COL380-A0:- Analysis of Sequential Matrix Multiplication

Yash Bansal (2022CS51133)

1 Time Reported by Python Script

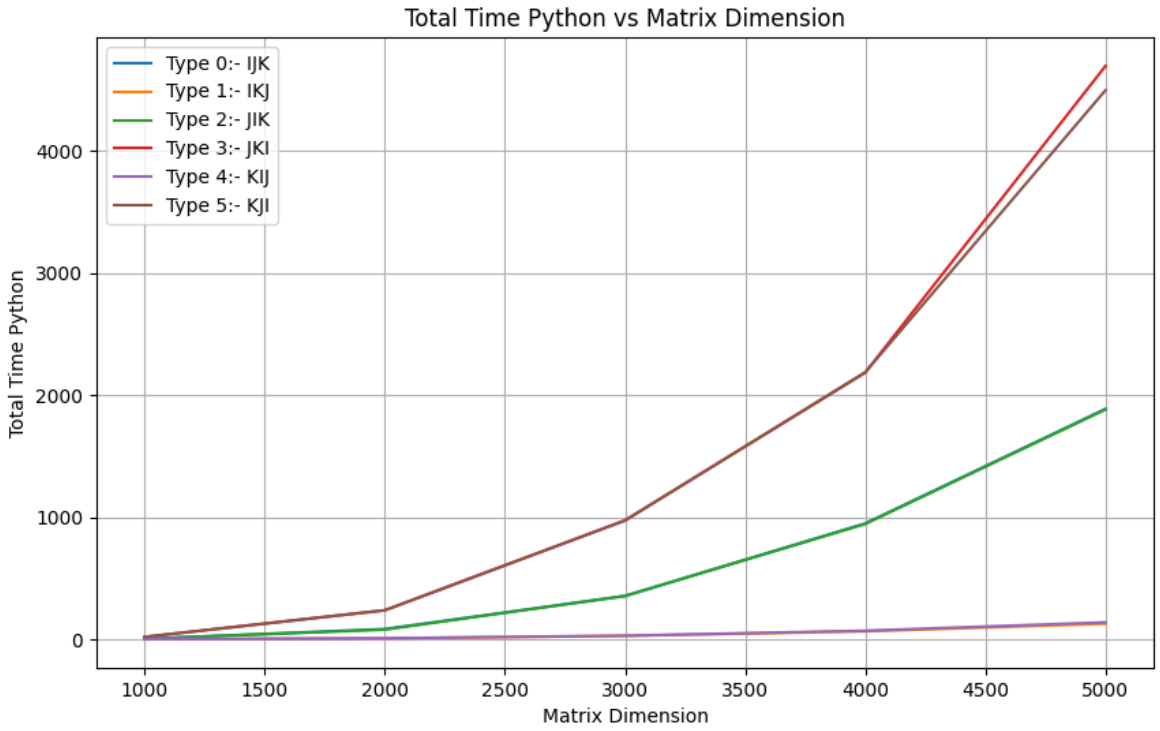

Figure 1: Plot for Time reported by the Python Script

- The time reported by the Python script increases consistently as the size of the matrix increases, which is expected as the time complexity varies as n cube.
- For all the six types, the following order of time taken is observed:-

$$JKI \approx KJI \geq JIK \approx IJK \geq IKJ \approx KIJ$$

This order is due to the difference in cache hit rates for different loop permutations, as will be discussed later.

