Problem 1 (25 Points) - Distinct Elements

Consider again the distinct elements problem that we saw in class. We are given a sequence of elements $a_1, \ldots, a_n$ from our universe $U = \{0, 1, \ldots, 2^b - 1\}$ as a stream, possibly with repetitions, and we would like to know how many distinct elements are there in the sequence. Since we are in the streaming setting, we will make only one pass through the sequence above, and we have little memory.

We will now analyze a different algorithm for the distinct elements problem:

- Let $N > n$ be an integer
- Pick at random a function $h : U \to [N]$ from a strongly 2-universal family.
- Let $m := \min\{h(a_1), \ldots, h(a_n)\}$
- Output N/m

1. Suppose that $a_1, \ldots, a_n$ contains k distinct elements. Show that

$$\Pr[\text{ algorithm outputs a number } > 4k] \leq \frac{1}{4}$$

2. Suppose that $a_1, \ldots, a_n$ contains k distinct elements. Show that

$$\Pr[\text{ algorithm outputs a number } < k/4] \leq \frac{1}{4}$$

3. Assuming that $U = [\mathsf{poly}(n)]$, what is the memory requirement of the algorithm above?

Problem 2 (25 Points) - Hardness of Approximation

Given a graph G, an *independent set* is a set of vertices such that no two are neighbors. The Maximum Independent Set (MIS) problem is a famous NP-complete problem.

Interestingly, the complement of an independent set is a vertex cover, so the complement of the MIS is the minimum vertex cover. We've seen (twice) how to get a two-approximation for vertex cover. Even though it is complementary, the situation for MIS is much worse.

Early in the study of approximation hardness, MIS was shown to be MAX-SNP-hard, meaning there is some constant to within which it *cannot* be approximated (unless P = NP).

Suppose one has an α-approximation algorithm for MIS. Consider the following "graph product" operation for a graph G. Create a distinct copy G_v of G for each vertex v of G. Then connect up the copies as follows: if (u, v) is an edge of G, then connect every vertex in G_u to every vertex in G_v.

(a) Prove that if there is an independent set of size k in G, then there is an independent set of size k^2 in the product graph.

(b) Prove that given an independent set of size s in the product graph, one can find an independent set of size $\sqrt{s}$ in G.

(c) Conclude from the MAX-SNP-hardness of MIS that MIS has *no* constant-factor approximation (unless P = NP).

Problem 3 (25 Points) - Matrix Multiplication and Determinant

Given a matrix $A \in \mathbb{Q}^{N \times N}$ where $N = 2^k$, prove that one can compute $\det(A)$ in time $O(N^\omega)$, where ω is the matrix multiplication exponent. You can assume that any matrix that you need to invert in the process is invertible.

Optional question: how would you remove the assumption given above?

Problem 4 (25 Points) - Consensus with crash failures

We will now analyze the consensus problem in the synchronous model with crash failures. In this model, there are n processors, which can communicate with one another by sending messages, and which are synchronized (thus communication happens in rounds).

We will assume that there will be at most f processors that will crash, where $f \leq n$. Once a processor crashes, it will not be able to send or receive messages anymore. While the processors are alive, they will not be malicious, and will follow the proposed protocol.

There are two types of crashes: **clean** and **general**.

- In a clean crash, if a processor crashes at round t, then it will not send or receive any messages at round t or later.
- In a general crash, if a processor p_j crashes at round t, then an arbitrary subset of the messages that p_j sends at round t will be delivered to the other processors.

The setup of the consensus problem is as follows:

- **Input:** Each processor i has an input value $x_i \in \mathbb{N}$.
- **Output:** Each processor i should output a value y_i such that:
  - **Agreement:** All alive processors output the same value.
  - **Validity:** If *all processors* have the same input value v, then $y_i = v$ for any *alive* processor.
  - **Termination:** All alive processors must output a value at the end.

Part 1 (10 points): give a protocol for consensus in the synchronous model with clean crashes, using one round of communication.

Part 2 (15 points): when there are general crashes, the processors can no longer reach consensus in one round. Hence, we will need a more resilient protocol to reach consensus.

Consider the following protocol for consensus under general crashes, where f is the number of crash failures:

Protocol: Consensus with at most f crash failures

Code for Processor p_i

1. Initially, set $V = \{x_i\}$
2. For round $k = 1, 2, \ldots, f + 1$ do:
   - Send $U := \{v \in V \mid p_i \text{ has not yet sent } v\}$ to all processors
   - Receive U_j from processor p_j
   - Set $V := V \cup \bigcup_{j \neq i} U_j$
   - if $k = f + 1$, output $\min V$

To prove correctness of the protocol above, let $V_i^{(k)}$ be the set V at the end of round k for processor p_i. Prove the following lemma (from which correctness easily follows):

1. In every execution of the protocol, $V_i^{(f+1)} = V_j^{(f+1)}$ for any two alive processors p_i and p_j.