CASPER

CS230: Digital Logic Design and Computer Architecture

Lecture 19: Caches-II

https://www.cse.iitb.ac.in/~biswa/courses/CS230/autumn23/main.html

Knobs of interest

Line size, associativity, cache size

Tradeoff: latency, complexity, energy/power

Tips: Think about the extremes:
Line size = one byte or cache size
Associativity = one or #lines
Cache size = Goal oriented: latency/bandwidth or capacity

https://github.com/HewlettPackard/cacti/

Cache misses

Cold Miss: cache starts empty and this is the first reference

Conflict Miss: Many mapped to the same index bits

Capacity Miss: Cache size is not sufficient

Coherence Miss: in Multi-core systems, only [not I/O coherence]

On a Miss, Replace a block, which block?

Think of each block in a set having a “priority”
Indicating how important it is to keep the block in the cache

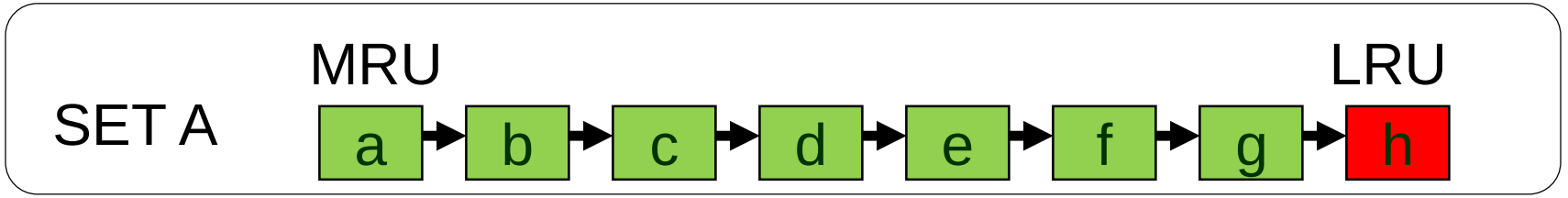
Key issue: How do you determine/adjust block priorities?

Ideally: Belady’s OPT policy, replace the block that will be used furthest in the future. No one knows the future though ☺

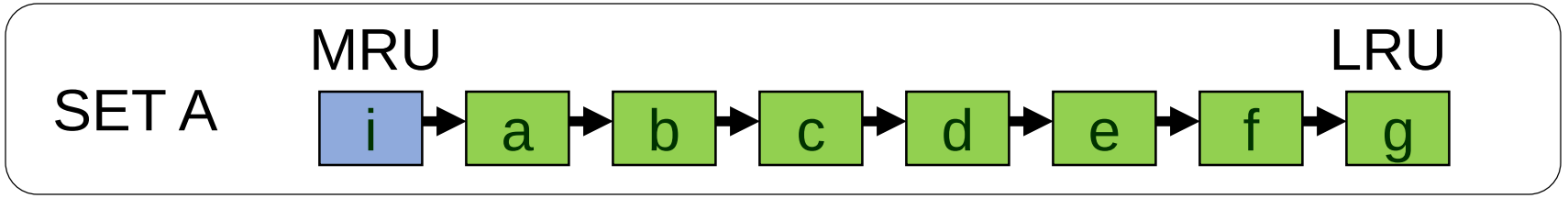
There are three key decisions in a set:
Insertion, promotion, eviction (replacement)

A simple LRU (Least-Recently-Used) Policy

Cache Eviction Policy: On a miss (block *i*), which block to evict (replace) ?



