Social Network Analysis

COSC–254 — March 25–??, 2019
Outline

Paths, shortest paths, diameter, and Breadth-First Search

Social networks properties

Centrality and prestige

Closeness centrality

Betweenness centrality

Finding communities with betweenness centrality
Section outline

Paths and shortest paths

Finding shortest paths: Breadth-First Search

2-Approximation algorithm for the diameter
Paths

Graph $G = (V, E)$.

Two vertices $u, v \in V$.

A path $p$ from $u$ to $v$ is an ordered sequence of vertices:

$$p = (u, w_1, \ldots, w_\ell, v)$$

such that $(u, w_1) \in E, (w_i, w_{i+1}) \in E$ for each $i = 1, \ldots, \ell - 1, w_\ell, v \in E$.

$p = (1, 3, 5, 2)$ is a path from 1 to 2
Shortest Paths

Length of a path \( p = (w_1, \ldots, w_\ell) \) is \( \ell - 1 \) (number of edges on the path)

\[ p = (2, 5, 1, 3, 4) \text{ is a path from 2 to 4 of length 4.} \]
Shortest Paths

Length of a path $p = (w_1, \ldots, w_\ell)$ is $\ell - 1$ (number of edges on the path)

The distance $d(u, v)$ from $u$ to $v$ is the *minimum length* of any path from $u$ to $v$

A *shortest path* (SP) from $u$ to $v$ is a path from $u$ to $v$ of length $d(u, v)$.

There may be multiple SPs between $u$ and $v$.

$\begin{array}{*{20}c} 
1 \\
2 \\
3 \\
4 \\
5 \\
\end{array}$

$p = (2, 5, 1, 3, 4)$ is a path from 2 to 4 of length 4.

$(2, 5, 3, 4)$ is a SP from 2 to 4 of length 3, so $d(2, 4) = 3$

$(2, 1, 3, 4)$ is also a SP from 2 to 4
**Shortest Paths**

*Length* of a path \( p = (w_1, \ldots, w_\ell) \) is \( \ell - 1 \) (number of *edges* on the path)

The *distance* \( d(u, v) \) from \( u \) to \( v \) is the *minimum length* of any path from \( u \) to \( v \)

A *shortest path* (SP) from \( u \) to \( v \) is a path from \( u \) to \( v \) of length \( d(u, v) \).

There may be multiple SPs between \( u \) and \( v \).

\[
p = (2, 5, 1, 3, 4)
\]

is a path from \( 2 \) to \( 4 \) of length \( 4 \).

\( (2, 5, 3, 4) \) is a SP from \( 2 \) to \( 4 \) of length \( 3 \), so \( d(2, 4) = 3 \)

\( (2, 1, 3, 4) \) is also a SP from \( 2 \) to \( 4 \)
Section outline

- Paths and shortest paths

  Finding shortest paths: Breadth-First Search

  2-Approximation algorithm for the diameter
Finding the Shortest Paths

The *Breadth-First-Search* algorithm computes, for a *source* vertex $v$, the SP distance $d(v, u)$ for every $u \in V$.

Idea:

1. Start from $v$ and *visit* all its neighbors.
   - They have distance 1 from $v$;
2. Now explore all the unvisited neighbors of the neighbors of $v$.
   - They have distance 2 from $v$;
   ...(continue as long as there are no unvisited nodes)

*All nodes* at distance $k$ from $v$ are *visited before any node* at distance $k + 1$ from $v$. 
BFS

**INPUT**: graph $G$, source node $v \in V$

**OUTPUT**: a list $d$ such that $d[u] = d(v, u)$

For each $u \in V$: $T[u] \leftarrow$ false, $d[u] \leftarrow \infty$;

$Q \leftarrow \emptyset$
BFS

INPUT: graph \( G \), source node \( v \in V \)
OUTPUT: a list \( d \) such that \( d[u] = d(v, u) \)
For each \( u \in V \): \( T[u] \leftarrow \text{false}, d[u] \leftarrow \infty \);
\( Q \leftarrow \emptyset \)
\( d[v] \leftarrow 0, T[v] \leftarrow \text{true}, \) Enqueue \( v \) in \( Q \)
BFS

