Lecture 8:

Sublinear time algorithms for problems in metric spaces

Course: Algorithms for Big Data

Instructor: Hossein Jowhari

Department of Computer Science and Statistics
Faculty of Mathematics
K. N. Toosi University of Technology

Spring 2021

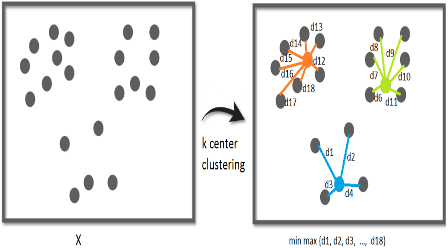
Outline

- k-center problem
- Approximating the diameter
- Approximating the average distance

The k-center problem

$$X = \{x_1, \dots, x_n\}$$

$$\min_{c_1,\dots,c_k \subseteq X} \max_{x \in X} \min_{j=1,\dots,k} \{\text{dist}(x, c_j)\}$$

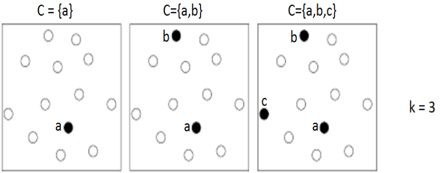


Facts: k-center problem is NP-Hard even for points $\mathbb{R}^2$. No polynomial-time algorithm with approximation factor better than 2 exists unless $P = NP$. There is a 2-factor approximation for k-center that runs in $O(nk)$ time when the distance function satisfy symmetry and triangle inequality.

An Approximation Algorithm

Algorithm: Initially choose a point $x \in X$. Let cluster centers $C = \{x\}$. Repeat the following:

Every time choose a point $y \in X$ that is farthest away from C and add y to C. Stop when $|C| = k$. Output C as the chosen centers.



Lemma: $cost(C) \leq 2cost(OPT)$

For proof, see the references.

Running time analysis: depends on data representation.

- (General metrics) when the distance function is represented by a $n \times n$ symmetric matrix. Input size is n^2.

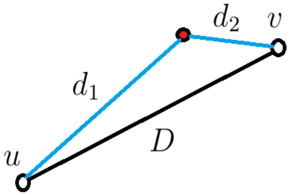
	1	2	3	4	5	6	7	8
1	0	12	3	23	1	5	32	56
2	12	0	9	18	3	41	45	5
3	3	9	0	89	56	21	12	49
4	23	18	89	0	87	46	75	17
5	1	3	56	87	0	55	22	86
6	5	41	21	46	55	0	21	76
7	32	45	12	75	22	21	0	11
8	56	5	49	17	86	76	11	0

At every stage of the algorithm, we find the farthest point in X from C. This takes $O(n)$ time. (Why?)

In total, we have at most k stages. Therefore the running time is $O(nk)$. Sublinear when $k = o(n)$

- (d-dimensional points) for example $X \in \mathbb{R}^d$ with the Euclidean distance. Here the input size is $O(nd)$. In this case, the running time is bounded by $O(ndk)$. Not sublinear!

Estimating the diameter



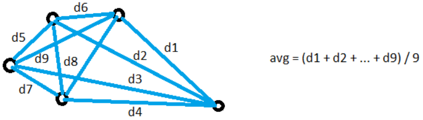
$$\text{dist}(u, v) = D$$

$$\frac{1}{2}D \le \max\{\, d_1, d_2 \,\} \le D$$

A $\frac{1}{2}$-factor approximation algorithm: Select any point $x \in X$. Check all distances $\{\text{dist}(x, u)\}_{u \in X}$. Output the maximum distance.

Running time: $O(n)$ for general metrics. $O(nd)$ for $X \in \mathbb{R}^d$ and Euclidean distance.

