CS 161 (Stanford, Fall 25) Section 6 Extra

1 Graph Algorithms

The Floyd–Warshall algorithm runs in $O(n^3)$ time on graphs with n vertices and m edges, whether or not the input graph contains a negative cycle. Modify the Floyd–Warshall algorithm so that it will detect negative cycles and stop faster. Specifically, design an algorithm with the following behavior:

- If the input graph contains *no* negative cycle, the algorithm should compute all-pairs shortest paths in the usual $O(n^3)$ time.
- If the input graph *does* contain a negative cycle, the algorithm should detect this in only $O(mn)$ time.

Hint: Can we try creating a new vertex and some associated edges, then use a different algorithm that runs in time O(mn)?

2 Rod Cutting

Suppose we have a rod of length k, where k is a positive integer. We would like to cut the rod into integer-length segments such that we maximize the *product* of the resulting segments' lengths. Multiple cuts may be made. For example, if $k = 8$, the maximum product is 18 from cutting the rod into three pieces of length $3, 3$, and 2. Write an algorithm to determine the maximum product for a rod of length k.

3 Matrix Chain Multiplication

Consider a scenario in which we would like to multiply a lot of matrices, with matrix A_i's dimensions given as $p_{i-1} \times p_i$. The goal is to determine the most efficient order in which to multiply the matrices, so that the total number of scalar multiplications is minimized. Note that matrix multiplication is associative, so the result does not depend on how the matrices are parenthesized, but the *number of operations* does.

Note that if A is a $p \times q$ matrix and B is a $q \times r$ matrix, then their product AB is a $p \times r$ matrix. Computing this product requires

$$p \cdot q \cdot r$$

scalar multiplications, since each of the pr entries in the result is obtained by taking a dot product of length q.

Example. Suppose we have only the first three matrices A_1 (10×30), A_2 (30×5), and A_3 (5×60), so $p = [10, 30, 5, 60]$. There are two possible parenthesizations:

$$(A_1 A_2) A_3 \quad \text{and} \quad A_1 (A_2 A_3).$$

For the first, the cost is:

$$(10 \times 30 \times 5) + (10 \times 5 \times 60) = 1500 + 3000 = 4500.$$

For the second, the cost is:

$$(30 \times 5 \times 60) + (10 \times 30 \times 60) = 9000 + 18000 = 27000.$$

Thus, the optimal order is $(A_1 A_2) A_3$.

Give the recurrence relation for performing dynamic programming on this problem, and also give pseudocode.