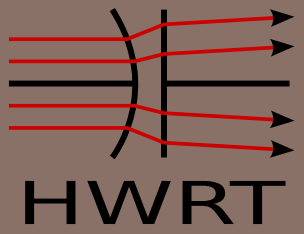
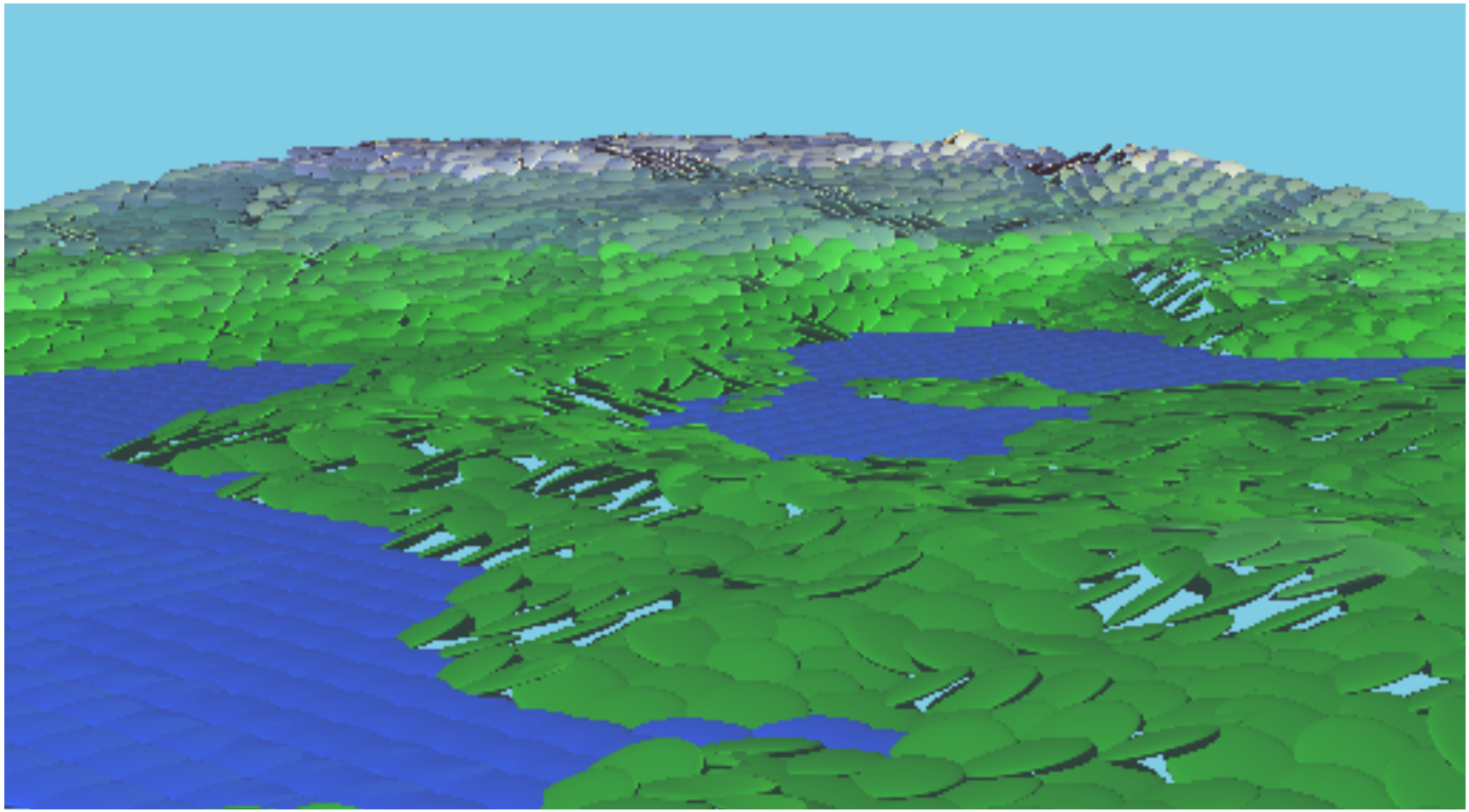


