CS G713: Advanced Algorithms	**Problem Set 3**
Assigned: October 24, 2008	Due: November 7, 2008 before class

Guidelines If you write "I do not know how to solve this problem" then you get 1/4 of the score for the problem. If you write nonsense then you get 0.

As we are going to learn in this class, *time* and *space* are very valuable resources. Strive to give effective, compact solutions. Your solutions should touch on all the main points, but long calculations or discussions are not required nor sought.

Do not worry if you sometimes "do not get it." The problems are meant to stimulate you, not to overwork you. Unless specified otherwise, you can collaborate, but you must acknowledge all your collaborators in your solutions. To hand in your solutions: Give it to me, slide it under my door West Village H (246), or email it to csg713-instructor@ccs.neu.edu.

Problem 1. Puzzle: Loop detecting An array $A[1], A[2], A[3], ...$ defines a directed graph with outdegree at most 1: Node i points to Node $A[i]$, and it points to nothing (i.e. it is a sink) if $A[i] = 0$. Each entry in the array is at most N in magnitude.

Consider a walk that starts at Node 1. Such walk will either terminate in a sink or loop forever. Derive a $O(\log N)$-space algorithm that distinguishes between these two cases and that runs in time linear in the size of the connected component which includes Node 1. (Note that, because of this latter requirement, a bound of N on the running time is *not* sufficient for this problem; in particular, you do not have the time to scan the entire array.)

Problem 2. Log-space randomized algorithm with zero error for connectivity You are given a randomized algorithm A that, given an unweighted graph G on n nodes and two nodes s, t, satisfies the following:

(1) It uses space $O(\log n)$,

(2) if s, t are not connected, it always says "NOT CONNECTED," while

(3) if s, t are connected, it says "CONNECTED" with probability at least 1/2.

Use A to construct another randomized algorithm B for connectivity with the following properties:

(1) It uses space $O(\log n)$,

(2) on every input, B outputs "DON'T KNOW" with probability at most 1/2, but

(3) whenever B does not say "DON'T KNOW," it always correctly computes whether s, t are connected or not.

Hint: First solve the problem assuming that the algorithm is given the total number P_n of pairs (u, v) in the graph that are connected. Then show how to compute P_n using an inductive approach similar to the one in the Floyd-Warshall algorithm seen in class.

You may find it helpful to use the fact that if the edge relation of another graph G' is computable from G in deterministic space $O(\log n)$, then one can simulate $A(G')$ with only a negligible overhead in space and time.

Problem 3. CLRS Exercise 26.3-4 Prove Hall's Theorem. Hint: Use the max-flow min-cut theorem.

Problem 4. CLRS Problem 26-5 Maximum flow by scaling Note: c is integer-valued.