CPSC 768: Scalable and Private Graph Algorithms

Lecture 4: Approximate Average Degree in the Sublinear Model

Quanquan C. Liu
quanquan.liu@yale.edu

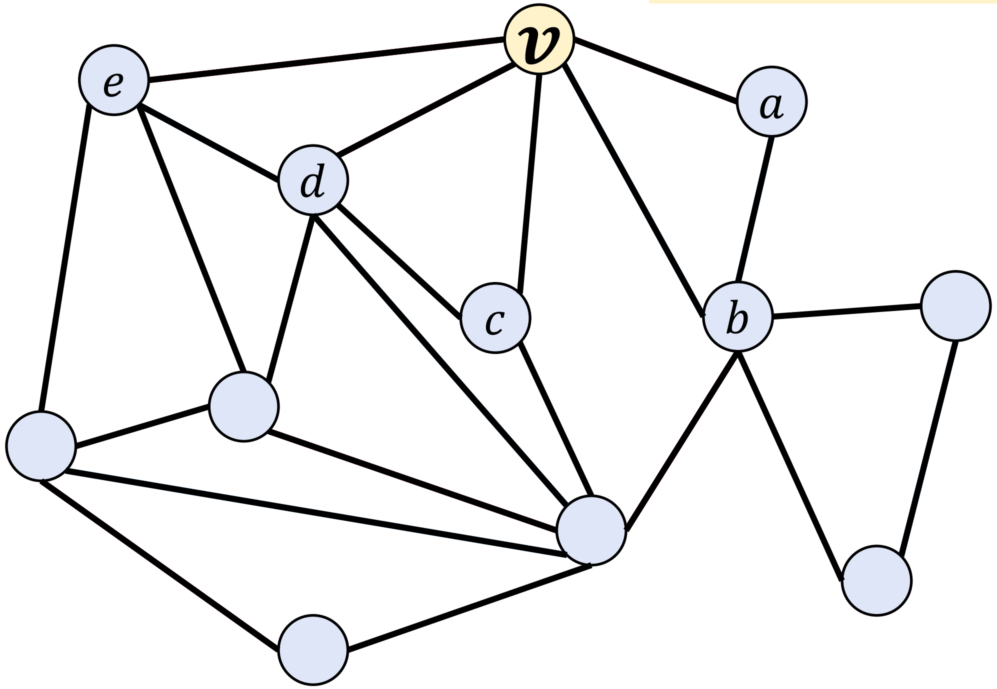
Open Problem Session Results

- Difficult to schedule a time when everyone is free!
- Tuesdays/Thursdays were unpopular
- **Proposed times**:
  - **Monday 3pm**
  - **Friday 3:30pm**
- Thoughts? Preferences?

Sublinear Graph Model: Query Models

- **Adjacency list query model:** $O(1)$ time per query
  - **Degree queries**: given a vertex $v \in V$, **output $\mathbf{deg}(v)$**
  - **Neighbor queries**: given a vertex vertex $v \in V$ and $i \in [n]$, **output the i-th neighbor of v** or $\perp$ if $i > \deg(v)$

Third neighbor of v is c



Approximate Average Degree

- Given a graph in the adjacency list query model, compute the approximate average degree $\tilde{\boldsymbol{d}}$ of the nodes in the graph
  - $\overline{\boldsymbol{d}}$ denotes the average degree
  - Correct with **probability at least $\mathbf{1-\delta}$**
  - Constant, **c-approximation**
    - $\boldsymbol{c = 1 + \varepsilon}$
    - $\boldsymbol{c = 2 + \varepsilon}$

Lower Bounds

- When $c < 1$, require linear queries
  - An empty graph
  - Graph with 1 edge
- **Hence we consider $\bar{d} \geq 1$**

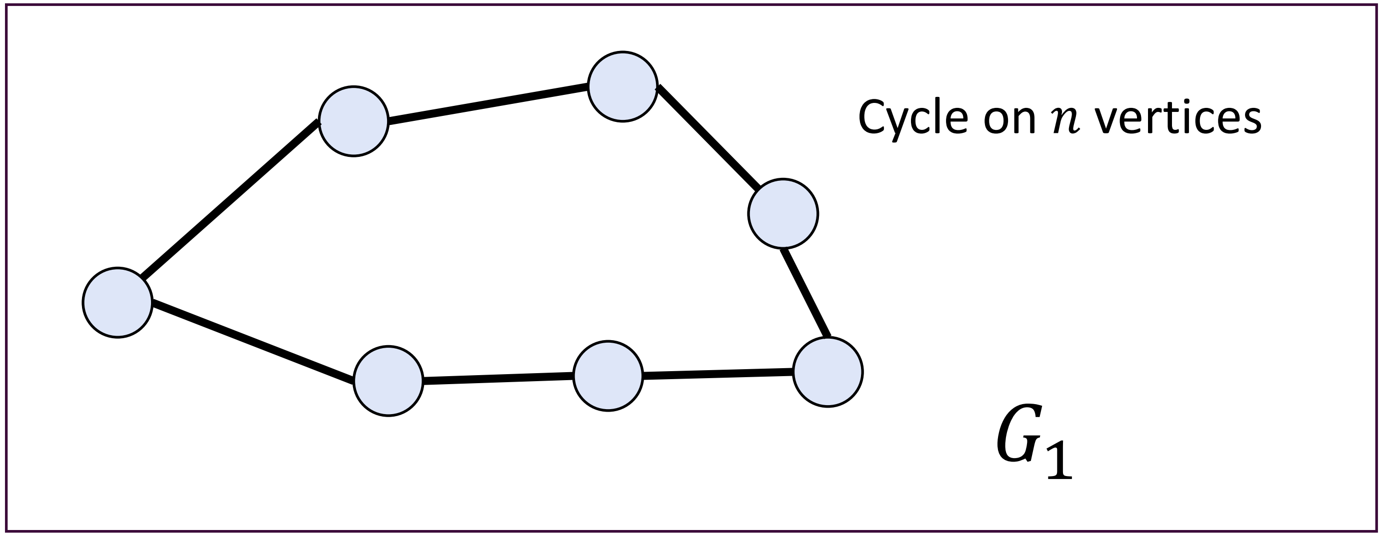
Lower Bounds

- When $c < 1$, require linear queries
  - An empty graph
  - Graph with 1 edge
- **Hence we consider $\bar{d} \geq 1$**

