Algorithms Design and Analysis [ETCS-301]

Dr. A K Yadav
Amity School of Engineering and Technology
(affiliated to GGSIPU, Delhi)
akyadav1@amity.edu
akyadav@akyadav.in
www.akyadav.in
+91 9911375598

August 29, 2019

Matrix Chain Multiplication(MCM)

- Matrix chain multiplication is not the multiplication of the matrices but it is the way to find the order of matrix multiplication with minimum number of scalar multiplication.
- It is mainly fully parenthesization of the matrices
- ABC can be multiplied by two way : $A(BC)$ or $(AB)C$.
- Which one cost the minimum number of scalar multiplication can be found using MCM
- Find the order of multiplication of the matrices $A_1 A_2 \ldots A_n$
- We need an array p of size $n+1$ to store the dimensions of n compatible matrices

Step 1: The structure of an optimal parenthesization I

- Let $A_{i \dots j} = A_i \dots A_j$ is the product of matrices from $A_i, A_{i+1}, \dots A_j$ for $i \leq j$
- if $i = j$ then there is only one matrix and number of scalar multiplication is zero.
- if $i < j$ then we can put parenthesis anywhere after A_i but before A_j
- Suppose we use parenthesis after matrix A_k i.e. $(A_i \dots A_k)(A_{k+1} \dots A_j)$ where $i \leq k < j$ for optimal solutions
- Now the total cost of the $A_{i \dots j}$ will be cost of $A_{i \dots k}$ plus cost of $A_{k+1 \dots j}$ plus the cost of multiplying them together.
- We supposed k is at optimal position so no other value of k is less costly.
- So $A_{i \dots k}$ and $A_{k+1 \dots j}$ is also optimal.

Step 1: The structure of an optimal parenthesization II

- We can find out the optimal solution of the the problem from optimal solution of the the sub-problem
- We have to take the correct value of $k, i \leq k < j$ such that the sub-problems having the optimal solutions.
- We have to examine all possible value of k for the optimal solution.

Step 2: A recursive solution I

We have to define cost of an optimal solution recursively in terms of the optimal solutions to sub-problems.

- Let $m[i,j]$ be the minimum number of scalar multiplications needed to compute the matrix $A_{i \ldots j}$ where $1 \leq i \leq j \leq n$
- The lowest cost to multiply $A_{1 \ldots n}$ will be $m[1,n]$.
- If $i = j$ that is there is only one matrix then no need to multiply. The $m[i,i] = 0$ for all $1 \leq i \leq n$.
- If $i < j$ then we split the matrix at position k for optimal solution.
- The total cost will be $m[i,j] = m[i,k] + m[k+1,j] + p_{i-1} \times p_k \times p_j$
- For all possible values of $k = i, i+1, \ldots, j-1$ we have to find out $m[i,j]$
- We pick that value of k for which $m[i,j]$ is minimum.

Step 2: A recursive solution II

- $\therefore$ Recursive definition for the minimum cost of parenthesizing the product $A_{i...j} = A_i A_{i+1} \ldots A_j$ will be:

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

- Store the value of k in $s[i,j]$ for which $m[i,j]$ is minimum.

Step 3: Computing the optimal costs I

- We can find the solution of the above recurrence by using two methods: Bottom-up or top-down.
- We compute overlapping sub-problems only once to avoid the exponential time cost.
- A_i is on dimensions $p_{i-1} \times p_i$.
- Table $m[1 \dots n, 1 \dots n]$ is used for storing $m[i, j]$ and $s[1 \dots n-1, 2 \dots n]$ is used to store the value of k for which $m[i, j]$ is minimum for all $i \leq k < j$.

Step 3: Computing the optimal costs II

Bottom-up Approach:

MATRIX-CHAIN-ORDER(p,n)

```
Let m[1...n, 1...n] and s[1...n-1, 2...n]
for i = 1 to n
    m[i,i] = 0
for l = 2 to n // l is the chain length
    for i = 1 to n - l + 1
        j = i + l - 1
        m[i,j] = ∞
        for k = i to j - 1
            q = m[i,k] + m[k+1,j] + p_{i-1} × p_k × p_j
            if q < m[i,j]
                m[i,j] = q
                s[i,j] = k
return m, s
```

Top-down Approach:

MEMOIZED-MATRIX-CHAIN(p,n)

1. Let $m[1 \dots n, 1 \dots n]$
2. for $i = 1$ to n
3.   for $j = i$ to n
4. $m[i, i] = \infty$
5. return LOOKUP-CHAIN(m,s,p,1,n)

LOOKUP-CHAIN(m,s,p,i,j)

1. if $m[i, j] < \infty$
2.   return $m[i, j]$
3. if $i = j$
4. $m[i, j] = 0$
5. else for $k = i$ to $j - 1$

Step 3: Computing the optimal costs IV

$q =$LOOKUP-CHAIN(m,s,p,i,k)
    +LOOKUP-CHAIN(m,s,p,k+1,j)$+p_{i-1}p_kp_j$
if $q < m[i,j]$
    $m[i,j] = q$
    $s[i,j] = k$
return $m[i,j]$

Step 4: Constructing an optimal solution I

- Table $m[i,j]$ gives the number of scalar multiplication to multiply matrices from A_i to A_j but does not show the order of multiplication
- Table $s[i,j]$ store the position of the parenthesis to partition the matrices from A_i to $A_k = A_{s[i,j]}$ and $A_{s[i,j]+1}$ to A_j.
- So the multiplication will be $(A_i \dots A_{s[i,j]})$ and $(A_{s[i,j]+1} \dots A_j)$

PRINT-OPTIMAL-PARENS(s,i,j)

1. if $i = j$
2.   print A_i
3. else print "("
4.   PRINT-OPTIMAL-PARENS$(s, i, s[i,j])$
5.   PRINT-OPTIMAL-PARENS$(s, s[i,j] + 1, j)$
6.   print ")"

Complexity of the MCM is $O(n^3)$

Thank you

Please send your feedback or any queries to akyadav1@amity.edu, akyadav@akyadav.in or contact me on +91 9911375598