Lecture Note 4

1 K-center

1.1 History

- K-center (Gonzalez 85)
- K-median (Hochbaum-Shmoys86)

1.2 Problem

- Define $c(i, j)$ to be the distance between i and j.
- Full set: S
- Goal: select $S' \subseteq S$, where $|S'| = k$.
- cost(v) is defined as $\min_{u \in S'} c(u, v)$.
- K-center: $\min \max_v \text{cost}(v)$,
- K-median: $\min \sum_j cost(v)$

1.3 NP-completeness

Takes $O(n^k)$ time, and will only work for small k.

- We cannot guarantee optimal solution, but we can guarantee that $\max_v \text{cost}(v) \leq 2 \cdot \text{cost}$ for an optimal placement.

1.4 Hochbaum-Shmoys Algorithm

- Guess optimal distance D
- While there are still uncovered nodes
  - Pick an uncovered node v
  - Assign everything in $2D$ to it, and mark them as covered

1.5 Gonzalez's Algorithm

- Initially pick any node v_0 as center, add it into S
- For $i = 1$ to $k - 1$:
  - Find the v_i node farthest to S, where distance is defined as $\text{dist}(v) = \min_{u \in S} c(u, v)$.
  - Add v_i to S.
- Return S.

1.6 Details

Details and proofs of previous two algorithms can be find here link

2 Invited speaker

- David Mount, email: mount at cs.umd.edu

2.1 Applications

- Geometric Approximation Algorithms
- Can be used in machine learning, and learn property vectors: $(x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$.
- Given point set $P \subseteq \mathbb{R}^d$. There are two cases:
  - d small: $2, 3, 4, \ldots, 20$
  - d large: hundreds, thousands... We need dimension reduction.
- Euclidean distance

2.2 Topics

- Kernels & directional width
- Clustering
  - Additive costs
  - Multiplicative costs
- Streaming [Very large data, small memory]

2.3 Coreset

2.3.1 Motivation

Given large data set (reals)

1. Want (Random sample + sample mean & sample median)
   - average
   - median

- min
- max

2. Why we do not randomly sample (counter-example)
   - Given points P in plane, compute diameter, which is defined as $\text{diam}(P) = \max_{p,q \in P} dist(p, q)$.
   - Random sampling will not work

2.3.2 Definition

Given problem X, $\epsilon > 0$ [error parameter], a point set P, a coreset for P (w.r.t X) is a subset $Q \subseteq P$ s.t. answer for X on Q is within error ϵ to answer for X on P.

- E.g. cost(P): $\frac{\text{cost}(P)}{1+\epsilon} \leq \text{cost}(Q) \leq \text{cost}(P)$, much smaller set behaves similar to P w.r.t. X.

2.3.3 Coreset for diameter? First try

- $O(n)$-estimate for diameter
  - Pick any point
  - Find dist to farthest
  - $z \leq \text{diam}(q) \leq 2 \cdot z$

2.3.4 Algorithm

Make a grid, diameter of each cell $= \epsilon \cdot Z/2 \leq \frac{\epsilon}{2}\text{diam}(P)$.

- In $O(1)$ time, can determine which cell contains any point (floor/integer division required)
- Use hashing to assign points to cell
- $Q \leftarrow$ take one point from every non-empty grid cell

2.3.5 How large is Q?

- #cells on a side $\leq \frac{\text{diam}(P)}{(\epsilon/2)\text{diam}(P)} = 2/\epsilon$
- Total #cells $\leq (2/\epsilon)^d = O((\frac{1}{\epsilon})^d)$
- $\text{diam}(Q) \leq \text{diam}(P)$, since Q is a subset
- $\text{diam}(Q) \geq \text{diam}(P) - 2\underbrace{(\epsilon/2 \cdot \text{diam}(P))}_{\text{Grid cell size}} = \text{diam}(P)(1 - \epsilon)$.

2.3.6 Diameter

- Compute ϵ-coreset for diameter in time $O(n + (1/\epsilon)^d)$ of size $O((1/\epsilon)^d)$
- Run naive on coreset in time: $O((1/\epsilon)^{2d})$, comparing to $O(n^2)$ exact algorithm.

2.3.7 Our $(1/\epsilon)^d$ coreset is too big

- Reason: grabs too many internal points
- Solution: Build an $\epsilon/2 \cdot \mathrm{diam}(P)$ size grid on one facet and extend it through
  - #cylinders $O((1/\epsilon)^{d-1})$.
  - $Q \leftarrow$ Take top most and bottom most from each cylinder.
- Claim: Q is ϵ-coreset for diameter
- Q is an ϵ-coreset for diameter
- Q is an ϵ-coreset for directional width
  - Query given unit vector $\vec{u}$ width in direction $\vec{u}$ that minimize distance between two parallel hyperplanes orthogonal to $\vec{u}$ that sandwich P
  - Proof: Fix any $\vec{u}$, $\omega(\vec{u}, Q) \geq \omega(\vec{u}, P) - 2(\frac{\epsilon}{2} \cdot \mathrm{diam}(P))$.

2.3.8 A trick

- If it is fat: good
- If it is skinny: fattening transformation

2.3.9 Size of coreset

$(1/\epsilon)^d) \rightarrow (1/\epsilon)^{d-1} \rightarrow (1/\epsilon)^{\frac{d-1}{2}}$. The last bound is a new result.

2.4 ϵ-coreset for k-center

- Given $P \subseteq \mathbb{R}^d$ and k

2.4.1 Show: Existential result $\exists$-coreset Q for k-center

- Known: Gonzalez alg (the algorithm mentioned before). $\rightarrow r_G \leq 2 \cdot r_{opt}$.
- Target: $|Q| = k(\frac{1}{\epsilon})^d$
- Error: $\epsilon \cdot r_{opt}$