6.896 Sublinear Time Algorithms	September 21, 2004
Homework 1	
Lecturer: Ronitt Rubinfeld	*Due Date: October 7, 2004*

Homework guidelines: You may work with other students, as long as (1) they have not yet solved the problem, (2) you write down the names of all other students with which you discussed the problem, and (3) you write up the solution on your own. No points will be deducted, no matter how many people you talk to, as long as you are honest. If you already knew the answer to one of the problems (call these "famous" problems), then let me know that in your solution writeup – it will not affect your score, but will help me in the future. It's ok to look up famous sums and inequalities that help you to solve the problem, but don't look up an entire solution.

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

1. (Don't turn this one in) Show that given any property tester A that runs in time $T(n)$ on inputs of size n with probability of error 1/3, one can convert it into a property tester that runs in time $O(T(n) \log 1/\beta)$ with probability of error at most β. (Hint: run A $O(\log 1/\beta)$ times and take the majority answer. Use Chernoff bounds.)

2. 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.

3. (Don't turn this one in) Prove the lemma that we skipped in class: If a graph G is ϵ-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.

4. 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] (I might be convinced to give a hint, but first try it without).

5. (Don't turn this one in) 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 $[1..w]$. How many more queries do you need?

6. 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 ϵ, even better!

7. 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 ϵ-far (that is, at least ϵdn 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 $1 \leq c \leq \infty$.

   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?)