CS180 Winter 2011

Homework 2

The following homework is due Wednesday, January 19 at the beginning of lecture.

When submitting your homework, please include your name at the top of each page. If you submit multiple pages, please staple them together. We also ask that you do something to indicate which name is your last name on the first page, such as underlining it.

Please provide complete arguments and time complexity analysis for all solutions, unless otherwise stated.

1. Give an algorithm for finding the diameter of a undirected tree using 2 BFSs. Prove the correctness of your algorithm.. (You will get 20% score for saying "I don't know!" But you will get zero points for writing arguments which are almost completely wrong.)

2. Give an algorithm to find the longest directed hop-count path in a DAG (directed acyclic graph).

3. Give an algorithm that pairs up odd degree nodes of a tree such that the nodes in these pairs are connected by paths that are edge disjoint.

3.4 from textbook
Inspired by the example of that great Cornellian, Vladimir Nabokov, some of your friends have become amateur lepidopterists (they study butterflies). Often when they return from a trip with specimens of butterflies, it is very difficult for them to tell how many distinct species they've caught - thanks to the fact that many species look very similar to one another.
One day they return with n butterflies, and they believe each belongs to one of two different species, which we'll call A and B for purposes of this discussion. They'd like to divide the n specimens into two groups - those that belong to A and those that belong to B - but it's very hard for them to directly label any one specimen. So they decide to adopt the following approach.
For each pair of specimens i and j, they study them carefully side by side. If they're confident enough in their judgment, then they label the pair (i, j) either "same" or "different." They also have the option of rendering no judgment on a given pair, in which case we'll call the pair *ambiguous*.
So now they have a collection of n specimens, as well as a collection of m judgments for the pairs that were not ambiguous. They'd like to know if this data is consistent with the idea that each butterfly is from one of species A or B. So more concretely, we'll declare the m judgments to be *consistent* if it is possible to label each specimen either A or B in such a way that for each pair (i, j) labeled "same," it is the case that i and j have the same label; and for each pair (i, j) labeled "different," it is the case that i and j have different labels.
They're in the middle of tediously working out whether their judgments are consistent, when one of them realizes that you probably have an algorithm that would answer this question right away.
Give an algorithm with running time $O(n + m)$ that determines whether the m judgments are consistent.