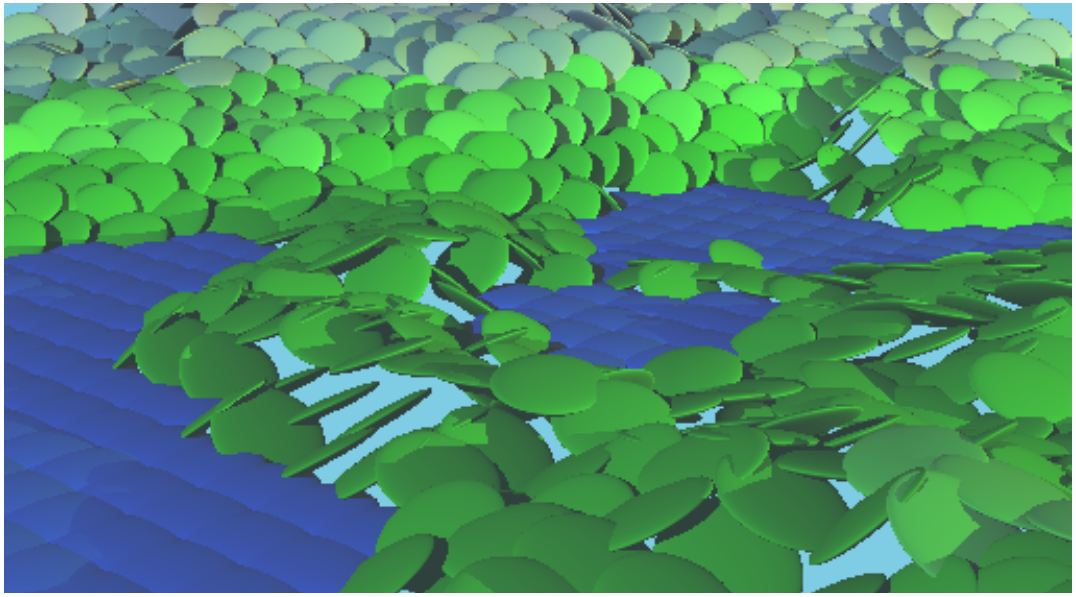
CS 6958
LECTURE 12
WRAP-UP CACHES

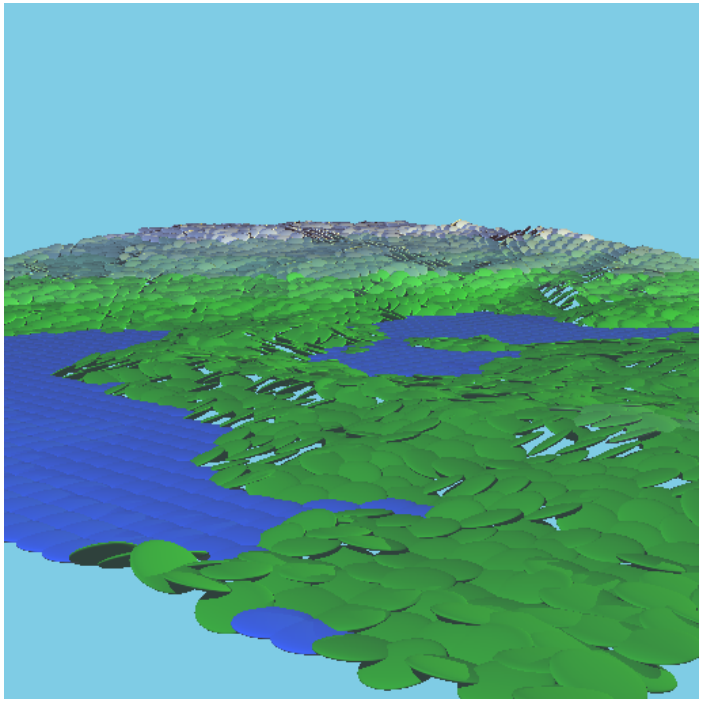
February 19, 2014

Creative



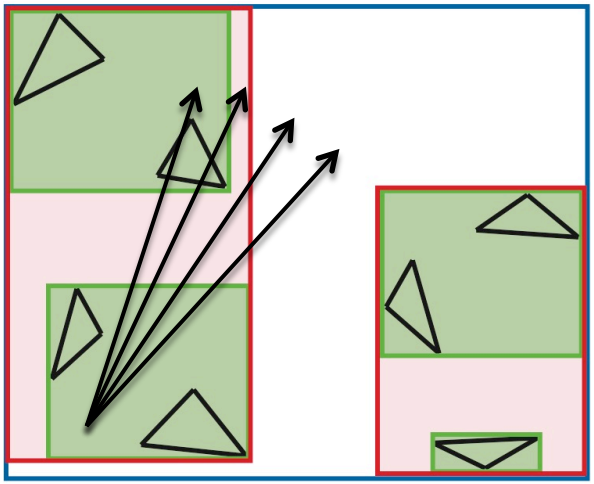
Creative



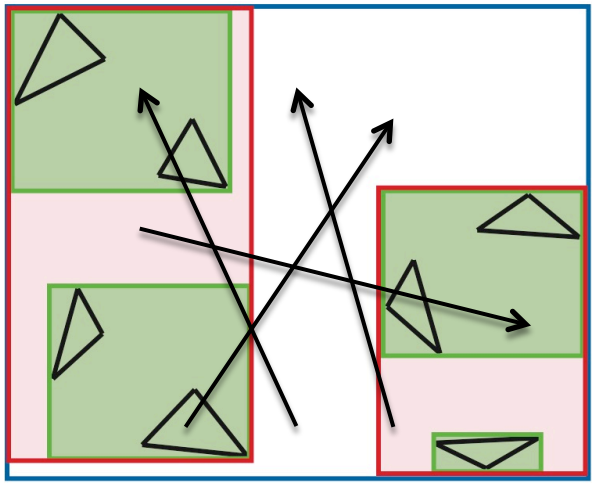


Ray Coherence

- Processing coherent rays simultaneously results in data locality
  - Lots of research involving collecting coherent rays
  - More on this later



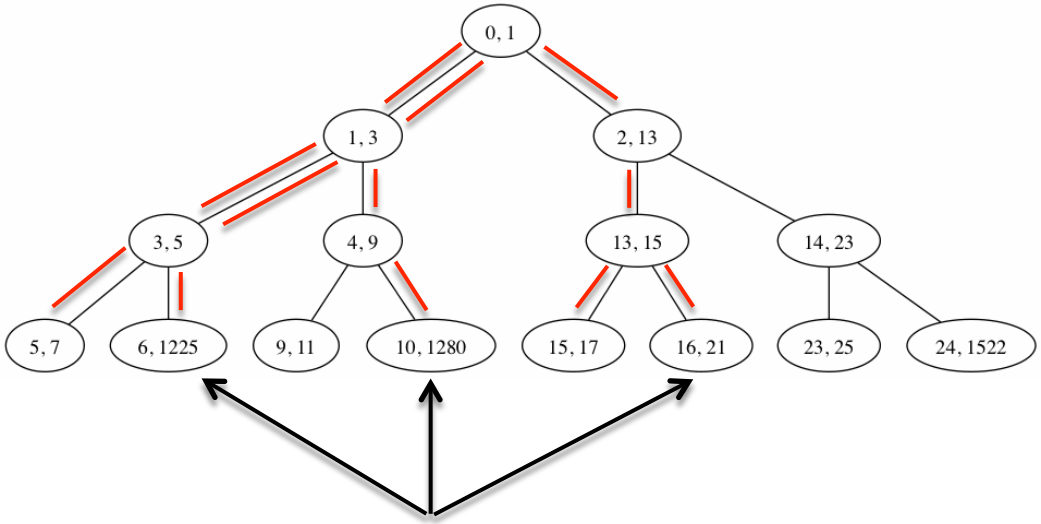
Coherent



Incoherent

Many-Core Shared Caches

All processed simultaneously



Suppose each of these nodes map to the same cache line (but different tag)

Line Size

- How big should lines be?
  - 1 word (4 bytes)
    - equivalent to larger RF
  - 64B
    - Typical (but seems pretty small)
  - Why not 512B, 1KB?

