Practice Problems 6

Problem 1. Matrix Chain Multiplication. You are given five matrices

$$A_1 : 30 \times 5, \qquad A_2 : 5 \times 60, \qquad A_3 : 60 \times 2, \qquad A_4 : 2 \times 20, \qquad A_5 : 20 \times 10.$$

Recall that multiplying an $a \times b$ matrix by a $b \times c$ matrix requires abc scalar multiplications. Let $\mathrm{OPT}(i, j)$ denote the minimum number of scalar multiplications needed to compute

$$A_i A_{i+1} \cdots A_j.$$

Also let $\mathrm{split}(i, j)$ denote a value of k at which the optimal recurrence for $\mathrm{OPT}(i, j)$ attains its minimum.

(a) Fill in the following two DP tables for $\mathrm{OPT}(i, j)$ and $\mathrm{split}(i, j)$. Entries below the diagonal do not need to be filled. Think about how the optimal parenthesization can be obtained by splitting the matrix chain at some position k.

$\mathrm{OPT}(i,j)$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$i=1$	0				
$i=2$	\	0			
$i=3$	\	\	0		
$i=4$	\	\	\	0	
$i=5$	\	\	\	\	0

$\mathrm{split}(i,j)$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$i=1$	–				
$i=2$	\	–			
$i=3$	\	\	–		
$i=4$	\	\	\	–	
$i=5$	\	\	\	\	–

(b) Using your $\mathrm{split}(i, j)$ values, give an optimal parenthesization of

$$A_1 A_2 A_3 A_4 A_5$$

and state the minimum number of scalar multiplications.

(c) Let $p_0, p_1, \ldots, p_5$ be the matrix dimensions, so that A_i has dimensions $p_{i-1} \times p_i$. Give the base case and recurrence for $\mathrm{OPT}(i, j)$. Explain in what order the subproblems should be computed in a bottom-up algorithm, and state its time and space complexities for a chain of n matrices.

(d) Let $P(n)$ be the number of different valid ways to fully parenthesize a product of n matrices while keeping their order fixed. Give a recurrence for $P(n)$ with an appropriate base case, and use it to compute $P(5)$.

Problem 2. Randomized Algorithms and Amplification. A randomized algorithm may use random choices during its execution. Its success probability is taken over these random choices, for each fixed input.

(a) Consider the following two algorithms:

- Algorithm A always produces a correctly sorted array, but its running time depends on randomly selected pivots.
- Algorithm B always terminates within a fixed time bound, but may return an incorrect answer depending on its random choices.

Which algorithm is *Las Vegas* and which is *Monte Carlo*? Briefly explain.

(b) Consider the problem of deciding whether an n-bit string contains at least one 1. An algorithm always outputs `YES`, without reading its input. If inputs are chosen uniformly from $\{0,1\}^n$, what fraction of inputs does it answer correctly? Does this imply that it has success probability at least $2/3$ for *every* input? Explain.

(c) Suppose a randomized algorithm answers a `YES/NO` question correctly with probability at least $2/3$ on every fixed input. Run it three times using independent randomness and output the majority answer. What is the probability that the majority answer is correct? Give a lower bound and show your calculation.

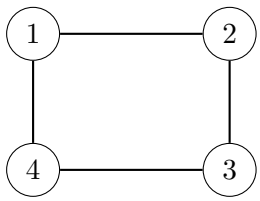
(d) Generalize part (c) to an odd number t of independent runs. Show that the majority-vote error probability is at most $e^{-t/18}$. You may use the following inequality for independent Bernoulli random variables $X_1, \ldots, X_t$:

$$\Pr\left[\sum_{i=1}^{t} X_i \le \mathbb{E}\left[\sum_{i=1}^{t} X_i\right] - a\right] \le e^{-2a^2/t} \qquad (a > 0).$$

How large should t be to make the error probability at most δ, where $0 < \delta < 1$?

Problem 3. Karger's Randomized Minimum Cut. Recall that a cut of an undirected graph partitions its vertices into two nonempty sets. The *size* of a cut is the number of edges crossing between the two sets. Karger's algorithm repeatedly selects a uniformly random edge and contracts it until only two super-vertices remain. Self-loops are removed, but parallel edges are retained.

(a) Consider the following undirected cycle graph. Give a minimum cut by specifying the two vertex sets, and state the size of the minimum cut.



(b) In the graph from part (a), suppose Karger's algorithm first contracts edge $(1,2)$, and then contracts the edge between the super-vertex $\{1,2\}$ and vertex 3. Describe the final multigraph. How many parallel edges remain between its two super-vertices, and what cut does this execution output?

(c) Now consider an arbitrary connected, undirected graph with $n \geq 2$ vertices and minimum cut size k. Fix one particular minimum cut. Show that a single run of Karger's algorithm finds this cut with probability at least

$$\frac{2}{n(n-1)}.$$

Hint: Condition on never contracting an edge of the fixed cut. When r super-vertices remain, what can you say about their degrees and the probability that the next contraction crosses this cut?

(d) Suppose we run Karger's algorithm independently t times and return the smallest cut found. Give an upper bound on the probability that none of the runs finds a minimum cut. How many runs suffice to find a minimum cut with probability at least $1 - 1/n$? You may use $1 - x \leq e^{-x}$.

Problem 4. Polynomial Identity Testing. You have black-box access to two polynomials $f(x)$ and $g(x)$ with real coefficients, each of degree at most d. You may query the value of either polynomial at any chosen real number. The goal is to determine whether $f(x)$ and $g(x)$ are identical polynomials. Let $h(x) = f(x) - g(x)$.

For this problem, you may use the fact that a nonzero polynomial of degree at most d has at most d distinct real roots. Let $S = \{1, 2, \ldots, N\}$, where $N > d \geq 1$.

(a) Consider $f(x) = x^2 - 6x + 8$ and $g(x) = 0$. Evaluate $h(2)$, $h(3)$, and $h(4)$. Why does finding a point where $f(x) = g(x)$ not necessarily mean that the two polynomials are identical?

(b) Consider the following randomized test: choose r uniformly at random from S, query $f(r)$ and $g(r)$, and output `DIFFERENT` if their values differ; otherwise output `EQUAL`. Show that this algorithm is always correct when $f = g$. When $f \neq g$, what is an upper bound on the probability that it incorrectly outputs `EQUAL`?

(c) Repeat the test from part (b) independently t times, outputting `DIFFERENT` if any test finds unequal values. Bound the probability of incorrectly outputting `EQUAL` when $f \neq g$. If $N \geq 2d$, how many independent tests suffice to make this error probability at most $1/16$?

(d) Suppose instead that the t query points are selected uniformly *without replacement* from S. If the nonzero polynomial h has exactly r distinct roots in S, give the probability that all t queries return zero. Compare this probability with independent sampling *with replacement*, and explain what happens when $t > d$.