Optimal Parenthesization for a Chained Matrix Product

Textbook Reading:

- Section 8.2 Optimal Parenthesization for Computing a Chained Matrix Product, pp. 333-339.

Order matrix multiplication is done makes a big difference in computing time

- Given two $n \times n$ matrices A and B are and an $n \times 1$ matrix (column vector) X
- The product ABX can be parenthesized in two ways:
  - $(AB)X$ resulting in $n^3 + n^2$ multiplications and
  - $A(BX)$ resulting in only $2n^2$ multiplications
  - The two ways of parenthesizing make a dramatic difference in the number of multiplications performed (order $\Theta(n^3)$ versus order $\Theta(n^2)$).

Number of multiplications in a matrix product

Let A and B be matrices having dimensions $p \times q$ and $q \times r$, respectively.

Their product $C = (c_{ij})_{p \times r}$ has dimensions $p \times r$.

c_{ij} is the **dot product** of the ith row of A and the jth column of B.

ith row of A and the jth column of B are both of **size *q*.**

To compute C, we perform ***pr* dot products** involving vectors of **size *q*.** Each dot product involves q multiplications. Therefore, to compute $C = AB$, we perform a total of ***pqr* multiplications.**

Fully-parenthesized matrix product

Given the sequence of matrices M_0,M_1,...,M_{n-1}, P is a **fully-parenthesized** matrix product $M_0M_1...M_{n-1}$, which for convenience we simply call a *parenthesization of* $M_0M_1...M_{n-1}$, if P satisfies

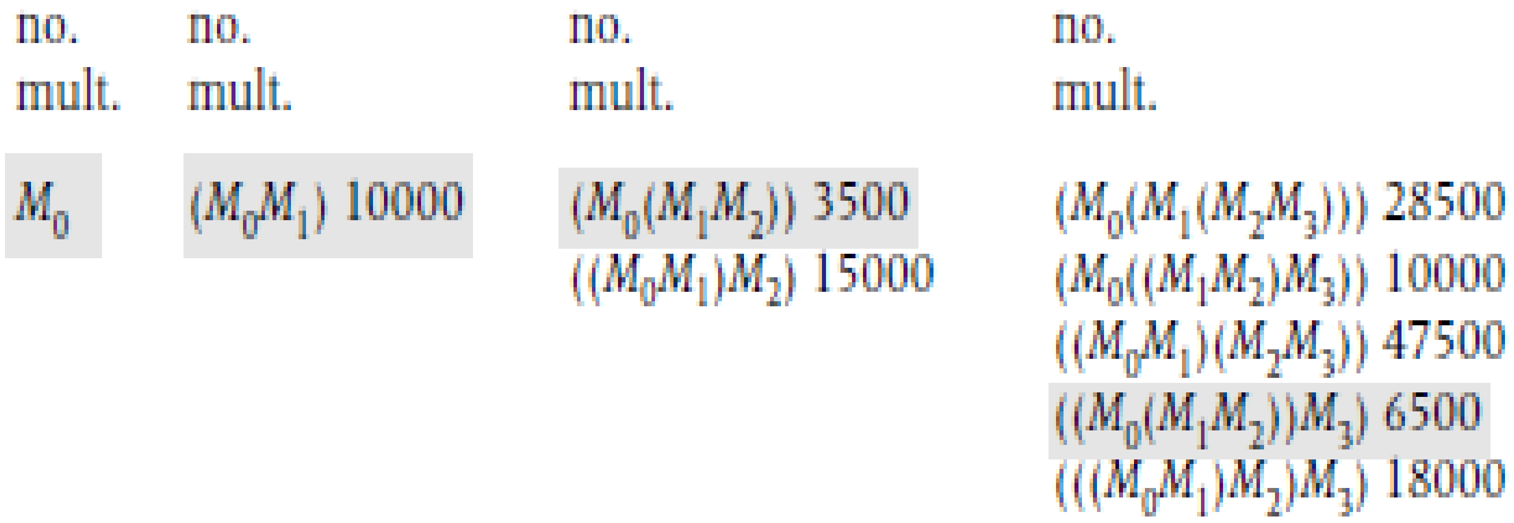
$$P = \begin{cases} P = M_0, & n = 1, \\ (P_1P_2), & n > 1. \end{cases}$$

where for some k, P_1 and P_2 are parenthesizations of the matrix products $M_0M_1...M_k$ and $M_{k+1}M_{k+2}...M_{n-1}$, respectively. We call P_1 and P_2 the ***left*** and ***right*** parenthesizations of P, respectively. We call the index k the ***first cut index*** *of* P.

