CS 590 Sublinear Algorithms Jan 26 & 28, 2021

Lecture 3 & 4: Estimating MST and Average Degree

Lecturer: Elena Grigorescu *Scribe: Raymond Song*

In lecture 2, we

- Tested connectivity in sparse graphs;
- Estimated number of connected components in graphs;
- Began estimating the minimal spanning tree (MST) of graphs.

In lecture 3, we will finish MST estimation and begin estimating average degree of graphs.

1 Estimating MST

Recall that in lecture 2, we proved the following result:

Theorem 1 *One can estimate the number of connected components in a graph upto an εn-additive approximation with probability $\frac{2}{3}$ and $\text{poly}(\frac{1}{\varepsilon})$ samples, or with probability $1-\delta$ and $\text{poly}(\frac{1}{\varepsilon}, \log\frac{1}{\delta})$ samples.*

We will use this result and the algorithm associated with it as a black box for the purpose of estimating the weight of the MST, $w(MST)$.

At the end of lecture 2, we also made the following observation:

Lemma 2 *Let G be a weighted graph where each edge weighs $\leq w$, and let G_i be a subgraph of G restricted to edges of weight $\leq i$. Let C_i be the amount of connected components of G_i. Let T be a MST of G. Then, the number of edges of weight $> i$ in T is equal to $C_i - 1$.*

Kruskal's algorithm provides an intuition to the correctness of this lemma. If any of T's edges with weight $> i$ is one that connects two vertices in the same component in G_i, T would not be a MST as Kruskal's algorithm will choose the other edges of the cycle. Thus any such edge can only connect different components of G_i, resulting in $C_i - 1$ possible such edges.

Corollary 3 *Let N_i denote the number of edges of weight i in T. Then*

$$\sum_{j \geq i+1} N_j = C_i - 1$$

This observation leads to the next lemma:

Lemma 4

$$w(MST(G)) = \sum_{i=0}^{w-1}(C_i - 1) = n - w + \sum_{i=1}^{w-1} C_i$$

where the latter equality is due to $C_0 = n$.

This lemma enables us to use the blackbox algorithm for estimating connected components to estimate each individual C_i and thus estimate the weight of the MST.

Proof By definition of weights, we have

$$w(MST(G)) = \sum_{i=1}^{w} iN_i$$

We can rewrite and expand this expression as follows:

$$\begin{aligned}
w(MST(G)) &= \sum_{i=1}^{w} iN_i \\
&= \sum_{i=1}^{w} (1 + 1 + \cdots + 1 \textit{(i times)}) N_i \\
&= N_w + N_{w-1} + \cdots + N_2 + N_1 \\
&+ N_w + N_{w-1} + \cdots + N_2 \\
&+ \cdots \\
&+ N_w + N_{w-1} \\
&+ N_w \\
&= C_0 - 1 + C_1 - 1 + \cdots + C_{w-1} - 1 \\
&= n - w + \sum_{i=1}^{w-1} C_i
\end{aligned}$$

as desired. ■

We can thus propose the following algorithm for approximating $w(MST)$. Given graph G, ε, weights w, do the following: For each $1 \leq i \leq w-1$ estimate C_i as $\hat{C}_i$ by running the blackbox algorithm above with parameters $\varepsilon' = \frac{\varepsilon}{w}$ and $\delta' = \frac{1}{3w}$, and outputing

$$\hat{w}(MST) = n - w + \sum_{i=1}^{w-1} \hat{C}_i$$

1.1 Analysis of MST Estimation Algorithm

Recall that for a fixed i, the blackbox algorithm gives an estimation $\hat{C}_i$ with the guarantee that $|\hat{C}_i - C_i| \leq \varepsilon' n = \frac{\varepsilon}{w} n$ with probability at least $1 - \delta' = 1 - \frac{1}{3w}$.

Consider then the event that for a specific i, $\hat{C}_i$ is estimated badly, or equivalently, $|\hat{C}_i - C_i| > \frac{\varepsilon}{w} n$. This event happens with probability at most $\frac{1}{3w}$. By the union bound over all w values of i, we have that

$$Pr[\text{at least one bad estimate}] \leq w \cdot \frac{1}{3w} = \frac{1}{3}$$

and

$$Pr[\text{all estimates are good}] \geq \frac{2}{3}$$

Conditioned on the event that all estimates are good, we can assume that

$$|\hat{C}_i - C_i| \leq \frac{\varepsilon}{w} n \qquad \forall i \in [w]$$

while by definition, we have

$$\hat{w}(MST) = n - w + \sum \hat{C}_i$$

$$w(MST) = n - w + \sum C_i$$

We can thus compare the difference between the true weight and our estimation by

$$\begin{aligned} |\hat{w}(MST) - w(MST)| &= |\sum_{i=1}^{w-1} \hat{C}_i - \sum_{i=1}^{w-1} C_i| \\ &\leq \sum_{i=1}^{w-1} |\hat{C}_i - C_i| \\ &\leq w \cdot \frac{\varepsilon}{w} n = \epsilon n \end{aligned}$$

as desired, where the first inequality is by the triangle inequality.

