Algorithms at Scale
(Week 1)

Puzzle of the Day:
Connect the dots.
Use ?? straight lines.
Don’t lift your pen from the paper.

5 x 4: Use 7 lines.
Can you end at the same place you began?

7 x 7: Use 12 lines
Sublinear Time / Sampling Algorithms

Basic question:

What can we do when we look at only a small part of our input data?

Examples:

• Given a graph... only look at a constant number of nodes/edges.
• Is the graph connected?
• How many connected components does it have?
• What is the weight of the MST?
• What is the average degree of the graph?
• What is the diameter of the graph?
• What is the best matching on the graph?
<table>
<thead>
<tr>
<th></th>
<th>Classical</th>
<th>Sublinear Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is the graph connected?</td>
<td></td>
<td></td>
</tr>
<tr>
<td># connected components?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weight of MST?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **n**: number of nodes
- **m**: number of edges
- **d**: degree of graph
- **W**: weight of MST
- **ε**: error / approximation parameter
## Comparison

<table>
<thead>
<tr>
<th></th>
<th>Classical</th>
<th>Sublinear Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is the graph connected?</td>
<td>$O(n + m)$</td>
<td></td>
</tr>
<tr>
<td># connected components?</td>
<td>$O(n + m)$</td>
<td></td>
</tr>
<tr>
<td>Weight of MST?</td>
<td>$O(m \log n)$</td>
<td>( W ) : weight of MST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \epsilon ) : error / approximation parameter</td>
</tr>
</tbody>
</table>

- \( n \) : number of nodes
- \( m \) : number of edges
- \( d \) : degree of graph
## Comparison

<table>
<thead>
<tr>
<th></th>
<th>Classical</th>
<th>Sublinear Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is the graph connected?</td>
<td>$O(n + m)$</td>
<td>$O\left(\frac{1}{\epsilon^2 d}\right)$</td>
</tr>
<tr>
<td># connected components?</td>
<td>$O(n + m)$</td>
<td>$O\left(\frac{d}{\epsilon^3}\right)$</td>
</tr>
<tr>
<td>Weight of MST?</td>
<td>$O(m \log n)$</td>
<td>$O\left(\frac{dW^4 \log W}{\epsilon^3}\right)$</td>
</tr>
</tbody>
</table>

- $n$: number of nodes
- $m$: number of edges
- $d$: degree of graph
- $W$: weight of MST
- $\epsilon$: error / approximation parameter
## Comparison

<table>
<thead>
<tr>
<th></th>
<th>Classical</th>
<th>Sublinear Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Is the graph connected?</strong></td>
<td>$O(n + m)$</td>
<td>$O\left(\frac{1}{\varepsilon^2 d}\right)$</td>
</tr>
<tr>
<td><strong># connected components?</strong></td>
<td>$O(n + m)$</td>
<td>$O\left(\frac{d}{\varepsilon^3}\right)$</td>
</tr>
<tr>
<td><strong>Weight of MST?</strong></td>
<td>$O\left(m \log n\right)$</td>
<td>$O\left(\frac{dW^4 \log W}{\varepsilon^3}\right)$</td>
</tr>
</tbody>
</table>

- $n$: number of nodes
- $m$: number of edges
- $d$: degree of graph
- $W$: weight of MST
- $\varepsilon$: error / approximation parameter

*does not matter how big the graph is!*
### Comparison

<table>
<thead>
<tr>
<th></th>
<th>Classical</th>
<th>Sublinear Approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average degree?</td>
<td>( O(n + m) )</td>
<td>( O\left(\frac{\sqrt{n}}{\epsilon^{9/2}}\right) )</td>
</tr>
<tr>
<td>Diameter?</td>
<td>( O(n(m + n)) )</td>
<td>( O\left(\frac{1}{\epsilon^3}\right) )</td>
</tr>
<tr>
<td>Maximum matching?</td>
<td>( O(mn^2) )</td>
<td>( O\left(\frac{d^4}{\epsilon^2}\right) )</td>
</tr>
</tbody>
</table>

- \( n \): number of nodes
- \( m \): number of edges
- \( d \): degree of graph
- \( W \): weight of MST
- \( \epsilon \): error / approximation parameter
Algorithms

Today

Toy example 1: array all 0’s?
• Gap-style question:
  All 0’s or far from all 0’s?
Toy example 2: Faction of 1’s?
• Additive $\pm \varepsilon$ approximation
• Hoeffding Bound
Is the graph connected?
• Gap-style question.
• $O(1)$ time algorithm.
• Correct with probability $2/3$.

