Language Benchmark of Matrix Multiplication

https://github.com/klapa0/big_data_studies

Kacper Klasen FB390707

October 23, 2025

1 Introduction

Matrix multiplication is a fundamental operation in many areas of computer science, including numerical simulations, graphics, and machine learning. Its computational complexity grows rapidly with the size of the matrices, making performance optimization crucial, especially for large datasets. Different programming languages and implementations can yield significantly different execution times due to variations in compiler optimizations, memory management, and underlying hardware utilization.

The purpose of this study is to benchmark the performance of matrix multiplication across three popular programming languages: C, Java, and Python.

2 Benchmarking methods and results

2.1 Java

In Java, we used the Java Microbenchmark Harness (JMH) to measure execution time accurately. The matrix multiplication function was implemented in a separate class, and a benchmark class was used to run the multiplication for matrices of varying sizes. Each size was run multiple times to account for warm-up effects.

- **Tool:** JMH (Java Microbenchmark Harness)
- **Matrix sizes:** 100, 200, 300, ..., 1000
- **Warmup runs:** 5
- **Number of runs per size:** 3
- **Metrics collected:** Average time per operation (ms/op)

Matrix Size	Average Time (ms)
100	0.367
200	2.965
300	10.136
400	23.697
500	47.742
600	82.474
700	134.271
800	196.131
900	281.047
1000	413.623

Table 1: Average Java JMH Benchmark Results for Matrix Multiplication

2.2 Python

In Python, we used the `pytest-benchmark` library to measure execution time for matrix multiplication. Each test was run multiple times to account for variability and obtain stable averages.

- **Tool:** pytest-benchmark
- **Matrix sizes:** 10, 20, 30, ..., 100
- **Metrics collected:** Mean time per operation (ms)

Matrix Size	Mean Time (ms)
10	0.4603
20	3.6245
30	11.9979
40	28.1368
50	55.0946
60	96.6164
70	153.0099
80	224.4107
90	324.2768
100	446.0290

Table 2: Average Python pytest-benchmark Results for Matrix Multiplication

2.3 C

In C was used a compiled program with flag -O3 and measured execution time using `perf stat`, which provides detailed CPU statistics including elapsed user time. Each matrix multiplication was repeated 100 times to obtain stable measurements.

- **Tool:** perf stat
- **Matrix sizes:** 100, 200, 300, ..., 1000
- **Number of repetitions per size:** 100
- **Metric collected:** User time elapsed (seconds)

Matrix Size	User Time (ms)
100	0.057
200	1.885
300	5.228
400	10.522
500	21.063
600	35.786
700	56.551
800	81.158
900	115.922
1000	170.374

Table 3: Average C perf-stat Results for Matrix Multiplication (100 repetitions)

3 Comparison of Results

The benchmark results clearly show the performance differences between C, Java, and Python for matrix multiplication. C achieved the fastest execution times due to compilation and aggressive optimizations with the -O3 flag. Java performed slower than C but benefits from Just-In-Time (JIT) compilation and automatic memory management. Python was the slowest among the three, reflecting its interpreted nature, although it is very easy to implement and test.

The execution time for all languages grows roughly cubically with the matrix size, consistent with the theoretical complexity of matrix multiplication.

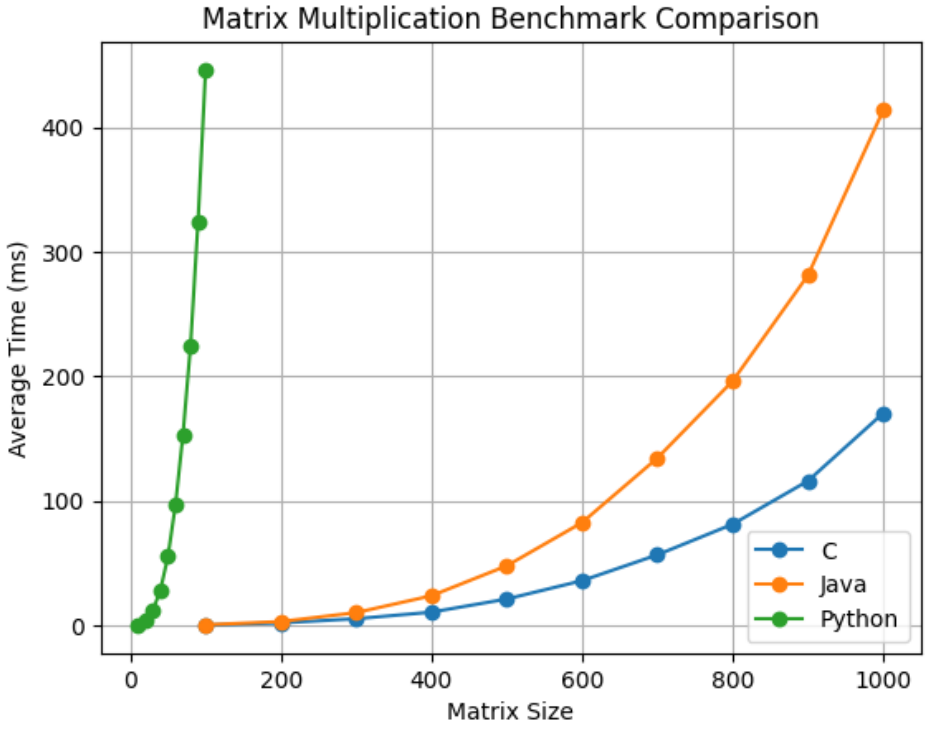

Figure 1: Comparison of Matrix Multiplication Times for C, Java, and Python

References

- Jenkov, Java Performance: JMH, 2025. Tutorial used for basic benchmarking in Maven. https://jenkov.com/tutorials/java-performance/jmh.html
- ChatGPT, Assistance for LaTeX formatting and writing the benchmark section. https://chat.openai.com/