CS 170 Efficient Algorithms and Intractable Problems
Spring 2016 Alessandro Chiesa and Umesh Vazirani **HW 5**

Due February 22, 11:59pm

1. (20 pts.) Short Questions

1. **(10 pts.)** Let G be a connected undirected graph with positive lengths on all the edges. Let s be a fixed vertex. Let $d(s,v)$ denote the distance from vertex s to vertex v, i.e., the length of the shortest path from s to v. If we choose the vertex v that makes $d(s,v)$ as small as possible, subject to the requirement that $v \neq s$, then does every edge on the path from s to v have to be part of every minimum spanning tree of G?
2. **(10 pts.)** The same question as above, except now no two edges can have the same length.

2. (20 pts.) Preventing Conflict

A group of n guests shows up to a house for a party, and any two guests are either friends or enemies. There are two rooms in the house, and the host wants to distribute guests among the rooms, breaking up as many pairs of enemies as possible. The guests are all waiting outside the house and are impatient to get in, so the host needs to assign them to the two rooms quickly, even if this means that it's not the best possible solution. Come up with an efficient algorithm that breaks up at least half the number of pairs of enemies as the best possible solution, and prove your answer.

3. (20 pts.) Graph Subsets

Let $G = (V,E)$ be a connected, undirected graph, with edge weights $w(e)$ on each edge e. Some edge weights *might be negative*. We want to find a subset of edges $E' \subseteq E$ such that $G' = (V,E')$ is connected, and the sum of the edge weights in E' is as small as possible.

1. Is it guaranteed that the optimal solution E to this problem will always form a tree?
2. Does Kruskal's algorithm solve this problem? If yes, explain why in a sentence or two; if no, give a small counterexample.
3. Describe an efficient algorithm for this problem. Be concise. You should be able to describe your algorithm in one or two sentences. (You dont need to prove your algorithm correct, justify it, show pseudocode, or analyze its running time.)

4. (20 pts.) Timesheets Suppose we have N jobs labeled $1,\ldots,N$. For each job, there is a bonus $V_i \geq 0$ for completing the job, and a penalty $P_i \geq 0$ per day that accumulates for each day until the job is completed. It will take $R_i \geq 0$ days to successfully complete job i.

Each day, we choose one unfinished job to work on. A job i has been finished if we have spent R_i days working on it. This doesn't necessarily mean you have to spend R_i consecutive days working on job i. We start on day 1, and we want to complete all our jobs and finish with maximum reward. If we finish job i at the end of day t, we will get reward $V_i - t \cdot P_i$. Note, this value can be negative if you choose to delay a job for too long.

Given this information, what is the optimal job scheduling policy to complete all of the jobs?

5. (20 pts.) A greedy algorithm – so to speak The founder of LinkedIn, the professional networking site, decides to crawl LinkedIn's relationship graph to find all of the *super-schmoozers*. (He figures he can make more money from advertisers by charging a premium for ads displayed to super-schmoozers.) A *super-schmoozer* is a person on LinkedIn who has a link to at least 20 other super-schmoozers on LinkedIn.

We can formalize this as a graph problem. Let the undirected graph $G = (V, E)$ denote LinkedIn's relationship graph, where each vertex represents a person who has an account on LinkedIn. There is an edge $\{u, v\} \in E$ if u and v have listed a professional relationship with each other on LinkedIn (we will assume that relationships are symmetric). We are looking for a subset $S \subseteq V$ of vertices so that every vertex $s \in S$ has edges to at least 20 other vertices in S. And we want to make the set S as large as possible, subject to these constraints.

Design an efficient algorithm to find the set of super-schmoozers (the largest set S that is consistent with these constraints), given the graph G.

Hint: There are some vertices you can rule out immediately as not super-schmoozers.