For query complexity, each call to the blackbox algorithm makes $O(\frac{1}{\varepsilon'^2} \log \frac{1}{\delta'}) = O(\frac{w^2}{\varepsilon^2} \log 3w)$ queries. With w total calls, this MST estimation algorithm makes $O(w\frac{w^2}{\varepsilon^2} \log 3w)$ queries. Notice that the query complexity depends entirely on w and ε and not n. With w small enough, say $w \leq n^{1/4}$, the algorithm is sublinear.

2 Estimating Average Degree of Graphs

Another natural question we can ask on graphs is their average degrees. Feige proposed the question and provided a $(2+\varepsilon)$ approximation, and Goldreich and Ron gave a $(1+\varepsilon)$ approximation on a different query model.

There are two major query models for estimating the average degree of a graph. The first one is degree query, which for any vertex v inputted, outputs the degree of v. Another interesting model is random neighbor: When given a vertex v, returns a random neighbor of v, and when given a pair of vertex and label (v, i), returns the i-th labeled neighbor of v.

We would like to devise and analyze a multiplicative approximation algorithm for estimating average degrees, with an upper bound and a lower bound on the approximation.

For the sake of convenience, assume that the graph is represented by a vertex-degree map and an adjacency list.

2.1 A Basic Idea

One of the most simplistic attempt one can make to estimate the average degree is to randomly sample nodes $v_1, \cdots, v_s$, and output their average degree $\frac{\sum deg(v_i)}{s}$.

While this idea is simple and convenient, it might not work properly on certain graphs. Consider for example the star graph of n vertices, S_n, in which every vertex other than vertex 1 is connected to vertex 1 only. The degree map for S_n consists of one entry of $n-1$ and $n-1$ entries of 1.

Recall the following version of the Chernoff Bound: Let $x_1, \cdots, x_t$ be i.i.d. random variables in $[0, 1]$. Let $X = \sum_{i=0}^{t} x_i$ be their sum, and let $p = E[x_i] = \frac{E[x]}{t}$ be their expected value. Then

$$Pr[|\frac{X}{t} - p| \geq \delta p] \leq e^{-\Omega(tp\delta^2)}$$

or equivalently

$$Pr[(1-\delta)p \leq \frac{X}{t} \leq (1+\delta)p] \geq 1 - e^{-\Omega(tp\delta^2)}$$

Thus for S_n, define $x_i = \frac{\deg v_i}{n}$, where v_i is the i-th sampled vertex. Then the x_i's satisfy the prerequisites of Chernoff, with expected value $p = E[x_i] = \frac{1}{n} \cdot \frac{2n-2}{n} \approx \frac{2}{n}$, where $2n-2$ is due to the total degree of S_n.

To get a satisfactory bound, we would like the approximation range factor δ to be at most constant, and the error probability $e^{-\Omega(tp\delta^2)}$ to be at most constant as well. Thus we have that $tp\delta^2 = \Omega(1)$, and with the bounds on p and δ we have that $t = \Omega(n)$. This implies that we would need at least linearly many queries on the degree map of the star graph to obtain a satisfactory result.

But notice that this simple idea is not without merits. Observe that if the degrees of vertices are nicely bounded such that $\alpha \leq deg(v_i) \leq (1+\varepsilon)\alpha$, we can instead define $x_i = \frac{deg(v_i)}{(1+\varepsilon)\alpha}$. Then, $p = E[x_i] \geq \frac{\alpha}{(1+\varepsilon)\alpha} = \frac{1}{1+\epsilon}$, which will be a constant instead. In such cases, to satisfy $tp\delta^2 = \Omega(1)$ with p and δ both constant, t can possibly be a constant as well! This implies that if the degrees are nicely bounded, one can make constant queries to obtain a good estimate of the average degree using Chernoff, as opposed to linearly many queries.

2.2 Bucketing Vertices with Degrees

An extension to the basic idea above in conjunction to our observation is to group vertices with similar degree magnitudes in the same bucket. It is then feasible to take Chernoff bound inside each bucket, and take Union bound over all buckets.

