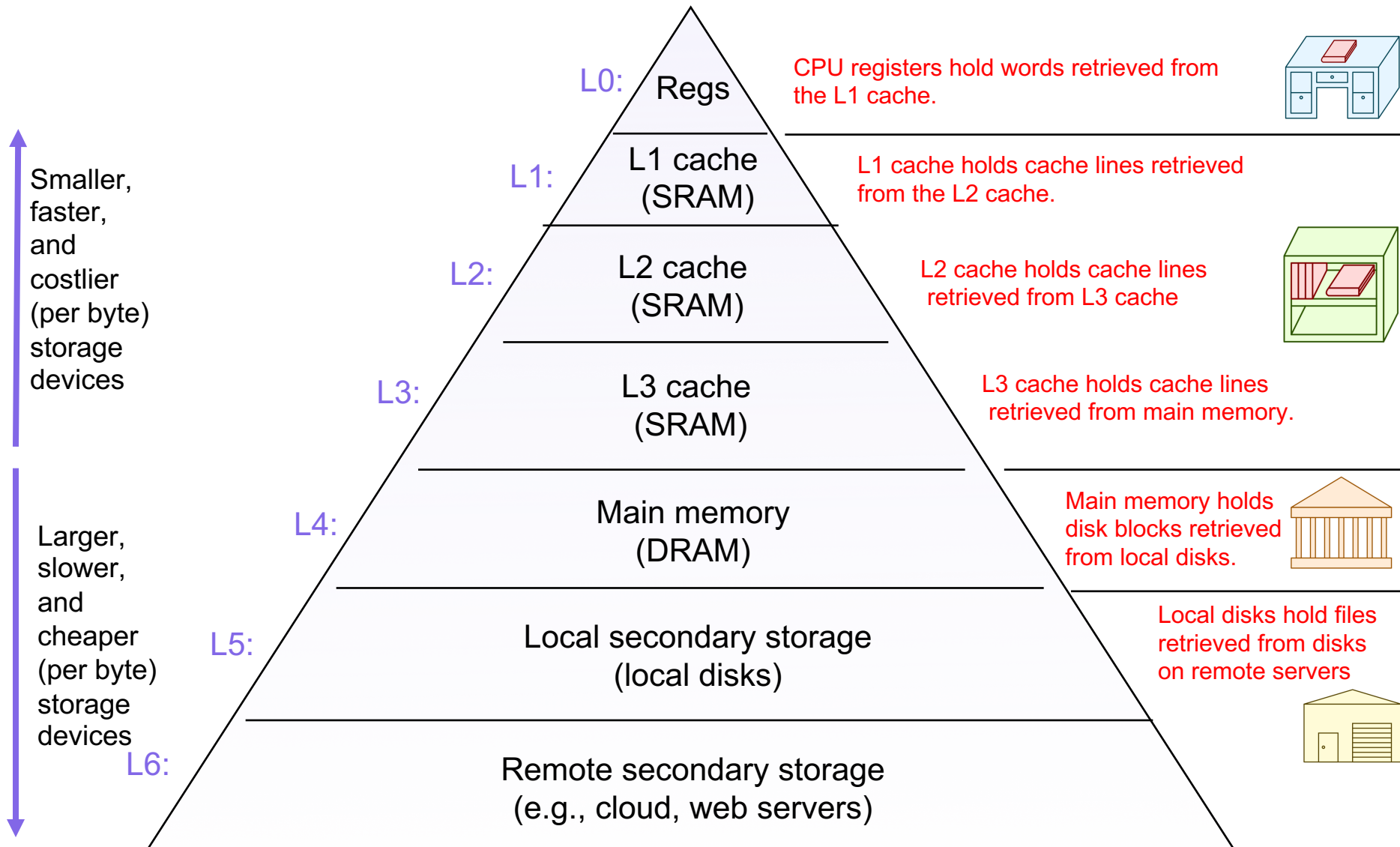


Lecture 11: Caches (cont'd)

