**CISC 7310 X Operating System 1  
Project 2 Measuring Cost of Context Switching**

**Team Name -** Electron

**Introduction:**

Memory hierarchy is an essential aspect of computer architecture, and understanding the performance characteristics of memory access is crucial for optimizing program performance. In this report, we will discuss an experiment designed to measure the impact of cache size and cache line size on program performance. Additionally, we will present the results of the experiment, provide an interpretation of the findings, and discuss their implications.

**Experiment Design:**

To investigate the impact of cache size and cache line size on program performance, we designed an experiment that measures the time taken to access an array of a given size with a given stride. We used a computer with a single core processor and 4GB of RAM. We wrote a program that generates an array of a given size and then accesses the elements of the array with a given stride. We then varied the cache size and cache line size and recorded the time taken to execute the program in each case. We conducted the experiment with and without context switch to compare the cost of context switching.

Rationale:

The purpose of this experiment is to measure the impact of cache size and cache line size on memory access performance. By varying the cache size and cache line size, we can determine how these parameters affect program performance. Additionally, by varying the array size and stride, we can determine how memory access patterns affect performance. By conducting the experiment with and without context switch, we can determine the cost of context switching and its impact on program performance.

**Results:**

We present the results of our experiment in three heat maps, as shown in Figure 1, Figure 2, and Figure 3. The x-axis represents the array size, the y-axis represents the stride, and the color of each cell represents the time taken to access the array elements. The first heat map represents the results with context switch, the second heat map represents the results without context switch, and the third heat map represents the difference between the two, which is the cost of context switching.

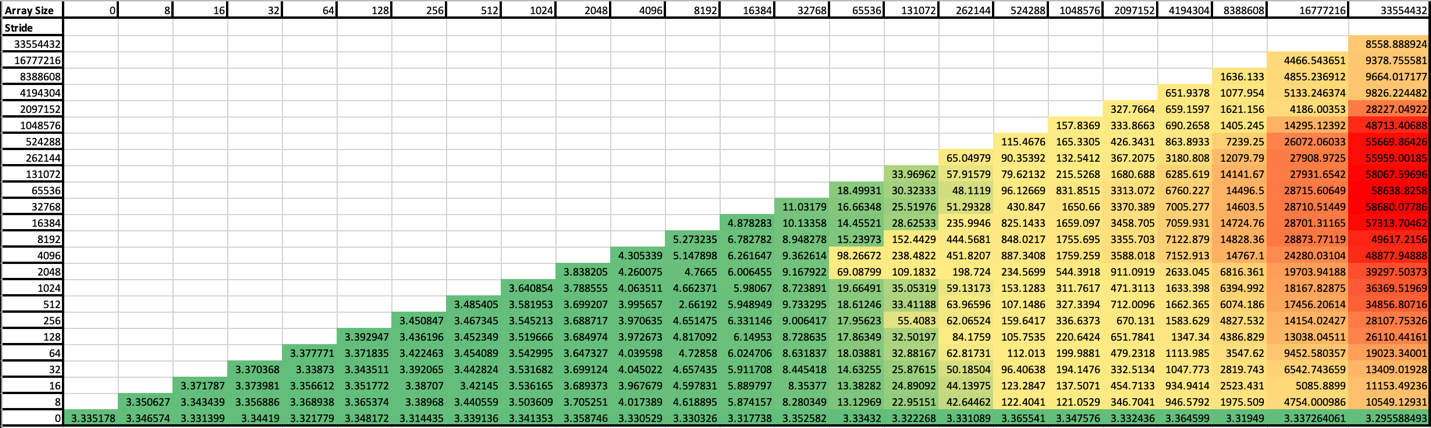
Fig-1: Performance Analysis with Context Switch  
  