Next week:

Number of connected components in a graph.
• Additive approximation algorithm.
Weight of MST
• Multiplicative approximation algorithm.
Trade-off: speed vs. accuracy

Approximate solutions:

Example: relative error

\[ MST(G)(1 - \epsilon) \leq ALG(G) \leq MST(G)(1 + \epsilon) \]

Example: absolute error

\[ MST(G) - \epsilon \leq ALG(G) \leq MST(G) + \epsilon \]

Example: gap error

• If G is connected, then return \text{TRUE}.
• If G is \textit{\epsilon-far} from connected, then return \text{FALSE}.
• Otherwise, don’t care.

\( MST(G) = \) weight of MST
\( ALG(G) = \) weight of spanning tree returned by algorithm
Warm-Up Problem: All Zeros

Assumptions:

Given n element array
• Each element is 0 or 1.
• 0 = good test.
• 1 = failed test.

Output: Is the array all 0?
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |

Algorithm: Check every cell.

Running time: $O(n)$

Can we do any better?
Warm-Up Problem: All Zeros

Algorithm: Check every cell.

Running time: $O(n)$

Can we do any better? No!

Lower bound: $\Omega(n)$

Challenge #1: Prove it.
Warm-Up Problem: All Zeros

Output: Is the array mostly 0?

Relaxed (approximate) version:

if all 0’s : return true
if $\geq \epsilon n$ 1’s : return false
otherwise : return true or false
Warm-Up Problem: All Zeros

Output: Is the array mostly 0?

Relaxed (approximate) version:

```plaintext
if all 0's : return true
if ≥ εn 1's : return false
otherwise : return true or false
```
Warm-Up Problem: All Zeros

Relaxed (approximate) version:

Output: Is the array mostly 0?

if all 0’s : return true
if $\geq \epsilon n$ 1’s : return false
otherwise : return true or false
Warm-Up Problem: All Zeros

Relaxed (approximate) version:

Output: Is the array mostly 0?

```python
if all 0's : return true
if ≥ \( \epsilon n \) 1's : return false
otherwise : return true or false
```
Warm-Up Problem: All Zeros

All-Zeros($A, \epsilon$)

Repeat $s$ times:

Choose random $i$ in $[1, n]$.

if $A[i] = 1$ then return FALSE.

Return TRUE.

Fix $s = \frac{2}{\epsilon}$
Warm-Up Problem: All Zeros

All-Zeros($A, \varepsilon$)

Repeat $s$ times:

Choose random $i$ in $[1, n]$.

if $A[i] = 1$ then return FALSE.

Return TRUE.

Claim 1: If array is all 0’s, then always returns TRUE.
Warm-Up Problem: All Zeros

All-Zeros(A, ε)

Repeat s times:
Choose random i in [1, n].
if A[i] = 1 then return FALSE.
Return TRUE.

Claim 2: If array has \( \geq \varepsilon n \) 1’s, then returns FALSE with probability at least 2/3.
Claim 2: If array has $\geq \varepsilon n$ 1’s, then returns FALSE with probability at least $2/3$.

Proof:
Assume $\geq \varepsilon n$ 1’s.
For sample $i$ : $\Pr[A[i] = 1] \geq \varepsilon n/n \geq \varepsilon$.

Pr\{all samples are 0\} $\leq (1 - \varepsilon)^s$
$\leq (1 - \varepsilon)^{2/\varepsilon}$
$\leq e^{-2}$
$\leq \frac{1}{3}$
Fix $s = \frac{2}{\epsilon}$

**Proof:**

Assume $\geq \epsilon n$ 1's.

For sample $i$: $\Pr[A[i] = 1] \geq \epsilon n/n \geq \epsilon$.

$\Pr\{\text{all samples are 0}\} \leq (1 - \epsilon)^s \\
\leq (1 - \epsilon)^{2/\epsilon} \\
\leq e^{-2} \\
\leq \frac{1}{3}$
Warm-Up Problem: All Zeros

\[
\begin{bmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

**All-Zeros(A, \(\varepsilon\))**

Repeat \(s\) times:

Choose random \(i\) in \([1, n]\).

If \(A[i] = 1\) then return FALSE.

Return TRUE.

Claim 1: If array is all 0’s, then always returns TRUE.

Claim 2: If array has \(\geq \varepsilon n\) 1’s, then returns FALSE with probability at least \(2/3\).
Warm-Up Problem: All Zeros

\[
\begin{array}{cccccccccccccccc}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

All-Zeros(\(A, \varepsilon\))

Repeat \(s\) times:

Choose random \(i\) in \([1, n]\).

if \(A[i] = 1\) then return FALSE.

Return TRUE.

What if we want the algorithm to be correct with probability \(\geq 1 - \delta\)?
Warm-Up Problem: All Zeros

All-Zeros(A, ∈)

Repeat s times:
    Choose random i in [1, n].
    if A[i] = 1 then return FALSE.
Return TRUE.

What if we want the algorithm to be correct with probability \( \geq 1 - \delta \)?

Fix \( s = ? \)
Warm-Up Problem: How Many Zeros

Input: n element array
• Each element is 0 or 1.
• 0 = good test, 1 = failed test.

