Graph Properties
Testing if a Graph is Connected [Goldreich Ron]

Input: a graph $G = (V, E)$ on $n$ vertices
  • in adjacency lists representation (a list of neighbors for each vertex)
  • maximum degree $d$, i.e., adjacency lists of length $d$ with some empty entries

Query $(v, i)$, where $v \in V$ and $i \in [d]$: entry $i$ of adjacency list of vertex $v$

Exact Answer: $\Omega(dn)$ time

• Approximate version:

  Is the graph connected or $\epsilon$-far from connected?

  $$\text{dist}(G_1, G_2) = \frac{\# \text{ of entries in adjacency lists on which } G_1 \text{ and } G_2 \text{ differ}}{dn}$$

  Time: $O\left(\frac{1}{\epsilon^2 d}\right)$ today

  No dependence on $n!$ + improvement on HW
Randomized Approximation: a Toy Example

Input: a string \( w \in \{0,1\}^n \)

Goal: Estimate the fraction of 1’s in \( w \) (like in polls)

It suffices to sample \( s = \frac{1}{\epsilon^2} \) positions and output the average to get the fraction of 1’s \( \pm \epsilon \) (i.e., additive error \( \epsilon \)) with probability \( \geq \frac{2}{3} \)

### Hoeffding Bound

Let \( Y_1, ..., Y_s \) be independently distributed random variables in \([0,1]\) and let \( Y = \sum_{i=1}^{s} Y_i \) (sample sum). Then \( \Pr[|Y - E[Y]| \geq \delta] \leq 2e^{-2\delta^2/s} \).

\( Y_i = \) value of sample \( i \). Then \( E[Y] = \sum_{i=1}^{s} E[Y_i] = s \cdot \) (fraction of 1’s in \( w \))

\( \Pr[|\text{(sample average)} - \text{(fraction of 1’s in } w\text{)}| \geq \epsilon] = \Pr[|Y - E[Y]| \geq \epsilon s] \leq 2e^{-2\delta^2/s} = 2e^{-2} < 1/3 \)

Apply Hoeffding Bound with \( \delta = \epsilon s \) substitute \( s = \frac{1}{\epsilon^2} \)
Approximating # of Connected Components

[Chazelle Rubinfeld Trevisan]

**Input:** a graph $G = (V, E)$ on $n$ vertices
- in adjacency lists representation (a list of neighbors for each vertex)
- maximum degree $d$

**Exact Answer:** $\Omega(dn)$ time

**Additive approximation:** # of CC $\pm \varepsilon n$
with probability $\geq 2/3$

**Time:**
- Known: $O\left(\frac{d}{\varepsilon^2} \log \frac{1}{\varepsilon}\right)$, $\Omega\left(\frac{d}{\varepsilon^2}\right)$
- Today: $O\left(\frac{d}{\varepsilon^3}\right)$.

Partially based on slides by Ronitt Rubinfeld:
Approximating # of CCs: Main Idea

• Let $C =$ number of components
• For every vertex $u$, define $n_u =$ number of nodes in $u$’s component
  – for each component $A$: $\sum_{u \in A} \frac{1}{n_u} = 1$
  \[ \sum_{u \in V} \frac{1}{n_u} = C \]
• Estimate this sum by estimating $n_u$’s for a few random nodes
  – If $u$’s component is small, its size can be computed by BFS.
  – If $u$’s component is big, then $1/n_u$ is small, so it does not contribute much to the sum
  – Can stop BFS after a few steps

Similar to property tester for connectedness [Goldreich Ron]
Approximating # of CCs: Algorithm

Estimating $n_u$ = the number of nodes in $u$’s component:

- Let estimate $\hat{n}_u = \min\left\{n_u, \frac{2}{\varepsilon}\right\}$
  - When $u$’s component has $\leq \frac{2}{\varepsilon}$ nodes, $\hat{n}_u = n_u$
  - Else $\hat{n}_u = \frac{2}{\varepsilon}$, and so $0 < \frac{1}{\hat{n}_u} - \frac{1}{n_u} < \frac{1}{\hat{n}_u} = \frac{\varepsilon}{2}$