**INPUT:** graph $G$, source node $v \in V$

**OUTPUT:** a list $d$ such that $d[u] = d(v, u)$

For each $u \in V$: $T[u] \leftarrow \text{false}$, $d[u] \leftarrow \infty$;

$Q \leftarrow \emptyset$

$d[v] \leftarrow 0$, $T[v] \leftarrow \text{true}$, Enqueue $v$ in $Q$

While $Q \neq \emptyset$:

$u \leftarrow \text{Dequeue}(Q)$

For each $z \in N(u)$:

If $T[z] = \text{false}$:

$d[z] \leftarrow d[u] + 1$

$T[z] \leftarrow \text{true}$

Enqueue $z$ in $Q$

Return $d$
BFS

**INPUT:** graph $G$, source node $v \in V$

**OUTPUT:** a list $d$ such that $d[u] = d(v, u)$

For each $u \in V$: $T[u] \leftarrow$ false, $d[u] \leftarrow \infty$;

$Q \leftarrow \emptyset$

$d[v] \leftarrow 0$, $T[v] \leftarrow$ true, Enqueue $v$ in $Q$

While $Q \neq \emptyset$:

$u \leftarrow$ Dequeue($Q$)

For each $z \in N(u)$:

If $T[z] =$false:

$d[z] \leftarrow d[u] + 1$

$T[z] \leftarrow$ true

Enqueue $z$ in $Q$

Return $d$

When we dequeue a node $u$, we iterate through $N(u)$ and enqueue the unvisited neighbors.
**BFS**

**INPUT:** graph $G$, source node $v \in V$

**OUTPUT:** a list $d$ such that $d[u] = d(v, u)$

For each $u \in V$: $T[u] \leftarrow \text{false}$, $d[u] \leftarrow \infty$; $Q \leftarrow \emptyset$

$d[v] \leftarrow 0$, $T[v] \leftarrow \text{true}$, Enqueue $v$ in $Q$

While $Q \neq \emptyset$:

1. $u \leftarrow \text{Dequeue}(Q)$
2. For each $z \in N(u)$:
   - If $T[z] = \text{false}$:
     - $d[z] \leftarrow d[u] + 1$
     - $T[z] \leftarrow \text{true}$
     - Enqueue $z$ in $Q$

Return $d$

When we dequeue a node $u$, we iterate through $N(u)$ and enqueue the unvisited neighbors.

Takes time $O(|N(u)|) = O(\deg(u))$

We dequeue each node exactly once. So the running time of BFS is $O(\sum_{u \in V} \deg(u)) = \ldots$
**BFS**

**INPUT:** graph $G$, source node $v \in V$

**OUTPUT:** a list $d$ such that $d[u] = d(v, u)$

For each $u \in V$: $T[u] \leftarrow$ false, $d[u] \leftarrow \infty$;  
$Q \leftarrow \emptyset$  
$d[v] \leftarrow 0$, $T[v] \leftarrow$ true, Enqueue $v$ in $Q$

While $Q \neq \emptyset$:

take out a node $u$  
For each $z \in N(u)$:

If $T[z] =$false:  
$d[z] \leftarrow d[u] + 1$  
$T[z] \leftarrow$ true  
Enqueue $z$ in $Q$

Return $d$

When we dequeue a node $u$, we iterate through $N(u)$ and enqueue the unvisited neighbors.

Takes time $O(|N(u)|) = O(\text{deg}(u))$

We dequeue each node exactly once.

So the running time of BFS is $O(\sum_{u \in V} \text{deg}(u)) = O(|E|)$
Section outline

- Paths and shortest paths
- Finding shortest paths: Breadth-First Search
- 2-Approximation algorithm for the diameter
**Diameter of a graph**

The *diameter of* $G$ is the *longest distance* between a pair of vertices of $V$:

$$\text{diam}(G) = \max \{ d(u, v) : u, v \in V \}$$

Computing $\text{diam}(G)$ requires computing *All-Pair Shortest Paths*

With $n$ BFSs, we can compute $\text{diam}(G)$ in time $O(|V| \cdot |E|)$
Triangle Inequality

Given $d(u, v)$ and $d(v, w)$, what can we say about $d(u, w)$?
Triangle Inequality

Given $d(u, v)$ and $d(v, w)$, what can we say about $d(u, w)$?