Line Size

- Number of lines = cache size / line size
  - What if only 1 line?
  - Data access usually only contiguous to certain extent

    (8, 16 words at a time?)
- Especially true for tree traversal
  - More lines → lower probability of conflict

Overfill / Underfill

- Overfill
  - Transferring too much data from L1, L2, DRAM
  - Locality only goes so far
  - Wastes a lot of energy, occupies DRAM channels
- Underfill
  - Transferring not enough data from L2, DRAM
  - Doesn't amortize expensive activation overheads
- Getting the right balance is tricky
  - Very rarely do we transfer exactly what we need

LOAD Stalls

- Data dependence stalls
  - Variable latency (1 – ??)
  - With --disable-usimm, latency is function of hit rate

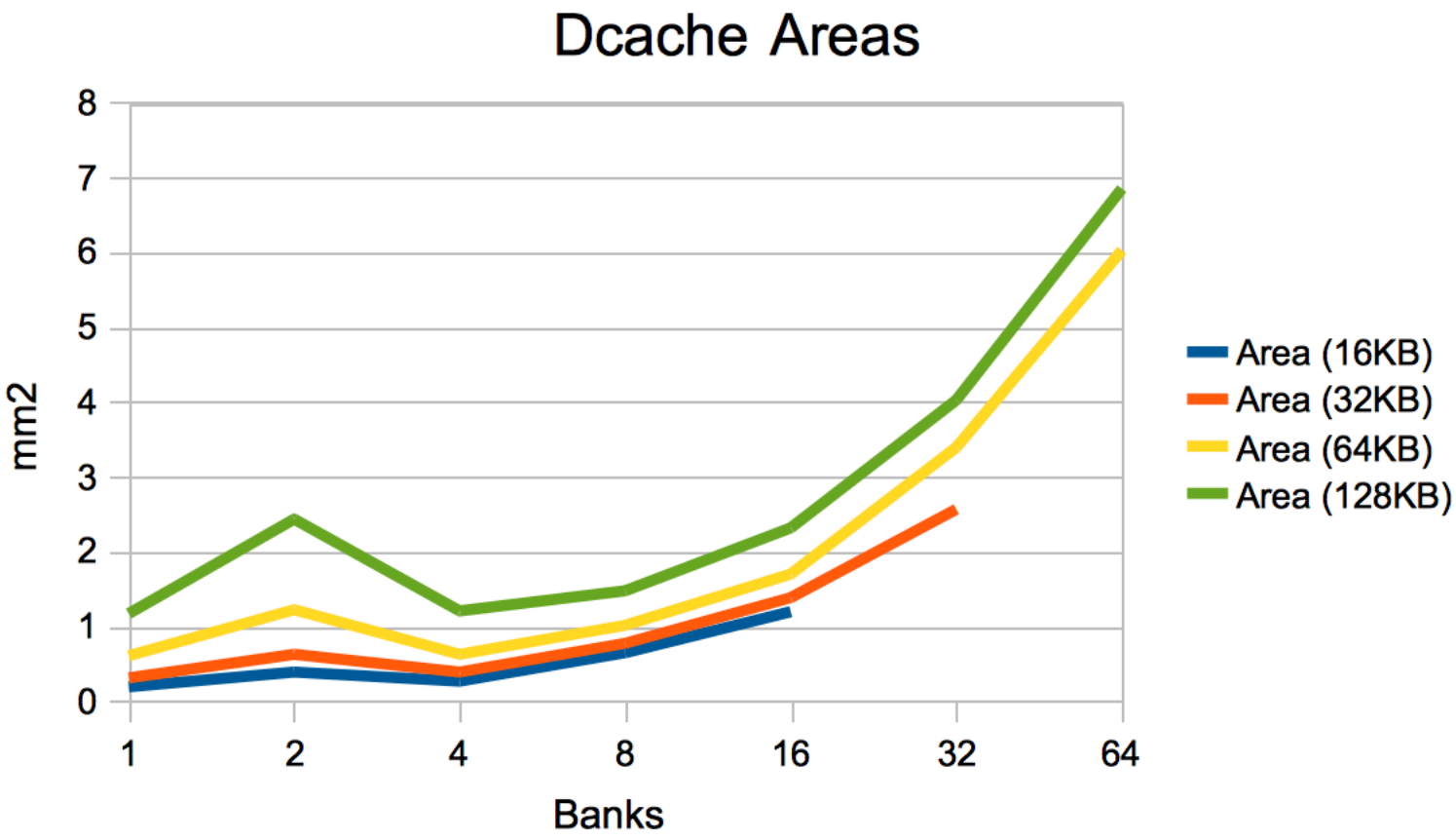
(32 threads)	**4KB**	**32KB**
Thread issue rate	53%	69%
Data Stalls (LOAD)	76M	18M

- Resource conflicts
  - Two threads trying to read same bank

(32 threads)	**1 bank**	**8 banks**
Thread issue rate	30%	69%
Resource conflicts (LOAD)	268M	1M

Cache Areas

- Function of capacity and num banks



Caches (config-file)

- L1 / L2

L1	1	8192	4	4
name	latency	capacity (words)	banks	log_2(linesize) (words)