Average distance in a finite metric space



$$X = \{x_1, \ldots, x_n\}$$

$$A = \sum_{i,j} \mathrm{dist}(x_i, x_j), \qquad \mathrm{avg} = \frac{A}{\binom{n}{2}}$$

Assumption: The finite metric (X, dist) is given by its $n \times n$ distance matrix.

The trivial algorithm computes the sum of distances A exactly in $O(n^2)$ time.

Estimating the average distance

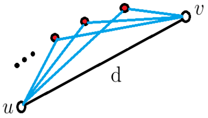
- As we observed earlier approximating the average of m arbitrary values $a_1, \ldots, a_m$ requires $\Omega(m)$ samples
- When the values $a_1, \ldots, a_m$ are degrees of a m-vertex graph, we saw that $O(\epsilon^{-1}\sqrt{m})$ samples was enough to get a $2 + \epsilon$ factor approximation of the average degree.
- How about when the values $a_1, \ldots, a_m$ are $m = \binom{n}{2}$ distances in a finite metric space defined over n points?
- Can we approximate the average distance using $o(n^2)$ samples?

Estimating the average distance

Theorem [P. Indyk 1999]. When the values $a_1, \ldots, a_m$ are $m = \binom{n}{2}$ distances in a finite metric space on n points, $O(\frac{n}{\epsilon^{3.5}})$ uniform independent samples are enough to get $1 + \epsilon$ factor approximation of the average distance.

Note: This gives a $O(\frac{n}{\epsilon^{3.5}})$ time randomized algorithm for estimating the average distance within $1 + \epsilon$ factor.

Main observation: In a finite metric space on n points if $\operatorname{dist}(x, y) = d$ then there are at least n distances with value at least $d/2$.



Reviewing Indyk's analysis

Assumption: All distances fall in the range $[1, D]$. D is the diameter of the metric.

Notation: Let $c = 1 + \epsilon$ where $\epsilon > 0$.

Assumption: D is a power of c ($D = c^k$ for some k.)

We split the interval $[1, Dc]$ into sub-intervals

$$I_0 = [c^0, c^1), \quad I_1 = [c^1, c^2), \quad \ldots \quad, I_k = [D, Dc)$$

$$n_i = \text{number of distances in the interval } I_i$$

$$s_i = \text{number of sample distances in the interval } I_i$$

Reviewing Indyk's analysis

Note: We are estimating the sum $A = \sum_{i,j} \mathrm{dist}(x_i, x_j)$

Definition: Let $\tilde{A} = \sum_i n_i c^i$

Observation: $\frac{A}{1+\epsilon} \le \tilde{A} \le A$

Let S be the set of sampled distances. Let $s = |S|$ and $m = \binom{n}{2}$. The algorithm outputs

$$A' = \frac{m}{s} \sum_{(i,j) \in S} \mathrm{dist}(x_i, x_j).$$

Definition: Let $\tilde{A}' = \frac{m}{s} \sum_i s_i c^i$

Observation: $\frac{A'}{1+\epsilon} \le \tilde{A}' \le A'$

Observation: Therefore it is enough to show that

$$\tilde{A}' = \frac{m}{s}\sum_{i}^{k} s_i c^i \quad \approx \quad \tilde{A} = \sum_{i}^{k} n_i c^i$$

Lemma $E[\tilde{A}'] = \tilde{A}$

Proof $E[\tilde{A}'] = \frac{m}{s}\sum_i c^i E[s_i] = \frac{m}{s}\sum_i c^i (\frac{n_i}{m} s) = \tilde{A}$

