

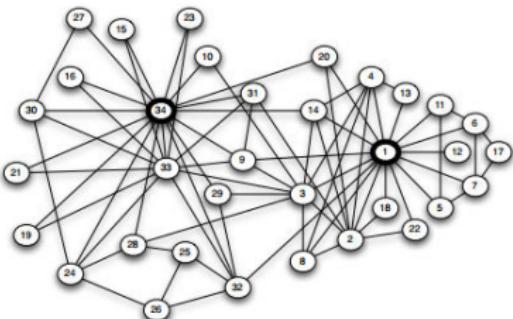
Python for Complex Network Analysis

Sajeev G P

February 17, 2020

Node Importance

Based on the structure of the network, which are the 5 most important node in the Karate Club friendship network?



Friendship network in a 34-person karate club
[Zachary 1977]

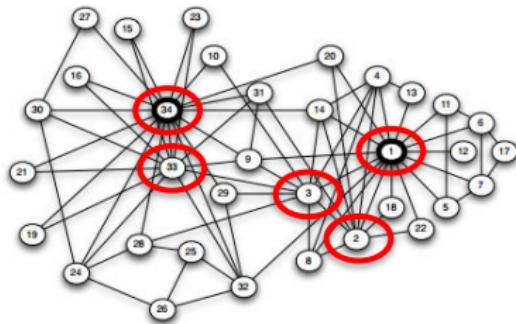
Node Importance

Different ways of thinking about “importance”.

Ex. Degree: number of friends.

5 most important nodes are:

34, 1, 33, 3, 2



Friendship network in a 34-person karate club
[Zachary 1977]

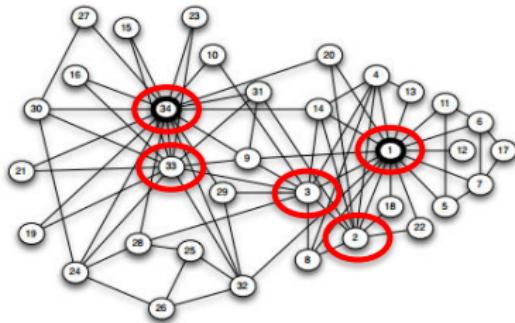
Node Importance

Different ways of thinking about “importance”.

Ex. Average proximity to other nodes.

5 most important nodes are:

1, 3, 34, 32, 9



Friendship network in a 34-person karate club
[Zachary 1977]

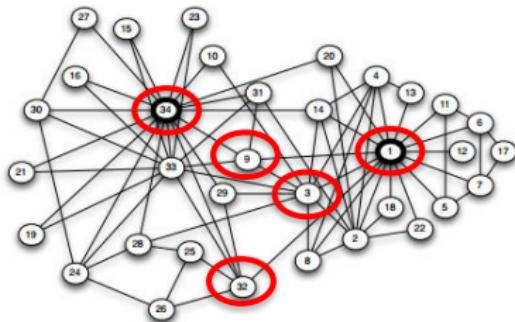
Node Importance

Different ways of thinking about “importance”.

Ex. Fraction of shortest paths that pass through node.

5 most important nodes are:

1, 34, 33, 3, 32



Friendship network in a 34-person karate club
[Zachary 1977]

Network Centrality

Centrality measures identify the most important nodes in a network:

- Influential nodes in a social network.
- Nodes that disseminate information to many nodes or prevent epidemics.
- Hubs in a transportation network.
- Important pages on the Web.
- Nodes that prevent the network from breaking up.

Centrality Measures

- **Degree centrality**
- **Closeness centrality**
- Betweenness centrality
- Load centrality
- Page Rank
- Katz centrality
- Percolation centrality

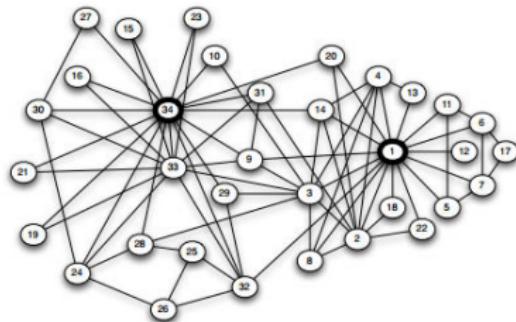
Degree Centrality

Assumption: important nodes have many connections.

