A Few Ways Can Take You a Long Way

Efficient and Highly Associative Caches with Scalable Partitioning for Many-Core CMPs

Daniel Sanchez and Christos Kozyrakis, Stanford University

ZCache: An Efficient Highly Associative Cache Design

ZCache Overview

- Caches increasingly critical to CMP performance and power. Trends:
  - Increasing LLC size (e.g. Nehalem-EX, 24MB L3, 50%+ chip area)
  - Increasing LLC associativity (e.g. Opteron 6100, 32-way L3)
- In set-associative caches,
  higher associativity → more ways → higher latency and energy
- ZCache [MICRO 2010]: Novel cache design that provides very high associativity cheaply (e.g. 64-associative cache with 4-ways)
  - Lower latency and energy consumption

ZCache

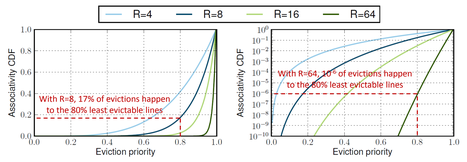
- Array of W ways, B lines/way
- Each way is indexed by a different hash function
  - A line can be in only one position per way
  - Hits take a single lookup
- Replacements trigger multiple accesses off the critical path that yield an arbitrarily large number of eviction candidates. Phases:
  - Walk: Multiple reads of tag array to find candidates
  - Relocations: Move lines to evict desired candidate, make space for new one
- Example:

Analytical Associativity Framework

- Goal: Compare associativity among cache designs independently of replacement policy
- Eviction priority: Rank of a line given by the replacement policy (e.g. LRU), normalized to [0,1]
  - Higher is better to evict (e.g. LRU line has 1.0 priority, MRU has 0.0)
- Associativity distribution: Probability distribution of the eviction priorities of evicted lines
- In a zcache, associativity distribution depends only on the number of replacement candidates (R):

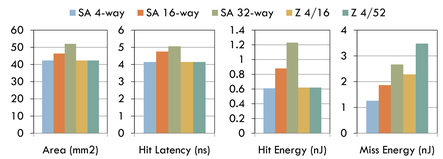
$$F_A(x) = P(A \le x) = x^R, \quad x \in [0,1]$$

  - Independent of ways, workload and replacement policy
  - Same behavior as picking uniform random candidates (due to good hashing and multiple hash functions)

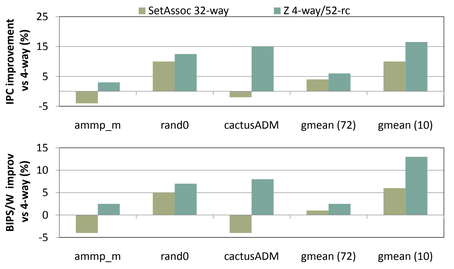


ZCache Evaluation

- Implementation Costs



  - Each design is optimized for area*delay*energy
  - ZCaches retain hit area, hit latency, hit energy of a 4-way SA cache
  - Energy per miss comparable to similarly-associative SA cache
- Performance and Energy-Efficiency



  - 72 multithreaded & multiprogrammed workloads (SPECCPU2006, OMP, PARSEC)
  - 32 in-order 1-issue x86-64 cores, 32KB L1I/D, shared 8MB L2 (set-assoc/zcache)
  - All L2 caches use hashing (H_3)
  - Z4/52 improves performance by 7%, full-system energy efficiency by 10%
- Conclusion: ZCaches enable efficient highly-associative caches
  - Small number of ways, associativity by increasing replacement candidates
  - Costs of high associativity (energy, tag bandwidth) paid only on misses
  - Analytical framework shows that associativity depends on number of replacement candidates, not ways

Vantage: Scalable Fine-Grain High-Associativity Cache Partitioning

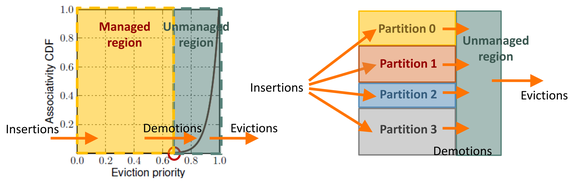
Vantage Overview

- Interference in shared caches a major problem in CMPs
  - Lack of isolation → no QoS
  - Poor cache utilization → degraded performance
- Cache partitioning addresses interference, but current partitioning techniques (e.g. way-partitioning) have serious drawbacks
  - Support few coarse-grain partitions → do not scale to many-cores
  - Hurt associativity → degraded performance
- Vantage [ISCA 2011] solves deficiencies of previous techniques
  - Leverages zcache's high, guaranteed associativity
  - Supports hundreds of fine-grain partitions
  - Maintains high associativity and strict isolation among partitions
  - Enables cache partitioning in many-cores