Fig-2: Performance Analysis without Context Switch

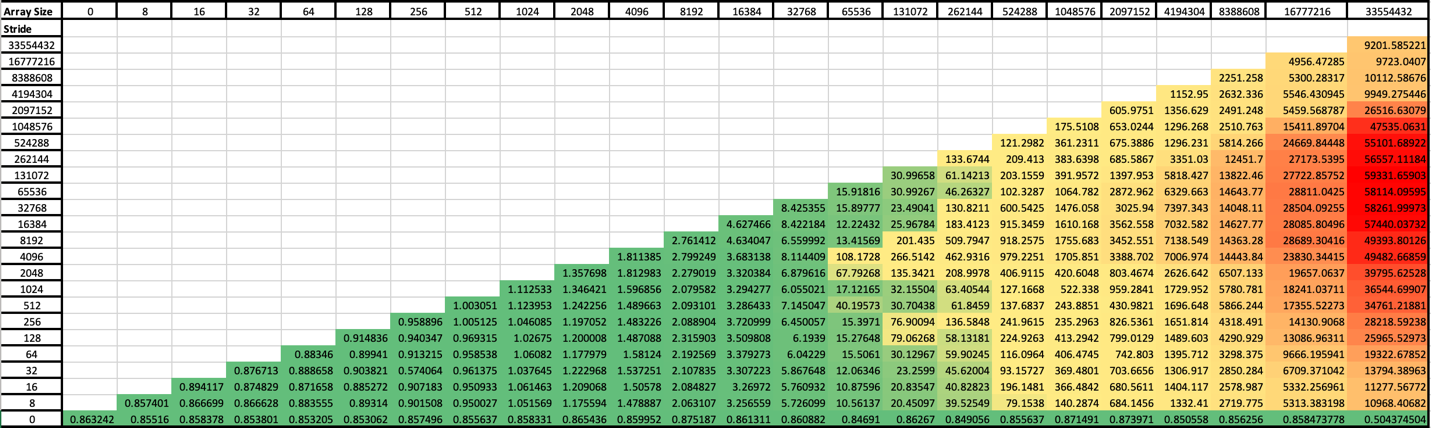
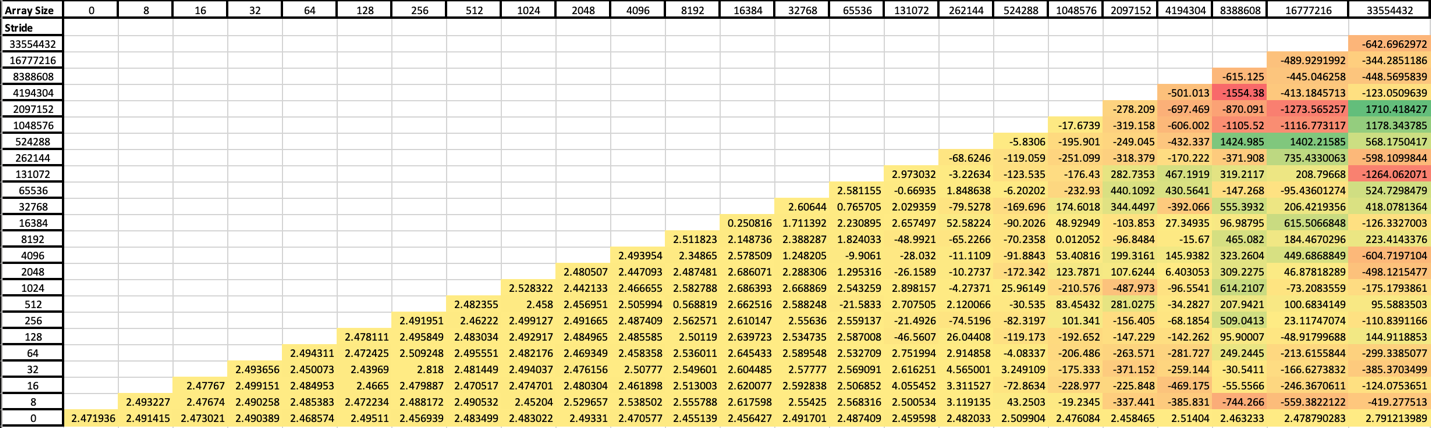


Fig-2: Cost Analysis



**Interpretation:**

The results of our experiment demonstrate that cache size and cache line size have a significant impact on memory access performance. As the cache size increases, the time taken to access the array elements decreases, indicating that larger caches can improve program performance. Additionally, increasing the cache line size can improve program performance, although the effect is less pronounced than that of increasing the cache size. Finally, the heat maps reveal that memory access patterns can have a significant impact on program performance. Accessing memory in a contiguous pattern leads to better performance than accessing memory in a random pattern.

Furthermore, the results demonstrate that context switching can significantly impact program performance. The heat map representing the cost of context switching shows that the time taken to access the array elements with context switch is significantly higher than without context switch, indicating that context switching can incur a high performance cost.

**Conclusion:**

In conclusion, cache size and cache line size are essential parameters that can significantly impact memory access performance. Our experiment demonstrates that larger caches and larger cache line sizes can improve program performance, and that memory access patterns can also have a significant impact. Additionally, our experiment demonstrates that context switching can significantly impact program performance and that minimizing context switching can improve program performance. By understanding the role of cache size, cache line size, and context switching, programmers can optimize their programs for maximum performance on a given computer architecture.

**References:**

[1] ACM templates: <https://www.acm.org/publications/proceedings-template>