Output: What fraction of the array is 1’s?

Approximation: ± ε

Probability correct: 2/3
Warm-Up Problem: How Many Zeros

\[ \begin{array}{cccccccccccccc}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
\end{array} \]

\[ \begin{array}{cccccccccccccc}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{array} \]

\[ n = 16 \quad \varepsilon = 0 \quad \text{answer} = \frac{3}{16} \]

\[ n = 16 \quad \varepsilon = \frac{1}{16} \quad \text{answer: } [\frac{1}{8}, \frac{1}{4}] \]

\[ n = 16 \quad \varepsilon = \frac{1}{100} \quad \text{answer: } [0.1775, 0.1975] \]
Fraction-Zeros(A, ε)

sum = 0
Repeat s times:
    Choose random i in [1, n]
    sum = sum + A[i]
Return sum/s

S = \frac{1}{\varepsilon^2}

time = O\left(\frac{1}{\varepsilon^2}\right)
Key tool: Hoeffding Bound

Requirements:

1) Random variables:
   \( Y_1, Y_2, ..., Y_s \) are random variables.
   
   \[ Z = \sum_{j=1}^{s} Y_j \]

2) Independent:
   \( Y_1, Y_2, ..., Y_s \) are independent.

3) Bounded:
   Each \( Y_j \) is in the range [0,1].
Key tool: Hoeffding Bound

Requirements:

1) Random variables:
\[ Y_1, Y_2, \ldots, Y_s \text{ are random variables.} \]

\[ Z = \sum_{j=1}^{s} Y_j \]

2) Independent:
\[ Y_1, Y_2, \ldots, Y_s \text{ are independent.} \]

3) Bounded:
\[ \text{Each } Y_j \text{ is in the range } [0,1]. \]

Example:

1) Random variables:
\[ Y_j = \text{value of array sampled in } j^{\text{th}} \text{ iteration of the loop.} \]

2) Independent:
\[ \text{Each sample is independent.} \]

3) Bounded:
\[ \text{Each array entry is in } [0,1]. \]
If $Y_1, Y_2, \ldots, Y_s$ are independent random variables in the range $[0,1]$ and if $Z = \sum_{j=1}^{s} Y_j$ then:

$$\Pr[|Z - \mathbb{E}[Z]| \geq \delta] \leq 2e^{-2\delta^2 / s}$$

**Conclusion:**

The value of $Z$ is within $\delta$ of its expected value with “good” probability.
Claim:

Imagine you flip a coin \( n \) times.

Then you will see:

- at least \( \frac{n}{2} - \sqrt{n} \) heads
- at most \( \frac{n}{2} + \sqrt{n} \) heads

with probability at least \( \frac{2}{3} \).
Claim:

Imagine you flip a coin \( n \) times.

Then you will see:

- at least \( \frac{n}{2} - \sqrt{n} \) heads
- at most \( \frac{n}{2} + \sqrt{n} \) heads

with probability at least \( \frac{2}{3} \).

Random variables:

1) Random variables:

\[
Y_j = 1 \text{ if flip } j \text{ is heads.} \\
Y_j = 0 \text{ if flip } j \text{ is tails.}
\]

\[
Z = \sum_{j=1}^{n} Y_j
\]

\[
E[Z] = \frac{n}{2}
\]

2) Independent: YES

3) Bounded: YES
Example: Flipping Coins

Hoeffding Bound:

\[ Y_j = 1 \text{ if flip } j \text{ is heads.} \]
\[ Y_j = 0 \text{ if flip } j \text{ is tails.} \]

\[ Z = \sum_{j=1}^{n} Y_j \]

\[ \mathbb{E}[Z] = n/2 \]

\[
\Pr[|Z - n/2| \geq \sqrt{n}] \leq \Pr[|Z - \mathbb{E}[Z]| \geq \sqrt{n}]
\leq 2e^{-2(\sqrt{n})^2/n}
\leq 2e^{-2}
\leq 1/3
\]
Claim:

Imagine you flip a coin \( n \) times.

Then you will see:

- at least \( \frac{n}{2} - \sqrt{n} \) heads
- at most \( \frac{n}{2} + \sqrt{n} \) heads

with probability at least \( \frac{2}{3} \).
If $Y_1, Y_2, ..., Y_s$ are independent random variables in the range $[0,1]$ and if $Z = \sum_{j=1}^{s} Y_j$ then:

$$\Pr[|Z - \mathbb{E}[Z]| \geq \delta] \leq 2e^{-2\delta^2/s}$$

Conclusion:

The value of $Z$ is within $\delta$ of its expected value with “good” probability.
Fraction-Zeros(A, ε)

```
sum = 0
Repeat s times:
    Choose random i in [1, n]
    sum = sum + A[i]
Return sum/s
```

\[ Y_j = 1 \text{ if sample } j \text{ is 1.} \]
\[ Y_j = 0 \text{ if sample } j \text{ is 0.} \]

\[ sum = \sum_{j=1}^{n} Y_j \]

Value returned: \( \frac{sum}{s} \)
Warm-Up Problem: How Many Zeros

Let $f = \text{fraction of 1's in the array.}$

(The "right" answer for the algorithm is $f.$)
Warm-Up Problem: How Many Zeros

Let $f$ = fraction of 1’s in the array.

