Algorithms

Lecture 2: Dynamic Programming

Anxiao (Andrew) Jiang

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 2 & 1 \\ 3 & 5 & 1 & 2 \\ 6 & 2 & 8 & 0 \end{bmatrix}$$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}_{2\text{x}3} \begin{bmatrix} 0 & 1 & 2 & 1 \\ 3 & 5 & 1 & 2 \\ 6 & 2 & 8 & 0 \end{bmatrix}_{3\text{x}4}$$

$$= \left[\begin{array}{c|c|c|c} ? & & & \\ \hline & & & \end{array}\right]_{2\text{x}4}$$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

$$\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix}_{2\text{x}3}\begin{bmatrix}0 & 1 & 2 & 1\\ 3 & 5 & 1 & 2\\ 6 & 2 & 8 & 0\end{bmatrix}_{3\text{x}4}$$

$$= \begin{bmatrix}24 & & & \\ & & & \end{bmatrix}_{2\text{x}4}$$

$$1 \times 0 + 2 \times 3 + 3 \times 6 = 24$$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

$$A_{p\times q}\,B_{q\times r} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,q} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,q} \\ \vdots & \vdots & \ddots & \vdots \\ a_{p,1} & a_{p,2} & \cdots & a_{p,q} \end{bmatrix} \begin{bmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,r} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,r} \\ \vdots & \vdots & \ddots & \vdots \\ b_{q,1} & b_{q,2} & \cdots & b_{q,r} \end{bmatrix}$$

$$= \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,r} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,r} \\ \vdots & \vdots & \ddots & \vdots \\ c_{p,1} & c_{p,2} & \cdots & c_{p,r} \end{bmatrix}$$

$$c_{i,j} = a_{i,1}b_{1,j} + a_{i,2}b_{2,j} + \cdots + a_{i,q}b_{q,j} = \sum_{k=1}^{q} a_{i,k}b_{k,j}$$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

$$A_{p\times q}\, B_{q\times r} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,q} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,q} \\ \vdots & \vdots & \ddots & \vdots \\ a_{p,1} & a_{p,2} & \cdots & a_{p,q} \end{bmatrix} \begin{bmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,r} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,r} \\ \vdots & \vdots & \ddots & \vdots \\ b_{q,1} & b_{q,2} & \cdots & b_{q,r} \end{bmatrix}$$

$$= \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,r} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,r} \\ \vdots & \vdots & \ddots & \vdots \\ c_{p,1} & c_{p,2} & \cdots & c_{p,r} \end{bmatrix}$$

Cost of multiplying 2 matrices:

1) Scalar multiplications: pqr

2) Additions: p(q-1)r

$$c_{i,j} = a_{i,1}b_{1,j} + a_{i,2}b_{2,j} + \cdots + a_{i,q}b_{q,j} = \sum_{k=1}^{q} a_{i,k}b_{k,j}$$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

$$A_{p\times q}\,B_{q\times r} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,q} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,q} \\ \vdots & \vdots & \ddots & \vdots \\ a_{p,1} & a_{p,2} & \cdots & a_{p,q} \end{bmatrix} \begin{bmatrix} b_{1,1} & b_{1,2} & \cdots & b_{1,r} \\ b_{2,1} & b_{2,2} & \cdots & b_{2,r} \\ \vdots & \vdots & \ddots & \vdots \\ b_{q,1} & b_{q,2} & \cdots & b_{q,r} \end{bmatrix}$$

$$= \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,r} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,r} \\ \vdots & \vdots & \ddots & \vdots \\ c_{p,1} & c_{p,2} & \cdots & c_{p,r} \end{bmatrix}$$

Cost of multiplying 2 matrices:

1) Scalar multiplications: pqr

2) Additions: p(q-1)r

$$c_{i,j} = a_{i,1}b_{1,j} + a_{i,2}b_{2,j} + \cdots + a_{i,q}b_{q,j} = \sum_{k=1}^{q} a_{i,k}b_{k,j}$$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

$A_{10\times 100}\ B_{100\times 5}\ C_{5\times 50}$ What is the cost of computing this chain?

1) $(A_{10\times 100}\ B_{100\times 5})\ C_{5\times 50}$

2) $A_{10\times 100}\ (B_{100\times 5}\ C_{5\times 50})$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

$$A_{10\times100}\ B_{100\times5}\ C_{5\times50}$$

What is the cost of computing this chain?

1) $(A_{10\times100}\ B_{100\times5})_{10x5}\ C_{5\times50}$

Cost: $10\times100\times5+10\times5\times50=7{,}500$

2) $A_{10\times100}\ (B_{100\times5}\ C_{5\times50})_{100x50}$

Cost: $100\times5\times50+10\times100\times50=75{,}000$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

Input: A chain of n matrices $A_1 A_2 \cdots A_n$

For $i = 1, 2, \cdots, n,$ the matrix A_i has size $p_{i-1} \times p_i$

Output: How to parenthesize the matrix chain such that the total cost is minimized?

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

Input: A chain of n matrices $A_1 A_2 \cdots A_n$

For $i = 1, 2, \cdots, n,$ the matrix A_i has size $p_{i-1} \times p_i$

Output: How to parenthesize the matrix chain such that the total cost is minimized?

