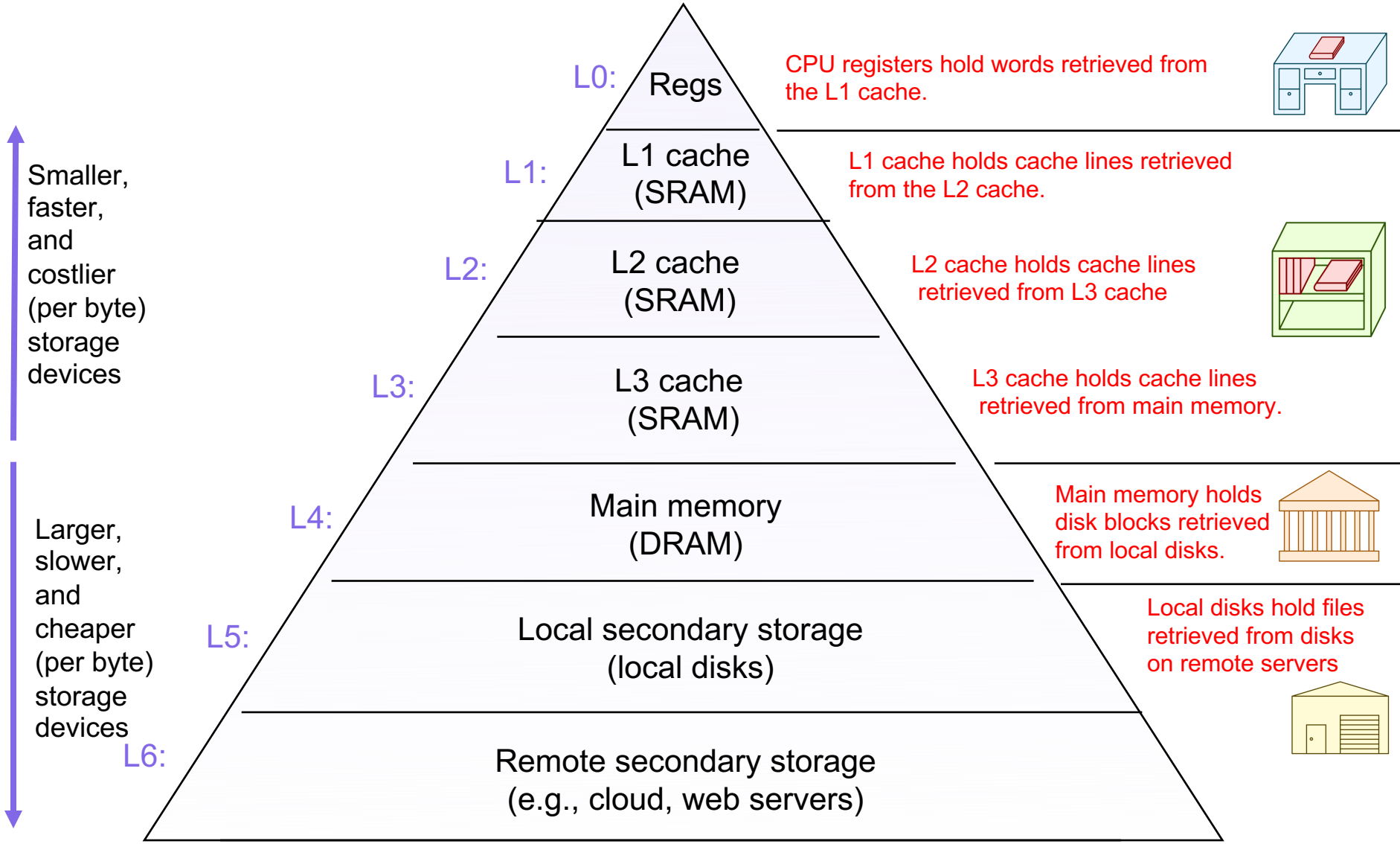
Lecture 15: Caches (cont'd)

CS 105

Spring 2021

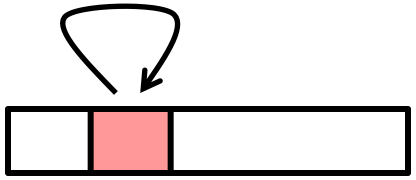
Review: Memory Hierarchy



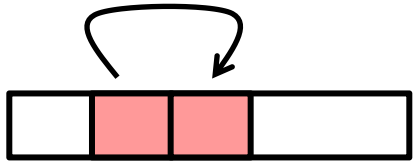
Review: Principle of Locality

Programs tend to use data and instructions with addresses near or equal to those they have used recently

