SARDAR PATEL INSTITUTE OF TECHNOLOGY

Name: Sujal Chordia

2021700015

CSE DS D1

Exp. 3: Strassen's Multiplication

AIM: Multiplying two matrices using Strassen's Multiplication Method (Divide and Conquer Approach)

Theory:

Let us consider two matrices X and Y. We want to calculate the resultant matrix Z by multiplying X and Y.

General Method

First, we will discuss naïve method and its complexity. Here, we are calculating Z = X × Y. Using Naïve method, two matrices (X and Y) can be multiplied if the order of these matrices are p × q and q × r. Following is the algorithm.

Algorithm: Matrix-Multiplication (X, Y, Z)

```
for i = 1 to p do
 for j = 1 to r do
 Z[i,j] := 0
 for k = 1 to q do
 Z[i,j] := Z[i,j] + X[i,k] × Y[k,j]
```

Complexity

Here, we assume that integer operations take O(1) time. There are three for loops in this algorithm and one is nested in other. Hence, the algorithm takes $O(n^3)$ time to execute.

Strassen's Matrix Multiplication Algorithm

In this context, using Strassen's Matrix multiplication algorithm, the time consumption can be improved a little bit.

Strassen's Matrix multiplication can be performed only on square matrices where n is a power of 2. Order of both of the matrices are $n \times n$.

Divide X, Y and Z into four $(n/2)\times(n/2)$ matrices as represented below −

Z=[IKJL]

X=[ACBD] and Y=[EGFH]

Using Strassen's Algorithm compute the following −

$M1=(A+C) \times (E+F)$

$M2=(B+D) \times (G+H)$

$M3=(A-D) \times (E+H)$

$M4 = A \times (F-H)$

$M5 = (C+D) \times (E)$

$M6 = (A+B) \times (H)$

$M7 = D \times (G-E)$

Then,

$I = M2+M3-M6-M7$

$J = M4+M6$

$K = M5+M7$

$L = M1-M3-M4-M5$

Analysis

$T(n)=\{c7xT(n^2)+dxn2$ if n=1 otherwise

where c and d are constants

Using this recurrence relation, we get T(n)=O(nlog7)

Hence, the complexity of Strassen's matrix multiplication algorithm is

O(nlog7)

PROGRAM:

```
#include<stdio.h>
int
main ()
{
int a[2][2], b[2][2], c[2][2], i, j;
int m1, m2, m3, m4, m5, m6, m7;

printf ("\nEnter the first matrix\n");

for (i = 0; i < 2; i++)
  {
  for (j = 0; j < 2; j++)
     {
    printf ("Enter the element %d%d: ", i+1, j+1);
    scanf ("%d", &a[i][j]);
    }
  }
printf ("\nEnter the second matrix\n");
for (i = 0; i < 2; i++)
  {
    for (j = 0; j < 2; j++)
       {
    printf ("Enter the element %d%d: ", i+1, j+1);
    scanf ("%d", &b[i][j]);
    }
  }
```

```
printf ("\nThe first matrix is\n\n");
for (i = 0; i < 2; i++)
    {
    printf ("\t");
    for (j = 0; j < 2; j++)
        {
        printf ("%d\t", a[i][j]);
        }
    printf ("\n");
}

printf ("\nThe second matrix is\n\n");
for (i = 0; i < 2; i++)
    {
    printf ("\t");
    for (j = 0; j < 2; j++)
        {
        printf ("%d\t", b[i][j]);
        }
    printf ("\n");
    }
m1 = (a[0][0] + a[1][1]) * (b[0][0] + b[1][1]);
m2 = (a[1][0] + a[1][1]) * b[0][0];
m3 = a[0][0] * (b[0][1] - b[1][1]);
m4 = a[1][1] * (b[1][0] - b[0][0]);
m5 = (a[0][0] + a[0][1]) * b[1][1];
m6 = (a[1][0] - a[0][0]) * (b[0][0] + b[0][1]);
```

```
m7 = (a[0][1] - a[1][1]) * (b[1][0] + b[1][1]);

c[0][0] = m1 + m4 - m5 + m7;
c[0][1] = m3 + m5;
c[1][0] = m2 + m4;
c[1][1] = m1 - m2 + m3 + m6;

printf ("\nAfter multiplication using Strassen's algorithm \n\n");

for (i = 0; i < 2; i++)
   {
   printf ("\t");
   for (j = 0; j < 2; j++)
       {
      printf ("%d\t", c[i][j]);
      }
   printf ("\n");
   }
return 0;
}
```

RESULT:

```
Enter the first matrix
Enter the element 11: 3
Enter the element 12: 4
Enter the element 21: 5
Enter the element 22: 6

Enter the second matrix
Enter the element 11: 3
Enter the element 12: 4
Enter the element 21: 5
Enter the element 22: 6

The first matrix is

        3       4
        5       6

The second matrix is

        3       4
        5       6

After multiplication using Strassen's algorithm

        29      36
        45      56
```

OBSERVATIONS:

Strassen's Multiplication is better than the Normal method of matrix multiplication as it's time complexity comes to $n^{2.81}$ compared to the general method which has time complexity of n^3.

We use only 7 multiplications in Strassen's Multiplication whereas we use 8 multiplications in the general method, so the time complexity is better.

While usual multiplication requires 8 multiplications and 4 additions Strassen's Multiplication requires 7 multiplications and 18 additions.

CONCLUSION:

I implemented Strassen's Matrix Multiplication method in this experiment and solved user-given matrices to find their product.