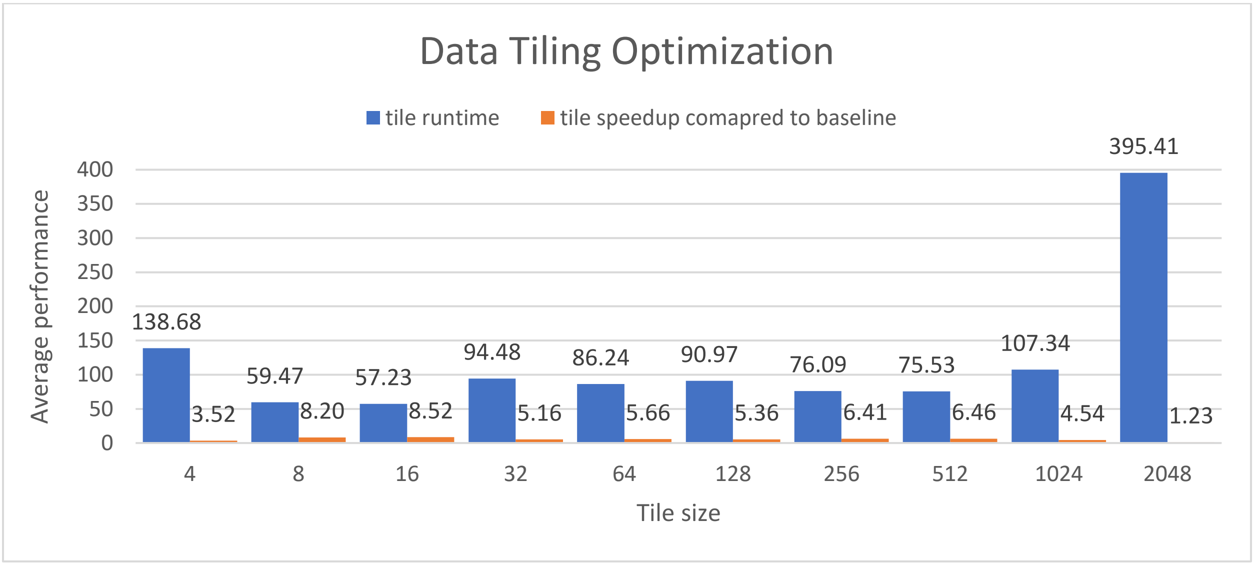
ENSC254 Lab 5 Report

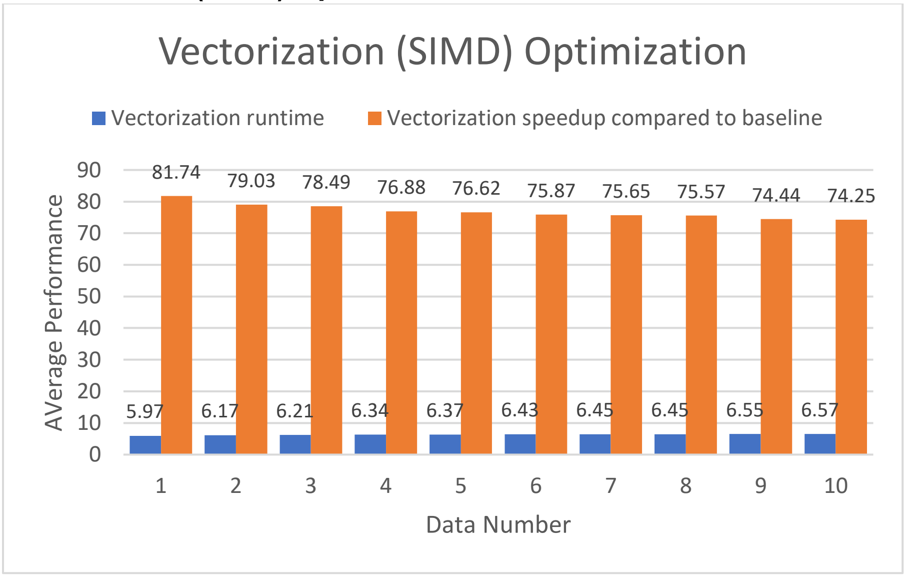
In this lab report, we conducted multiple runs of the baseline case and each optimization, using the same parameters such as tile size and thread number. Each configuration was executed three times, and the average results were obtained for analysis. The average of baseline version is 487.7555417800 seconds.

