## CS 3340 Computer Architecture – Fall 2019 – Mazidi

## Student’s Name: Medha Aiyah

## Homework: Cache Comparisons

## Worth 100 points

Objective: Compare two sorting algorithms in terms of number of instructions and cache utilization.

Turn in, zipped together: Bubble sort program, Selection sort program, Document containing counts and commentary

Instructions:

* Modify the Bubble sort demonstrated in class (<https://github.com/kjmazidi/CS3340/blob/master/Code%20Samples/zybook_4/bubble_sort.asm>) to sort 500 data items which are defined in the .data section. Remove any functionality in the program that is not sorting, such as printing the array before and after.
* Use the same data as you use in the Bubble sort but copy your Selection sort from Homework 5 into a program and modify it to run the same data. The two sorts should be in two different programs and should do nothing but sort the 500 integers in memory.
* Using the Instruction Counter tool, fill in the instruction comparison table below.
* Using the Data cache Simulator tool, fill in the cache comparison table below.
* Answer questions 1-5 below.

Grading Rubric:

20 points: Bubble Sort works (check memory to make sure it sorted)

20 points: Selection Sort works (check memory to make sure it sorted)

20 points: Instruction comparison table (below)

20 points: Cache comparison table (below)

20 points: Answer the questions below

Reminder:

To get to the tools:

* Assemble the program
* Use the Tools menu to find the tool
* Hit the ‘Connect to MIPS’ button on the tool
* Hit the green run arrow at the top of MIPS

Instruction Comparison Table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Number Instr. | R-type | I-type | J-type | Avg I/item |
| Bubble Sort | 1138000 | 454594 | 569755 | 113651 | 2276 |
| Selection Sort | 1504138 | 626999 | 628044 | 249095 | 3008.276 |

Cache Comparison Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Memory Access Count | Cache Hit Count | Cache Miss Count | Cache Hit Rate |
| Bub Sort | 340452 | 326426 | 14026 | 96% |
| Sel Sort | 251292 | 195932 | 55360 | 78% |

Questions:

1. How similar are the two algorithms in terms of average instructions executed per item sorted? Did this surprise you?

*For the Bubble Sort, the average instructions executed per items sorted was 2276. For the Selection Sort, the average instructions executed per items sorted was 3008.276. Based on these two numbers, the average instructions were similar but the number for the Selection Sort was greater than the number for the Bubble Sort. These numbers did not surprise me because I previously knew that the selection sort is a lot more efficient than the Bubble Sort. In order for the selection sort to be more efficient than the bubble sort, there should be a greater number of the average instructions running which was clearly shown.*

1. How similar is the distribution of R, I, and J instructions for the two algorithms? Comment on why this might be the case.

*For the Bubble sort, the instructions for R-type was around 454,000, the instructions for I-type it was around 570,000, and the instructions for the J-type was around 114,000. For the Selection sort, the instructions for the R-type was around 627,000, the instructions for the I-type was around 628,000, and the instructions for the J-type was around 249,000. By looking at the following numbers, one can observe that the distribution for the R, I, and J instructions are much greater for the selection sort compared to the bubble sort. The distribution of the instructions is greater for the selection sort because the selection sort in general is a more complex algorithm therefore there will be more instructions that will be used in order for this sort to work.*

1. Compare the hit rates of the two algorithms using default settings. Given your understanding of the patterns in which these two algorithms access memory, how do you explain this difference?

*The Cache Hit Rate percentage shows how amount of the instructions that are reused in a following sort. The Cache Hit Rate for the Bubble Sort was 96% whereas the Cache Hit Rate for the Selection Sort was 78%. There is a difference between the two Cache Hit Rates because the two algorithms for the Bubble Sort and the Selection Sort are different. Since the Bubble Sort’s algorithm is not as complex as the Selection Sort, it reuses more instructions therefore having a higher Cache Hit Rate.*

1. Try modifying the placement or replacement schemes? Did you get a different result?

*When I modified the placement scheme to Fully Associative for the selection sort and it is LDU for the replacement scheme, I get different results than what is in the table. When I modified the placement scheme to N-way set associative for the selection sort and it is LDU for the replacement scheme, I do not get different results than what is in the table. When I modified the replacement scheme to Random and used a placement scheme of Direct mapping for the selection sort, I did not get any different results than what is in the table. When I modified the replacement scheme to Random and used a placement scheme of Fully Associative for the selection sort, I got different results than what is in the table. When I modified the replacement scheme to Random and used a placement scheme of N-way set associative for the selection sort, I do not get different results than what is in the table.*

*For the bubble sort when I modified the placement scheme to Fully Associative for the selection sort and kept the replacement scheme as LDU, I get different results (for the cache miss count) than what is on the table. For the bubble sort, when I modified the placement scheme to N-way set associative and keep the replacement scheme as LDU, I get the same results as what is in the table. For the bubble sort when I change the replacement scheme to random and have a placement scheme of direct mapping, I do not get different results than what is in the table. For the bubble sort, when I change the replacement scheme to random and have a placement scheme of direct mapping, I get different results than what is in the table. For the bubble sort, when I change the replacement scheme to random and have a placement scheme of N-way set associative, I do not get different results than what is on the table.*

*The main reason that the Fully Associative placement scheme was different compared to the other two placement schemes is because it has a structure where the location is not fixed and a block can be placed in any location in the cache.*

1. Try modifying block size or number of blocks. Do not change the total cache size. Did you get different results?

*For the bubble sort, when I modify the block size or the number of blocks, I do get different results. For the selection sort, when I modify the block size or the number of blocks, I do get different results. The main reason I got different results by modifying the block size is because the larger the block size is, there is a higher associativity than the primary cache given the focus of reducing miss rates.*