



BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.
(Autonomous College Affiliated to University of Mumbai)
MASTER OF COMPUTER APPLICATIONS

Class : F.Y.MCA Semester : II Academic Year : 2024-25

Course Name : Design and Analysis of Algorithm MC507

Subject Incharge : Prof.Nikhita Mangaonkar

UCID: 2024510001 BATCH: A NAME: Atharva Vasant Angre

EXPERIMENT NO: 02

EXPERIMENT TITLE: To implement Matrix multiplication and its variant.

2.1 To Implement Matrix Multiplication

2.2 To Implement Strassen's Matrix Multiplication

2.3 To analyze Time and space complexity of the above with the help of the tool.

Compare the time and space complexity and state what is the difference
between the two methods

Objective:

- 1.To implement matrix multiplication methods to understand the faster techniques.
2. To compare which has a better time and space complexity.



BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.
(Autonomous College Affiliated to University of Mumbai)
MASTER OF COMPUTER APPLICATIONS

Class : F.Y.MCA Semester : II Academic Year : 2024-25

Course Name : Design and Analysis of Algorithm MC507

Subject Incharge : Prof.Nikhita Mangaonkar

**Program code: -
Matrix Multiplication**

```
matrixMultiplication.java
import java.util.*;
import java.util.Scanner;

public class matrixMultiplication {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int r1=0,c1=0;
        int r2=0,c2=0;
        int sum=0;

        System.out.println("Enter the value of row of Matrix1: ");
        r1 = sc.nextInt();
        System.out.println("Enter the value of row of Column of Matrix1: ");
        c1 = sc.nextInt();

        System.out.println("Enter the value of row of Matrix2: ");
        r2 = sc.nextInt();
        System.out.println("Enter the value of row of Column of Matrix2: ");
        c2 = sc.nextInt();

        if (c1 != r2) {
            System.out.println("Matrix multiplication is not possible. Reason at: " +
                "Column of Matrix1 does not equal the number of rows of Matrix2.");
            return;
        }

        int[][] mat1 = new int[r1][c1];
        int[][] mat2 = new int[r2][c2];
        int[][] res = new int[r1][c2];

        for (int i = 0; i < r1; i++) {
            for (int j = 0; j < c2; j++) {
                mat1[i][j] = sc.nextInt();
                mat2[i][j] = sc.nextInt();
                res[i][j] = 0;
            }
        }

        for (int i = 0; i < r1; i++) {
            for (int j = 0; j < c2; j++) {
                for (int k = 0; k < c1; k++) {
                    res[i][j] += mat1[i][k] * mat2[k][j];
                }
            }
        }

        System.out.println("Resultant Matrix: " + Arrays.deepToString(res));
    }
}
```

```
matrixMultiplication.java
public class matrixMultiplication {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int r1=0,c1=0;
        int r2=0,c2=0;
        int sum=0;

        System.out.println("Enter the value of row of Matrix1: ");
        r1 = sc.nextInt();
        System.out.println("Enter the value of row of Column of Matrix1: ");
        c1 = sc.nextInt();

        System.out.println("Enter the value of row of Matrix2: ");
        r2 = sc.nextInt();
        System.out.println("Enter the value of row of Column of Matrix2: ");
        c2 = sc.nextInt();

        if (c1 != r2) {
            System.out.println("Matrix multiplication is not possible. Reason at: " +
                "Column of Matrix1 does not equal the number of rows of Matrix2.");
            return;
        }

        int[][] mat1 = new int[r1][c1];
        int[][] mat2 = new int[r2][c2];
        int[][] res = new int[r1][c2];

        for (int i = 0; i < r1; i++) {
            for (int j = 0; j < c2; j++) {
                for (int k = 0; k < c1; k++) {
                    res[i][j] += mat1[i][k] * mat2[k][j];
                }
            }
        }

        System.out.println("Resultant Matrix: " + Arrays.deepToString(res));
    }
}
```

```
matrixMultiplication.java
public class matrixMultiplication {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int r1=0,c1=0;
        int r2=0,c2=0;
        int sum=0;

        System.out.println("Enter the value of row of Matrix1: ");
        r1 = sc.nextInt();
        System.out.println("Enter the value of row of Column of Matrix1: ");
        c1 = sc.nextInt();

        System.out.println("Enter the value of row of Matrix2: ");
        r2 = sc.nextInt();
        System.out.println("Enter the value of row of Column of Matrix2: ");
        c2 = sc.nextInt();

        if (c1 != r2) {
            System.out.println("Matrix multiplication is not possible. Reason at: " +
                "Column of Matrix1 does not equal the number of rows of Matrix2.");
            return;
        }

        int[][] mat1 = new int[r1][c1];
        int[][] mat2 = new int[r2][c2];
        int[][] res = new int[r1][c2];

        for (int i = 0; i < r1; i++) {
            for (int j = 0; j < c2; j++) {
                for (int k = 0; k < c1; k++) {
                    res[i][j] += mat1[i][k] * mat2[k][j];
                }
            }
        }

        System.out.println("Resultant Matrix: " + Arrays.deepToString(res));
    }
}
```



BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.
(Autonomous College Affiliated to University of Mumbai)
MASTER OF COMPUTER APPLICATIONS

Class : F.Y.MCA Semester : II Academic Year : 2024-25

Course Name : Design and Analysis of Algorithm MC507

Subject Incharge : Prof.Nikhita Mangaonkar

Strassen's Matrix Multiplication:

```
matrixMultiplication.java strassen_algo.java
1 import java.util.Arrays;
2 import java.util.Scanner;
3
4 public class strassen_algo {
5     // ...
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8
9         int r1=0,c1=0;
10        int r2=0,c2=0;
11        int n=0;
12
13        System.out.println("Enter the value of row of Matrix1: ");
14        r1 = sc.nextInt();
15        System.out.println("Enter the value of column of Matrix1: ");
16        c1 = sc.nextInt();
17
18        System.out.println("Enter the value of row of Matrix2: ");
19        r2 = sc.nextInt();
20        System.out.println("Enter the value of column of Matrix2: ");
21        c2 = sc.nextInt();
22
23        if (c1 != r2) {
24            System.out.println("Matrix multiplication is not possible. Reason of ? = \n" +
25                "column of Matrix1 must equal the number of rows in Matrix2.");
26            return;
27        }
28
29        int[][] mat1 = new int[r1][c1];
30        int[][] mat2 = new int[r2][c2];
31        int[][] res = new int[r1][c2];
32    }
33 }
```

```
matrixMultiplication.java strassen_algo.java
1 public class strassen_algo {
2     public static void main(String[] args) {
3         // ...
4         return;
5     }
6
7     int[][] mat1 = new int[r1][c1];
8     int[][] mat2 = new int[r2][c2];
9     int[][] res = new int[r1][c2];
10
11     for (int i = 0; i < r1; i++) {
12         for (int j = 0; j < c2; j++) {
13             System.out.println("Enter the Value for Mat1: ");
14             mat1[i][j] = sc.nextInt();
15         }
16     }
17
18     for (int i = 0; i < r2; i++) {
19         for (int j = 0; j < c2; j++) {
20             System.out.println("Enter the Value for Mat2: ");
21             mat2[i][j] = sc.nextInt();
22         }
23     }
24
25     System.out.println("Matrix 1: " + Arrays.deepToString(mat1));
26     System.out.println("Matrix 2: " + Arrays.deepToString(mat2));
27
28     res = strassen(mat1, mat2);
29
30     System.out.println("Resultant Matrix: " + Arrays.deepToString(res));
31 }
32 }
```

```
matrixMultiplication.java strassen_algo.java
1 public class strassen_algo {
2     // ...
3     @ static int[][] strassen(int[][] A, int[][] B) {
4         // ...
5         int n = A.length;
6         int[][] result = new int[n][n];
7
8         if (n == 1) {
9             result[0][0] = A[0][0] * B[0][0];
10            return result;
11        }
12
13        int newSize = n / 2;
14        int[][] A11 = new int[newSize][newSize];
15        int[][] A12 = new int[newSize][newSize];
16        int[][] A21 = new int[newSize][newSize];
17        int[][] A22 = new int[newSize][newSize];
18        int[][] B11 = new int[newSize][newSize];
19        int[][] B12 = new int[newSize][newSize];
20        int[][] B21 = new int[newSize][newSize];
21        int[][] B22 = new int[newSize][newSize];
22
23        for (int i = 0; i < newSize; i++) {
24            for (int j = 0; j < newSize; j++) {
25                A11[i][j] = A[i][j];
26                A12[i][j] = A[i][j + newSize];
27                A21[i][j] = A[i + newSize][j];
28                A22[i][j] = A[i + newSize][j + newSize];
29                B11[i][j] = B[i][j];
30                B12[i][j] = B[i][j + newSize];
31                B21[i][j] = B[i + newSize][j];
32                B22[i][j] = B[i + newSize][j + newSize];
33            }
34        }
35    }
36 }
```

```
matrixMultiplication.java strassen_algo.java
1 public class strassen_algo {
2     // ...
3     static int[][] strassen(int[][] A, int[][] B) {
4         // ...
5         B21[i][j] = B[i + newSize][j];
6         B22[i][j] = B[i + newSize][j + newSize];
7     }
8
9     int[][] P1 = strassen(add(A11, A22), add(B11, B22));
10    int[][] P2 = strassen(add(A12, A22), B11);
11    int[][] P3 = strassen(A11, subtract(B12, B22));
12    int[][] P4 = strassen(A12, subtract(B21, B11));
13    int[][] P5 = strassen(add(A11, A12), B22);
14    int[][] P6 = strassen(subtract(A21, A11), add(B11, B12));
15    int[][] P7 = strassen(subtract(A12, A22), add(B21, B22));
16
17    int[][] C11 = add(subtract(add(P1, P4), P5), P7);
18    int[][] C12 = add(P3, P5);
19    int[][] C21 = add(P2, P6);
20    int[][] C22 = add(subtract(add(P1, P3), P2), P6);
21
22    for (int i = 0; i < newSize; i++) {
23        for (int j = 0; j < newSize; j++) {
24            result[i][j] = C11[i][j];
25            result[i][j + newSize] = C12[i][j];
26            result[i + newSize][j] = C21[i][j];
27            result[i + newSize][j + newSize] = C22[i][j];
28        }
29    }
30
31    return result;
32 }
33 }
```



BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.
(Autonomous College Affiliated to University of Mumbai)
MASTER OF COMPUTER APPLICATIONS

Class : F.Y.MCA Semester : II Academic Year : 2024-25

Course Name : Design and Analysis of Algorithm MC507

Subject Incharge : Prof.Nikhita Mangaonkar

```
1 public class Streamer_Alg {
2     static int[][] multiply(int[][] A, int[][] B) {
3         // ...
4     }
5
6     static int[][] multiplyRec(int[][] A, int[][] B) {
7         // ...
8     }
9
10    static int[][] submatrix(int[][] A, int[][] B) {
11        // ...
12    }
13 }
```




BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.
(Autonomous College Affiliated to University of Mumbai)
MASTER OF COMPUTER APPLICATIONS

Class : F.Y.MCA Semester : II Academic Year : 2024-25

Course Name : Design and Analysis of Algorithm MC507

Subject Incharge : Prof.Nikhita Mangaonkar

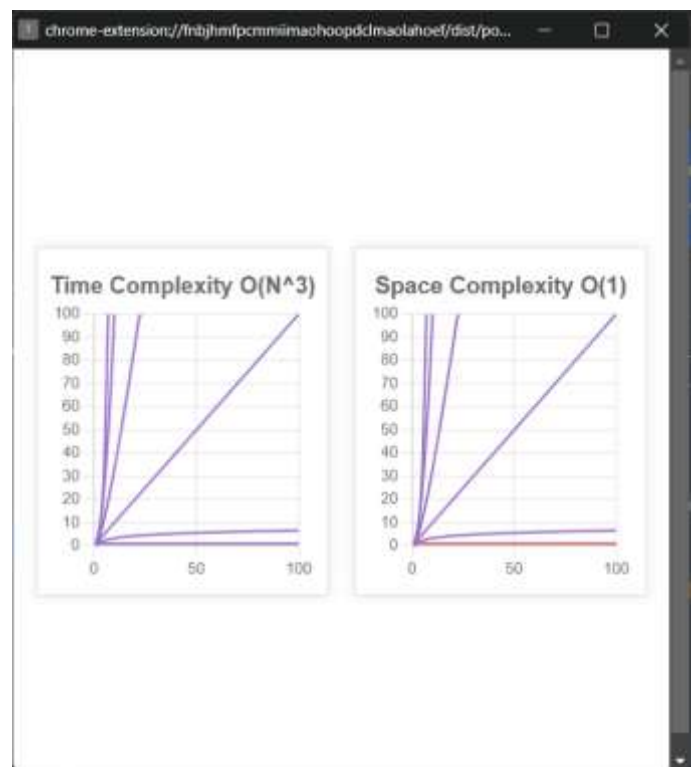
Output:

