Darshan Singh Chauhan

22162581001

BTech CSE

Batch 52

Institute of Computer Technology
B. Tech Computer Science and Engineering

Sub: Algorithm Analysis and Design

Practical 6

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).

```
import streamlit as st

def matrix_chain_order(p):
    n = len(p) - 1
    m = [[0] * n for _ in range(n)]
    s = [[0] * n for _ in range(n)]

    for l in range(2, n + 1):  # l is chain length
```

Darshan Singh Chauhan

22162581001

BTech CSE

Batch 52

```
        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 print_optimal_parens(s, i, j):
    if i == j:
        return f"A{i + 1}"
    else:
        return "(" + print_optimal_parens(s, i, s[i][j]) + print_optimal_parens(s, s[i][j] + 1, j) + ")"

# Streamlit UI
st.title("Matrix Chain Multiplication")
input_dimensions = st.text_input("Enter matrix dimensions (comma-separated):", "5,10,3,12,5,50,6")

if st.button("Calculate"):
    try:
        dimensions = list(map(int, input_dimensions.split(',')))
        m, s = matrix_chain_order(dimensions)
        min_multiplications = m[0][len(dimensions) - 2]
        optimal_parenthesization = print_optimal_parens(s, 0, len(dimensions) - 2)

        st.write(f"Minimum number of multiplications is: {min_multiplications}")
```

Darshan Singh Chauhan

22162581001

BTech CSE

Batch 52

```
        st.write(f"Optimal Parenthesization is: {optimal_parenthesization}")
    except Exception as e:
        st.error(f"Error: {e}")
```