Sampling and taking the average degree **doesn't work in general**

Lower Bounds

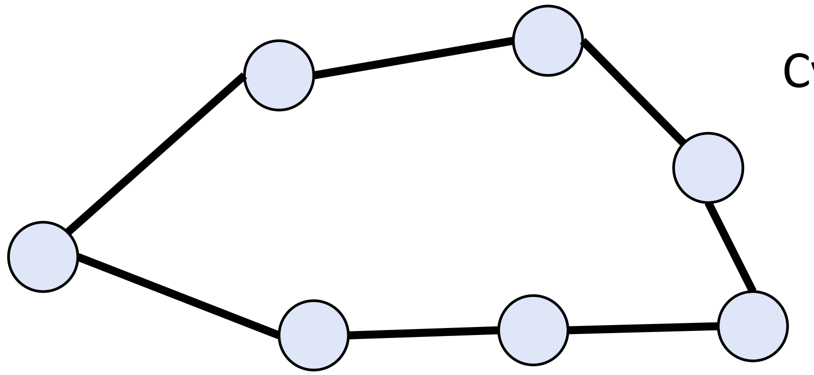
- When $c < 1$, require linear queries
  - An empty graph
  - Graph with 1 edge
- **Hence we consider $\bar{d} \geq 1$**



Sampling and taking the average degree **doesn't work in general**

Lower Bounds

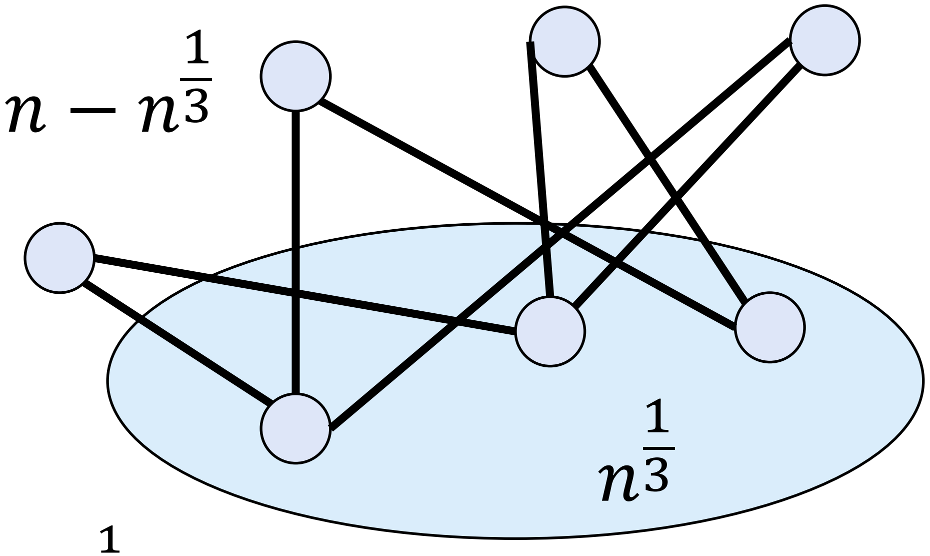
- When $c < 1$, require linear queries
  - An empty graph
  - Graph with 1 edge
- **Hence we consider $\overline{d} \geq 1$**



Cycle on n vertices

G_1

$n - n^{\frac{1}{3}}$



$n^{\frac{1}{3}}$

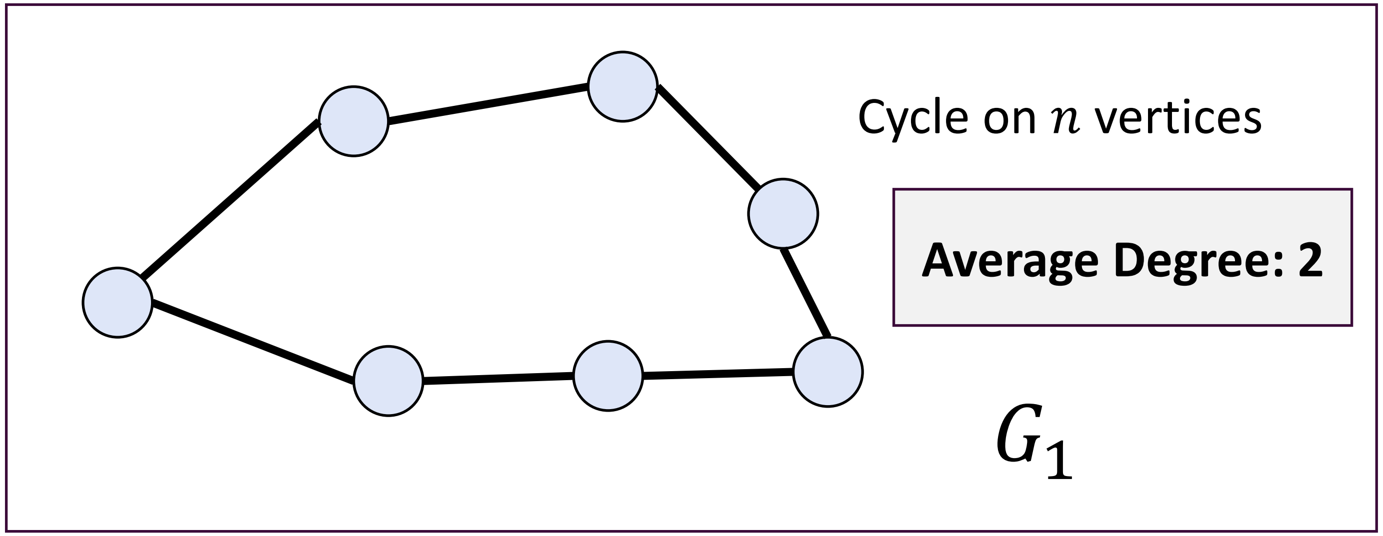
$n^{\frac{1}{3}}$ stars each with degree $2n^{\frac{2}{3}} - 2 + n^{\frac{1}{3}}$

