LECTURE 3

Last time

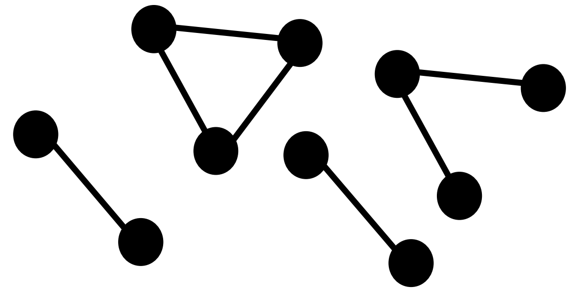
- Properties of lists and functions.
- Testing if a list is sorted/Lipschitz and if a function is monotone.

Today

- Testing if a graph is connected.
- Estimating the number of connected components.
- Estimating the weight of a MST

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: Ω(dn) time

- Approximate version:

Is the graph connected or ϵ-far from connected?

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

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

No dependence on n!

+ improvement on HW

Testing Connectedness: Algorithm

Connectedness Tester(n, d, ε, query access to G)

1. **Repeat** s=8/εd times:
2. pick a random vertex u
3. determine if connected component of u is small:

   perform BFS from u, stopping after at most 4/εd new nodes
4. **Reject** if a small connected component was found, otherwise **accept.**

Run time: O($d/\varepsilon^2 d^2$)=O($1/\varepsilon^2 d$)

Analysis:

- Connected graphs are always accepted.
- Remains to show:

  If a graph is ϵ-far from connected, it is rejected with probability $\geq \frac{2}{3}$

Testing Connectedness: Analysis

Claim 1

If G is ε-far from connected, it has $\geq \frac{\varepsilon dn}{2}$ connected components.

Claim 2

If G is ε-far from connected, it has $\geq \frac{\varepsilon dn}{4}$ connected components of size at most 4/εd.

- By Claim 2, at least $\frac{\varepsilon dn}{4}$ nodes are in small connected components.
- By Witness lemma, it suffices to sample $\frac{2 \cdot 4}{\varepsilon dn/n} = \frac{8}{\varepsilon d}$ nodes to detect one from a small connected component.

Testing Connectedness: Proof of Claim 1

Claim 1

If G is ε-far from connected, it has $\geq \frac{\varepsilon d n}{2}$ connected components.

We prove the contrapositive:

If G has $< \frac{\varepsilon d n}{2}$ connected components, one can make G connected by modifying < ε fraction of its representation, i.e., $< \varepsilon d n$ entries.

- If there are no degree restrictions, k components can be connected by adding k-1 edges, each affecting 2 nodes. Here, $k < \frac{\varepsilon d n}{2}$, so $2k - 2 < \varepsilon d n$.
- What if adjacency lists of all vertices in a component are full,

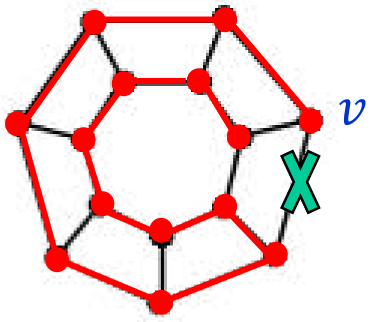
  i.e., all vertex degrees are d?

Freeing up an Adjacency List Entry

Claim 1

If G is ε-far from connected, it has $\geq \frac{\varepsilon dn}{2}$ connected components.

What if adjacency lists of all vertices in a component are full,
i.e., all vertex degrees are d?



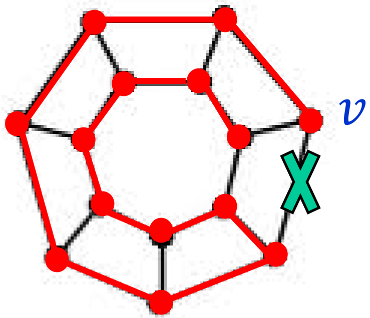
- Consider an MST of this component.
- Let v be a leaf of the MST.
- Disconnect v from a node other than its parent in the MST.
- Two entries are changed while keeping the same number of components.

Freeing up an Adjacency List Entry

Claim 1

If G is ε-far from connected, it has $\geq \frac{\varepsilon d n}{2}$ connected components.

What if adjacency lists of all vertices in a component are full, i.e., all vertex degrees are d?



- Apply this to each component with <2 free spots in adjacency lists.
- Now we can connect all the components using the freed up spots while ensuring that we never change more than 2 spots per component.
- Thus, k components can be connected by changing 2k spots.

Here, $k < \frac{\varepsilon d n}{2}$, so $2k < \varepsilon d n$.

