CASPER

# CS773-2022-Autumn: Computer Architecture for Performance and Security

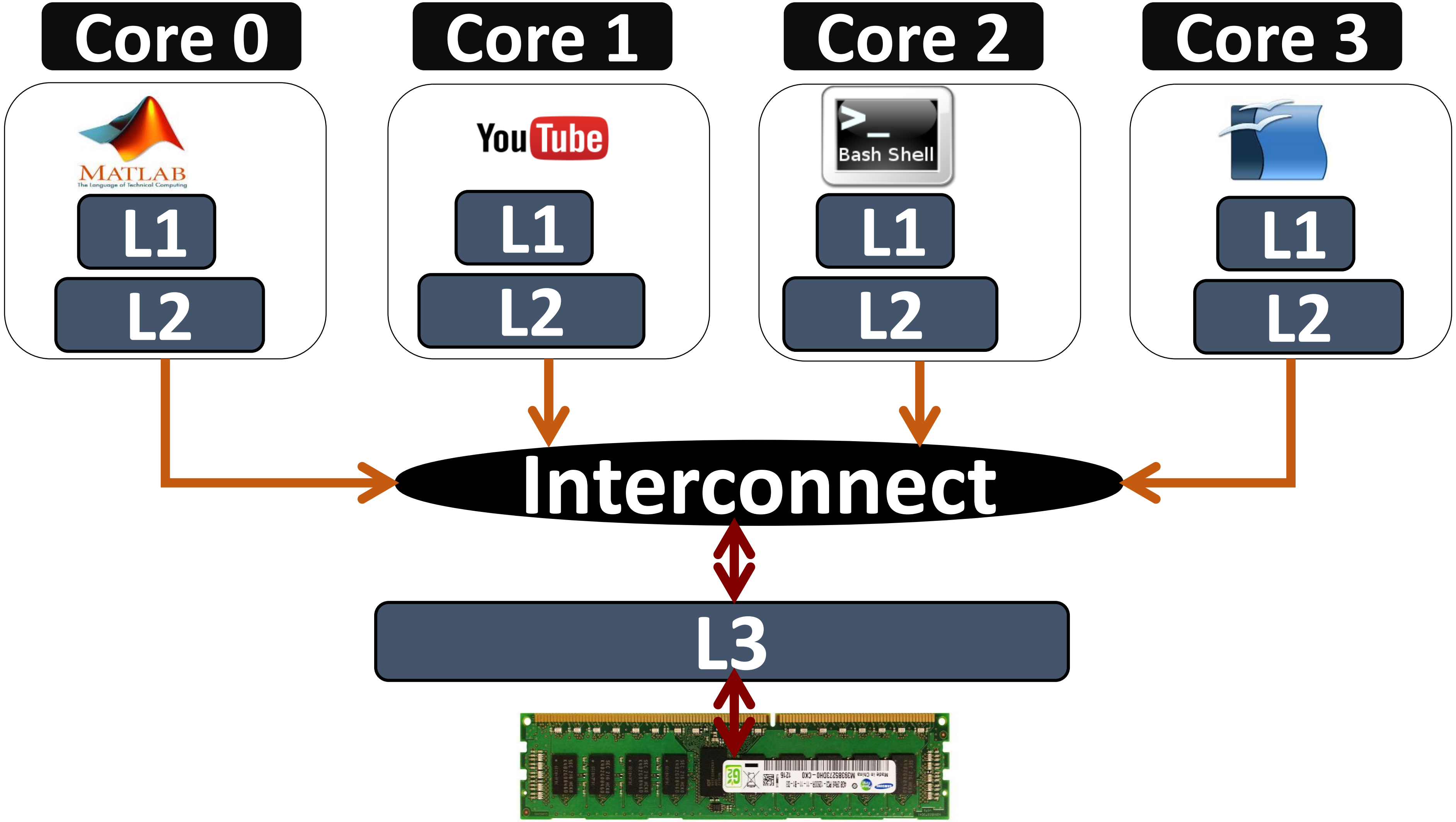
## Lecture 4: Catch the cache-II

https://www.cse.iitb.ac.in/~biswa/

