DAA - Strassen's Matrix Multiplication

In this chapter, first we will discuss the general method of matrix multiplication and later we will discuss Strassen's matrix multiplication algorithm.

Problem Statement

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

Naïve 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 × n**.

Divide **X**, **Y** and **Z** into four (n/2)×(n/2) matrices as represented below −

$Z = \begin{bmatrix} I & J \\ K & L \end{bmatrix}$ $X = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$ and $Y = \begin{bmatrix} E & F \\ G & H \end{bmatrix}$

Using Strassen's Algorithm compute the following −

$$M_1 := (A + C) \times (E + F)$$

$$M_2 := (B + D) \times (G + H)$$

$$M_3 := (A - D) \times (E + H)$$

$$M_4 := A \times (F - H)$$

$$M_5 := (C + D) \times (E)$$

$$M_6 := (A + B) \times (H)$$

$$M_7 := D \times (G - E)$$

Then,

$$I := M_2 + M_3 - M_6 - M_7$$

$$J := M_4 + M_6$$

$$K := M_5 + M_7$$

$$L := M_1 - M_3 - M_4 - M_5$$

Analysis

$T(n) = \begin{cases} c & if\, n = 1 \\ 7\, x\, T(\frac{n}{2}) + d\, x\, n^2 & otherwise \end{cases}$ where *c* and *d* are constants

Using this recurrence relation, we get $T(n) = O(n^{log7})$

Hence, the complexity of Strassen's matrix multiplication algorithm is $O(n^{log7})$.