Example is 32KB with 64B line size

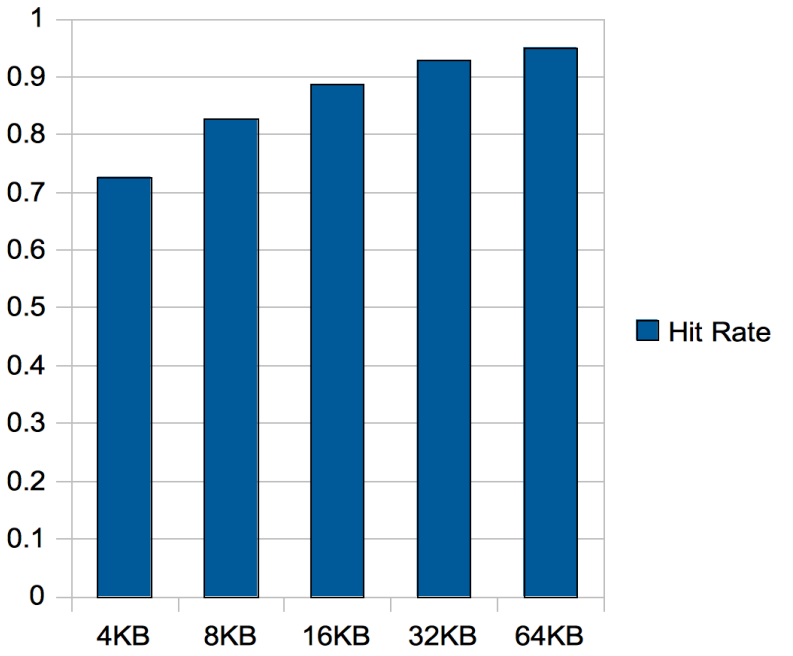
Cache Specifications

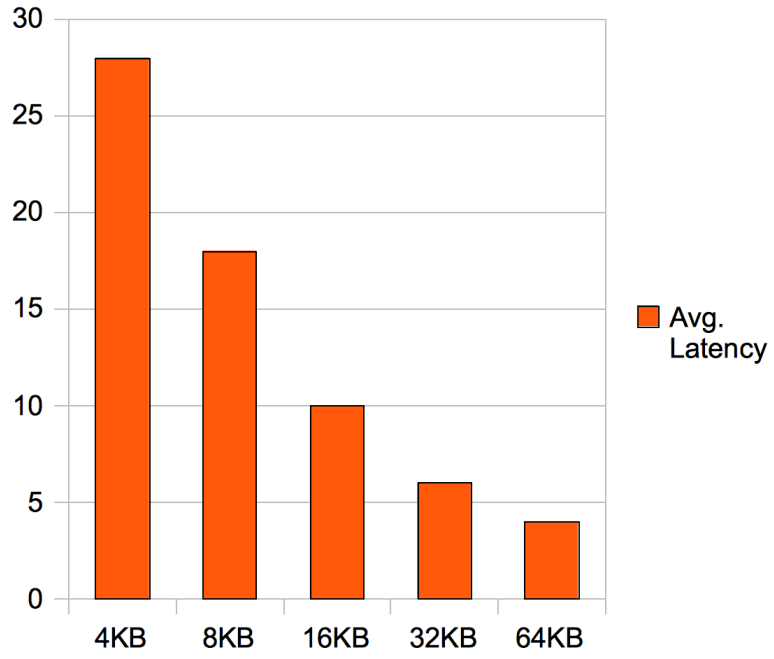
- samples/configs/dcacheparams.txt
  - All reasonable cache capacity/numbanks/linesize configurations
  - Some combinations not feasible and don't exist
  - Specified in bytes, not words!

- Area, energy estimates using Cacti
  - http://www.hpl.hp.com/research/cacti/

L1 Hit Rates

- Diminishing returns?
  - Not exactly





Hit Rates

- What's the difference between 98% and 99%

Hit Rates

- What's the difference between 98% and 99%
  - How many fewer reads make it past the cache?
  - ½

- 0% → 10% == 10% better
- 70% → 80% == 33% better

Hit Rates (L1 + L2)

- What is the difference between:
  - L1: 98% → 99%

  Vs.

  - L1: 98% + L2: 50%

Hit Rates (L1 + L2)

- What is the difference between:
  - L1: 98% → 99%

  Vs.
  - L1: 98% + L2: 50%

- Which is easier to achieve?
  - In terms of:
  - design
  - area
  - energy

Cache Statistics

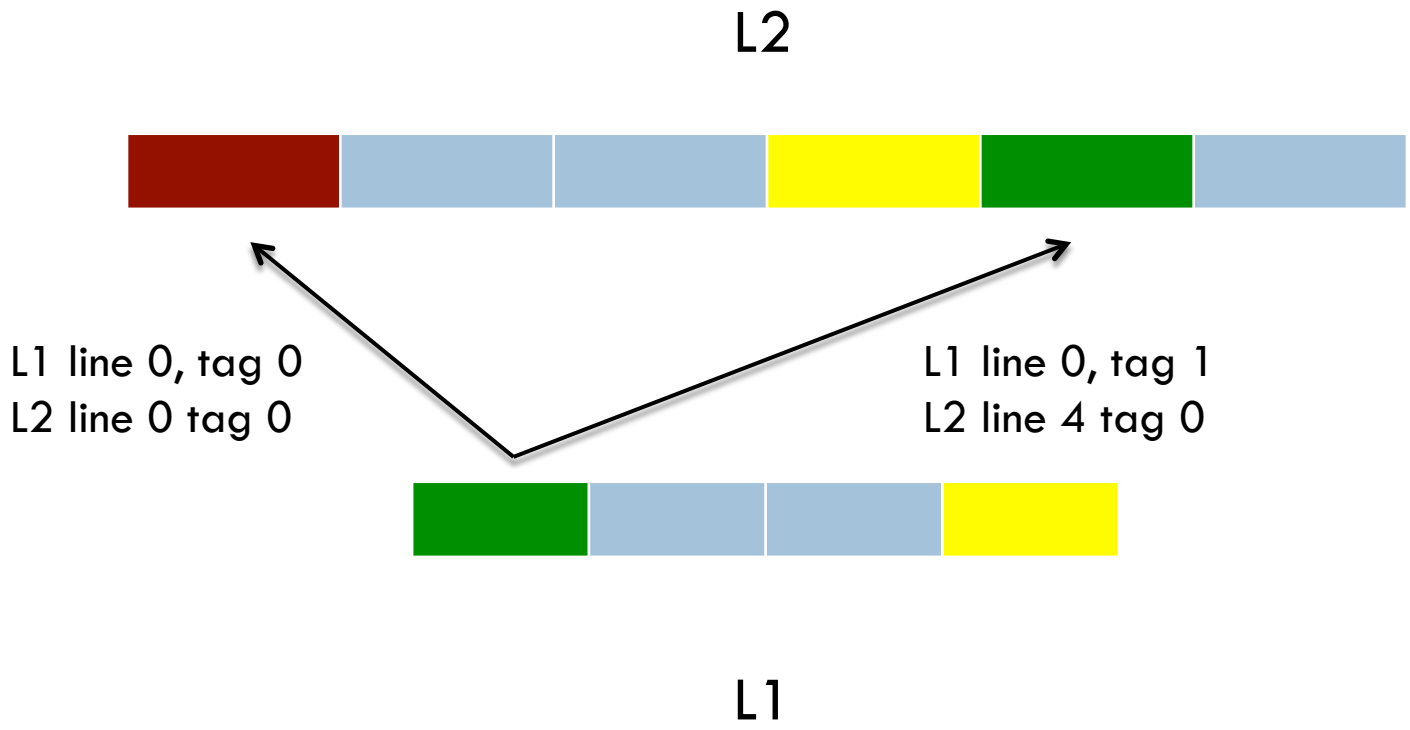
System-wide L1 stats (sum of all TMs):

