Lecture 22: Cache Hierarchies, Memory

- Today’s topics:
  - Cache hierarchies
  - DRAM main memory
Locality

Why do caches work?

- Temporal locality: if you used some data recently, you will likely use it again
- Spatial locality: if you used some data recently, you will likely access its neighbors

No hierarchy: average access time for data = 300 cycles

32KB 1-cycle L1 cache that has a hit rate of 95%:

average access time = 0.95 \times 1 + 0.05 \times (301) = 16 cycles
Accessing the Cache

Direct-mapped cache: each address maps to a unique cache location.

8-byte words

Sets

Data array

Offset

8 words: 3 index bits

Byte address

101000
The Tag Array

Direct-mapped cache: each address maps to a unique address
Example Access Pattern

Direct-mapped cache: each address maps to a unique address

Tag array

Data array

8-byte words

Comparing

101000

Byte address

Tag

Assume that addresses are 8 bits long

How many of the following address requests are hits/misses?

4, 7, 10, 13, 16, 68, 73, 78, 83, 88, 4, 7, 10…
Increasing Line Size

A large cache line size → smaller tag array, fewer misses because of spatial locality

32-byte cache line size or block size
Associativity

Set associativity → fewer conflicts; wasted power because multiple data and tags are read
Associativity

How many offset/index/tag bits if the cache has 64 sets, each set has 64 bytes, 4 ways

Tag array

Compare

Tag

Byte address 10100000

Way-1 Way-2

Data array
Example 1

- 32 KB 4-way set-associative data cache array with 32 byte line sizes

- How many sets?

- How many index bits, offset bits, tag bits?

- How large is the tag array?
Example 1

• 32 KB 4-way set-associative data cache array with 32 byte line sizes

  \[ \text{cache size} = \#\text{sets} \times \#\text{ways} \times \text{block size} \]

• How many sets? 256

• How many index bits, offset bits, tag bits?
  
  8  5  19

• How large is the tag array?
  
  \[ \text{tag array size} = \#\text{sets} \times \#\text{ways} \times \text{tag size} \]
  
  = 19 Kb = 2.375 KB
Example 2

- A pipeline has CPI 1 if all loads/stores are L1 cache hits
- 40% of all instructions are loads/stores
- 85% of all loads/stores hit in 1-cycle L1
- 50% of all (10-cycle) L2 accesses are misses
- Memory access takes 100 cycles
- What is the CPI?
Example 2

• A pipeline has CPI 1 if all loads/stores are L1 cache hits
  40% of all instructions are loads/stores
  85% of all loads/stores hit in 1-cycle L1
  50% of all (10-cycle) L2 accesses are misses
Memory access takes 100 cycles
What is the CPI?

Start with 1000 instructions
1000 cycles   (includes all 400 L1 accesses)
+ 400 (I/s) x 15% x 10 cycles   (the L2 accesses)
+ 400 x 15% x 50% x 100 cycles   (the mem accesses)
= 4,600 cycles
CPI = 4.6
Cache Misses

• On a write miss, you may either choose to bring the block into the cache (write-allocate) or not (write-no-allocate)

• On a read miss, you always bring the block in (spatial and temporal locality) – but which block do you replace?
  ➢ no choice for a direct-mapped cache
  ➢ randomly pick one of the ways to replace
  ➢ replace the way that was least-recently used (LRU)
  ➢ FIFO replacement (round-robin)
Writes

• When you write into a block, do you also update the copy in L2?
  ➢ write-through: every write to L1 → write to L2
  ➢ write-back: mark the block as dirty, when the block gets replaced from L1, write it to L2

• Writeback coalesces multiple writes to an L1 block into one L2 write

• Writethrough simplifies coherency protocols in a multiprocessor system as the L2 always has a current copy of data
Types of Cache Misses

• Compulsory misses: happens the first time a memory word is accessed – the misses for an infinite cache

• Capacity misses: happens because the program touched many other words before re-touching the same word – the misses for a fully-associative cache

• Conflict misses: happens because two words map to the same location in the cache – the misses generated while moving from a fully-associative to a direct-mapped cache
Off-Chip DRAM Main Memory

• Main memory is stored in DRAM cells that have much higher storage density

• DRAM cells lose their state over time – must be refreshed periodically, hence the name *Dynamic*

• A number of DRAM chips are aggregated on a DIMM to provide high capacity – a DIMM is a module that plugs into a bus on the motherboard

• DRAM access suffers from long access time and high energy overhead
DIMM: a PCB with DRAM chips on the back and front

The memory system is itself organized into ranks and banks; each bank can process a transaction in parallel.

Each bank has a row buffer that retains the last row touched in a bank (it’s like a cache in the memory system that exploits spatial locality) (row buffer hits have a lower latency than a row buffer miss)
Title

• Bullet