Lecture 1

Lecturer: Ronitt Rubinfeld *Scribe: Yuancheng Yu*

1 Diameter of a point set

Given distance matrix D of m points, where $D_{ij} = D_{ji}$ is the distance between i and j, and the triangle inequality is satisfied, i.e., for any i, j, k, $D_{ij} \le D_{ik} + D_{kj}$, let $\max D_{ij}$ be the *diameter* of the point set. Output (k, l) such that D_{kl} approximates the diameter.

1.1 2-approximation algorithm

For some arbitrary $k \in \{1, \dots, m\}$, find l that maximizes D_{kl}, and output (k, l).

Running time It takes $O(m) = O(\sqrt{n})$ time.

Correctness Let D_{ij} be the diameter, then $D_{kl} = \frac{1}{2}(D_{kl} + D_{kl}) \ge \frac{1}{2}(D_{ki} + D_{kj}) = \frac{1}{2}(D_{ik} + D_{kj}) \ge \frac{1}{2}D_{ij}$, i.e., D_{kl} is a 2-approximation of the diameter.

2 Approximation of the number of connected components

Given a graph $G(V, E)$ (adjacency list representation), max degree d, and ϵ, output y such that $|y - c| \le \epsilon n$ with high probability, where c is the number of connected components (additive approximation to within ϵn).

2.1 A different characterization of the number of components

For any node v, let n_v be the size of v's component. Observe that for any component $A \subset V$,

$$\sum_{u \in A} \frac{1}{n_u} = \sum_{u \in A} \frac{1}{|A|} = 1,$$

and hence the number of components

$$c = \sum_{u \in V} \frac{1}{n_u}.$$

Computing $\frac{1}{n_u}$ and summation over n terms both need $O(n)$ time; can we give a good estimate faster?

2.2 Estimating $c = \sum_{u \in V} \frac{1}{n_u}$

We would like to estimate $\frac{1}{n_u}$ quickly and estimate $\sum_u \frac{1}{n_u}$ via sampling bounds. Let $\hat{n_u} \equiv \min\{n_u, \frac{2}{\epsilon}\}$, $\hat{c} \equiv \sum_{u \in V} \frac{1}{\hat{n}_u}$.

Lemma 1 *$\hat{n}_u$ is a "good" estimate, i.e., for any u,*

$$\left| \frac{1}{\hat{n}_u} - \frac{1}{n_u} \right| \le \frac{\epsilon}{2}.$$

Proof If $n_u \le \frac{2}{\epsilon}$, $\hat{n}_u = n_u$. Otherwise $n_u > \frac{2}{\epsilon} = \hat{n}_u$, and $\frac{\epsilon}{2} = \frac{1}{\hat{n}_u} > \frac{1}{n_u} > 0$, so $\left| \frac{1}{\hat{n}_u} - \frac{1}{n_u} \right| \le \frac{\epsilon}{2}$.

Corollary 2 *$\hat{c}$ is a "good" estimate, i.e.,*

$$|c - \hat{c}| \leq \sum_{u \in V} \left| \frac{1}{n_u} - \frac{1}{\hat{n}_u} \right| \leq \frac{\epsilon n}{2}$$

This estimation is useful if we can compute $\hat{c}$ faster.

2.3 Algorithm to compute $\hat{n}_u$

Run BFS from u for $\frac{2}{\epsilon}$ steps (stop if the entire component is visited), and output the number of nodes visited.

Correctness If the entire component is visited, $n_u \leq \frac{2}{\epsilon}$, and the output is $n_u = \hat{n}_u$. Otherwise $n_u > \frac{2}{\epsilon}$, and the output is $\frac{2}{\epsilon} = \hat{n}_u$.

Runtime Since each BFS step takes $O(d)$ time, we can compute $\hat{n}_u$ in $O(\frac{d}{\epsilon})$ time.

Summing all $\hat{n}_u$ gives a linear time algorithm. If we can estimate the average component size faster, we can simply multiply it by n.

2.4 Algorithm to estimate $\hat{c}$

Let $r = \frac{b}{\epsilon^3}$ for some constant b to be determined, sample r random nodes $u_1, \dots, u_r$, compute $\hat{n}_{u_i}$ for $i = 1, \dots, r$, and output $\tilde{c} = \dfrac{n}{r} \displaystyle\sum_{i=1}^{r} \frac{1}{\hat{n}_{u_i}}$

Runtime $O\left(\frac{1}{\epsilon^3}\right) O\left(\frac{d}{\epsilon}\right) = O\left(\frac{d}{\epsilon^4}\right)$.

Theorem 3 (Chernoff Bound) *$X_1, \dots, X_r$ iid, $X_i \in [0,1]$, $S = \sum_{i=1}^{r} X_i$, $p = E[X_i] = E[s]/r$, then*

$$\Pr\left[\left|\frac{s}{r} - p\right| \geq \delta p\right] \leq e^{-\Omega(rp\delta^2)}.$$

Theorem 4 *$\Pr\left[|\tilde{c} - \hat{c}| \leq \frac{\epsilon n}{2}\right] \geq 3/4$.*

