Algorithmic Excursions: Topics in Computer Science II **Spring 2016**

Lecture 7 & 8 : Term Paper Topics and Clustering

Lecturer: Kasturi Varadarajan *Scribe: Jianshu Zhang*

Application and Term Paper Topics:
*Hint: topics with * are only recommended to students with special background*

Topic 1: ϵ-net : cutting, partition and geometric set cover.

Topic 2*: Guarantee size of ϵ-net with VC-dimension as d to $\frac{d}{\varepsilon}\log\frac{1}{\varepsilon}$.

Topic 3*: Improvements for Geometric set systems
Example 1: Points + Half planes in this system you can get ϵ-net of size $O(\frac{1}{\varepsilon})$.
Example 2: the same happens to Points + Half spaces in $\mathcal{R}^3$ system.
Example 3: Fat triangles + Stabbing in $\mathcal{R}^2$ system *–focusing on the set of triangles, pick a point, the triangles that content this point will be in the subset–* for this system, will get $O(\frac{1}{\varepsilon}\log(\log\frac{1}{\varepsilon}))$.

Topic 4: However, improvement is not possible in general, such as Points + Half spaces in $\mathcal{R}^4$, Rectangles or Normal triangles(not fat) in $\mathcal{R}^2$ + stabbing, they only could get $\Omega(\frac{1}{\varepsilon}\log\frac{1}{\varepsilon})$.

Topic 5: Suppose $(X, \mathcal{R})$, where $|X| = |\mathcal{R}| = n$, Disc:$\sqrt{n\log n}$ can be improved to $\sqrt{n}$.

Topic 6*: If Shatter function of $(X, \mathcal{R})$ is bounded by $C \times m^d$ for constants c and d, discrepancy can be improve to $n^{\frac{1}{2}-\frac{1}{2}d}$ *Example: Points + Half planes with shatter function* $\leqslant m^2$, *then* $n^{\frac{1}{2}-\frac{1}{4}} = n^{\frac{1}{4}}$.

Topic 7*: This yields improved ϵ-approximations: –Application of ϵps-approximation to Core Sets (going to talk about later) –VC-dimension, ϵps-approximation in learning(topic)

Topic 8: Bounding VC-dimension and shatter function for Geometric set systems

Topic 9*: Sampling to preserve other kinds of stuff *Example: Cut specification in Graphs.(Sample Graph need to preserve some information in Graphs)*

Topic 10: Deterministic construction of ϵps-approximation

Clustering – Chapter 4 in Geometric Approximation Algorithms

Definition 3.1 *Suppose we are given a set of points, and a distance function :* $d : P \times P(\textit{two points}) \longrightarrow \mathcal{R}^{+}(\textit{real number})$ *that defines a metric:*

- $d(p,q) = 0$*, if and only if* $p = q$
- $d(p,q) = d(q,p)$
- $d(p,\gamma) \leqslant d(p,q) + d(q,\gamma)$

Notation: For $P' \subseteq P$, $d(P',q) = \min_{p \in P'} d(p,q)$

$C_1 \leftarrow$ any point in P
for $i \leftarrow 2$ to n **do**
  $\gamma_{i-1} \leftarrow \max_{q \in P} d(\{C_1, C_2, \ldots, C_{i-1}\}, q)$
  $C_i \leftarrow \arg\max_{q \in P} d(\{C_1, C_2, \ldots, C_{i-1}\}, q)$
return $C_1, C_2, \ldots, C_n$

Suppose γ_5 is the furthest distance between points in $P \setminus \{C_1, \ldots, C_5\}$ to $\{C_1, \ldots, C_5\}$ which return from the algorithm. Then if we use $\{C_1, \ldots, C_5\}$ as centers and γ_5 as radius to make balls, the balls will content all the points in the point set, the balls could partition the points into clusters. Since $\{C_1\} \subseteq C_1, C_2 \subseteq \ldots \subseteq \{C_1, C_2, \ldots, C_n\}$, then $\gamma_1 \geqslant \gamma_2 \geqslant \cdots \geqslant \gamma_{n-1}$ and we define $\gamma_n = 0$.

