Algorithms: Design and Analysis - CS 412

Weekly Challenge 04

Ali Muhammad Asad - aa07190

1. (1 point) Applying the standard mathematical method to define an algorithm for matrix multiplication gives an $O(n^3)$ solution. Given two $n \times n$ matrices, a brute force solution requires n^3 arithmetic operations. Just like fast integer multiplication, a solution exists to reduce the time complexity of matrix multiplication. For example, Strassen's algorithm completes matrix multiplication in $O(n^2.81)$. Start by identifying a fast matrix multiplication algorithm with complexity better than $O(n^3)$. Use the algorithm to compute the matrix product:

$$\begin{pmatrix} 1 & 3 \\ 6 & 5 \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

Clearly list all the steps of the solution.

Next, compare the dimension of matrix where your algorithm outperforms the brute force algorithm.

Solution: We will use the **Strassen's Algorithm** for fast matrix multiplication. The brute force method requires 8 multiplications and 4 additions. The result for two arbitrary 2×2 matrices, A and B is:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{pmatrix}$$

where the first matrix is A, and the second matrix is B, and the result is AB.

Strassen's Algorithm is a divide and conquer algorithm, where it divides the matrix into submatrices of equal size and then makes computations on those matrices to find the resultant matrix. For two arbitrary 2×2 matrices it works as follows:

1. Compute seven products as such:
    - $p_1 = a(f - h)$
    - $p_2 = (a + b)h$
    - $p_3 = (c + d)e$
    - $p_4 = d(g - e)$
    - $p_5 = (a + d)(e - h)$
    - $p_6 = (b - d)(g + h)$

- $p_7 = (a - c)(e + f)$

2. We compute the results as:
   - $AB_{11} = p_5 + p_4 - p_2 + p_6 = ae + ah + de + dh + dg - de - ah - bh + bg + bh - dg - dh = ae + bg$
   - $AB_{12} = p_1 + p_2 = af - ah + ah + bh = af + bh$
   - $AB_{21} = p_3 + p_4 = ce + de + dg - de = ce + dg$
   - $AB_{22} = p_1 + p_5 - p_3 - p_7 = af - ah + ae + ah + de + dh - ce - de - ae - af + ce + cf = cf + dh$

The above simplification by expansion has only been done to verify that the method produces the same results as the brute force approach. With values, there will only be 7 multiplication operations and 8 addition operations. Since multiplication is a costly operation, while addition is not, so for large values of n, Strassen's Algorithm performs better than the brute force approach.

The above example using Strassen's Algorithm can be done as follows:

1. Compute the seven products:
   - $p_1 = 1(b - d) = b - d$
   - $p_2 = (1 + 3)d = 4d$
   - $p_3 = (6 + 5)a = 11a$
   - $p_4 = 5(c - a) = 5c - 5a$
   - $p_5 = (1 + 5)(a + d) = 6a + 6d$
   - $p_6 = (3 - 5)(c + d) = -2c - 2d$
   - $p_7 = (1 - 6)(a + b) = -5a - 5b$

   Then we can compute the results as:
   - $AB_{11} = p_5 + p_4 - p_2 + p_6 = 6a + 6d + 5c - 5a - 4d - 2c - 2d = a + 3c$
   - $AB_{12} = p_1 + p_2 = b - d + 4d = b + 3d$
   - $AB_{21} = p_3 + p_4 = 11a + 5c - 5a = 6a + 5c$
   - $AB_{22} = p_1 + p_5 - p_3 - p_7 = b - d + 6a + 6d - 11a + 5a + 5b = 6b + 5d$

Our final result is:

$$\begin{pmatrix} 1 & 3 \\ 6 & 5 \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a + 3c & b + 3d \\ 6a + 5c & 6b + 5d \end{pmatrix}$$

$n = 100$ should be a reasonable estimate for the dimensions beyond which Strassen's Algorithm outperforms the Brute Force Approach.