Matrix-Chain Multiplication

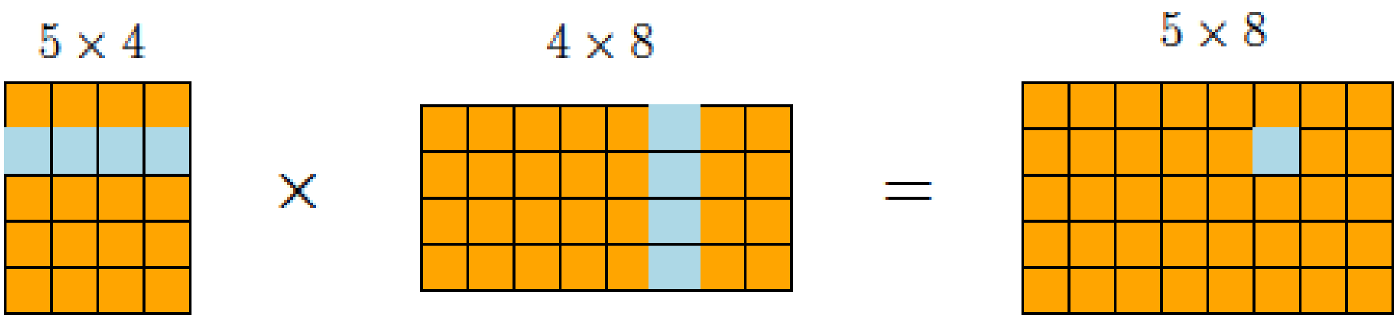
Version of March 5, 2019

Matrix-chain Multiplication

The product of two matrices $A_{p \times q}$ and $B_{q \times r}$

(with dimensions $p \times q$ and $q \times r$) is a matrix $C_{p \times r}$.

Generating $C_{p \times r}$ requires pqr scalar multiplications.



Given matrices A, B with entries $a_{i,j}, b_{i,j}$ the entries in the product matrix C= A B are

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

Diagrams on this and the next few pages modified from
http://ramos.elo.utfsm.cl/~lsb/elo320/aplicaciones/aplicaciones/CS460AlgorithmsandComplexity/lecture17

Matrix-chain Multiplication

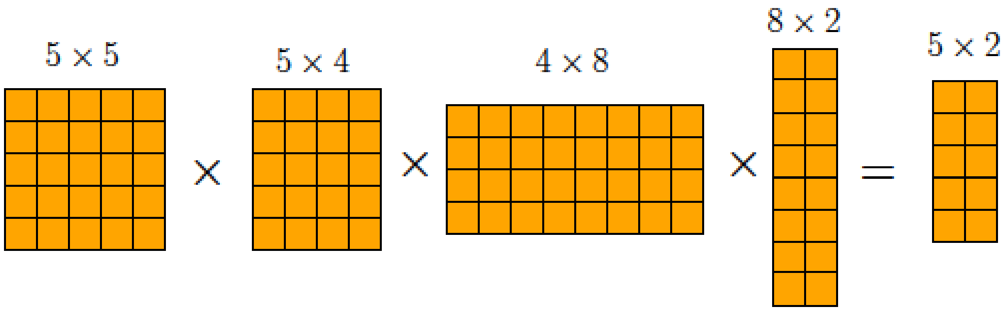
The product of two matrices $A_{p\times q}$ and $B_{q\times r}$ (with dimensions $p\times q$ and $q\times r$) is a matrix $C_{p\times r}$. Generating $C_{p\times r}$ requires pqr scalar multiplications.

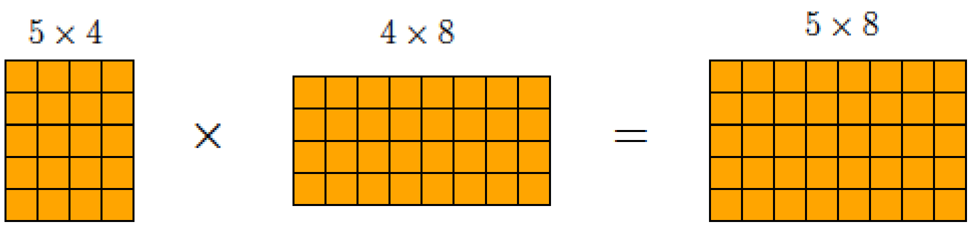
For three matrices (e.g., $A_{10\times 100}$, $B_{100\times 5}$, and $C_{5\times 50}$) there are 2 ways to parenthesize

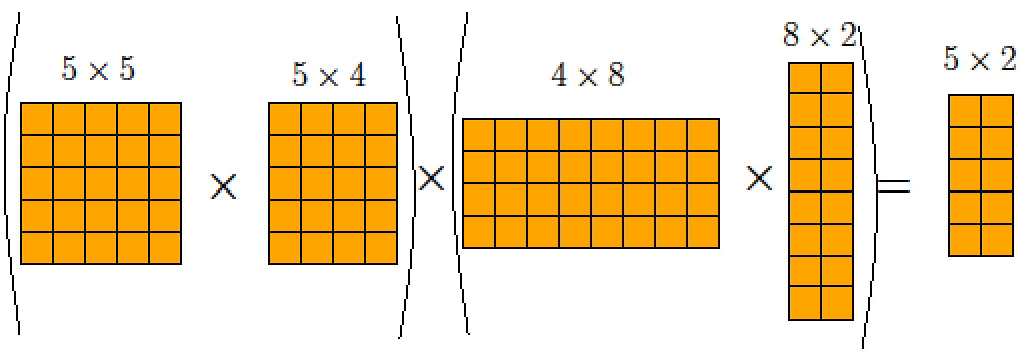
- $((AB)C) = D_{10\times 5} \cdot C_{5\times 50}$
  - $AB \Rightarrow 10\cdot 100\cdot 5 = 5{,}000$ scalar multiplications
  - $DC \Rightarrow 10\cdot 5\cdot 50 = 2{,}500$ scalar multiplications
- $(A(BC)) = A_{10\times 100} \cdot E_{100\times 50}$
  - $BC \Rightarrow 100\cdot 5\cdot 50 = 25{,}000$ scalar multiplications
  - $AE \Rightarrow 10\cdot 100\cdot 50 = 50{,}000$ scalar multiplications

General problem: We have a sequence or chain $A_1, A_2, \ldots, A_n$ of n matrices and want to determine the optimal way to parenthesize (i.e., the solution with the minimum number of scalar multiplications).