2 Times reported by Perf

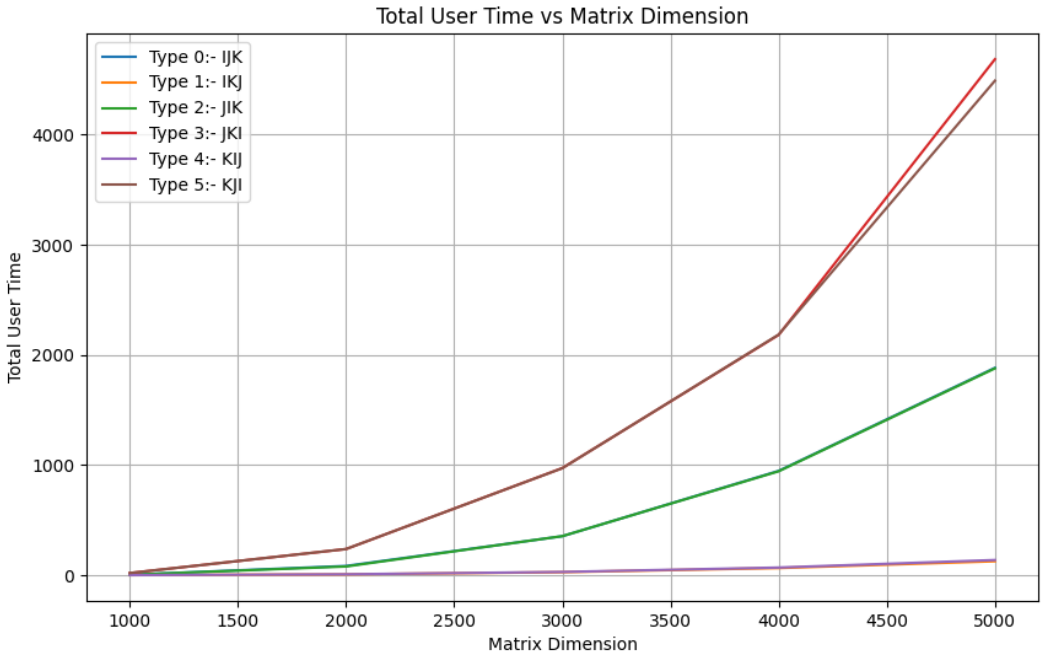

Figure 2: Plot for Total User time reported by Perf

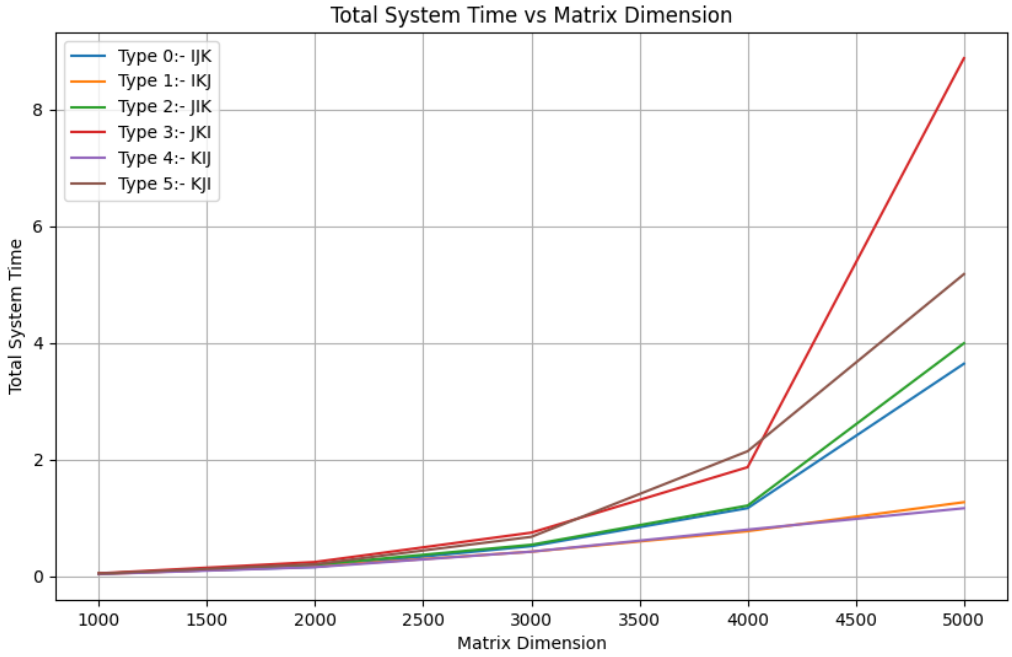

Figure 3: Plot for Total System time reported by Perf

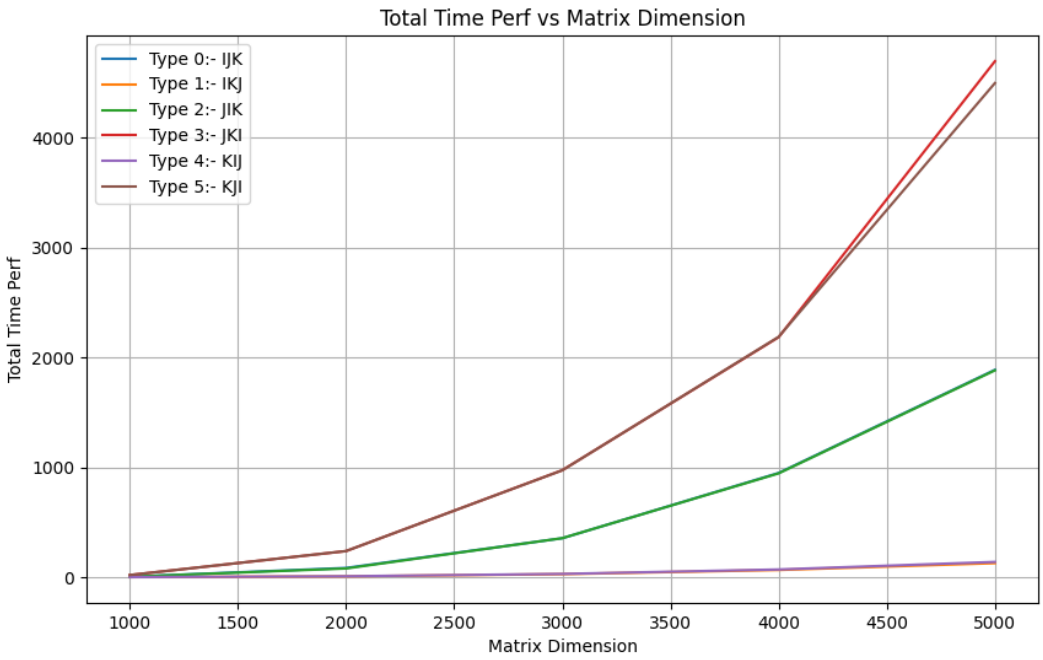

Figure 4: Plot for Total Time elapsed reported by Perf

- The total time elapsed reported by perf is slightly less than that reported by the Python script, with a small difference of approximately 0.4-0.5 seconds. This small difference is due to Python taking the time to invoke the subprocess to run the cpp code.
- The same trends for total user time are followed for both perf and Python time.
- The total system time also increases with matrix size, with the same trend as user times.