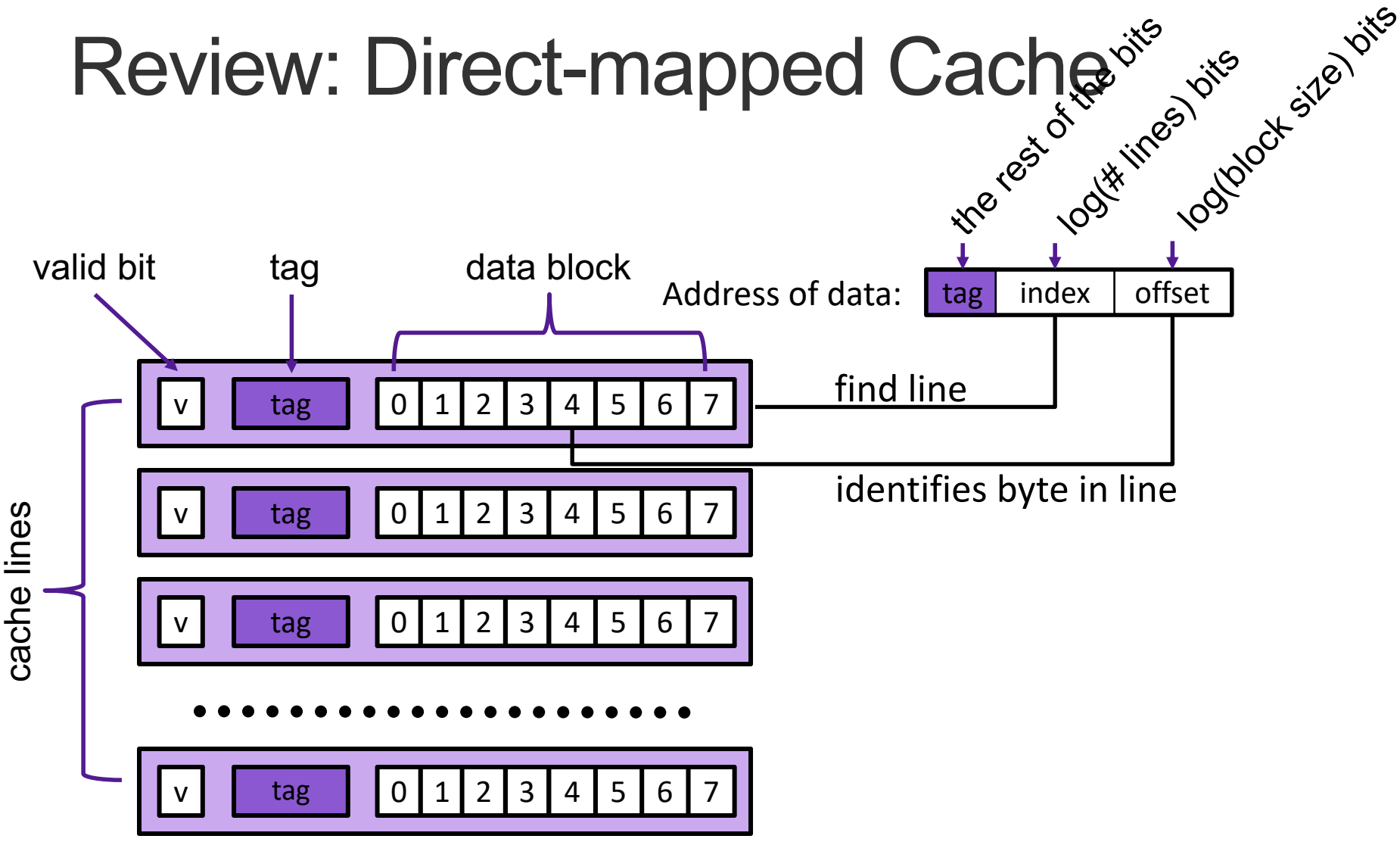
- Temporal locality:
  - Recently referenced items are likely to be referenced again in the near future



- Spatial locality:
  - Items with nearby addresses tend to be referenced close together in time



