INSTITUTE OF COMPUTER TECHNOLOGY

B-TECH COMPUTER SCIENCE ENGINEERING 2025-26

SUBJECT:-ALGORITHM ANALYSIS & DESIGN

NAME: Rahul Prajapati

ENRLL. NO: 23162171020

BRANCH: CYBER SECURITY

BATCH: 52

PRACTICAL_06

Aim: Given a sequence of matrices, we want to find the most efficient way to multiply these matrices together to obtain the minimum number of multiplications. The problem is not actually to perform the multiplication of the matrices but to obtain the minimum number of multiplications. We have many options because matrix multiplication is an associative operation, meaning that the order in which we multiply does not matter. The optimal order depends only on the dimensions of the matrices.

The brute-force algorithm is to consider all possible orders and take the minimum. This is a very inefficient method.

Implement the minimum multiplication algorithm using dynamic programming and determine where to place parentheses to minimize the number of multiplications.

Find an optimal parenthesization of a matrix chain product whose sequence of dimensions are (5,10, 3, 12, 5, 50, 6).

CODE:

```python
def matrix_chain_order(p):
    n = len(p) - 1 # number of matrices
    m = []
    for i in range(n):
        row = []
        for j in range(n):
            row.append(0)
        m.append(row)
    s = []
    for i in range(n):
        row = []
        for j in range(n):
            row.append(0)
        s.append(row)

    for l in range(2, n+1):
        for i in range(n - l + 1):
            j = i + l - 1
            m[i][j] = float('inf')
            for k in range(i, j):
                q = m[i][k] + m[k+1][j] + p[i]*p[k+1]*p[j+1]
                if q < m[i][j]:
                    m[i][j] = q
                    s[i][j] = k
    return m, s
def min_multi(s, i, j):
    if i == j:
        return f"A{i+1}"
    else:
        left = min_multi(s, i, s[i][j])
        right = min_multi(s, s[i][j]+1, j)
        return f"({left} x {right})"

dimensions = [5, 10, 3, 12, 5, 50, 6]
m, s = matrix_chain_order(dimensions)
min_mult = m[0][len(dimensions)-2]
min_req = min_multi(s, 0, len(dimensions)-2)

print("Minimum number of multiplications:", min_mult)
print("Optimal parenthesization:", min_req)
```

OUTPUT:

```
PS C:\Users\Hp\OneDrive\Desktop\SEM_05\Algorithm Analysis & Design\SOURCE_CODES> python .\Practical6_1.py
Minimum number of multiplications: 2010
Optimal parenthesization: ((A1 x A2) x ((A3 x A4) x (A5 x A6)))
```