Partner 1
Partner 2
Partner 3

CSCI 1550 / 2540
February 20, 2025

Homework 2

Due: March 6, 2025
Remember to show your work for each problem to receive full credit.

Problem 1 (40 Points)

In this exercise, we design a randomized algorithm for the following packet routing problem. We are given a network that is an directed connected graph G, where nodes represent processors and the edges between the nodes represent wires. We are also given a set of N packets to route. For each packet we are given a source node, a destination node, and the exact route (path in the graph) that the packet should take from the source to its destination. (We may assume that there are no loops in the path.) In each time step, at most one packet can traverse any single edge. A packet can wait at any node during any time step, and we assume unbounded queue sizes at each node.

A *schedule* for a set of packets specifies the timing for the movement of packets along their respective routes. That is, it specifies which packets should move and which should wait at each time step. Our goal is to produce a schedule for the packets that tries to minimize the total time and the maximum queue size needed to route all the packets to their destinations.

(a) The dilation d is the maximum distance traveled by any packet. The congestion c is the maximum number of packets that must traverse a single edge during the entire course of the routing. Argue that the time required for any schedule should be at least $\Omega(c + d)$.

(b) Consider the following unconstrained schedule, where many packets may traverse an edge during a single time step. Assign each packet an integer delay chosen randomly, independently, and uniformly from the interval $[1, \lceil \frac{\alpha c}{\log(Nd)} \rceil]$, where $\alpha \leq 1$ is a constant. A packet that is assigned a delay of x waits in its source node for x time steps; then it moves on to its final destination through its specified route without ever stopping. Prove that with probability $1 - (Nd)^{-\frac{1}{3\alpha}}$ no more than $\frac{2\log(Nd)}{\alpha}$ packets use a particular edge E, at a particular step t.

(c) Again using the unconstrained schedule of part (b), show that there exists a constant α such that the probability that more than $O(\log(Nd))$ packets pass through any edge at any time step is at most $\frac{1}{Nd}$. [Hint: argue that since there are N packets, and each packet traverses $\leq d$ edges, we need to apply union bound over no more than Nd events.]

(d) Use the unconstrained schedule to devise a simple randomized algorithm that, with high probability (in N), produces a schedule of length $O(c + d\log(Nd))$ using queues of size $O(\log(Nd))$ and following the constraint that at most one packet crosses any single edge per time step.

Problem 2 (20 points)

In many wireless communication systems, each receiver listens on a specific frequency. The bit $b(t)$ sent at time t is represented by a 1 or -1. Unfortunately, noise from other nearby communications can affect the receiver's signal. A simplified model of this noise is as follows. There are n other senders, and the ith has strength $p_i \leq 1$. At any time t, the ith sender is also trying to send a bit $b_i(t)$ that is represented by 1 or -1. The receiver obtains the signal $s(t)$ given by

$$s(t) = b(t) + \sum_{i=1}^{n} p_i b_i(t)$$

If $s(t)$ is closer to 1 than -1, the receiver assumes that the bit sent at time t was a 1 otherwise, the receiver assumes that it was a -1.

Assume that all the bits $b_i(t)$ can be considered independent, uniform random variables. Give a Chernoff bound to prove the probability that the receiver makes an error in determining $b(t)$ is less than or equal to following quantity

$$\exp\left(\frac{-1}{2\sum_{i=1}^{n} p_i^2}\right).$$

Problem 3 (35 points)

Bob is facing a very challenging math question in **CSCI1550/2540**. Even if this math question is very hard, it has a simple binary answer $Y \in \{0,1\}$ (both answers are equally likely). Bob asks for help from n fellow math-loving friends (numbered from 1 to n), and each of them provides an answer to this math question. However, as this math question is very hard, there is no guarantee that these answers are the same. In particular, friend i provides an answer $X_i \in \{0,1\}$, for $i = 1, \ldots, n$. Bob knows the expertise of each friend; in particular, he knows that for each $i = 1, \ldots, n$, we have that:

$$X_i = \begin{cases} Y & \text{with probability } p_i > 1/2 \\ 1 - Y & \text{with probability } 1 - p_i \end{cases}$$

Formally, $X_1, \ldots, X_n$ are random variables function of Y. Bob also assumes that these friends won't collaborate with each other; that is, given Y, the random variables $X_1, \ldots, X_n$ are independent.