(The "right" answer for the algorithm is $f$.)
**Fraction-Zeros(A, \( \varepsilon \))**

- \( \text{sum} = 0 \)
- Repeat \( s \) times:
  - Choose random \( i \) in \([1, n]\)
  - \( \text{sum} = \text{sum} + A[i] \)
- Return \( \text{sum}/s \)

\[
Y_j = 1 \text{ if sample } j \text{ is } 1. \\
Y_j = 0 \text{ if sample } j \text{ is } 0. \\
\text{sum} = \sum_{j=1}^{n} Y_j \\
E[\text{sum}] = s \cdot \text{sum} \\
\text{value returned: } \frac{\text{sum}}{s} \\
E[\text{value returned}] = f
\]

Unbiased estimator! That’s good...
Use a Hoeffding Bound!

\[ V = \text{value returned} \]

\[
\Pr[|V - f| \leq \epsilon] = \Pr[|V - \mathbb{E}[V]| \leq \epsilon]
\]

Can we use a Hoeffding Bound here?

\begin{align*}
Y_j & = 1 \text{ if sample } j \text{ is 1.} \\
Y_j & = 0 \text{ if sample } j \text{ is 0.} \\
\text{sum} & = \sum_{j=1}^{n} Y_j \\
\mathbb{E}[\text{sum}] & = s \cdot \text{sum} \\
\text{value returned: } & \frac{\text{sum}}{s} \\
\mathbb{E}[\text{value returned}] & = f
\end{align*}
Use a Hoeffding Bound!

\[ V = \text{value returned} \]

\[
\Pr[|V - f| \leq \epsilon] = \Pr[|V - \mathbb{E}[V]| \leq \epsilon]
\]

Can we use a Hoeffding Bound here?

No, not yet.

\( V \) is not a sum of random variables.
Use a Hoeffding Bound!

\[ V = \text{value returned} \]

\[
\Pr[|V - f| \leq \epsilon] = \Pr[|V - \mathbb{E}[V]| \leq \epsilon] = \Pr[|sV - sf| \leq s \cdot \epsilon]
\]

\[ Y_j = 1 \text{ if sample } j \text{ is } 1, \]
\[ Y_j = 0 \text{ if sample } j \text{ is } 0. \]

\[ \text{sum} = \sum_{j=1}^{n} Y_j \]
\[ \mathbb{E}[\text{sum}] = s \cdot \text{sum} \]

value returned: \[ \text{sum} / s \]
\[ \mathbb{E}[\text{value returned}] = f \]

Multiple everything by \( s \).
Use a Hoeffding Bound!

V = value returned

\[
\begin{align*}
\Pr[|V - f| \leq \epsilon] &= \Pr[|V - E[V] \leq \epsilon] \\
&= \Pr[|sV - sf| \leq s \cdot \epsilon] \\
&= \Pr[|sum - E[sum]| \leq s \cdot \epsilon]
\end{align*}
\]

Can we use a Hoeffding Bound here?

Yes.
The random variable \textit{sum} is the sum of independent 0/1 random variables.

\[Y_j = 1 \text{ if sample } j \text{ is 1.} \]
\[Y_j = 0 \text{ if sample } j \text{ is 0.}\]

\[
sum = \sum_{j=1}^{n} Y_j \\
E[sum] = s \cdot sum
\]

value returned: \(sum/s\)
\[E[\text{value returned}] = f\]
Use a Hoeffding Bound!

\[ V = \text{value returned} \]

\[
\Pr[|V - f| \leq \epsilon] = \Pr[|V - \mathbb{E}[V] | \leq \epsilon] \\
= \Pr[|sV - sf| \leq s \cdot \epsilon] \\
= \Pr[|\text{sum} - \mathbb{E}[\text{sum}]| \leq s \cdot \epsilon] \\
\leq 2e^{-2s^2\epsilon^2/s}
\]
Use a Hoeffding Bound!

$V = \text{value returned}$

$$\Pr[|V - f| \leq \epsilon] = \Pr[|V - \mathbb{E}[V] | \leq \epsilon]$$
$$= \Pr[|sV - sf| \leq s \cdot \epsilon]$$
$$= \Pr[|\text{sum} - \mathbb{E}[\text{sum}]| \leq s \cdot \epsilon]$$
$$\leq 2e^{-2s^2 \epsilon^2 / s}$$
$$\leq 2e^{-2s \epsilon^2}$$

$Y_j = 1$ if sample $j$ is 1.
$Y_j = 0$ if sample $j$ is 0.