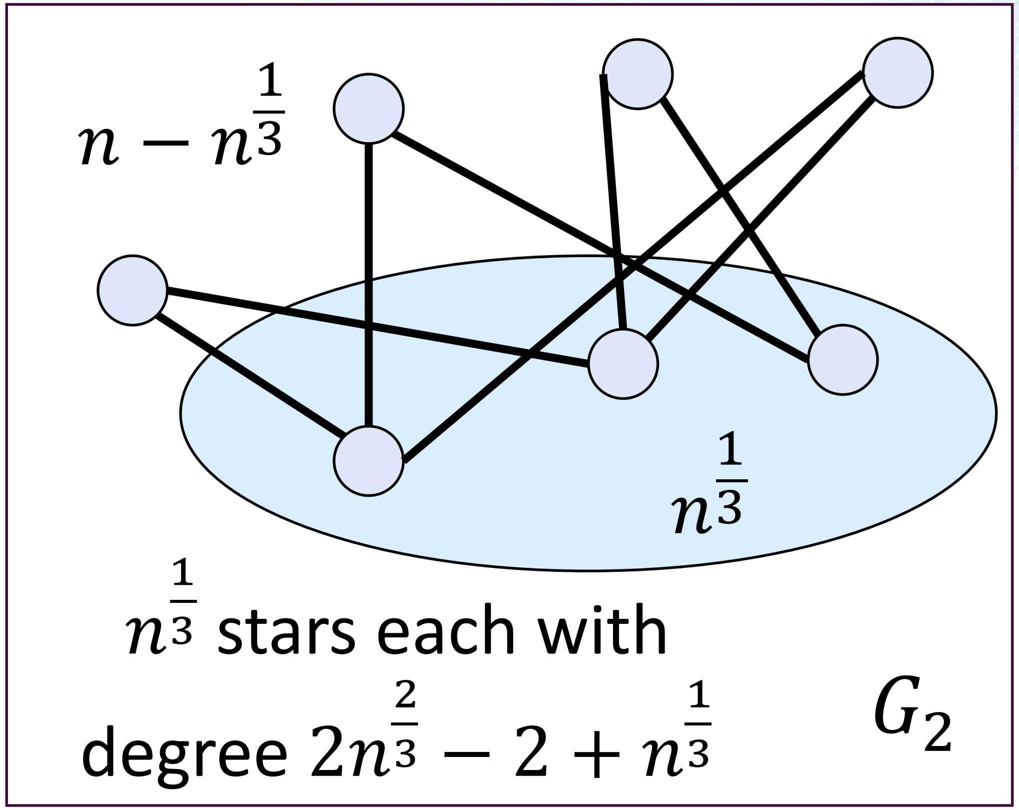
G_2

Sampling and taking the average degree **doesn't work in general**

Lower Bounds

- When $c < 1$, require linear queries
  - An empty graph
  - Graph with 1 edge
- **Hence we consider $\bar{d} \geq 1$**