Example

Number of multiplications performed for each full parenthesization are shown for matrices M_0,M_1,M_2, M_3 having dimensions 20×10, 10×50, 50×5, 5×30.

The optimal parenthesizations are shaded.



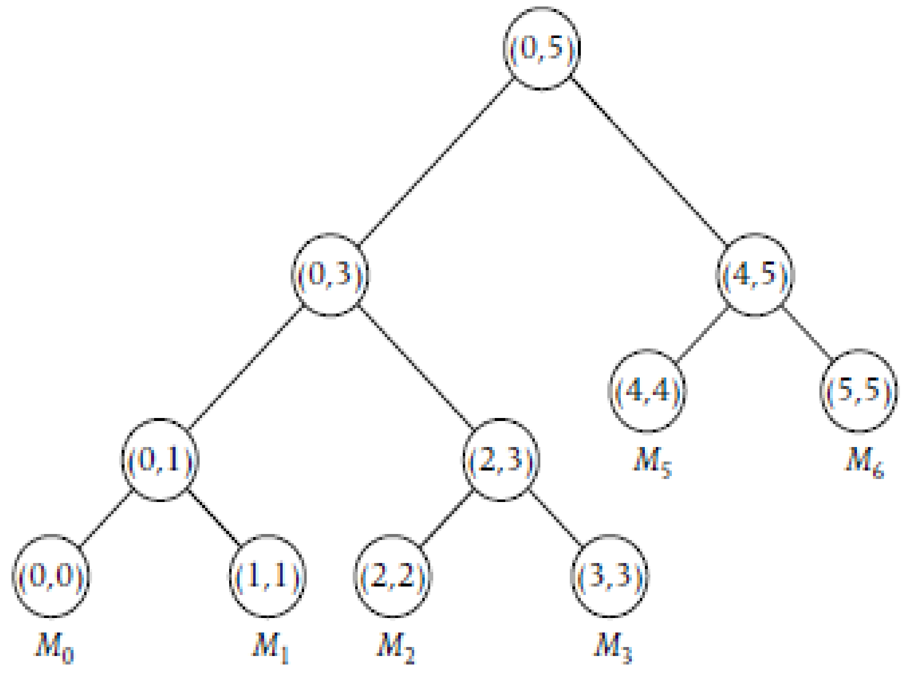
no. mult.	no. mult.	no. mult.	no. mult.
M_0	(M_0M_1) 10000	$(M_0(M_1M_2))$ 3500	$(M_0(M_1(M_2M_3)))$ 28500
		$((M_0M_1)M_2)$ 15000	$(M_0((M_1M_2)M_3))$ 10000
			$((M_0M_1)(M_2M_3))$ 47500
			$((M_0(M_1M_2))M_3)$ 6500
			$(((M_0M_1)M_2)M_3)$ 18000

The first cut indices for the parenthesized matrix product in the last column are 0, 0, 1, 2, 2.

1-to-1 correspondence with 2-trees

- There is one-to-one correspondence between parenthesizations of $M_0M_1...M_{n-1}$ and 2-trees having n leaf nodes.
- Given a parenthesization P of $M_0M_1...M_{n-1}$, if $n = 1$ its associated 2-tree $T(P)$ consists of a single node corresponding to the matrix M_0; Otherwise, $T(P)$ has left subtree $T(P_1)$ and right subtree $T(P_2)$, where P_1 and P_2 are the left and right parenthesizations of P.