The triangle inequality must hold:

$$d(u, w) \leq d(u, v) + d(v, w)$$

(defining property of distances and norms).

$v = 3, u = 2, w = 5,$
Upper bound to the distances

Fix a vertex $v$, and assume to know $d(v, z)$ for every $z \in V$.
From the triangle inequality it holds that for any $u$ and $w$

$$d(u, w) \leq d(u, v) + d(v, w)$$

Can we find an upper bound to the r.h.s.?
Upper bound to the distances

Fix a vertex $v$, and assume to know $d(v, z)$ for every $z \in V$.

From the triangle inequality it holds that for any $u$ and $w$

$$d(u, w) \leq d(u, v) + d(v, w)$$

Can we find an *upper bound* to the r.h.s.?

There must be a node $q$ such that

$$d(v, q) = \max\{d(v, z), z \in V\}$$

but then, $d(u, v) \leq d(v, q)$, and $d(v, w) \leq d(v, q)$. 
Upper bound to the distances

Fix a vertex $v$, and assume to know $d(v, z)$ for every $z \in V$.

From the triangle inequality it holds that for any $u$ and $w$

$$d(u, w) \leq d(u, v) + d(v, w)$$

Can we find an upper bound to the r.h.s.?

There must be a node $q$ such that

$$d(v, q) = \max\{d(v, z), z \in V\}$$

but then, $d(u, v) \leq d(v, q)$, and $d(v, w) \leq d(v, q)$. Hence,

$$d(u, w) \leq 2 \max\{d(v, z), z \in V\}$$
\(d(u, w) \leq 2 \max\{d(v, z), z \in V\}\)

\[v = 3, \ u = 2, \ w = 5,\]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>
Upper bound to the diameter

For any $v, u,$ and $w$, it holds $d(u, w) \leq 2 \max\{d(v, z), z \in V\}$
Upper bound to the diameter

For any \( v, u, \) and \( w, \) it holds \( d(u, w) \leq 2 \max\{d(v, z), z \in V\} \)

True for any \( u, w, \) thus for any \( u^*, w^* \) such that \( d(u^*, w^*) = \text{diam}(G) \). So,

\[
\text{diam}(G) \leq 2 \max\{d(v, z), z \in V\}
\]

\[
\begin{array}{|c|c|c|}
\hline
u & v & d(u, v) \\
\hline
1 & 2 & 1 \\
1 & 3 & 1 \\
1 & 4 & 2 \\
1 & 5 & 1 \\
2 & 3 & 2 \\
2 & 4 & 3 \\
2 & 5 & 1 \\
3 & 4 & 1 \\
3 & 5 & 1 \\
4 & 5 & 2 \\
\hline
\end{array}
\]

\( \quad v = 3 \)
Approximation algorithm for the diameter

**Input:** A graph $G = (V, E)$, \hspace{1cm} **Output:** a 2-approximation of $\text{diam}(G)$.

1. Choose any vertex $v \in V$
2. Run BFS from $v$, to obtain $\{d(v, z), z \in V\}$
3. Return $2 \max\{d(v, z), z \in V\}$
Paths and shortest paths

Finding shortest paths: Breadth-First Search

2-Approximation algorithm for the diameter
Paths, shortest paths, diameter, and Breadth-First Search

Social networks properties

Centrality and prestige

Closeness centrality

Betweenness centrality

Finding communities with betweenness centrality
Section outline

Homophily, triadic closure, and clustering coefficient

Preferential attachment and its consequences
Social networks

Facebook, Twitter, Instagram, phone calls, collaborations, scientific citations …

*Actors*: entities participating in the network, represented as nodes

They have *features* that characterizes them

*Relationships* between entities, represented as edges

Could be *directed or undirected*

The graph is “*not completely random*”, especially at a local level (i.e., at small scale).
Homophily

A first example of *local non-randomness*

Vertices that are connected to each other, are *likely* to have *similar features*

That’s what the edge represents:

there is some connection, similarity, shared aspect of life between the vertices

A property of *features* that is *inferred* from the *graph structure*

A property of the structure that is *caused* by a property of features
Triadic closure

*Triadic closure:* If two actors $u, v$ have a *common friend* $w$, then they are *more likely to be connected* than two actors that have no common friends.

