Matrix Memory Access Patterns

1. Introduction

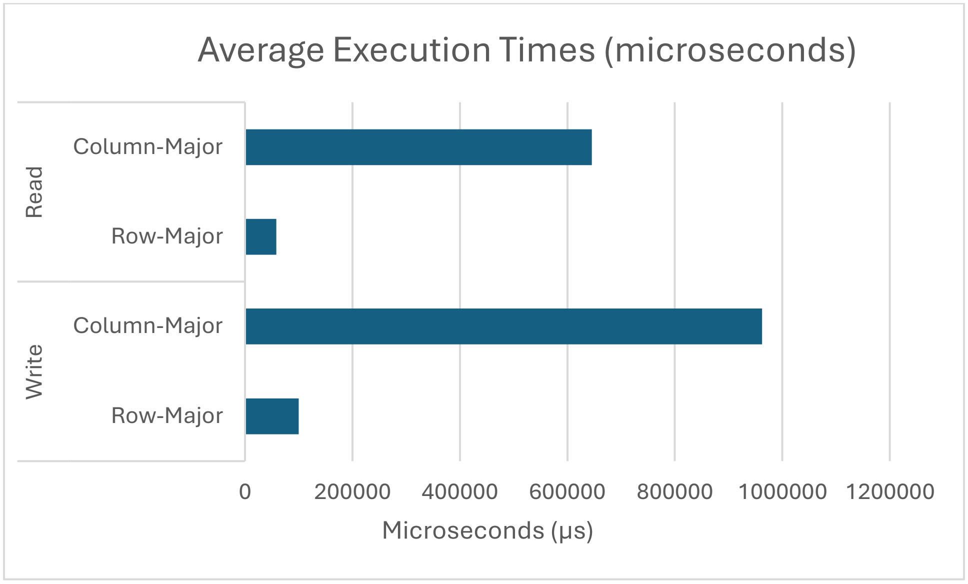
The objective of this experiment was to analyze the performance of large matrix operations in different memory access patterns. Specifically, the program focused on row-major and column-major memory access patterns, measuring the impact of these access patterns on execution time. The experiment was conducted using a 2D matrix of size 20480 rows, with each row occupying one page (4096 bytes). The matrix operations tested were:

1. **Write Row-Major**: Writing values to the matrix in row-major order.
2. **Write Column-Major**: Writing values to the matrix in column-major order.
3. **Read Row-Major**: Reading values from the matrix in row-major order.
4. **Read Column-Major**: Reading values from the matrix in column-major order.

The goal was to measure the time taken for each of these operations and interpret the results in terms of system performance and memory access behavior.

2. Experiment Results

The following table shows the average execution times for each operation:



In the table above it shows that the row-major access resulted in a faster execution time compared to column-major access. Also reading data from the matrix was faster than writing data into the matrix. This is very apparent when comparing the times of column-major access from read to writing data.

3. Conclusion

This experiment successfully demonstrated the impact of memory access patterns on the performance of large matrix operations. The row-major access pattern consistently outperformed the column-major access pattern, mainly due to better cache locality. The experiment also highlighted the challenges of managing large datasets that exceed available memory, particularly with respect to memory access patterns and cache utilization.