$\text{sum} = \sum_{j=1}^{n} Y_j$
$\mathbb{E}[\text{sum}] = s \cdot \text{sum}$

value returned: $\text{sum} / s$
$\mathbb{E}[\text{value returned}] = f$
Use a Hoeffding Bound!

\[ V = \text{value returned} \]

\[
\Pr[|V - f| \leq \varepsilon] = \Pr[|V - \mathbb{E}[V] \leq \varepsilon] \\
= \Pr[|sV - sf| \leq s \cdot \varepsilon] \\
= \Pr[|\text{sum} - \mathbb{E}[\text{sum}]| \leq s \cdot \varepsilon] \\
\leq 2e^{-2s^2\varepsilon^2/s} \\
\leq 2e^{-2s\varepsilon^2} \\
\leq 2e^{-2\varepsilon^2/\varepsilon^2} \\
\]

\[ S = \frac{1}{\varepsilon^2} \]

\[ Y_j = 1 \text{ if sample } j \text{ is 1.} \]
\[ Y_j = 0 \text{ if sample } j \text{ is 0.} \]

\[ \text{sum} = \sum_{j=1}^{n} Y_j \]

\[ \mathbb{E}[\text{sum}] = s \cdot \text{sum} \]

value returned: \( \text{sum}/s \)

\[ \mathbb{E}[\text{value returned}] = f \]
Use a Hoeffding Bound!

V = value returned

\[
\Pr[|V - f| \leq \epsilon] = \Pr[|V - \mathbb{E}[V] | \leq \epsilon] = \Pr[|sV - sf| \leq s \cdot \epsilon] = \Pr[|\text{sum} - \mathbb{E}[\text{sum}] | \leq s \cdot \epsilon] \leq 2e^{-2s^2\epsilon^2}/s \\
\leq 2e^{-2s\epsilon^2} \\
\leq 2e^{-2\epsilon^2}/\epsilon^2 \\
\leq 2e^{-2} \\
\leq 1/3
\]

Conclusion: value returned V is equal to f ± \epsilon w.p. ≥ 2/3.
Fraction-Zeros($A, \varepsilon$)

\[ S = \frac{1}{\varepsilon^2} \]

\[ \text{time} = O \left( \frac{1}{\varepsilon^2} \right) \]

\[ \text{sum} = 0 \]

Repeat $s$ times:

Choose random $i$ in $[1, n]$

\[ \text{sum} = \text{sum} + A[i] \]

Return $\text{sum}/s$

Conclusion: $\text{sum}/s$ is equal to $f \pm \varepsilon$ w.p. $\geq 2/3$. 
# Algorithms

## Today

**Toy example 1:** array all 0’s?
- Gap-style question:
  - All 0’s or far from all 0’s?

**Toy example 2:** Faction of 1’s?
- Additive $\pm \varepsilon$ approximation
- Hoeffding Bound

**Is the graph connected?**
- Gap-style question.
- $O(1)$ time algorithm.
- Correct with probability $2/3$.

## Next week:

**Number of connected components in a graph.**
- Additive approximation algorithm.

**Weight of MST**
- Multiplicative approximation algorithm.
Assumptions:
Graph $G = (V,E)$
- Undirected
- $n$ nodes
- $m$ edges
- maximum degree $d$

Output:
Is the graph connected?

Example: output NO
Connectivity

Assumptions:

Graph $G = (V,E)$
- Undirected
- $n$ nodes
- $m$ edges
- maximum degree $d$

Format:
- Graph is given as an adjacency list.
- Query: $\text{nbr}(u, i)$ returns $i^{\text{th}}$ neighbor of node $u$.

Example: output NO
Connectivity

Exact answer:

BFS solves in: $O(m + n)$

Cannot do faster than: $\Omega(m + n)$

Challenge #2:
Prove it.
Assumptions:

Graph G = (V,E)
- Undirected
- n nodes
- m edges
- maximum degree d

Output:
Is the graph *close to*” connected or “*far from*” connected?

Example: output NO
Connectivity

Gap Approximation:

If $G$ is connected:
   Return TRUE

If $G$ is $\epsilon$-far from connected:
   Return FALSE

Otherwise:
   Don’t care.
Definition:

$G$ is $\varepsilon$-close to connected if you can add/modify at most $\varepsilon n d$ entries in the adjacency list to create a new graph $G'$ that is connected.

Note: adding an edge requires modifying two entries in the adjacency list.
 Connectivity

Definition:

$G$ is $\varepsilon$-close to connected if you can add/modify at most $\varepsilon n d$ entries in the adjacency list to create a new graph $G'$ that is connected.

Note: adding an edge requires modifying two entries in the adjacency list.
Connectivity

Definition:

G is $\epsilon$-close to connected if you can add/modify at most $\epsilon n d$ entries in the adjacency list to create a new graph $G'$ that is connected.

