CS 222A Winter 2011, HW 8 Do the first three problems at least, and if you have time, try problem 4. This is due on Monday March 14, but you can turn it in until March 16.

1. Given a tree T with n nodes, suppose you label the nodes during a Depth-first traversal of T. You can label the nodes by an in-order labeling, a post-order labeling, or a pre-order labeling. (If you don't know what these are, look them up - maybe on Wikipedia.) The DFT can do *a little* additional work so long as it only takes $O(n)$ time.

After the DF traversal, your algorithm will be given pairs of nodes (x, y) it has to determine if x is an ancestor of y, or y is an ancestor of x, or neither is an ancestor of the other. This has to be done in $O(1)$ time. This can be answered by an LCA query, but it is a very restricted form of an an LCA query. Therefore, it can be solved much more simply than by a general LCA query.

Explain completely what the preprocessing is, and how the ancestry queries can be answered in $O(1)$ time. The question calls for an answer that is *much* simpler than using the LCA method we are learning in class. Find the simplest answer you can, and show how it works with each of the three types of nodes labeling methods.

2. The algorithm we learned for global min-cut was for an undirected graph. Now suppose we have a directed graph where each edge has an edge weight. A directed graph G is said to be *strongly connected* if and only if it has the property that for every ordered pair of nodes (u, v) in G, there is a directed path from u to v in G. Suppose that G is strongly connected. In a strongly connected directed graph G, a global min-cut is defined as the smallest weight set of edges such that after the removal of those edges from G, the resulting graph is not strongly connected.

Show how to modify the global min-cut algorithm so that it finds a global min-cut in a strongly connected directed graph. Prove that it does. This really involves going through the proof in the undirected case to see if it goes through, and making whatever changes in the proof are required.

UPDATE March 7: Actually, this seems pretty hard (although maybe I am missing something). So now I will change the problem as follows: To compute a global min cut in a directed, weighted graph, we could pick an arbitrary node v and then compute the max flow from v to each other node w, and also compute the associated min v, w cut. This would involve $n - 1$

max flow computations. Then we would compute the max flow from each node w to v, and also compute the associated w, v cut. Because the graph is directed, the min v, w cut and the min w, v cut can be very different with very different capacities. But the min over these $2(n-1)$ cuts is a global min cut in a directed graph. Prove that this is correct.

OPTIONAL QUESTION: Next, we want to reduce the number of max flows from $2(n-1)$ to just $n-1$, along with an algorithm whose time is at most the time for computing a global min cut in an undirected graph. I claim we can do this using a modification of the undirected global min cut method, plus $n-1$ max flow computations. Show how to do this.

3. Do problem 28 on p 519 of the book. This may at first seem trivial, but I don't think it is.

4. Try problem 2 on p. 594 of the book. I have not yet tried this, so I don't know how hard it is.