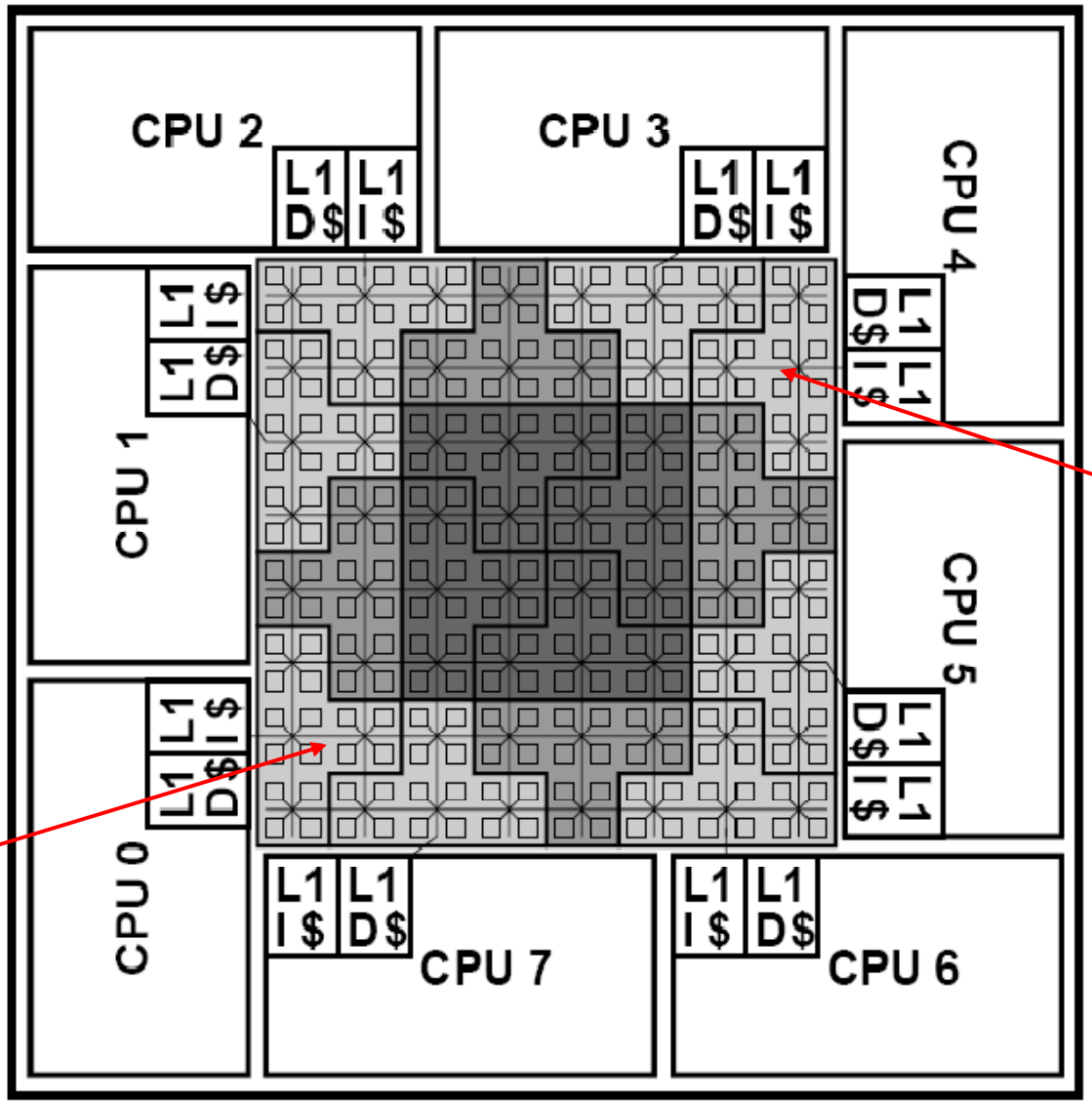
Lecture 12: Large Cache Design

Papers (papers from last class and…):

- Co-Operative Caching for Chip Multiprocessors, Chang and Sohi, ISCA'06
- Victim Replication, Zhang and Asanovic, ISCA'05
- Interconnect Design Considerations for Large NUCA Caches, Muralimanohar and Balasubramonian, ISCA'07
- Design and Management of 3D Chip Multiprocessors using Network-in-Memory, Li et al., ISCA'06
- A Domain-Specific On-Chip Network Design for Large Scale Cache Systems, Jin et al., HPCA'07
- Nahalal: Cache Organization for Chip Multiprocessors, Guz et al., Comp. Arch. Letters, 2007

Beckmann and Wood, MICRO'04



Bankcluster Key

- Local
- Inter.
- Center

Latency 65 cyc

Latency 13-17cyc

Data must be placed close to the center-of-gravity of requests

Examples: Frequency of Accesses

Dark → more accesses

←OLTP (on-line transaction processing)

Ocean → (scientific code)

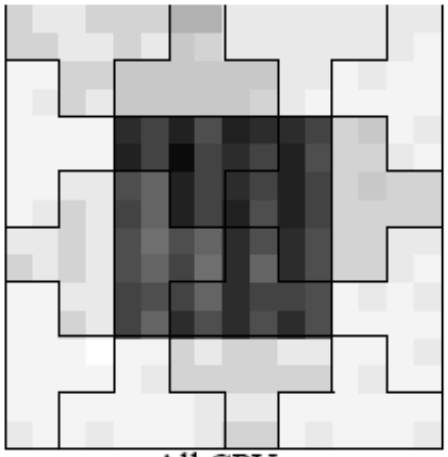


Figure 10. oltp L2 Hit Distribution

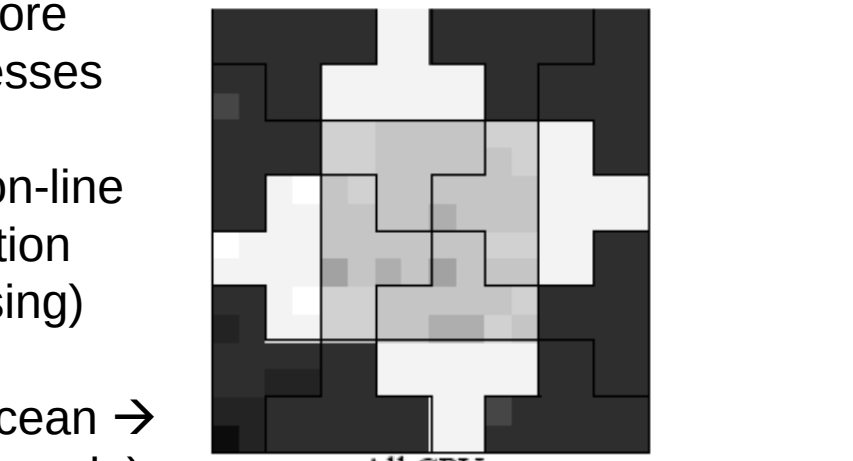
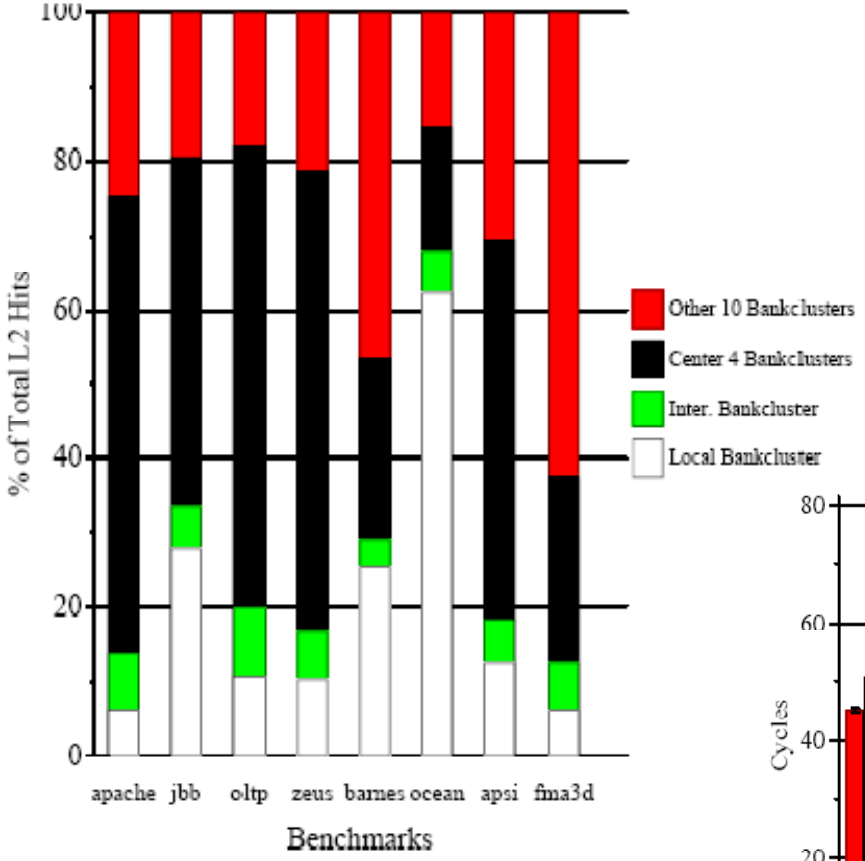


Figure 11. ocean L2 Hit Distribution

Block Migration Results



While block migration reduces avg. distance, it complicates search.

