CE/CZ 3001:
Advanced Computer Architecture

Module 5: Memory Systems

- cache performance

Asst Prof Liu Weichen
School of Computer Science and Engineering
Nanyang Technological University, Singapore

Summary of pre-recorded videos

- Memory hierarchy
- Cache design principles
- General organization of cache
- Cache placement policy
- Direct-mapped cache
- Fully associative cache
- Set associative cache
- Cache replacement policies
- Cache write policies
- Instruction cache vs data cache
- Cache performance and cache miss
- Cache architecture parameters and cache miss
- Cache performance analysis
- Multiple levels of caches
- Average memory access time

The Impact of Cache on Performance

- Assumptions
  - CPI_exec = 1 clock cycle (ignoring memory stalls)
  - Miss rate = 2%
  - Miss penalty = 200 clock cycles
  - Average memory references per instruction = 1.5 (instruction and data)

- $(CPI)_{no\_cache} = 1 + 1.5 \times 200 = 301$
- $(CPI)_{with\_cache} = 1 + (1.5 \times 0.02 \times 200) = 7$

Cache Performance: Average memory access time

- Could be in the unit of time or # of cycles.
- The time required for the cache miss depends on both the latency and bandwidth of the memory.
  - Latency determines the time to retrieve the first word of the block
  - bandwidth determines the time to retrieve the rest of this block

- The average memory access time includes two parts:

$$\text{Average memory access time} = \text{Hit time} + \text{Miss rate} \times \text{Miss penalty}$$

Cache Optimization Techniques

Average memory access time =
Hit time + Miss rate × Miss penalty

- Reducing the miss rate
  larger block size, larger cache size, and higher associativity
- Reducing the miss penalty
  multi-level caches and giving reads priority over writes
- Reducing the time to hit in the cache

Multi-level Cache to Reduce Miss Penalty

- Choice 1: Make the cache smaller/faster to keep pace with CPU speed
- Choice 2: Make the cache larger to overcome the widening gap between CPU and memory?
- Multi-Level Cache try to do both:
  - The 1st-level can be small enough to match the cycle time of the processor.
  - The 2nd-level can be large enough to capture many accesses that would go to main memory, thereby lessening the effective miss penalty.

$$\text{Average memory access time} = \text{Hit time}_{L1} + \text{Miss rate}_{L1} \times \text{Miss penalty}_{L1}$$

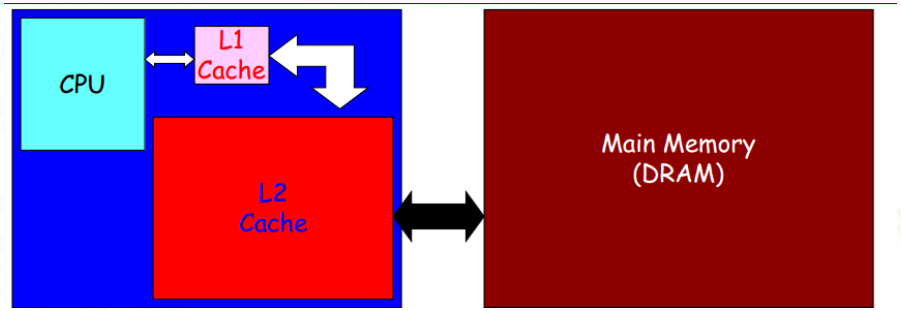
and

$$\text{Miss penalty}_{L1} = \text{Hit time}_{L2} + \text{Miss rate}_{L2} \times \text{Miss penalty}_{L2}$$

so

$$\text{Average memory access time} = \text{Hit time}_{L1} + \text{Miss rate}_{L1} \times (\text{Hit time}_{L2} + \text{Miss rate}_{L2} \times \text{Miss penalty}_{L2})$$

Reducing Miss Penalty with Multi-Level Caches



- Cache design is essentially balancing fast hits and few misses!
- L1 Cache design
  - smaller and with lower associativity, in order to minimize hit time
- L2 Cache design
  - minimize miss penalty
    - typically < 10 CPU cycles (while Main Memory latency > 100 CPU cycles)
  - typically 10 times larger than L1 Cache

Example: Assume that the CPI of a given computer is 1.0 when all memory accesses are cache hits. The only data accesses are loads and stores, and they arise in 40% of all instructions executed. If the miss rate is 5% for both instruction and data accesses, and the miss penalty is 25 clock cycles, how much faster would the machine be if all memory accesses were all hits?

Without cache misses, total number of cycles required:

IC * CPI = IC * 1.0 = IC

With cache misses, memory stall cycles are:

IC * memory accesses/instruction * miss rate * miss penalty

= IC * (1+0.4) * 5% * 25 = 1.75*IC

Hence, the total excution time with memory stalls is:

CPU clock cycles + Memory stall cycles

= IC * (1 + 1.75) = 2.75*IC

Without cache misses, the computer would be 2.75 times faster

(cont.) If the CPI is 2.0 when all memory accesses are cache hits, and the cache miss rate is 2% instead of 5%, with all the rest being equal, is this machine faster/slower compared to the previous 5% scenario?

With cache misses, memory stall cycles:

IC*memory accesses/instruction*miss rate*miss penalty

= IC * (1+0.4) * 0.02 * 25 = 0.70*IC

Hence, the total cycles required with memory stalls:

IC * (2 + 0.70) = 2.70*IC

Compared to the previous scenario, the computer is

2.75 / 2.70 = 1.02 times faster

(cont.) By comparing the execution time of the two scenarios, what conclusion do you draw about cache miss rates?

Given: cache miss penalty: 25 clock cycles

memory access instructions: 40%

We have:

	(a)	(b)
CPI	1.0	2.0
Cache miss rate	5%	2%
Performance (No. clock cycles)	2.75*IC	2.70*IC

Conclusion: For small percentage difference in cache misses, the computer with faster (and more expensive) CPU becomes slower, defeating all the efforts that a designer put in to the instruction execution pipeline. This speaks of the importance of maintaining a high hit rate in order to match the high execution rate of CPU.

Summary: Cache System

- Cache design
  - Organization schemes: direct mapped, set associative, fully associative
  - Write policy: write through, write back
  - Replacement policies: random, FIFO, LRU, etc.
  - Cache miss reasons and optimization
- Cache performance
  - Address and size related calculations
  - CPI, AMAT