L1 accesses:	14232064
L1 hits:	13630310
L1 misses:	601754
L1 bank conflicts:	761313
L1 stores:	49152
L1 hit rate:	0.957718
Hit under miss:	357529

Doesn't include hit under miss
(Hit + H.U.M. rate = 98.3%)

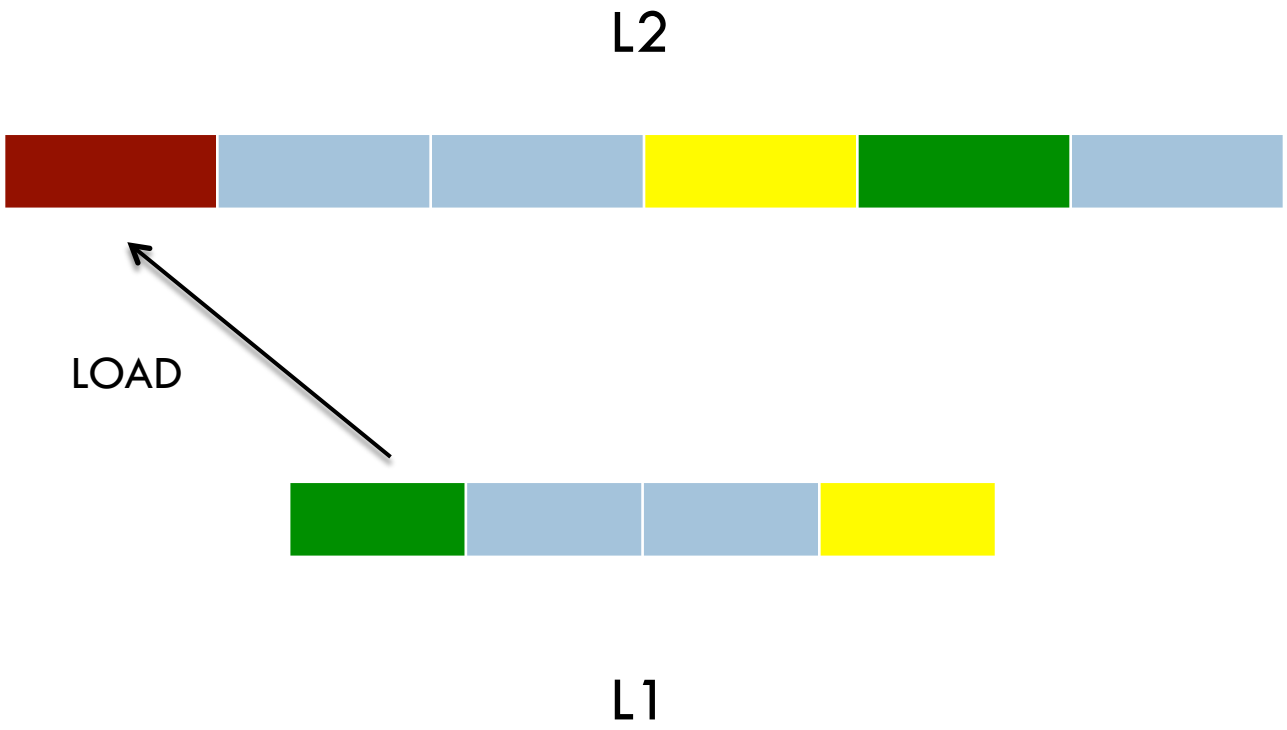
L1 → L2 Interaction

- For L2 to catch extra misses, they must contain different lines
  - L2 much larger: address → line mapping changes