Testing Connectedness: Proof of Claim 2

Claim 1

If G is ε-far from connected, it has $\geq \frac{\varepsilon dn}{2}$ connected components.

Claim 2

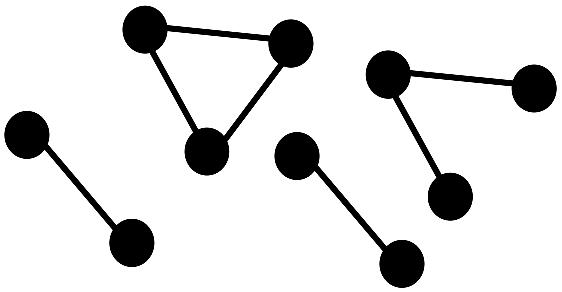
If G is ε-far from connected, it has $\geq \frac{\varepsilon dn}{4}$ connected components of size at most 4/εd.

- By Claim 1, there are at least $\frac{\varepsilon dn}{2}$ connected components.
- Their average size is at most $\frac{n}{\varepsilon dn/2} = \frac{2}{\varepsilon d}$.
- By an averaging argument (or Markov inequality), at least half of the components are of size at most twice the average.

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*



Connected or

ε-far from connected?

$O\left(\frac{1}{\varepsilon^2 d}\right)$ time ✓

(no dependence on n)

Randomized Approximation in sublinear time

A Simple Example

Randomized Approximation: a Toy Example

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

0	0	0	1	…	0	1	0	0

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

It suffices to sample $s = 1 / \varepsilon^2$ positions and output the average
to get the fraction of 1's $\pm\varepsilon$ (i.e., additive error ε) with probability $\geq$ 2/3

Hoeffding Bound

Let $Y_1, \dots, Y_s$ be independently distributed random variables in [0,1].

Let $Y = \frac{1}{s} \cdot \sum_{i=1}^{s} Y_i$ (called *sample mean*). Then $\Pr[|Y - E[Y]| \geq \varepsilon] \leq 2e^{-2s\varepsilon^2}$.

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

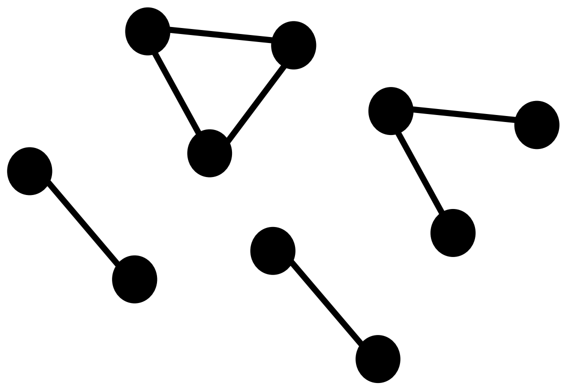
$$\Pr[|(\text{sample mean}) - (\text{fraction of 1's in } w)| \geq \varepsilon]$$
$$\leq 2e^{-2s\varepsilon^2} = 2e^{-2} < 1/3$$

Apply Hoeffding Bound (for $\leq$)

substitute $s = 1 / \varepsilon^2$ (for $=$)

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: Ω(dn) time

Additive approximation: # of CC ±εn

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: $O\left(\frac{d}{\varepsilon^3}\right)$.

No dependence on n!

Partially based on slides by Ronitt Rubinfeld:
http://stellar.mit.edu/S/course/6/fa10/6.896/courseMaterial/topics/topic3/lectureNotes/lecst11/lecst11.pdf

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

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

Breaks C up into contributions of different nodes

- 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 $\leq 2/\varepsilon$ nodes, $\hat{n}_u = n_u$
  - Else $\hat{n}_u = 2/\varepsilon$, and so $0 < \frac{1}{\hat{n}_u} - \frac{1}{n_u} < \frac{1}{\hat{n}_u} = \frac{\varepsilon}{2}$

  $\left|\frac{1}{\hat{n}_u} - \frac{1}{n_u}\right| \leq \frac{\varepsilon}{2}$

- Corresponding estimate for C is $\hat{C} = \sum_{u\in V} \frac{1}{\hat{n}_u}$. It is a good estimate:

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

APPROX_#_CCs (n, d, ε, query access to G)

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

Run time: O(d /ε^3)

Approximating # of CCs: Analysis

Want to show: $\Pr\left[\left|\tilde{C} - \hat{C}\right| > \frac{\varepsilon n}{2}\right] \le \frac{1}{3}$ ✓

Hoeffding Bound

Let $Y_1, \dots, Y_s$ be independently distributed random variables in [0,1].

Let $Y = \frac{1}{s} \cdot \sum_{i=1}^{s} Y_i$ (called *sample mean*). Then $\Pr[|Y - E[Y]| \ge \varepsilon] \le 2e^{-2s\varepsilon^2}$.