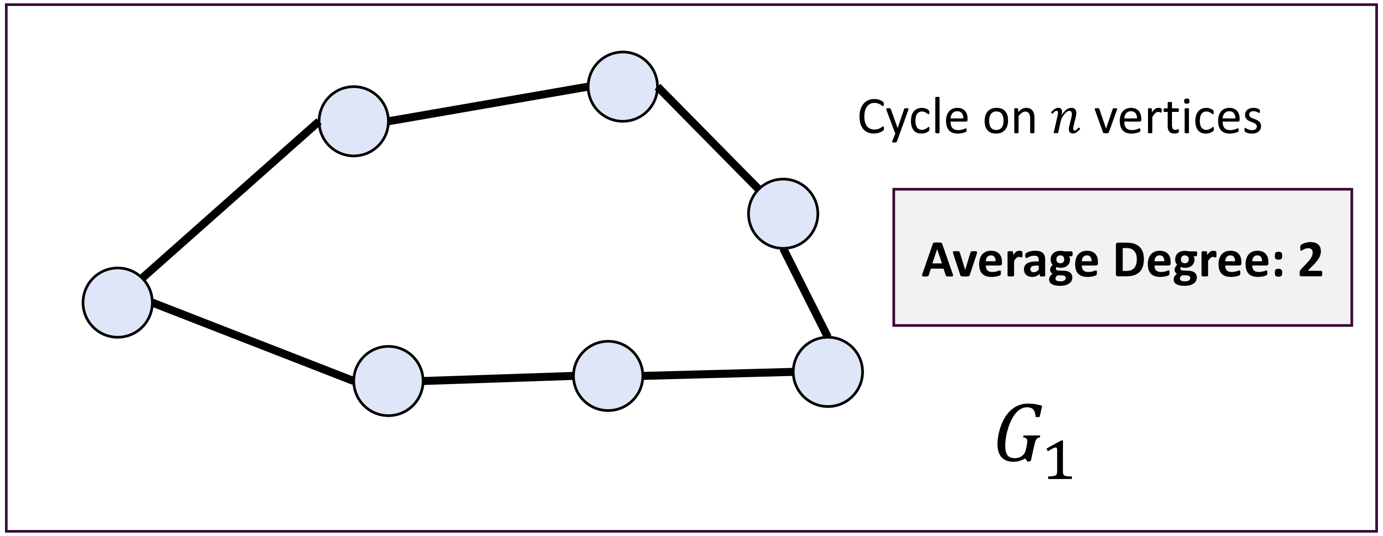


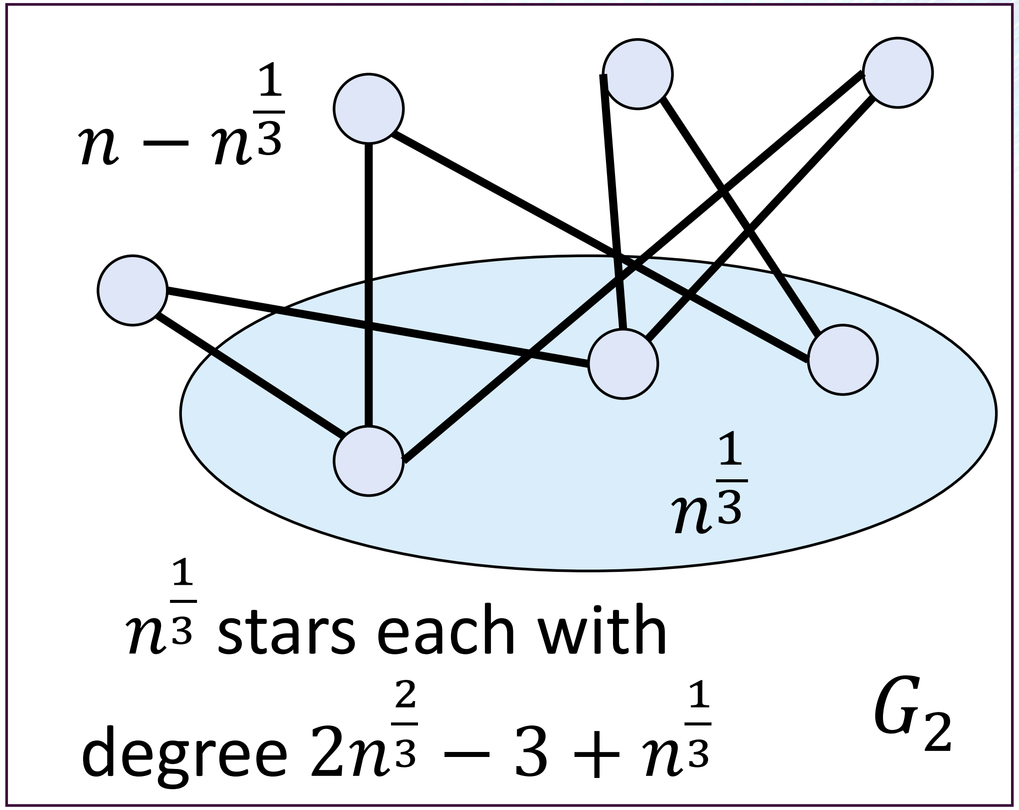
Sampling and taking the average degree **doesn't work in general**

Average Degree: $\left(n^{\frac{1}{3}} \cdot \left(2n^{\frac{2}{3}} - 3 + n^{\frac{1}{3}}\right) + 2\left(n - n^{\frac{1}{3}}\right)\right)/n \approx (4 - \varepsilon)$

Lower Bounds

- When $c < 1$, require linear queries
  - An empty graph
  - Graph with 1 edge
- **Hence we consider $\bar{d} \geq 1$**



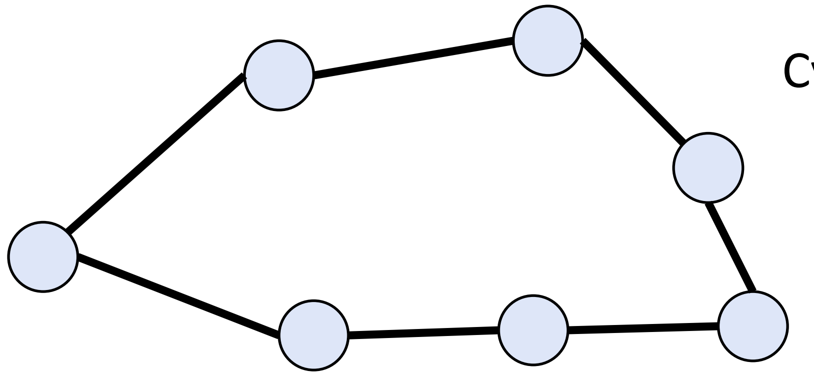


Sampling and taking the average degree **doesn't work in general**

Average Degree: $(n^{\frac{1}{3}} \cdot \left(2n^{\frac{2}{3}} - 3 + n^{\frac{1}{3}}\right) + 2(n - n^{\frac{1}{3}}))/n \approx (4 - \varepsilon)$