If $u, v$ are not connected now, they are *more likely to become connected* in the future.
Triadic closure: If two actors $u, v$ have a common friend $w$, then they are more likely to be connected than two actors that have no common friends.

If $u, v$ are not connected now, they are more likely to become connected in the future.

Consequence of homophily: $u$ and $w$ (may) have similar features, and so so for $v$ and $w$, thus $v$ and $u$ may have similar features.

Inherently a property of the structure
Clustering coefficient

A measure of *how strong triadic closure is* for the neighbors of a node $v$

Cluster coefficient of $v$: *How often* are two of $v$’s friends also *friends with each other*?
Clustering coefficient

A measure of *how strong triadic closure is* for the neighbors of a node \( v \)

Cluster coefficient of \( v \): *How often* are two of \( v \)’s friends also *friends with each other*?

Neighbors of \( v \): \( N(v) = \{(u, v) \in E\} \), Cluster coefficient of \( v \):

\[
\eta(v) = \frac{|\{(u, w) \in E : u, w \in N(v)\}|}{(|N(v)|^2)} \in [0, 1]
\]
Clustering coefficient

\[ \eta(v) = \frac{|\{(u, w) \in E : u, w \in N(v)\}|}{\binom{n_v}{2}} \]

\[ v = 3, \quad N(v) = \{4, 5, 1\}, \quad \eta(v) = \frac{1}{3} \]

\[ v = 5, \quad N(3) = \{3, 1, 2\}, \quad \eta(3) = \frac{2}{3} \]
What the clustering coefficient tells us

A community $A$ is a subset of $V$ such that the density

$$\text{density}(A) = \frac{\left| \{(u, v) \in E : u, v \in A\} \right|}{\binom{|A|}{2}}$$

is high.

Vertices in a community are very likely to be connected to each other.

The more communities $v$ belongs to, the lower $\eta(v)$ is likely to be.
Homophily, triadic closure, and clustering coefficient

 Preferential attachment and its consequences
Modeling network formation

Social networks and many other graphs (e.g., the Web) _grows over time_

Most of the time, new edges are created. From time to time, nodes are added. Rarely, edges and vertices are deleted.

We can _model the evolution_ with a _random process_ of network formation.

The properties of a network are _determined by the model_ we choose.

“All models are wrong, but some are useful” — G. Box (attrib.)

Useful models _create networks_ with properties _similar to real networks_.

26
Preferential attachment

A useful model of network formation, although not perfect

There are better ones, but too complex for us

Parameters: new-vertex probability \( a \in (0, 1) \), preference parameter \( g > 0 \)

At time \( t \):

1. With probability \( a \), add a new unconnected vertex;
2. Add an edge \((u, v)\), choosing \( u \) and \( v \) independently, with probability proportional to their current degree

\[ \pi_t(w) \propto \text{deg}^g(w) = |N(v)|^g \]

Different values of \( g \) create different networks.

For social networks, usually \( g \approx 1 \).
Consequences of preferential attachment

Power-law degree distribution

Friendship paradox

Giant connected component

Densification

Small-world property

Shrinking diameter
Power-law degree distribution

*Few high-degree nodes* attract most of the newly-added edges:

The richer gets richer

The fraction $P(k)$ of nodes with degree $k$ has a *power-law* distribution:

$$P(k) \propto k^{-\gamma}, \quad \text{with } 2 < \gamma < 3$$

Values of $\gamma$ closer to 2 have a less skewed distribution
Friendship paradox

Most people have \textit{fewer} friends than their friends have, \textit{on average}.

Average number of friends:
Friendship paradox

Most people have fewer friends than their friends have, on average.

Average number of friends: \[
\frac{1}{|V|} \sum_{v \in V} \deg(v) = \]

Let's show that \(\mu \leq \frac{1}{2} |E| \sum_{v \in V} \deg(v)\) =
Friendship paradox

Most people have \textit{fewer} friends than their friends have, \textit{on average}.

Average number of friends: \[
\frac{1}{|V|} \sum_{v \in V} \text{deg}(v) = \frac{2|E|}{|V|} = \mu
\]
Friendship paradox

Most people have *fewer* friends than their friends have, *on average*.