Bob wants to use a function $f(X_1, \ldots, X_n) : \{0,1\}^n \to \{0,1\}$ to obtain the final answer to the hard math problem. He would like to minimize the error that the function f makes a mistake, i.e., he wants to minimize:

$$\Pr(f(X_1, \ldots, X_n) \neq Y) \tag{1}$$

If a function f minimizes (1), we say that f is optimal. Let $\vec{X} = (X_1, \ldots, X_n)$.

(a) For $y \in \{0,1\}$, let

$$g(\vec{x}, y) = \Pr(\vec{X} = \vec{x} | Y = y) = \prod_{i:x_i=y} p_i \prod_{i:x_i=1-y} (1-p_i) = \exp\left(\sum_{i:x_i=y} \log p_i + \sum_{i:x_i=1-y} \log(1-p_i)\right)$$

Show that a function f is optimal if and only if for any $\vec{x} \in \{0,1\}^n$, it holds that

$$f(\vec{x}) = \arg\max_{y \in \{0,1\}} g(\vec{x}, y)$$

(b) Bob considers a family of functions that is called weighted majority vote. That is, he wants to assign a different weight to the answer of the different friends, based on their competence. Let $\vec{w} = (w_1, \ldots, w_n) \in \mathbb{R}^n$. Given $\vec{w}$, we define:

$$f(\vec{x}; \vec{w}) = \begin{cases} 1 & \text{if } \sum_{i=1}^n X_i w_i \geq \sum_{i=1}^n (1 - X_i) w_i \\ 0 & \text{otherwise} \end{cases}$$

Let $\vec{w}^* = (w_1^*, \ldots, w_n^*)$, where $w_i^* = \ln\left(\frac{p_i}{1-p_i}\right)$. Use the answer to question a. to show that the function $f(\bullet; \vec{w}^*)$ is optimal.

(c) Let $f^*(\bullet) = f(\bullet; \vec{w}^*)$. Use Hoeffding's bound to show an upper bound on the error probability

$$\Pr(f^*(X_1, \ldots, X_n) \neq Y)$$

Hint: Show that if $f^*(X_1, ..., X_n) \neq Y$, then the sum of weights w_i, whose corresponding answer X_i is correct, is less than or equal to $\frac{1}{2}\sum_{i=1}^{n} w_i$.

(d) Suppose that for each $i = 2, \ldots, n$, we have that $p_i = 0.9$, and let $p_1 \to 1$. What happens to the upper bound computed in question c.? Is this upper bound useful or not in this scenario?

Problem 4 (25 points)

This problem demonstrate the difference between additive and multiplicative error deviation bounds.

Let $G = (V, E)$ be an undirected graph, $V = \{1, \ldots, n\}$ and $E \subseteq \{\{i, j\} : i, j \in V \text{ and } i \neq j\}$. We know the number of vertices $|V| = n$. We want to estimate the fraction of pairs $\{i, j\}$ of connected by an edge, $\rho = m/\binom{n}{2}$, where $m = |E|$. We can query an oracle, that given a pair $\{i, j\}$, tells us if i and j are connected by an edge in the graph G, i.e. whether $\{i, j\} \in E$ or not.

(a) **Additive error bound:** Use the Hoeffding's bound to bound the number of queries of pairs, chosen uniformly at random, needed to estimate ρ within an ϵ additive error, i.e. output $\tilde{\rho}$ such that $|\tilde{\rho} - \rho| \leq \epsilon$ with probability at least $1 - \delta$.

(b) **Multiplicative error bound:**

1. Assume that you given a lower bound d on the fraction ρ. If this lower bound is true, how many random queries are needed to find an estimate $\tilde{\rho}$ that satisfies an ϵ multiplicative error, i.e. $|\tilde{\rho} - \rho| \leq \epsilon\rho$, with probability at least $1 - \delta$?
2. Assume now that you don't have a lower bound of ρ. Design and analyze an algorithm that estimates ρ with a number of sample adjusted to the unknown ρ. [**Hint:** Assume first that $\rho > 1/4$, if the condition doesn't hold assume $\rho > 1/8$, etc. Remember to bound the total error probability.]
3. For which values of ρ is it better to just check all pairs?