Faculty of Media Engineering and Technology
Dept. of Computer Science and Engineering
Dr. Milad Ghantous

CSEN 702: Microprocessors
Winter 2024

Practice assignment 3-Solution

Exercise 1

Assume we have a computer where the cycles per instruction (CPI) is 1.0 when all memory accesses hit in the cache.
The only data accesses are loads and stores, and these total 50% of the instructions.
If the miss penalty is 25 clock cycles and the miss rate is 2%, how much faster would the computer be if all instructions were cache hits?

Solution:

- **Case 1: All accesses are hits**.
  - memory stall cycles = 0

$$\begin{aligned}\text{CPU execution time} &= (\text{CPU clock cycles} + \text{Memory stall cycles}) \times \text{Clock cycle}\\ &= (\text{IC} \times \text{CPI} + 0) \times \text{Clock cycle}\\ &= \text{IC} \times 1.0 \times \text{Clock cycle}\end{aligned}$$

- **Case 2: <u>Not</u> all accesses are hits**.
  - memory stall cycles > 0
  - we must compute it

$$\text{Memory stall cycles} = \text{IC} \times \frac{\text{Memory accesses}}{\text{Instruction}} \times \text{Miss rate} \times \text{Miss penalty}$$

$$= \text{IC} \times (1 + 0.5) \times 0.02 \times 25$$
$$= \text{IC} \times 0.75$$

Why (1+0.5)?

- Each instruction, from any type, needs an access to the instruction cache.
- Loads and stores need an extra access to the data cache.
- Since only 50% of the time we have loads and stores, this means the average memory accesses/instruction = 1 + 50%x1 = 1.5

- So:

$$\text{CPU execution time}_{\text{cache}} = (\text{IC} \times 1.0 + \text{IC} \times 0.75) \times \text{Clock cycle}$$
$$= 1.75 \times \text{IC} \times \text{Clock cycle}$$

The performance ratio is the inverse of the execution times:

$$\frac{\text{CPU execution time}_{\text{cache}}}{\text{CPU execution time}} = \frac{1.75 \times \text{IC} \times \text{Clock cycle}}{1.0 \times \text{IC} \times \text{Clock cycle}}$$
$$= 1.75$$

The computer with no cache misses is 1.75 times faster.

Assume that the cache miss penalty is 200 clock cycles, and all instructions normally take 1.0 clock cycles (ignoring memory stalls).
Assume that the average miss rate is 2%, there is an average of 1.5 memory references per instruction, and the average number of cache misses per 1000 instructions is 30.
What is the impact on performance when behavior of the cache is included? Also what's the impact when no cache is used at all.
Hint: Calculate the impact using both misses per instruction and miss rate.

Solution

CPU time = IC x ($CPI_{execution}$ + Memory stall cycles/instruction) x clock cycle time
= IC x ($CPI_{execution}$ + Misses/instruction x miss penalty) x clock cycle time

Using the misses per instruction number, we get:

CPU time = IC x [1 + (30/1000) x 200] x clock cycle time = IC x 7 x clock cycle time.

Using the miss rate number, the formula can be re-written as:

CPU time =
IC x ($CPI_{execution}$ + miss rate x (memory access/instruction) x Miss penalty) x clock cycle time

= IC x [1 + 0.02 x (1.5) x 200] x clock cycle time = IC x 7 x clock cycle time

Of course, we got the same answer in both, which is the CPU time increases 7 times due to stalls coming from cache misses, compared with a perfect cache that doesn't miss.

If a processor doesn't have a cache at all:

Meaning that every memory instruction needs to stall for 200 cycles, the CPU time would be = IC x (1 + 1.5 x 200) x clock cycle time = IC x 301 x clock cycle time.
Compare 301 to only 7 when using caches to conclude the importance of caches in performance.

