6.889 Sublinear Time Algorithms September 1, 2020

Lecture 1

Lecturer: Ronitt Rubinfeld *Scribe: Mehtaab Sawhney*

1 Outline

The following topics were addressed during the first lecture.

- Overview of the Course/Sublinear Algorithms
- Diameter of a Point Set
- Number of Connected Components in a Graph

We refer the reader to the slides on the course homepage for the first item.

2 Diameter of a Point Set

Our first example of a sublinear algorithm (due to Piotr Indyk) will be computing a 2-approximation to the diameter of a point set in sublinear time. This algorithm has the unique property of being the only *deterministic* algorithm in this class.

Input - We are given m points described by a distance matrix $\mathcal{D}$ such that $\mathcal{D}_{i,j}$ is the distance from i to j. Furthermore we are guaranteed that the distances satisfy

- (Symmetry) $\mathcal{D}_{i,j} = \mathcal{D}_{j,i}$ for all $i, j \in [m]$
- (Triangle Inequality) $\mathcal{D}_{i,j} \le \mathcal{D}_{i,k} + \mathcal{D}_{k,j}$ for all $i, j, k \in [m]$

Note here that the input size is $n = \Theta(m^2)$ as we are given all pairs of distances.

Output - Let the diameter $D = \max_{i,j} \mathcal{D}_{i,j}$. Our output is a pair (k, ℓ) such that $\mathcal{D}_{k,\ell} \ge D/2$ (e.g. a 2-approximation to the diameter)

Algorithm - Choose an arbitrary index k. Output (k, ℓ) such that $D_{k,\ell}$ is maximized. (The psuedocode for the algorithm is given below.)

Algorithm 1: Diameter-Estimator

```
Pick k arbitrarily from {1, ··· , m} ;
ℓ = argmax_j D_{k,j};
Return k, ℓ, D_{k,ℓ}
```

Running Time - Note that we read only $O(m) = O(\sqrt{n})$ entries of the distance matrix $\mathcal{D}$.

Correctness - Let $D = \mathcal{D}_{i,j}$. Now note that

$$\begin{aligned}
\mathcal{D}_{i,j} &\le \mathcal{D}_{i,k} + \mathcal{D}_{k,j} && \text{[Triangle Inequality]}\\
&\le \mathcal{D}_{k,i} + \mathcal{D}_{k,j} && \text{[Symmetry]}\\
&\le \mathcal{D}_{k,\ell} + \mathcal{D}_{k,\ell} && \text{[Definition of } \ell\text{]}\\
&= \mathcal{D}_{k,\ell}.
\end{aligned}$$

The desired result follows immediately.

Lower Bound - We now sketch an argument that any $(2-\delta)$ approximation to the diameter requires reading the entire matrix $\mathcal{D}$. (This answers a question raised by one of the students in class.)

- Define the distance matrix $\mathcal{M}$ to have $\mathcal{M}_{i,i} = 0$ and $\mathcal{M}_{i,j} = 1$ otherwise.

- Define the distance matrix $\mathcal{N}^{i,j}$ to be identical to $\mathcal{M}$ except $\mathcal{N}^{i,j}_{i,j} = \mathcal{N}^{i,j}_{j,i} = (2-\delta)$.
- It can easily be checked that $\mathcal{M}, \mathcal{N}^{i,j}$ satisfy the triangle inequality and symmetry. Furthermore, even if one is given the promise that the distance matrix $\mathcal{D}$ is one of the $\binom{m}{2}+1$ examples given it takes take $\Theta(m^2)$ time to tell if any of the entries is larger than 1 giving the desired lower bound as $\mathcal{N}^{i,j}$ has diameter $2-\delta$ while $\mathcal{M}$ has diameter 1.

3 Number of Connected Components in a Graph

Our second example of a (randomized) sublinear time algorithm that will be an εn-approximation the the diameter of an input graph G in time $\mathrm{poly}(1/\varepsilon)$.

Input - We are given $G = (V, E)$ in an adjacency list representation. As is standard, we will let $n = |V|$ and $m = |E|$.

Output - Let C denote the number of connected components. We will output $\widehat{C}$ such $|C - \widehat{C}| \leq \varepsilon n$ with probability $3/4$.

The first key insight we will need is an alternate characterization of the number of connected components of a graph G.