- Corresponding estimate for $C$ is $\hat{C} = \sum_{u \in V} \frac{1}{\hat{n}_u}$. It is a good estimate:
  \[
  |\hat{C} - C| = \left|\sum_{u \in V} \frac{1}{\hat{n}_u} - \sum_{u \in V} \frac{1}{n_u}\right| \leq \sum_{u \in V} \left|\frac{1}{\hat{n}_u} - \frac{1}{n_u}\right| \leq \frac{\varepsilon n}{2}
  \]

**APPROX_#_CCs (G, d, \varepsilon)**

1. **Repeat** $s = \Theta(\frac{1}{\varepsilon^2})$ times:
2. pick a random vertex $u$
3. compute $\hat{n}_u$ via BFS from $u$, stopping after at most $\frac{2}{\varepsilon}$ new nodes
4. **Return** $\tilde{C} = \left(\text{average of the values } \frac{1}{\hat{n}_u}\right) \cdot n$

**Run time:** $O(d / \varepsilon^3)$
Want to show: $\Pr \left[ |\tilde{C} - \hat{C}| > \frac{\varepsilon n}{2} \right] \leq \frac{1}{3}$

**Hoeffding Bound**

Let $Y_1, \ldots, Y_s$ be independently distributed random variables in $[0,1]$ and let $Y = \sum_{i=1}^{s} Y_i$ (sample sum). Then $\Pr[|Y - E[Y]| \geq \delta] \leq 2e^{-2\delta^2/s}$.

Let $Y_i = 1/\hat{n}_u$ for the $i^{th}$ vertex $u$ in the sample.

- $Y = \sum_{i=1}^{s} Y_i = \frac{s\tilde{C}}{n}$ and $E[Y] = \sum_{i=1}^{s} E[Y_i] = s \cdot E[Y_1] = s \cdot \frac{1}{n} \sum_{u \in V} \frac{1}{\hat{n}_v} = \frac{s\tilde{C}}{n}$

$$\Pr \left[ |\tilde{C} - \hat{C}| > \frac{\varepsilon n}{2} \right] = \Pr \left[ \left| \frac{n}{s} Y - \frac{n}{s} E[Y] \right| > \frac{\varepsilon n}{2} \right] = \Pr \left[ |Y - E[Y]| > \frac{\varepsilon s}{2} \right] \leq 2e^{-\frac{\varepsilon^2 s}{2}}$$

- Need $s = \Theta \left( \frac{1}{\varepsilon^2} \right)$ samples to get probability $\leq \frac{1}{3}$
Approximating # of CCs: Analysis

So far: \[ |\hat{C} - C| \leq \frac{\varepsilon n}{2} \]

\[ \Pr \left[ |\tilde{C} - \hat{C}| > \frac{\varepsilon n}{2} \right] \leq \frac{1}{3} \]

• With probability \( \geq \frac{2}{3} \),

\[ |\tilde{C} - C| \leq |\tilde{C} - \hat{C}| + |\hat{C} - C| \leq \frac{\varepsilon n}{2} + \frac{\varepsilon n}{2} \leq \varepsilon n \]

Summary:

The number of connected components in \( n \)-vertex graphs of degree at most \( d \) can be estimated within \( \pm \varepsilon n \) in time \( O \left( \frac{d}{\varepsilon^3} \right) \).
Minimum spanning tree (MST)

• What is the cheapest way to connect all the dots?

Input: a weighted graph with n vertices and m edges

• Exact computation:
  – Deterministic $O(m \cdot \text{inverse-Ackermann}(m))$ time [Chazelle]
  – Randomized $O(m)$ time [Karger Klein Tarjan]
Approximating MST Weight in Sublinear Time

[Chazelle Rubinfeld Trevisan]

Input: a graph \( G = (V, E) \) on \( n \) vertices
- in adjacency lists representation
- maximum degree \( d \) and maximum allowed weight \( w \)
- weights in \( \{1, 2, \ldots, w\} \)

Output: \((1+\varepsilon)\)-approximation to MST weight, \( W_{MST} \)

Time:
- Known: \( O\left(\frac{dw}{\varepsilon^3} \log \frac{dw}{\varepsilon}\right) \), \( \Omega\left(\frac{dw}{\varepsilon^2}\right) \)
- Today: \( O\left(\frac{dw^3 \log w}{\varepsilon^3}\right) \)

No dependence on \( n \)!
Idea Behind Algorithm

• Characterize MST weight in terms of number of connected components in certain subgraphs of $G$

• Already know that number of connected components can be estimated quickly
MST and Connected Components: Warm-up

• Recall Kruskal’s algorithm for computing MST exactly.