	Way partitioning	PIPP	Reconfig. caches	Page coloring	Vantage
Scalable & fine-grain	×	×	×	×	✓
Strict isolation	✓	×	✓	✓	✓
Dynamic	✓	✓	×	×	✓
Maintains assoc.	×	✓	✓	✓	✓
Indep. of repl. policy	✓	×	✓	✓	✓
Simple	✓	✓	×	✓	✓
Partitions whole cache	✓	✓	✓	✓	× (most)

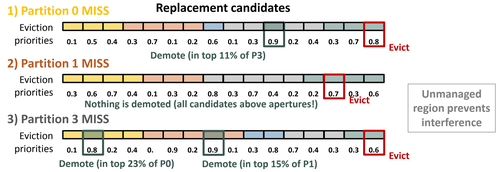
Vantage

- Vantage partitions most of the cache through the replacement process
  - No restrictions on line placement
  - Derived from analytical models, providing strict bounds on sizes and interference
  - Vantage guarantees rely on caches with guaranteed associativity (e.g. zcache)



- Churn-based management
  - Problem: always demoting from inserting partition does not scale
  - Instead, demote to match demotion rate to insertion rate (churn)
  - Aperture: Portion of lines to demote from each partition
  - Example:

Apertures	Partition 0	Partition 1	Partition 2	Partition 3
	23%	15%	12%	11%



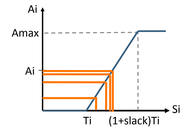
  - Aperture depends on partition churn/ size
  - Smaller aperture ↔ better associativity
- Stability: Controlling aperture not enough in high churn/size partitions
  - Set a max aperture Amax (e.g. 40%); if a partition needs Ai > Amax, let it grow
  - Key result: Regardless of number of partitions that need to grow beyond their targets, the worst-case total growth over their target sizes is bounded and small!

$$\frac{1}{A_{max}} \frac{1}{R}$$

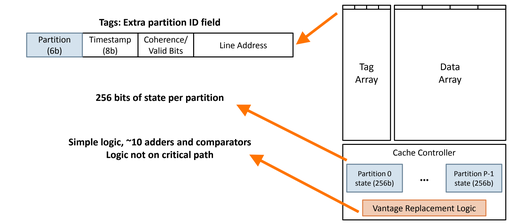
    - 5% of the cache with R=52, Amax=40%
    - Size unmanaged region with that extra slack → Stability and scalability are guaranteed

Vantage Controller

- Directly implementing these techniques is impractical
  - Must constantly compute apertures, estimate churns
  - Need to know eviction priorities of every block
- Use negative feedback to derive apertures and lines below aperture
  - Practical implementation that maintains analytical guarantees
- Feedback-based aperture control
  - Adjust aperture by letting partition size (Si) grow over its target (Ti)
  - Need small extra space in unmanaged region (e.g. 0.5% with R=52, Amax=40%, slack=10%)



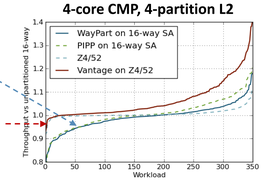
- Small implementation costs (see paper for details):



Vantage Evaluation

- Simulated small (4-core) and large (32-core) systems with shared L2
- Partitioning policy: Utility-based partitioning [Qureshi 2006]
  - Assign more space to threads that can use it better
- Partitioning schemes: Way-partitioning, PIPP, Vantage
- Workloads: 350 multiprogrammed mixes from SPECCPU2006 (full suite)
- Performance:

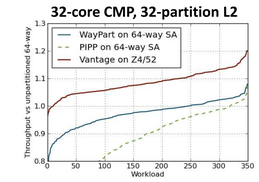
4-core CMP, 4-partition L2



Way-partitioning, PIPP degrade throughput for 45% of workloads

Vantage + 4-way zcache improves throughput for most workloads (6.2% gmean speedup, 26% for the 50 most memory intensive)

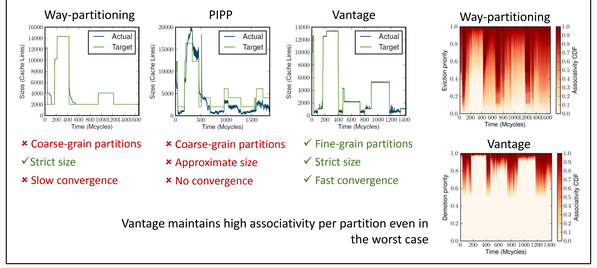
32-core CMP, 32-partition L2



Way-partitioning, PIPP degrade throughput for most workloads with 64-way caches

Vantage still improves throughput for most workloads with the same Z4/52 zcache

- Partition Sizes and Associativity



Vantage maintains high associativity per partition even in the worst case