Lemma 1 *Fix a graph $G = (V, E)$. For a vertex $v \in V$, let n_v denote the number of vertices in the connected component of v and let C be the total number of connected components. Then we have that*

$$C = \sum_{v \in V} \frac{1}{n_v}.$$

Proof By splitting G into connected components, it suffices to prove the claim for a graph G which is connected. However, in this case, note that $n_v = |V|$ and therefore

$$\sum_{v \in V} \frac{1}{n_v} = |V| \left(\frac{1}{|V|} \right) = 1$$

as desired. ■

One naive attempt given this characterization is to simply sample small number of vertices v at random from the graph G, compute n_v for each sampled vertex, and output n the average of $1/n_v$ over the vertices sampled. However, there is a large issue in that computing n_v already is already takes linear time! The second insight therefore is to realize that if n_v is large, $1/n_v$ is small and therefore we do not need to compute n_v as precisely.

Lemma 2 *Let*

$$\widehat{n_v} = \min(n_v, 2/\varepsilon).$$

We have that

$$\left| \sum_{v \in V} \frac{1}{n_v} - \sum_{v \in V} \frac{1}{\widehat{n_v}} \right| \leq \frac{\varepsilon n}{2}$$

and that for a given vertex v, $\widehat{n_v}$ can be computed in $O(1/\varepsilon^2)$ time.

Proof We first prove that

$$\left| \frac{1}{n_v} - \frac{1}{\widehat{n_v}} \right| \leq \frac{\varepsilon}{2};$$

the first claim then follows by noting that by triangle inequality

$$\left| \sum_{v \in V} \frac{1}{n_v} - \sum_{v \in V} \frac{1}{\widehat{n_v}} \right| \leq \sum_{v \in V} \left| \frac{1}{n_v} - \frac{1}{\widehat{n_v}} \right| \leq n \cdot \frac{\varepsilon}{2}.$$

To prove that

$$\left|\frac{1}{n_v} - \frac{1}{\widehat{n_v}}\right| \leq \frac{\varepsilon}{2}$$

we split into cases based on the size of n_v.

- If $n_v \leq \frac{2}{\varepsilon}$, we are done immediately as $n_v = \widehat{n_v}$.
- If $n_v \geq \frac{2}{\varepsilon}$, note that $\frac{1}{n_v} \leq \frac{1}{\widehat{n_v}}$ and $\widehat{n_v} = \frac{2}{\varepsilon}$ and therefore

$$\left|\frac{1}{n_v} - \frac{1}{\widehat{n_v}}\right| = \frac{1}{\widehat{n_v}} - \frac{1}{n_v} \leq \frac{1}{\widehat{n_v}} = \frac{2}{\varepsilon}.$$

Now in order to compute the $\widehat{n_v}$ in $\Theta(1/\varepsilon^2)$ time we simply run BFS starting at the vertex v and output the number of vertices in the corresponding component, short-cutting if we ever have processed more than $2/\varepsilon$ vertices. Note that if the connected component of v is less than $2/\varepsilon$ vertices we will read the entire component in $O(1/\varepsilon^2)$-time and thus we are able to compute n_v and thus $\widehat{n_v}$ exactly. Otherwise we have $n_v \geq \frac{2}{\varepsilon}$ and the BFS will short-circuit after reading $2/\varepsilon$ vertices and we will compute (correctly) that $\widehat{n_v} = \frac{2}{\varepsilon}$. For the running time in this case note that we only process $2/\epsilon$-vertices and for each vertex we only process at most $2/\epsilon$ vertices in total (as otherwise we can short-circuit). ■

Given the above we are now in position to state our algorithm.

Algorithm - Choose $s = \Theta(1/\varepsilon^2)$ vertices $v_1, \ldots, v_s$ uniformly at random from the the vertices of G. Compute $\widehat{n_{v_i}}$ for $i \in [s]$ and return

$$\widehat{C} := \frac{n}{s}\left(\sum_{i \in [s]} \frac{1}{\widehat{n_{v_i}}}\right).$$

(The psuedocode for the algorithm is given below.)