The most basic measure of centrality: number of neighbors.

Undirected networks: use degree

Directed networks: use in-degree or out-degree

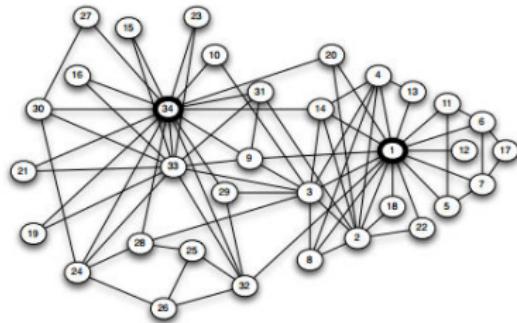


Friendship network in a 34-person karate club
[Zachary 1977]

Degree Centrality – Undirected Networks

$C_{deg}(v) = \frac{d_v}{|N|-1}$, where N is the set of nodes in the network and d_v is the degree of node v .

```
In: G = nx.karate_club_graph()  
In: G =  
nx.convert_node_labels_to_integers(G,first_label=1)  
In: degCent = nx.degree_centrality(G)  
In: degCent[34]  
Out: 0.515 # 17/33  
In: degCent[33]  
Out: 0.182 # 6/33
```



Friendship network in a 34-person karate club
[Zachary 1977]

Degree Centrality – Directed Networks

$$C_{indeg}(v) = \frac{d_v^{in}}{|N|-1}, \text{ where}$$

N = set of nodes in the network,

d_v^{in} = the in-degree of node v .

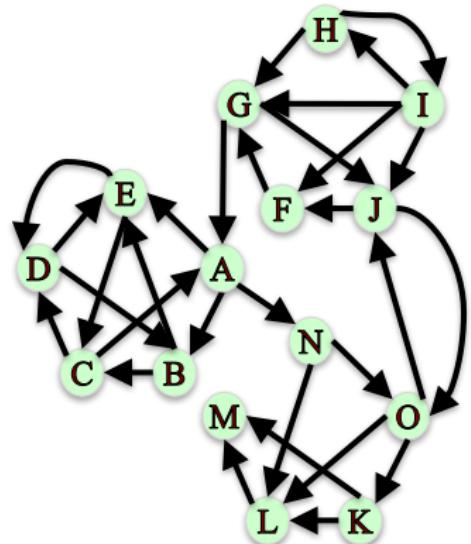
```
In: indegCent = nx.in_degree_centrality(G)
```

```
In: indegCent['A']
```

```
Out: 0.143 # 2/14
```

```
In: indegCent['L']
```

```
Out: 0.214 # 3/14
```



Degree Centrality – Directed Networks

$$C_{outdeg}(v) = \frac{d_v^{out}}{|N|-1}, \text{ where}$$

N = set of nodes in the network,

d_v^{out} = the out-degree of node v .

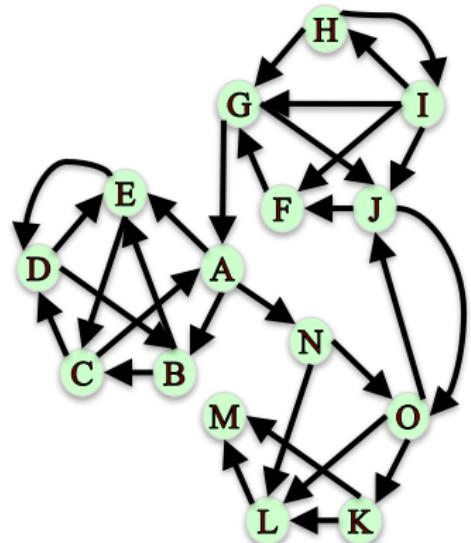
```
In: outdegCent = nx.out_degree_centrality(G)
```

```
In: outdegCent['A']
```

```
Out: 0.214 # 3/14
```

```
In: outdegCent['L']
```

```
Out: 0.071 # 1/14
```



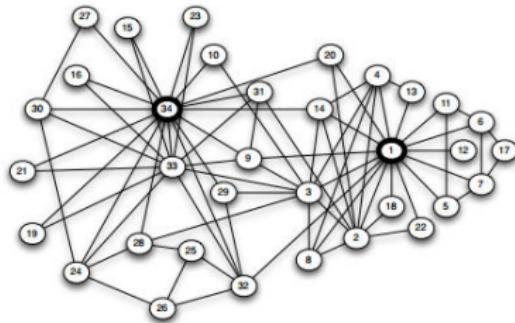
Closeness Centrality

Assumption: important nodes are close to other nodes.

$$C_{close}(v) = \frac{|N|-1}{\sum_{u \in N \setminus \{v\}} d(v,u)}, \text{ where}$$

N = set of nodes in the network,

$d(v,u)$ = length of shortest path from v to u .



