DYNAMIC PROGRAMMING

CSE [AIML], 6TH SEMESTER

DESIGN & ANALYSIS OF ALGORITHMS

DYNAMIC PROGRAMMING

We follow sequence of 4 steps:

1. Characterize the structure of an optimal solution.
2. Recursively define the value of an optimal solution
3. Compute the value of an optimal solution
4. Construct an optimal solution from computed information.

MATRIX CHAIN MULTIPLICATION

- We state the Matrix Chain Multiplication as follows:

Given a chain {A1, A2, …, An} of matrices, where for i =1, 2, …, n, Ai has dimension $p_{i-1}\ X\ p_i$, fully parenthesized the product A1, A2, …, An in a way that minimized the number of scaler multiplication.

MATRIX CHAIN MULTIPLICATION

1. We can build an optimal solution to an instance of the problem by splitting the problem into two sub problems.
2. Finding optimal solutions to sub problem instances.
3. Combining these optimal sub problems solutions.

MATRIX CHAIN MULTIPLICATION

- A recursive solution (Formula):

$$m[i, j] = \begin{cases} 0, & if\ i = j \\ \min\limits_{i \le k < j}\{m[i, k] + m[k+1, j] + p_{i-1}p_kp_j\} & if\ i < j \end{cases}$$

MATRIX CHAIN MULTIPLICATION EXAMPLE

A1 A2 A3 A4

3x2 2x4 4x2 2x5

m[1,2] = min(k=1){m[1,1] + m[2,2]+ 3x2x4} = 24

m[2,3] = min(k=2){m[2,2] + m[3,3]+ 2x4x2} = 16

m[3,4] = min(k=3){m[3,3] + m[4,4]+ 4x2x5} = 40

MATRIX CHAIN MULTIPLICATION EXAMPLE CONTINUES…

A1 A2 A3 A4

3x2 2x4 4x2 2x5

m[1,3] = min(k = 1, k = 2 {m[1,1] + m[2,3]+ 3x2x2, m[1,2]+m[3,3]+3x4x2} = min{28,48} = 28

m[2,4] = min(k = 2, k = 3 {m[2,2] + m[3,4]+ 2x4x5, m[2,3]+m[4,4]+2x2x5} = min{80,36}=36

m[1,4] = min(k = 1, k = 2, k=3 {m[1,1] + m[2,4]+ 3x2x5, m[1,2]+m[3,4]+3x4x5, m[1,3]+m[4,4]+3x2x5} = min{86,124,58}= 58

MATRIX CHAIN MULTIPLICATION EXAMPLE CONTINUES…

m

0	24	28	58
	0	16	36
		0	40
			0

k

	1	1	3
		2	3
			3

The minimum number of scaler multiplication to multiply the 4 matrices is m[1,4] = 58

TIME & SPACE COMPLEXITY

- The loops are nested three times and each loop indexed at most n-1 values. This yields a running time of O(n^3).

The algorithm requires θ(n^2) space to store m and k tables.