# Phones on silence, please

# Thank You

# Multicore



Core 0

Core 1

Core 2

Core 3

L1

L2

L1

L2

L1

L2

L1

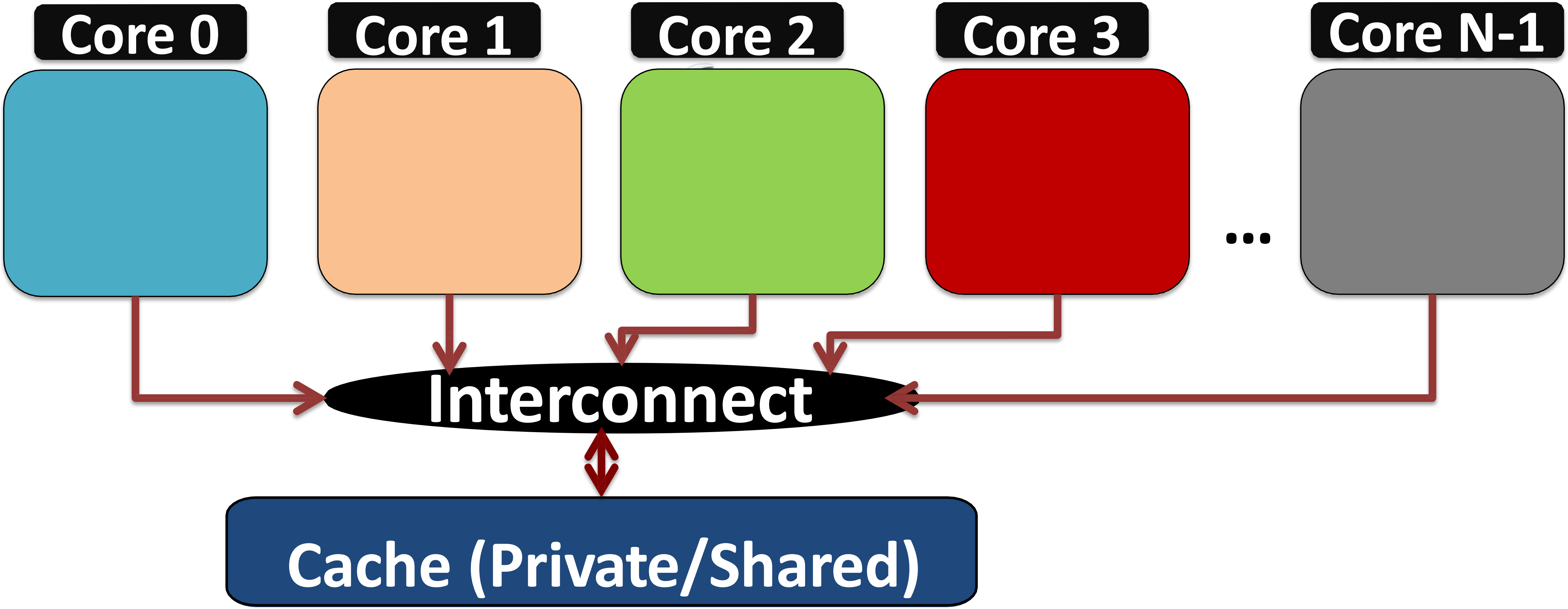
L2

Interconnect

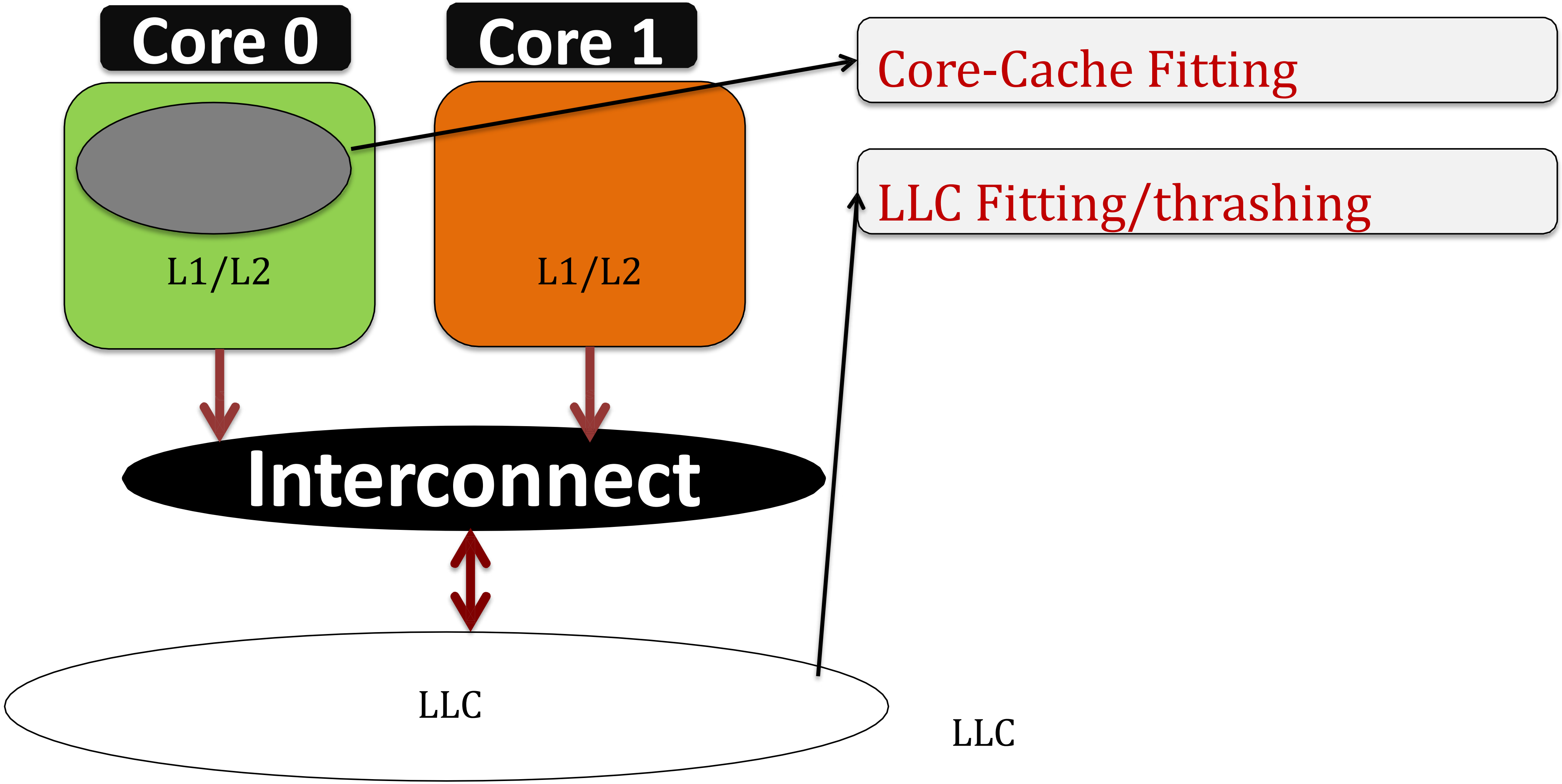
L3

CASPER