Review: Direct-mapped Cache

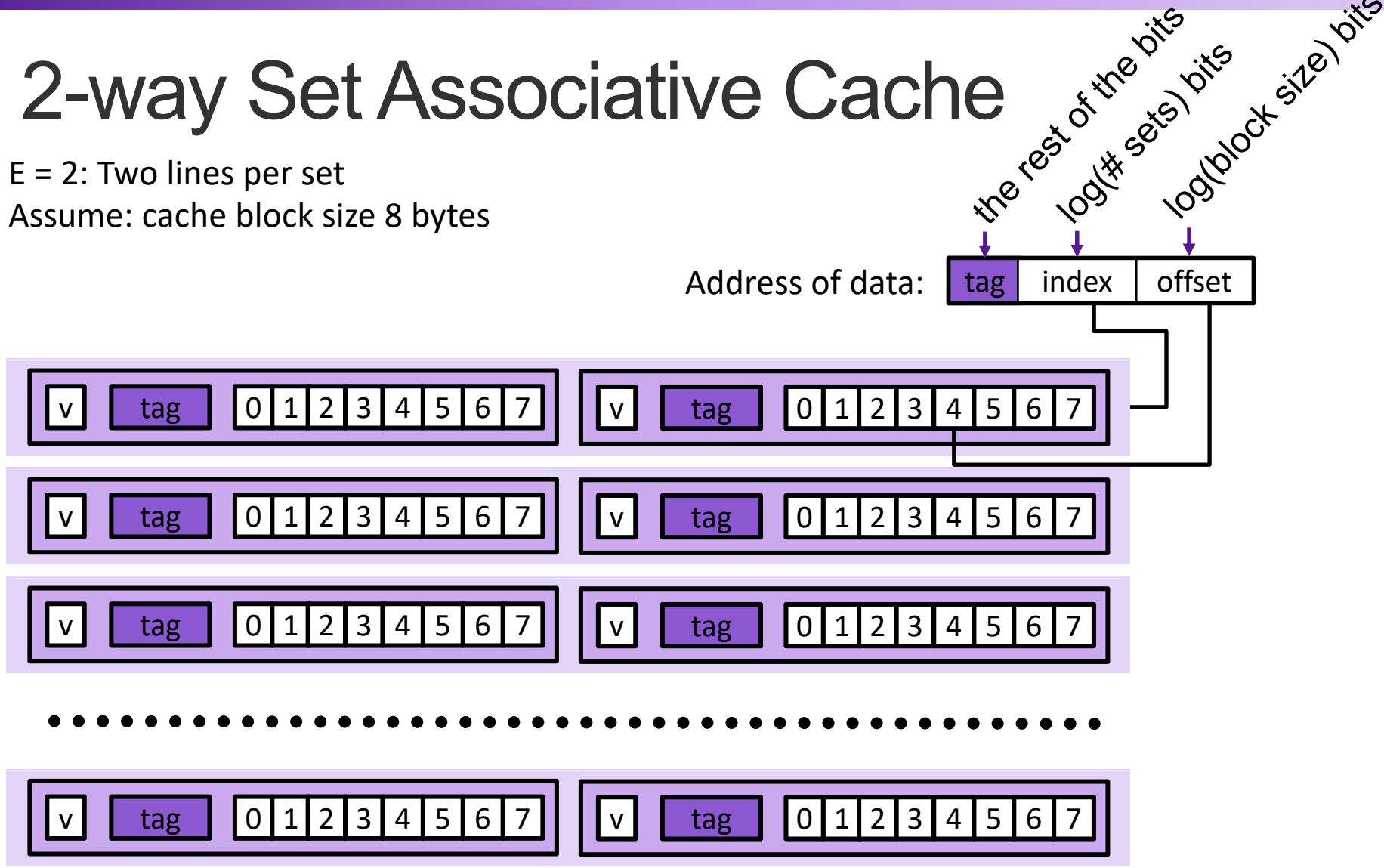


How well does this take advantage of spacial locality?
How well does this take advantage of temporal locality?