Example



- Associated expression 2-tree for parenthesization

$$(((M_0M_1)(M_2M_3))(M_4M_5))$$

- The label (i,j) inside each node indicates that the matrix product associated with the node involves matrices

$$M_i,M_{i+1},\ldots,M_j\,.$$

Number of parenthesizations is exponential

From previous slide it follows that the number of parenthesizations for a product of *n* matrices equals the number of 2-trees having *n* vertices which equals

$$\frac{1}{n}\binom{2n-2}{n-1} \geq \frac{4^{n-1}}{2n^2 - n}\,.$$

Brute force is lousy

Number of cases to consider is exponential, so a brute force algorithm will not work in real time for large *n*.

Dynamic Programming Solution

We will use recursion

$$P = \begin{cases} P = M_0, & n = 1, \\ (P_1 P_2), & n > 1. \end{cases}$$

Let $m(P)$ denote the number of multiplication involved using parenthesization P. Then

$$m(P) = m(P_1) + m(P_2) + pqr,$$

where $p \times q$ and $q \times r$ are the dimensions of the matrices P_1 and P_2, respectively.

Principle of Optimality Holds

The combine operation is

$$m(P) = m(P_1) + m(P_2) + pqr$$

PSN. Give a proof by contradiction that shows that both the left and right parenthesizations P_1 and P_2 of P must be optimal for P to be optimal.

Input to our algorithm

The values of the entries of the matrices are not relevant to computing an optimal parenthesization, only their dimensions. Thus, the input to our algorithm is the vector

$$d = (d_0, d_1, \ldots, d_n).$$

where $d_i \times d_{i+1}$ are the dimensions of the matrix M_i, $i = 0, \ldots, n - 1$.

Recurrence Relation

- Let m_{ij} denote the number of multiplications performed using an optimal parenthesization of $M_i M_{i+1} \dots M_j$, $0 \leq i \leq j \leq n - 1$.
- By the Principle of Optimality, we have the following recurrence for m_{ij} based on making an optimal choice for the first cut index:

$$m_{ij} = \min_k \{ m_{ik} + m_{k+1,j} + d_i d_{k+1} d_{j+1} : 0 \leq i \leq k < j \leq n - 1 \}$$
$$m_{ii} = 0, i = 0, \dots, n - 1$$

Dynamic Programming Algorithm

We store the m_{ij} values in the upper triangular portion of the matrix $m[0:n-1, 0:n-1]$. Note that $m_{ii} = 0$, so we initialize the main diagonal entries to 0. Then, we compute

$m_{01}, m_{12}, \ldots, m_{n-1,n-2}, m_{n-2,n-1}$

$m_{02}, m_{13}, \ldots, m_{n-3,n-1}$

.

.

.

$m_{0,n-2}, m_{1,n-1}$

$m_{0,n-1}$.

At each stage we keep track of the **first cut** index that we used in obtaining m_{ij}. Note that $m_{0,n-1}$ is the minimum number of multiplications over all parenthesizations. The parenthesization that achieve this is obtained using the first cuts we maintained.

```
procedure OptimalParenthesization($d[0:n]$,$m[0:n-1,0:n-1]$, $FirstCut[0:n-1,0:n-1]$)
Input:  $d[0:n]$ (dimension sequence for matrices $M_0, M_1,\ldots,M_{n-1}$)
Output:  $m[0:n-1,0:n-1]$ ($m[i,j]$ = number of multiplications performed in an optimal parenthesization for computing $M_i\ldots M_j$, $0 \le i \le j \le n-1$)
        $FirstCut[0:n-1,0:n-1]$          // index of first cut in optimal  parenthesization of $M_i\ldots M_j$, $0 \le i \le j \le n-1$
          for $i \leftarrow 0$ to $n-1$ do
                $m[i,i] \leftarrow 0$
          endfor
          for $diag \leftarrow 1$ to $n-1$ do
              for $i \leftarrow 0$ to $n-1-diag$ do
                  $j \leftarrow i + diag$
                  $Min \leftarrow m[i+1,j] + d[i]*d[i+1]*d[j+1]$
                  $TempCut \leftarrow i$
                  for $k \leftarrow i+1$ to $j-1$ do
                      $Temp \leftarrow m[i,k] + m[k+1,j] + d[i]*d[k+1]*d[j+1]$
                      if $Temp < Min$ then
                          $Min \leftarrow Temp$
                          $TempCut \leftarrow k$
                      endif
                  endfor
                  $m[i,j] \leftarrow Min$
                  $FirstCut[i,j] \leftarrow TempCut$
              endfor
          endfor
end OptimalParenthesization
```