# Caches: Private/Shared



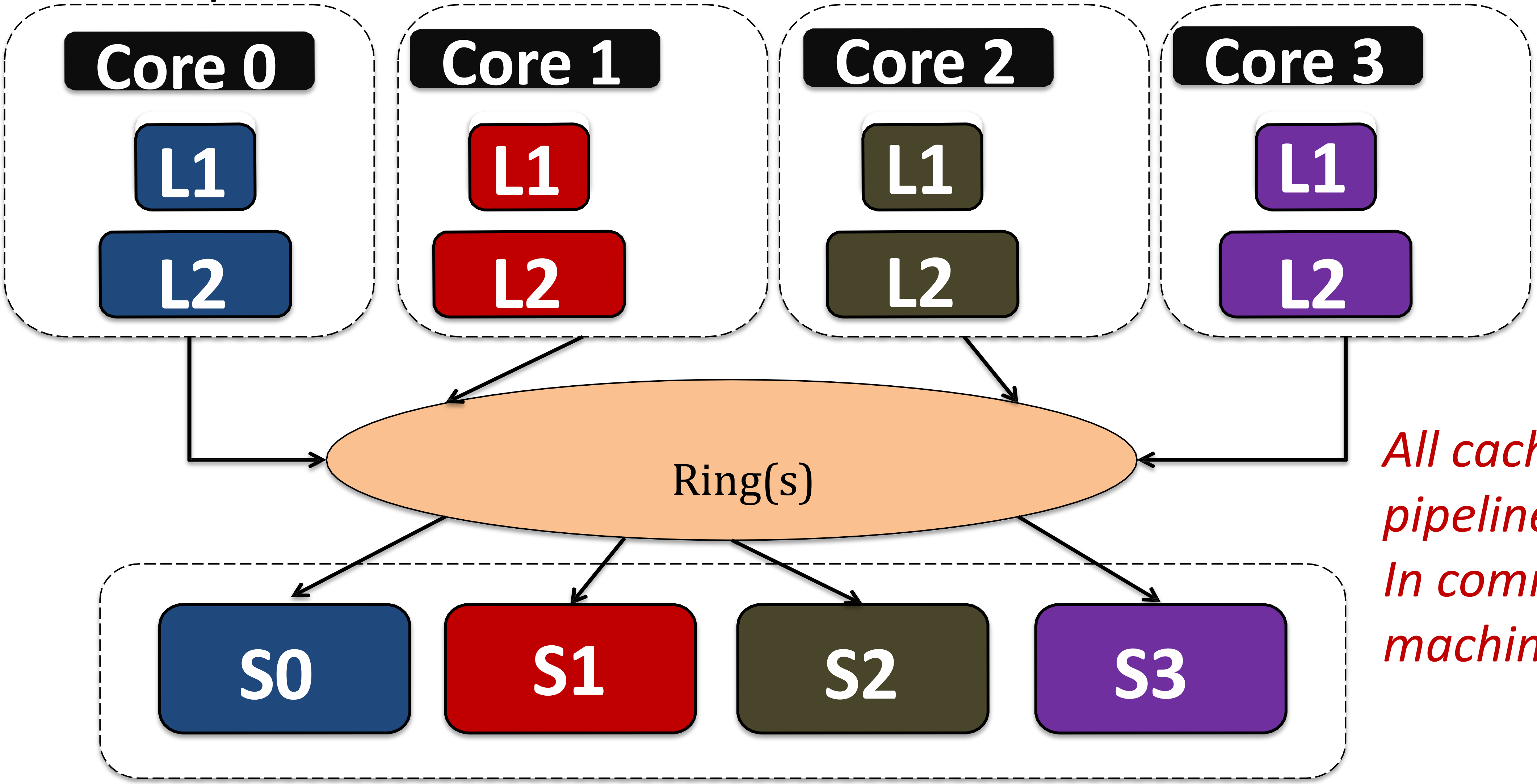
# Application behavior



Core-Cache Fitting

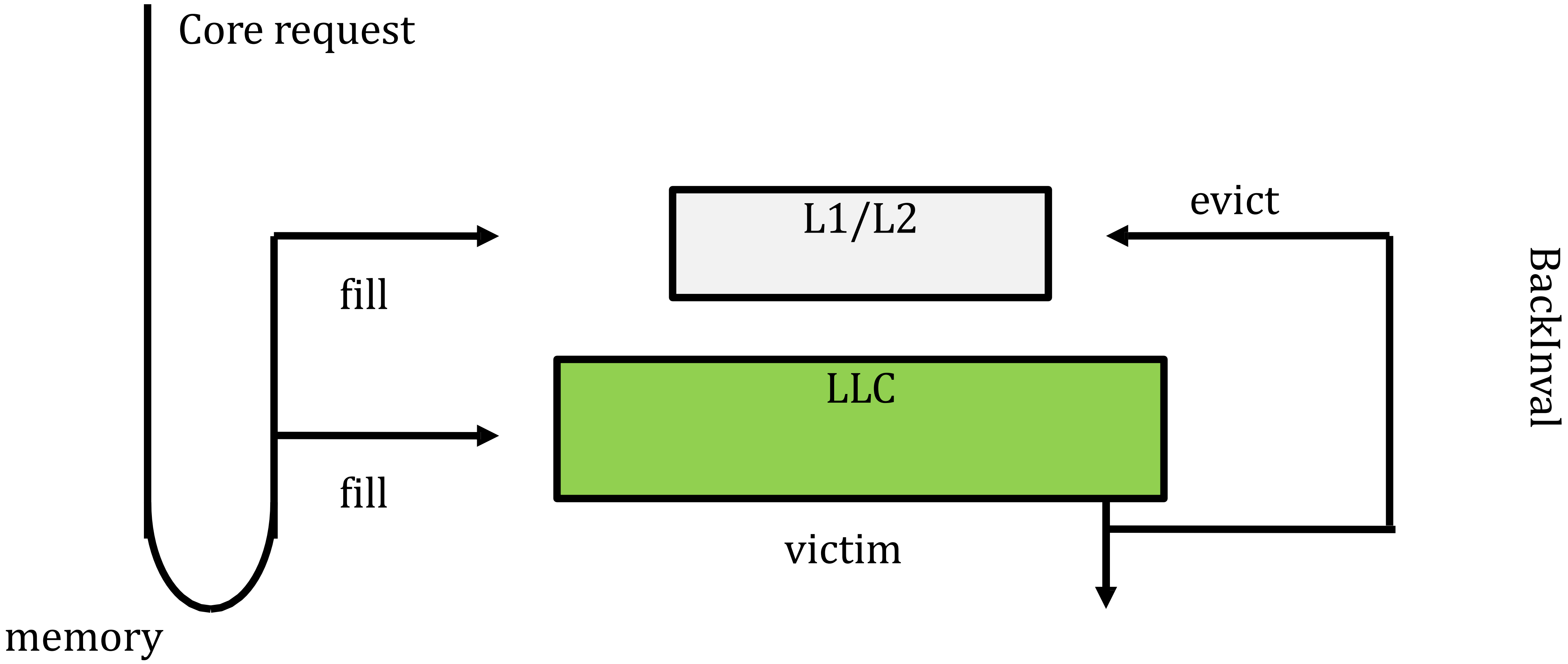
LLC Fitting/thrashing

# Sliced/Banked LLC

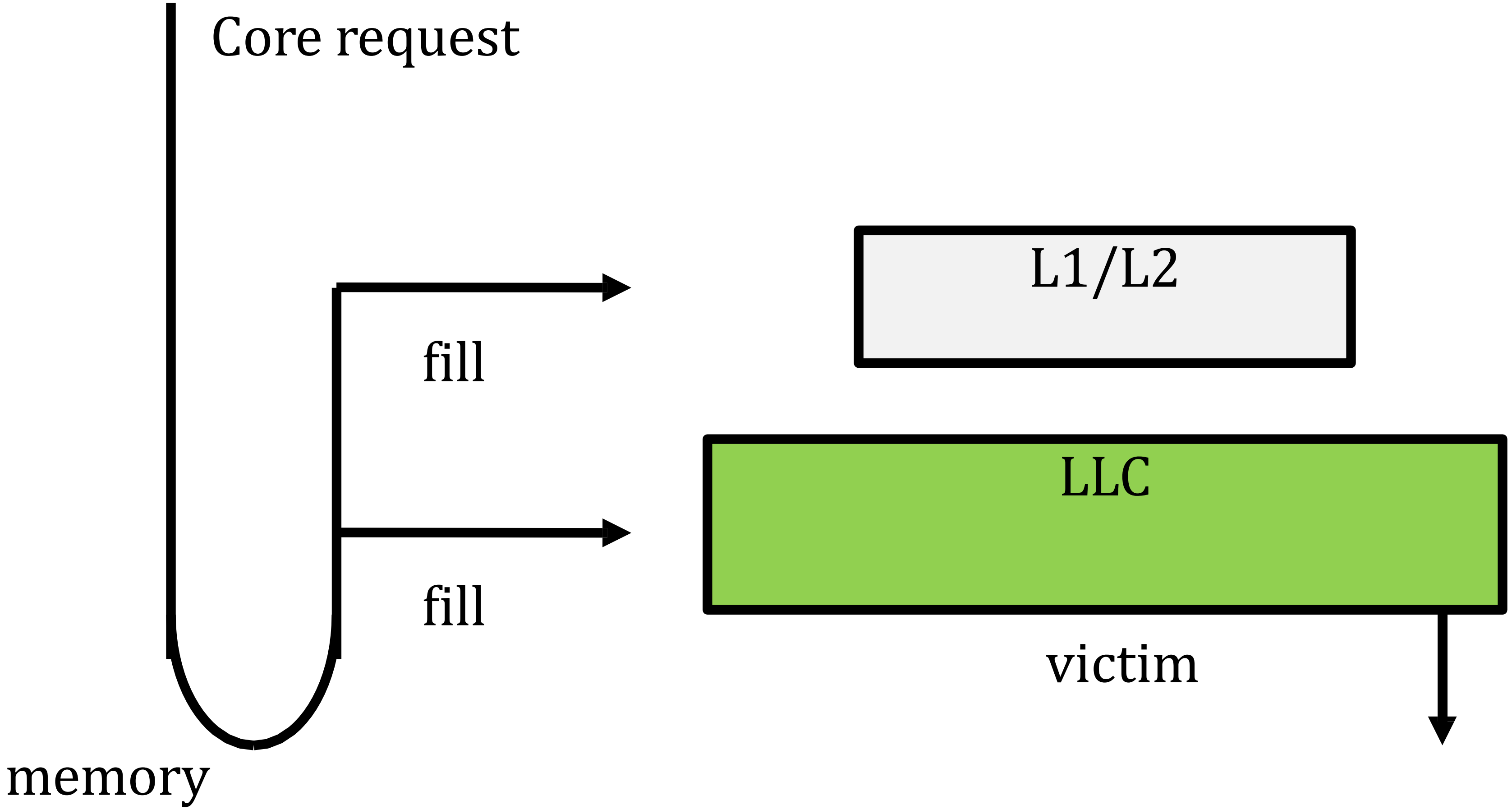