Lower Bounds

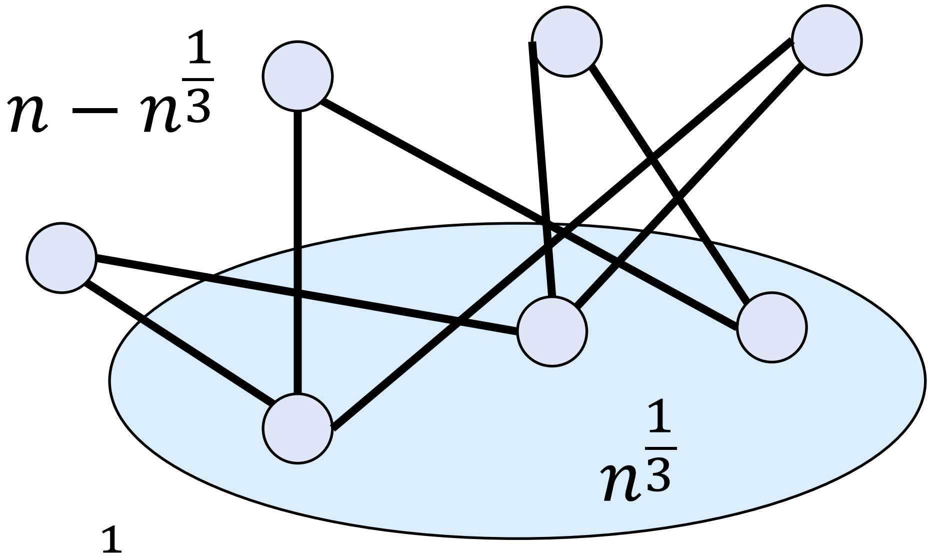
- When $c < 1$, require linear queries
  - An empty graph
  - Graph with 1 edge
- **Hence we consider $\overline{d} \geq 1$**



Cycle on n vertices

Average Degree: 2

G_1



$n^{\frac{1}{3}}$ stars each with degree $2n^{\frac{2}{3}} - 2 + n^{\frac{1}{3}}$

G_2

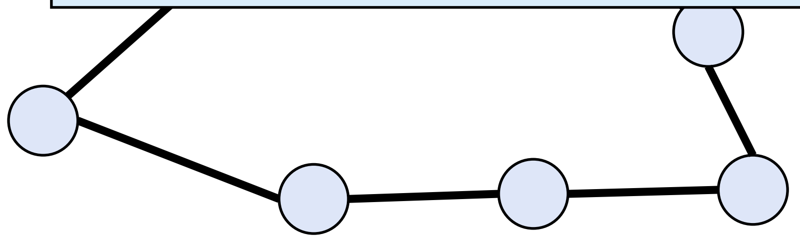
Requires $\Omega\left(n^{\frac{2}{3}}\right)$ samples

Average Degree: $\left(n^{\frac{1}{3}} \cdot \left(2n^{\frac{2}{3}} - 3 + n^{\frac{1}{3}}\right) + 2\left(n - n^{\frac{1}{3}}\right)\right)/n \approx (4 - \varepsilon)$

Lower Bounds

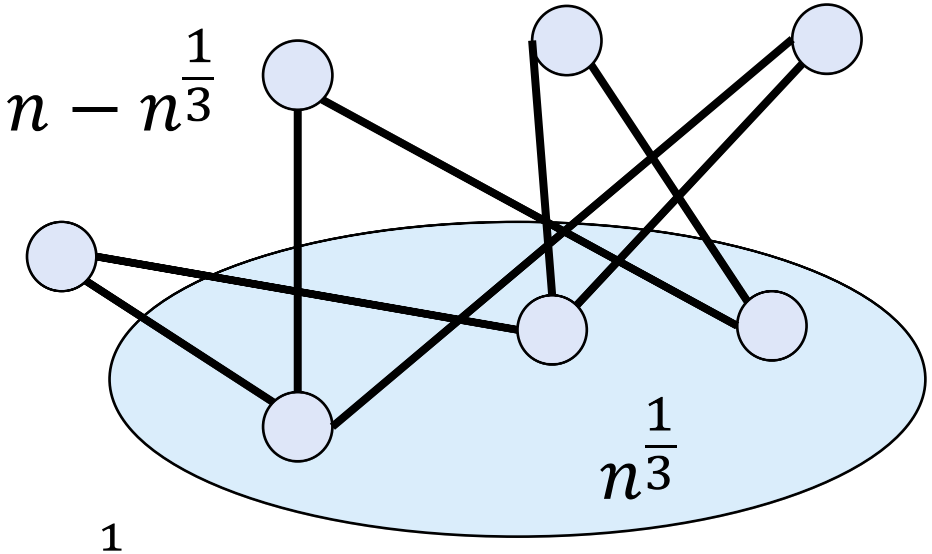
- When $c < 1$, require linear queries

Another problem: high variance, **small number of nodes** make ***large* degree contributions**



Average Degree: 2

G_1



$n^{\frac{1}{3}}$ stars each with degree $2n^{\frac{2}{3}} - 2 + n^{\frac{1}{3}}$

G_2

Requires $\Omega\left(n^{\frac{2}{3}}\right)$ samples

However, the strategy works for almost regular graphs!

- All vertices have degree in $[d, 10d]$ **for some known** d
- Expectation of any sample is equal to $\bar{d}$
  - $\sum_{i=1}^{n} \frac{1}{n} \cdot \deg(u_i) = \frac{1}{n} \cdot \sum_{i=1}^{n} \deg(u_i) = \bar{d}$
- **Sample** $k = \frac{50}{\varepsilon^2} \cdot \ln\left(\frac{1}{\delta}\right)$ **samples**

Additive Chernoff Bound: Let $Y_1, Y_2, \ldots, Y_k$ **be independent random variables with values in** $[0, 1]$ **and** $Y = \sum_{i=1} Y_i$ **. Then, for any** $b \geq 1$**,**

$$Pr[|Y - E[Y]| > b] \leq 2 \cdot \exp\left(-\frac{2b^2}{k}\right).$$

However, the strategy works for <u>almost</u> regular graphs!