Proof Let $X_i = \dfrac{1}{\hat{n}_{u_i}}$, $p = E\left(\dfrac{1}{\hat{n}_{u_i}}\right) = \dfrac{1}{n} \displaystyle\sum_{u \in V} \frac{1}{\hat{n}_{u_i}} = \dfrac{\hat{c}}{n}$, $\delta = \dfrac{\epsilon}{2}$, $\dfrac{s}{r} = \dfrac{1}{r} \displaystyle\sum_{i=1}^{r} \frac{1}{\hat{n}_{u_i}} = \dfrac{\tilde{c}}{n}$, by Chernoff

$$\Pr\left[\left|\frac{\tilde{c}}{n} - \frac{\hat{c}}{n}\right| \geq \frac{\epsilon}{2}\frac{\hat{c}}{n}\right] = \Pr\left[|\tilde{c} - \hat{c}| \geq \frac{\epsilon}{2}\hat{c}\right] \leq \exp\left(-\frac{b}{\epsilon^3}\frac{\hat{c}}{n}\frac{\epsilon^2}{4}\right) \leq \exp\left(\frac{b}{\epsilon}\frac{\epsilon}{2}\frac{1}{4}\right) < \frac{1}{4},$$

when we pick $b \geq 16$, and where $\hat{c} = \sum_u \frac{1}{\hat{n}_u} \geq \frac{\epsilon}{2} n$ since $\frac{1}{\hat{n}_u} \geq \frac{\epsilon}{2}$.

Corollary 5 *: $\Pr[|c - \tilde{c}| \leq \epsilon n] \geq \frac{3}{4}$.*

Proof If $|\tilde{c} - \hat{c}| \leq \frac{\epsilon n}{2}$, by triangle inequality, $|c - \tilde{c}| \leq |c - \hat{c}| + |\hat{c} - \tilde{c}| \leq \frac{\epsilon n}{2} + \frac{\epsilon n}{2} = \epsilon n$, so

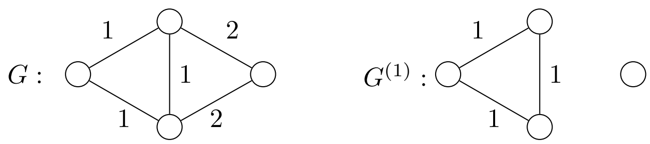
$$\Pr[|c - \tilde{c}| \leq \epsilon n] \geq \Pr\left[|\tilde{c} - \hat{c}| \leq \frac{\epsilon n}{2}\right] \geq \frac{3}{4}.$$

3 Approximating Minimum Spanning Tree (MST)

Given a connected graph $G(V, E)$ (adjacency list representation), max degree d, edge weights $w_{uv} \in \{1, \ldots, w\} \cup \{\infty\}$ ($w_{uv} = \infty \iff (u, v) \notin E$), and ϵ, output $\hat{M} \in [(1-\epsilon)M, (1+\epsilon)M]$, where M is the weight of the MST. Note that the weight range implies that $n - 1 \leq M \leq w(n-1)$.

3.1 A different characterization of MST

Let $E^{(i)} = \{(u, v) \mid w_{uv} \in \{1, \ldots, i\}\}$, $G^{(i)} = (V, E^{(i)})$, and $C^{(i)}$ be the number of components in $G^{(i)}$. For example, when $w = 1$, $G^{(1)} = G$, and $M = n - 1$ since G is connected. For $w = 2$ such as below,



The idea of Kruskal's algorithm is to use as many weight 1 edges as possible, and only use $C^{(1)} - 1$ weight 2 edges to connect the components in $G^{(1)}$. Since the $n - 1$ edges of the MST have weight at least 1, and $C^{(1)} - 1$ of them have additional weight $2 - 1 = 1$, the total weight of the MST is

$$M = (n-1) + \left(C^{(1)} - 1\right) = n - 2 + C^{(1)}$$

Claim 6 *In general, $M = n - w + \sum_{i=1}^{w-1} C^{(i)}$.*

Proof Let α_i be the number of weight i edges in any MST of G (Kruskal's algorithm implies that all MSTs have the same α_i). Then $\sum_{i>l} \alpha_i = C^{(l)} - 1$, where $\sum_{i=1}^{w} \alpha_i = C^{(0)} - 1 = n - 1$, and

$$M = \sum_{i=1}^{w} i\alpha_i = \sum_{i=1}^{w} \alpha_i + \sum_{i=2}^{w} \alpha_i + \cdots + \alpha_w = n - 1 + c^{(1)} - 1 + \cdots + c^{(w-1)} - 1 = n - w + \sum_{i=1}^{w-1} c^{(i)}.$$

3.2 Approximation Algorithm

For $i = 1$ to $w - 1$, approximate the number of components $\hat{C}^{(i)}$ to within $\frac{\epsilon}{2w} n = \epsilon' n$ additive error. Output $\hat{M} = n - 2 + \sum_{i=1}^{w-1} \hat{C}^{(i)}$.

Runtime Each number of components approximation takes $\tilde{O}(d/\epsilon'^4) = \tilde{O}(dw^4/\epsilon^4)$ time (the $\epsilon' = \frac{\epsilon}{2w}$ error introduces poly $\left(\log \frac{w}{\epsilon}\right)$ overhead), and the total runtime is $\tilde{O}(dw^5/\epsilon^4)$.

Note that to compute $G^{(i)}$, we can simply ignore edges with weights greater than i. The runtime can be improved to $O(dw \log(dw/\epsilon)/\epsilon^2)$ and has a lower bound of $\Omega(dw/\epsilon^2)$.

Approximation guarantee Approximate the number of components $\hat{C}^{(i)}$ within ϵ' error with probability at least $1 - 1/(4w)$. Then by union bound, the probability that all $w - 1$ approximations are within ϵ' error is at least $1 - w/(4w) = 3/4$. And $\left|M - \hat{M}\right| \leq w \frac{\epsilon}{2w} n = \frac{\epsilon n}{2}$, a small additive error. Since all weights are at least 1, $M \geq n - 1 \geq n/2$, and $\left|M - \hat{M}\right| \leq \epsilon M$, a small multiplicative error.

Remark The runtime depends only on $d, w, 1/\epsilon$, and we can bound additive/multiplicative errors.