Lec 2  Sublinear Time Algorithms

Today:

Sublinear time algorithms for:

1. Estimating MST weight
2. Estimating Average degree

Recall from last time:

- Can estimate $\pm$ connected components of degree $\leq d$ graph to within $\pm \epsilon n$
  in time $O(d/\epsilon^4)$

- $\omega \Rightarrow$ thus can give estimate to within $\pm \epsilon n$ in time $O(d/poly(\epsilon) \cdot \log \beta)$

  falls in this range with prob $\geq 3/4$ "success probability"

  falls in this range with prob $\geq 1 - \beta$ "failure probability"
Approximating Min Spanning Tree (MST)

Input
1. \( G = (V,E) \)
2. Adj list representation \( n = |V| \)
3. Max degree \( d \)
4. Each edge has weight \( w_{uv} \in \mathbb{R} \) \( u,v \in V \) \( \delta \leq w_{uv} \leq \omega \)

Output
1. \( \hat{M} = \min \{ w(T) \} \)
2. \( T \) spans \( G \)
3. \( \hat{M} \) is a tree
4. \( \hat{M} \) touches every node

Output
1. \( \hat{M} \) such that \( (1-\varepsilon)M \leq \hat{M} \leq (1+\varepsilon)M \)

Assumption on \( w_{uv} \) \( \Rightarrow \) \( n-1 \leq w(T) \leq w(n-1) \)

A different characterization of \( \text{MST} \):

\[
\text{def } \ G^{(i)} = (V_i, E^{(i)}) \quad \text{where} \quad E^{(i)} = \{ (u,v) \mid w_{uv} \in \mathbb{R}, u,v \in V_i \}
\]

\[
C^{(i)} = \# \text{ conn comp of } G^{(i)}
\]
Some examples before characterization:

1) $w=1$
   - Only size 1 weights, connected by assumption here $M = n - 1$

2) $w=2$
   - Weights $\in \{1, 2\}$

[Diagram of two graphs with weights labeled]

Idea of Kruskal:
- Use as many $w=1$ edges as you can
- Only need $w=2$ edges to connect the components

$\Rightarrow$ Need $\binom{C^{(i)}}{1}$ $w=2$ edges in MST

(Recall that total # of edges is $n-1$)

Total $w$-t of MST:

$$M = (n-1) + (\binom{C^{(i)}}{1} - 1) = n - 2 + \binom{C^{(i)}}{i}$$

$\uparrow$ for each edge
$\uparrow$ additional 1 for $w=2$ edges
Claim \[ M = n - w + \sum_{i=1}^{w-1} C(i) \]

**Pf.**

Let \( \alpha_i \) be edges of \( w_i \) in any MST of \( G \). According to Kruskal's algorithm, all MST's have the same value of \( \alpha_i \).

\[ \sum_{i=1}^{w} \alpha_i = \# \text{ conn comp of } G^{(w)} - 1 \]

\[ = C^{(w)} - 1 \]

where \( C^{(0)} = n \) (no edges in \( G^{(0)} \)).

\[ M = \sum_{i=1}^{w} i \cdot \alpha_i \]

\[ = \sum_{i=1}^{w} \alpha_i + \sum_{i=2}^{w} \alpha_i + \sum_{i=3}^{w} \alpha_i + \ldots + \sum_{i=w}^{w} \alpha_i \]

\[ = \alpha_{w} \]

\[ = (n-1) + (C^{(1)} - 1) + (C^{(2)} - 1) + \ldots + (C^{(w-1)} - 1) \]

\[ = n - w + \sum_{i=1}^{w-1} C(i) \]
Approximation Algorithm:

For $i = 1$ to $w - 1$

$\hat{C}(i) = \text{approx} \# \text{cc} \text{ of } G(i) \text{ to within } \frac{\epsilon}{2w}, n$ (additive error)

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

Runtime:

$\tilde{O}\left( \frac{d}{(\epsilon')^2} \right) = \tilde{O}\left( \frac{dw^5}{\epsilon^4} \right)$ for each call to approx $\# \text{cc}$.