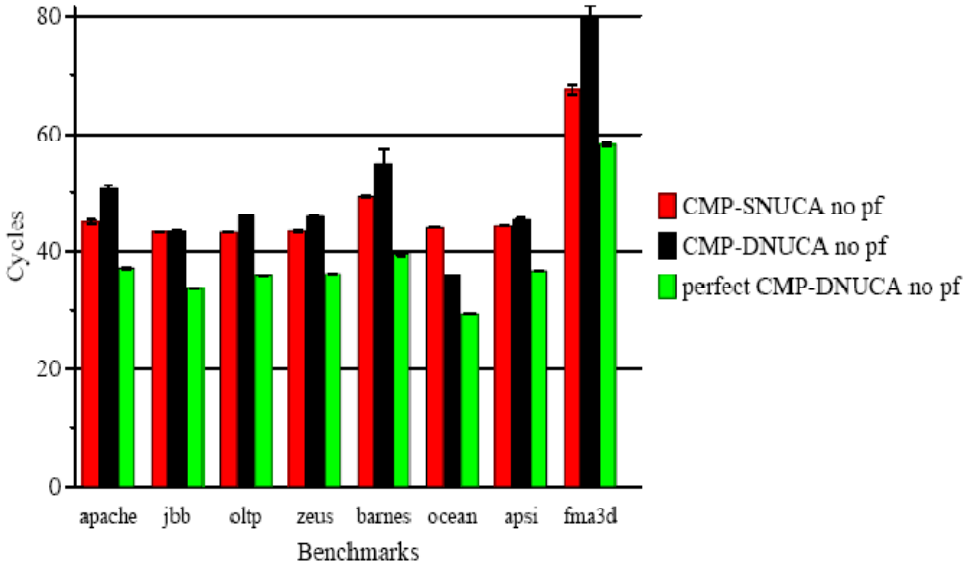
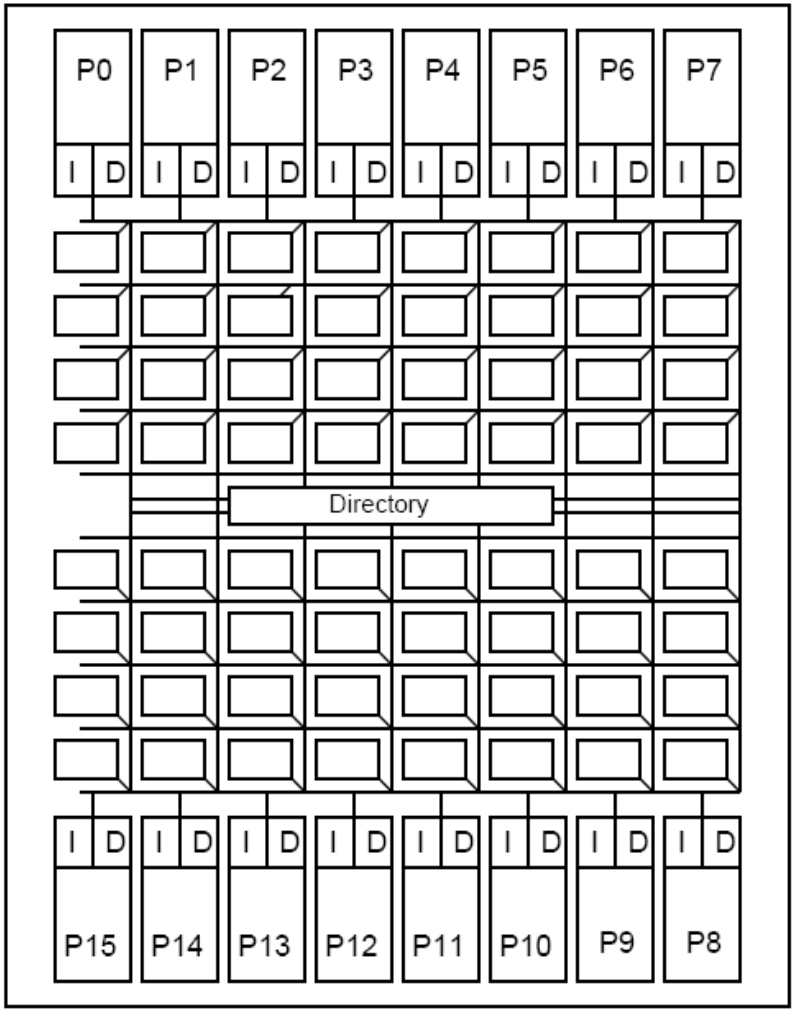


Figure 12. Avg. L2 Hit Latency: No Prefetching

Alternative Layout

(a) CMP Substrate: 16 CPUs 8x8 Banks



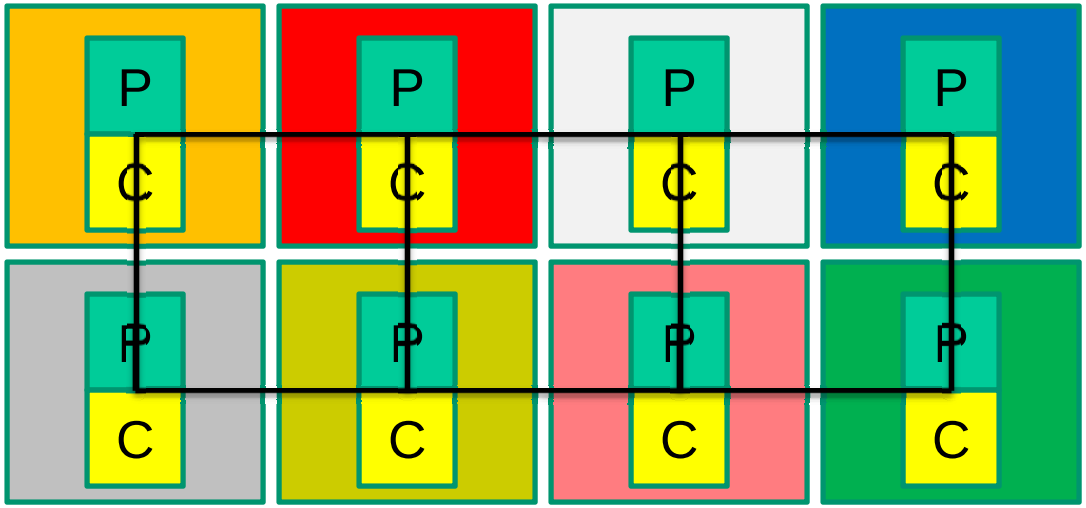
From Huh et al., ICS'05:

- Paper also introduces the notion of sharing degree
- A bank can be shared by any number of cores between N=1 and 16.
- Will need support for L2 coherence as well

Cho and Jin, MICRO'06

- Page coloring to improve proximity of data and computation
- Flexible software policies
- Has the benefits of S-NUCA (each address has a unique location and no search is required)
- Has the benefits of D-NUCA (page re-mapping can help migrate data, although at a page granularity)
- Easily extends to multi-core and can easily mimic the behavior of private caches

Page Coloring Example



- Recent work (Awasthi et al., HPCA'09) proposes a mechanism for hardware-based re-coloring of pages without requiring copies in DRAM memory

Private L2s

Arguments for private L2s:

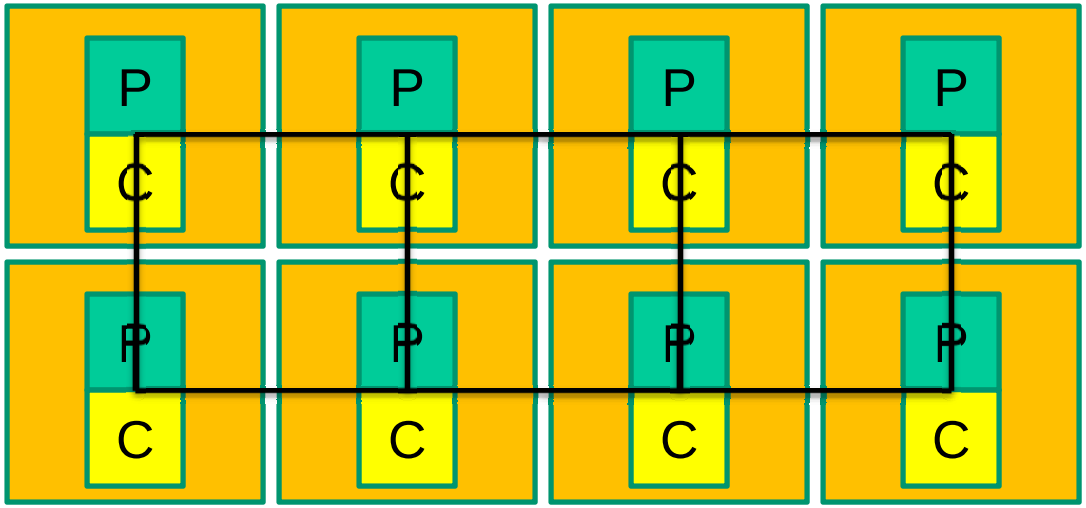
- Lower latency for L2 hits
- Fewer ways have to be looked up for L2 hits
- Performance isolation (little interference from other threads)
- Can be turned off easily (since L2 does not have directory info)
- Fewer requests on the on-chip network

