IT584 Approximation Algorithms January 23, 2024

Lecture 07

Lecturer: Rachit Chhaya *Scribe/s: Vimal Parmar*

1 The K-Center Problem

Consider a scenario with a large dataset exhibiting varying degrees of similarity among data points. The goal is to group similar data together, potentially in a specified number of clusters. To achieve this, a subset of data points needs to be selected as cluster centers, allowing each data point to be assigned to its nearest cluster center and identifying the clusters.

1.1 Problem Definition

Input: An undirected complete graph $G = (V, E)$ with distances $d_{ij} \geq 0$ between each pair $i, j \in V$, where the distances follow the "Metric" rule, and a positive integer K.

Goal: Find k clusters, denoted by $S \subseteq V$ with $|S| = k$, grouping together the vertices that are most similar to each other. Each vertex assigns itself to its closest cluster center. The K-Center problem aims to minimize the maximum distance of a vertex to its cluster.

1.2 Geometric Interpretation

. The distance function $d(\cdot)$ must satisfy the following properties:

1. $d(x, y) \geq 0$ for all $x, y \in V$, and $d(xy) = 0$ if and only if $x = y$.
2. $d(x, y) = d(y, x)$.
3. $d(x, y) \leq d(x, z) + d(z, y)$.

Our goal is to identify a collection S comprising k vertices, denoted as cluster centers. The aim is to minimize the maximum distance from any vertex to its respective cluster center. In this context, for each vertex i belonging to the set V, the assignment involves associating it with the cluster centered around the nearest vertex s within S.

$$d(i, S) = \min_{s \in S} d(i, s), \quad \text{where } s \in S$$

Considering our selected cluster centers in set S, we represent the radius of S in the following manner:

$$r = \max_{i \in V} d(i, S)$$

We can reformulate our aim as discovering S in a way that minimizes the radius of S. In other words, our task involves finding S while adhering to the condition:

$$\min_{S \subseteq V : |S| = k} \max_{i \in V} d(i, S)$$

1.3 A Greedy Algorithm

On an intuitive level, we aim for the selected centers to be distributed as widely as possible. This serves the purpose of:

1. Providing all vertices in the graph with an opportunity to have a nearby vertex as their center.

2. Ensuring that outlier vertices, located far away from others, do not overly impact the maximum distance.

The algorithm unfolds as follows: initially, choose a vertex $i \in V$ arbitrarily and include it in our set S of cluster centers. Subsequently, it is logical for the next cluster center to be positioned as far away as possible from all the other cluster centers. While $|S| < k$, iteratively identify a vertex $j \in V$ for which the distance $d(j, S)$ is maximized (or, in simpler terms, which establishes the diameter of set S). Incorporate it into S. The process continues until $|S| = k$, at which point we conclude and return S.

The Greedy Algorithm for K-Centering

Commence by selecting an arbitrary vertex $s \in V$ and initialize the set $S = \{s\}$. Continue until $|S| < k$ with the following steps:

1. Choose s as the vertex that maximizes the distance $d(s, S)$.
2. Update S by adding s to it: $S \leftarrow S \cup \{s\}$.

Approximation Analysis

Claim : The greedy algorithm provides a 2-approximation solution for the k-clustering problem.

Proof : Consider $S^* = \{s_1, s_2, \dots, s_k\}$ as the representation of the optimal solution, with r^* as its corresponding radius. This optimal solution divides the nodes V into clusters $V_1^*, V_2^*, \dots, V_k^*$, where each point i is assigned to V_l^* if it is the closest to s_l among all the points in S^*. In cases of ties, the decision is made arbitrarily.

Firstly, it's important to note that for any pair of points i and j within the same cluster V_l^*, their maximum separation is $2r^*$. Applying the triangle inequality, the distance $d(i, j)$ between them is at most the combined sum of $d(i, s_l)$ (the distance from i to the center s_l and $d(j, s_l)$ (the distance from the center s_l to j). Both of these individual distances are confined within r^*, hence:

$$d(i, j) \leq d(i, s_l) + d(j, s_l) \leq r^* + r^* = 2r^*$$

Now, let's contemplate the set $S \subseteq V$ consisting of points chosen by the greedy algorithm. Specifically, focus on the initial iteration where the algorithm opts for a point $i \in V_l^*$ to include in S, despite having previously selected another point $i' \in V_l^*$ in an earlier iteration. Before the addition of i', each center incorporated into S was chosen from distinct optimal clusters of S^*.

For all points j covered by centers added prior to i', it follows from the earlier argument that $d(j, S) \leq r^*$. Given the nature of the greedy algorithm, which consistently selects points farthest from the current set of points in S, the distance for any other point $j \in V$, not yet added to S, must be constrained by

$$d(j, S) \leq d(i, i')$$

If j had not been added to S before i', it would imply that i should have been prioritized for inclusion before i'. Nevertheless, i and i' both belong to the same optimal cluster V_l^*, indicating that $d(i, i') \leq 2r^*$. Consequently, for all points covered after i' is incorporated into the cluster centers, the distance between each point and its nearest center is also limited to at most $2r^*$.

For every point $i \in V$, the distance is therefore constrained by $d(i, S) \leq 2r^*$. If r denotes the radius of S obtained from our greedy solution, we can deduce:

$$r^* \leq r \leq 2r^*$$

This implies that the algorithm provides a 2-approximation as needed.