- All vertices have degree in $[d, 10d]$ **for some known** d
- Expectation of any sample is equal to $\bar{d}$
  - $\sum_{i=1}^{n} \frac{1}{n} \cdot \deg(u_i) = \frac{1}{n} \cdot \sum_{i=1}^{n} \deg(u_i) = \bar{d}$
- **Sample $k = \frac{50}{\varepsilon^2} \cdot \ln\left(\frac{1}{\delta}\right)$ samples**

$Y_i = \deg(u_i)$ not in $[0, 1]$, what do we do?

<u>Additive Chernoff Bound</u>: Let $Y_1, Y_2, \ldots, Y_k$ **be independent random variables with values in $[0, 1]$ and $Y = \sum_{i=1} Y_i$. Then, for any $b \geq 1$,**

$$Pr[|Y - E[Y]| > b] \leq 2 \cdot \exp\left(-\frac{2b^2}{k}\right).$$

However, the strategy works for <u>almost</u> regular graphs!

- All vertices have degree in $[d, 10d]$ **for some known d**
- Expectation of any sample is equal to $\bar{d}$
  - $\sum_{i=1}^{n} \frac{1}{n} \cdot \deg(u_i) = \frac{1}{n} \cdot \sum_{i=1}^{n} \deg(u_i) = \bar{d}$
- **Sample $k = \frac{50}{\varepsilon^2} \cdot \ln\left(\frac{1}{\delta}\right)$ samples**

Full Analysis Left as an Exercise for the Reader

<u>Additive Chernoff Bound</u>: Let $Y_1, Y_2, \ldots, Y_k$ **be independent random variables with values in** $[0, 1]$ **and** $Y = \sum_{i=1} Y_i$ **. Then, for any** $b \geq 1$**,**

$$Pr[|Y - E[Y]| > b] \leq 2 \cdot \exp\left(-\frac{2b^2}{k}\right).$$

However, the strategy works for <u>almost</u> regular graphs!

- All vertices have degree in $[d, 10d]$ **for some known d**
- Expectation of any sample is equal to $\bar{d}$
  - $\sum_{i=1}^{n} \frac{1}{n} \cdot \deg(u_i) = \frac{1}{n} \cdot \sum_{i=1}^{n} \deg(u_i) = \bar{d}$
- **Sample $k = \frac{50}{\varepsilon^2} \cdot \ln\left(\frac{1}{\delta}\right)$ samples**

Normalization is a BIG issue in general! Need to normalize by 1/n!!

<u>Additive Chernoff Bound</u>: Let $Y_1, Y_2, \ldots, Y_k$ **be independent random variables with values in $[0, 1]$ and $Y = \sum_{i=1} Y_i$. Then, for any $b \geq 1$,**

$$\Pr[|Y - E[Y]| > b] \leq 2 \cdot \exp\left(-\frac{2b^2}{k}\right).$$

A $(2+\varepsilon)$-Approximate Algorithm

- Gets worse approximation but **bucketing is a very important concept used in many algorithms**

A $(2+\varepsilon)$-Approximate Algorithm

- Gets worse approximation but **bucketing is a very important concept used in many algorithms**
- Separate estimating nodes with **different degrees**
  - Let $\boldsymbol{\beta = \frac{\varepsilon}{c}}$ (constant c) and $t = O\left(\frac{\log n}{\varepsilon}\right)$, then i-th bucket is defined as

A $(2+\varepsilon)$-Approximate Algorithm

- Gets worse approximation but **bucketing is a very important concept used in many algorithms**
- Separate estimating nodes with **different degrees**
  - Let $\boldsymbol{\beta = \frac{\varepsilon}{c}}$ (constant c) and $t = O\left(\frac{\log n}{\varepsilon}\right)$, then i-th bucket is defined as

$$\boldsymbol{B_i = \{v \in V \mid (1+\beta)^{i-1} < \deg(v) \leq (1+\beta)^i\}}$$

for $i \in \{0, 1, \dots, t-1\}$

A $(2 + \varepsilon)$-Approximate Algorithm

- **Key point: intuitively we want to correctly estimate sizes of each bucket**
  - Knowing the correct sizes lets us get **good approximations** of the average degree

A $(2 + \varepsilon)$-Approximate Algorithm

- **Key point: intuitively we want to correctly estimate sizes of each bucket**
  - Knowing the correct sizes lets us get **good approximations** of the average degree
- **Problem**: some buckets are **small with large degrees!**

A $(2+\varepsilon)$-Approximate Algorithm

- **Key point: intuitively we want to correctly estimate sizes of each bucket**
  - Knowing the correct sizes lets us get **good approximations** of the average degree
- **Problem**: some buckets are **small with large degrees!**
- **Solution**: just **ignore the small buckets**

A $(2 + \varepsilon)$-Approximate Algorithm

- **Key point: intuitively we want to correctly estimate sizes of each bucket**
  - Knowing the correct sizes lets us get **good approximations** of the average degree
- **Problem**: some buckets are **small with large degrees!**
- **Solution**: just **ignore the small buckets**

Also the classification of small or large depends on our samples

A $(2+\varepsilon)$-Approximate Algorithm

You've seen $\log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^2}$ factors many times now!