3 Matrix Multiplication time by Perf

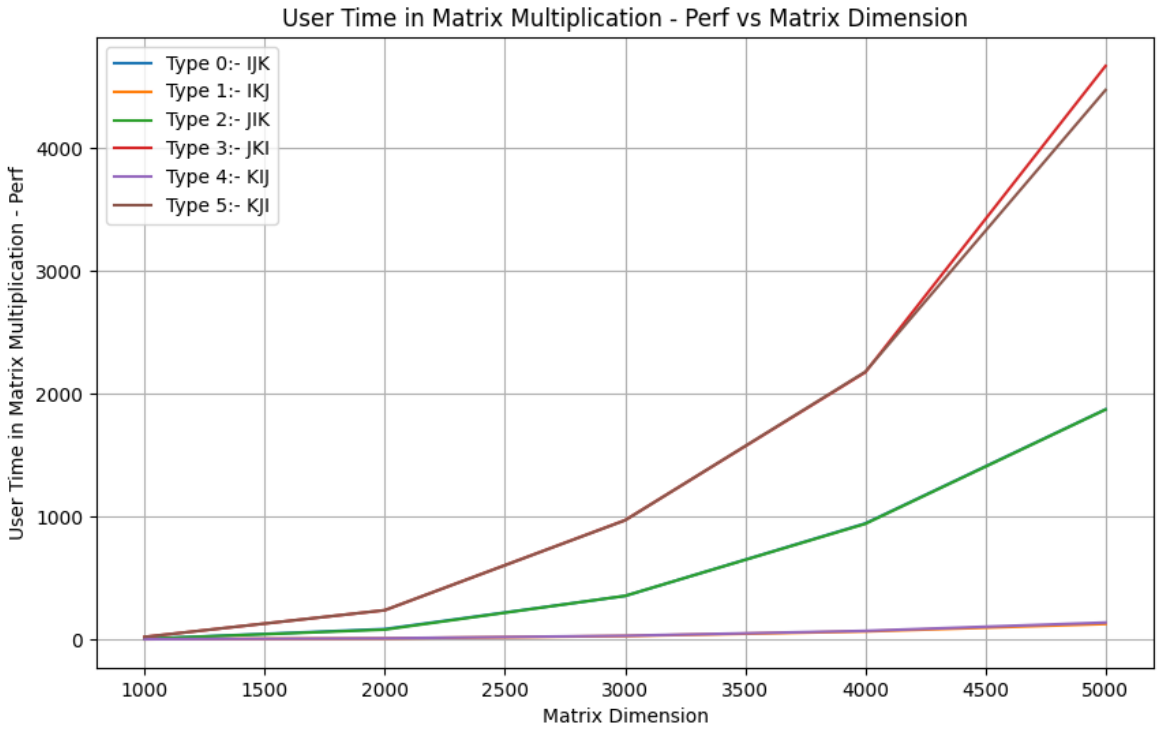

Figure 5: Plot for Matrix Multiplication time reported by Perf

- The matrix multiplication time also follows the same trends as the total time taken by the code. This is because this is the major time taking part of the code.