Primary disadvantage:

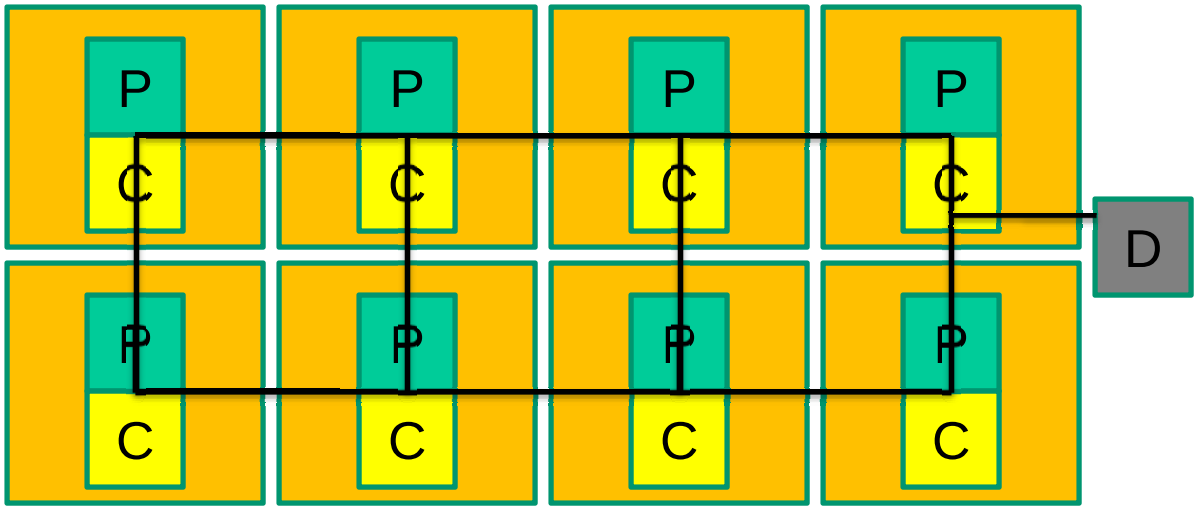
- More off-chip accesses because of higher miss rates

Victim Replication

- Large shared L2 cache (each core has a local slice)
- On an L1 eviction, place the victim in local L2 slice (if there are unused lines)
- The replication does not impact correctness as this core is still in the sharer list and will receive invalidations
- On an L1 miss, the local L2 slice is checked before fwding the request to the correct slice

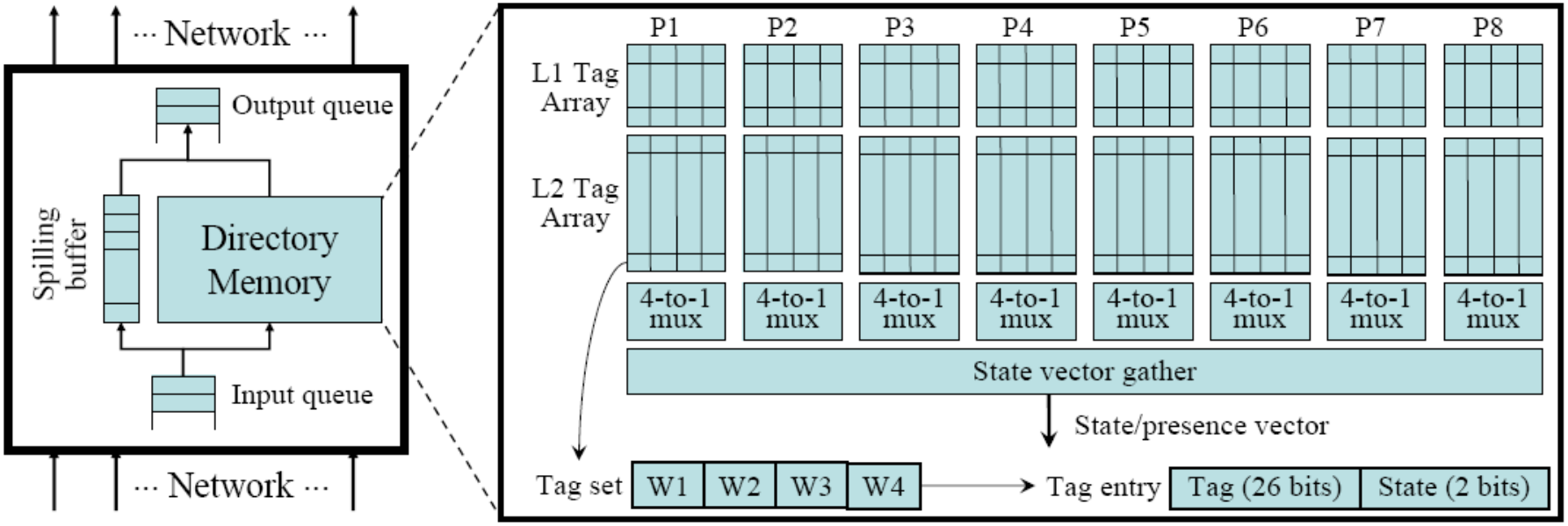


Coherence among L2s



- On an L2 miss, can broadcast request to all L2s and off-chip controller (snooping-based coherence for few cores)
- On an L2 miss, contact a directory that replicates tags for all L2 caches and handles the request appropriately (directory-based coherence for many cores)

The Directory Structure



- For 64-byte blocks, 1 MB L2 caches, overhead ~432 KB
- Note the complexities in maintaining presence vectors, non-inclusion for L1 and L2
- Note that clean evictions must also inform the central directory
- Need not inform directory about L1-L2 swaps (the directory is imprecise about whether the block will be found in L1 or L2)

Co-operation I

- Cache-to-cache sharing
- On an L2 miss, the directory is contacted and the request is forwarded to and serviced by another cache
- If silent evictions were allowed, some of these forwards would fail

Co-operation II

- Every block keeps track of whether it is a *singlet* or *replicate* – this requires notifications from the central directory every time a block changes modes
- While replacing a block, replicates are preferred (with a given probability)
- When a singlet block is evicted, the directory is contacted and the directory then forwards this block to another randomly selected cache (weighted probabilities to prefer nearby caches or no cache at all) (hopefully, the forwarded block will replace another replicate)

Co-operation III

- An evicted block is given a *Recirculation Count* of N and pushed to another cache – this block is placed as the LRU block in its new cache – every eviction decrements the RC before forwarding (this paper uses N=1)

- Essentially, a block has one more chance to linger in the cache – it will stick around if it is reused before the new cache experiences capacity pressure

- This is an attempt to approximate a global LRU policy among all 32 ways of aggregate L2 cache

- Overheads per L2 cache block: one bit to indicate “once spilled and not reused” and one bit for “singlet” info

Results

Table 5. Multithreaded Workload Miss Rate and L1 Miss Breakdown

	Thousand misses per transaction Off-chip (Private / Shared / CC)	L1 Misses breakdown (Private / Shared / CC)		
		Local L2	Remote L2	Off-chip
OLTP	9.75 / 3.10 / 3.80	90% / 15% / 86%	7% / 84% / 13%	3% / 1% / 1%
Apache	1.60 / 0.90 / 0.94	65% / 9% / 51%	15% / 77% / 36%	20% / 14% / 13%
JBB	0.13 / 0.08 / 0.10	72% / 10% / 57%	14% / 80% / 32%	14% / 10% / 11%
Zeus	0.71 / 0.46 / 0.49	67% / 9% / 45%	15% / 78% / 41%	19% / 12% / 13%

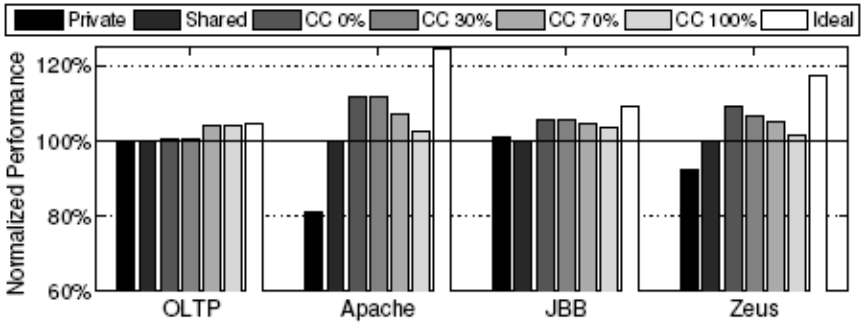


Figure 4. Multithreaded Workload Performance

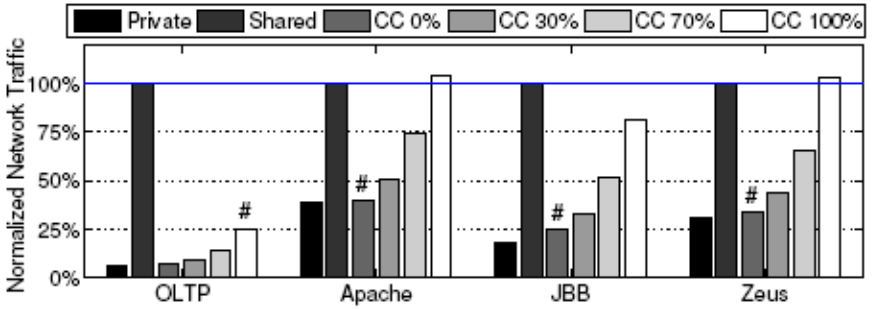


Figure 6. Multithreaded Workload Bandwidth ("#" indicates the best performing CC scheme)

