601.435 / 601.635 Approximation Algorithms **Spring 2024**
Homework #1 **Due:** February 15, 2024, 9:00am

Reminder: you may work in groups of up to three people, but must write up solutions entirely on your own. Collaboration is limited to discussing the problems – you may not look at, compare, reuse, etc. any text from anyone else in the class. Please include your list of collaborators on the first page of your submission. Many of these problems have solutions which can be found on the internet – please don't look. You can of course use the internet (including the links provided on the course webpage) as a learning tool, but don't go looking for solutions.

Please include proofs with all of your answers, unless stated otherwise.

---

1 k-suppliers (33 points)

The k-suppliers problem is similar to k-center. We are given a metric space (V, d) and a natural number k. However, in k-suppliers the set of points V is partitioned into two sets: the *suppliers* F and the *customers* $D = V \setminus F$. The goal is to find a set of suppliers $S \subseteq F$ with $|S| = k$ that minimizes $\max_{u \in D} d(u, S)$. Give a 3-approximation algorithm for this problem. Hint: think about the greedy 2-approximation for k-center from class

2 Hardness of Minimum Degree Spanning Tree (33 points)

Recall the minimum degree spanning tree problem: given a graph $G = (V, E)$, find the spanning tree which minimizes the maximum degree. Prove that unless P = NP, there is no α-approximation for this problem with $\alpha < 3/2$. Hint: consider the Hamiltonian Path problem.

3 Edge-Disjoint Paths (34 points)

In the edge-disjoint paths problem (EDP), the input is an undirected graph $G = (V, E)$ and a set $T = \{(s_1, t_2), (s_2, t_2), \ldots, (s_k, t_k)\}$, such that $s_i, t_i \in V$ for all $i \in [k]$. A feasible solution is a set $I \subseteq [k]$ and for all $i \in I$ a path P_i between s_i and t_i, with the additional constraint that $P_i \cap P_j = \emptyset$ for $i, j \in I$ with $i \neq j$ (where we view paths as edge sets). In other words, a feasible solution is a set of edge-disjoint paths between a subset of the pairs in T. The objective is to maximize $|I|$, i.e., the number of edge-disjoint paths that we can find.

Consider the following greedy algorithm, where initially $I \leftarrow \emptyset$:

1. Initially $I = \emptyset$ and $H = G$
2. Repeat until all pairs (s_i, t_i), $i \in [k] \setminus I$ are disconnected in H:
   (a) Let $i \in [k] \setminus I$ be the index which minimizes the distance between s_i and t_i in H, i.e., $i = \arg\min_{j \in [k] \setminus I} d_H(s_j, t_j)$
   (b) Let P_i be a shortest path between s_i and t_i in H.
   (c) Add i to I and choose path P_i for i, and remove all edges of P_i from H.
3. Return I and the paths $\{P_i : i \in I\}$

Informally, this algorithm just always picks the shortest possible path remaining, then deletes this path from the graph and continues. Prove that this is an $O(\sqrt{m})$-approximation (where $m = |E|$).

Hint: divide paths up into short paths (length at most $\sqrt{m}$) and long paths (length larger than $\sqrt{m}$).