1. (15 points) **(Designing a Tournament)** This question will help you understand the applications of Chernoff bound and Union bound more, as you'll deal with tuning parameters to optimize the efficiency of the system. You'll also get an understanding of why NBA tournaments are designed the way they are, with more repeated matches being played as we get closer to the finals. For example, in the NBA, the early rounds are *best-of-five*, and the later rounds become *best-of-seven*. You will understand why, and also learn how to design tournaments with n participants, for large n.

   Suppose there are n teams, and they are totally ranked. That is, there is a well-defined best team, second ranked team and so on. It's just that we (the tournament designer) don't know the ranking. Moreover, assume that for any given match between two players, the better ranked team will win the match with probability $p = \frac{1}{2} + \delta$, *independent of all other matches between these players and all other players also.* Here δ is a small positive constant. All your answers below will depend on the value of δ.

   (a) (3 points) Let n be a power of two, and fix an *arbitrary tournament tree* starting with $n/2$ matches, then $n/4$ matches and so on. That is, initially, team 1 plays team 2, team 3 plays team 4, and so on (each series is only 1 game). The winners advance, and pair up and play each other once, and so on until one team remains. What is the probability that the best team wins the tournament?

   Solution:

   Proof. $1 + 1 = 2$. □

   (b) (5 points) Now change the tournament to make each series as a *best-of-k* series. How large should k be, and so how many games do you end up conducting in total to get a $1 - \epsilon$ probability of the best team winning the overall series?

   Solution:

   Proof. $1 + 1 = 2$. □

   (c) (8 points) Can you get better dependence on n if you allow different number of games in each round: example, try having k_1 games in the first series, k_2 games in the next series, etc. and optimize for these values to get a total of $O_\epsilon(n)$ games which still retains $1-\epsilon$ probability of the best team winning eventually. Here $O_\epsilon(n)$ means $O(n)$ for all constant $\epsilon > 0$.

   Solution:

   Proof. $1 + 1 = 2$. □

2. (15 points) **(Different Algorithms for Weighted Vertex Cover)** Here we will analyze different possible algorithms for weighted vertex cover. For each of the following

approximation algorithms for Min-Vertex-Cover with positive vertex weights, prove the best approximation ratio guarantee that you can.

(a) (2 points) Super Naive: Consider all the edges in some arbitrary order. If the edge $\{u, v\}$ being considered is not covered yet, pick whichever of u or v has less weight.

Solution:

Proof. $1 + 1 = 2$. □

(b) (2 points) Naive: Consider all the edges in some arbitrary order. If the edge $\{u, v\}$ being considered is not covered yet, pick both u and v.

Solution:

Proof. $1 + 1 = 2$. □

(c) (6 points) Randomized: Consider all the edges in some arbitrary order. If the edge $\{u, v\}$ being considered is not covered yet, pick u with probability $w_u/(w_u + w_v)$ and v with the remaining probability. We want to bound the expected cost of our solution in terms of the optimal cost.

Solution:

Proof. $1 + 1 = 2$. □

(d) (5 points) Local search: Define two solutions $S \subseteq V$ and $S' \subseteq V$ to be neighbors if S can be obtained from S' by adding, deleting, or swapping a vertex. The local search moves are simple: Start with any solution $S \subseteq V$; if you are at some solution S, move to any neighboring solution S' that has less weight. If you are at a local optimum where all the neighbors have at least as much weight output this local optimum. (Dont worry about the running time for this algorithm.)

3. (15 points) **The Police Station Problem.** The problem is as follows: The input is a positive integer k and a complete undirected graph G with distances $d(u, v)$ on each edge (u, v). The distances form what is called a *metric*: that is, $d(v, v) = 0$ for all v, $d(u, v) = d(v, u)0$ for all $u \neq v$, and $d(u, w) \leq d(u, v) + d(v, w)$ for all triplets u, v, w.

A valid output is a subset S of at most k vertices. The cost (to be minimized) of a solution is the maximum distance of any vertex from S; i.e., $\max_v \min_{s \in S} d(v, s)$. One can imagine being able to open k police stations, in an attempt to minimize the distance of everyone in the city to the closest station.

(a) (7 points) (Hardness of Approximation) Show that approximating this problem to within ratio $2 - \epsilon$ is NP-hard for all $\epsilon > 0$. (Hint: show that Dominating-Set is NP-hard by first assuming that vertex cover is NP-hard, and then reduce our problem to dominating set.)

Solution:

Proof. $1 + 1 = 2$. □

(b) (8 points) Show that the following algorithm achieves a 2-approximation. Pick any vertex v_1 in the graph to begin. Pick a vertex v_2 furthest from v_1. Pick a vertex v_3 furthest from the set $S_2 = \{v_1, v_2\}$ that is, v_3 is the vertex maximizing $\min_{s \in S_2} d(v, s)$, and so on. In general, pick v_j furthest from $S_{j1} = \{v_1, v_2, ..., v_{j1}\}$. Stop when you have k vertices. Show that this is a 2-approximation.

Solution:

Proof. $1 + 1 = 2$. □

4. (10 points) **Fun with Submodularity.** In this exercise we will get a better understanding of submodular functions.

(a) (5 points) In class we defined submodularity as the definition that $f(S \cup \{i\}) - f(S) \geq f(T \cup \{i\}) - f(T)$ for all $S \subseteq T$ and $i \notin T$. Show that this is the same as the condition that $f(S) + f(T) \geq f(S \cap T) + f(S \cup T)$ for all $S, T \subseteq [n]$.

(b) (5 points) Let $\Omega = \{v_1, v_2, \ldots, v_n\}$ be the vertices of an undirected graph. For any set of vertices $S \subseteq \Omega$ let $f(S)$ denote the number of edges $e = (u, v)$ such that $u \in S$ and $v \in \Omega - S$. Show that f is submodular.

Solution:

Proof. $1 + 1 = 2$. □

5. (0 points) **Difficulty Level.** How difficult was this homework? How much time would you have spent on these questions?

Solution:

Proof. $1 + 1 = 2$. □