Assignment ACA

Nhóm 2

Members

Bùi Đức Hiếu

Đoàn Dũ

Lê Nguyễn Khánh Duy

Lê Nguyên Dũng

Outline

Different between:

- Data Cache and Instruction Cache
- Unified Cache and split Cache

Multi-level Cache

What is cache performance?

This presentation focus on answer teacher's questions. Answers are showed through inside contents.

Data cache and Instruction Cache

Data cache and Instruction Cache

First and probably foremost, data stored in instruction cache (i cache) is generally somewhat different than in data cache (d cache).

Data cache and Instruction Cache

Second, it simplifies circuitry a bit.

- Data cache deal with reads and writes.
- Instruction cache only deals with reads.

Data cache and Instruction Cache

Third, it increases bandwidth.

Modern processors can read data from i & d cache simultaneously so they can actually do two reads and one write in any given cycle.

Data cache and Instruction Cache

Fourth, it can save power.

- Power down when the associated circuitry not used.
- Separate caches can power up circuits to increasing the chances of a circuit remain un-powered.

Multi-level cache

Multi-level cache

A CPU cache is used to reduce the average time to access memory by processor.

The problem is the balance between latency and cache hit time.

Multi-level cache

Multi-level cache

Multi-level caches generally operate by checking the fastest cache L1 first.

Hit, the processor proceeds at high speed.

Miss, the next fastest cache (L2) is checked.

And so on before memory is checked.

L1 cache

The fastest cache, very very high speed.

It usually comes within the processor.

8-64kb (4 - 32kb).

Created by SRAM.

Seperated i & d cache.

L2 cache

Secondary cache

High speed, but slower then L1.

64kb - 4MB (512kb/1 core).

Between L1 and memory.

Not seperate i & d cache.

L3 cache

Extra cache.

Between L2 and memory.

Slower than L2, but higher then memory.

Share by all of core.

4 - 8 MB.

Multi level Cache:

Average Memory Access Time (AMAT)

AMAT

With cache L1 and L2

$$\text{AMAT} = \text{Hit time}_{L1} + \text{Miss rate}_{L1} \text{x} \text{Miss penalty}_{L1}$$

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

$$\text{AMAT} = \text{Hit time}_{L1} + \text{Miss rate}_{L1} \text{x} \left(\text{Hit time}_{L2} + \text{Miss rate}_{L2} \text{x} \text{Miss penalty}_{L2}\right)$$

AMAT

Local miss rate = Miss rate (L1) hoặc Miss rate (L2)

Global miss rate = Number of miss/ total number of memory accesses = Miss rate (L1), but miss rate (L1) x Miss rate (L2)

Note

Như vậy, trong phần trên, nhóm 2 đã trả lời 3 câu hỏi:

Tại sao lại có multi-cache?

Sự khác nhau của các loại cache L1, L2, L3?

Công thức tính AMAT cho mỗi loại cache?

L4 Cache?

Multilevel cache is the compromise of multi processor and cost, performance.

Each core read/write with its own L1.

L2 is shared by cores.

L4 Cache?

L1, L2, L3 are created by SRAM.

SRAM is very expensive.

L2 stores frequently data.

L3 stores less frequently data.

What is cache performance?

What is cache performance?

Instruction count (IC) is independent of the hardware.

Miss rate is independent of the speed of the hardware.

Miss rate can be just as misleading as instruction count.

What is cache performance?

Better measure of memory hierarchy performance is average memory access time (AMAT).

AMAT = Hit time + Miss rate x Miss penalty

This formula can help us decide between split caches and a unified cache.

What is cache performance?

The performance, including cache misses

$$\text{CPU time} = \text{IC} \times \left(\text{CPI}_{\text{execution}} + \frac{\text{Memory stall clock cycles}}{\text{Instruction}}\right) \times \text{Clock cycle time}$$

Performance using miss rate

$$\text{CPU time} = \text{IC} \times \left(\text{CPI}_{\text{execution}} + \text{Miss rate} \times \frac{\text{Memory accesses}}{\text{Instruction}} \times \text{Miss penalty}\right) \times \text{Clock cycle time}$$

Example

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? Calculate the impact using both misses per instruction and miss rate.

