

Due in class: September 22.

Warning: some of the problems require thought – do not wait until the last day to start working on them! I changed the due date to Thursday, so people have more time to talk to the TA or me during office hours.

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) Given a rooted tree T with n nodes, we wish to pre-process the tree in $O(n)$ time, so that we can answer the following queries in $O(1)$ time: given a pair of nodes u and v , is u an ancestor of v in the tree? How can we do this?
- (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) Problem 2 (page 107 from book).
- (5) Problem 4 (page 23 from book).