CS 105

March 4, 2020

Memory Hierarchy

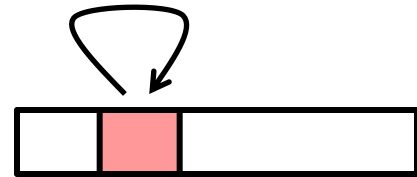


Principle of Locality

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

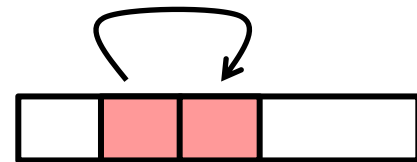
- ▶ **Temporal locality:**

- ▶ Recently referenced items are likely to be referenced again in the near future

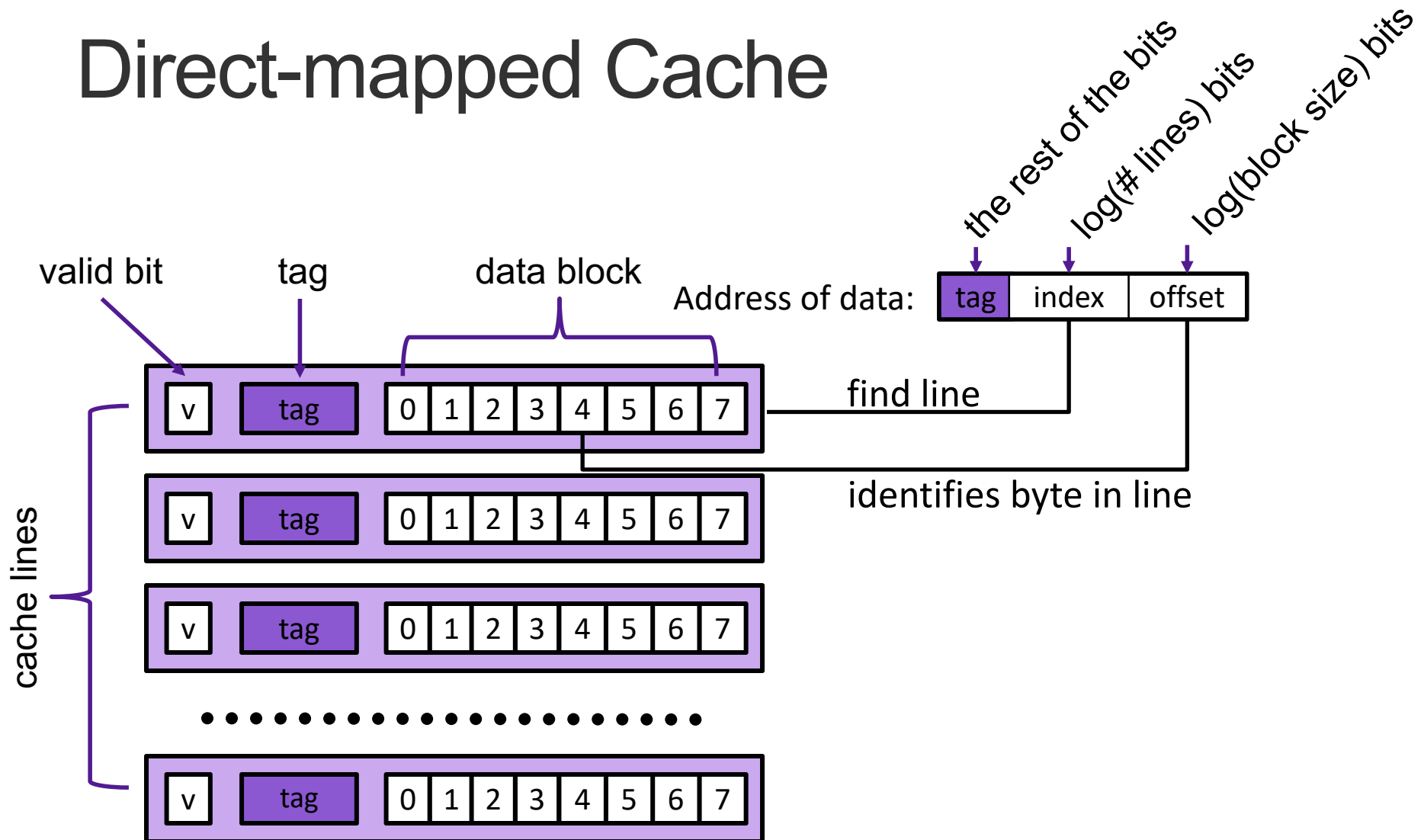


- ▶ **Spatial locality:**

- ▶ Items with nearby addresses tend to be referenced close together in time



Direct-mapped Cache

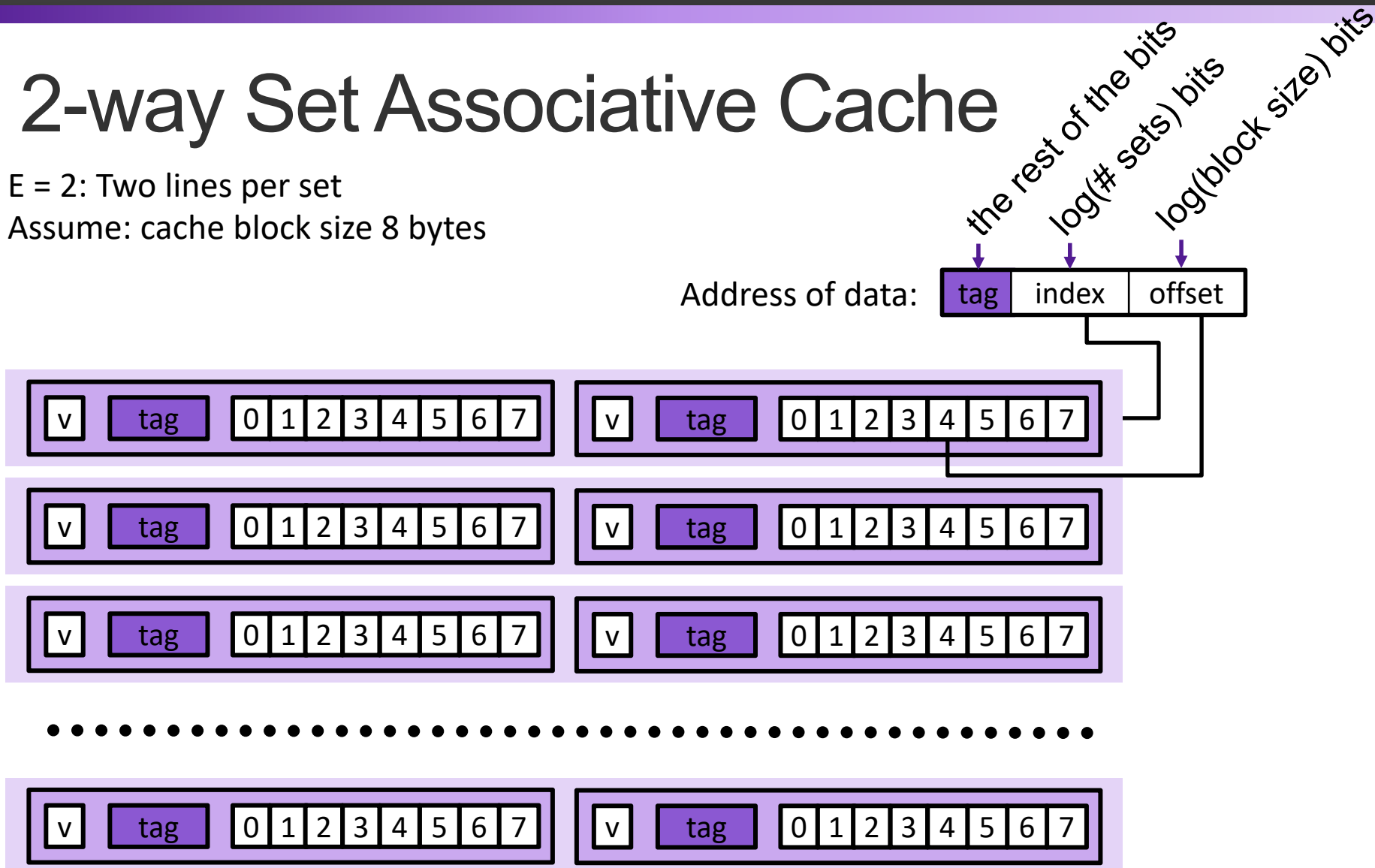


How well does this take advantage of spacial locality?
 How well does this take advantage of temporal locality?