Results

Table 6. Multiprogrammed Workload Miss Rate and L1 Miss Breakdown

	Misses per thousand instructions Off-chip (Private / Shared / CC)	L1 Misses breakdown (Private / Shared / CC)		
		Local L2	Remote L2	Off-chip
Mix1	3.1 / 2.0 / 2.4	78% / 19% / 67%	3% / 73% / 22%	19% / 9% / 11%
Mix2	3.0 / 1.6 / 1.8	64% / 35% / 75%	4% / 55% / 14%	32% / 9% / 11%
Mix3	1.2 / 0.7 / 0.8	91% / 20% / 87%	1% / 77% / 9%	7% / 3% / 4%
Mix4	0.6 / 0.3 / 0.3	95% / 12% / 90%	0% / 86% / 8%	4% / 2% / 2%
Rate1	0.8 / 0.6 / 0.8	90% / 20% / 80%	3% / 76% / 13%	7% / 4% / 6%
Rate2	53 / 51 / 41	31% / 7% / 24%	11% / 47% / 34%	58% / 46% / 42%

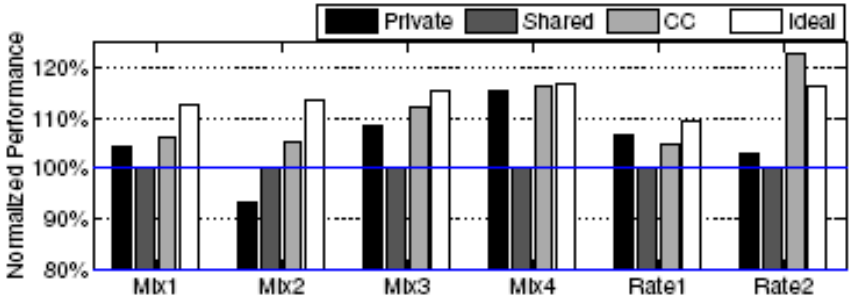


Figure 7. Multiprogrammed Workload Performance

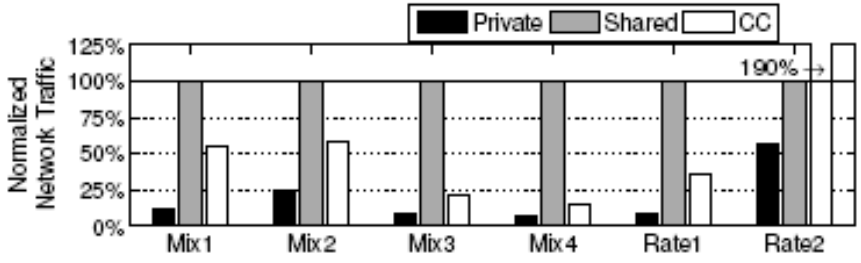
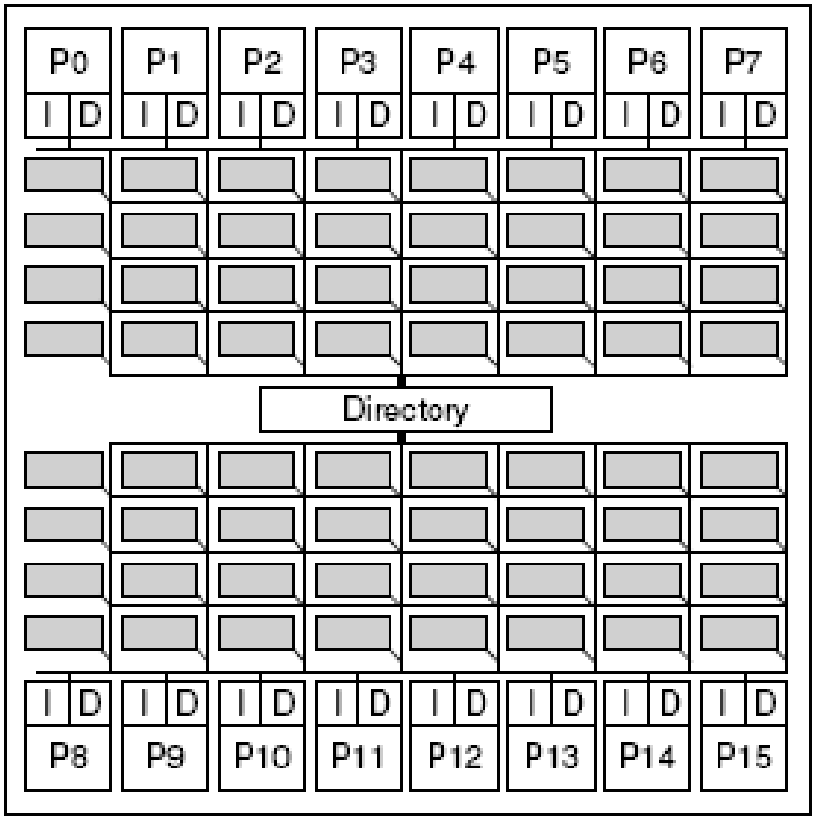
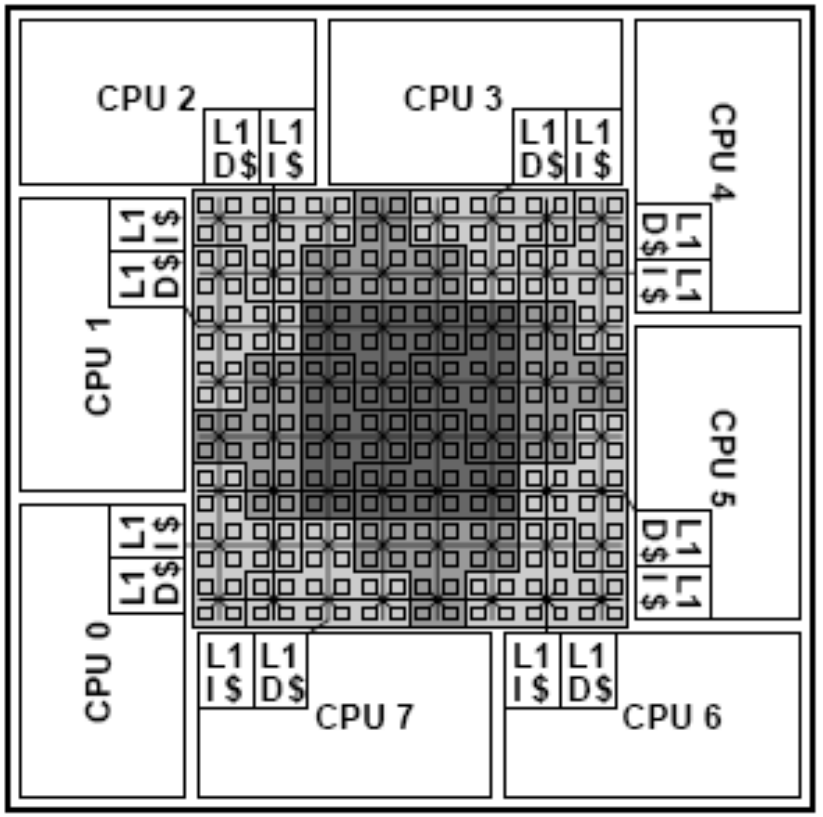


