Unit 5 Activity: Dynamic Programming

As you may recall, Matrix Chain Multiplication is the problem of finding the optimal order by which to evaluate a product of matrices. Expressed more formally, given a set of matrices, {A_1, A_2, A_3, … , A_n}, find the optimal arrangement of matrices such that the time cost of performing the multiplication is minimized.

As stated above, the time complexity of a matrix product is dependent on the order in which the matrices are multiplied. Because multiplication is an associative operation, we can switch the order of the matrices in the multiplication and still achieve the same result. For example:

$$(A_1 * (A_2 * A_3) *A_4)$$

is equivalent to:

$$(A_1 * A_2 * (A_3 * A_4))$$

Matrix Chain Multiplication is not the problem of actually evaluating the product, an operation that is trivial, but instead, it is about determining which arrangement of parenthesis yields an operation that has the smallest possible time complexity. There exists a recursive solution to this problem, but it involves repeating the same operation multiple times, producing a sub-optimal solution. Given the fact that Dynamic Programming allows us to store the results of our operations and access them when necessary, we have a way of solving the same problem with drastic improvements in time complexity due to the lack of repeated operations.

Implement the dynamic programming algorithm, as discussed in the lecture, using the dynamic programming approach, and compare the runtime improvements in results over the regular method for matrix multiplication.