Average number of friends: 

$$\frac{1}{|V|} \sum_{v \in V} \deg(v) = \frac{2|E|}{|V|} = \mu$$

Number of friends of an average friend: How to compute it?

$$\frac{1}{2|E|} \sum_{(u,z) \in E} (\deg(z) + \deg(u))$$
Friendship paradox

Most people have *fewer* friends than their friends have, *on average*.

Average number of friends: \[ \frac{1}{|V|} \sum_{v \in V} \deg(v) = \frac{2|E|}{|V|} = \mu \] Number of friends of an average friend: How to compute it?

\[ \frac{1}{2|E|} \sum_{(u,z) \in E} (\deg(z) + \deg(u)) = \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 \]
Friendship paradox

Most people have \textit{fewer} friends than their friends have, \textit{on average}.

Average number of friends: \( \frac{1}{|V|} \sum_{v \in V} \deg(v) = \frac{2|E|}{|V|} = \mu \)

Number of friends of an average friend: How to compute it?

\[
\frac{1}{2|E|} \sum_{(u,z) \in E} \left( \deg(z) + \deg(u) \right) = \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2
\]

Let’s show that

\[
\mu \leq \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2
\]
\[
\frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 = \frac{1}{\mu} \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 \quad \left(\text{recall } \mu = \frac{2|E|}{|V|}\right)
\]
\[
\frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 = \frac{1}{\mu} \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 \quad \text{(recall } \mu = \frac{2|E|}{|V|})
\]

\[
= \frac{1}{\mu} \left( \sum_{v \in V} \deg(v)^2 + \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]
\[
\frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 = \frac{1}{\mu} \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 \quad \left( \text{recall } \mu = \frac{2|E|}{|V|} \right)
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \deg(v)^2 + \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \left( \deg(v)^2 + \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]
\[
\frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 = \frac{1}{\mu} \cdot \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 \quad \text{\(\text{recall } \mu = \frac{2|E|}{|V|}\)}
\]

\[
= \frac{1}{\mu} \cdot \frac{1}{|V|} \left( \sum_{v \in V} \deg(v)^2 + \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu} \cdot \frac{1}{|V|} \left( \sum_{v \in V} \left( \deg(v)^2 + \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu} \cdot \frac{1}{|V|} \left( \sum_{v \in V} \left( \deg(v) - \frac{2|E|}{|V|} \right)^2 - \frac{4|E|^2}{|V|} + \frac{4|E|}{|V|} \sum_{v \in V} \deg(v) \right)
\]
\[
\frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 = \frac{1}{\mu} \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 \quad \text{(recall } \mu = \frac{2|E|}{|V|})
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \deg(v)^2 + \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \left( \deg(v)^2 + \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \left( \deg(v) - \frac{2|E|}{|V|} \right)^2 - \frac{4|E|^2}{|V|} + \frac{4|E|}{|V|} \sum_{v \in V} \deg(v) \right)
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} (\deg(v) - \mu)^2 - \frac{4|E|^2}{|V|} + \frac{8|E|^2}{|V|} \right)
\]
\[
\frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 = \frac{1}{\mu} \frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 \quad \text{(recall } \mu = \frac{2|E|}{|V|})
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \deg(v)^2 + \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \left( \deg(v)^2 + \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \left( \deg(v) - \frac{2|E|}{|V|} \right)^2 - \frac{4|E|^2}{|V|} + \frac{4|E|}{|V|} \sum_{v \in V} \deg(v) \right)
\]

\[
= \frac{1}{\mu} \frac{1}{|V|} \left( \sum_{v \in V} \left( \deg(v) - \mu \right)^2 - \frac{4|E|^2}{|V|} + \frac{8|E|^2}{|V|} \right) = \frac{1}{\mu} \left( \frac{1}{|V|} \sum_{v \in V} (\deg(v) - \mu)^2 + \frac{1}{|V|} \frac{4|E|^2}{|V|} \right)
\]
\[
\frac{1}{2|E|} \sum_{v \in V} \deg(v)^2 = \frac{1}{2|E|} \mu \sum_{v \in V} \deg(v)^2 \quad \text{(recall } \mu = \frac{2|E|}{|V|})
\]