To formalize, let $\beta = \frac{\varepsilon}{c}$ for some $c \geq 0$, and let there be $t = O(\log_{1+\beta} n/\varepsilon)$ buckets in total, where bucket i, is defined as

$$B_i = \{v | (1+\beta)^{i-1} \leq \deg(v) \leq (1+\beta)^i\}$$

With this, the bucket B_1 would contain all vertices with $1 \leq \deg(v) \leq 1+\beta$, and B_2 would contain all vertices with $(1+\beta) \leq \deg(v) \leq (1+\beta)^2$, and so on, and finally B_t would contain all vertices with $(1+\beta)^{t-1} \leq \deg(v) \leq n$.

These buckets forms a nice partition of the vertices, thus we can bound and calculate many nice properties of the vertices based on the buckets, such as the total degree of vertices in bucket B_i:

$$(1+\beta)^{i-1}|B_i| \leq d_{B_i} \leq (1+\beta)^i |B_i|$$

or the average degree of the graph:

$$\sum_{i=1}^{t} (1+\beta)^{i-1} \frac{|B_i|}{n} \leq \bar{d} \leq \sum_{i=1}^{t} (1+\beta)^i \frac{|B_i|}{n}$$

Can this serve as a good approximation for the average degree of the graph? If we can obtain good estimates for all the $\frac{|B_i|}{n}$ values, then the sum would estimate well. But problem arises if some buckets B_i are small - they are highly likely to never be hit and sampled from.

A solution to this is to sample a set S of vertices of fixed size, and consider $S_i = S \cap B_i$ the intersection of said set with our bucket. As long as we are sampling uniformly, we have approximately

$$\frac{|B_i|}{n} \approx \frac{|S_i|}{|S|}$$

To formalize, we have the following claim:

Claim 5

$$E[\frac{|S_i|}{|S|}] = \frac{|B_i|}{n}$$

Proof Let x_j be the indicator variable of the j-th sampled vertex v_j belonging to B_i. Then

$$E[x_j] = \frac{|B_i|}{n}$$

and

$$E[\frac{|S_i|}{|S|}] = \frac{1}{|S|}\sum_j E[x_j] = \frac{1}{|S|}|S|\frac{|B_i|}{n} = \frac{|B_i|}{n}$$

■

We can thus use S to sample from the buckets, and use $\rho_i = \frac{|S_i|}{|S|}$ instead of $\frac{|B_i|}{n}$ to estimate the average degree of the graph. Our estimates would then be $\sum_i \rho_i(1+\beta)^{i-1}$, and we would like to claim that:

Claim 6

$$\bar{d} \leq \sum_i \rho_i(1+\beta)^{i-1} \leq \bar{d}(1+\beta)$$

How does this help with the issue with small buckets? With this sampling method, we will show that we can complete ignore the small B_i's while still achieving reasonable approximation factors. By showing that there are at most $O(\sqrt{\varepsilon n})$ vertices (with reasonably small constant factors) in small buckets, they can participate in at most $O(\varepsilon n)$ edges, resulting in an εn-additive approximation and thus a $(1+\varepsilon)$-multiplicative approximation.

2.3 Bucket Approximation Algorithm

Based on our observations and analyses, we propose the following approximation algorithm for average degrees of graphs.

Given graph G, first sample set S of $\Theta(\sqrt{n}\text{poly}\log n\text{poly}\frac{1}{\varepsilon})$ vertices. For each of the t buckets B_i, let $S_i = S \cap B_i$. We estimate $\frac{|B_i|}{n}$ for each B_i as follows:

- If $|S_i| \geq \sqrt{\frac{\varepsilon}{n}}\frac{|S|}{ct}$, let $\rho_i = \frac{|S_i|}{|S|}$.

- Otherwise, let $\rho_i = 0$.

Finally we output $\sum_i \rho_i (1+\beta)^{i-1}$ as our estimate, as usual.

We call S_i 'big' if

$$|S_i| \geq \sqrt{\frac{\varepsilon}{n}} \frac{|S|}{ct} = \sqrt{\frac{\varepsilon}{n}} \frac{\sqrt{n}\text{poly}\log n \text{poly}\frac{1}{\varepsilon}}{ct} = \text{poly}(\log n, \frac{1}{\varepsilon})$$

The reason behind this selection is that we would need approximately $\text{poly}(\log n, \frac{1}{\varepsilon})$ samples to obtain a reasonable estimate of the size of a bucket.

To analyze this algorithm, we first make a crude estimate of the average degree. For all i, with high probability we have that

$$(1-\gamma)\frac{|B_i|}{n} \leq \rho_i \leq (1+\gamma)\frac{|B_i|}{n}$$

By applying Chernoff bound within each bucket, and applying Union bound over all buckets. Then we have that

$$\sum_i \rho_i (1+\beta)^{i-1} \leq \sum_i \frac{|B_i|}{n}(1+\beta)^i$$

However, in the above analysis, only vertices in big buckets are accounted for, which leads to two different sources of error. Consider the following three types of edges:

- Edges between two vertices of big buckets, and
- Edges between one vertex of a big bucket and one vertex of a small bucket, and
- Edges between two vertices of small buckets.