Algorithm 2: Connected Components-Estimator

```
sum ← 0;
for 1 ≤ i ≤ s do
    Sample v_i uniformly from V;
    sum ← sum + 1/n̂_{v_i};
Ĉ ← (n/s)(sum) return Ĉ
```

Running Time - The running time is dominated by computing $\widehat{n_{v_i}}$ for sampled vertices n_{v_i}. There are $\Theta(1/\varepsilon^2)$ vertices and each run takes $\Theta(1/\varepsilon^2)$-times giving a total running time of $\Theta(1/\varepsilon^4)$.

Correctness - In order to prove correctness it essentially suffices by Lemma 2 to prove that

$$\frac{1}{s}\sum_{i \in [s]} \frac{1}{\widehat{n_v}} \approx \frac{1}{n}\sum_{v \in V} \frac{1}{\widehat{n_v}};$$

the key tool here will be Chernoff bounds.

Theorem 3 (Chernoff Bounds) *Fix $\delta \in [0, 1]$. Let X_i be iid random variables in $[0, 1]$ with $p = \mathbb{E}[X_i]$. Let $X = \sum_{i=1}^{r} X_i$ and $\mu = \mathbb{E}[X] = rp$. Then*

$$\mathbb{P}[|X - \mu| \geq \delta\mu] = \mathbb{P}[|X - rp| \geq \delta rp] \leq \exp(-\Theta(\delta^2 rp)).$$

Theorem 4 *Let C be the number of connected components of G. The output of Algorithm 2, $\widehat{C}$, satisfies that*

$$\mathbb{P}\left[\left|C - \widehat{C}\right| \geq \varepsilon n\right] \leq \frac{1}{4}.$$

Proof By the first part of Lemma 2 and triangle inequality it suffices to prove that

$$\mathbb{P}\left[\left|\sum_{v\in V} \frac{1}{\widehat{n_v}} - \widehat{C}\right| \geq \frac{\varepsilon n}{2}\right] \leq \frac{1}{4}.$$

Note by definition that

$$\widehat{C} = \frac{n}{s}\left(\sum_{i\in[s]} \frac{1}{\widehat{n_{v_i}}}\right)$$

and therefore the desired claim is equivalent to

$$\mathbb{P}\left[\left|\sum_{v\in V} \frac{1}{\widehat{n_v}} - \frac{n}{s}\left(\sum_{i\in[s]} \frac{1}{\widehat{n_{v_i}}}\right)\right| \geq \frac{\varepsilon n}{2}\right] \leq \frac{1}{4}.$$

This is equivalent to the expression

$$\mathbb{P}\left[\left|\left(\sum_{i\in[s]} \frac{1}{\widehat{n_{v_i}}}\right) - \frac{s}{n}\sum_{v\in V} \frac{1}{\widehat{n_v}}\right| \geq \frac{\varepsilon s}{2}\right] = \mathbb{P}\left[\left|\left(\sum_{i\in[s]} \frac{1}{\widehat{n_{v_i}}}\right) - \mathbb{E}\left[\left(\sum_{i\in[s]} \frac{1}{\widehat{n_{v_i}}}\right)\right]\right| \geq \frac{\varepsilon s}{2}\right] \leq \frac{1}{4}.$$

Note that we have simply applied linearity of expectation at this stage. This is precisely the setup for Chernoff-bounds and now it is simply a matter of picking parameters appropriately.

First note that expected summand is at least $\epsilon/2$ as we always have $1/\widehat{n_v} \geq \frac{\epsilon}{2}$ and thus $p \geq \frac{\epsilon}{2}$. Now choosing $\delta = \frac{\epsilon}{2p} \leq 1$ we find that

$$\mathbb{P}\left[\left|\left(\sum_{i\in[s]} \frac{1}{\widehat{n_{v_i}}}\right) - \mathbb{E}\left[\left(\sum_{i\in[s]} \frac{1}{\widehat{n_{v_i}}}\right)\right]\right| \geq \frac{\varepsilon s}{2}\right] \leq \exp(-\Theta(\delta^2 sp)) = \exp(-\Theta(\varepsilon^2 s/(4p))) \leq \exp(-\Theta(\varepsilon^2 s)).$$

Note in the final step we have used that $p \leq 1$ which follows as $1/\widehat{n_v} \in [0,1]$. Thus taking s a sufficiently large multiple of $\Theta(1/\varepsilon^2)$ the result follows. ■