4 Plots for Cache data

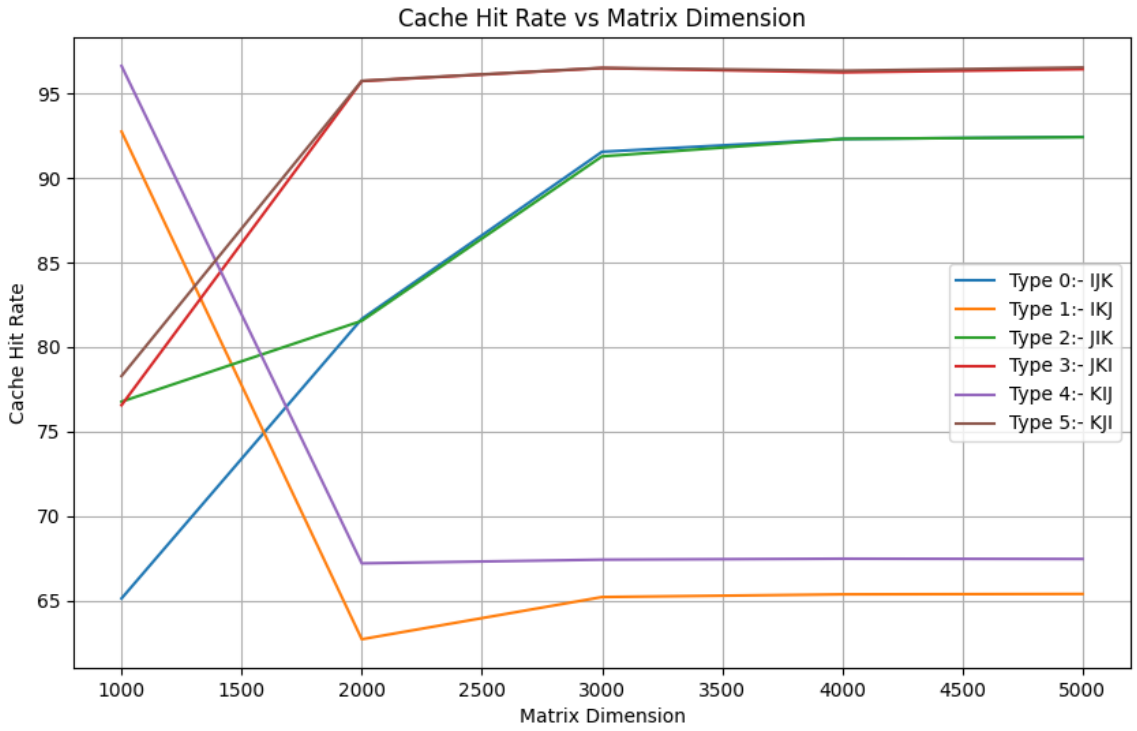

Figure 6: Plot for Cache Hit Rate

- Except for the case when the size of matrix is 1000, the cache hit percentage follow the following order:-

$$JKI \approx KJI \geq JIK \approx IJK \geq IKJ \approx KIJ$$

5 Best loop permutation analysis

The best loop permutation, as can be seen from minimum execution time analysis, is IKJ or KIJ. This is because number of cache hits is highest in these loop permutations as compared to other permutations. The best permutation does not depend on the size of matrix.

