More Dynamic Programming:

Matrix Chain Multiplication

Slides based on Kevin Wayne / Pearson-Addison Wesley

Matrix Chain Multiplication

Problem Statement

- Given a chain of matrices to be multiplied together, determine a parenthesizing of the chain that minimizes the total number of steps required to complete the multiplication

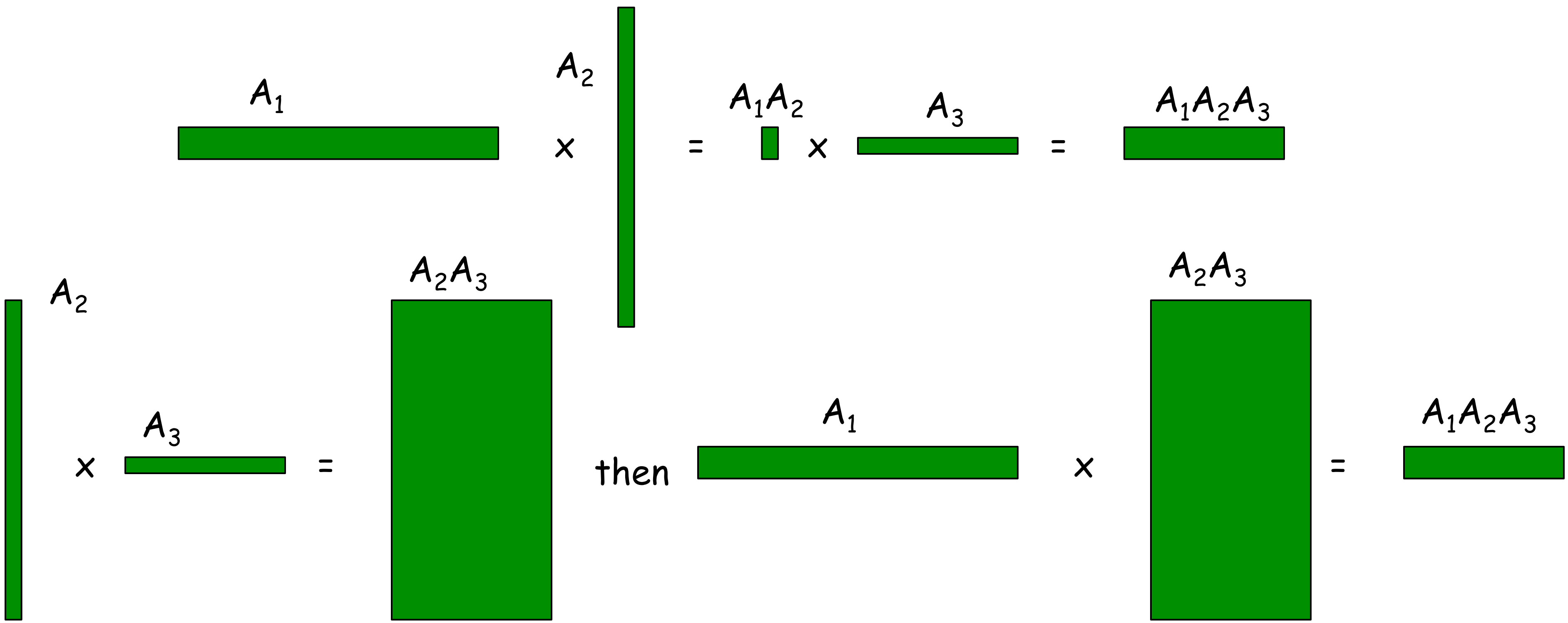
- Matrix Multiplication
  - Given $A_{p \times q}$ and $B_{q \times r}$
  - AB requires pqr total element-wise multiplications
  - (and a similar number of additions, but we will ignore these)

  - Example: A_1 10 x 100, A_2 100 x 5, A_3 5 x 50
    - $(A_1 A_2) A_3$ requires 10 x 100 x 5 + 10 x 5 x 50 = 7500
    - $A_1 (A_2 A_3)$ requires 100 x 5 x 50 + 10 x 100 x 50 = 75000

Matrix Chain Multiplication

Example

- A_1: 10 x 100
- A_2: 100 x 5
- A_3: 5 x 50
  - $(A_1\ A_2)\ A_3$ requires 10 x 100 x 5 + 10 x 5 x 50 = 7500
  - $A_1\ (A_2\ A_3)$ requires 100 x 5 x 50 + 10 x 100 x 50 = 75000



Matrix Chain Multiplication

How Many Ways P(n) of Parenthesizing are Possible for $A_1A_2A_3 \ldots A_n$?