Example: $n = 10$, $d = 3$

Note: adding an edge requires modifying two entries in the adjacency list.
Connectivity

Definition:

$G$ is $\varepsilon$-close to connected if you can add/modify at most $\varepsilon n d$ entries in the adjacency list to create a new graph $G'$ that is connected.

Example: $n = 10$, $d = 3$
Add 3 edges to connect graph.
Modify 6 entries in adjacency list.

Note: adding an edge requires modifying two entries in the adjacency list.
Connectivity

Definition:

$G$ is $\epsilon$-close to connected if you can add/modify at most $\epsilon n d$ entries in the adjacency list to create a new graph $G'$ that is connected.

Example: $n = 10$, $d = 3$
- Add 3 edges to connect graph.
- Modify 6 entries in adjacency list.
- $nd = 30$
- $G$ is $1/5$-close to connected.

Note: adding an edge requires modifying two entries in the adjacency list.
Connectivity

**Definition:**

$G$ is $\varepsilon$-close to connected if you can add/modify at most $\varepsilon$nd entries in the adjacency list to create a new graph $G'$ that is connected.

$G$ is $\varepsilon$-far from connected if it is not $\varepsilon$-close to connected.

*Note: adding an edge requires modifying two entries in the adjacency list.*
Connectivity

Assumptions:
Graph $G = (V,E)$
- Undirected
- $n$ nodes
- $m$ edges
- maximum degree $d$

Output:
If $G$ is connected: return TRUE.
If $G$ is $\varepsilon$-far from connected: return FALSE.
Else: don’t care.

Correct: with probability $\geq 2/3$.

Example: output NO
Lemma:
If $G$ is $\epsilon$-far from connected, then it has $\epsilon dn/4$ connected components.
Lemma:
If G is \( \varepsilon \)-far from connected, then it has \( \geq \varepsilon dn/4 \) connected components.

Proof:
Assume G has \( \leq \varepsilon dn/4 \) connected components.

Then add \( \varepsilon dn/4-1 \) edges to build connected graph \( G' \). That requires modifying \( \leq \varepsilon dn/2 \) entries in adjacency list.

\[ \rightarrow \]
G is \( \varepsilon \)-close to connected.
Key Claim

Lemma: If $G$ is $\epsilon$-far from connected, then it has $\geq \epsilon d n/4$ connected components.

Proof:
Assume $G$ has $\leq \epsilon d n/4$ connected components.

Then add $\epsilon d n/4 - 1$ edges to build connected graph $G'$. That requires modifying $\leq \epsilon d n/2$ entries in adjacency list.

$\Rightarrow$

$G$ is $\epsilon$-close to connected.

Oops!
Cannot always add an edge without increasing the degree of the graph.
Lemma: If $G$ is $\varepsilon$-far from connected, then it has $\geq \varepsilon dn/4$ connected components.

Proof: Assume $G$ has $\leq \varepsilon dn/4$ connected components.

For each connected component, if every node has degree $d$: 
Lemma:
If $G$ is $\varepsilon$-far from connected, then it has $\geq \varepsilon dn/4$ connected components.

Proof:
Assume $G$ has $\leq \varepsilon dn/4$ connected components.

For each connected component, if every node has degree $d$:

If it has $k$ nodes, find a spanning tree with $k-1$ edges. Remove any one edge not in spanning tree.
Lemma:
If $G$ is $\varepsilon$-far from connected, then it has $\geq \varepsilon dn/4$ connected components.

Proof:
Assume $G$ has $\leq \varepsilon dn/4$ connected components.

For each connected component, if every node has degree $d$:
- number of edges $= \frac{k(k-1)}{2} > k - 1$

If it has $k$ nodes, find a spanning tree with $k-1$ edges. Remove any one edge not in spanning tree.
Lemma:
If G is $\varepsilon$-far from connected, then it has $\geq \varepsilon dn/4$ connected components.

Proof:
Assume G has $\leq \varepsilon dn/4$ connected components.
Delete $\leq \varepsilon dn/4$ edges so each connected component has at least one node with degree $< d$.
Then add $\leq \varepsilon dn/4-1$ edges to build connected graph $G'$.
Lemma:
If G is $\varepsilon$-far from connected, then it has $\geq \varepsilon dn/4$ connected components.

Proof:
Assume G has $\leq \varepsilon dn/4$ connected components.

Delete $\leq \varepsilon dn/4$ edges so each connected component has at least one node with degree $< d$.

Then add $\leq \varepsilon dn/4-1$ edges to build connected graph $G'$.

Modifies $\leq \varepsilon dn$ entries in adjacency list.