6 Percentage time in each function

6.1 As reported by perf

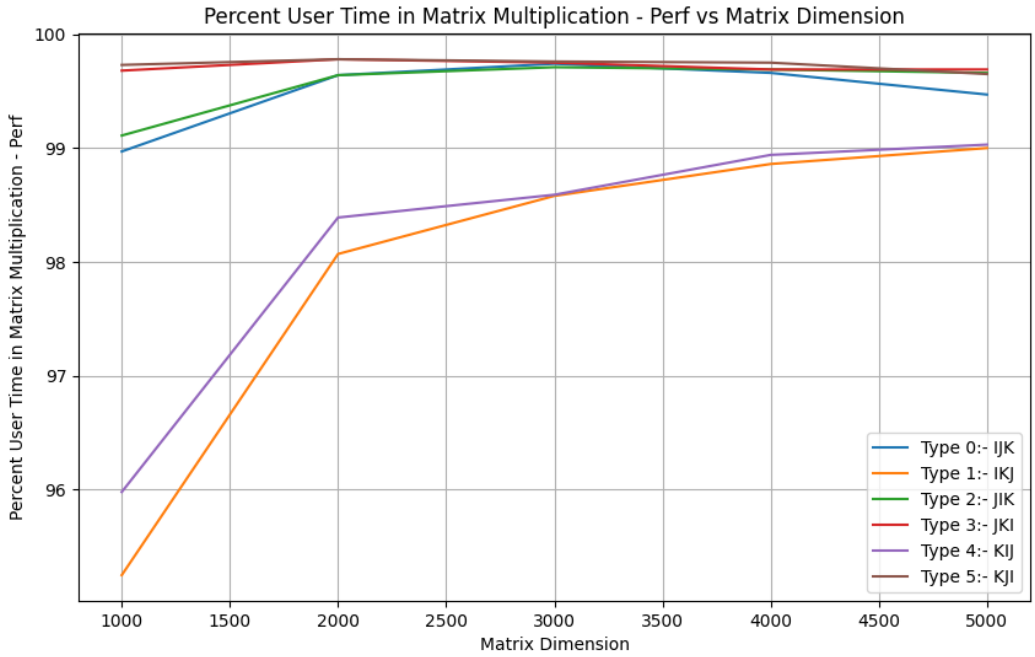

Figure 7: Percent time in matrix multiplication reported by perf

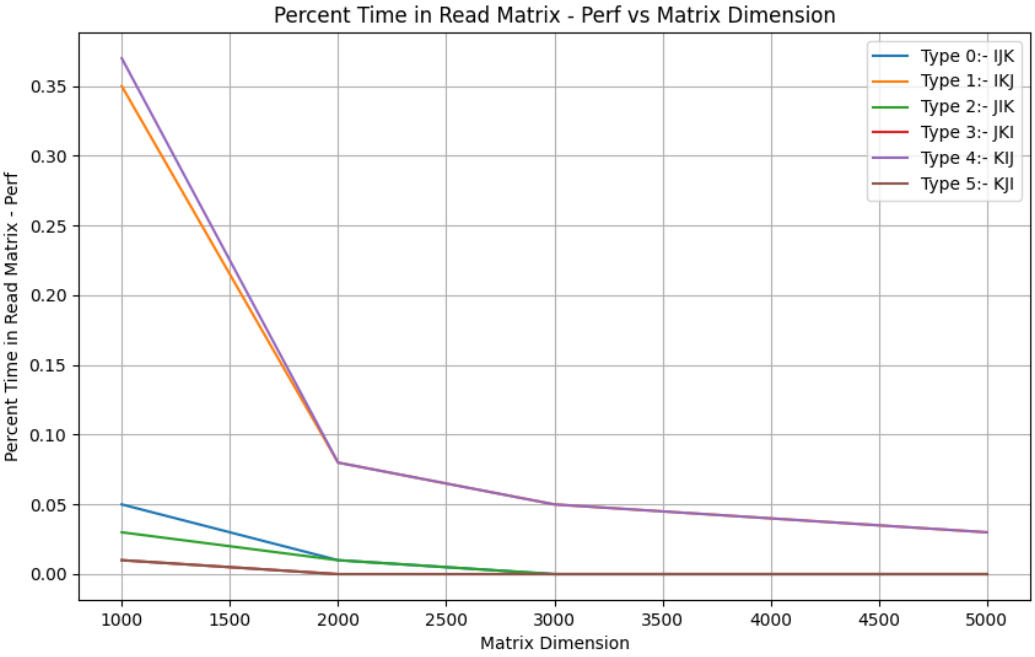