Suppose all weights are 1 or 2. Then MST weight
\[
= \text{(\# weight-1 edges in MST)} + 2 \cdot \text{(\# weight-2 edges in MST)} \\
= n - 1 + \text{(\# of weight-2 edges in MST)} \\
= n - 1 + \text{(\# of CCs induced by weight-1 edges)} - 1
\]

By Kruskal
MST and Connected Components

In general:

Let \( G_i \) = subgraph of \( G \) containing all edges of weight \( \leq i \)

\( C_i \) = number of connected components in \( G_i \)

Then MST has \( C_i - 1 \) edges of weight \( > i \).

Claim

\[
w_{MST}(G) = n - w + \sum_{i=1}^{w-1} C_i
\]

- Let \( \beta_i \) be the number of edges of weight \( > i \) in MST
- Each MST edge contributes 1 to \( w_{MST} \), each MST edge of weight \( > 1 \) contributes 1 more, each MST edge of weight \( > 2 \) contributes one more, ...

\[
w_{MST}(G) = \sum_{i=0}^{w-1} \beta_i = \sum_{i=0}^{w-1} (C_i - 1) = -w + \sum_{i=0}^{w-1} C_i = n - w + \sum_{i=1}^{w-1} C_i
\]
## Algorithm for Approximating $W_{MST}$

**APPROX_MSTweight** ($G, w, d, \varepsilon$)

1. For $i = 1$ to $w - 1$ do:
2. \[ \tilde{C}_i \leftarrow \text{APPROX\_#CCs}(G_i, d, \varepsilon/w) \]
3. Return \[ \tilde{w}_{MST} = n - w + \sum_{i=1}^{w-1} \tilde{C}_i \]

**Claim.** $w_{MST}(G) = n - w + \sum_{i=1}^{w-1} C_i$

### Analysis:

- Suppose all estimates of $C_i$’s are good: $|\tilde{C}_i - C_i| \leq \frac{\varepsilon}{w} n$.

  Then $|\tilde{w}_{MST} - w_{MST}| = |\sum_{i=1}^{w-1} (\tilde{C}_i - C_i)| \leq \sum_{i=1}^{w-1} |\tilde{C}_i - C_i| \leq w \cdot \frac{\varepsilon}{w} n = \varepsilon n$

- $\Pr[\text{all } w - 1 \text{ estimates are good}] \geq (2/3)^{w-1}$

- Not good enough! Need error probability $\leq \frac{1}{3w}$ for each iteration

- Then, by Union Bound, $\Pr[\text{error}] \leq w \cdot \frac{1}{3w} = \frac{1}{3}$

Can amplify success probability of any algorithm by repeating it and taking the median answer.

Can take more samples in **APPROX\_#CCs**. What’s the resulting run time?
Multiplicative Approximation for $w_{MST}$

For MST cost, additive approximation $\Rightarrow$ multiplicative approximation

$$w_{MST} \geq n - 1 \quad \Rightarrow \quad w_{MST} \geq n/2 \text{ for } n \geq 2$$

- $\varepsilon n$-additive approximation:

  $$w_{MST} - \varepsilon n \leq \hat{w}_{MST} \leq w_{MST} + \varepsilon n$$

- $(1 \pm 2\varepsilon)$-multiplicative approximation:

  $$w_{MST}(1 - 2\varepsilon) \leq w_{MST} - \varepsilon n \leq \hat{w}_{MST} \leq w_{MST} + \varepsilon n \leq w_{MST}(1 + 2\varepsilon)$$