Chain: $A_1 \quad A_2 \quad A_3 \quad \cdots \quad A_{n-1} \quad A_n$

Size: $p_0 \times p_1 \quad p_1 \times p_2 \quad p_2 \times p_3 \quad \cdots \quad p_{n-2} \times p_{n-1} \quad p_{n-1} \times p_n$

Should we use exhaustive search? Time complexity is too high.

$(A_1 A_2 A_3)(A_4 A_5 A_6) \quad \cdots \quad (A_{n-2} A_{n-1} A_n)$ # of choices $> 2^{\lfloor n/3 \rfloor}$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

Chain: $A_1 \quad A_2 \quad A_3 \quad \cdots \quad A_{n-1} \quad A_n$

Size: $p_0 \times p_1 \quad p_1 \times p_2 \quad p_2 \times p_3 \quad \cdots \quad p_{n-2} \times p_{n-1} \quad p_{n-1} \times p_n$

For $1 \leq i \leq j \leq n$

let $m[i,j]$ be the minimum cost of computing the sub-chain $A_i A_{i+1} \cdots A_j$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

Chain: $A_1 \quad A_2 \quad A_3 \quad \cdots \quad A_{n-1} \quad A_n$

Size: $p_0 \times p_1 \quad p_1 \times p_2 \quad p_2 \times p_3 \quad \cdots \quad p_{n-2} \times p_{n-1} \quad p_{n-1} \times p_n$

For $1 \leq i \leq j \leq n$

let $m[i, j]$ be the minimum cost of computing the sub-chain $A_i A_{i+1} \cdots A_j$

$$A_i A_{i+1} \cdots A_k \; A_{k+1} \cdots A_j$$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

Chain: $A_1 \quad A_2 \quad A_3 \quad \cdots \quad A_{n-1} \quad A_n$

Size: $p_0 \times p_1 \quad p_1 \times p_2 \quad p_2 \times p_3 \quad \cdots \quad p_{n-2} \times p_{n-1} \quad p_{n-1} \times p_n$

For $1 \leq i \leq j \leq n$

let $m[i, j]$ be the minimum cost of computing the sub-chain $A_i A_{i+1} \cdots A_j$

$$(A_i A_{i+1} \cdots A_k)(A_{k+1} \cdots A_j)$$

Final
Multiplication

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

Chain: $A_1 \quad A_2 \quad A_3 \quad \cdots \quad A_{n-1} \quad A_n$

Size: $p_0 \times p_1 \quad p_1 \times p_2 \quad p_2 \times p_3 \quad \cdots \quad p_{n-2} \times p_{n-1} \quad p_{n-1} \times p_n$

For $1 \le i \le j \le n$

let $m[i, j]$ be the minimum cost of computing the sub-chain $A_i A_{i+1} \cdots A_j$

$$\underbrace{\left(A_i A_{i+1} \cdots A_k\right)}^{m[i,k]} \overbrace{\left(A_{k+1} \cdots A_j\right)}^{m[k+1, j]}$$

Final Multiplication

$p_{i-1} p_k p_j$

CH 15. Dynamic Programming

15.2 Matrix Chain Multiplication

Chain: $A_1 \quad A_2 \quad A_3 \quad \cdots \quad A_{n-1} \quad A_n$

Size: $p_0 \times p_1 \quad p_1 \times p_2 \quad p_2 \times p_3 \quad \cdots \quad p_{n-2} \times p_{n-1} \quad p_{n-1} \times p_n$

For $1 \le i \le j \le n$

let $m[i,j]$ be the minimum cost of computing the sub-chain $A_i A_{i+1} \cdots A_j$

$$(A_i A_{i+1} \cdots A_k)(A_{k+1} \cdots A_j)$$

Total cost = $m[i,k] + m[k+1,j] + p_{i-1}p_k p_j$

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

Recursive function

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

Compute bottom-up (from smaller to bigger):

Sub-chains of length 1: $m[1,1], \quad m[2,2], \quad m[3,3], \quad \cdots \quad, m[n,n]$ =0

Sub-chains of length 2: $m[1,2], \quad m[2,3], \quad m[3,4], \quad \cdots \quad, m[n-1,n]$

$A_1A_2 \quad A_2A_3 \quad A_3A_4 \quad A_{n-1}A_n$

Sub-chains of length 3: $m[1,3], \quad m[2,4], \quad m[3,5], \quad \cdots \quad, m[n-2,n]$

$A_1A_2A_3 \quad A_2A_3A_4 \quad A_3A_4A_5 \quad A_{n-2}A_{n-1}A_n$

……

Sub-chains of length n: $m[1,n]$

Time Complexity: $O(n^3)$

Recursive function $m[i,j] = \begin{cases} 0 & i = j \\ \min\limits_{i \le k < j} \underbrace{m[i,k] + m[k+1,j] + p_{i-1}p_kp_j}_{O(1)} & i < j \end{cases}$

O(n)

Complexity of computing each $m[i,j]$: $O(n)$

Since $1 \le i \le j \le n$, we have $O(n^2)$ such $m[i,j]$.