*All caches are pipelined In commercial machines*

# Inclusive Cache Hierarchy

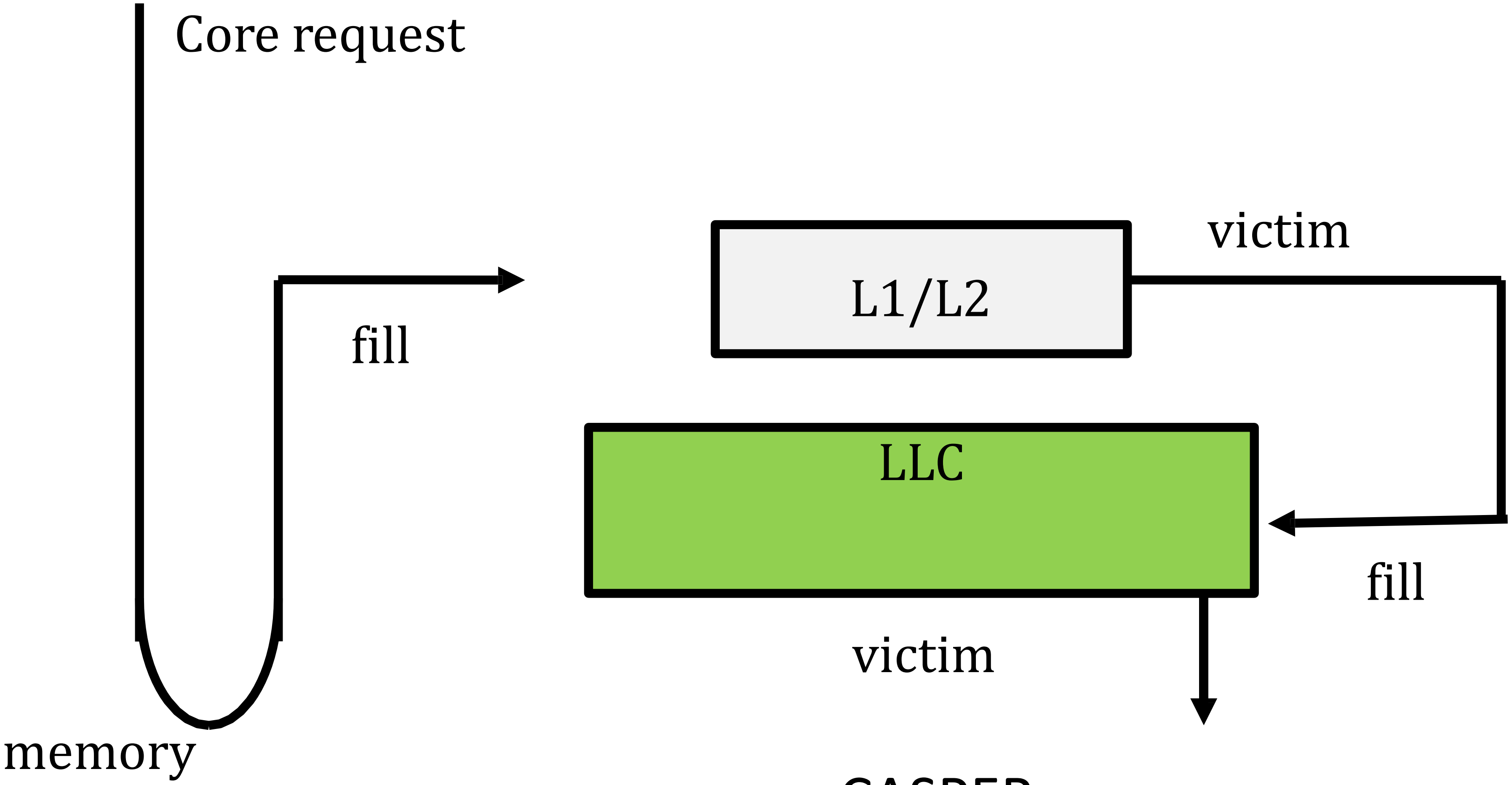


# Non-inclusive (many commercial machines)



CASPER

# Exclusive hierarchy



# 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]

# Cache Performance

How good is the cache for a given application?

Hit rate

Miss rate

Misses per kilo instructions (MPKI)

But Why?

