Time Complexity Analysis :

Here, we get recurrence relation as:

$$\therefore T(n) = \begin{cases} 1 & n \leq 2 \\ 7T(n/2) + n^2 & n > 2 \end{cases}$$

Here,

$\therefore T(n) = 7T(n/2) + n^2$

Using master method,

$a = 7$, $b = 2$ and $f(n) = n^2$

Then,

$$\therefore h(n) = \frac{f(n)}{n^{\log_b a}} = \frac{n^2}{n^{\log_2 7}} = \frac{n^2}{n^{2.8}} = n^{-0.8}$$

Comparing with n^r,

$r = -0.8 < 0$, so $U(n) := O(1)$

Now,

$$\therefore T(n) = n^{\log_b a} . U(n)$$
$$= n^{\log_2 8} U(n)$$
$$= n^{2.81} . 1$$
$$= n^{2.81}$$

$\Rightarrow O(n^{2.81})$

In summary, the time complexity of Strassen's algorithm can vary between $O(n^3)$ in the best case scenario, $O(n^{2.81})$ in the worst case scenario, and also $O(n^{2.81})$ in the average case scenario, although the latter may vary depending on the input data distribution and characteristics.

Experiment : 5 Date : 16/01/2024

Title : Strassen Matrix Multiplication

Aim : To implement and analyze strassen matrix multiplication algorithm.

Algorithm :

Step 1 : Start

Step 2 : Define a function 'strassen' to multiply two matrices recursively using Strassen's algorithm.

Step 3 : Divide the matrices A and B into submatrices.

Step 4 : Compute the 7 intermediate matrices (M1 to M7) recursively using Strassen's algorithm.

Step 5 : Combine the intermediate matrices to compute the resulting matrix C.

Step 6 : Implement a main function to input matrix sizes and elements, perform matrix multiplication using 'strassen', and print the result.

Step 7 : Stop

Program - Implementation :

```
# include < stdio.h >
# include < conio.h >
void strassen (int n, int A[][n], int B[][n], int C[][n]){
    if (n == 1){
        C[0][0] = A[0][0] * B[0][0];
        return;
    }
    int i, j;
    int A11[n/2][n/2], A12[n/2][n/2], A21[n/2][n/2],
        A22[n/2][n/2];
    int B11[n/2][n/2], B12[n/2][n/2], B21[n/2][n/2],
        B22[n/2][n/2];
    int C11[n/2][n/2], C12[n/2][n/2], C21[n/2][n/2],
        C22[n/2][n/2];
```

Dry run with sample input and output :

Sample Input :

Enter the size of the matrices : 2

Enter elements of matrix A:

1 2

3 4

Enter elements of matrix B :

5 6

7 8

Sample Output :

Resultant matrix C :

19 22

43 50

```
int M1 [n/2] [n/2], M2 [n/2] [n/2], M3 [n/2] [n/2],
  M4 [n/2] [n/2], M5 [n/2] [n/2], M6 [n/2] [n/2],
  M7 [n/2] [n/2];
int temp1 [n/2][n/2], temp2 [n/2] [n/2];
for (i=0 ; i<n/2 ; i++) {
    for (j=0 ; j<n/2 ; j++){
        A11[i][j] = A[i][j];
        A12[i][j] = A[i] [j+n/2];
        A21 [i] [j] = A [i+n/2] [j];
        A22 [i] [j] = A [i+n/2][j+n/2];
        B11 [i][j] = B[i][j];
        B12 [i] [j] = B [i] [j+n/2];
        B21 [i] [j] = B [i+n/2] [j];
        B22 [i] [j] = B [i+n/2] [j+n/2];
    }
}
for (i=0; i<n/2 ; i++){
    for (j=0 ; j < n/2; j++) {
        temp1[i][j] = A11[i][j] + A22 [i][j];
        temp2 [i][j] = B11 [i][j] + B22[i][j];
    }
}
strassen (n/2, temp1, temp2, M1);
for (i=0; i<n/2 ; i++) {
    for (j=0; j<n/2 ; j++){
        temp1 [i][j] = A21 [i][j] + A22[i][j];
    }
}
strassen (n/2, temp1, B11, M2);

for (i=0; i < n/2 ; i++){
    for (j=0 ; j<n/2; j++) {
        temp1 [i][j] = B12 [i] [j]- B22 [i][j];
    }
}
strassen (n/2, A11, temp1, M3);
```

```
for (i=0; i<n/2 ; i++){
    for(j=0 ; j<n/2 ; j++){
        temp 1[i][j] = B21[i][j] - B11[i][j];
    }
}
strassen (n/2, A22, temp1, M4);
for (i=0; i<n/2 ; i++){
    for(j=0; j<n/2; j++){
        temp1[i][j] = A11[i][j] + A12 [i][j];
} }
strassen (n/2, temp 1, B22, M5);
for (i=0; i <n/2 ; i++){
    for(j=0 ; j<n/2; j++){
        temp1 [i][j] = A21 [i][j] - A11 [i][j];
        temp 2 [i][j] = B 11 [i] [j] + B12[i][j];
    }
}
strassen (n/2, temp1, temp2, M6);
for (i=0; i<n/2; i++){
    for (j=0; j<n/2 ; j++){
        temp1 [i][j] = A12[i][j] - A22 [i] [j] ;
        temp2 [i][j] = B21[i][j] + B22[i][j];
} }
strassen (n/2, temp 1, temp 2, M7);
for (i=0; i< n/2 ; i++){
    for (j=0; j< n/2 ; j++){
    C[i][j] = M1[i][j] + M4[i][j] - M5[i][j] +
              M7[i][j];
    }
}
for (i=0; i<n/2 ; i++){
    for(j=0; j<n/2 ; j++){
    C[i][j+n/2] = M3 [i][j] + M5 [i] [j] ;
    }
}
```

```c
    for (i=0; i<n/2; i++){
        for (j=0; j<n/2; j++){
            C[i+n/2][j] = M2[i][j] + M4[i][j];
        }
    }
    for (i=0; i<n/2; i++){
        for (j=0; j<n/2; j++){
            C[i+n/2][j+n/2] = M1[i][j] - M2[i][j] +
                              M3[i][j] + M6 [i][j];
        }
    }
}
void printMatrix (int n, int M[][100]){
    for (int i=0; i<n; i++){
        for (int j=0; j<n; j++){
            printf("%d ", M[i][j]);
        }
        printf("\n");
    }
}
int main(){
    int i, j, n;   int A[100][100], B[100][100], C[100][100];
    clrscr();
    printf("Enter the size of the matrices : ");
    scanf("%d", &n);
    printf("Enter elements of matrix A: \n");
    for (i=0; i<n; i++){
        for (j=0; j<n; j++){
            scanf("%d", &A[i][j]);
        }
    }
    printf("Enter elements of matrix B: \n");
    for (i=0; i<n; i++){
        for (j=0; j<n; j++){
            scanf("%d", &B[i][j]);
        }
    }
```

```
strassen (n, A, B, c);
printf ("Resultant matrix C :\n");
printMatrix (n, C);
getch ();
return 0;
}
```

~~Output :~~

Result :

Strassen matrix multiplication algorithm was implemented and analyzed successfully.