Assignment 6 Solutions
Caches

Alice Liang

June 4, 2013

1 Introduction to caches

For a direct-mapped cache design with a 32-bit address and byte-addressable memory, the following bits of the address are used to access the cache:

1.1

Tag	Index	Offset
31-10	9-5	4-0

1.1.1

What is the cache line size (in words)?

Cache line size = $2^{\text{offset bits}} = 2^5$ bytes = 2^3 words = 8 words

1.1.2

How many entries (cache lines) does the cache have? Cache lines are also called blocks.

Entries = $2^{\text{index bits}} = 2^5$ lines

1.2

Tag	Index	Offset
31-12	11-6	5-0

1.2.1

What is the cache line size (in words)?

Cache line size = $2^{\text{offset bits}} = 2^6$ bytes = 2^4 words = 16 words

1.2.2

How many entries (cache lines) does the cache have? Cache lines are also called blocks.

Entries = $2^{\text{index bits}} = 2^6$ lines

2 Your Backpack

2.1

The Ca$h processor has a very strange ISA. Popular programs on it have the following instruction breakdown:

```
45% ALU
05% Branch
10% Store
40% Loads
```

They also have some interesting properties:

- all loads are to distinct, often random, memory locations
- all stores are to locations that have been loaded from before.
- there are no forward branches.
- branch targets are never more than 5 instructions back.

Assume there is no branch prediction. The following questions deal with the L1 cache behavior for the Ca$h processor.

For each of these pairs, identify which is likely to be higher, and why.

- **Spatial locality of instruction cache references > spatial locality of data cache references.**
  The data-cache has no spatial locality since it is randomly accessed.

  Instruction accesses are highly sequential, since there are very few branches, and even the branches contribute to spatial locality since they branch no more than 5 instructions back.

- **Temporal locality of instruction cache references < spatial locality of instruction cache references.**
  I-cache temporal locality occurs only when there is a branch. Since there are very few, this is lower than the spatial locality.

- **Instruction cache hit rate > instruction cache miss rate.**
  Hit rate = 1 - miss rate. Since I-cache spatial locality is extremely high, the hit rate is likely to be very high as well.

  This is assuming a cache line can hold more than 1 instruction! If a cache line only fits one instruction, spatial locality doesn't help.

- **Data cache hit rate < instruction cache hit rate.**
  The data cache has a terrible hit rate – loads are almost always misses because they are random. Stores might always be hits, but if the loads evict data from other loads, stores might miss too.

- **Data cache miss rate > data cache hit rate.**
  See above.

- **Average CPI for loads > average CPI for stores. (assume that on a cache hit, the CPI = 1 for both).**
  The loads are almost always guaranteed of missing in the cache. The stores might hit, depending on how close they are to the load from the same address.

2.2

Both your L1 instruction and data caches separate a 32-bit address as follows:

```
bits  0 -  3 = offset
bits  4 - 14 = index
bits 15 - 31 = tag
```

What is the size of each cache? How much space is required to store the tags for the L1 instruction cache?

Size of cache line: $2^{\text{offset bits}} = 2^4 = 16$ bytes
Number of cache lines: $2^{\text{index bits}} = 2^{11} = 2048$
Total cache size: $16 * 2048 = 32$KB
Total tag size: $17 * 2048 = 34$Kb

2.3

Refer to the cache structure from `2.1b`. For each of the following, identify whether it might increase or decrease (or it's impossible to tell) the hit rate for the instruction cache and data cache.

2.3.1

Increase the offset to bits 0-9, decrease the index to 10-14.

Increasing the offset and decreasing the index keeps the cache size the same. There are fewer cache lines, but they are longer. Longer cache lines expand spatial locality by holding values from more adjacent addresses. Fewer cache lines can lead to more conflict misses, so the window for temporal locality is worse. e.g. *Imagine having only one cache line – even if a program has high temporal locality, if there is one other address in between two of the same addresses, you get no locality from the cache.*

For this machine, the I-cache accesses (instruction addresses) have high spatial locality, so this would increase the hit rate. The D-cache accesses have bad temporal locality, so with fewer cache lines, there would be more conflict misses and therefore decrease the hit rate.