# 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?

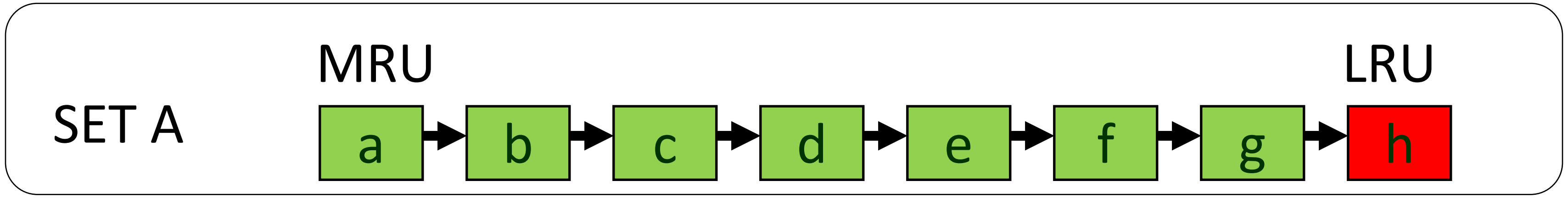
Ideally: Belady’s OPT policy, replace the block that will be used furthest in the future. No one knows the future though ☺

There are three key decisions in a set:

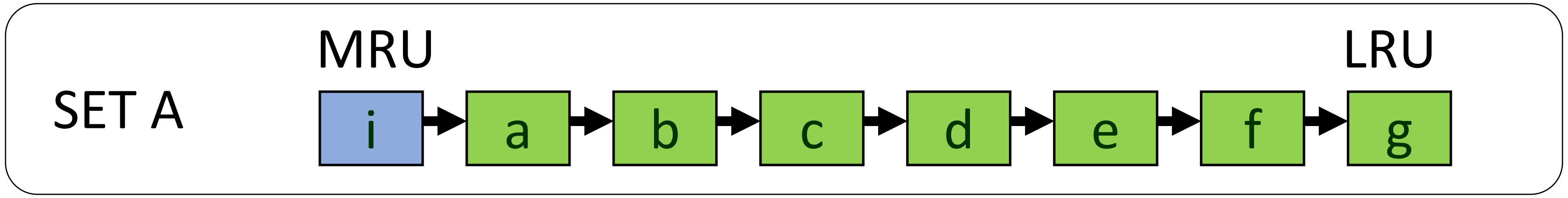
Insertion, promotion, eviction (replacement)

# A simple LRU (Least-Recently-Used) Policy

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



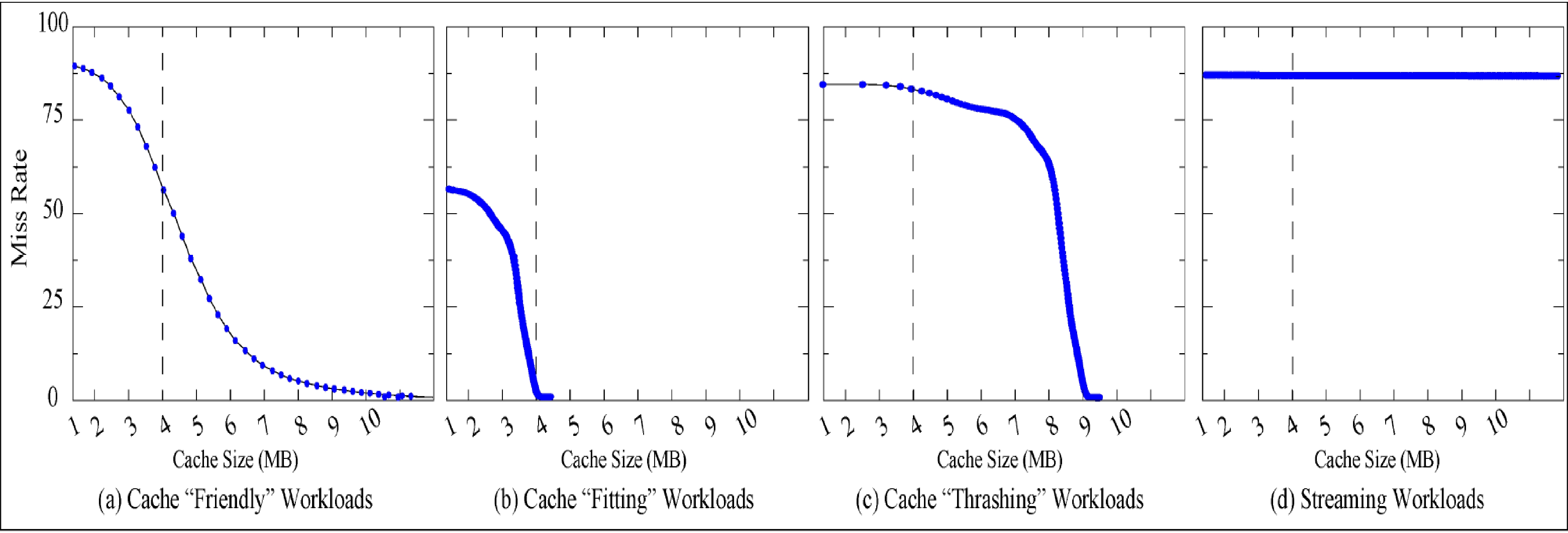
Cache Insertion Policy: New block *i* inserted into MRU.