Figure 9. Multiprogrammed Workload Bandwidth

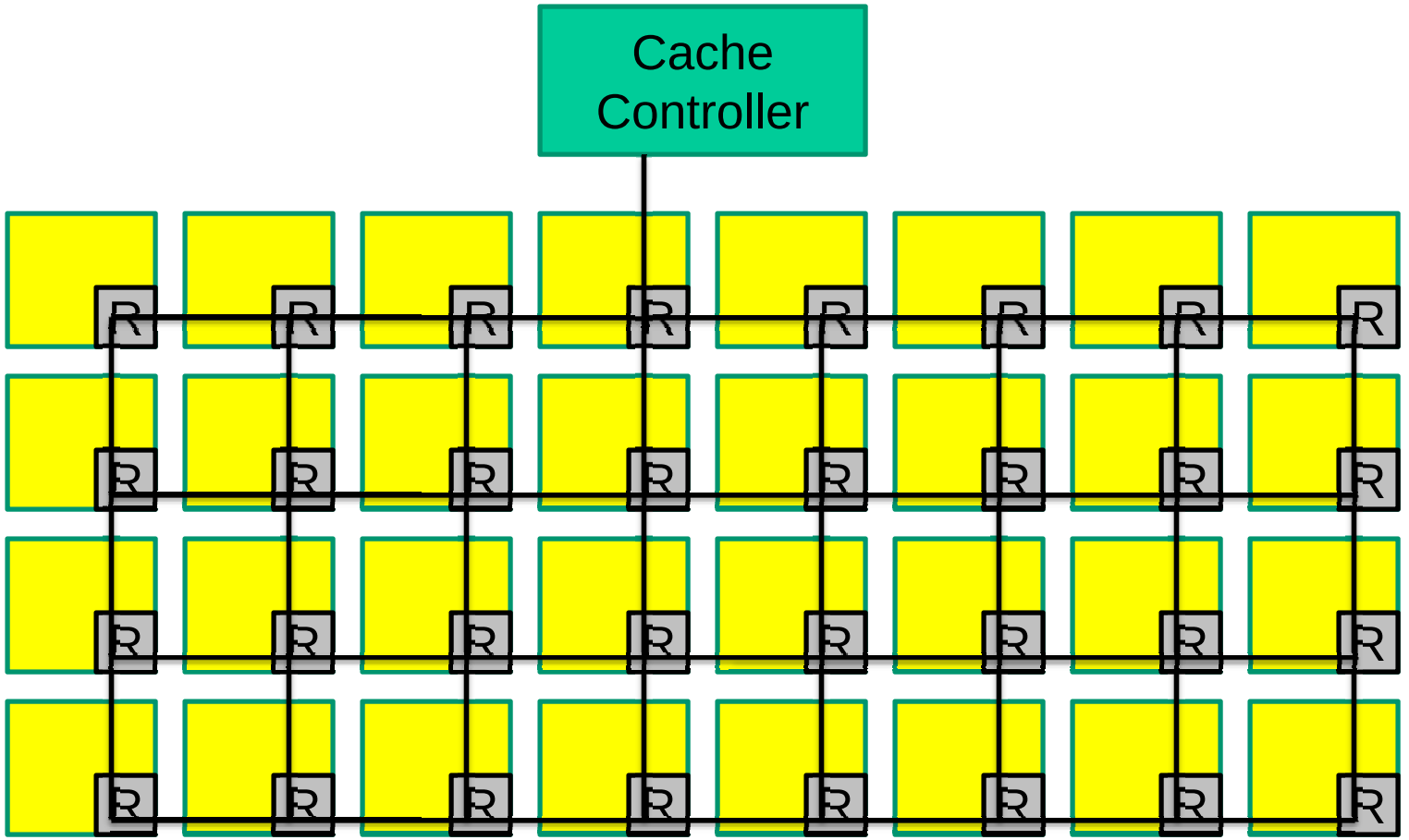
Traditional Networks



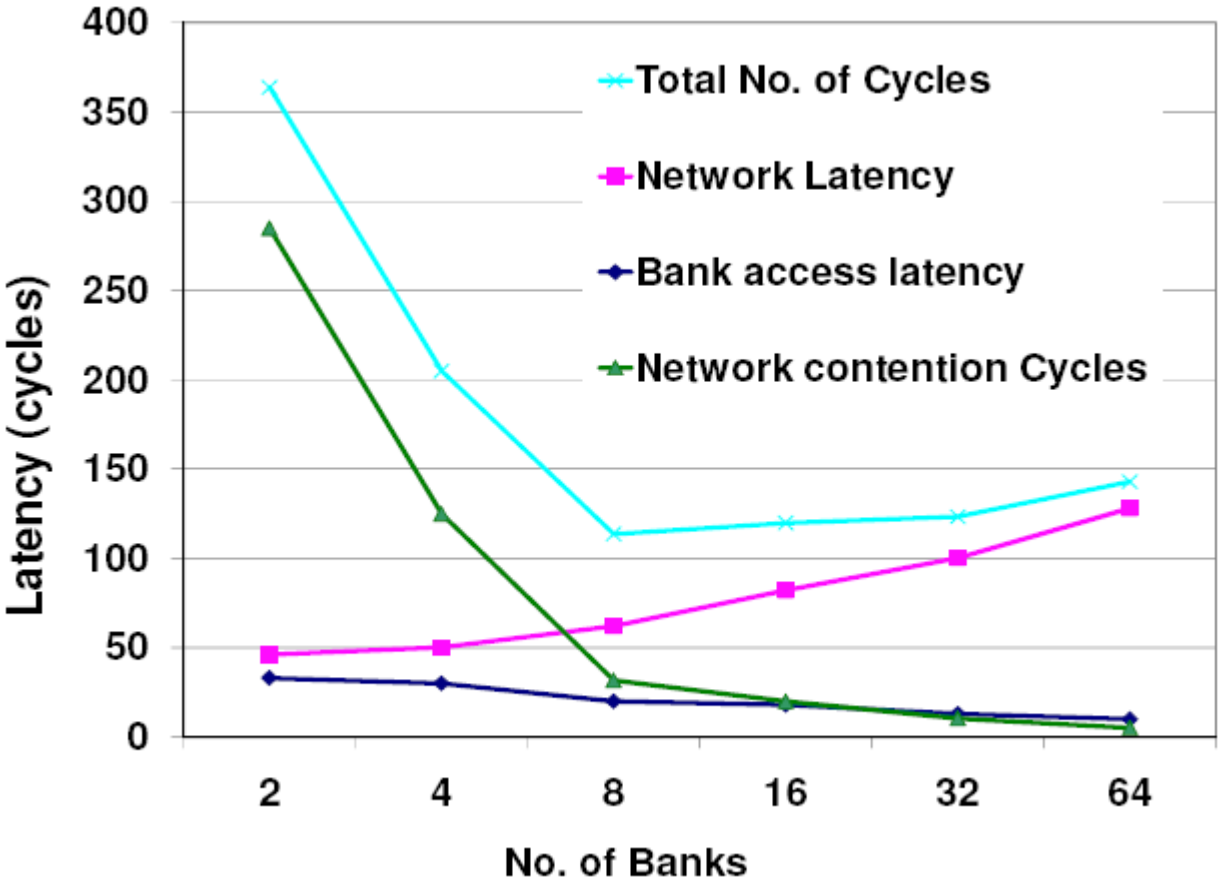


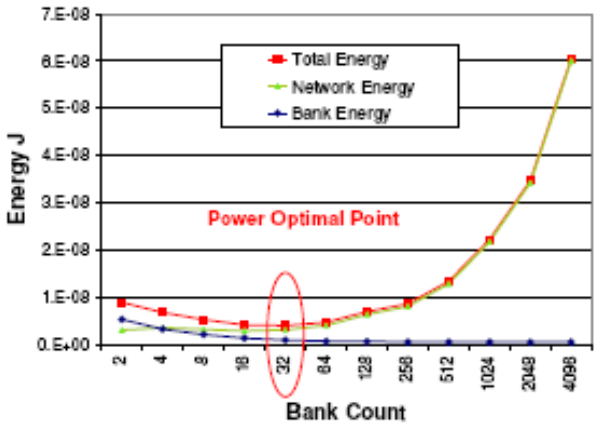
Example designs for contiguous L2 cache regions

