Matrix multiplication: dynamic programming

Amanda House, Matthew Gharrity

July 4, 2017

Problem statement

Given a matrix multiplication $A_1 \times A_2 \times \cdots \times A_s$, where A_i has dimensions $m_i \times n_i$, and $n_i = m_{i+1}$ for $i \in [1, s)$, find the minimal number of integer multiplications needed to evaluate the matrix product. You may place parentheses anywhere.

Approach

Note the cost of multiplying $A_{m\times n} \times A_{a\times b}$, where $n = a$, is on the order of $m \times n \times b$.

First, consider the base case. If there is one matrix, evaluating the matrix product of the matrix and itself requires zero multiplications.

Then, define and solve the subproblem. Consider a set of x matrices. Assume the optimal answer for every contiguous sequence of length 0 to $x-1$ matrices is already known. With this information, find the optimal for all x matrices. The approach can be thought of as finding the boundary i which minimizes the costs of multiplying matrices within the group matrix 1 up to matrix i (exclusive), within the group i through x (inclusive), and multiplying at the boundary between the two.

Thus, the dynamic programming solution is:

Let $dp[l][r]$ be the optimal number of integer multiplications for $\prod\limits_{i\in[l,r)} A_i$.

$$dp[l][r] = \begin{cases} 0 & \text{if } l = r - 1 \\ \min\limits_{l<i<r} (n_l \cdot m_i + dp[l][i] + dp[i][r]) & \text{otherwise} \end{cases}$$

Lastly, consider the order in which to fill the array dp. For example, when solving for $dp[0][3]$, the value for $dp[2][3]$ must be known. This requires the algorithm to proceed diagonally in the array. In index arithmetic, this means the algorithm must solve for all l and r that have a difference of 1, then all l and r with a difference of 2, and so on.

Solution

The code is in MatrixMult.java.