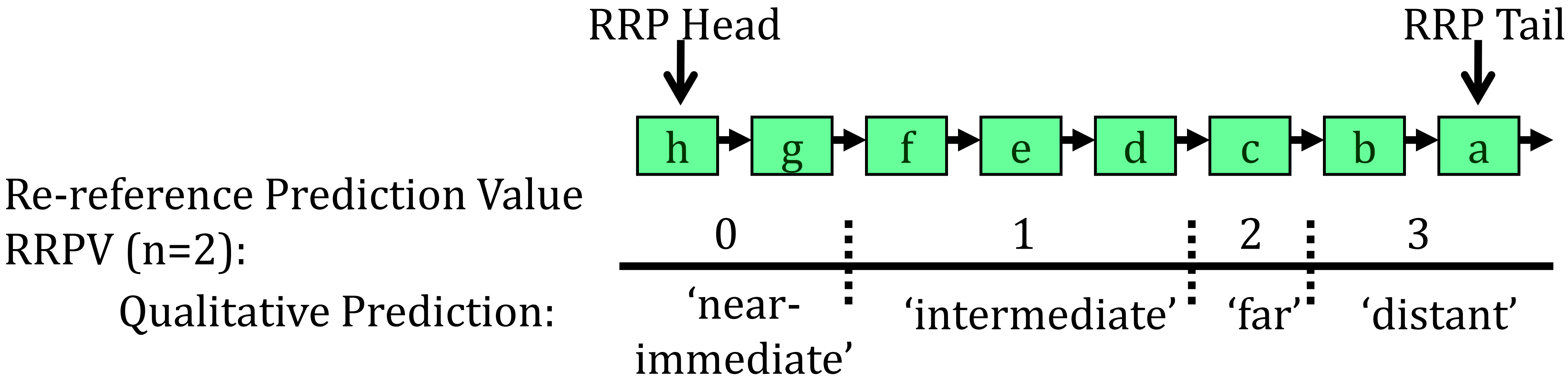
Cache Promotion Policy: On a future hit (block *i*), promote to MRU

*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



# LRU is not effective for Shared Caches



Intuition: New cache block will not be re-referenced soon. Replaces block with distant RRPV. Only two bits per block.

Insert with RRPV=2, Evict with RRPV=3, increment RRPVs till we get a block with RRPV=3, promote blocks with RRPV=0.

# Average Access Time

On average, how much time it takes for a LOAD to complete

*Average memory access time (AMAT) =*

*Hit time + Miss rate x Miss penalty*

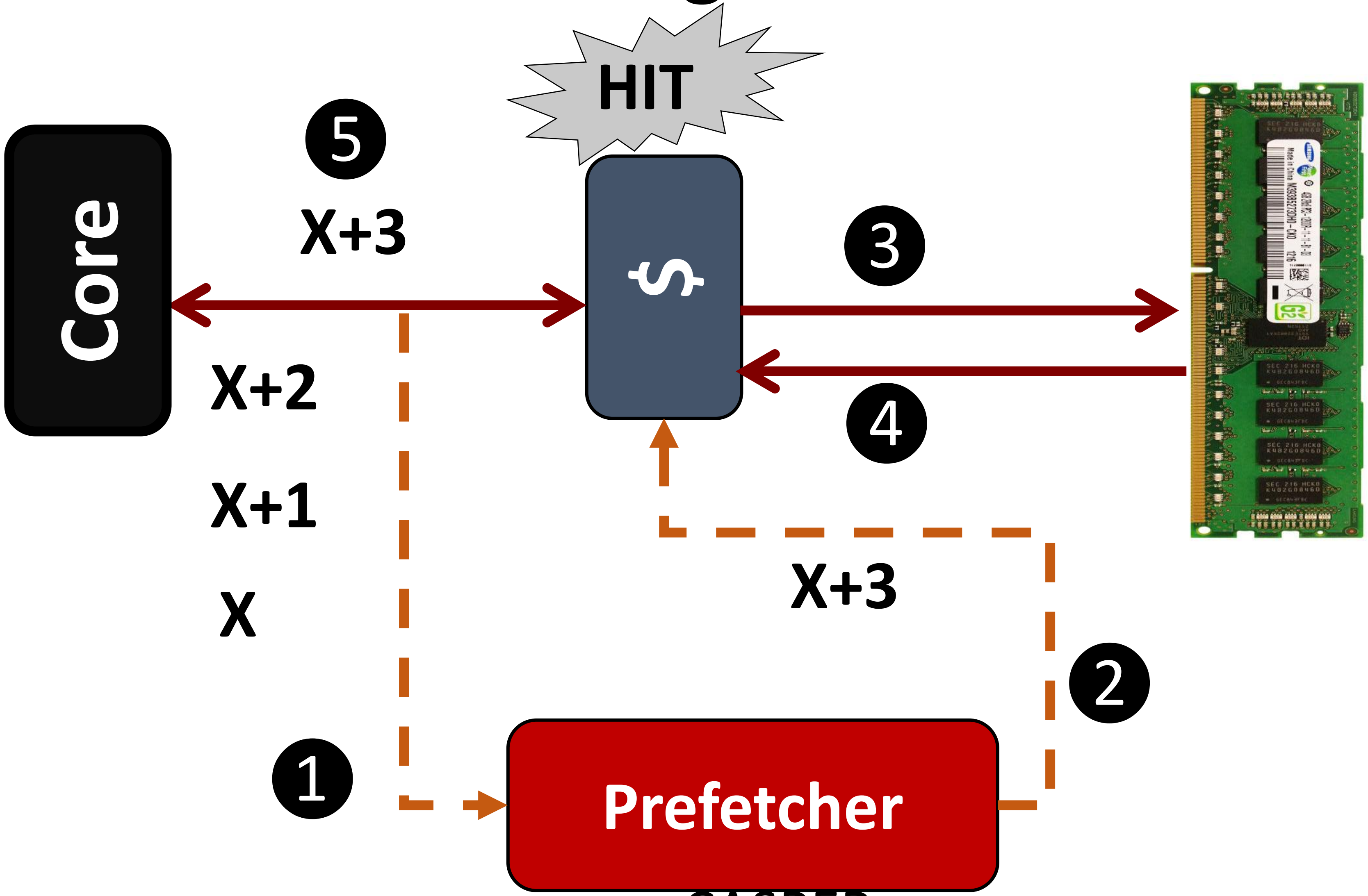
*Hit time-L1 + Miss rate-L1 x Miss penalty-L1*

