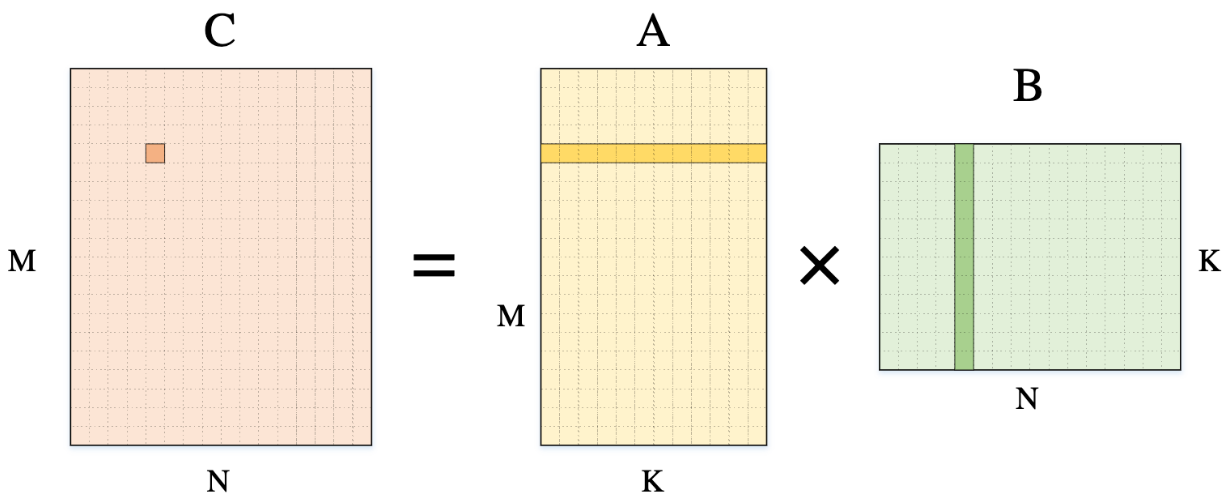
Kennesaw State University

HPC & Parallel Computing

Project – Matrix Multiplication

Instructor: Kun Suo
Points Possible: 100
Difficulty: ★★★☆☆



Mathematically, an M x N matrix is a rectangular array of elements arranged in m rows and n columns. Matrices are a common mathematical tool in advanced algebra and are also frequently used in applied mathematics disciplines such as statistical analysis. Matrix operations are an important problem in the field of numerical analysis.

General matrix multiplication (GEMM) is typically defined as:

$$C = AB$$

$$C_{m,n} = \sum_{n=1}^{N} A_{m,n} B_{n,k}$$

where A is an m*n matrix and B is an n*k matrix.

Task 1 (30 points):

Please implement matrix multiplication in C language according to the definition:

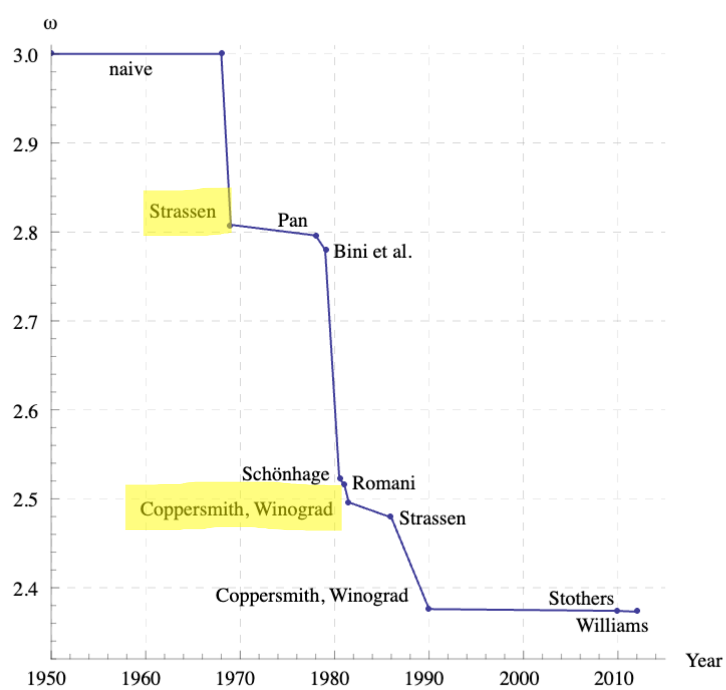
Problem: Implement general matrix multiplication in C language

- Input: Three integers M, N, and K (512 ~ 2048)
- Problem Description: Randomly generate two matrices A (M*N) and B (N*K), and perform matrix multiplication to obtain matrix C.
- Output: The time taken for the matrix calculation.