2-way Set Associative Cache

E = 2: Two lines per set

Assume: cache block size 8 bytes

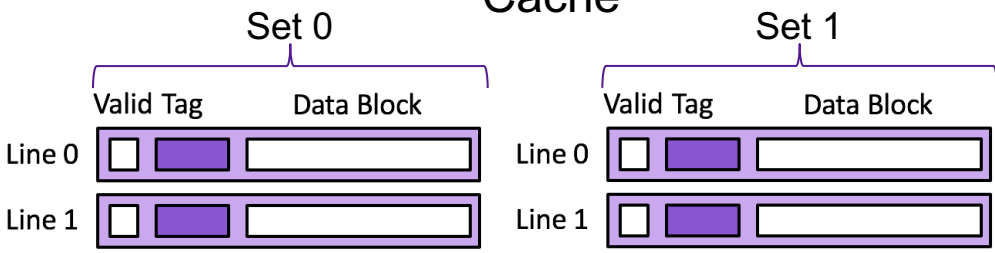


Exercise 1: 2-way Set Associative Cache

Memory

0x14	18
0x10	17
0x0c	16
0x08	15
0x04	14
0x00	13

Cache



Assume 8 byte data blocks

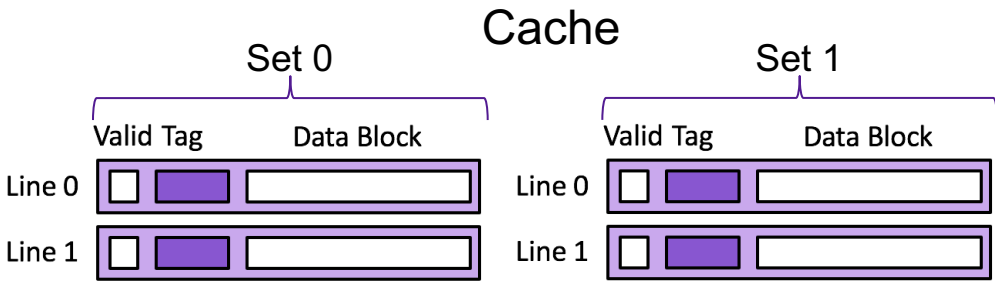
Access	tag	idx	off	h/m
rd 0x00				
rd 0x04				
rd 0x14				
rd 0x00				
rd 0x04				
rd 0x14				

Set 0						Set 1					
Line 0			Line 1			Line 0			Line 1		
0	0	47 48	0	1	47 48	0	0	47 48	0	1	47 48

Exercise 1: 2-way Set Associative Cache

Memory

0x14	18
0x10	17
0x0c	16
0x08	15
0x04	14
0x00	13



Assume 8 byte data blocks

Access	tag	idx	off	h/m
rd 0x00	0000	0	000	m
rd 0x04	0000	0	100	h
rd 0x14	0001	0	100	m
rd 0x00	0000	0	000	h
rd 0x04	0000	0	100	h
rd 0x14	0001	0	100	h
rd 0x20	0010	0	000	m

Set 0							Set 1						
Line 0				Line 1			Line 0				Line 1		
0	0	47	48	0	1	47 48	0	0	47	48	0	1	47 48
1	0	13	14										
				1	1	17 18							

Eviction from the Cache

On a cache miss, a new block is loaded into the cache

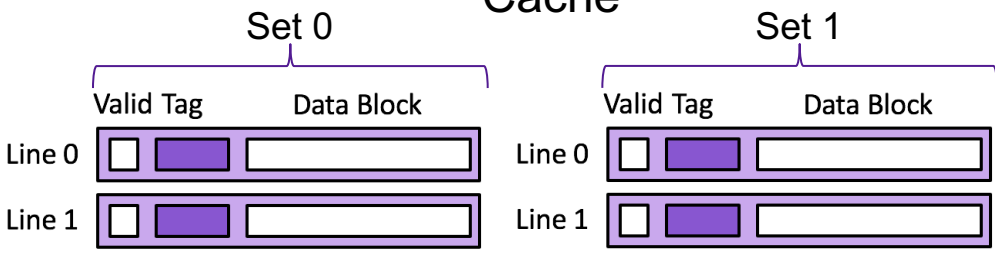
- Direct-mapped cache: A valid block at the same location must be evicted—no choice
- Associative cache: If all blocks in the set are valid, one must be evicted
  - Random policy
  - FIFO
  - LIFO
  - Least-recently used; requires extra data in each set
  - Most-recently used; requires extra data in each set
  - Most-frequently used; requires extra data in each set