Total $\tilde{O}\left( \frac{dw^5}{\epsilon^4} \right)$

(Can improve to $O(\frac{dw}{\epsilon^2} \log \frac{dw}{\epsilon})$)

\[ \text{need } O(\frac{dw}{\epsilon^2}) \]

Approximation guarantee: "failure" if approx error of approx $\# \text{cc}$ is too big ($\geq \epsilon'$) How?

Call approx $\# \text{cc}$ with "failure" probability $\leq \frac{1}{4w}$

$\Pr \left[ \sum \text{all calls to approx } \# \text{cc give output that is } \epsilon' \text{ additive approx} \right] \geq 1 - \frac{w}{4w}$ union bound

If $\tilde{M}$ happens: $|M - \tilde{M}| \leq \frac{\epsilon n}{2w} = \frac{\epsilon n}{2} \leftarrow \text{small additive error}$

$\Rightarrow$ since $M \geq n - 1 \geq n/2 \Rightarrow |M - \tilde{M}| \leq \epsilon M \leftarrow \text{small multiplicative error}$

Conclusion: runtime depends only on $d, w, \epsilon$ gives additive/multiplicative error.
Approximating Average Degree

def average degree \( \bar{d} = \frac{\sum_{u \in V} d(u)}{n} \)

Assume: G simple (no parallel edges, self-loops) 
\( \Omega(n) \) edges (not "ultra-sparse")

representation: adjacency list + degrees
\[ d(v): \text{node } v \]

\[
\begin{array}{cccccccc}
3 & 1 & 2 & 5 & 7 \\
1 & 2 & 3 & 4 & 5 \\
& 1 & 2 & 3 & 4 \\
& & 1 & 2 & 3 \\
\end{array}
\]

- degree queries: on \( v \) return \( d(v) \)
- neighbor queries: for \( (v, j) \) return \( j^{th} \) nbr of \( v \)

Random sampling:
Pick \( ?? \) sample nodes \( v_1 \ldots v_s \)

output \( \frac{1}{s} \sum_{i=1}^{s} d(v_i) \) (ave degree of sample)

using straightforward Chernoff/Hoeffding \( \Rightarrow \mathcal{O}(\log n) \) samples needed
Degree sequences are special?

\[(n-1, 0, 0, 0, \ldots, 0) \quad \text{not possible} \]
\[(n-1, 1, 1, \ldots, 1) \quad \text{is possible} \]

Some lower bounds:

"Ultra-sparse case":

- Need linear time to get any multiplicative approx
- Graph with 0 edges vs. graph with 1 edge
  - Ave deg = 0
  - Ave deg = \( \frac{1}{n} \)
  - Need \( O(n) \) queries to distinguish

Ave deg \( \geq 2 \):

- \( n \)-cycle \( \bar{d} = 2 \)
- \( n - cn^{1/2} \) cycle \( \bar{d} \approx 2 + c^2 \)
- \( cn^{1/2} \)-clique

Need \( \Omega(n^{3/2}) \) queries to find clique node
Algorithm Idea:

- Group nodes of similar degrees
- Estimate average within each group

- Each group has bounded variance
- Doesn't work for estimating average of arbitrary numbers, why should it work here?

Bucketing:

Set parameters:

\[ B_i \]  

\[ \beta = \frac{\epsilon}{c} \]

\[ t = O(\log n / \epsilon) \]  

Number of buckets

\[ B_0 = \{ v \mid (1 + \beta)^{i-1} \leq d(v) \leq (1 + \beta)^i \} \]

for \( i \in \{0, \ldots, t-1\} \)

Note:

- Total degree of nodes in \( B_i \)

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

- Total degree of graph

\[ \sum_{i=0}^{t-1} (1 + \beta)^i |B_i| \leq d_{\text{total}} \leq \sum_{i=0}^{t-1} (1 + \beta)^i |B_i| \]
First idea for algorithm:

- Take sample $S$ of nodes
  
  $$S_i \leftarrow S \cap B_i$$  \hspace{1cm} (samples that fall in $i$th bucket use degree queries to determine this)

- Estimate average degree contribution from $B_i$
  
  using $S_i$
  
  \[ \rho_i \leftarrow \frac{|S_i|}{|S|} \hspace{1cm} \text{note:} \hspace{0.5cm} \forall i \hspace{1cm} E[\rho_i] = E\left[ \frac{|S_i|}{|S|} \right] \]
  
  \[= \frac{|B_i|}{n} \]

