

Strassen Matrix Multiplication: A Comprehensive Report

Bhavya Piyush

October 16, 2023

1 Introduction

Strassen Matrix Multiplication is an efficient algorithm for multiplying two matrices. Traditional matrix multiplication has a time complexity of $O(n^3)$, which can be impractical for large matrices. Strassen's algorithm reduces this complexity to approximately $O(n^{2.81})$, making it a valuable technique for numerical computations and applications like image processing and machine learning.

2 Algorithm Description

Strassen's algorithm divides matrix multiplication into smaller sub-matrix multiplications and uses recursive techniques. Here's an outline of the algorithm:

Algorithm 1 Strassen Matrix Multiplication

```
1: procedure STRASSEN( $A, B$ )
2:   if matrices  $A$  and  $B$  are  $1 \times 1$  then
3:     Compute the product directly
4:   else
5:     Divide matrices  $A$  and  $B$  into four sub-matrices
6:     Recursively compute seven products  $P_1, P_2, \dots, P_7$ 
7:     Compute the final product matrices  $C_{11}, C_{12}, C_{21}, C_{22}$  using  $P$  values
8:     Combine  $C$  matrices to get the result
```

3 C Language Code

Here's an example of Strassen's algorithm implemented in C language:

```
// Include necessary headers and define matrix multiplication functions
// ...
```

```

void strassen(int **A, int **B, int **C, int n) {
    // Implement Strassen's algorithm
    // ...
}

int main() {
    // Initialize matrices A and B
    // ...

    // Create result matrix C
    int **C = create_matrix(n);

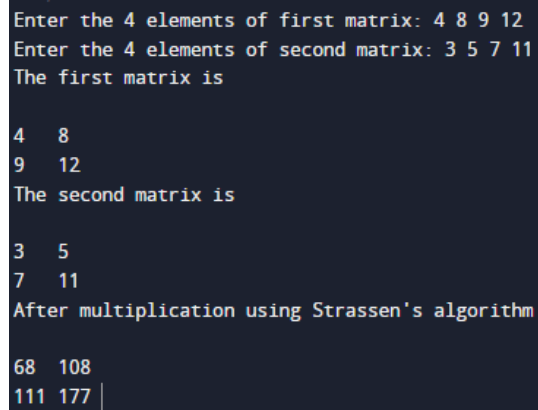
    // Perform matrix multiplication using Strassen's algorithm
    strassen(A, B, C, n);

    // Print the result matrix C
    // ...

    return 0;
}

```

4 Output



```

Enter the 4 elements of first matrix: 4 8 9 12
Enter the 4 elements of second matrix: 3 5 7 11
The first matrix is

4  8
9  12
The second matrix is

3  5
7  11
After multiplication using Strassen's algorithm

68  108
111 177 |

```

Figure 1: Output in C Language

5 Complexity Analysis

Strassen's algorithm has a time complexity of approximately $O(n^{2.81})$. While it's more efficient than the traditional $O(n^3)$ algorithm for large matrices, it has higher overhead due to recursive calls, making it less practical for small matrices. The space complexity is $O(n^2)$.

6 Flowchart

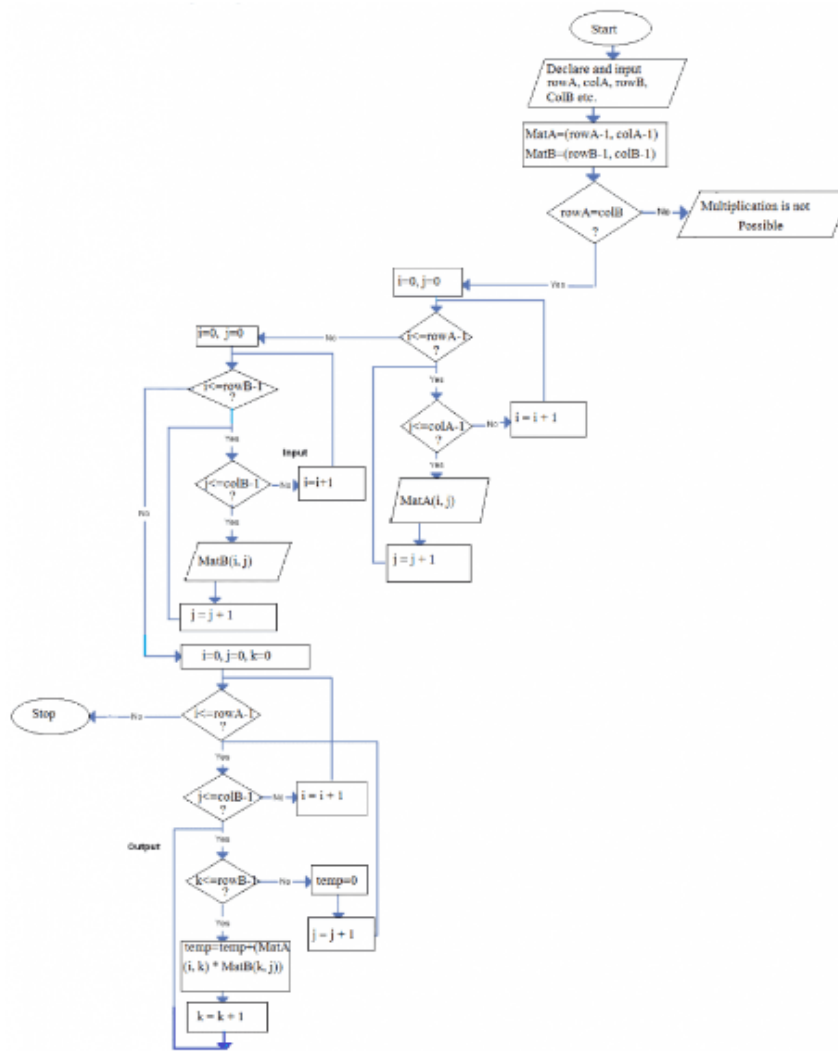


Figure 2: Flowchart of Strassen Matrix Multiplication

7 Advantages

- Improved time complexity for large matrix multiplication. - Suitable for parallelization, making it efficient for multi-core processors. - Used in numerical simulations, scientific computing, and computer graphics.

8 Disadvantages

- Higher overhead for small matrices compared to traditional methods. - Complexity constant is large, making it inefficient for small-scale problems. - Requires careful handling of edge cases in implementation.

9 Real-Life Applications

Strassen Matrix Multiplication is used in various real-life applications, including: - Image processing for filters and transformations. - Scientific simulations involving large datasets. - Machine learning algorithms for training large models. - Numerical methods in engineering for solving differential equations.

10 Conclusion

Strassen Matrix Multiplication is a powerful algorithm that significantly improves the efficiency of matrix multiplication for large matrices. While it has limitations for small matrices and requires careful implementation, its impact is felt in numerous fields, from scientific computing to machine learning.