Exercise 2: Cache Eviction

Memory

0x14	18
0x10	17
0x0c	16
0x08	15
0x04	14
0x00	13

Cache



Assume 8 byte data blocks, LRU eviction

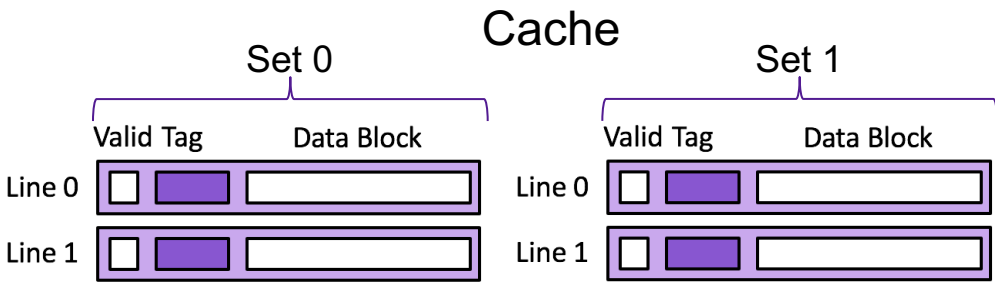
Access	tag	idx	off	h/m
rd 0x00	0000	0	000	m
rd 0x04	0000	0	100	h
rd 0x14	0001	0	100	m
rd 0x00	0000	0	000	h
rd 0x04	0000	0	100	h
rd 0x14	0001	0	100	h
rd 0x20	0010	0	000	m

Set 0							Set 1						
Line 0			Line 1				Line 0			Line 1			
0	0	47 48	0	1	47 48		0	0	47 48	0	1	47 48	
1	0	13 14											
			1	1	17 18								

Exercise 2: Cache Eviction

Memory

0x14	18
0x10	17
0x0c	16
0x08	15
0x04	14
0x00	13



Assume 8 byte data blocks, LRU eviction

Access	tag	idx	off	h/m
rd 0x00	0000	0	000	m
rd 0x04	0000	0	100	h
rd 0x14	0001	0	100	m
rd 0x00	0000	0	000	h
rd 0x04	0000	0	100	h
rd 0x14	0001	0	100	h
rd 0x20	0010	0	000	m

Set 0							Set 1						
Line 0			Line 1				Line 0			Line 1			
0	0	47 48	0	1	47 48		0	0	47 48	0	1	47 48	
1	0	13 14											
			1	1	17 18								
1	2	21 22											

Caching Organization Summarized

- A cache consists of lines
- A **line** contains
  - A **block** of bytes, the data values from memory
  - A **tag**, indicating where in memory the values are from
  - A **valid bit**, indicating if the data are valid
- Lines are organized into sets
  - **Direct-mapped cache:** one line per set
  - **k-way associative cache:** k lines per set
  - **Fully associative cache:** all lines in one set

Caching Vocabulary

- **Size:** the total number of bytes that can be stored in the cache

- **Cache Hit:** the desired value is in the cache and returned quickly
- **Cache Miss:** the desired value is not in the cache and must be fetched from a more distant cache (or ultimately from main memory)

- **Miss rate:** the fraction of accesses that are misses

- **Hit time:** the time to process a hit
- **Miss penalty:** the *additional* time to process a miss

- **Average access time:** hit-time + miss-rate * miss-penalty

Categorizing Misses

- **Compulsory:** first-reference to a block
- **Capacity:** cache is too small to hold all of the data
- **Conflict:** collisions in a specific set