Matrix-chain Multiplication







There are 5 different ways to multiply ABCD together

1. (A (B (CD)))
2. (A ((BC) D))
3. (((AB) (CD)))
4. ((A (BC)) D)
5. (((AB) C) D)

Costs are

1. 5(5)2 + 5(4)2 + 4(8)2 = 154
2. 5(5)2 + 5(4)8 + 5(8)2 = 290
3. 5(5)4 + 4(8)2 + 5(4)2 = 204
4. 5(5)8 + 5(4)8 + 5(8)2 = 440
5. 5(5)4 + 5(4)8 + 5(8)2 = 340

Recall: Multiplying
p×q and q×r matrices requires
p×q×r multiplications
And yields a p×r matrix

Matrix-chain Multiplication Problem Definition

- Input: Values $p_0\ p_1 \ldots p_{n-1}\ p_n$
- These represent sizes of n matrices $A_1\ A_2 \ldots A_n$
  Matrix A_i has dimensions p_{i-1} X p_i

- $A_{i..j}$: matrix that is the product of $A_i\ A_{i+1} \ldots A_j$
  By construction $A_{i..j}$ has dimensions p_{i-1} X p_j

- Goal: To find a minimum cost way of multiplying $A_1\ A_2 \ldots A_n$ to get the final result $A_{1..n}$.
  cost = # of total scalar multiplications performed

- This is known as an optimal parenthesization of $A_1\ A_2 \ldots A_n$ because the parentheses denote how to perform the multiplications,
  - e.g., (((AB) (CD))) means first calculate X=AB then calculate Y=CD and finally get the final result XY

Structure of an optimal solution

- Given: Values $p_0\, p_1 \ldots p_{n-1}\, p_n$ *s.t.* Matrix A_i has size p_{i-1} X p_i
- $A_{i..j}$: matrix that results from the product $A_i\, A_{i+1} \ldots A_j$
- An optimal parenthesization of $A_1 A_2 \ldots A_n$ splits the product between A_k and A_{k+1} for some integer k where $1 \le k < n$
  $A_{1..n} = (A_1\, A_2 \ldots A_k) \cdot (A_{k+1} A_{k+2} \ldots A_n) = A_{1..k} \cdot A_{k+1..n}$
- First compute matrices $A_{1..k}$ and $A_{k+1..n}$; then, multiply them to get the final matrix $A_{1..n}$
- **Observation**: If the parenthesization of the chain $A_1 A_2 \ldots A_n$ is optimal => parenthesizations of the subchains $A_1 A_2 \ldots A_k$ and $A_{k+1} A_{k+2} \ldots A_n$ must also be optimal (why?)

  => The optimal solution to the problem contains within it the optimal solution to subproblems

Recursive definition for optimal solution

- Let $m[i, j]$ be minimum number of scalar multiplications necessary to compute $A_{i..j}$
- Suppose the optimal parenthesization of $A_{i..j}$ splits the product between A_k and A_{k+1} for some integer k where $i \le k < j$
  - $A_{i..j} = (A_i A_{i+1} \ldots A_k) \cdot (A_{k+1} A_{k+2} \ldots A_j) = A_{i..k} \cdot A_{k+1..j}$
- Cost of computing $A_{i..j}$ =
  cost of computing $A_{i..k}$ + cost of computing $A_{k+1..j}$ + cost multiplying $A_{i..k}$ and $A_{k+1..j}$
  - Cost of multiplying $A_{i..k}$ and $A_{k+1..j}$ is $p_{i-1} p_k p_j$
- But… optimal parenthesization occurs at one value of k among all possible $i \le k < j$. Check all these and select the best one

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

DP Algorithm

Input: Array $p[0 \ldots n]$ containing matrix dimensions and n

Result: Minimum-cost table m and split table s

MATRIX-CHAIN-ORDER($p[\ ], n$)

> **for** $i \leftarrow 1$ **to** n
>
> $m[i, i] \leftarrow 0$
>
> **for** $l \leftarrow 2$ **to** n
>
>   **for** $i \leftarrow 1$ **to** $n\text{-}l+1$
>
> $j \leftarrow i+l\text{-}1$
>
> $m[i, j] \leftarrow \infty$
>
>   **for** $k \leftarrow i$ **to** $j\text{-}1$
>
> $q \leftarrow m[i, k] + m[k+1, j] + p[i\text{-}1]\, p[k]\, p[j]$
>
>   **if** $q < m[i, j]$
>
> $m[i, j] \leftarrow q$
>
> $s[i, j] \leftarrow k$
>
> **return** m and s

Time: $O(n^3)$, Space: $O(n^2)$

Example

The initial set of dimensions are <5, 4, 6, 2, 7>.

We are multiplying A_1 (5x4) x A_2 (4x6) x A_3 (6x2) x A_4 (2x7).

Optimal sequence is $((A_1(A_2A_3))\,A_4)$.