- **Algorithm:**
  - Take $|S| = \Theta\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**

A $(2 + \varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|S| = \Theta\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \dots, t-1\}$:

Iterate through every bucket

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \dots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$

Figure out how many sampled elements are in each bucket

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \dots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$

If large number of samples, go ahead and **estimate size of the bucket**

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \dots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n}} \cdot \frac{|S|}{c \cdot t}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$

Otherwise, bucket is small and **ignore the bucket**

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|S| = \Theta\left(\sqrt{n} \cdot \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^4} \log^2 n\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n}} \cdot \frac{|S|}{c \cdot t}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

Return number of elements in the bucket times degree of bucket

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^4} \log^2 \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \dots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- First, let's compute $E[\rho_i]$

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^{2} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- First, let's compute $E[\rho_i]$
  - $E\left[\frac{|S_i|}{|S|}\right] = E\left[\sum_{j=1}^{|S|} \frac{\sigma_j^i}{|S|}\right]$
  - $\sigma_j^i = 1$ if sample j falls in bucket i and 0 otherwise

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|S| = \Theta\left(\sqrt{n} \cdot \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^4} \log^2 n\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- First, let's compute $E[\rho_i]$
  - $E\left[\frac{|S_i|}{|S|}\right] = E\left[\sum_{j=1}^{|S|} \frac{\sigma_j^i}{|S|}\right]$
  - $\sigma_j^i = 1$ if sample j falls in bucket i and 0 otherwise
  - Each sample has probability $\frac{|B_i|}{n}$ of being in bucket i

- $E\left[\sum_{j=1}^{|S|} \frac{\sigma_j^i}{|S|}\right] = \frac{\left(|S| \cdot \frac{|B_i|}{n}\right)}{|S|} = \frac{|B_i|}{n}$

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|S| = \Theta\left(\sqrt{n} \cdot \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^4} \log^2 n\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

With **enough samples** from the bucket **we can estimate the size of the bucket**!

- $E\left[\sum_{j=1}^{|S|} \frac{\sigma_j^i}{|S|}\right] = \frac{\left(|S| \cdot \frac{|B_i|}{n}\right)}{|S|} = \frac{|B_i|}{n}$

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \mathbf{log}^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Do we get enough samples?

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \mathbf{log}^{\mathbf{2}}\, \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Do we get enough samples?
  - $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{\sqrt{n}}{c \cdot t} \cdot \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^4} \cdot \log^2(n)}$
  - $\geq \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^2} \cdot \log^2(n)$

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \mathbf{log}^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Do we get enough samples?
  - $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{\sqrt{n}}{c \cdot t} \cdot \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^4} \cdot \log^2(n)}$
  - $\geq \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{2}}} \cdot \mathbf{log}^{\mathbf{2}}(\boldsymbol{n})$

Large enough sample using the techniques we've learned (i.e. Chernoff bound and median trick)

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \mathbf{log}^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Do we get enough samples?
  - $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{\sqrt{n}}{c \cdot t} \cdot \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^4} \cdot \log^2(n)}$
  - $\geq \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{2}}} \cdot \mathbf{log}^{\mathbf{2}}(\boldsymbol{n})$

Extra factors of $\log(n)$ is for union bound over all vertices

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \mathbf{log}^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Do we get enough samples?
  - $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{\sqrt{n}}{c \cdot t} \cdot \log\left(\frac{1}{\delta}\right) \cdot \frac{1}{\varepsilon^4} \cdot \log^2(n)}$
  - $\geq \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{2}}} \cdot \mathbf{log}^{\mathbf{2}}(\boldsymbol{n})$

Hence, we get $(1+\varepsilon)$-approx. of $\frac{|B_i|}{n}$

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \log\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^{2} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Show what our estimate gets:

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^{2} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \dots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Show what our estimate gets:
  - First, $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1} \leq \overline{d}$
    - **Accurate $\boldsymbol{\rho_i}$**
    - This is because degree in each bucket i lower bounded by $(1+\beta)^{i-1}$

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \log^2 \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Show what our estimate gets:
  - First, $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1} \leq \bar{d}$
    - **Accurate $\boldsymbol{\rho_i}$**
    - This is because degree in each bucket i lower bounded by $(1+\beta)^{i-1}$
  - **Approximate $\boldsymbol{\rho_i}$** is $(\mathbf{1}+\boldsymbol{\varepsilon})$-estimate of $\frac{|B_i|}{n}$

$$\boxed{\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1} \leq (1+\varepsilon) \cdot \bar{d}}$$

A $(2+\varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{4}} \log^2 \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \ldots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Show what our estimate gets:
  - First, $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1} \geq (1-\beta) \cdot \overline{d}$
    - **Accurate $\boldsymbol{\rho_i}$**
    - This is because degree in each bucket i lower bounded by $(1+\beta)^{i-1}$

A $(2 + \varepsilon)$-Approximate Algorithm

- **Algorithm:**
  - Take $|\boldsymbol{S}| = \boldsymbol{\Theta}\left(\sqrt{\boldsymbol{n}} \cdot \mathbf{log}\left(\frac{\mathbf{1}}{\boldsymbol{\delta}}\right) \cdot \frac{\mathbf{1}}{\boldsymbol{\varepsilon}^{\mathbf{4}}} \mathbf{log}^{\mathbf{2}} \boldsymbol{n}\right)$ **samples**
  - For $i \in \{0, \dots, t-1\}$:
    - $S_i \leftarrow S \cap B_i$
    - If $|S_i| \geq \sqrt{\frac{\varepsilon}{n} \cdot \frac{|S|}{c \cdot t}}$, then set $\rho_i \leftarrow \frac{|S_i|}{|S|}$
    - Else, $\rho_i \leftarrow 0$
  - Return $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1}$