2-way Set Associative Cache

E = 2: Two lines per set

Assume: cache block size 8 bytes



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

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

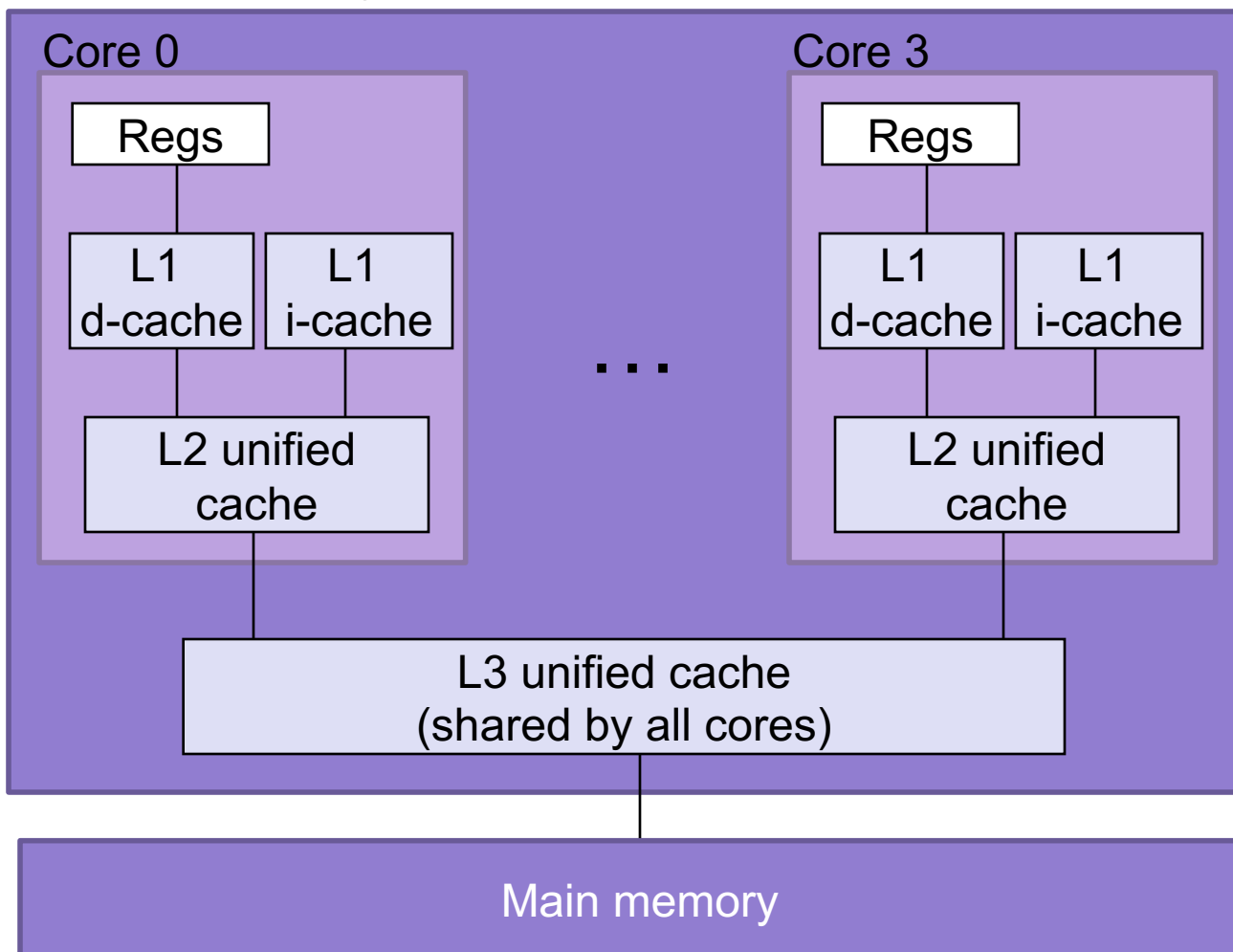
Categorizing Misses

- **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

Typical Intel Core i7 Hierarchy

Processor package



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**