Complexity Analysis

A simple loop counting shows that the worst-case complexity of *OptimalParenthesization* is given by

$$W(n) \in \Theta(n^3)$$

Performing Matrix Product Using Optimal Parenthesization

```
function ChainMatrixProd(i,j) recursive
Input:   i,j (indices delimiting matrix chain $M_i,...,M_j$)
         $M_0,...,M_{n-1}$ (global matrices)
         FirstCut[0:n – 1,0:n – 1] (global matrix computed by
                                        OptimalParenthesization)
Output: $M_i...M_j$ (chain matrix product)
         if $j > i$ then
                X ← ChainMatrixProd(i,FirstCut[i,j])
                Y ← ChainMatrixProd(FirstCut[i,j] + 1,j)
                return(MatrixProd(X,Y))
         else
                return($M_i$)
         endif
end ChainMatrixProd
```

Example

Find optimal parenthesization for

$d = (20,10,50,5,30)$

M_0, M_1, M_2, M_3 having dimensions

$20 \times 10,\ 10 \times 50,\ 50 \times 5,\ 5 \times 30$

$$d = (20, 10, 50, 5, 30)$$

$m_{00} = m_{11} = m_{22} = m_{33} = 0$

$m_{01} = 20 \times 10 \times 50 = 10000$

$m_{12} = 10 \times 50 \times 5 = 2500$

$m_{23} = 50 \times 5 \times 30 = 7500$

$m_{02} = \min\{m_{00} + m_{12} + 20 \times 10 \times 5\ , m_{01} + m_{22} + 20 \times 50 \times 5\}$

$\quad = \min\{\mathbf{0} + \ \mathbf{2500} + \mathbf{1000}, 10000 + 0 + 5000\} = 3500,$ First Cut = 0

$m_{13} = \min\{m_{11} + m_{23} + 10 \times 50 \times 30\ , m_{12} + m_{33} + 10 \times 5 \times 30\}$

$\quad = \min\{0 + 7500 + 15000, \mathbf{2500} + \mathbf{0} + \mathbf{1500}\} = 4000,$ First Cut = 2

m_{03}

$= \min\{m_{00} + m_{13} + 20 \times 10 \times 30, m_{01} + m_{23} + 20 \times 50 \times 30, m_{02} + m_{33} + 20 \times 5 \times 30\}$

$= \min\{0 + 4000 + 6000, 10000 + 7500 + 3000, \mathbf{3500} + \mathbf{0} + \mathbf{3000}\} = 6500,$ First Cut = 2

Solution cont'd

Associated matrices:

$$m = \begin{pmatrix} 0 & 10000 & 3500 & 6500 \\ & 0 & 2500 & 4000 \\ & & 0 & 7500 \\ & & & 0 \end{pmatrix} \quad \textit{FirstCut} = \begin{pmatrix} & & 0 & 2 \\ & & & 2 \\ & & & \\ & & & \end{pmatrix}$$

Computing optimal parenthesization:

$$(M_0 M_1 M_2 M_3)$$

For 03, first cut at *k* = 2:

$$((M_0 M_1 M_2) M_3)$$

For 02 first cut at *k* = 0:

$$(((M_0 (M_1 M_2) M_3)$$

PSN

Find optimal parenthesization for

$d = (3,10,4,10,1)$

M_0, M_1, M_2, M_3 having dimensions

$3 \times 10,\ 10 \times 4,\ 4 \times 10,\ 10 \times 1$

Prove 11 = 10 = 9

XI = X = IX

for any matrix X
where I is the
identity matrix