Miss penalty-L1 = Hit time-L2 + Miss rate-L2 x Miss penalty-L2

*Hit time: Low, Miss rate: Low , Miss penalty: Low*

*Ideally, miss rate = 0.00% and hit time should be one cycle, so all LOADs will take just one cycle ☺*

# Hardware Prefetching



# 10K Feet View

*What?*

Latency-hiding technique - Fetches data before the core demands.

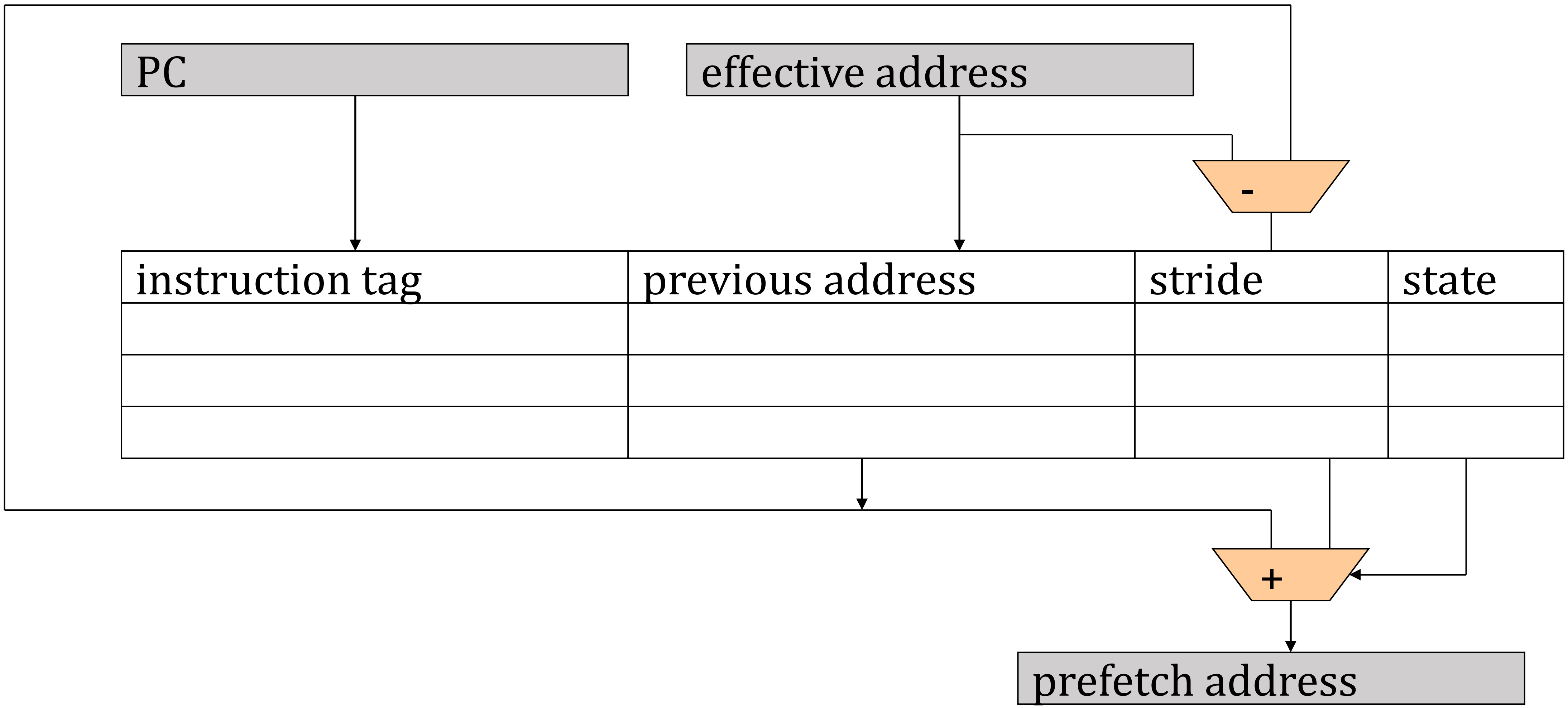
*Why?*

Off-chip DRAM latency has grown up to 400 to 800 cycles.

*How?*

By observing/predicting the demand access (LOAD/STORE) patterns.

# IP-stride prefetcher



# Time for Assignment-1 (Behind every urgency there is lack of planning)