Figure 8: Percent time in read matrix reported by perf

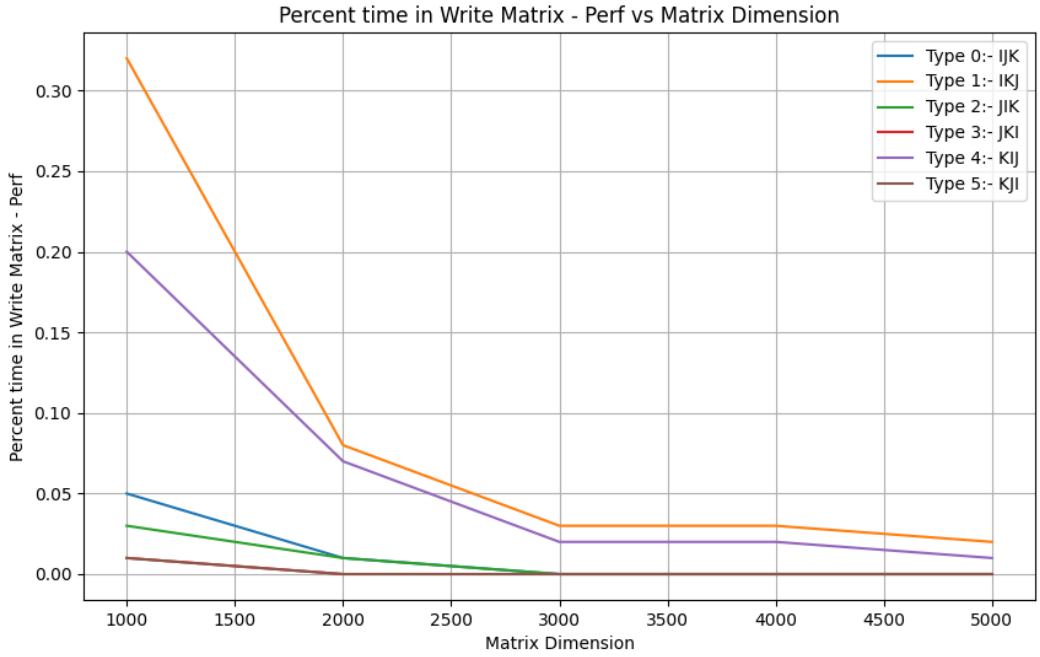

Figure 9: Percent time in write matrix reported by perf

6.2 As reported by GProf

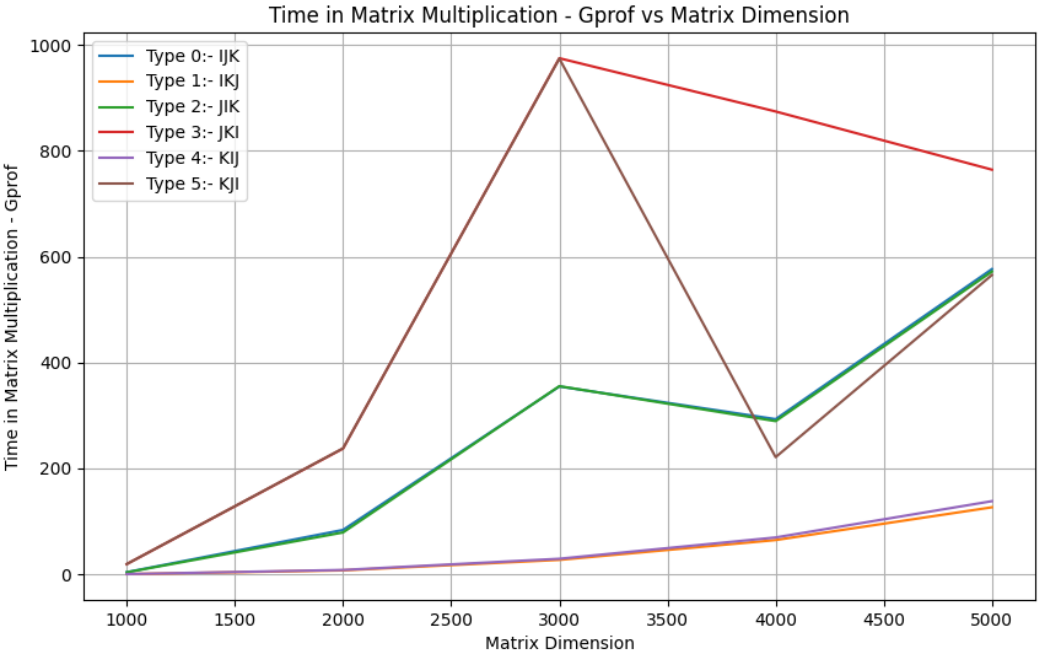