Answer

The performance, including cache misses

CPU time = IC ×[1.0 + (30/1000 ×200)] Clock cycle time

= IC ×7.00 ×Clock cycle time

The performance using miss rate

CPU time = IC ×[1.0 + (1.5 ×2% ×200)] Clock cycle time

= IC ×7.00 ×Clock cycle time

Effects of Cache Performance on CPU performance

Effects of Cache Performance on CPU Performance

Low CPI machines suffer more relative to some fixed CPI memory penalty.

- A machine with a CPI of 5 suffers little from a 1 CPI penalty.
- However, a processor with a CPI of 0.5 has its execution time tripled.

Effects of Cache Performance on CPU Performance

Cache miss penalties are measured in cycles, not nanoseconds.

- A faster machine will stall more cycles on the same memory system.

Improving Cache Performance

Improving Cache Performance

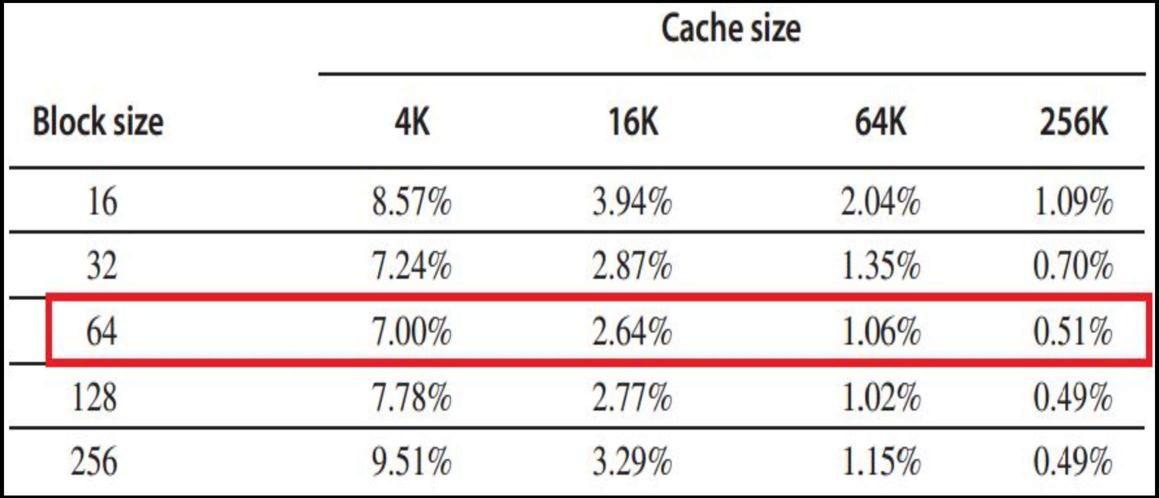
Reducing the miss rate: larger block size, larger cache size, and higher associativity.

Reducing the miss penalty: multilevel caches and giving reads priority over writes.

Reducing the time to hit in the cache.

Improving Cache Performance

Reducing Miss Rate: *larger block size, larger cache size*.



	Cache size			
Block size	4K	16K	64K	256K
16	8.57%	3.94%	2.04%	1.09%
32	7.24%	2.87%	1.35%	0.70%
64	7.00%	2.64%	1.06%	0.51%
128	7.78%	2.77%	1.02%	0.49%
256	9.51%	3.29%	1.15%	0.49%

Increasing block size up to 64-byte since when increasing block size up to 128-byte, the miss rate is increasing again.

Improving Cache Performance

Reducing Miss Rate: *higher associativity*.

