Due on March 24th, 2014
75 points total

General Directions: If you are asked to provide an algorithm, you should clearly define each step of the procedure, establish its correctness, and then analyze its overall running time. There is no need to write pseudo-code; an unambiguous description of your algorithm in plain text will suffice.

All the answers must be typed, preferably using LaTeX. If you are unfamiliar with LaTeX, you are strongly encouraged to learn it. However, answers typed in other text processing software and properly converted to a pdf file will also be accepted. Before submitting the pdf file, please make sure that it can be opened using any standard pdf reader (such as Acrobat Reader) and your entire answer is readable. **Handwritten answers or pdf files that cannot be opened will not be graded and will not receive any credit.**

Finally, please read the detailed collaboration policy given on the course website. You are **not** allowed to discuss homework problems in groups of more than 3 students. **Failure to adhere to these guidelines will be promptly reported to the relevant authority without exception.**

Problem 1 (25 points)
Show that the following algorithm for minimum spanning trees is correct: Initialize F to the empty set. In each round of the algorithm, for each connected component S of F, add the minimum weight edge in the cut $(S, V \setminus S)$ to F. Terminate when F contains a single connected component, and output F. (Assume that all edge weights are distinct.)

(Hint: First, show that F remains acyclic throughout the algorithm. Then, use the generic property of minimum spanning trees proved in class to show that the algorithm computes a minimum spanning tree.)

Problem 2 (5 points)
Alice Algorithmix and Bob Bitfiddler are quarreling once again about travel times in Durham! They are comparing two situations, one where there is no congestion on any road segment and another where there is exactly the same delay on each road segment because of congestion. Alice claims that neither the minimum spanning tree nor the shortest path tree that she had computed for the first scenario can be reused in the second scenario, while Bob claims that both can. Can you resolve their argument?

(Hint: Does the minimum spanning tree or the shortest path tree change if the length of every edge in a graph is increased by the same amount?)

Problem 3 (10 points)
You are planning a second party (remember the first party?) for spring break. You have a total of n friends, each of whom has given you two lists with a subset of these n friends of yours. Each friend will attend your party if and only if everyone in their first list and no one in their second list is invited to your party. You quickly realize that you must throw multiple parties if you want to invite every friend. What is the most efficient algorithm you can come up with to decide if you can throw a set of parties such that each of your n friends attends a party?

Problem 4 (10 points)
Use the potential method to show that the amortized running time of each union operation in the star data structure for disjoint sets is $O(\log n)$, where n is the total number of elements.

Problem 5 (25 points)
Consider the following randomized algorithm for computing a spanning tree S for an *unweighted* undirected graph $G = (V, E)$ where $|V| = n$ and $|E| = m$. Denote by E_v the set of edges incident on vertex $v \in V$:

1. Pick an arbitrary root $r \in V$.
2. Starting at r, begin traversing the graph randomly, i.e., when at a given vertex v pick an edge $e = (v, u)$ from E_v uniformly at random and then travel along e to u.
3. Whenever we travel along an edge e and reach a vertex for the first time, add e to S.

We continue this random traversal until we visit every vertex. Let p_e be the probability that edge e is added to S during the random process. The following questions ask you to analyze this algorithm.

(a) (5 points) Prove that the edges in S at the end of the algorithm form a spanning tree.

(b) (5 points) Show that $\sum_{e \in E} p_e = n - 1$.

(c) (15 points) For an edge $e = (u, v)$, define $d_e = \min(\deg(u), \deg(v))$, where $\deg(v) = |E_v|$ is the *degree* of a vertex v. Prove that $\sum_{e \in E}(p_e \cdot d_e) \leq 2m$.