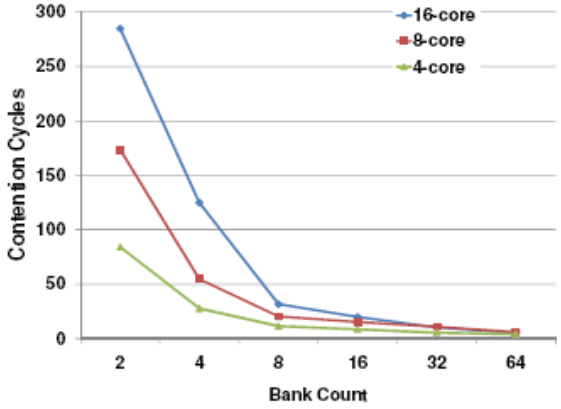
NUCA Delays



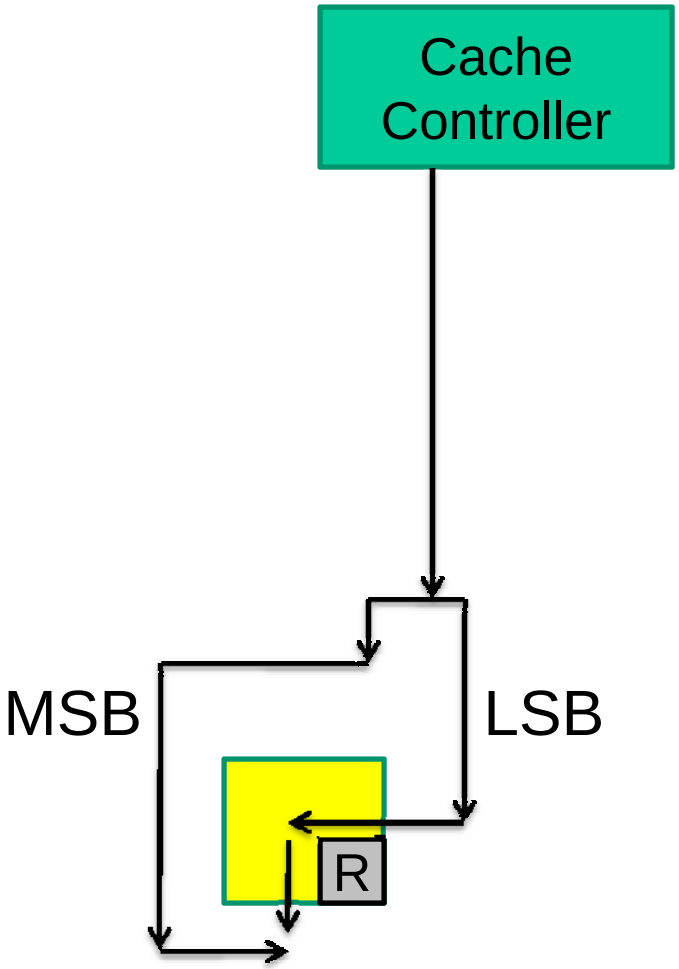
Explorations for Optimality





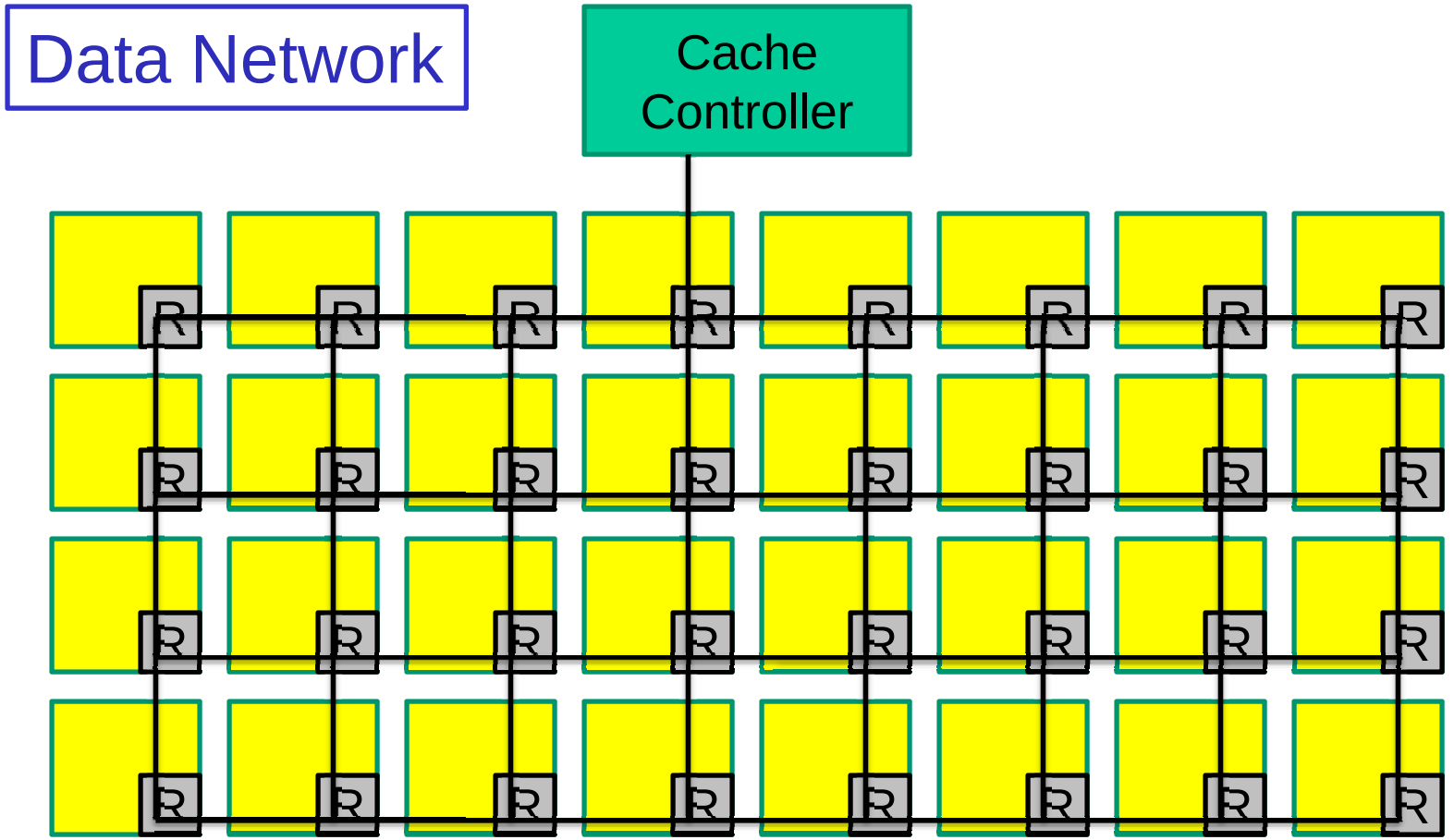


Early and Aggressive Look-Up

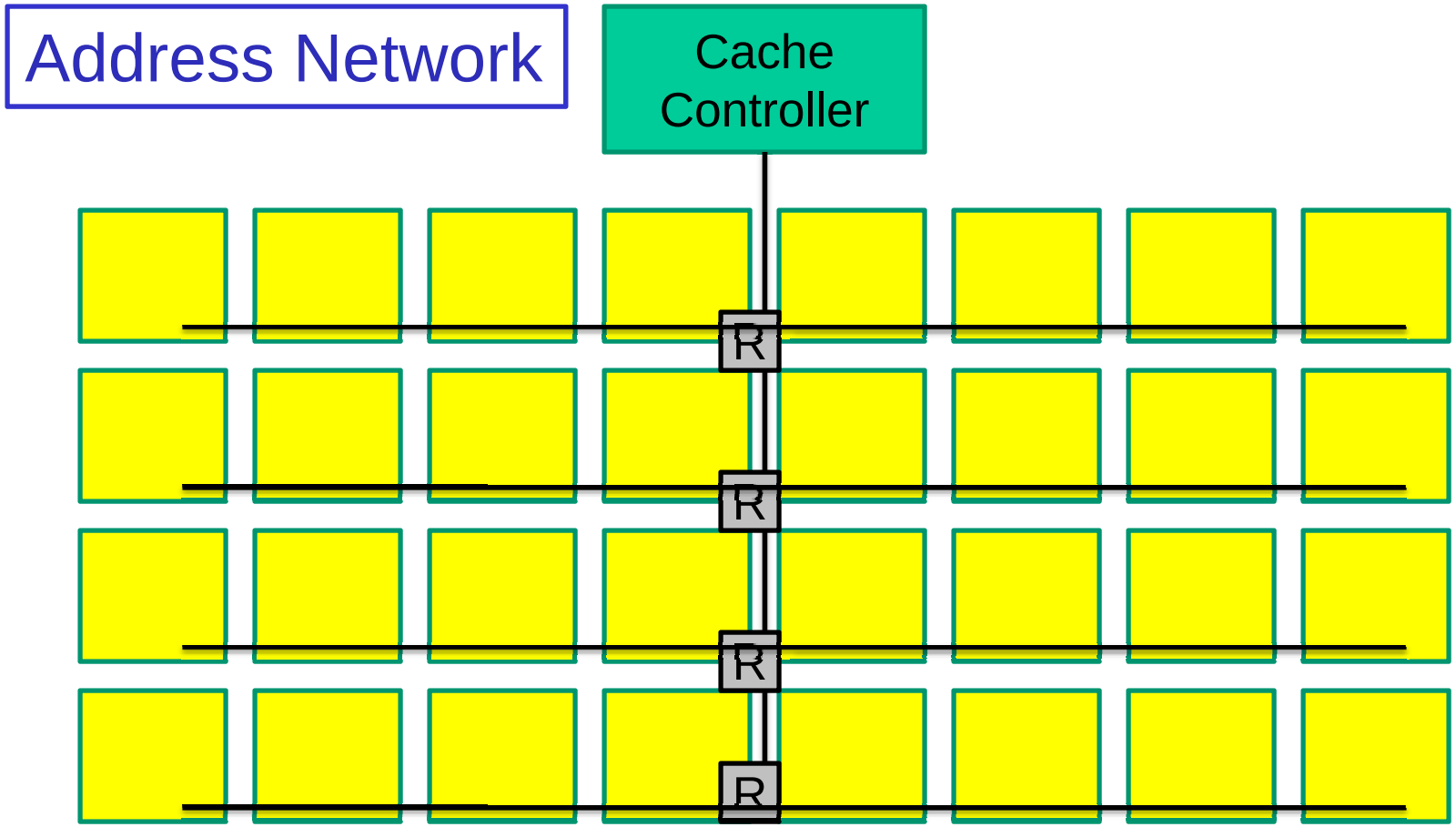


- Address packet can only contain LSB and can use latency-optimized wires (transmission lines / fat wires)
- Data packet also contains tags and can use regular wires
- The on-chip network can now have different types of links for address and data