The first type of edges is well-accounted for, as their contribution to both endpoint's degrees are well-estimated by ρ_i. However, the third type of edges between vertices in small buckets are completely ignored and unaccounted for. Moreover, edges of the second type contribute 2 to the total degree, but is only counted once, due to the degree of the endpoint in a small bucket being ignored.

We justify the error induced by edges between small vertices first. Observe that if a bucket B_i has size $|B_i| > \frac{2\sqrt{\varepsilon n}}{ct}$, we have that $E[S_i] = \frac{|B_i|}{n} \geq 2\sqrt{\frac{\varepsilon}{n}}\frac{1}{ct}$. Notice that this is twice the threshold for S_i to be identified as a big bucket, so by Chernoff bound, after sampling S, with high probability, the algorithm will consider S_i big.

Suppose otherwise that $|B_i| < \frac{2\sqrt{\varepsilon n}}{ct}$. In the worst case, all such buckets are recognized as small buckets. The sum of size of all such buckets is

$$\bigcup_{B_i \text{ small}} |B_i| < t \cdot \frac{2\sqrt{\varepsilon n}}{ct} = \frac{2\sqrt{\varepsilon n}}{c} = O(\sqrt{\varepsilon n})$$

As our discussion above implies, there can thus be at most εn edges between vertices in small buckets, and thus they will induce at most a $(1+\varepsilon)$ approximation factor to the algorithm.

The error induced by edges between small and big vertices is more problematic. These edges are only accounted for once instead of twice, and thus introduces a factor of 2 approximation.

With these two error inducing factors, we arrive at a $(2+\varepsilon)$-approximation algorithm for average degrees of a graph with $O(\sqrt{n}\text{poly}\log n \text{poly}\frac{1}{\varepsilon})$ degree queries. The bound on the amount of queries is in fact tight: There exists graphs that requires $\Omega(\sqrt{n})$ degree queries to estimate.

3 Better Approximation with Random Neighbor Queries

A question that immediate follows the results above is: Can we achieve a $(1+\varepsilon)$-multiplicative approximation? The answer is to use random neighbor queries introduced at the beginning. Recall that a random neighbor query when inputted a vertex v, chooses a random neighbor of v to output. Another type of random neighbor query takes in a pair of vertex and index (v, i), and outputs the i-th neighbor of v according to an indexing.

This optimization in approximation factors comes from the edges between big bucket vertices and small bucket vertices. Previously these edges are only accounted for once instead of twice, introducing an approximation factor of 2. To improve to a $(1+\varepsilon)$-approximation, it suffices to estimate the amount of such edges properly.

We propose the following algorithm which estimates the amount of such edges adjacent to a specific big bucket B_i.

Let B_i be the fixed big bucket, and let T_i be the amount of edges from B_i to another small bucket B_j. We repeat the following procedure $s = O(\frac{1}{\delta})$ times: pick a random vertex $v \in B_i$, and make a random neighbor query on v to obtain u.

Let a_j be the indicator random variable of the event that the edge (v, u) is a big-small edge. We estimate the expected value by taking the average across all s instances, with $\hat{a} = \frac{\sum a_j}{s}$.

Notice that by definition of expected values, we can express

$$E[a_j] = \frac{\text{number of edges from } B_i \text{ to small buckets}}{\text{number of edges from } B_i \text{ to anywhere}}$$

The numerator is by definition identical to T_i. The denominator is equal to the total degree of B_i, and can be bounded by

$$(1+\beta)^{i-1}|B_i| \leq d_{B_i} \leq (1+\beta)^i|B_i|$$

Thus we estimate T_i by

$$\hat{a}(1+\beta)^{i-1}|B_i| \leq T_i \leq \hat{a}(1+\beta)^i|B_i|$$

We also know by Chernoff that with high probability, for all i,

$$(1-\gamma)|B_i| \leq \frac{|S_i|}{|S|}n \leq (1+\gamma)|B_i|$$

and with high probability,

$$(1-\gamma)E[a_j] \leq \hat{a} \leq (1+\gamma)E[a_j]$$

we can output $T_i \approx \hat{a}(1+\beta)^{i-1}\frac{|S_i|}{|S|}n$.

With this estimation, we can account for the errors that was previously induced by these big-small edges, thus we acquired a $(1+\varepsilon)$-approximation for average degrees of graphs, with the same query complexity, with degree queries and random neighbor queries.