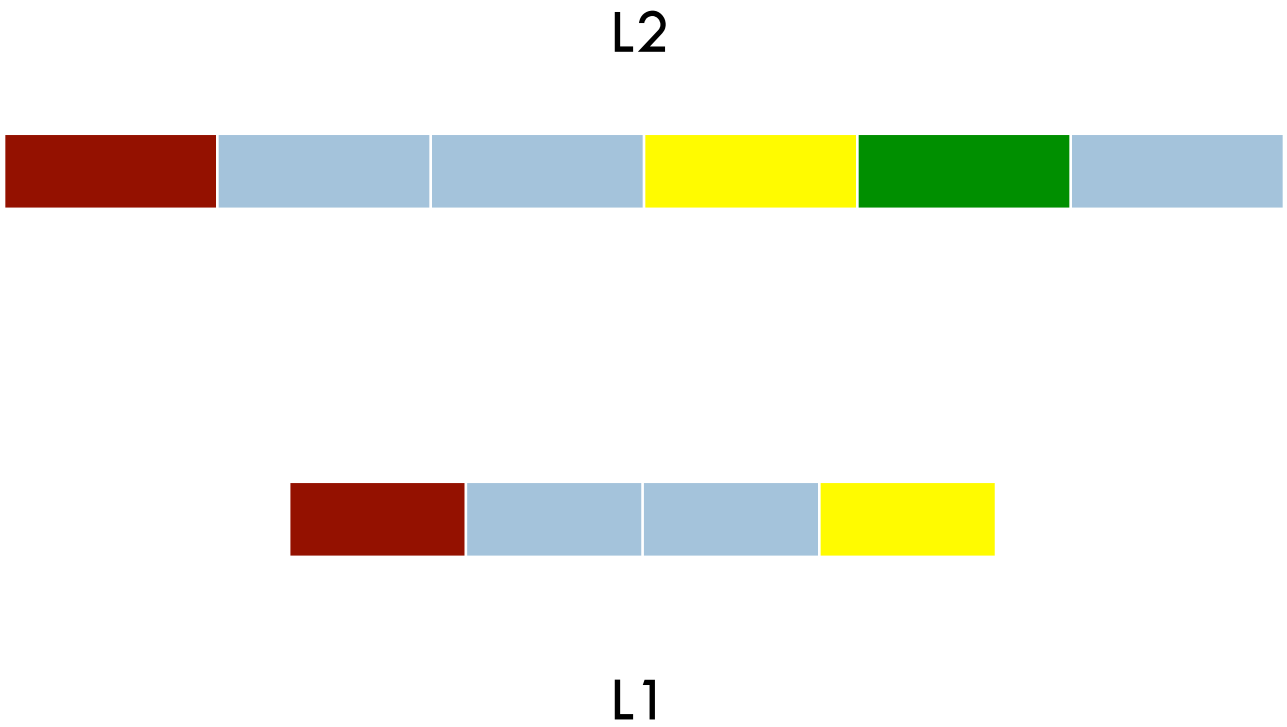
L1 → L2 Interaction

- If we must evict green line from L1, it is not completely thrown away



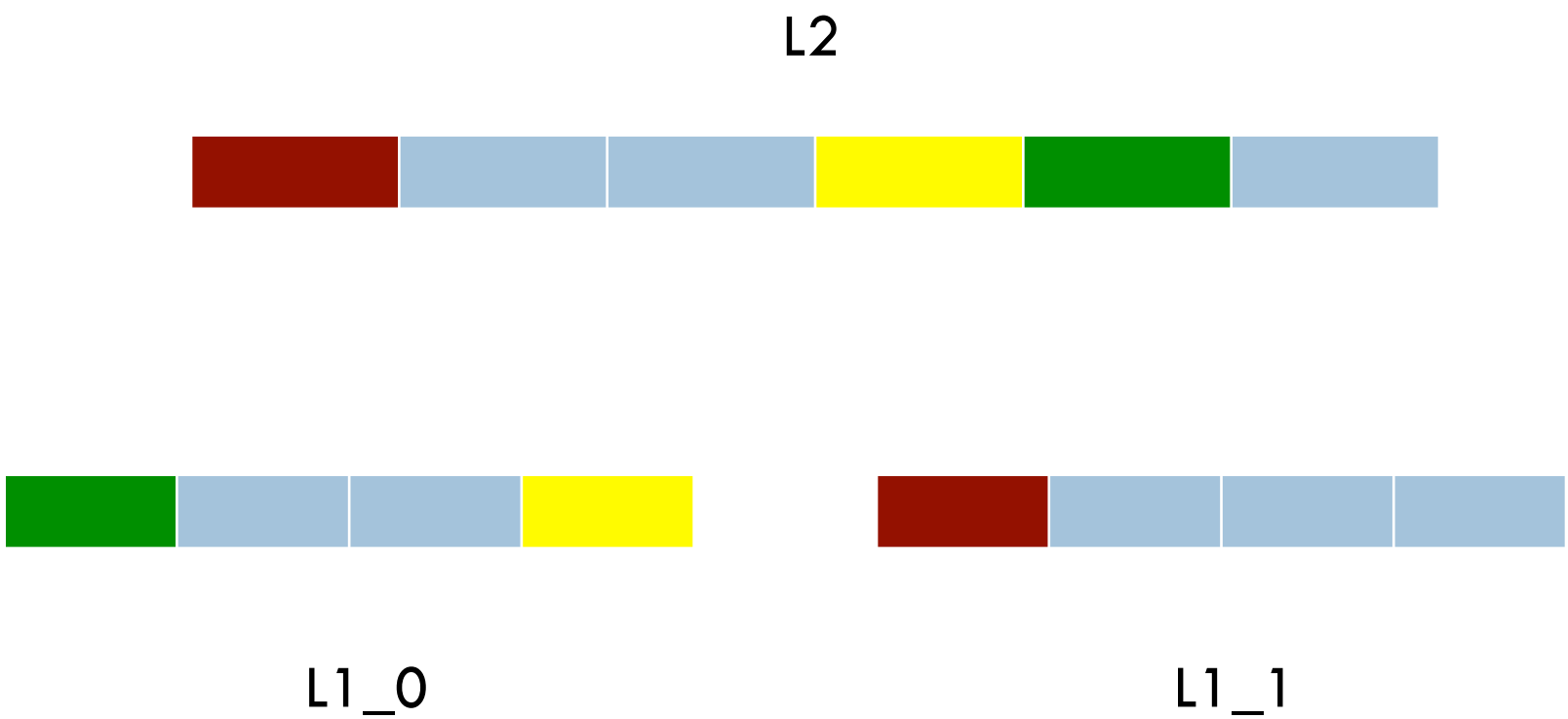
L1 → L2 Interaction

- Extra line (green) is still saved if needed later
- Cache hierarchy almost like extra associativity