Data Tiling Optimization:



In this experiment, we kept the tile size consistent for loops I, j, and k, ranging from 2^2 to 2^{11}. Among the various tile sizes tested, we observed that using a tile size of 16 resulted in the lowest average runtime of 57.23 and achieved a speedup of 8.52 compared to the baseline version of the program.

Vectorization (SIMD) Optimization:

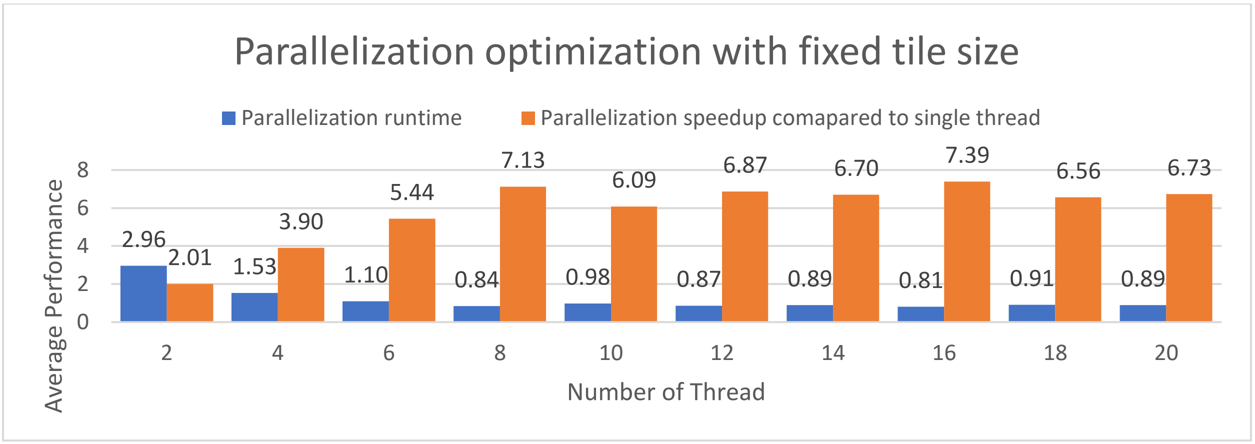


Data Number	Tile size of I loop	Tile size of j loop	Tile size of k loop
1	32	2048	32
2	32	1024	32
3	16	2048	16
4	128	2048	128
5	64	2048	64
6	512	2048	512
7	256	2048	256
8	16	1048	16
9	32	512	32
10	32	4096	32

For the SIMD optimization, we explored different tile sizes for loops I, j, and k. A total of 85 combinations for each loop's tile size were tested. The bar graph above presents the top 10 performing sets. The data numbers in the graph correspond to the table on the right, indicating the specific tile sizes for each loop.

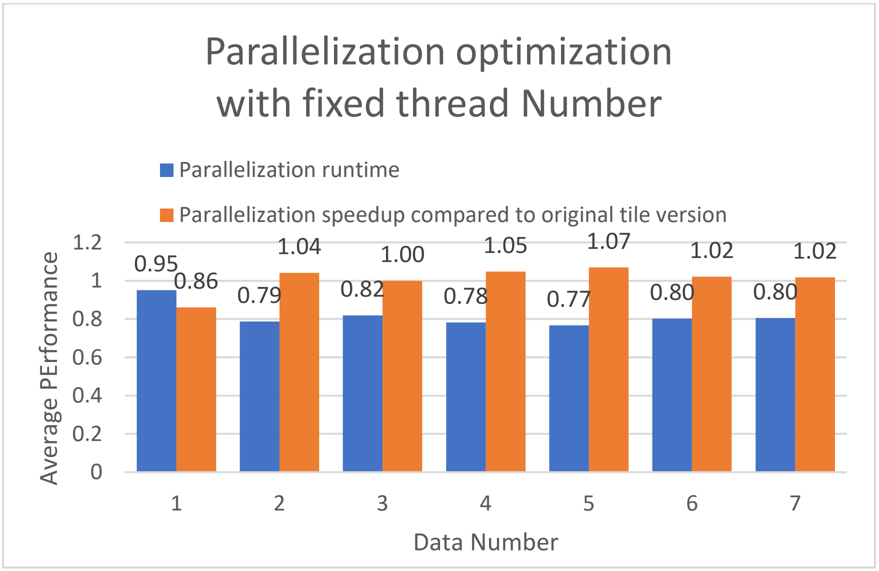
We observed that using tile sizes of 32, 2048, and 32 for loops I, j, and k, respectively, resulted in the shortest runtime and a remarkable speedup of 81.74 times faster compared to the baseline version.