- Reduce conflict misses.
- Rules of thumb

  8-way = fully associative

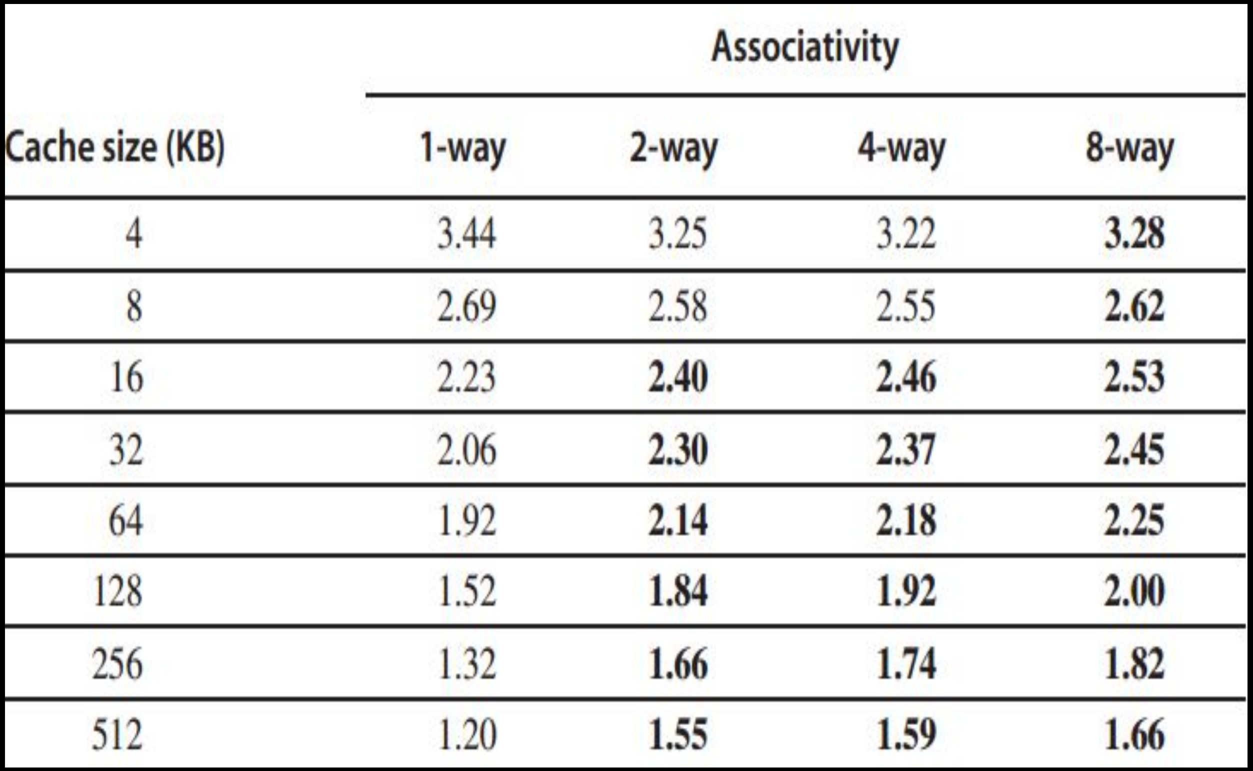
  Direct mapped size N=2-way set associative N /2

Improving Cache Performance

But

- Size N associative is larger than Size N direct mapped.
- Associative typically slower that direct mapped.

Improving Cache Performance



	Associativity			
Cache size (KB)	1-way	2-way	4-way	8-way
4	3.44	3.25	3.22	**3.28**
8	2.69	2.58	2.55	**2.62**
16	2.23	**2.40**	**2.46**	**2.53**
32	2.06	**2.30**	**2.37**	**2.45**
64	1.92	**2.14**	**2.18**	**2.25**
128	1.52	**1.84**	**1.92**	**2.00**
256	1.32	**1.66**	**1.74**	**1.82**
512	1.20	**1.55**	**1.59**	**1.66**

Improving Cache Performance

Reducing Miss Penalty: Multilevel cache

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

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

$$\Downarrow$$

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

Improving Cache Performance

Reducing Miss Penalty: Giving Priority to Read Misses over Writes.

This optimization serves reads before writes have been completed.

Improving Cache Performance

With the complexities of a write buffer:

- With a write-through cache, write buffer of the proper size is the most important improvement.
- Write buffers do complicate memory accesses since they might hold the updated value of a location needed on a read miss.

Improving Cache Performance

Reducing the time to hit in the cache

Make Caches smaller and simpler

  Hit Time = 1 cycle is good.

Smaller blocks.

Improving Cache Performance

For writes:

No write allocate: no “hit” on cache, just write to write buffer.

Write allocate: to avoid two cycles (first check for hit, then write) pipeline writes via a delayed write buffer to cache.