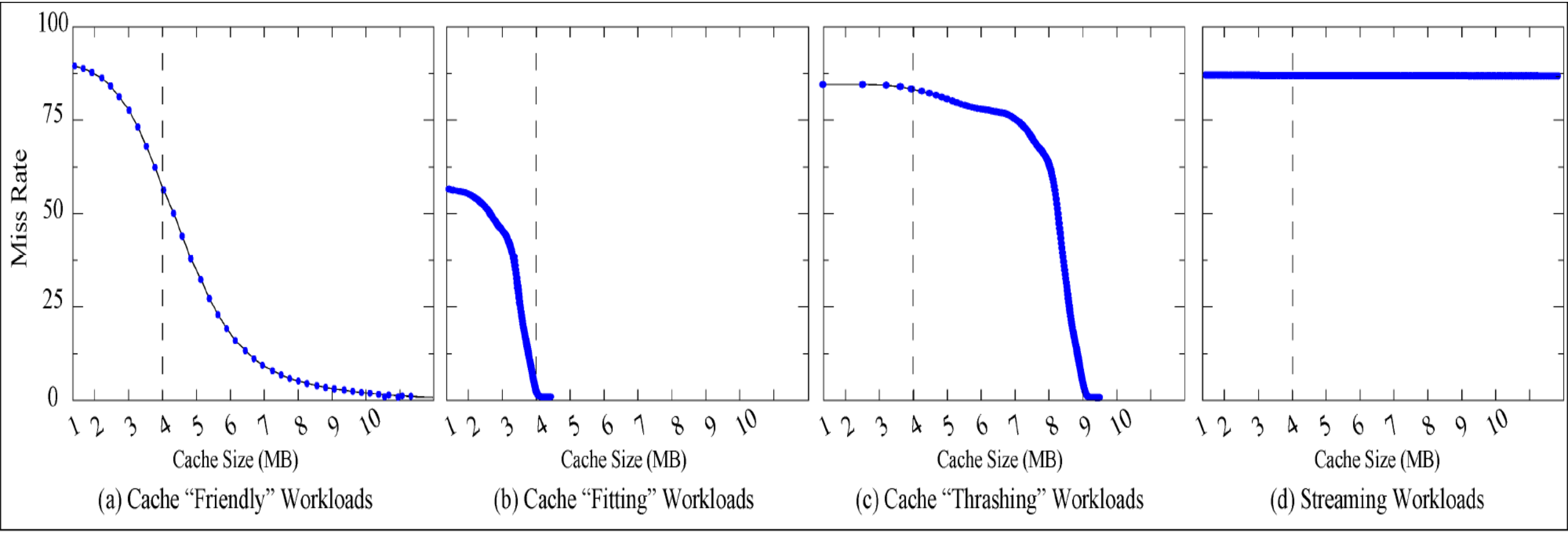
Cache Insertion Policy: New block *i* inserted into MRU.



Cache Promotion Policy: On a future hit (block *i*), promote to MRU

We need priority bits per block. For example, a 16-way cache will need four bit/block LRU causes thrashing when working set > cache size

Types of Applications



Let's redefine cache misses

Compulsory: first reference to a line (a.k.a. cold start misses)

- *misses that would occur even with infinite cache*

Capacity: cache is too small to hold all data

- *misses that would occur even under perfect (Belady's) replacement policy*

Conflict: misses that occur because of collisions due to line-placement strategy

- *misses that would not occur with ideal full associativity*

Cache knobs and Misses

- Larger cache size
  +reduces capacity and conflict misses?
  - hit time will increase
- Higher associativity
  +reduces conflict misses
  - increase hit time
- Larger line size
  +reduces compulsory misses
  - increases conflict misses and miss penalty

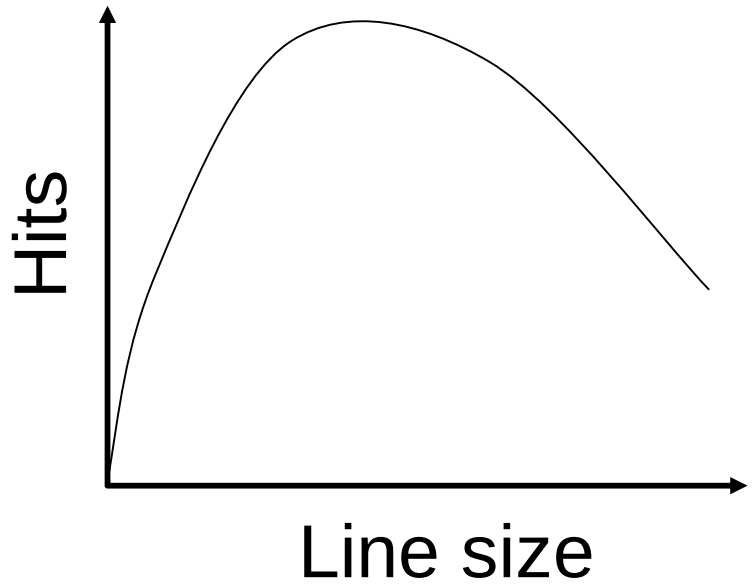
Line size

Too small blocks:

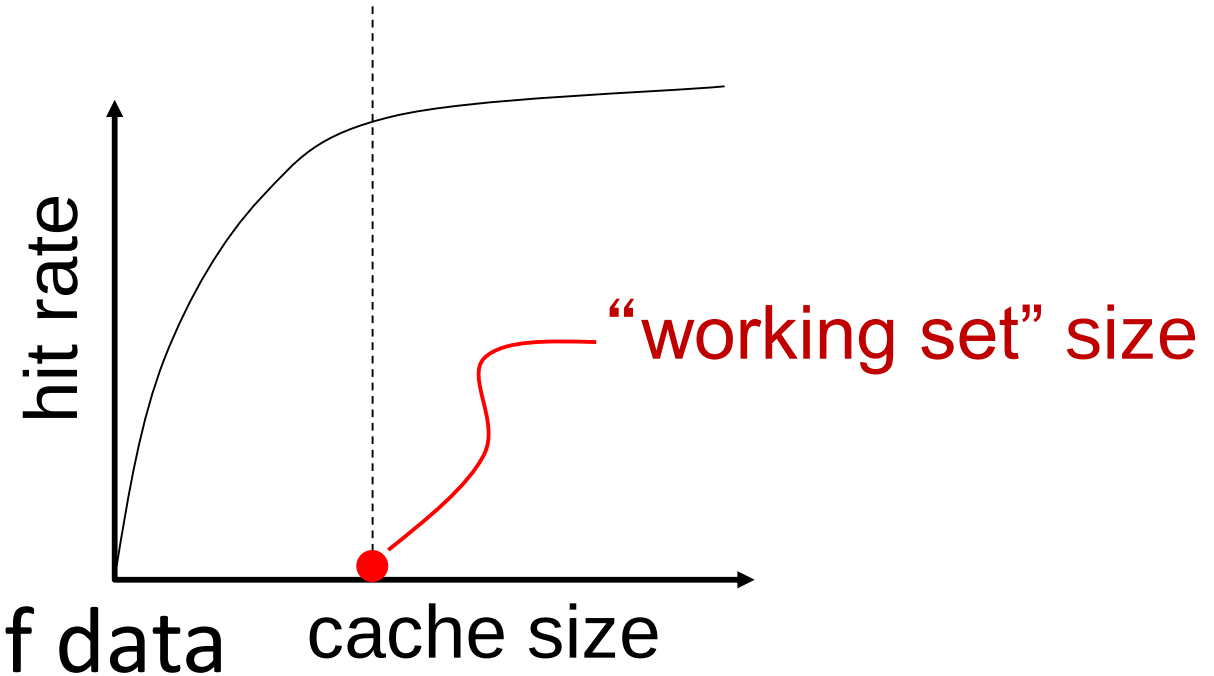
- don’t exploit spatial locality well
- have larger tag overhead

Too large blocks:

- too few total # of blocks
  - likely-useless data transferred
  - Extra bandwidth/energy consumed



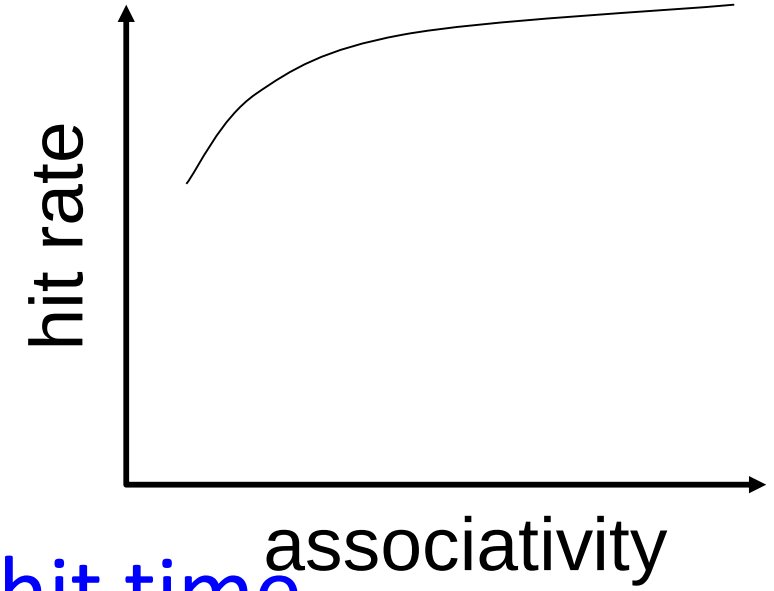
Cache size



Working set: the whole set of data the executing application references within a time interval

Associativity

Myth: It should be power of two. NO!!



L1 cache: lower associativity, hit time

L3 cache: higher associativity