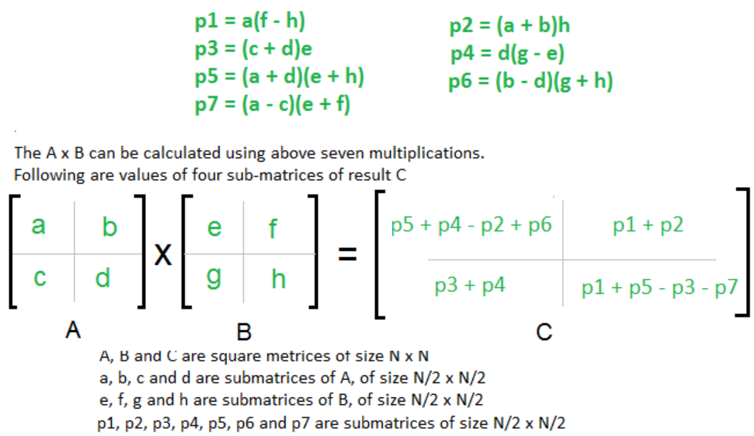
Name: Yash Ravindra Burad

Roll no.: 2022301004

Subject: Design and analysis of algorithm

Practical: Experiment on Strassens matrix multiplication

Theory:



Procedures:

Divide a matrix of order of 2*2 recursively till we get the matrix of 2*2.

Use the previous set of formulas to carry out 2*2 matrix multiplication.

In this eight multiplication and four additions, subtraction are performed.

Combine the result of two matrixes to find the final product or final matrix.

Algorithm:

1. Take inputs as matrix1 and matrix2

2. elements of matrix 1: a11, a12, a13, a14

   elements of matrix 2: b11, b12, b13, b14

3. perform following addition:

D1 = (a11 + a22) * (b11 + b22)

D2 = (a21 + a22)*b11

D3 = (b12 – b22)*a11

D4 = (b21 – b11)*a22

D5 = (a11 + a12)*b22

D6 = (a21 – a11) * (b11 + b12)

D7 = (a12 – a22) * (b21 + b22)

C00= d1 + d4 – d5 + d7

C01 = d3 + d5

C10 = d2 + d4

C11 = d1 + d3 – d2 – d6

4. print the reasultant matrix

Program:

```
#include<stdio.h>
int main()
{
    int i,j,array1[2][2],array2[2][2];
    printf ("first matrix\n");
    for (i=0;i<2;i++)
    {
        for (j=0;j<2;j++)
        {
           scanf ("%d",&array1[i][j]);
        }

    }

     for (i=0;i<2;i++)
    {
        for (j=0;j<2;j++)
        {
           printf  ("%d \t",array1[i][j]);
        }
    printf ("\n");
    }

    printf ("second matrix\n");

    for (i=0;i<2;i++)
    {
```

```
        for (j=0;j<2;j++)
        {
          scanf ("%d",&array2[i][j]);
        }

    }

     for (i=0;i<2;i++)
    {
        for (j=0;j<2;j++)
        {
          printf  ("%d \t",array2[i][j]);
        }
    printf ("\n");
    }

/*printf ("%d\n",array1[0][0]);
printf ("%d\n",array1[0][1]);
printf ("%d\n",array1[1][0]);
printf ("%d\n",array1[1][1]);

printf ("%d\n",array2[0][0]);
printf ("%d\n",array2[0][1]);
printf ("%d\n",array2[1][0]);
printf ("%d\n",array2[1][1]);*/

int s1[1],s2[1],s3[1],s4[1],s5[1],s6[1],s7[1],s8[1],s9[1],s10[1];
s1[0] = array2[0][1] - array2[1][1];
s2[0] = array1[0][0] + array1[0][1];
s3[0] = array1[1][0] + array1[1][1];
s4[0] = array2[1][0] - array2[0][0];
s5[0] = array1[0][0] + array1[1][1];
s6[0] = array2[0][0] + array2[1][1];
s7[0] = array1[0][1] - array1[1][1];
s8[0] = array2[1][0] + array2[1][1];
s9[0] = array1[0][0] -array1[1][0];
s10[0] = array2[0][0] + array2[0][1];

int p1[1], p2[1], p3[1], p4[1], p5[1], p6[1], p7[1];
p1[0] = array1[0][0] * s1[0];
p2[0] = s2[0] * array2[1][1];
p3[0] = s3[0] * array2[0][0];
p4[0] = array1[1][1] * s4[0];
```

```
p5[0] = s5[0] * s6[0];
p6[0] = s7[0] * s8[0];
p7[0] = s9[0] * s10[0];

int c11[1], c12[1], c21[1], c22[1];
printf ("\nMultiplication of two metrices\n");
c11[0] = p5[0] + p4[0] - p2[0] + p6[0];
c12[0] = p1[0] + p2[0];
c21[0] = p3[0] + p4[0];
c22[0] = p5[0] + p1[0] - p3[0] - p7[0];
printf ("%d\t",c11[0]);
printf ("%d\n",c12[0]);
printf ("%d\t",c21[0]);
printf ("%d\n",c22[0]);
    return 0;
}
```

Output:

```
first matrix
10 20
30 40
10      20
30      40
second matrix
20 40
60 80
20      40
60      80

Multiplication of two metrices
1400    2000
3000    4400


...Program finished with exit code 0
Press ENTER to exit console.
```

Observation:

Srassens matrix multiplication reduces the number of multiplication as it takes more time complexity than the addition and subtraction. So, operations are replaced by addition. For the big metrics, this method is very useful and fast.

Time complexity of Srassen's Matrix multiplication:

T(n) = 7T(n/2) + O(n^2) = O(n^log(7)).

Conclusion:

In this practical, I performed the Strassen's Matrix multiplication where I multiplied 2X2 matrix using above mentioned method. In this method, number of multiplication operations get reduced and reduces the time taken by the algorithm, number of operations also reduces.