- For n = 1, P(n) = 1
- For n > 1, the final product is the product of two fully parenthesized matrix subproducts, where the split can occur anywhere after matrix 1,2,…,n-1

$$P(n) = \sum_{k=1}^{n-1} P(k)P(n-k)$$

- $P(n) = \Omega(4^n / n^{3/2})$
  - Exponential time to check all possible parentheses placements

Matrix Chain Multiplication

Dynamic Programming Solution

- Optimal substructure
  - Total number of multiply operations needed for a given split:
    - Multiplications needed for optimal solution to subproduct 1
    - Multiplications needed for optimal solution to subproduct 2
    - Multiplications needed to combine subproducts 1 and 2
  - The optimal solution will be the best choice among all possible splits

Matrix Chain Multiplication

Dynamic Programming Solution

- Suppose we have n matrices: $A_1 \times A_2 \times \ldots \times A_n$
  - The dimensions of all the matrices can be completely specified with $a_0, a_1, a_2, \ldots, a_n$
  - A_i has dimensions $a_{i-1} \times a_i$
  - Interior dimensions of neighboring matrices are the same

First Try

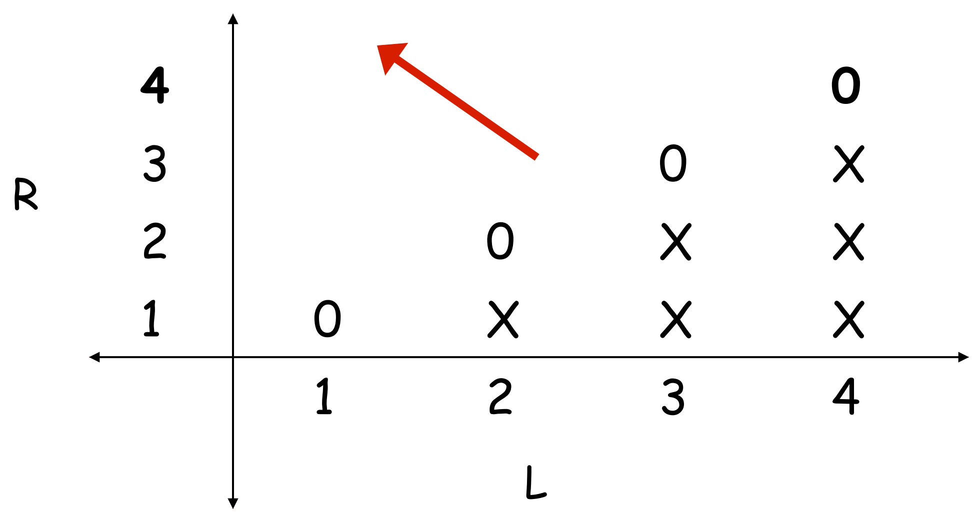
- $S[n]$:= minimum number of multiplications needed to combine the first n matrices
  - $S[n] = \min_{k=1,2,\ldots,n-1} (a_0 a_k a_n + S[k] + S[?])$
  - Need a subproblem corresponding to the optimal solution for multiplying $A_{k+1} \times \ldots \times A_n$

Matrix Chain Multiplication

Dynamic Programming Solution: Second Try

- $S[L][R]$:= minimum number of multiplications needed to combine matrices A_L through A_R
  - $S[L][R] = \min_{k=L,L+1,\ldots,R-1} (a_{L-1}a_k a_R + S[L][k] + S[k+1][R])$ for $1 \le L < R \le n$
  - $S[L][L] = 0$

- How is the matrix filled in?
  - From diagonal outward, using increasing size of [L,R] interval



Matrix Chain Multiplication

Dynamic Programming Solution:

- Complexity: $O(n^3)$
  - Three nested loops each iterating over at most n items

```
MATRIX-CHAIN-MULTIPLICATION (a_0,…,a_n)
for L=1 to n do S[L][L] = 0
for d=1 to n-1 do                              d loops over the size of interval
    for L=1 to n-d do                          L loops over possible left endpoints
        R = L+d
        S[L][R] =∞
        for k=L to R-1
            tmp = S[L][k]+S[k+1][R]+a_{L-1}.a_k.a_R
            if S[L][R] > tmp then S[L][R] = tmp
return S[1][n]
```