**ZCache: An Efficient Highly Associative Cache Design**

**Implementation Costs**
- ZCaches retain hit area, hit latency, hit energy of a 4-way SA cache.
- Costs of high associativity (energy, tag bandwidth) paid only on misses.
- Size unmanaged region with that extra slack for the 50 most memory intensive.
- No restrictions on line placement.
- Example: Cache partitioning.

**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). hit line has 3.0 priority, MRU has 0.0.
- Associativity distribution: Probability distribution of the eviction priorities of evicted lines.
- In a zache, associativity distribution depends only on the number of replacement candidates (S).

**ZCache Evaluation**
- A line can be in only one position per way.
- Hits take a single lookup.
- Replacements trigger multiple accesses off the critical path.
- Same behavior as picking uniform random candidates (due to good hashing and multiple hash functions).

**Vantage: Scalable Fine-Grain High-Associativity Cache Partitioning**
- Interference in shared caches a major problem in CMPs.
- Lack of isolation → Poor Out.
- Poor cache utilization → Degraded performance.
- Cache partitioning addresses interference, but current partitioning techniques (e.g., pin-partitions) have serious drawbacks.
- Support fine-grain partitions → Do not scale to many cores.
- Vantage: Solves deficiencies of previous techniques.

**Vantage Controller**
- Directly implementing these techniques is impractical.
- Most commonly compute apertures, estimate chums.
- Need to know eviction priorities of every block.
- Need to provide appropriate cache sizes.
- 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).
- Use small extra space in unmanaged region (e.g. 0.0% with B=52, Amav=40, shrink=20%).

**Small implementation costs** (see paper for details):