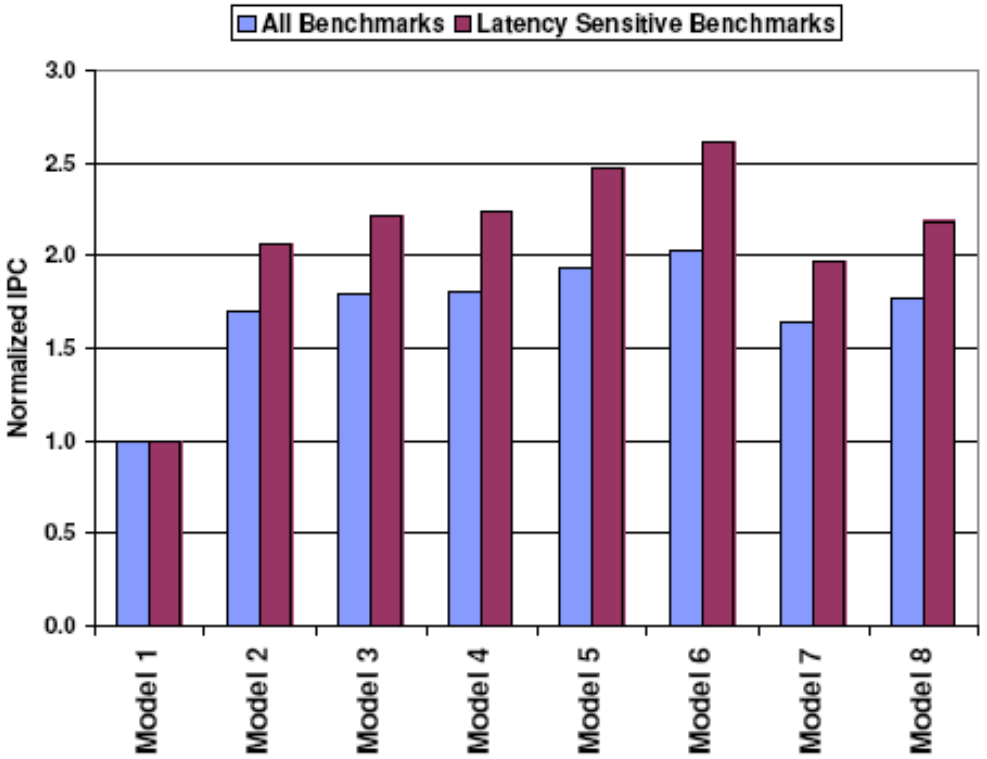
Hybrid Network



Hybrid Network

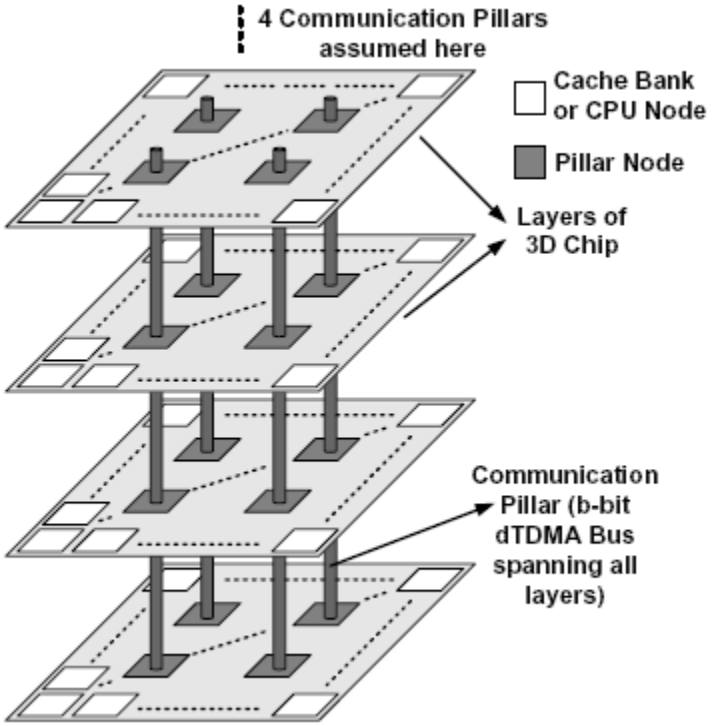


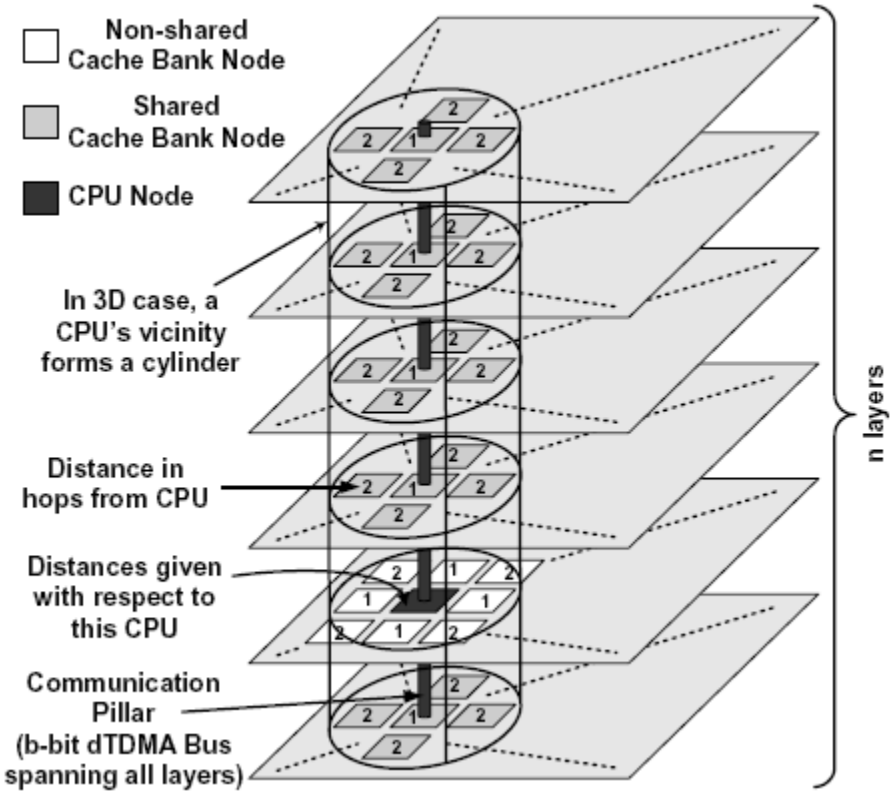
Results



Model	Link latency (vert,horiz)	Bank access time	Bank count	Network link contents	Description
Model 1	1,1	3	512	B-wires (256D, 64A)	Based on prior work
Model 2	4,3	17	16	B-wires (256D, 64A)	Derived from CACTIL2
Model 3	4,3	17	16	B-wires (128D, 64A) & L-wires (16A)	Implements early look-up
Model 4	4,3	17	16	B-wires (128D) & L-wires (24A)	Implements aggressive look-up
Model 5	hybrid	17	16	L-wires (24A) & B-wires (128D)	Latency-bandwidth tradeoff
Model 6	4,3	17	16	B-wires (256D), 1cycle Add	Implements optimistic case
Model 7	1,1	17	16	L-wires (40A/D)	Latency optimized
Model 8	4,3	17	16	B-wires (128D) & L-wires (24A)	Address-L-wires & Data-B-wires

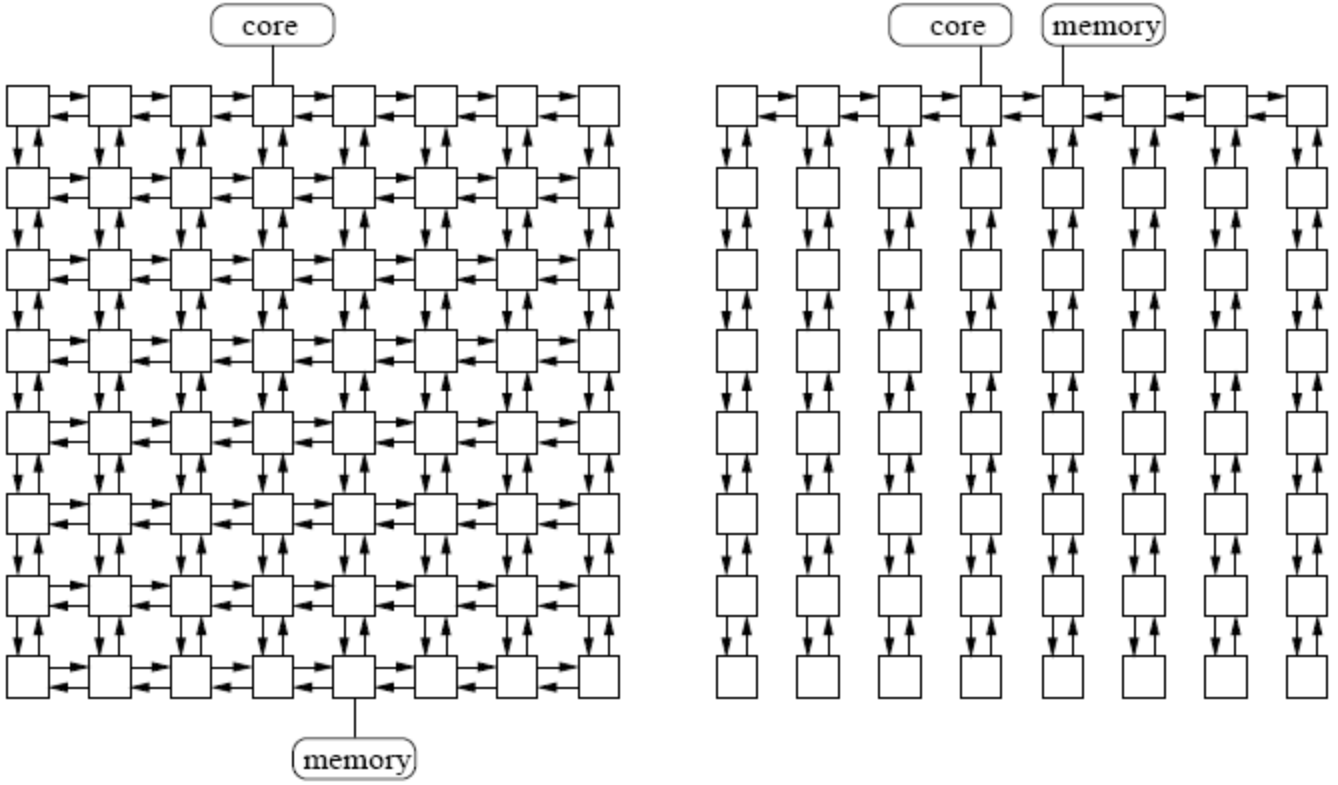
3D Designs, Li et al., ISCA'06





- D-NUCA: first search in cylinder, then multicast search everywhere
- Data is migrated close to requester, but need not jump across layers

