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.

Student: Ha Manh Bui (hbui13@jhu.edu)

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.

3-approximation algorithm:

1. Initially $D' \leftarrow \{u\}$, for $u \in D$ arbitrary
2. While $|D'| < k$:
   (a) Let $u \in D \setminus D'$ be the element maximizing $d(u, D')$
   (b) $D' \leftarrow D' \cup \{u\}$
3. Return $S = \{f_i | d_i \in D'\}$, where $f_i \in F$ is nearest supplier of customer $d_i \in D'$.

Proof: Let $d(u, v)$ be the distance between u and v, and OPT be the optimal solution to k-supplier problem, then we have

$$d(d_i, f_i) \leq OPT, \forall i \in [|S|]. \tag{1}$$

Therefore, if $|D| \leq k$, then $S = \{f_i | d_i \in D'\}$ gives an optimal solution.
Consider $|D| > k$, since the first 2 steps of the algorithm are applying k-center to D, so we know that this is a 2-approximation and $\forall u \in D, d(u, D') \leq 2 \cdot OPT$. Therefore:

(a) If $\exists d_i \in D'$ s.t. $d(u, d_i) \leq 2OPT, \forall u \in D$, then by the triangle inequality, we have

$$d(u, f_i) \leq d(u, d_i) + d(d_i, f_i) \leq 2OPT + OPT = 3 \cdot OPT. \tag{2}$$

(b) Else, then $\forall d_i, d_j \in D'$, we have

$$d(d_i, d_j) > 2 \cdot OPT, \tag{3}$$

yielding $D' \cup \{u\}$ consists of $k + 1$ customers s.t. any pair is at a distance $> 2OPT$. This is impossible because $\geq k + 1$ suppliers are needed for a solution with value OPT.

As a result, we get $\forall u \in D, d(u, S) \leq 3 \cdot OPT$, yielding $\max_{u \in D} d(u, S) \leq 3 \cdot OPT$, since the objective function is minimizing, we obtain the algorithm is a 3-approximation. ■

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.

Proof: Recall the Theorem 2.18 in The Design of Approximation Algorithms book. "The Minimum Degree Spanning Tree is NP-complete to decide whether or not a given graph has a minimum-degree spanning tree of maximum degree two.".
By reduction of the Hamiltonian Path problem, i.e., finding a path that goes through every vertex exactly once in the graph $G = (V, E)$, we can see that the problem of a graph having a spanning tree with a maximum degree ≤ 2 is the same as the Hamiltonian path problem.
So, let the graph G be an instance of the Hamiltonian Path problem. Assume there is an α-approximation for this problem with $\alpha < 3/2$, apply this algorithm to G, we have:

(a) If G has a Hamiltonian Path, then the algorithm will output a value $\leq \alpha \cdot 2 < 3$.

(b) Else, then the algorithm will output a value of ≥ 3.

Therefore, this algorithm can solve the Hamiltonian Path problem in polynomial time, which is impossible assuming that $P \neq NP$. As a result, we obtain that unless $P = NP$, there is no α-approximation for this problem with $\alpha < 3/2$. ■

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 $\sqrt{m}$).

Proof: Let OPT be the optimal solution and ALL be the greedy solution to the Edge-Disjoint Paths problem. Divide paths up into short paths (length $\leq \sqrt{m}$) and long paths (length $> \sqrt{m}$), then decompose the optimal solution in long and short paths, we have

$$OPT = OPT_s + OPT_l, \tag{4}$$

where OPT_s and OPT_l represent the optimal value on short and long paths respectively.
Firstly, let P_{OPT} be a fixed short path in OPT, then P_i from ALL will cut P_{OPT} for the first time, yielding P_{OPT} gets charged to P_i and all previously selected paths of ALL are disjoint with P_{OPT}. So, P_{OPT} is still available, but ALL select P_i, implying the length of $P_i < \sqrt{m}$. As a result, each short path in OPT gets charged to some short path in ALL. On the other hand, since paths in OPT are disjoint, in the worst case each edge of a short path selected by ALL cuts a different path of OPT, yielding each short path in ALL gets charged $\leq \sqrt{m}$ times. Therefore, we get

$$OPT_s \leq ALL \cdot \sqrt{m}. \tag{5}$$

Secondly, since paths in OPT are disjoint, we know that $\frac{m}{\sqrt{m}}$ long paths will cover all m edges, i.e.,

$$OPT_l \leq \frac{m}{\sqrt{m}} = \sqrt{m}. \tag{6}$$

Plugging the result in Eq. 5 and Eq. 6 to Eq. 4, we obtain

$$\begin{aligned} OPT &\leq ALL \cdot \sqrt{m} + \sqrt{m} && (7)\\ &\leq ALL \cdot \sqrt{m} + ALL \cdot \sqrt{m} \text{ (since } ALL \geq 1) && (8)\\ &= 2 \cdot ALL \cdot \sqrt{m}, && (9) \end{aligned}$$

i.e.,

$$\frac{OPT}{ALL} \leq 2 \cdot \sqrt{m}. \tag{10}$$

As a result, since $\sqrt{m} \geq 1$, where $m = |E|$ and the objective is maximizing, we obtain the greedy is an $\mathcal{O}(\sqrt{m})$-approximation. ■