$\Rightarrow$
G is $\varepsilon$-close to connected.
Key Claim

Lemma:
If $G$ is $\varepsilon$-far from connected, then it has $\varepsilon dn/4$ connected components.
Lemma:
If $G$ is $\epsilon$-far from connected, then it has $\epsilon dn/8$ connected components of size $\leq 8/\epsilon d$. 
Lemma:
If $G$ is $\epsilon$-far from connected, then it has $\epsilon dn/8$ connected components of size $\leq 8/\epsilon d$.

Proof:
Counting argument.
Lemma:
If $G$ is $\varepsilon$-far from connected, then it has $\varepsilon dn/8$ connected components of size $\leq 8/\varepsilon d$.

Proof:
Counting argument.
Assume not.
• There are at least $\varepsilon dn/4$ connected components.

At most half can be twice the average size...
Lemma:
If $G$ is $\varepsilon$-far from connected, then it has $\varepsilon dn/8$ connected components of size $\leq 8/\varepsilon d$.

Proof:
Counting argument.
Assume not.
• There are at least $\varepsilon dn/4$ connected components.
• At most $\varepsilon dn/8$ are of size $\leq 8/\varepsilon d$.
• At least $\varepsilon dn/8$ are of size $> 8/\varepsilon d$. 
Lemma:
If $G$ is $\epsilon$-far from connected, then it has $\epsilon dn/8$ connected components of size $\leq 8/\epsilon d$.

Proof:
Counting argument.
Assume not.
• There are at least $\epsilon dn/4$ connected components.
• At most $\epsilon dn/8$ are of size $\leq 8/\epsilon d$.
• At least $\epsilon dn/8$ are of size $> 8/\epsilon d$.

$\left(\frac{\epsilon dn}{8}\right) \left\lfloor \frac{8}{\epsilon d} + 1 \right\rfloor > n \quad \Rightarrow \quad \text{CONTRADICTION}$
Key Claim 2

Lemma:
If $G$ is $\varepsilon$-far from connected, then it has $\varepsilon d n / 8$ connected components of size $\leq 8 / \varepsilon d$.

Proof:
Counting argument.
Connectivity

\textbf{Connected}(G, n, d, \varepsilon)

Repeat $16/\varepsilon d$ times:

- Choose random node $u$.
- Do a BFS from $u$, stopping after $8/\varepsilon d$ nodes are found.
- If CC of $u$ has $\leq 8/\varepsilon d$ nodes, return FALSE.

Return TRUE
Connected(G, n, d, ε)

Repeat $16/\varepsilon d$ times:

- Choose random node $u$.
- Do a BFS from $u$, stopping after $8/\varepsilon d$ nodes are found.
- If CC of $u$ has $\leq 8/\varepsilon d$ nodes, return FALSE.

Return TRUE

Claim: Each BFS takes time at most $d(8/\varepsilon d)$. 
**Connectivity**

**Connected(G, n, d, ε)**

Repeat $16/\varepsilon d$ times:

- Choose random node $u$.
- Do a BFS from $u$, stopping after $8/\varepsilon d$ nodes are found.
- If CC of $u$ has $\leq 8/\varepsilon d$ nodes, return **FALSE**.

Return **TRUE**

**Claim:** Each BFS takes time at most $d(8/\varepsilon d) = 8/\varepsilon$.

**Proof:** Explore at most $(8/\varepsilon d)$ nodes of degree at most $d$. 
Connected\((G, n, d, \varepsilon)\)

Repeat \(16/\varepsilon d\) times:

- Choose random node \(u\).
- Do a BFS from \(u\), stopping after \(8/\varepsilon d\) nodes are found.
- If CC of \(u\) has \(\leq 8/\varepsilon d\) nodes, return \text{FALSE}.

Return \text{TRUE}

Claim: Total time is \(O(1/\varepsilon^2 d)\).

Proof: 
\[
\left(\frac{16}{\varepsilon d}\right) \left(\frac{8}{\varepsilon}\right) = O \left(\frac{1}{\varepsilon^2 d}\right)
\]
Connected($G, n, d, \varepsilon$)

Repeat $16/\varepsilon d$ times:

• Choose random node $u$.
• Do a BFS from $u$, stopping after $8/\varepsilon d$ nodes are found.
• If CC of $u$ has $\leq 8/\varepsilon d$ nodes, return $\text{FALSE}$.

Return $\text{TRUE}$

Claim: If $G$ is connected, returns $\text{TRUE}$.

Proof: Immediate. No component has $\leq 8/\varepsilon d$ nodes.
Connectivity

Connected\( (G, n, d, \varepsilon) \)

Repeat \( \frac{16}{\varepsilon d} \) times:

- Choose random node \( u \).
- Do a BFS from \( u \), stopping after \( \frac{8}{\varepsilon d} \) nodes are found.
- If CC of \( u \) has \( \leq \frac{8}{\varepsilon d} \) nodes, return \text{FALSE}.