Let $Y_i = 1/\hat{n}_u$ for the ith vertex u in the sample

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

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

- Need $s = \Theta\left(\frac{1}{\varepsilon^2}\right)$ samples to get probability $\le \frac{1}{3}$

Approximating # of CCs: Analysis

So far: $\left|\hat{C} - C\right| \leq \frac{\varepsilon n}{2}$

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

- With probability $\geq \frac{2}{3}$,

$$\left|\tilde{C} - C\right| \leq \left|\tilde{C} - \hat{C}\right| + \left|\hat{C} - C\right| \leq \frac{\varepsilon n}{2} + \frac{\varepsilon n}{2} \leq \varepsilon n \checkmark$$

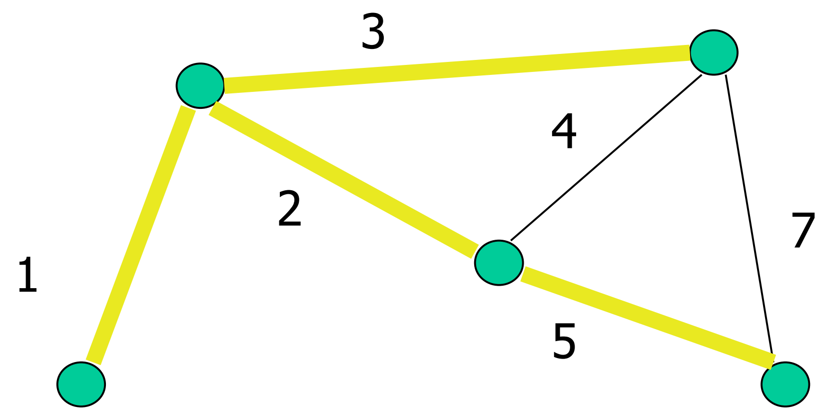
Summary:

The number of connected components in n-vetex 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]

Partially based on slides by Ronitt Rubinfeld:
http://stellar.mit.edu/S/course/6/fa10/6.896/courseMaterial/topics/topic3/lectureNotes/lecst11/lecst11.pdf

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,…,*w*}

Output: (1+ ε)-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: $O\left(\frac{dw^4 \log w}{\varepsilon^3}\right)$

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

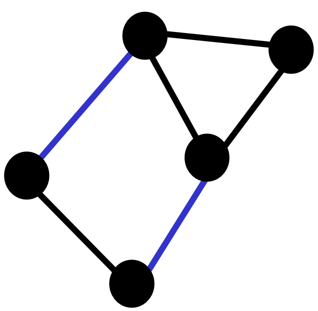
= (# weight-1 edges in MST) + 2 · (# weight-2 edges in MST)

$= \ n - 1 \ +$ (# of weight-2 edges in MST)

$= \ n - 1 \ +$ (# of CCs induced by weight-1 edges) -1

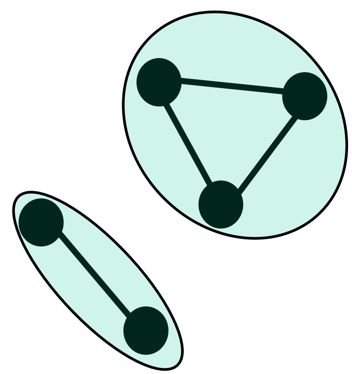
MST has $n - 1$ edges

By Kruskal

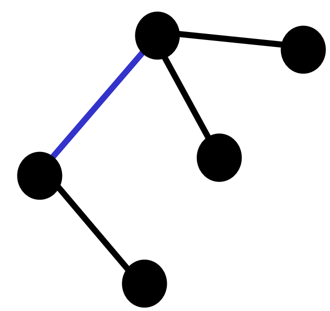


weight 1

weight 2



connected components
induced by weight-1 edges



MST

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 β_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 (n, d, w, ε; G)

1. **For** $i = 1$ to $w - 1$ **do**:
2. $\qquad \tilde{C}_i \leftarrow$ APPROX_#CCs$(n, d, \frac{\varepsilon}{w}; G_i)$.
3. **Return** $\widetilde{w}_{MST} = n - w + \sum_{i=1}^{w-1} \tilde{C}_i$.

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

Analysis:

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

  Then $|\widetilde{w}_{MST} - w_{MST}| = |\sum_{i=1}^{w-1}(\tilde{C}_i - C_i)| \leq \sum_{i=1}^{w-1} |\tilde{C}_i - C_i| \leq w \cdot \frac{\varepsilon}{w}\, n = \varepsilon n$
- Pr[all $w - 1$ 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[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 $\Longrightarrow$ multiplicative approximation

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

- εn-additive approximation:

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

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

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