Figure 10: Percent time in matrix multiplication reported by GProf

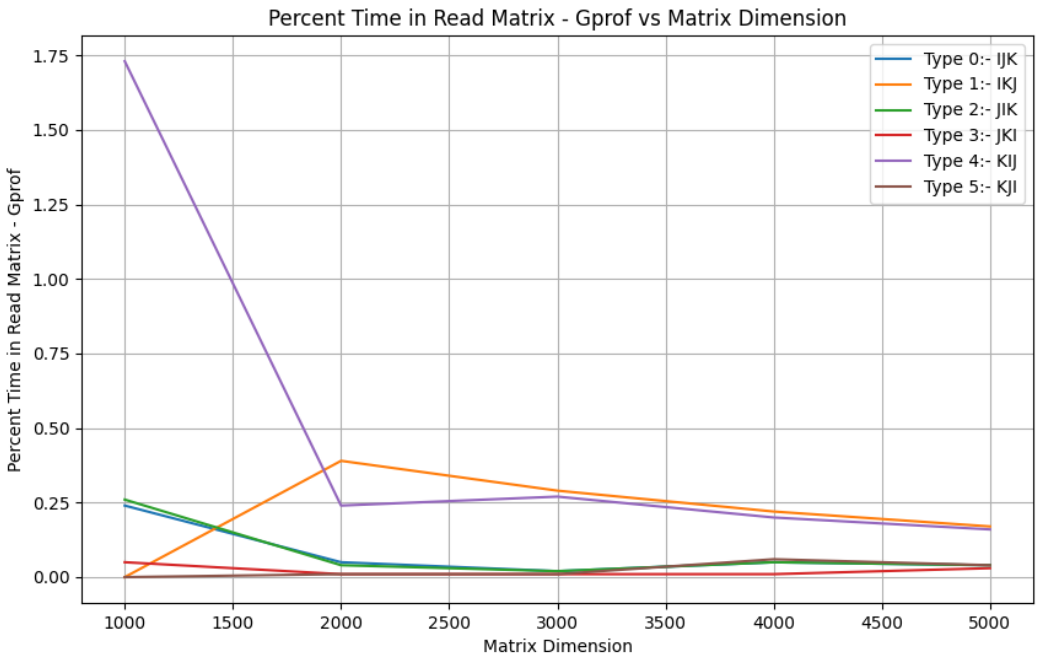

Figure 11: Percent time in read matrix reported by GProf

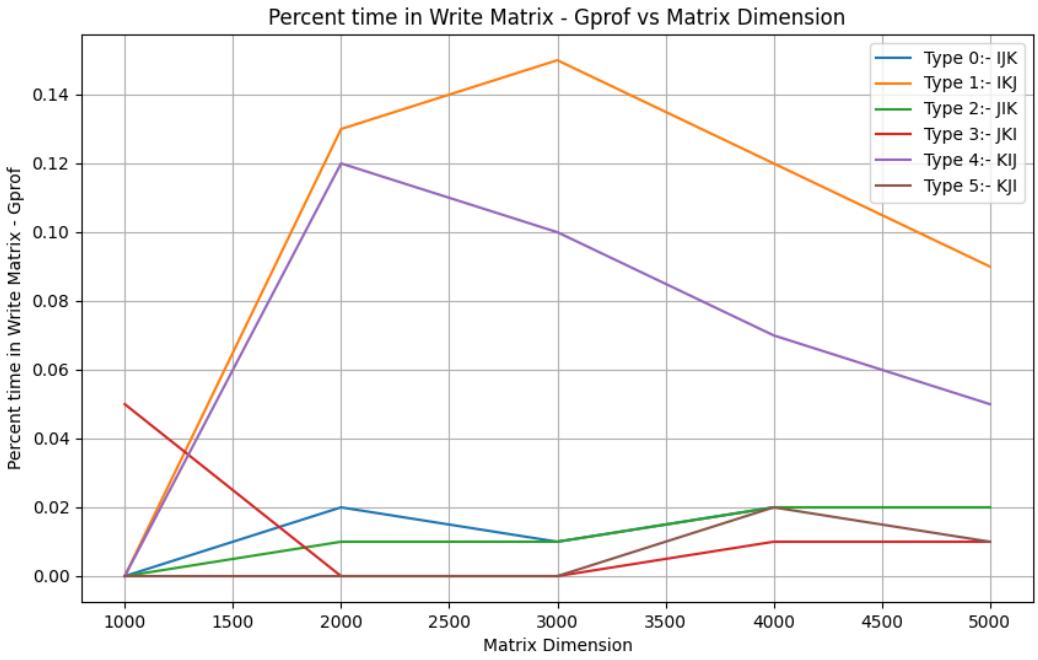

Figure 12: Percent time in write matrix reported by GProf