\[
= \frac{1}{\mu |V|} \left( \sum_{v \in V} \deg(v)^2 + \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu |V|} \left( \sum_{v \in V} \left( \deg(v)^2 + \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) - \sum_{v \in V} \left( \frac{4|E|^2}{|V|^2} - \frac{4|E|}{|V|} \deg(v) \right) \right)
\]

\[
= \frac{1}{\mu |V|} \left( \sum_{v \in V} \left( \deg(v) - \frac{2|E|}{|V|} \right)^2 - \frac{4|E|^2}{|V|} + \frac{4|E|}{|V|} \sum_{v \in V} \deg(v) \right)
\]

\[
= \frac{1}{\mu |V|} \left( \sum_{v \in V} \left( \deg(v) - \mu \right)^2 - \frac{4|E|^2}{|V|} + \frac{8|E|^2}{|V|} \right) = \frac{1}{\mu} \left( \frac{1}{|V|} \sum_{v \in V} (\deg(v) - \mu)^2 + \frac{1}{|V|} \frac{4|E|^2}{|V|} \right)
\]

\[
= \frac{1}{\mu} (\sigma^2 + \mu^2) = \frac{\sigma^2}{\mu} + \mu \geq \mu
\]
Giant connected component

There is a *path from most nodes to most nodes*

Why?
Giant connected component

There is a *path from most nodes to most nodes*

Why?

1. Most high-degree nodes are connected to each other;
2. Most other nodes are connected to at least one high-degree node.
Densification

The graph density

\[ \frac{|E|}{\binom{|V|}{2}} \]

grows over time.

Why?
Densification

The graph density

\[ \frac{|E|}{\binom{|V|}{2}} \]

grows over time.

Why? We are *adding more edges than nodes*.

If \( n(t) \) is the number of nodes at time \( t \), the number of edges is

\[ e(t) \propto n(t)^\beta \quad \text{with } 1 \leq \beta \leq 2 \]

\( \beta = 1 \Rightarrow \) average degree does not change over time
\( \beta = 2 \Rightarrow \) the network does not become denser
Small-world Property

Small-world: the *average SP distance* is small.

Milgram’s experiment: $\leq 6$ *degrees of separation* between most individuals in the U.S.
Small-world Property

Small-world: the *average SP distance* is small.

Milgram’s experiment: \( \leq 6 \) *degrees of separation* between most individuals in the U.S. (the experiment failed)
Small-world Property

Small-world: the \textit{average SP distance} is small.

Milgram’s experiment: \( \leq 6 \ \text{degrees of separation} \) between most individuals in the U.S. (the experiment failed) Why do we see this effect?
Small-world Property

Small-world: the *average SP distance* is small.

Milgram’s experiment: $\leq 6$ *degrees of separation* between most individuals in the U.S. (the experiment failed) Why do we see this effect?

High-degree nodes allow to *move very fast* across the network.
Many models predict the average SP distance to grow as $O(\log n)$.

In practice, the *average SP distance shrinks*

Why?
Many models predict the average SP distance to grow as $O(\log n)$. 

In practice, the average SP distance shrinks

Why? We are adding more edges than nodes, each edge shortens the distance between at least some pairs of nodes

This behavior was unexpected when it was observed (before the preferential attachment model was developed)
Homophily, triadic closure, and clustering coefficient

Preferential attachment and its consequences
Outline

✔ Paths, shortest paths, diameter, and Breadth-First Search

✔ Social networks properties

Centrality and prestige

Closeness centrality

Betweenness centrality

Finding communities with betweenness centrality
Degree centrality and prestige

Closeness centrality: definition and computation

Approximating closeness centrality
Analysis of social networks focuses on finding/studying *important actors*, a.k.a. *central nodes*

Need a *formal concept of importance*

It may vary depending on the applications or the focus of the study
Centrality measures

Analysis of social networks focuses on finding/studying *important actors*, a.k.a. *central nodes*

Need a *formal concept of importance*

It may vary depending on the applications or the focus of the study

*Centrality measure* (or centrality score):

\[ f : V \rightarrow \mathbb{R}^+ \]