Task 2 (70 points):

In addition to the optimizations based on hardware features (e.g., cache) discussed in our slides, matrix multiplication can also be optimized through algorithmic improvements.

Algorithm analysis shows that the time complexity of the naive matrix multiplication algorithm is $O(n^3)$. Since 1969, through the efforts of many scientists, the complexity bound has been continuously reduced. Among these advancements, the Strassen algorithm and the Coppersmith–Winograd algorithm are two milestones in the optimization of matrix multiplication algorithms.



- In 1969, Volker Strassen proposed a matrix multiplication algorithm with a complexity of $O(n^{2.807})$: https://en.wikipedia.org/wiki/Strassen_algorithm. This was the first time in history that computational complexity of matrix multiplication was reduced below $O(n^3)$.

- In 1990, Don Coppersmith and Shmuel Winograd made a groundbreaking achievement by reducing the complexity to $O(n^{2.3727})$.
  Paper: https://www.sciencedirect.com/science/article/pii/S0747717108800132

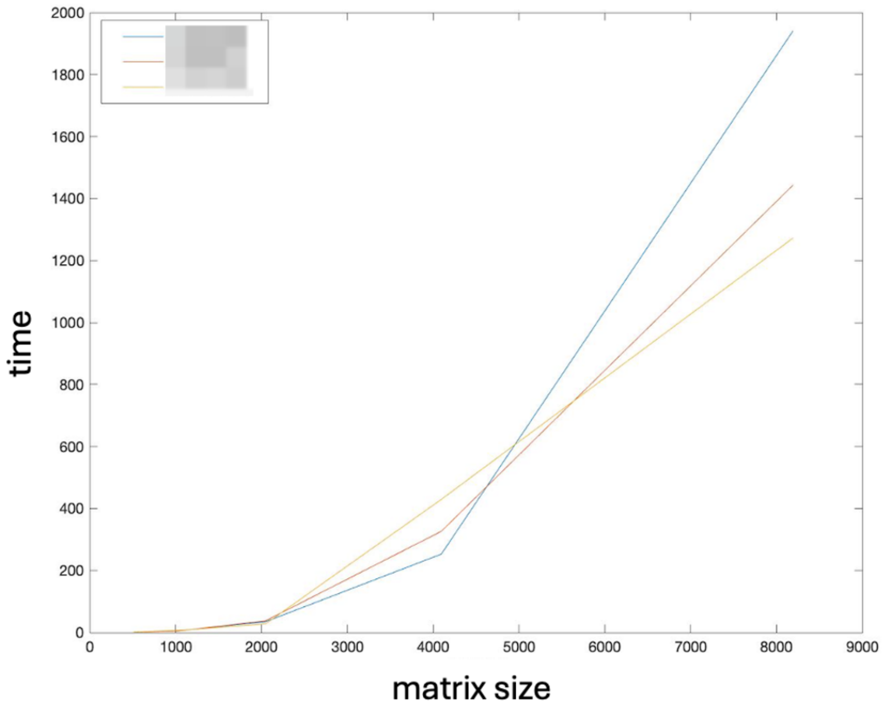
Please learn the two algorithms mentioned above independently, provide a detailed description of the optimization methods. Optimize your source code through these algorithms and make a comparison of the computation time with GEMM.

In your implementation, you must include a verification function that checks whether the output matrix C produced by each optimization is identical to the original matrix C from Task 1. (Note that if your matrix elements are floating-point numbers, elements within a certain tolerance can be considered the same. You can set a very small threshold (Epsilon), such as 10^{-9}. The equality check formula is: $|a - b| < \varepsilon$. If the difference between a and b is less than this small value, we consider them "equal".) The time of verification function should not be included in the parallel solution.

Performance comparison table:

Order of Matrix	**1024**	**2048**	**4096**	**8192**
GEMM Time				
Strassen algorithm Time				
Coppersmith–Winograd algorithm Time				

Comparison graph:



Submitting Assignment

Submit your assignment file through D2L using the appropriate link.

The submission must include the ***source code***, and ***a report describe your code logic. Output screenshot of your code*** and ***dot-line-chart & Tables*** should be included in the report.