Definition 3.2 *A set* $\mathcal{Q} \subseteq P$ *is called an* γ*-packing if the following properties holds:*

- *Covering Property: For any* $p \in P$, $d(\mathcal{Q}, p) \leqslant \gamma$
- *Separation Property: For any* $p_1, p_2 \in \mathcal{Q}$, $d(p_1, p_2) \geqslant \gamma$

We claim $\{C_1, \ldots, C_5\}$ is an γ_5-packing, and for any $1 \leqslant k \leqslant n$, $\{C_1, C_2, \ldots, C_k\}$ is an γ_k-packing.
Homework: Proof the conclusion above.

Definition 3.3 ***k-Center Clustering:***
Given P *and* $1 \leqslant k \leqslant |P|$*, compute a set* $C \subseteq P$ *with* k *points, So as to minimize:*

$$\lambda(C) := \max_{q \in P} d(C, q) \tag{3.1}$$

Alternatively, find the minimum λ_* *such that there exist* k *balls of radius* λ_* *that "Cover"* P.
Time expensive of this clustering method is $O(k^2 n)$

Claim 3.4 *Let* $C_1, C_2, \ldots, C_n$ *be a greedy permutation of* P *(Selected by the algorithm above, which* C_1 *is any point and* C_2 *is the furthest point to* $\{C_1\}$ *and so on.) For any* k*, and any* $\mathcal{C}$ *with* k *points,* $\lambda(\{C_1, C_2, \ldots, C_k\}) \leqslant 2\lambda(\mathcal{C})$

As we regard $\{C_1, C_2, \ldots, C_k\}$ as center of clusters and γ_k as the radius of each cluster, this is a clustering solution, which is not the best, but a OK solution. $\{C_1, C_2, \ldots, C_k\}$ *is a* γ_k*-rpacking.*

Proof: This is obvious if k = $|P|$.

$For \{C_1, C_2, \ldots, C_k\}$

$$\begin{array}{ccccc} \gamma_1 \geqslant & \gamma_2 \geqslant & \ldots \geqslant & \gamma_{k-1} \geqslant & \gamma_k \\ d(\{C_1\}, C_2) \geqslant & d(\{C_1, C_2\}, C_3) \geqslant & \ldots \geqslant & d(\{C_1, C_2, \ldots, C_{k-1}\}, C_k) & \end{array}$$

And $\lambda(\{C_1, C_2, \ldots, C_k\}) = \gamma_k$ From Algorithm

γ_k *is the furthest distance of a point to set* $\{C_1, C_2, \ldots, C_k\}$
Fix C with k points, we'll show $\lambda(\mathcal{C}) \geqslant \frac{\gamma_k}{2}$

$$\text{Map each point in } \{C_1, C_2, \ldots, C_{k+1}\} \text{ to the nearest point in } \mathcal{C}$$
$$\text{There exists two points } C_i \text{ and } C_j, \text{ that are mapped to some point } \bar{C} \in \mathcal{C}$$
$$\gamma_k \leqslant d(C_i, C_j) \leqslant d(C_i, \bar{C}) + d(C_j, (\bar{C})) \leqslant \lambda(\mathcal{C}) + \lambda(\mathcal{C}) \Rightarrow \lambda(\mathcal{C}) \geqslant \frac{\gamma_k}{2}$$

■

Definition 3.5 *K-median Clustering: Given P, metric d and 1* $\leqslant k \leqslant |P|$, *find a set* $\mathcal{C}$ *of k points that minimize:*

$$cost(\mathcal{C}) \equiv \sum_{q \in P} d(q, \mathcal{C})$$

* *k-center algorithm clustering is very easy to be influenced by noise*

1: $\mathcal{C} \leftarrow$ any subset of size k
2: **while** there exist $\bar{c} \in \mathcal{C}$ and $p \in P \setminus \mathcal{C}$ such that $cost(\mathcal{C} - \bar{c} + p) < cost(\mathcal{C})$ **do**
3: $\quad \mathcal{C} \leftarrow \mathcal{C} - \bar{c} + p$
4: **return** $\mathcal{C}$

Homework: Show an example where the above algorithm fails to com up with optimal solution.

Notation:

$$L - \text{Solution returned by local search}$$
$$C_{opt} - \text{optimal solution}$$

We'll show $cost(L) \leqslant 5\, cost(C_{opt})$