The higher is \( f(v) \), the more central is \( v \).
Degree centrality and prestige

*Degree centrality:* the *degree of* \(v\), i.e., the number of neighbors of \(v\):

\[
f_{\text{deg}}(v) = |N(v)| = |\{(v, u) \in E\}|
\]

It is a natural choice, given the preferential attachment model.
Degree centrality and prestige

**Degree centrality**: the degree of \( v \), i.e., the number of neighbors of \( v \):

\[
f_{\text{deg}}(v) = |N(v)| = |\{(v, u) \in E\}|
\]

It is a natural choice, given the preferential attachment model.

**Prestige**: how many followers do you have?

Solves shortcomings of degree on *directed graphs*:

only counts *incoming edges*:

\[
f_{\text{pres}}(v) = |\{(u, v) \in E\}|_{\text{in-degree of } v}
\]
Shortcomings

Degree centrality and prestige are *myopic* measures:

They only take into account the *immediate neighborhood*, not the *whole structure*.

Figure from C. C. Aggarwal, *Data Mining — The Textbook*
Section outline

- Degree centrality and prestige
- Closeness centrality: definition and computation
- Approximating closeness centrality
Closeness centrality

What’s our intuition of the word \textit{central}?
Closeness centrality

What’s our intuition of the word *central*?
A node is central when the *average distance* to any other node else is small

$$\text{avgd}(v) = \frac{1}{n} \sum_{u \in V} d(v, u)$$

Is this a centrality measure?
Closeness centrality

What’s our intuition of the word *central*?
A node is central when the *average distance* to any other node else is small

\[
\text{avgd}(v) = \frac{1}{n} \sum_{u \in V} d(v, u)
\]

Is this a centrality measure?
No, because it is *high* when the node is *not central*, and low otherwise.
Closeness centrality

What’s our intuition of the word *central*?
A node is central when the *average distance* to any other node else is small

$$\text{avgd}(v) = \frac{1}{n} \sum_{u \in V} d(v, u)$$

Is this a centrality measure?
No, because it is *high* when the node is *not central*, and low otherwise.

*Closeness centrality:*

$$cc(v) = \frac{1}{\text{avgd}(v)}$$

Only for *undirected connected* graphs
Computing closeness centrality

\[ cc(v) = \frac{1}{\sum_{u \in V} d(v, u)} \]

How to get \( cc(v) \) for a single node \( (v) \)?
Computing closeness centrality

\[ cc(v) = \frac{1}{\sum_{u \in V} d(v, u)} \]

How to get \( cc(v) \) for a single node \((v)\)?

Run BFS from \( v \) to get all the \( d(v, u), u \in V \), then average and take inverse. Takes time \( O(|E|) \).
Computing closeness centrality

\[ cc(v) = \frac{1}{\sum_{u \in V} d(v, u)} \]

How to get \( cc(v) \) for a single node \( (v) \)?

Run BFS from \( v \) to get all the \( d(v, u), u \in V \), then average and take inverse.

Takes time \( O(|E|) \).

How to compute \( cc(v) \) for all nodes \( v \in V \)?

“Efficient” has a different meaning when working with large data: more than quadratic is too slow. (Sometimes even quadratic.)
Computing closeness centrality

$$cc(v) = \frac{1}{\sum_{u \in V} d(v, u)}$$

How to get $cc(v)$ for a single node $(v)$?

Run BFS from $v$ to get all the $d(v, u)$, $u \in V$, then average and take inverse.

Takes time $O(|E|)$.

How to compute $cc(v)$ for all nodes $v \in V$?

There’s no **efficient** algorithm: must run BFS from each $v$.

Takes time $O(|V||E|)$.

“Efficient” has a different meaning when working with large data: **more than quadratic is too slow**. (Sometimes even quadratic)
Section outline

- Degree centrality and prestige
- Closeness centrality: definition and computation
- Approximating closeness centrality
Approximating closeness centrality

Why computing approximations of closeness centrality?
Approximating closeness centrality

Why computing approximations of closeness centrality?