Improving Cache Performance

Summary

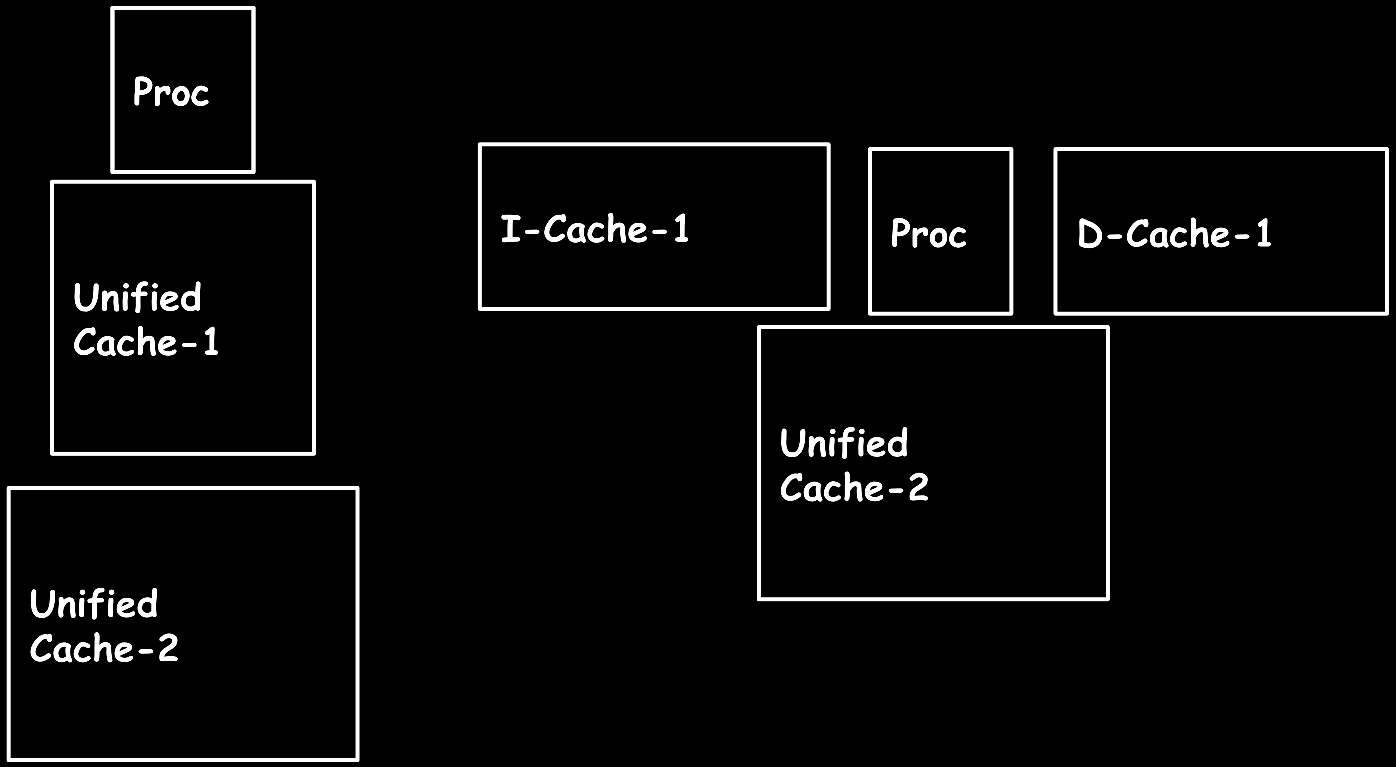
Hit time depends on cache size.

Cache is larger => hit time is increased => miss rate is decreased.

We cannot reduce all of components: miss rate, miss penalty, hit time at the same time.

Unified and split cache

Unified and split cache



Unified and split cache

Unified Cache

- Low Miss ratio because more space available for either instruction or data.
- Low cache bandwidth because instruction and data cannot be read at the same time due to one port.

Unified and split cache

Split Cache

- High miss ratio because either instructions or data may run out of space even though space is available at other cache.
- High bandwidth because an instruction and data can be accessed at the same time.

Example

16KB I&D:

Inst miss rate=0.64%,

Data miss rate=6.47%

32KB unified:

Aggregate miss rate=1.99%

Which is better (ignore L2 cache)?

Answer

Assume 33% data ops $\Rightarrow$ 75% accesses from instructions (1.0/1.33)

Hit time=1, miss time=50

Note that *data* hit has 1 stall for unified cache (only one port)

$\text{AMAT}_{\text{Harvard}}$=75%x(1+0.64%x50)+25%x(1+6.47%x50) = 2.05

$\text{AMAT}_{\text{Unified}}$=75%x(1+1.99%x50)+25%x(1+1+1.99%x50)= 2.24

Q & A

Thank you for attention.