HW7.4. Cache Programming

Q1: The following C program is run (with no optimization) on a processor with a direct-mapped data cache (**the cache starts out empty**) with a size of 1 KiB and a block size of 32 bytes:

```
int i, j, array[256*256];

/* ... */

for (i = 1 ; i < 256 ; i++) {
    for (j = 0 ; j < 256 ; j++) {
        array[256*j] += array[256*j+i];
    }
}
```

Assume `sizeof(int) == 4` and `array == 0x0000 4000`.
The sizeof(int) is highly relevant for the following questions. How many ints can fit in a cache block?

Q1.1: For the first iteration of the outer loop (`i = 1`), what is the hit rate of this code?
Hint: How would you expand the expression array[x] += array[y]? How many memory accesses are in there? Which of those would miss?

Hit Rate= number (2 significant figures)

Q1.2: After the first `n` iterations of the outer loop, the hit rate changes. What is `n`?
*Hint: Try writing out the array indices for every iteration of the loop. Recall the sizeof(int). At what point will array[256*j+i] exceed a cache block?*

n= integer

Q1.3: After the first `n` iterations of the outer loop, the hit rate changes. What is the new hit rate of each iteration of the outer loop?
*Hint: From 1.2: if array[256*j+i] exceeds a cache block, how many misses would we have?*

Hit Rate= number (2 significant figures)

Q1.4: What is the overall hit rate of this code?
Hint: How many total memory accesses do we have for one iteration of the outer loop? How would you incorporate the pattern you observed from the previous questions?

Hit Rate= number (2 significant figures)

Q2: We decide to rewrite our code to be more cache-efficient, while maintaining the same behavior. Fill in the blanks of this code (write your answers in the same format as the original code):

```
int i, j, array[256*256];

/* ... */

for ([CODE A]; i < 256 ; i++) {
    for ([CODE B]; j < 256 ; j++) {
        array[[CODE C]] += array[[CODE D]];
    }
}
```

Hint: Coming from the previous questions, we now want to have more consecutive memory accesses on the inner loop so that we get more cache hits.

Q2.1: CODE A:

Q2.2: CODE B:

Q2.3: CODE C:

Homework 7

Assessment overview

Total points: 0/100

Score: 0%

Question

Value: 30

History:

Awarded points: 0/30

Report an error in this question

Previous question

Next question

Attached files

No attached files

Attach a file

Attach text

Q2.4: CODE D:

Q2.5: What is the new hit rate of this code?
Hint: How many total memory accesses do we have for one iteration of the outer loop? How many misses do we get now after code reordering?

Hit Rate= number (2 significant figures)

Save & Grade *20 attempts left* | Save only

Additional attempts available with new variants