1. The exact algorithm is too expensive;
2. The network is dynamic: the exact values keep changing, so there's no point in chasing them.
3. No application requires the exact ranking of nodes w.r.t. closeness centrality. Big-data tenet: Don't do more work than necessary for the application.
Why computing approximations of closeness centrality?

1. The exact algorithm is too expensive;

2. The network is *dynamic*: the exact values keep changing, so there’s no point in chasing them.

3. No application requires the *exact* ranking of nodes w.r.t. closeness

Big-data tenet: Don’t do more work than *necessary for the application*. 
Approximating closeness centrality

Exact algorithm to compute the CC of all vertices:

For each $v \in V$
  - Run BFS from $v$ to obtain $d(v, u)$ for every $u \in V$

Return $cc(v) = 1 / \sum_{u \in V} d(v, u)$
Approximating closeness centrality

Exact algorithm to compute the CC of all vertices:

For each \( v \in V \):
   - Run BFS from \( v \) to obtain \( d(v, u) \) for every \( u \in V \)
   - Return \( cc(v) = 1/\sum_{u \in V} d(v, u) \)

The distance \( d(v, u) \) computed when running BFS from \( v \) also contributes to \( cc(u) \).

After \( k \) iterations, we have \( k \) terms of \( \sum_{z \in V} d(u, z) \) for each \( u \) that was not a source yet.
Approximating closeness centrality

Exact algorithm to compute the CC of all vertices:

For each $v \in V$:
   Run BFS from $v$ to obtain $d(v, u)$ for every $u \in V$
   Return $cc(v) = 1/\sum_{u \in V} d(v, u)$

The distance $d(v, u)$ computed when running BFS from $v$ also contributes to $cc(u)$.

After $k$ iterations, we have $k$ terms of $\sum_{z \in V} d(u, z)$ for each $u$ that was not a source yet.

The average of these $k$ terms is an estimate for $avgd(u)$. 

Approximating closeness centrality

**INPUT**: Undirected connected graph $G$, *sample size* $k$

For each $v \in V$: $S[v] \leftarrow 0$

For each $i \leftarrow 1$ to $k$:

- Select a node $v$ uniformly at random
- Run BFS from $v$ to obtain $d(v, u)$ for every $u \in V$

For each $u \in V$: $S[u] \leftarrow S[u] + d(v, u)$

Return $(k/S[u])_{u \in V}$ // Estimations of $cc(u) = 1/\text{avgd}(u), u \in V$
Approximating closeness centrality

**INPUT:** Undirected connected graph $G$, sample size $k$

For each $v \in V$: $S[v] \leftarrow 0$

For each $i \leftarrow 1$ to $k$:

- Select a node $v$ uniformly at random
- Run BFS from $v$ to obtain $d(v, u)$ for every $u \in V$
  - For each $u \in V$: $S[u] \leftarrow S[u] + d(v, u)$ // Can be done while running BFS from $v$

Return $(k/S[u])_{u \in V}$ // Estimations of $cc(u) = 1/avgd(u)$, $u \in V$

The *pseudocode is a lie*: think when you implement!
Approximating closeness centrality

**Input:** Undirected connected graph $G$, *sample size* $k$

**Output:** How large should $k$ be to get good approximations?
Approximating closeness centrality

**INPUT:** Undirected connected graph $G$, *sample size* $k$

**OUTPUT:** How large should $k$ be to get good approximations?
Approximating closeness centrality

**Input:** Undirected connected graph $G$, *sample size* $k$

**Output:** How large should $k$ be to get good approximations?
Approximating closeness centrality

**Input:** Undirected connected graph $G$, sample size $k$

**Output:** How large should $k$ be to get good approximations? Depends on $\text{diam}(G)$, and the desired quality of the approximation, and on $|V|$. 

Section outline

✓ Degree centrality and prestige

✓ Closeness centrality: definition and computation

✓ Approximating closeness centrality
Outline

✓ Paths, shortest paths, diameter, and Breadth-First Search

✓ Social networks properties

✓ Centrality and prestige

✓ Closeness centrality

Betweenness centrality

Finding communities with betweenness centrality
Section outline

Betweenness centrality: definition and properties

Computation

Approximation algorithm

Finding communities with betweenness centrality
Who is central/important?

Node 1 has the highest degree centrality.
Node 3 has the highest closeness centrality.