Exercise 3

What is the impact of two different cache organizations on the performance of a processor? Assume

that the CPI with a perfect cache is 1.6, the clock cycle time is 0.35 ns, there are 1.4 memory references per instruction, the size of both caches is 128 KB, and both have a block size of 64 bytes. One cache is *direct mapped* and the other is *two-way set associative*.
Since the speed of the processor can be tied directly to the speed of a cache hit, assume the processor clock cycle time must be stretched 1.35 times to accommodate the added selection hardware of the set associative cache.
The cache *miss penalty* is 65 ns for either cache organization.
Assume the *hit time* is 1 clock cycle, the *miss rate* of a direct-mapped 128 KB cache is 2.1%, and the *miss rate* for a two-way set associative cache of the same size is 1.9%.

1) Calculate the average memory access time. Discuss the result.
2) Calculate the processor performance. Discuss the result.

Solution

1) The formula to find the average memory access time is:

Avg. mem. Access time = Hit time + Miss Rate x Miss penalty

Hits are considered to take 1 clock cycle.

Direct mapped: avg. access time = 0.35 + 0.021 x 65 = 1.72 ns
2-Way Set associative: avg. access time = 0.35 x1.35 + 0.019 x 65 = 1.71 ns
{Remember that the clock cycle for hitting the set associative is longer by 1.35 times due to added hardware)

conclusion: The set associative is slightly better than direct mapped in terms of access times.

2) The formula of CPU time is just like in exercise 1.

CPU time =
IC x ($CPI_{execution}$ + miss rate x (memory access/instruction) x Miss penalty) x clock cycle time

However, the miss penalty in the above formula is expressed in cycles, while in the given it's expressed in seconds which means it's the combined (**miss penalty x clock cycle time**) so we must split our equation to accommodate that. Hence we get:

CPU time = IC x $\left[(CPI_{execution}) \times \text{clock cycle time} + \text{miss rate} \times (\text{mem access/instr.}) \times 65 \right]$

For Direct mapped:
CPU time = IC x [(1.6) x 0.35 + 0.021 x (1.4) x 65] = 2.47 x IC

For 2-way set associative
CPU time = IC x [(1.6) x 0.35 x1.35 + 0.019 x (1.4) x 65] = 2.49 x IC longer than Direct mapped!

Conclusion: In contrast to the results of average memory access time comparison, the direct mapped cache leads to slightly better average performance because the clock cycle is stretched for all instructions for the two-way set associative case, even if there are fewer misses. Since CPU time is our bottom-line evaluation and since direct mapped is simpler to build, the preferred cache is direct mapped in this exercise.

Exercise 4

Compare the two below schemes by comparing A) the miss rates and B) average memory access time.

- **Scheme 1:** 16 KB *instruction cache* with a 16 KB *data cache*
- **Scheme 2:** 32 KB unified cache
    - Both caches are write-through, use the table below for misses per 1000 instructions
    - 36% of the instructions are data transfer instructions.
    - Assume a hit takes 1 clock cycle and the miss penalty is 200 clock cycles.
    - A load or store hit takes **1 extra clock** cycle on a unified cache if there is only one cache port to satisfy two simultaneous requests.

Table 1. Misses per 1000 instructions

Size (KB)	Instruction cache	Data cache	Unified cache
8	8.16	44.0	63.0
16	3.82	40.9	51.0
32	1.36	38.4	43.3

Solution:

Part A)

- **First, let's compute the miss rate for both schemes.**
- **Recall that:**

So:

$$\text{Miss rate} = \frac{\dfrac{\text{Misses}}{\text{1000 Instructions}} / 1000}{\dfrac{\text{Memory accesses}}{\text{Instruction}}}$$

$$\frac{\text{Misses}}{\text{Instruction}} = \text{Miss rate} \times$$

Case 1: Split caches (16KB instruction and a 16KB data)

$$\text{Miss rate}_{16\text{ KB instruction}} = \frac{3.82/1000}{1.00} = 0.004$$

All instructions access the instruction cache once, hence the division by 1. (mem access per instruction =1)

$$\text{Miss rate}_{16\text{ KB data}} = \frac{40.9/1000}{0.36} = 0.114$$

Only 36% of instructions access the data cache since they are data transfer instructions. Hence the 0.36. (mem access per instruction is 0.36)

- ***However, It's better to find a unified value for miss rate for split cache scheme. To compute the overall miss rate, we need to know the percentage of accesses to memory as if it was a one component and not split.***
- ***How much instructions and how much data?***

Let's say we have 100 instructions issues:

- ***100 instruction references memory access***
- ***36 additional access for loads and stores***

  Total 136 accesses.

