CS 170 Homework 13

Due **5/2/2022, at 10:00 pm (grace period until 11:59pm)**

1 Study Group

List the names and SIDs of the members in your study group. If you have no collaborators, you must explicitly write "none".

2 Duplicate Streams

We have two streams of bits, $A = a_1, a_2, \ldots, a_n$ and $B = b_1, b_2, \ldots, b_n$ (that is, each a_i, b_i is either 0 or 1). We want to design an algorithm that streams A, then streams B, and we want to detect if $A = B$ (that is,$a_1 = b_1, a_2 = b_2, \ldots$ are all true) or not. Your algorithms know when A ends and B begins, and you may assume that the index of each bit is provided to the algorithm along with the bit (i.e., it doesn't need to spend additional memory keeping track of the current index).

(a) Give an algorithm using $O(1)$ bits of memory that outputs "True" if $A = B$ and "False" if A, B differ in an odd number of positions (if A, B differ in an even number of positions, we don't care what it outputs).

(b) A hash function family from $[n]$ to $[m]$, is k-wise independent if for any k distinct inputs $x_1, x_2, \ldots x_k$ to the hash function and h chosen uniformly at random from this family, $h(x_1), h(x_2), \ldots, h(x_k)$ is equally likely to be any of the m^k possible k-tuples of values in $[m]$.

Give a streaming algorithm using a hash function sampled from a k-wise hash function family from $[n]$ to $[m]$ (for an m of our choice) and $O(1)$ additional bits of memory that outputs "True" if $A = B$ and outputs "False" with probability at least $1/2$ if A, B differ in at most k positions. (Hint: Use your approach in part a on a subset of the stream).

(c) Give a randomized algorithm using $O(\frac{n}{k} \log n)$ bits of memory that outputs "True" if $A = B$ and outputs "False" with at least constant probability if A, B differ in at least k positions.

(d) Suppose it takes $O(k \log n)$ bits of memory to store a hash function from a k-wise independent family from $[n]$ to $[m]$. By combining the previous two parts, what memory requirement do we get for distinguishing $A = B$ and $A \neq B$ with at least constant probability, regardless of how many positions they differ in?

3 Multiway Cut

In the multiway cut problem, we are given a graph $G(V, E)$ with k special vertices $s_1, s_2 \dots s_k$. Our goal is to find the smallest set of edges F which, when removed from the graph, disconnect the graph into at least k components, where each s_i is in a different component. When $k = 2$, this is exactly the min s-t cut problem, but if $k \geq 3$ the problem becomes NP-hard.

Consider the following algorithm: Let F_i be the set of edges in the minimum cut with s_i on one side and all other special vertices on the other side. Output F, the union of all F_i. Note that this is a multiway cut because removing F_i from G isolates s_i in its own component.

(a) Explain how each F_i can be found in polynomial time.

(b) Let F^* be the smallest multiway cut. Consider the components that removing F^* disconnects G into, and let C_i be the set of vertices in the component with s_i. Let F_i^* be the set of edges in F^* with exactly one endpoint in C_i. How many different F_i^* does each edge in F^* appear in? Which is larger: F_i and F_i^*?

(c) Using your answer to the previous part, show that $|F| \leq 2|F^*|$. (Challenge: How could you modify this algorithm to output F such that $|F| \leq (2 - \frac{2}{k})|F^*|$?)

(As an aside, consider the minimum k-cut problem, where we want to find the smallest set of edges F whose removal disconnects the graph into at least k components. The following greedy algorithm for minimum k-cut gets a $(2 - \frac{2}{k})$-approximation: Initialize F to the empty set. While $G(V, E - F)$ has less than k components, find the minimum cut within each component of $G(V, E - F)$, and add the edges in the smallest of these cuts to F. Showing this is a $(2 - \frac{2}{k})$-approximation is fairly difficult.)

4 $\sqrt{n}$ coloring

(a) Let G be a graph of maximum degree Δ. Show that G is $(\Delta + 1)$-colorable.

(b) Suppose G is a 3-colorable graph. Let v be any vertex in G. Show that the graph induced on the neighborhood of v is 2-colorable. *Clarification: the graph induced on the neighborhood of v refers to the subgraph of G obtained from the vertex set V' comprising vertices adjacent to v (but not v itself) and edge set comprising all edges of G with both endpoints in V'.*

(c) Give a polynomial time algorithm that takes in a 3-colorable n-vertex graph G as input and outputs a valid coloring of its vertices using $O(\sqrt{n})$ colors. Prove that your algorithm is correct and also analyze its runtime.
Hint: think of an algorithm that first colors "high-degree" vertices and their neighborhoods, and then colors the rest of the graph. The previous two parts might be useful.

5 One-Sided Error and Las Vegas Algorithms

An RP algorithm is a randomized algorithm that runs in polynomial time and always gives the correct answer when the correct answer is 'NO', but only gives the correct answer with probability greater than $1/2$ when the correct answer is 'YES'.

(a) Prove that every problem in RP is in NP (i.e., show that $RP \subseteq NP$). *Hint: it may be helpful to view a randomized algorithm $R(x)$ as a deterministic algorithm $A(x, r)$ where x is the input and r is the result of the 'coin flips' which the algorithm uses for its randomness.*

(b) A *Las Vegas Algorithm* is a random algorithm which always gives the right solution, but whose runtime is random. A ZPP algorithm is a Las Vegas algorithm which runs in expected polynomial time (ZPP stands for Zero-Error Probabilistic Polytime). Prove that if a problem has a ZPP algorithm, then it has an RP algorithm. *Hint: Since we have not told you anything about the structure of the ZPP algorithm, your RP algorithm can only use it by running it somehow. What can you do to make sure the runtime is bounded? Use Markov's inequality.*