Lemma $Var[\tilde{A}'] \leq \frac{m}{s}\sum_i n_i c^{2i}$

By Chebyshev Inequality,

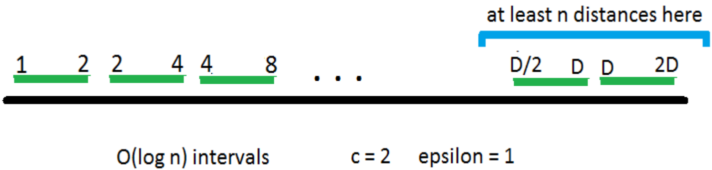
$$P = Pr[|\tilde{A}' - \tilde{A}| \geq \epsilon \tilde{A}] \leq \frac{Var[\tilde{A}']}{\epsilon^2 E^2[\tilde{A}']} \leq \frac{\frac{m}{s}\sum_i n_i c^{2i}}{\epsilon^2 (\sum_i n_i c^i)^2} \leq \frac{\frac{m}{s}}{\epsilon^2} \overbrace{\left(\frac{\sum_i n_i c^{2i}}{\sum_i n_i^2 c^{2i}}\right)}^{F}$$

We need to bound

$$F = \frac{\sum_i n_i c^{2i}}{\sum_i n_i^2 c^{2i}}$$

Here we use the properties of the metric space.

Observation: Suppose $\mathrm{dist}(x, y) = D$. Then there are at least n distances with value at least $\frac{D}{2}$.



We show $F = O(\frac{1}{n})$

$$D \in I_k = [D, Dc)$$

Corollary: by Pigeonhole Principle there must be an interval I_{k-j} where $0 \le j \le \log_c 2$ where $n_{k-j} \ge \frac{n}{\log_c 2}$

$$I_0, \ldots, \quad \overbrace{I_{k-\log_c 2}, \ldots, I_k}^{\text{contains at least } n \text{ distances}}$$

Note: $\frac{D}{2}$ falls in the interval $I_{n-\lfloor \log_c 2 \rfloor}$ because if we set $\frac{D}{2} = \frac{D}{c^i}$

Let $t = \alpha n$ for some $\alpha > 0$. We define $B = \{i : n_i \ge t\} - \{k - j\}$

$$N_1 = \sum_{i \in B} n_i c^{2i}, \qquad N_2 = \sum_{i \notin B} n_i c^{2i}$$

$$M_1 = \sum_{i \in B} n_i^2 c^{2i}, \qquad M_2 = \sum_{i \notin B} n_i^2 c^{2i}$$

$$F = \frac{N_1 + N_2}{M_1 + M_2} \leq \max\{\frac{N_1}{M_1}, \frac{N_2}{M_2}\}$$

Observation: $\frac{N_1}{M_1} \leq \frac{1}{t}$

Observation: $N_2 \leq t \sum c^{2i} \leq t \frac{c^{2k+1}}{c^2 - 1} \leq t \frac{D^2(1+\epsilon)^2}{\epsilon}$

Observation: $M_2 \geq (\frac{D}{2} \frac{n}{\log_c 2})^2$

Corollary: $\frac{N_2}{M_2} \leq \frac{1}{n} \frac{4 \log_c^2 2\alpha(1+\epsilon)^2}{\epsilon}$

Corollary: $F \leq \max\{\frac{N_2}{M_2}, \frac{N_1}{M_1}\} \leq \frac{1}{n} \max\{\frac{4 \log_c^2 2\alpha(1+\epsilon)^2}{\epsilon}, \frac{1}{\alpha}\}$

We set $\alpha = \Theta(\epsilon^{3/2})$ and we obtain

$$F = O(\frac{1}{\epsilon^{3/2}} \frac{1}{n})$$

Therefore

$$P = Pr[|\tilde{A}' - \tilde{A}| \geq \epsilon \tilde{A}] \leq \frac{\frac{m}{s}}{\epsilon^2} F = O(\frac{\binom{n}{2}}{sn\epsilon^{3.5}}) < \frac{1}{4}$$

We get $s = \Omega(\frac{n}{\epsilon^{3.5}})$ is enough.

References

- P. Indyk. Sublinear time algorithms for metric space problems. STOC 99.
- T. Gonzales. Clustering to minimize the maximum inter-cluster distance. Theoretical Computer Science. 1985.


سال نو مبارک
Nowruz . Happy new year.