王柯皓 2021533025 wangkh@shanghaitech.edu.cn

Computer Architecture Homework 5

Spring 2023, April

1 True or False

Please fill your answer (T or F) in the table below:(10 pts)

1	2	3	4
F	T	F	F

1. Any cache miss that occurs when the cache is full is a capacity miss.
2. Cache replacement policy is used for choosing which cache line should be evicted.
3. For a 1-level, LRU cache, cache blocking will greatly speed up matrix transposition when the cache size is much larger than the matrix size.
4. In a 32-bit machine (word size is 32 bit), the clock frequency is 2GHz. It takes 3 cycles to access L1, L1 cache has a hit rate of 30%.It takes 40 cycles to access L2, L2 cache has a hit rate of 80%. It takes 400 cycles to access physical memory. The average memory access time is 87ns.

2 AMAT Analysis

A program is running on a system with a single-level, write-back, 512B direct-mapped cache that has the block size of 4B. It traverses an array with step size = 1. Choose how the modification changes each component of AMAT. If there are multiple choices, select them all.(15 pts)

Modification	Hit Time	Miss Rate	Miss Penalty
Change to 2-way Associativity (cache size and block size keeps the same)	A.Increase B.Decrease C.No effect	A.Increase B.Decrease C.No effect	A.Increase B.Decrease C.No effect
Your Choice	B	C	A

3 N-way Set Associative Cache

A program is running on a byte-addressed system with a single-level cache, where memory addresses are 10 bits long. The entire cache is shown below. The block size of this cache is 16 B.

Index	Tag1	Tag2
0b00		
0b01		
0b10		
0b11		

1. Is this cache direct-mapped, set-associative, or fully-associative? If it is associative, also write down its associativity.(5 pts) Set-associative.

Associativity = 2.

2. Write down the width (in bits) of Tag, Set index and Block offset.(5 pts)

Tag	Set index	Block offset
4	2	4

3. How many bytes of data can this cache contain? Please show your process.(5 pts)

Total capacity of cache = 2^2 ✖ 2 ✖ 2^4 = 2^7 bits = 2^4 bytes = 16 bytes

4. We will access the data of addresses as follows, one by one. Fill in the blanks, and determine if each access would be a hit or miss based on the cache state shown above (suppose the cache is empty at first). This cache uses LRU replacement policy. If it's a miss, classify the possible miss type(s), select all possible miss types.(20 pts)

Address				Your Choice
0b0011010000	A.Hit	B.Compulsory miss	C.Conflict miss	B
0b0011011011	A.Hit	B.Compulsory miss	C.Conflict miss	A
0b1101000100	A.Hit	B.Compulsory miss	C.Conflict miss	B
0b0011001100	A.Hit	B.Compulsory miss	C.Conflict miss	B
0b0110100010	A.Hit	B.Compulsory miss	C.Conflict miss	B
0b0010111100	A.Hit	B.Compulsory miss	C.Conflict miss	B
0b1110100010	A.Hit	B.Compulsory miss	C.Conflict miss	B
0b1011000000	A.Hit	B.Compulsory miss	C.Conflict miss	C
0b1111111111	A.Hit	B.Compulsory miss	C.Conflict miss	B
0b0011001101	A.Hit	B.Compulsory miss	C.Conflict miss	A

4 Cache Friendly Programming

Assume a 32-bit machine, suppose the cache has the following settings:

Cache levels	1
Block size	16 bytes
Number of sets	4
Cache size	128 bytes
Block replacement policy	LRU

Suppose the following code is running on a system with the above cache. Answer the questions below and show your progress/explanation.

```
#define array_size    64    //line 1
#define repeat_times  1
#define step_size     2
int main(){
    int array[array_size] = {};
    for (int r = 0; r < repeat_times; r++){
        for (int i = 0; i < array_size; i += step_size){
            array[i] = array[i] + 2333;
        }
    }
    return 0;
}
```

1. What is the total number of accesses to the cache?(5 pts)

Since the array_size is 64, the inner loop will access half of the elements in the array, which is 32 elements, due to the step size of 2. And since the outer loop iterates only once, the total number of accesses to the cache = Number of elements accessed in the inner loop * Number of iterations of the outer loop = 32 * 1 = 32. So, the total number of accesses to the cache is 32.

2. What is the hit rate?(10 pts)

As the loop iterates, it will access a total of 32 / 2 = 16 unique cache blocks. Since the cache uses the LRU replacement, each time a new cache block is accessed, the oldest block in the set will be evicted. Since there are 16 unique cache blocks and only 2 blocks per set, there will be cache evictions during the loop. Therefore, the total number of cache misses will be equal to the number of unique cache blocks accessed, which is 16. So the hit rate = 16 / 32 = 50%. So, the hit rate for the given code and cache settings is 50%.

3. Which type(s) of miss occur(s)(5 pts)?

Compulsory miss: when the code first accesses the elements of the array in the inner loop, it is likely that none of the blocks corresponding to the accessed array elements are present in the cache.

Capacity miss: the cache size is 128 bytes and the block size is 16 bytes, which means the cache can hold 128/16 = 8 blocks at a time. If the code accesses more than 8 blocks during its execution, some of the blocks may be evicted from the cache and result in capacity misses.

Conflict miss: the iterations of the inner loop may cause multiple elements to map to the same cache set.

4. Suppose repeat_times **goes to infinity** (only for this question), what number will the hit rate converge to?(5 pts)

As the cache block size is 16 bytes, each cache block can accommodate 4 integers. Since the loop accesses every second element with a step size of 2, it will only access 32 integers, which is exactly the size of the cache. This means that all the accessed elements will fit into the cache. Since the cache size is equal to the total size of data accessed by the loop, there will be no cache misses and the hit rate will be 100% as long as the loop runs infinitely, as all the accessed data will always be present in the cache.

5. If repeat_times **is changed to 2** (only for this question), try to swap two lines of the above code to maximize the hit rate without disturbing the results. Which two lines will you choose and what is the maximized hit rate?(10 pts)

Line 6 and line 7.
The total number of cache hits is 16. Since the cache size is 128 bytes and the array size is 64 bytes, the entire array can fit in the cache without any cache misses.
Therefore, the hit rate after the modification is 100%.

6. **For the modified code in the previous question**, suppose again repeat_times **goes to infinity** (only for this question), what number will the hit rate converge to?(5 pts)

Since the entire array can fit in the cache without any cache misses, and the code repeats the same memory accesses infinitely, the hit rate will converge to 100%.