Matrix Multiplication

```
matrixMultiplication.java
Run matrixMultiplication

"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:
Enter the Value of num of Rows in Matrix1: 2
Enter the Value of num of Columns in Matrix1: 2
Enter the Value of num of Rows in Matrix2: 2
Enter the Value of num of Columns in Matrix2: 2
Enter the Value for Mat1: 1
Enter the Value for Mat1: 2
Enter the Value for Mat1: 3
Enter the Value for Mat1: 4
Enter the Value for Mat2: 5
Enter the Value for Mat2: 6
Enter the Value for Mat2: 7
Enter the Value for Mat2: 8
Matrix 1: [[1, 2], [3, 4]]
Matrix 2: [[5, 6], [7, 8]]
Resultant Matrix: [[19, 22], [43, 56]]

Process finished with exit code 0
```

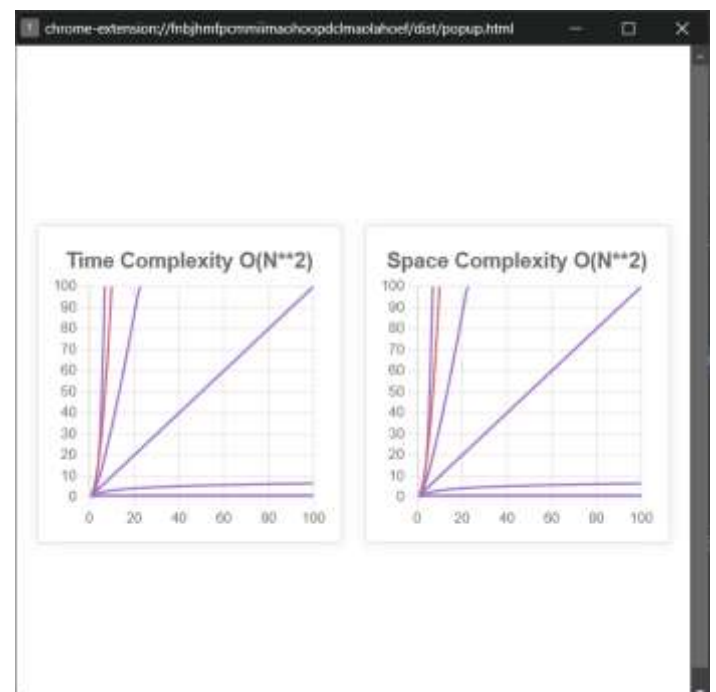


Strassen's Matrix Multiplication

```
Project
Run strassen_algo

"C:\Program Files\Java\jdk-23\bin\java.exe" "-javaagent:C:\P
Enter the Value of num of Rows in Matrix1: 2
Enter the Value of num of Columns in Matrix1: 2
Enter the Value of num of Rows in Matrix2: 2
Enter the Value of num of Columns in Matrix2: 2
Enter the Value for Mat1: 1
Enter the Value for Mat1: 2
Enter the Value for Mat1: 3
Enter the Value for Mat1: 4
Enter the Value for Mat2: 5
Enter the Value for Mat2: 6
Enter the Value for Mat2: 7
Enter the Value for Mat2: 8
Matrix 1: [[1, 2], [3, 4]]
Matrix 2: [[5, 6], [7, 8]]
Resultant Matrix: [[19, 22], [43, 56]]

Process finished with exit code 0
```





BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058.
(Autonomous College Affiliated to University of Mumbai)
MASTER OF COMPUTER APPLICATIONS

Class : F.Y.MCA Semester : II Academic Year : 2024-25

Course Name : Design and Analysis of Algorithm MC507

Subject Incharge : Prof.Nikhita Mangaonkar

Result:

- Best Case: Strassen's algorithm performs better for large matrices due to fewer multiplication operations, whereas the naive method remains at $O(n^3)$.
- Worst Case: Even in the worst-case scenario, Strassen's method outperforms the naive approach in terms of time complexity. However, it requires additional space.
- Average Case: Strassen's method is generally more efficient for large matrices, but for small matrices, the naive approach may be preferable due to lower overhead.

Conclusion:

- Strassen's algorithm provides an optimized approach for matrix multiplication compared to the naive method, especially for large matrices.
- However, due to recursion and additional storage requirements, it may not be suitable for very small matrices.
- Applications: This technique is widely used in image processing, machine learning, graphics transformations, scientific computing, and large-scale simulations.