2.3.2

Decrease the offset to bit 0, shift the index to bits 1-11, increase the tag to bits 12-31.

With no offset bits, the cache lines hold 1 byte. The number of cache lines is the same, so the cache size is reduced. This would decrease the hit rate for both caches, because of the drastic reduction in cache size. It would hurt the instruction-cache more, however, because its spatial locality is useless.

3 Too Small a Backpack

You have a 2-way set associative L1 cache that is 8KB, with 4-word cache lines. Writing and reading data to L2 takes 10 cycles. You get the following sequence of writes to the cache – each is a 32-bit address in hexadecimal:

```
0x1000
0x1004
0x1010
0x11c0
0x2000
0x21c0
```

```
0x3400
0x3404
0x3f00
0x2004
0x1004
```

Cache line size: 4 words = 2^4 bytes
Offset size: $\log_2(\text{cache line size}) = \log_2(2^4) = 4$ bits
# of lines: $\frac{\text{cache size}}{\frac{\text{cache line size}}{\text{\# of sets}}} = \frac{2^{13}}{\frac{2^4}{2}} = 2^8$
Index size: $\log_2(\text{\# of lines}) = \log_2(2^8) = 8$ bits
Tag size: $32 - 4 - 8 = 20$ bits

Address	Index	Tag	Access Type
0x1000	0x00	0x1	Compulsory miss
0x1004	0x00	0x1	Hit
0x1010	0x01	0x1	Compulsory miss
0x11c0	0x1c	0x1	Compulsory miss
0x2000	0x00	0x2	Compulsory miss
0x21c0	0x1c	0x2	Compulsory miss
0x3400	0x40	0x3	Compulsory miss
0x3404	0x40	0x3	Hit
0x3f00	0xf0	0x3	Compulsory miss
0x2004	0x00	0x2	Hit
0x1004	0x00	0x1	Hit

3.1

How many cache misses occur with an LRU policy?

7 compulsory cache misses.

3.2

How many cache misses occur with a most-recently used policy?

There are no evictions; as a result, changing the replacement policy does not effect the number of cache misses.

3.3

Would the miss-rate increase or decrease if the cache was the same size, but direct-mapped? Explain.

If the cache was direct-mapped, you would have double the cache lines, so 1 extra index bit. This would cause 0x1000 and 0x2000 to still go to different cache lines, so it would not affect the miss-rate.

3.4

How long does a read-miss eviction take if the cache is write-back, write-allocate? What about a write-miss? (Assume the cache line is dirty)

On a read-miss, assuming the L1 cache line chosen to evict is dirty, first write back the old data to L2 which takes 10 cycles. Then read the new data from L2 in 10 cycles to write to L1. The L1 cache line is now marked clean. The caches could potentially do both at the same time though. Total time for a read-miss can be either 10 or 20 cycles.

For a write-miss, assuming the L1 cache line chosen to evict is dirty, write the old data back to L2 in 10 cycles, read the new data from L2 in 10 cycles to write in the L1 cache line, then write the new data to the L1 cache line. The L1 cache line remains dirty. Total time for a write-miss is 20 cycles.

3.5

How long does a read-miss eviction take if the cache is write-through, write-allocate? What about a write-miss?

On a read-miss, the old data in the L1 cache line does not need to be written to L2. Read the data from L2 in 10 cycles to write to L1. Total time for a read-miss is 10 cycles.

On a write-miss, since the L1 cache is write-through, the new data needs to be written to L2 in 10 cycles. However, since the L1 cache is also write-allocate, the new data needs to be written to L1 as well. The old data evicted in the L1 cache line does not need to be written to L2, but reading the new data from L2 to write to L1 will take 10 cycles. Total time for a write-miss is 20 cycles.