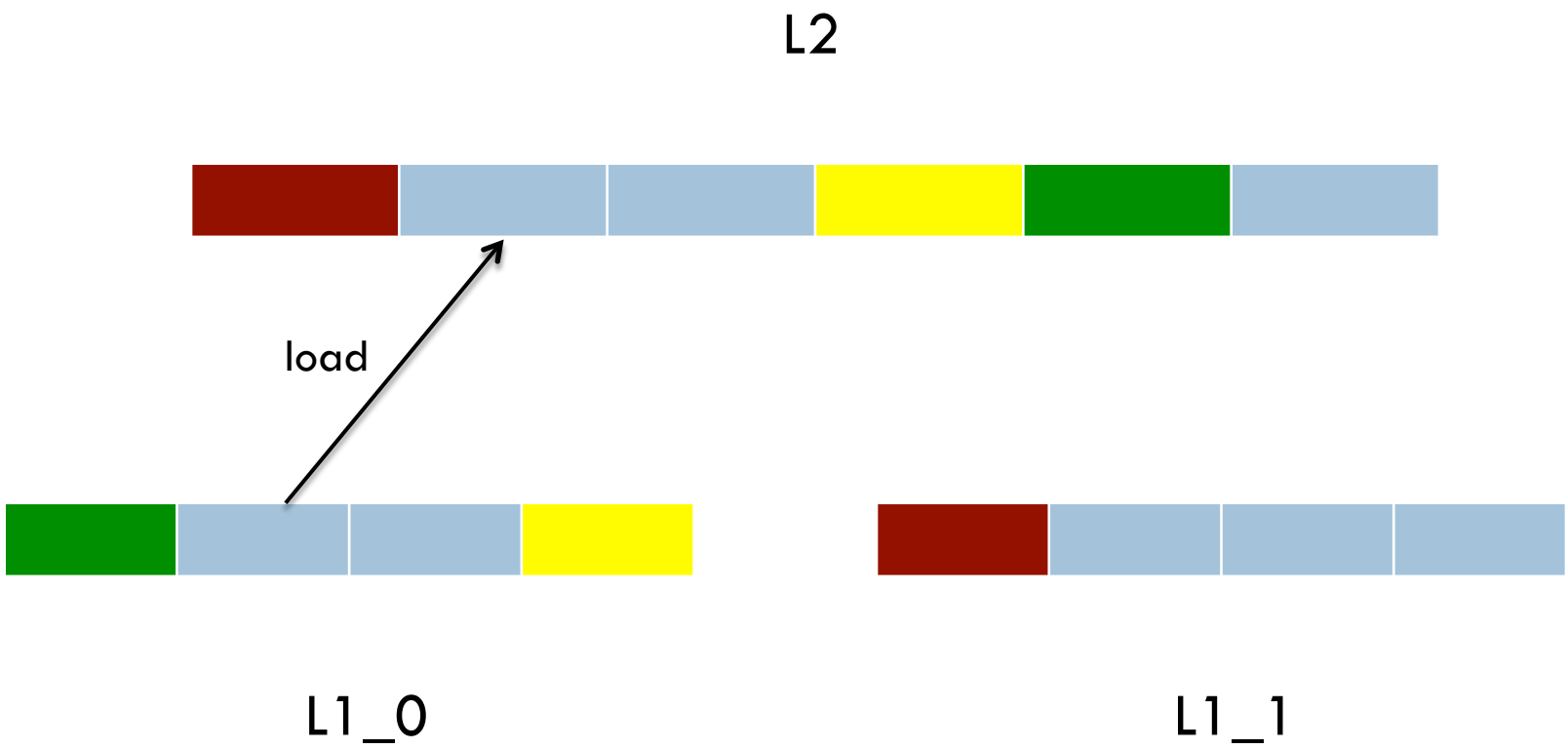
L1 → L2 Interaction

- L2 usually shared by multiple L1s
  - Non-exclusive
  - Lines contained in L2 *may* also be contained in L1



