Matrix Chain Multiplication

Problem: Chain of n matrices to be multiplied & we wish to compute the product

Using standard algorithm for multiplying pair of matrices at a time we find the complete solution after having n ~~multip~~ matrix multiplication.

The ~~Cop~~ Cost of multiplication of two matrices depends on order of the matrices

Say matrix $A_{m \times n}$ & Matrix $B_{n \times l}$

then the cost of Product of Matrices A & B is $m \times n \times l$ which means the total cost depends on order of matrices at each iterations

No of ways to solve Product of chain of n matrices is $\frac{2(n-1)!}{n!(n-1)!}$

Let us take an example.

$\langle A_1 A_2 A_3 \rangle$ with order 10×100, 100×5, 5×50 respectively

Now are more than one ways to find the product of chain of Matrices.

(i) $((A_1 \times A_2) \times A_3) = 10 \times 100 \times 5 + 10 \times 5 \times 50 = 5000 + 2500 = 7500$

(ii) $(A_1 \times (A_2 \times A_3)) = 100 \times 5 \times 50 + 10 \times 100 \times 50 = 25000 + 50000 = 75000$

So we find that the total cost depends on the how we parenthesize the matrices to find the solution here method 1 is better than Method 2.

So formally the Problem is defined, as

Given a Chain $\langle A_1, A_2, \ldots, A_n \rangle$ of n matrices, where for $i = 1, 2, \ldots n$, matrix A_i has dimension $P_{i-1} \times P_i$ fully parenthesize the product $A_1 A_2 \ldots A_n$ in a way that minimizes the total Cost of Multiplications.

In Matrix Chain "Multiplication" problem, we are not actually Multiplying Matrices.
The Goal is to find the order for multiplying Matrices that has the lowest Cost.

```
OPT (P, i, j)
{
    if j = i
        return 0          if M[i,j] > -1     } Memoization
                              return M[i,j]  }

    Cost = min_k { OPT(P,i,k) + OPT(k+1, j, P) + P_{i-1} * P_k * P_j

    M[i,j] = Cost            // memoization
    return Cost
}.
```

Iterative version

```
Matrix Chain Order (P)
    n ← length[P] - 1
    for i ← 1 to n
        do m[i,i] ← 0
    for l ← 2 to n
        for i ← 1 to n-l+1
            do j ← i+l-1
                m[i,j] ← ∞
                for k ← i to j-1
                    do 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
```

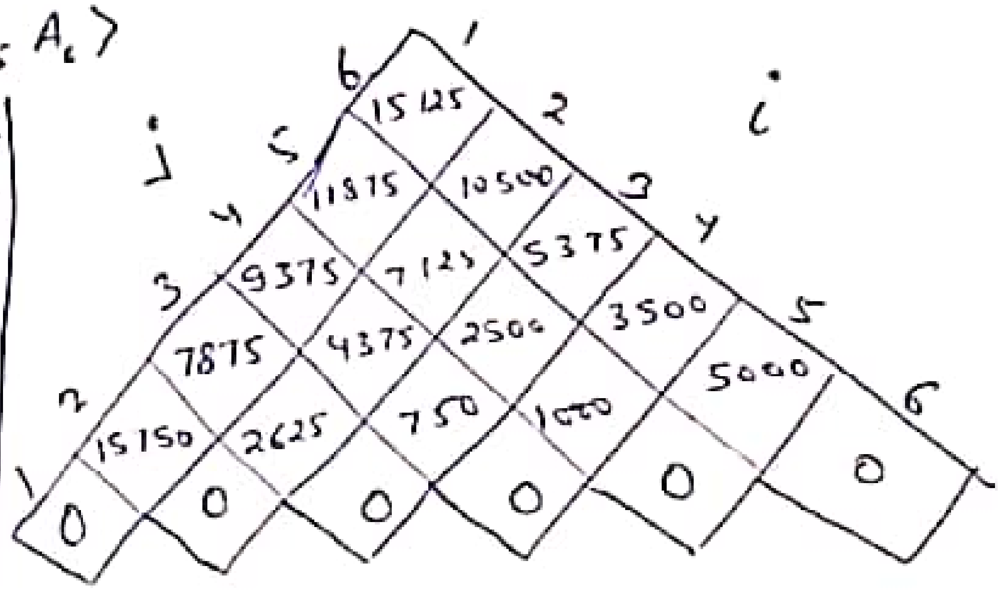
```
Print_OPTIMAL_PARENS(s, i, j)
{
 if i = j
      Print "A_i"
 else Print "("
      Print_OPTIMAL_PARENS(s, i, s[i,j])
      Print_OPTIMAL_PARENS(s, s[i,j]+1, j)
      Print ")"
```

Ex Problem CLRS 15.2 page 337, 2nd edition.

m

$\langle A_1\ A_2\ A_3\ A_4\ A_5\ A_6 \rangle$

A_1	30×35
A_2	35×15
A_3	15×5
A_4	5×10
A_5	10×20
A_6	20×25



Chain Cost

$\langle A_1 A_2 \rangle = 30 \times 35 \times 15$

$= 15750$

Cost

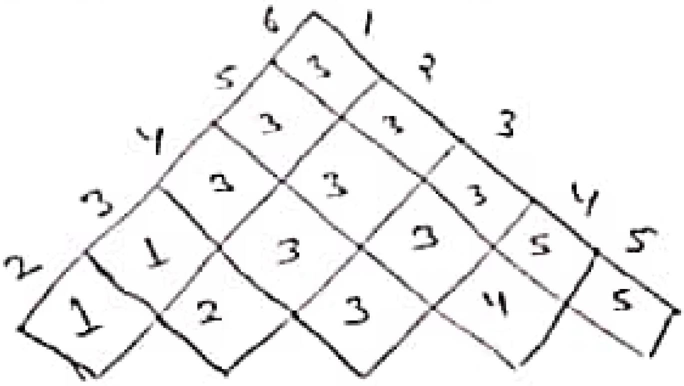
$\langle A_1 A_2 A_3 \rangle = ((A_1)\underbrace{(A_2 A_3)}_{35 \times 5}) = 35 \times 15 \times 5 + 30 \times 35 \times 5 = 7875$

$= (\underbrace{(A_1 A_2)}_{30 \times 15}(A_3)) = 30 \times 35 \times 15 + 30 \times 15 \times 5 = 18000$

$((A_1)(A_2 A_3))$ have better cost & here $k = 1$ corresponding to which cost is minimum

Similarly

S



S

Optimal cost of product operation is 15125

Optimal Parenthesis is

$((A_1 (A_2 A_3))((A_4 A_5) A_6))$