLORING

Lingxiang Xiang

Issues of Current CPU Cache

- Cache performance is critical to computer systems.
  - memory access: several hundred cycles
  - cache hit: L1 several cycles; L2 10~20 cycles
- Cache performance is poor.
  - more than half of CPU's area is dedicated to cache
  - 30% of total execution time is due to L2 cache miss [Basu, MICRO'06]
  - low utilization: more than 40% of lines are evicted before reuse [Qureshi, ISCA'07]

Issues of Current CPU Cache (cont.)

- Hardware manages cache at the grain of cache lines.
  - single program: data with different locality are mixed together
  - shared cache: uncontrolled sharing threads
    - compete for space -> interference
- Using OS as an auxiliary to manage cache
  - high-level knowledge of program
  - running state of the entire system
  - how? page coloring

Address Mapping in Cache

- physical memory address and cache
  - cache size = line size * way size * # of set
  - 512KB, 16-way, 64B line size L2 cache: 512 sets (9 bits to index)

32 bits addr. set index line offset

L2 Cache

Way-1 Way-16
What is Page Color?

- **color:**
  - cache: a group of cache sets
  - memory: a group of physical pages
  - (page N, N+4, N+8, ...)

- **page color:**
  - data belonging to a page color can only be cached by cache sets with same color

OS and Page Coloring

- What is the rule of OS?
  - control the mapping between virtual memory pages and physical pages via page table

Application of Page Coloring

- Control programs’ cache usage in page grain
  - e.g. mapping all pages of a program to a same color
  - the program can only use 1/(color num) cache size

- Single program
  - Reducing the Harmful Effects of Last-level Cache Polluters with an OS-Level, Software-Only Pollute Buffer [Soares et al., MICRO’08]

- Multicore environment: cache partitioning
  - Managing Shared L2 Caches on Multicore Systems in Software [Tam et al. WIOSCA’07]
Cache Pollution

- L2 Cache Profile of Art
  - Profile generated at run-time

Run-time OS Cache Filtering Service

- 3 Steps
  1. Profile cache behaviour at run-time (using hardware performance counters)
  2. Identify polluting pages
  3. Map polluting pages to small cache slice (Pollute Buffer)

- Low overhead online profiling
  - phase-based sampling with hardware performance counters
  - overall average overhead ~1.6%

- Advantages
  - No changes to architecture
  - No changes to application code or binaries
  - No overhead for CPU-bound applications

Reduce Pollution via Pollute Buffer

- Pollute Buffer: a particular color (cache region) used to store polluting memory pages
- Migrate polluting pages to the pollute buffer
**Effect on Art**

1. Improved L1 MPKI because L1 prefetches hit L2
2. L2 miss reduction ~18%, speedup ~20%

**Shared Caches on Multicore Systems**

- Uncontrolled sharing threads

**Page Coloring Based Cache Partitioning**

- Hardware partitioning mechanisms, not available today
- Page coloring based cache partitioning
  - Implemented on real system
  - Guided phys pages allocation
    -> controlled L2 cache usage
  - Goal: optimize system overall performance

**Benefits of Cache Partitioning**

- Introducing page coloring based cache partitioning can significantly improve system performance, especially in multicore systems where uncontrolled sharing can lead to performance degradation.

Run `mcf,art` on Dual Core Power 5.
Implementation of Cache Partitioning

• Static partitioning
  • profiling + analyze
• Dynamic partitioning
  • identify hot pages
  • re-coloring according to cache partition policy
  • reduce runtime overhead
• Towards Practical Page Coloring Based Multi-core Cache Management [Xiao et al., EuroSys’09]

Summary

• In addition to hardware, OS can be used to control cache via page coloring.
• Advantages over pure hardware management
  • practical
  • utilize high level knowledge about programs and whole system
• Useful for:
  • both single program and multi-programming
  • both single core and multi-core processes
• Issues
  • overhead of page profiling and page re-coloring

THANK YOU!

Disclaimer: The slides are only intended for class representation in CSC 456, at the University of Rochester. All copyrighted materials belong to their original owner(s). Parts of the slides are from Livio Soares et al.’s slides in MICRO’08 David Tam et al.’s slides in WIOCSA’07