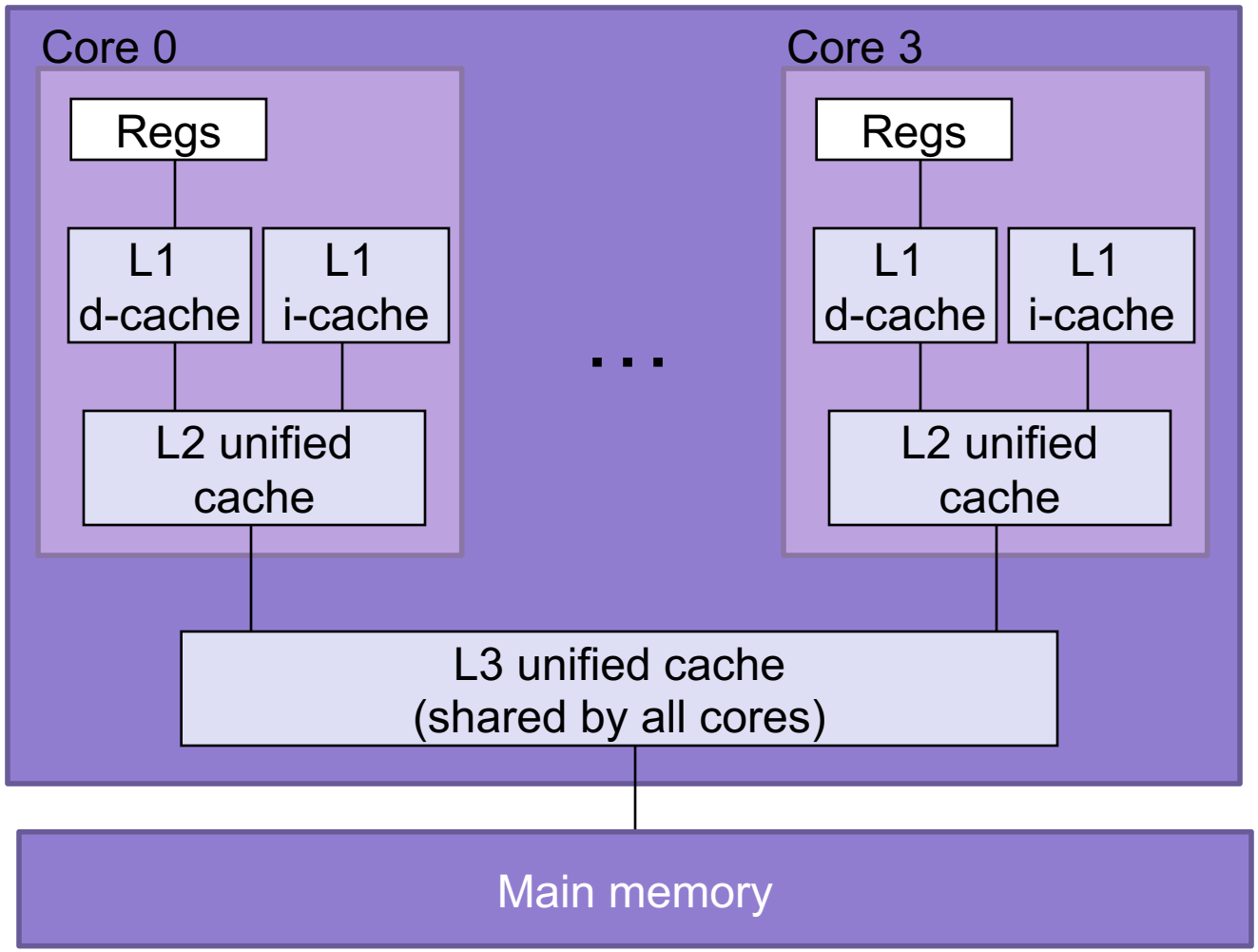
Average access time: hit-time + miss-rate * miss-penalty

Exercise 3: Categorizing Misses

- For each of the cache misses in Exercise 1, categorize that miss as (1) compulsory, (2) capacity, or (3) conflict

- Based on your categorizations, would you recommend (1) increasing the block size, (2) increasing the associativity, or (3) increasing the total cache size

Typical Intel Core i7 Hierarchy



L1 d-cache and i-cache:
32 KB, 8-way,
Access: 4 cycles

L2 unified cache:
256 KB, 8-way,
Access: 10 cycles

L3 unified cache:
8 MB, 16-way,
Access: 40-75 cycles

Block size: 64 bytes for all caches.

Caching and Writes

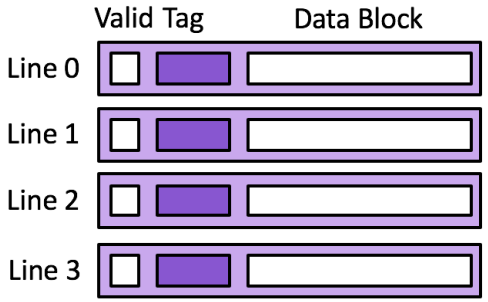
- What to do on a write-hit?
  - **Write-through:** write immediately to memory
  - **Write-back:** defer write to memory until replacement of line
    - Need a dirty bit (line different from memory or not)
- What to do on a write-miss?
  - **Write-allocate:** load into cache, update line in cache
    - Good if more writes to the location follow
  - **No-write-allocate:** writes straight to memory, does not load into cache
- Typical
  - Write-through + No-write-allocate
  - **Write-back + Write-allocate**