- Show what our estimate gets:
  - First, $\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1} \geq (1-\beta) \cdot \bar{d}$
    - **Accurate $\boldsymbol{\rho_i}$**
    - This is because degree in each bucket i lower bounded by $(1+\beta)^{i-1}$
  - **Approximate $\boldsymbol{\rho_i}$** is $(\mathbf{1} + \boldsymbol{\varepsilon})$-estimate of $\frac{|B_i|}{n}$

$$\boxed{\sum_{i=0}^{t-1} \rho_i (1+\beta)^{i-1} \geq (1-\varepsilon)^2 \cdot \bar{d}}$$

A $(2 + \varepsilon)$-Approximate Algorithm

- How much do we lose from our **ignored buckets**?
  - **Big-big:** both endpoints belong to big buckets
    - **Counted from both sides**

A $(2 + \varepsilon)$-Approximate Algorithm

- How much do we lose from our **ignored buckets**?
  - **Big-big:** both endpoints belong to big buckets
    - **Counted from both sides**
  - **Big-small:** one endpoints in big bucket one in small
    - **Count from one side**
      - **2-approx**

A $(2+\varepsilon)$-Approximate Algorithm

Fine given our **assumption that $\overline{d} \geq 1$!**

- How much do we lose from our **ignored buckets**?
  - **Big-big:** both endpoints belong to big buckets
    - **Counted from both sides**
  - **Big-small:** one endpoints in big bucket one in small
    - **Count from one side**
      - **2-approx**
- How much do we lose from our **ignored buckets**?
  - **Small-small:** both endpoints in small buckets

A $(2+\varepsilon)$-Approximate Algorithm

Fine given our **assumption that $\overline{d} \geq 1$!**

- How much do we lose from our **ignored buckets**?
  - **Big-big:** both endpoints belong to big buckets
    - **Counted from both sides**
  - **Big-small:** one endpoints in big bucket one in small
    - **Count from one side**
      - **2-approx**

- How much do we lose from our **ignored buckets**?
  - **Small-small:** both endpoints in small buckets
  - $|B_i| \leq \frac{2(\sqrt{\varepsilon \cdot n})}{ct}$ **whp**

A $(2+\varepsilon)$-Approximate Algorithm

Fine given our **assumption that $\overline{d} \geq 1$!**

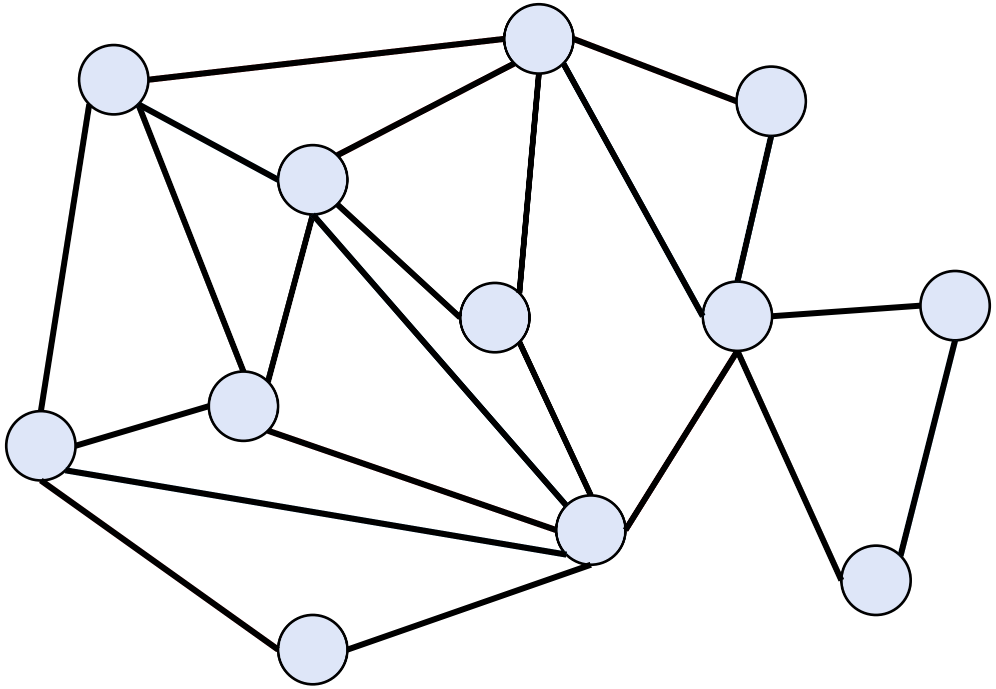
- How much do we lose from our **ignored buckets**?
  - **Big-big:** both endpoints belong to big buckets
    - **Counted from both sides**
  - **Big-small:** one endpoints in big bucket one in small
    - **Count from one side**
      - **2-approx**

- How much do we lose from our **ignored buckets**?
  - **Small-small:** both endpoints in small buckets
  - $|B_i| \leq \frac{2\left(\sqrt{\varepsilon \cdot n}\right)}{ct}$ **whp**
  - At most $t \cdot 2 \cdot \frac{\sqrt{\varepsilon \cdot n}}{ct} = \frac{2\sqrt{\varepsilon n}}{c}$ nodes in small buckets

At most $\left(\frac{2\sqrt{\varepsilon n}}{c}\right)^2 = O(\varepsilon \cdot n)$ edges

Food for Thought Till Next Time

- $(1+\varepsilon)$-approx. for average degree + useful graph property!



Orient all edges from low to high degree, what's the max out-degree that you see?