Fall 2001 **CMSC 451: Homework 1** Samir Khuller

Due in class: September 13.

Warning: some of the problems require thought – do not wait until the last day to start working on them!

If you cannot come up with algorithms that run in the required time, then provide (correct) slower algorithms for partial credit. Write your answers using *pseudo-code* in the same style as the textbook. These make the algorithm description precise, and easy to read (as opposed to code in C or some other language).

(1) Write out a pseudo-code description of an algorithm that will write down a list of the vertices that are cut vertices (or articulation vertices) in a given graph $G = (V, E)$. You may assume that G is connected. Each cut vertex should be listed exactly once. The algorithm should run in time $O(|E| + |V|)$.

(2) A cut edge of G is an edge whose removal disconnects G. Design an algorithm with running time $O(|E| + |V|)$ to find all the cut edges of a graph.

(3) In a directed graph, a **get-stuck** vertex is one that has in-degree $|V| - 1$ and out-degree 0. Assume that the adjacency matrix representation is used. Design an $O(|V|)$ algorithm to determine if a given graph has a sink. (Yes, this problem can be solved without even looking at the entire input matrix.) Write a proof of correctness for your algorithm.

(4) The diameter of a tree $T = (V, E)$ is given by

$$\max_{u,v \in V} \delta(u, v)$$

where $\delta(u, v)$ is the distance between u and v in the tree T. Give an $O(|V|)$ algorithm for computing the diameter of the tree. Write a proof of correctness for your algorithm.

(5) A bipartite graph is one whose vertex set can be partitioned into two sets A and B, such that each edge in the graph goes between a vertex in A and a vertex in B. (No edges between nodes in the same set are allowed.)

Give an $O(|E| + |V|)$ algorithm that takes an input graph (represented in adjacency list form) and decides if the graph is bipartite. If the graph is bipartite, the algorithm should also produce the bipartition.

(6) (FOR GRADUATE STUDENTS ONLY) An *Euler tour* of a connected graph is a cycle that traverses each edge of G exactly once, although it may visit a vertex more than once. Show that G has an Euler tour of and only if each vertex has even degree. Design a fast algorithm to output the Euler tour. (For full credit, the algorithm should run in linear time.)