Return \text{TRUE}

Claim: If \( G \) is \( \varepsilon \)-far from connected, then returns \text{FALSE} with probability \( \geq \frac{2}{3} \).

Proof: ...
Claim: If $G$ is $\varepsilon$-far from connected, then returns **FALSE** with probability $\geq 2/3$.

Proof: If $G$ is $\varepsilon$-far from connected, then it has at least $\varepsilon dn/8$ connected components of size $\leq 8/\varepsilon d$. 
Connectivity

Claim: If $G$ is $\varepsilon$-far from connected, then returns $\text{FALSE}$ with probability $\geq 2/3$.

Proof:
If $G$ is $\varepsilon$-far from connected, then it has at least $\varepsilon dn/8$ connected components of size $\geq 1$ and $\leq 8/\varepsilon d$.

Each iteration / sample has probability at least $\frac{\varepsilon dn}{8} \geq \frac{\varepsilon d}{8}$ of finding small connected component and returning $\text{FALSE}$.
Connectivity

Claim: If $G$ is $\varepsilon$-far from connected, then returns $\text{FALSE}$ with probability $\geq 2/3$.

Proof:
Each iteration / sample has probability at least $\frac{\varepsilon dn}{8} \geq \frac{\varepsilon d}{8}$ of finding small connected component and returning $\text{FALSE}$.

$$\Pr[\text{no iteration finds a small connected component}] \leq \left(1 - \frac{\varepsilon d}{8}\right)^s$$

$$\leq \left(1 - \frac{\varepsilon d}{16}\frac{16}{\varepsilon d}\right)$$

$$\leq \left(1 - \frac{\varepsilon d}{8}\right)^{16}$$

$$\leq e^{-2}$$

$$\leq 1/3$$
Connectivity

Claim: If $G$ is $\varepsilon$-far from connected, then returns FALSE with probability $\geq 2/3$.

Proof:
Each iteration / sample has probability at least $\frac{\varepsilon dn}{8} \geq \frac{\varepsilon d}{8}$ of finding small connected component and returning false.

$$\Pr[\text{no iteration finds a small connected component}] \leq \left(1 - \frac{\varepsilon d}{8}\right)^s \leq \left(1 - \frac{\varepsilon d}{8}\right)^{\frac{16}{\varepsilon d}} \leq e^{-2} \leq \frac{1}{3}$$

$\Rightarrow$ Some iteration finds a small CC and returns FALSE with probability at least $2/3$. 

Death Bed Fact: $(0 < x < 1)$

$$e^{-x} = 1 - x + \frac{x^2}{2} - \ldots$$

$$\frac{1}{e^2} \leq \left(1 - \frac{1}{x}\right)^x \leq \frac{1}{e}$$
**Connectivity**

\[ \text{Connected}(G, n, d, \varepsilon) \]

Repeat \( \frac{16}{\varepsilon d} \) times:
- Choose random node \( u \).
- Do a BFS from \( u \), stopping after \( \frac{8}{\varepsilon d} \) nodes are found.
- If CC of \( u \) has \( \leq \frac{8}{\varepsilon d} \) nodes, return \text{FALSE}.

Return \text{TRUE}

**Claim:** If \( G \) is \( \varepsilon \)-far from connected, then returns \text{FALSE} with probability \( \geq \frac{2}{3} \).
Connectivity

**Connected(G, n, d, \(\varepsilon\))**

Repeat \(16/\varepsilon d\) times:

- Choose random node \(u\).
- Do a BFS from \(u\), stopping after \(8/\varepsilon d\) nodes are found.
- If CC of \(u\) has \(\leq 8/\varepsilon d\) nodes, return **FALSE**.

Return **TRUE**

Claim: Total time is \(O(1/\varepsilon^2d)\).

Claim: If \(G\) is connected, returns **TRUE**.

Claim: If \(G\) is \(\varepsilon\)-far from connected, then returns **FALSE** with probability \(\geq 2/3\).
Connectivity

General idea:

• Use sampling and local approximation to understand global graph properties.
• For what other interesting properties can you do this?

Questions to think about:

• Is gap approximation useful?
• Is there a better notion of “close to connected”?
• For what values of $\epsilon$ and $d$ is this actually fast?
• What happens in dense graphs?
• Can you find a faster algorithm? In theory? In practice?
Announcements / Reminders

Problem sets:

Problem Set 1 will be released tomorrow.

Problem Set 1 will be due next week.
Summary

Last Week:

Toy example 1: array all 0’s?
• Gap-style question:
  All 0’s or far from all 0’s?

Toy example 2: Fraction of 1’s?
• Additive $\pm \varepsilon$ approximation
• Hoeffding Bound

Is the graph connected?
• Gap-style question.
• $O(1)$ time algorithm.
• Correct with probability 2/3.

Today:

Number of connected components in a graph.
• Approximation algorithm.

Weight of MST
• Approximation algorithm.