CS49/Math59 Lab 7

This lab assignment is due **before the start of class** on Wednesday, November 25. Homework handed in during class but after I begin the lecture will be counted as late submissions. Some things to note:

- This is a **one week lab**.
- You may have one partner for this assignment, but are not required to. If you work with a partner, submit just one writeup.
- Aside from your partner, you should not discuss problems in detail with anyone. It's OK to discuss approaches at a high level. In fact, I encourage you to discuss general strategies. However, you should not reveal specific details of a solution, nor should you show your written solution to anyone else.
- Make sure your names are on your submission, and show your work to maximize partial credit.

This lab ties together several recent topics we've discussed in class, including the probabilistic method, communication complexity, randomized algorithms, and error-correcting codes.

Historically, there have been two models of randomized communication: the *public-coin* and the *private-coin* models. In the public-coin model, there is an infinitely long random string $R \in \{0,1\}^*$. Both players see this random string, and can use it to help decide what messages to send. In the private-coin model, Alice and Bob each have their own infinitely-long random string $R_A, R_B : \{0,1\}^*$ respectively. Alice uses R_A and Bob uses R_B, and neither player sees the other player's random string (just like they don't see each other's inputs). Randomized protocols are judged by two criteria: *cost* and *error*.

- The *cost* of a protocol $\mathcal{P}$, written $\text{cost}(\mathcal{P})$, is defined as the worst-case number of bits that get sent during the run of a protocol, taken over all input pairs and all possible choices for the random string(s).
- The *error* of a protocol $\mathcal{P}$, written $\text{err}(\mathcal{P})$, is defined as:

$$\max_{x,y} \Pr[\mathcal{P}(x,y) \neq f(x,y)] \ .$$

In this lab, you will examine the randomized communication complexity of the EQUALITY function. Recall that $\text{EQ} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is defined as $\text{EQ}(x,y) = 1$ iff $x == y$. This can be modeled as a communication problem by giving Alice x and Bob y. We've seen previously that Alice and Bob must exchange at least n bits total to compute EQ using a *deterministic* protocol. Things get easier if we allow *randomized* communication—there is an 8-bit, 1/100-error public-coin protocol for EQ.

1. Consider the following simple two-bit public-coin protocol $\mathcal{P}$ for EQUALITY. Alice and Bob use R to generate a random index $i \in \{1, \ldots, n\}$. Then, they exchange x_i and y_i and output 1 iff $x_i == y_i$. Protocol $\mathcal{P}$ is efficient, but has high error. Compute the error of $\mathcal{P}$.

2. Note that the previous protocol has *one-sided error*: if $x == y$, then the protocol always correctly outputs that x and y are equal. Use this observation to create a new protocol $\mathcal{P}'$ with error $\leq 1/4$. What is cost($\mathcal{P}'$)? (This protocol has good error, but is not terribly communication-efficient.)

3. In the lab exercises for week 10, you showed how to take a randomized algorithm $\mathcal{A}$ that takes one of N possible inputs and had 1/4 error and produce another randomized algorithm $\mathcal{A}'$ for the same problem that had 1/3 error and used only $O(\log \log N)$ random bits.

   Use this derandomization result together with the $O(1)$-bit, (1/100)-error public-coin protocol for EQ and show there exists a 1/3-error public-coin EQ protocol that uses only $O(\log n)$ random bits.

4. Using your answer to the previous problem, show that there exists a 1/3-error, private-coin protocol[1] Q for EQUALITY that has cost(Q) $= O(\log n)$.

5. The previous answer showed that it's possible to compute EQUALITY using private randomness and only $O(\log n)$ communication. However, this result has a drawback—the protocol itself is not explicitly constructed. In this problem, your task is to design an *explicit* $O(\log n)$-bit, 1/3-error protocol for EQUALITY.

   Hint: First, create an $O(\log n)$-bit, $(1-1/320)$-error protocol by using error-correcting codes. Your protocol should have one-sided error, erring only when $x \neq y$. Then, repeat this initial protocol several times to drive down the error. How many times do you need to repeat the protocol to achieve error 1/3?

6. **Attribution.** Did you get assistance on any of the problems on this assignment from anyone aside from me and/or your lab partner? For example, did you discuss any problems at a high level with other students? Did you accidentally stumble on solutions while doing a websearch on related material? If so, descibe the nature of the assistance here. (e.g. "We briefly discussed problem 1 with X,Y, and Z" or "We saw a solution on ⟨this website⟩ before finding our own solution") If you (and your partner) worked alone, please say so here.

7. **Lab Questionnaire.** (None of these questions will have an impact on your grade, this is to help provide the feedback I need to make the course the best it can be)

   (a) Approximately how many hours per partner did you spend on this lab?

   (b) How difficult did you find this lab? (enter a number 1-5, with 5 being very difficult and 1 being very easy)

   (c) Describe the biggest challenge you faced on this lab.

[1] In fact, this result doesn't hold just for EQUALITY. The derandomization result from lab during week 10 actually yields the following result: for any function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, if there is a public-coin, ε-error protocol that uses t bits of communication, then there is a $(\varepsilon + \delta)$-error, **private-coin** protocol for f that uses $t + O(\log(n/\delta))$ bits of communication.