| | |
|---|---|
| **6.896 Sublinear Time Algorithms** | March 20, 2007 |
| Homework 1 | |
| *Lecturer: Ronitt Rubinfeld* | *Due Date: April 5, 2007* |

Don't turn these in:

1. You are given an approximation scheme $\mathcal{A}$ for $f$ such that $Pr[\frac{f(x)}{1+\epsilon} \leq \mathcal{A}(x) \leq f(x)(1+\epsilon)] \geq 3/4$, and $\mathcal{A}$ runs in time polynomial in $\epsilon, |x|$. Construct an approximation scheme $\mathcal{B}$ for $f$ such that $Pr[\frac{f(x)}{1+\epsilon} \leq \mathcal{B}(x) \leq f(x)(1+\epsilon)] \geq 1-\delta$, and $\mathcal{B}$ runs in time polynomial in $\frac{1}{\epsilon}, |x|, \log\frac{1}{\delta}$.

2. In class, we gave a property tester for connectivity that required that we fail graphs for which $\epsilon dn$ edges need to be added in order to connect it. We mentioned that it might make more sense to require we fail graphs for which $\epsilon dn$ edges need to be changed in order to turn it into a connected graph with degree at most $d$. In this problem, we will prove the main lemma, which combined with essentially the same analysis from lecture, would give the latter tester.

   Prove the following: If a graph $G$ is $\epsilon$-far from the class of $n$-vertex, degree bound $d \geq 2$ graphs, then $G$ has at least $\epsilon dn/8$ connected components, each containing less than $8/(\epsilon d)$ vertices.

3. In class we gave an approximation algorithm for graphs in which the weights on each edge were integers in the set $\{1..w\}$. Show that one can get an approximation algorithm when the weights can be any value in the range $[1..w]$ (it is ok to get a slightly worse running time).

Turn in your solution to *each* of the following problems on a separate sheet of paper, with your name on each one.

1. A *tournament* digraph $G$ is one in which for all nodes $u, v$, $G$ contains exactly one of the directed edges $(u, v)$ and $(v, u)$. Assume $G$ is represented by a $n \times n$ matrix in which entry $(u, v)$ is 1 if $(u, v) \in G$ and $-1$ if $(u, v) \notin G$ (that is $(v, u) \in G$). A *sink* is a node $u$ such that $u$'s row is all 1, that is, for all nodes $v$, entry $(u, v)$ is 1.

   Show that one can find a sink (if it exists) in $O(n)$ queries.

2. In class we saw a property testing algorithm for connectivity which made $O(1/(\epsilon^2 d))$ queries. Give a property testing algorithm for connectivity which makes only $O(\frac{1}{\epsilon} polylog(1/(\epsilon d)))$ queries.[1]

3. Show a lower bound on giving a multiplicative estimate on the MST: Give two distributions over graphs of degree at most $d$ and weights in the range $\{1, \ldots, w\}$ such that

   (a) graphs in one distribution have an MST weight that is at least twice the MST weight of the graphs in the in other distribution

   (b) in order to distinguish the two distributions with constant probability of success, one must make at least $\Omega(w)$ queries

[1] By $O(polylog(m))$, we mean that there is a constant $c$, such that the quantity is $O(\log^c m)$.

If you can get the lower bound to have some nontrivial dependence on $d$ and $\epsilon$, even better!

4. The diameter of an unweighted graph is the maximum distance between any pair of nodes. Give a tester for graphs with degree at most $d$ (in the adjacency list model) that have low diameter. The tester should have the following specific behavior:

   (a) Graphs with diameter at most $D$ are always accepted.

   (b) Graphs which are $\epsilon$-far (that is, at least $\epsilon dn$ edges must be added) from having diameter $4D + 2$ are failed with probability at least $2/3$.

   (c) The query complexity of the tester should be $O(1/\epsilon^c)$ for some constant $1 \le c \le \infty$.

   Hint: Show that if for every vertex, there are at least $k$ other vertices that are within distance $C$, then the graph can be transformed into a graph with diameter at most $4C+2$ by adding at most $n/k$ edges. What if one only knows that for at least $(1-1/k)n$ vertices, there are at least $k$ other vertices that are within distance $C$?

   (Make sure you know the answer to the following question, but don't bother to turn it in: Why didn't I ask for a tester in the adjacency matrix model?)