Exercise 4: Write-through + No-write-allocate

Memory

0x24	22
0x20	21
0x1c	20
0x18	19
0x14	18
0x10	17

Cache



Assume 4 byte data blocks

Access	tag	idx	off	h/m
rd 0x10				
wr 8,0x10				
wr 9,0x24				
rd 0x24				
rd 0x20				

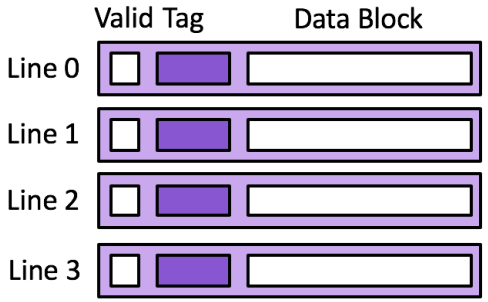
Line 0			Line 1			Line 2			Line 3			W
0	0	47	0	1	47	0	2	47	0	3	47	

Exercise 4: Write-through + No-write-allocate

Memory

0x24	9
0x20	21
0x1c	20
0x18	19
0x14	18
0x10	8

Cache



Assume 4 byte data blocks

Access	tag	idx	off	h/m
rd 0x10	0001	00	00	m
wr 8,0x10	0001	00	00	h
wr 9,0x24	0010	01	00	m
rd 0x24	0010	01	00	m
rd 0x20	0010	00	00	m

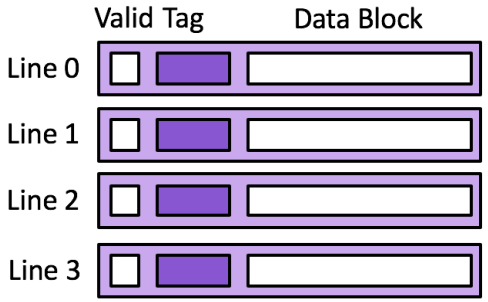
Line 0			Line 1			Line 2			Line 3			W
0	0	47	0	1	47	0	2	47	0	3	47	
1	1	17										N
1	1	8										Y
												Y
			1	2	9							N
1	2	21										N

Exercise 5: Write-back + Write-allocate

Memory

0x24	22
0x20	21
0x1c	20
0x18	19
0x14	18
0x10	17

Cache



Assume 4 byte data blocks

Access	tag	idx	off	h/m
rd 0x10				
wr 8,0x10				
wr 9,0x24				
rd 0x24				
rd 0x20				

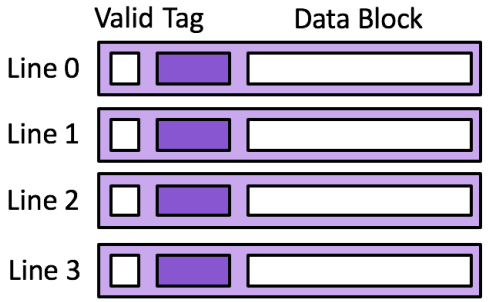
Line 0			Line 1			Line 2			Line 3			W
0	0	47	0	1	47	0	2	47	0	3	47	

Exercise 5: Write-back + Write-allocate

Memory

0x24	22
0x20	21
0x1c	20
0x18	19
0x14	18
0x10	8

Cache



Assume 4 byte data blocks

Access	tag	idx	off	h/m
rd 0x10	0001	00	00	m
wr 8,0x10	0001	00	00	h
wr 9,0x24	0010	01	00	m
rd 0x24	0010	01	00	h
rd 0x20	0010	00	00	m

Line 0			Line 1			Line 2			Line 3			W
0	0	47	0	1	47	0	2	47	0	3	47	
1	1	17										N
1	1	8										N
			1	2	9							N
												N
1	2	21										Y

Exercise 6: Feedback

1. Rate how well you think this recorded lecture worked
    1. Better than an in-person class
    2. About as well as an in-person class
    3. Less well than an in-person class, but you still learned something
    4. Total waste of time, you didn't learn anything
2. How much time did you spend on this video lecture (including time spent on exercises)?
3. Do you have any questions that you would like me to address in this week's problem session?
4. Do you have any other comments or feedback?