- Output $\sum_i \rho_i (1+\beta)^{i-1}$ \hspace{1cm} \text{\color{red}{too undercounting}}

Problem:

- Let $|S_i|$ is small \hspace{1cm} (for these, our estimate of $|S_i|$ could be terrible)
  
  likely come from $i$ such $|B_i|$ small

Example of problem:

- $3$ nodes, deg $n-3$
- $n-3$ nodes, deg $3$

$$a \leftarrow i \text{ st. } (1+\beta)^{i-1} \leq 3 \leq (1+\beta)^i$$
$$\text{ contributes } (n-3) \cdot 3 \text{ edges}$$

$$b \leftarrow i \text{ st. } (1+\beta)^{i-1} \leq n-3 \leq (1+\beta)^i$$
$$\text{ contributes } 3 \cdot (n-3) \text{ edges}$$

\[|B_a| = n-3 \hspace{1cm} |B_b| = 3 \hspace{1cm} \text{Never sampled but contributes }\frac{n}{2} \text{ edges!} \]

Still, maybe good enough for 2-approximation?
Next idea: use $0$ for small buckets

Algorithm:

- Sample $S$
- $S_\lambda \leftarrow S \cap B_\lambda$
- For all $i$
  - if $|S_\lambda| \geq \frac{T_\lambda}{n} \cdot \frac{|S|}{c \cdot t}$
    - use $\rho_i \leftarrow \frac{|S_\lambda|}{|S|}$
    - Call $i$ "big"
  - else $\rho_i \leftarrow 0$
    - Call $i$ "small"
- Output $\sum_i \rho_i (1 + \beta)^{i-1}$

Analysis:

1) Output not too large

idealistic (but unrealistic) $\implies$ Suppose $\forall i \rho_i = \frac{|B_\lambda|}{n}$, then $\sum_i \rho_i (1 + \beta)^{i-1} = \sum_i \frac{|B_\lambda|}{n} (1 + \beta)^{i-1}$

realistic case Suppose $\forall i \rho_i \leq \frac{|B_\lambda|}{n} (1 + \beta)$

$\implies \sum_i \rho_i (1 + \beta)^{i-1} \leq d (1 + \beta)$
2) Can output be too small?

\[
\text{if } \forall i \quad p_i = \frac{|B_i|}{n} \text{ then } \sum_i p_i (1+\beta)^{d-i} = \sum_i \frac{|B_i|}{n} (1+\beta)^{d-i} \\
\geq (1-\beta) \sum_i \frac{|B_i|}{n} (1+\beta)^i \\
\geq (1-\beta)^d \cdot \deg \text{ of node in } B_i
\]

By sampling, for big \( i \), \( p_i \geq \frac{|B_i|}{n} (1-\beta) \)

For small \( i \) ????

How much undercounting?

divide edges into 3 types:

1) big-big - both endpts in big buckets counted twice
2) big-small - one endpt in big bucket counted once
3) small-small - both endpts in small buckets never counted

[See example]

Note: big-big & big-small get counted (off by factor of two)
but small-small can be a real problem
Example

ave deg 5
bucket a
big

ave deg 7
bucket b
small

ave deg n-5
bucket c
small

Total degree
LHS of bipartite RHS of bipartite 5-clique
\(5(n-\delta) + (n-\delta)(3) + 4.5 = 8(n-\delta) + 20\)

ave deg \(\approx 8n\) Algorithm will output \(\approx 5n\)

Samples

\[6, 109, 13, 127, 17, 14, \ldots\]

\(\emptyset\)

\(\emptyset\)

bucket a
bucket b
bucket c

\(\Rightarrow (\text{whp}) \text{ bucket a is big, in fact,}\)

\(\text{whp } p_a \approx 1\)

\(\Rightarrow (\text{whp}) b \neq c \text{ are small}\)

\(p_b \approx 0 \quad p_c \approx 0\)

Output \(\times 5\)

# big-small edges: \(3(n-\delta)\)

Fraction: \(\frac{3(n-\delta)}{5(n-\delta)} = \frac{3}{5}\)

\(E[a_j] = \frac{3}{5}\) Output \(1 \cdot \left(1 + \frac{3}{5}\right) \approx 8\) \(\approx 5\)