Parallelization Optimization: Different Thread Numbers, Fixed Tile Size



In this section, we utilized the optimal tile size (32, 2048, 32) from the previous analysis and measured the runtime with thread numbers ranging from 2 to 20. With 16 threads, the runtime was at its lowest, achieving an 0.81 seconds. This configuration exhibited a speedup of 7.39 times compared to the single-threaded version of the program.

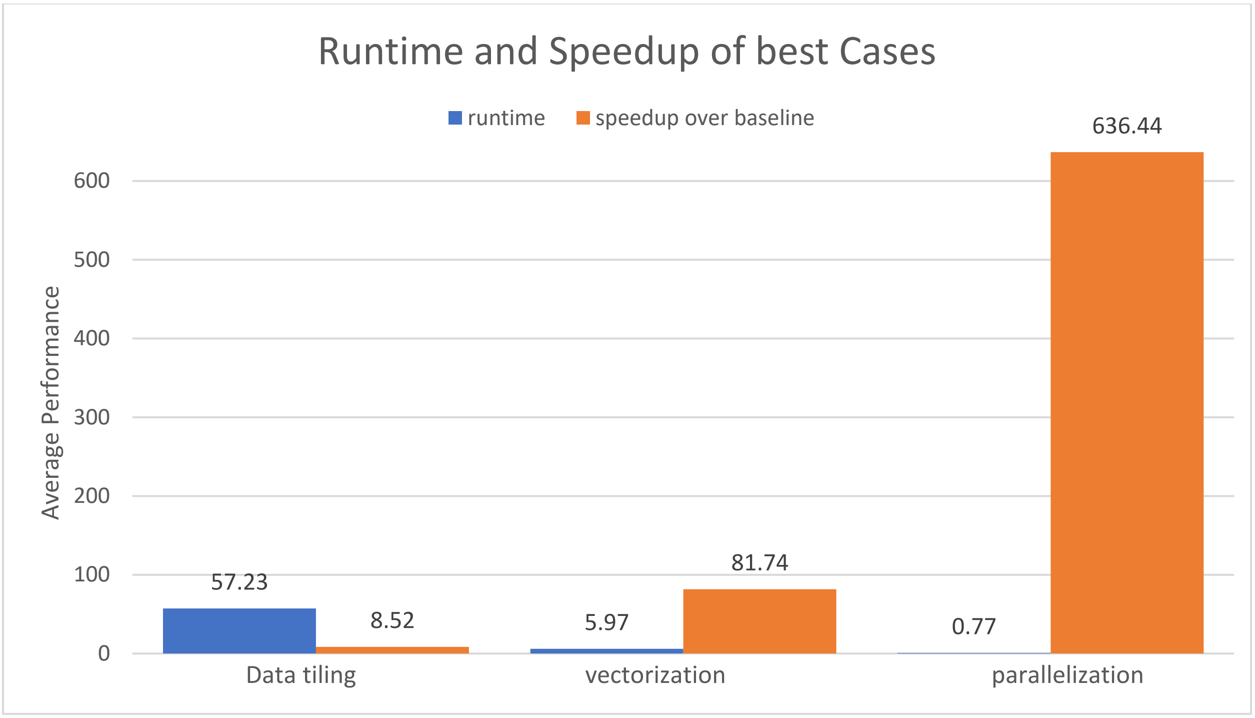
Parallelization Optimization: Different Thread Numbers, Fixed Thread Number



Data Number	Tile size of I loop	Tile size of j loop	Tile size of k loop
1	4	1024	4
2	8	1024	8
3	16	16	16
4	16	1024	16
5	16	1024	4
6	16	1024	2
7	8	2048	2

After determining the optimal thread number of 20, we proceeded to experiment with various combinations of tile sizes for loops I, j, and k. Among potential combinations, the configuration with tile sizes of 16, 1024, and 4 (for loops I, j, and k, respectively) achieved the most favorable runtime of 0.7663846 seconds and a speedup of 1.07 compared to original tile size.

Step-by-Step Speedup over Baseline Version



In lab 5, we carried out three major optimizations to improve the program's performance. First, data tiling proved effective, particularly when employing a tile size of 16 for each loop. This optimization resulted in a remarkable speedup of 8.52 compared to the baseline.

Next, we explored vectorization and tested various loop tile size combinations. The most improvement achieved was a speedup of 81.74.

Finally, we implemented parallelization optimization using OpenMP pragmas, experimenting with different thread numbers and tile sizes. This collective effort yielded a speedup of 636.44 over the baseline code, reducing the runtime to 0. 7663846 seconds.