4 Los Angeles

You have an L1 data cache, L2 cache, and main memory. The hit rates and hit times for each are:

```
50%  hit rate, 2   cycle hit time to L1.
70%  hit rate, 15  cycle hit time to L2.
100% hit rate, 200 cycle hit time to main memory.
```

4.1

What fraction of accesses are serviced from L2? From main memory?

L1 miss rate = $1 - 0.5 = 50\%$
Fraction serviced from L2: (L1 miss rate)*(L2 hit rate) = $0.5 * 0.7 = 35\%$

L2 miss rate = $1 - 0.7 = 30\%$
Fraction serviced from main memory: (L1 miss rate)*(L2 miss rate) = $0.5 * 0.3 = 15\%$

4.2

What is the miss rate and miss time for the L2 cache?

L2 miss rate = $1 - 0.7 = 30\%$
L2 miss time = main memory hit time = 200 cycles

4.3

What is the miss rate and miss time for the L1 cache? (Hint: depends on previous answer).

L1 miss rate = $1 - 0.5 = 50\%$

L1 miss time = (L2 hit rate)*(L2 hit time) + (L2 miss rate)*(L2 miss time)
$= 0.7 * 15 + 0.3 * 200 = 70.5$ cycles

4.4

If main memory is improved by 10%, what is the improvement in L1 miss time?
New main memory hit time: $\frac{200}{0.9} = 180$ cycles

New L1 miss time = (L2 hit rate)*(L2 hit time) + (L2 miss rate)*(L2 miss time)
$= 0.7 * 15 + 0.3 * 180 = 64.5$ cycles

Improvement: $\frac{70.5}{64.5} = 1.093$ / -6 cycles

4.5

You remove the L2 to add more L1. As a result, the new L1 hit rate is 75%. What is the improvement in L1 miss time?

New L1 miss time = main memory hit time = 200 cycles
Improvement: $\frac{70.5}{200} = 0.353$ / +129.5 cycles

5 Cache Design

You have 3 cache designs for a 16-bit address machine.

```
Dictator:
Direct-mapped cache.
Each cache line is 1 byte.
10-bit index, 6-bit tag.
1 cycle hit time.

Oligarch:
2-way set associative cache.
Each cache line is 1 word (4 bytes).
7-bit index, 7-bit tag.
2 cycle hit time.

Democle:
fully associative cache with 256 cache lines.
Each cache line is 1 word.
14-bit tag.
5 cycle hit time.
```

5.1

What is the size of each cache?

Dictator: 1024 1B lines = 1KB.
Oligarch: 128 4B lines * 2 ways = 1KB
Democle: 256 4B lines = 1KB

5.2

How much space does each cache need to store tags?

Dictator: 1024 x 6-bit tags = 6Kb
Oligarch: 256 x 7-bit tags = 1792 bits
Democle: 256 x 14-bit tags = 3584 bits

5.3

Which cache design has the most conflict misses? Which has the least?

Dictator probably has the most conflict misses, since it is direct mapped. Democle, because it is fully associative, can never have conflict misses.

5.4

If someone told you the hit rate for the 3 caches is 50%, 70% and 90% but did not tell you which hit rate corresponds to which cache, which cache would you guess corresponded to which hit rate? Why?

Since the caches are the same size and the reason in the previous answer, Dictator is 50%, Oligarch is 70%, and Democle is 90%.

5.5

Assuming the miss time for each is 20 cycles, what is the average service time for each? (Service Time = (hit rate)*(hit time) + (miss rate)*(miss time)).

Based on the previous answer, average service time:

Dictator: 0.5*1 + 0.5*20 = 10.5 cycles
Oligarch: 0.7*2 + 0.3*20 = 7.4 cycles
Democle: 0.9*5 + 0.1*20 = 6.5 cycles.