CMPSCI 611 FALL '17: HOMEWORK 4
DUE 8PM, FRIDAY NOVEMBER 10TH

- Homework may be completed in group of size with at most four students. You're not allowed to use material from the web or talk about the homework with anybody outside your collaboration group (aside from the lecturer or TA.)
- Solutions should be typed and uploaded as a pdf to `gradescope.com`.
- To get full marks, answers must be sufficiently detailed, supported with rigorous proofs (of both correctness and running time analysis). Faster algorithms will typically get more marks than slower algorithms.

Question 1. Suppose we have an unsorted list of n distinct values. Consider the following sorting algorithm: if the list is currently unsorted, pick the first two adjacent entries in the list, $[\ldots, a, b, \ldots]$ such that $a > b$ and swap the position of a and b. Repeat until the list is sorted. If the list is originally ordered randomly, what is the expected number of swaps necessary before the list is in sorted order.

Question 2. Suppose your friend designs a randomized algorithm that tries to compute the size of the minimum vertex cover of a graph that has n nodes. With probability $2/3$ the algorithm returns the correct answer. Otherwise the algorithm returns an incorrect answer that may be too large or too small. By running the algorithm multiple times, how can you determine the size of the minimum vertex cover with probability at least 0.999? You may assume that the answer returned in each repetition of the algorithm is independent from the answers of the other repetitions of the algorithm. Your solution should involve running the algorithm as few times as possible. If, when algorithm didn't return the correct value, it was equally likely to output any value in $\{0, 1, 2, \ldots, n\}$ that is incorrect, how can you improve your solution on the assumption that n is very large?

Question 3. Design randomized algorithms whose expected running time is polynomial in n and m for the following two problems:

(1) For an undirected graph $G = (V, E)$ with n nodes and m edges, define the induced subgraph on $X \subset V$, to be the graph $G[X]$ whose node set is X and whose edge set consists of all edges of G for which both endpoints are in X. Design and analyze a randomized algorithm that, given G and $1 \leq k \leq n$, finds $X \subset V$ such that $G[X]$ has exactly k nodes and at least $\frac{mk(k-1)}{n(n-1)}$ edges.

(2) Given an undirected graph with n nodes and m edges, design and analyze a randomized algorithm that finds a cut of size at least $m/2$.

Question 4. Consider two strings $a = a_0 a_1 \ldots a_{n-1}$ and $b = b_0 b_1 \ldots b_{n-1}$ where each a_i and b_i is a value in the range $0, 1, \ldots, m-1$. In this question, we consider two ways to randomly map (or "fingerprint") a and b to values $h(a)$ and $h(b)$ that can be expressed in a small number of bits with the property that if $a \neq b$ then $h(a) \neq h(b)$ with probability at least $1 - 1/n$. Define

$$f_a(x) = \sum_{i=0}^{n-1} a_i x^i \quad \text{and} \quad f_b(x) = \sum_{i=0}^{n-1} b_i x^i$$

(1) *First Method:* Let $h(a) = f_a(r) \bmod p$ and $h(b) = f_b(r) \bmod p$ where r is randomly chosen from $\{1, 2, 3, \ldots, t\}$ and p is a prime. What's the smallest values you can pick for t and p

if you want to ensure $a \neq b$ implies $\mathbb{P}[h(a) \neq h(b)] \geq 1 - 1/n$. Remember to prove your answer.

(2) *Second Method:* Let $h(a) = f_a(r) \bmod p$ and $h(b) = f_b(r) \bmod p$ where p is a prime randomly chosen from the first t prime numbers. What's the smallest values you can pick for r and t if you want to ensure $a \neq b$ implies $\mathbb{P}[h(a) \neq h(b)] \geq 1 - 1/n$. Remember to prove your answer.

How would you modify the first method if you wanted instead to check whether a could be reordered to make b? Prove the correctness of the modification.

Question 5. In this question, we revisit Karger's Min-Cut algorithm. Let $G = (V, E)$ be an unweighted graph on n nodes where λ is the size of the minimum cut.

(1) By considering the analysis of Karger's Min-Cut algorithm, prove that there are at most n^2 cuts in G of size λ. Identify a graph with min cut of size two in which there are $\Omega(n^2)$ cuts of size two.

(2) Generalize the analysis of Karger's Min-Cut algorithm to show that for any $\alpha > 0$, there are $O(n^{2\alpha})$ cuts of size $\alpha\lambda$ where λ is the size of the minimum cut.

(3) We say a weighted graph $G' = (V, E')$ is an ϵ-sparsifier for G if the size of any cut (A, B) in G is within a $1 + \epsilon$ factor of the size of the cut (A, B) in G'. Note that the size of a cut in a weighted graph is the total weight of the edges across the cut. Suppose that the size of the minimum cut in G is $\lambda = \sqrt{n}$. Prove that there exists an ϵ-sparsifier for G that has $O(\epsilon^{-2} n^{3/2} \log n)$ edges. **Hint:** For some value of p, consider deleting each edge in G with probability $1 - p$ and giving weight $1/p$ to the remaining edges.