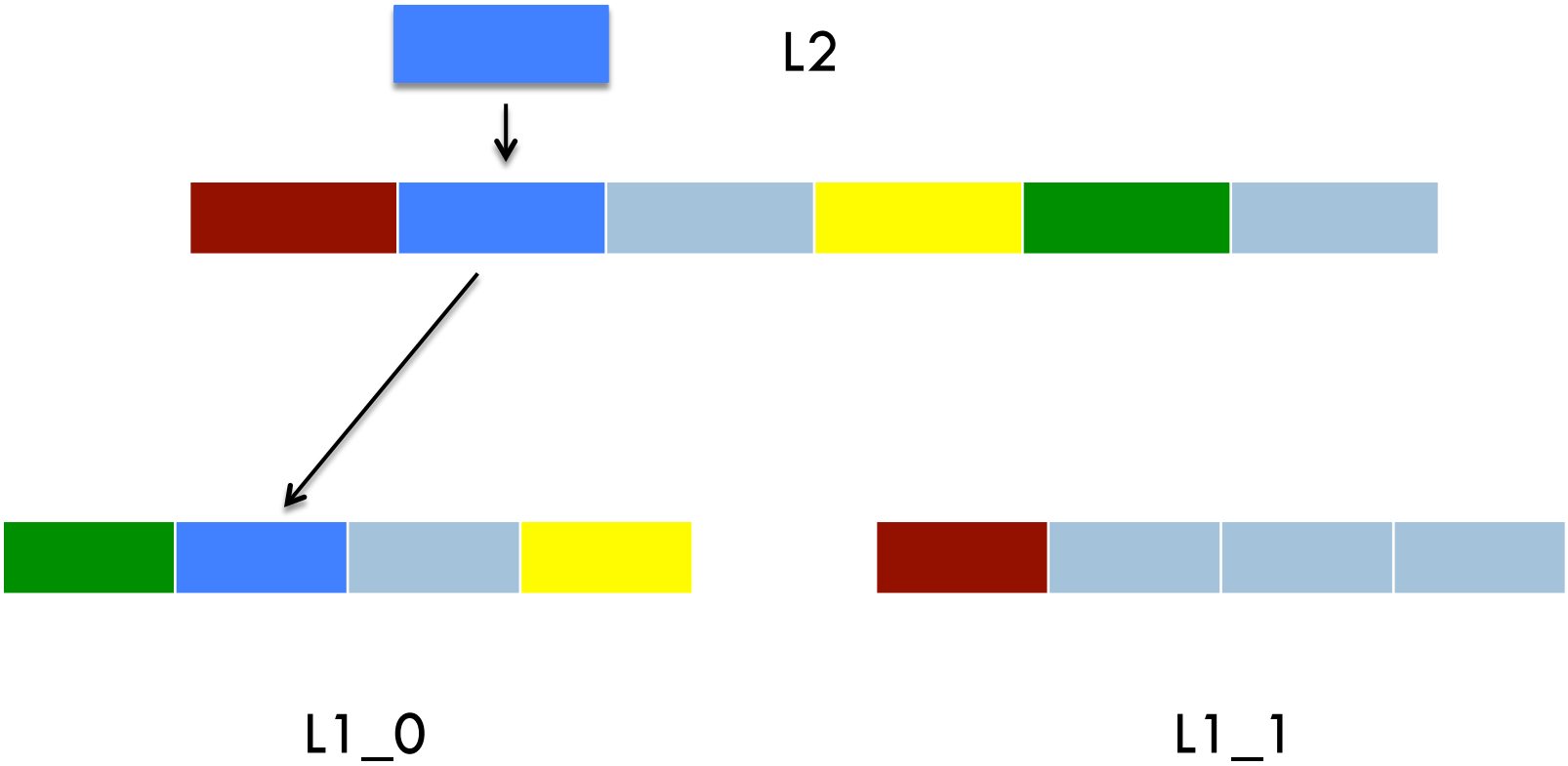
L1 → L2 Interaction

- Shared cache interaction gets more intricate



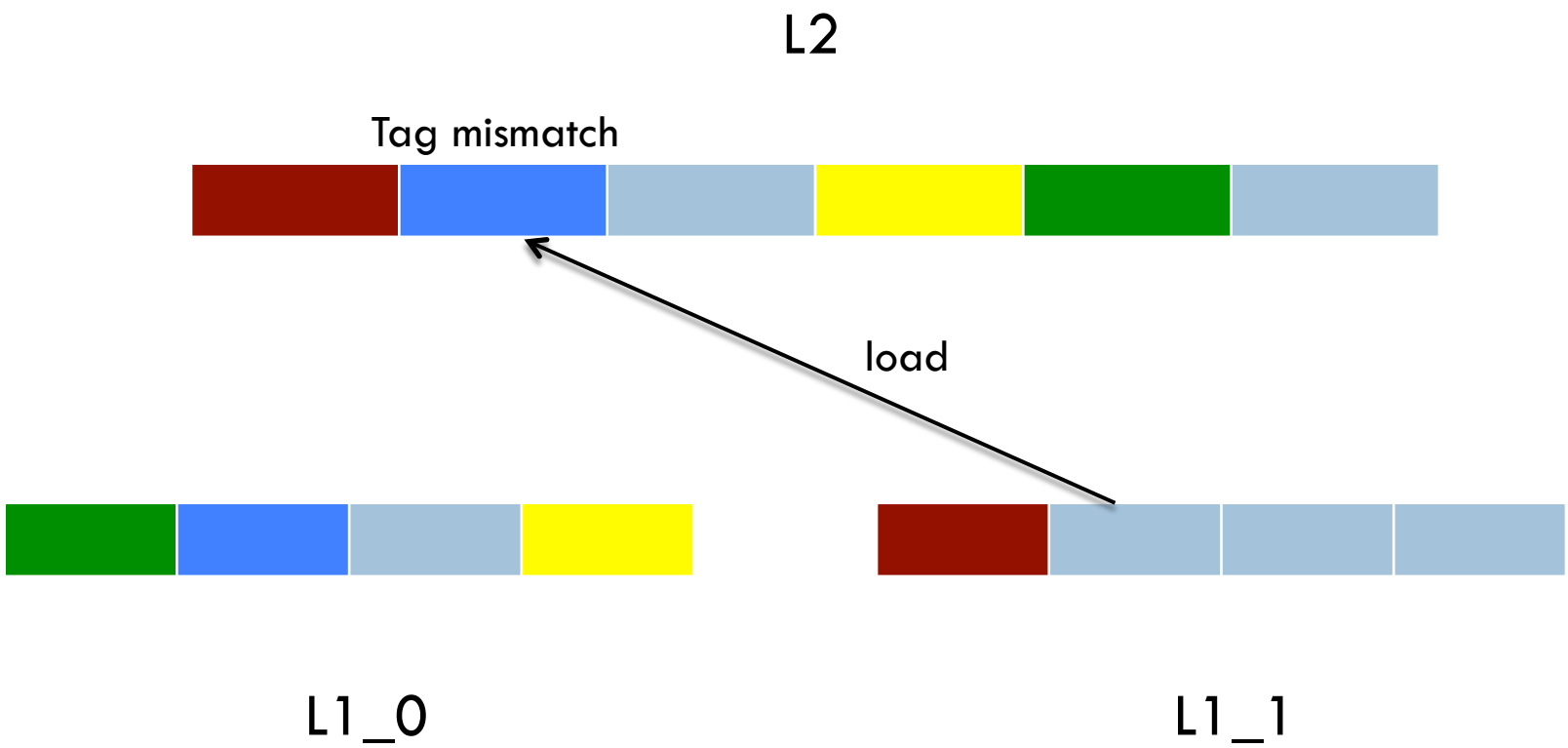
L1 → L2 Interaction

- L1_1 may benefit from someone else's fetch



L1 → L2 Interaction

- If they disagree, L1_0 keeps its own copy



L1 → L2 Interaction

- L2 lines replicated in at least one L1
- L1 lines not necessarily in L2