100 out 136 of them instructions (=74%), 36 out 136 (=26%) of them data

- ***N.B. This also applies to unified cache.***

So:
Overall Miss Rate for split caches = (74% x 0.004 + 26% x 0.114) =0.0326

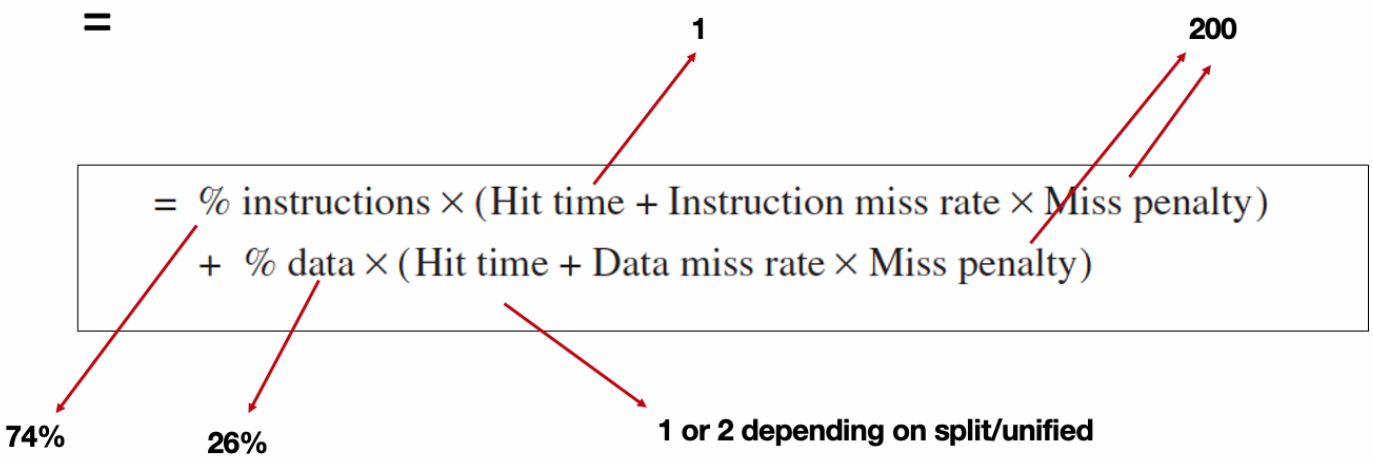
Case 2: Unified cache (32 KB)

$$\text{Miss rate}_{32\text{ KB unified}} = \frac{43.3/1000}{1.00 + 0.36} = 0.0318$$

Each instruction requires 1 access to get the instruction and then 36% of those require an extra access to the memory
1+0.36

Verdict: Miss rate of unified cache (0.0318) is < the split caches (0.0326) but let's dig further!

Part B)
Average memory access time = Hit time + Miss rate x Miss penalty



Case 1: split caches

$$\text{Average memory access time}_{\text{split}}$$
$$= 74\% \times (1 + 0.004 \times 200) + 26\% \times (1 + 0.114 \times 200)$$
$$= (74\% \times 1.80) + (26\% \times 23.80) = 1.332 + 6.188 = 7.52$$

Case 2: unified cache

$$\text{Average memory access time}_{\text{unified}}$$
$$= 74\% \times (1 + 0.0318 \times 200) + 26\% \times (1 + 1 + 0.0318 \times 200)$$
$$= (74\% \times 7.36) + (26\% \times 8.36) = 5.446 + 2.174 = 7.62$$

Verdict: Access time of unified is longer, which makes the split better! Different than what we got only relying on miss rates.