ECE368 Homework #11

IMPORTANT: Write your user (login) ID at the TOP of EACH page. Also, be sure to *read* and *sign* the *Academic Honesty Statement* that follows:

"In signing this statement, I hereby certify that the work on this exercise is my own and that I have not copied the work of any other student while completing it. I understand that, if I fail to honor this agreement, I will receive a score of ZERO for this exercise and will be subject to possible disciplinary action."

Printed Name: **No submission required**

Login:

Signature:

Please acknowledge those people who have helped you with this homework.

# of Question	Credits

1. Dijkstra's

We are given a directed graph $G = (V, E)$ on which each edge $(u,v) \in E$ has an associated value $r(u,v)$, which is a real number in the range $0 \le r(u,v) \le 1$ that represents the reliability of a communication channel from vertex u to vertex v. We interpret $r(u,v)$ as the probability that the channel from u to v will not fail, and we assume that these probabilities are independent. Give an efficient algorithm to find the most reliable path between two given vertices.

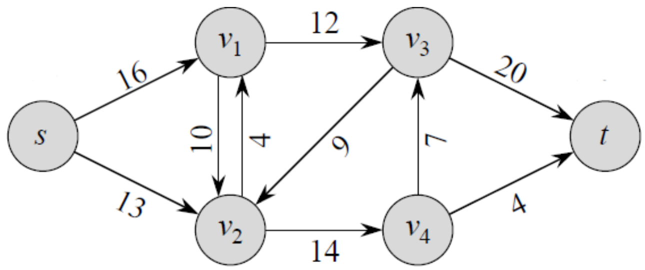
2. **Dijkstra's**

Suppose we change line 4 of Dijkstra's algorithm to the following:

4 while $|Q| > 1$

This change causes the while loop to execute $|V| - 1$ times instead of $|V|$ times. Is this proposed algorithm correct?

3. **Max flow graph**. Following the example shown in the lecture slides, show how *Ford-Fulkerson* works on the following graph to find the max s-t flow. In each step: show the flow on the input graph; show the residual graph; highlight the augmenting path on the residual graph. Describe how you identify the augmenting path in each step.



4. **Min spanning Tree.**
   Suppose that we represent the graph G = (V, E) as an adjacency matrix. Give a simple implementation of Prims algorithm for this case that runs in $O(V^2)$ time.

5. Min spanning tree

Professor Borden proposes a new divide-and-conquer algorithm for computing minimum spanning trees, which goes as follows. Given a graph $G = (V, E)$, partition the set V of vertices into two sets V_1 and V_2 such that $|V_1|$ and $|V_2|$ differ by at most 1. Let E_1 be the set of edges that are incident only on vertices in V_1, and let E_2 be the set of edges that are incident only on vertices in V_2. Recursively solve a minimum-spanning-tree problem on each of the two subgraphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Finally, select the minimum-weight edge in E that crosses the cut V_1, V_2, and use this edge to unite the resulting two minimum spanning trees into a single spanning tree.

Either argue that the algorithm correctly computes a minimum spanning tree of G, or provide an example for which the algorithm fails.