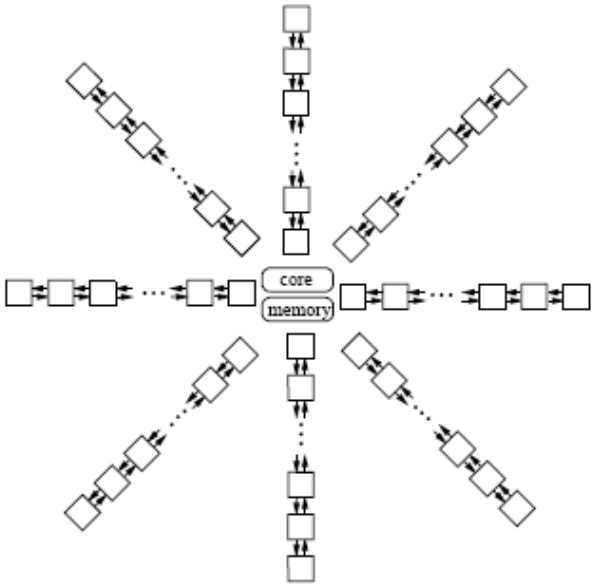
Halo Network, Jin et al., HPCA'07



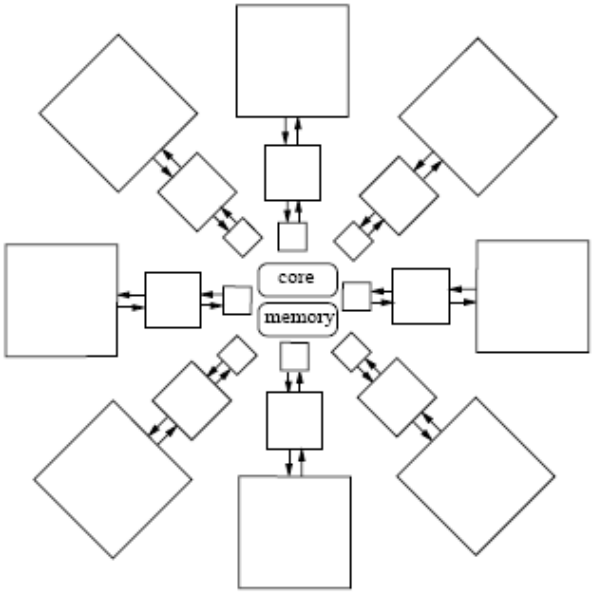
(a) Mesh (b) Simplified Mesh

- D-NUCA: Sets are distributed across columns;
  Ways are distributed across rows

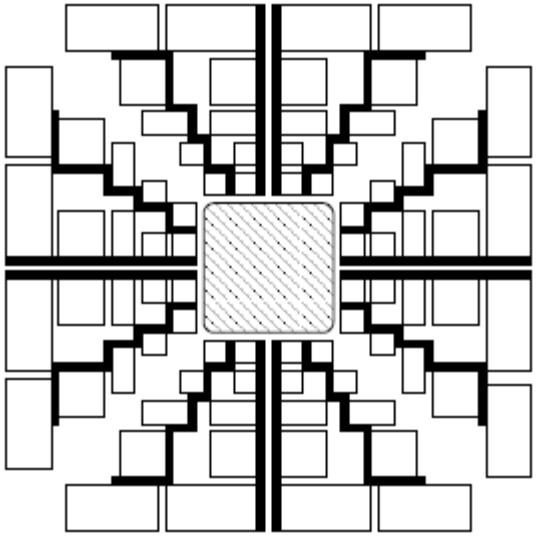
Halo Network



(c) Halo Constructed with Uniform Size Banks

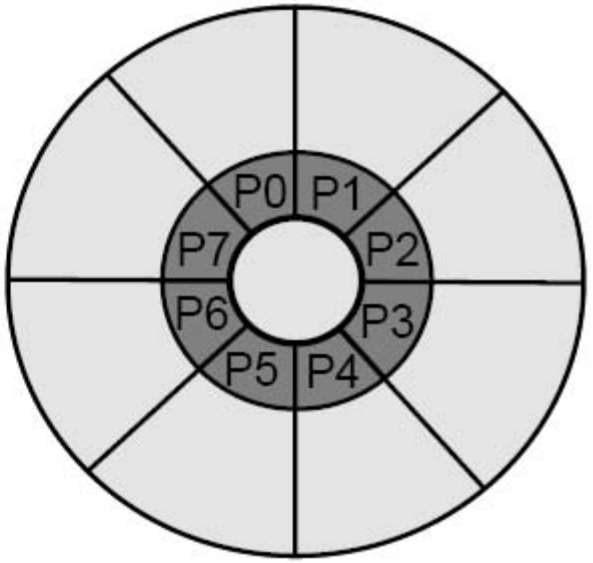


(d) Halo Constructed wit Non-uniform Size Banks



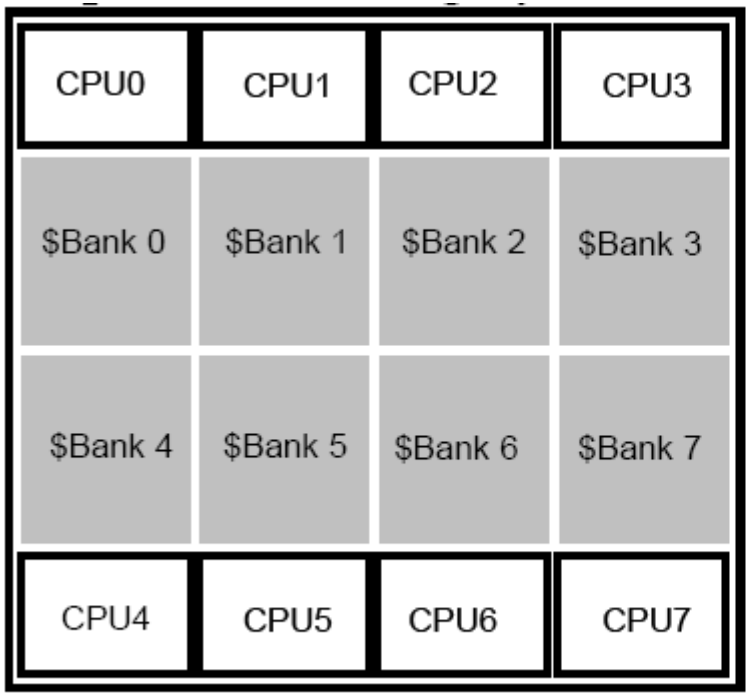
Nahalal, Guz et al., CAL'07

(a) Aerial view of Nahalal Village.

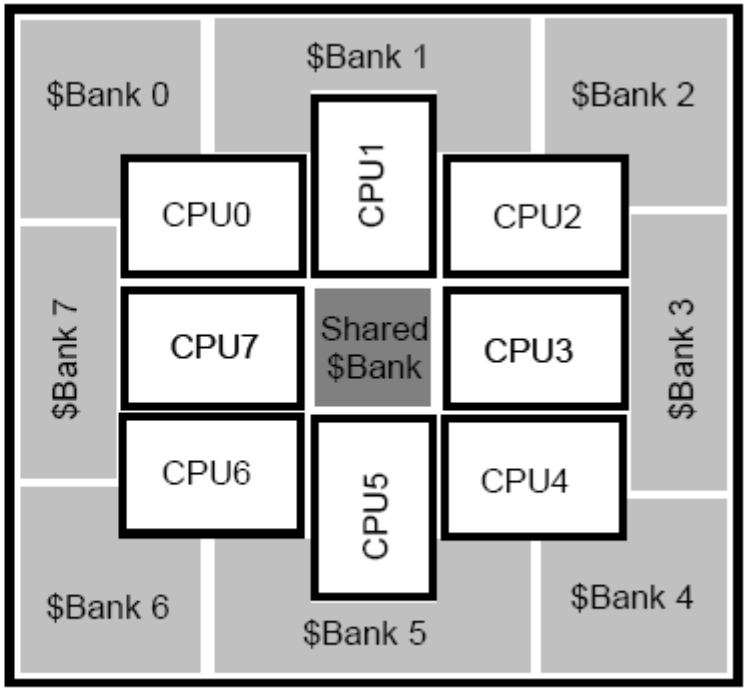


(b) CMP conceptual layout scheme.

Nahalal



(a) CIM layout.



(b) Nahalal layout.

- Block is initially placed in core's private bank and then swapped into the shared bank if frequently accessed by other cores
- Parallel search across all banks

Title

- Bullet