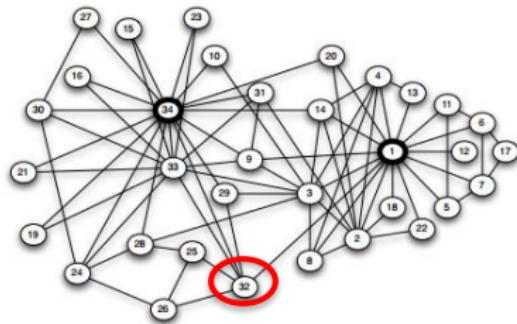
Friendship network in a 34-person karate club
[Zachary 1977]

Closeness Centrality

Assumption: important nodes are close to other nodes.

```
In: closeCent = nx.closeness_centrality(G)  
In: closeCent[32]  
Out: 0.541
```

```
In: sum(nx.shortest_path_length(G,32).values())  
Out: 61  
In: (len(G.nodes())-1)/61.  
Out: 0.541
```



Friendship network in a 34-person karate club
[Zachary 1977]

Disconnected Nodes

How to measure the closeness centrality of a node when it cannot reach all other nodes?

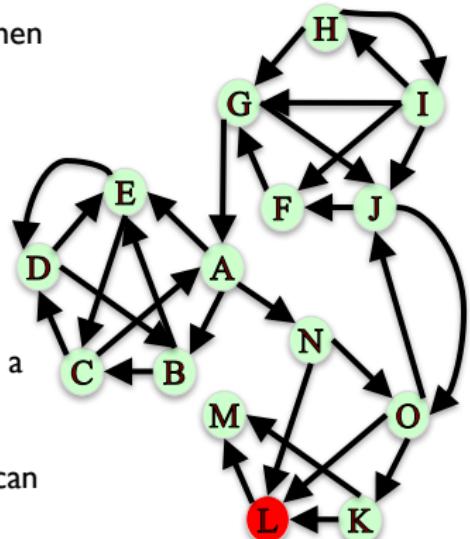
What is the closeness centrality of node L?

Option 1: Consider only nodes that L can reach:

$$C_{close}(L) = \frac{|R(L)|}{\sum_{u \in R(L)} d(L,u)}, \text{ where } R(L) \text{ is the set of nodes L can reach.}$$

$C_{close}(L) = \frac{1}{1} = 1$, since L can only reach M and it has a shortest path of length 1.

Problem: centrality of I is too high for a node than can only reach one other node!



Disconnected Nodes

How to measure the closeness centrality of a node when it cannot reach all other nodes?

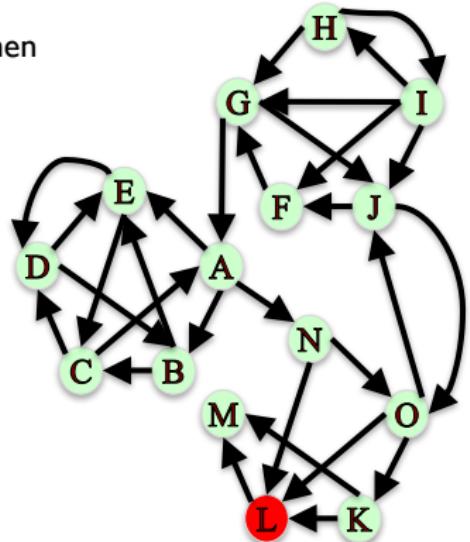
What is the closeness centrality of node L?

Option 2: Consider only nodes that L can reach and normalize by the fraction of nodes L can reach:

$$C_{close}(L) = \left[\frac{|R(L)|}{|N-1|} \right] \frac{|R(L)|}{\sum_{u \in R(L)} d(L,u)}$$

$$C_{close}(L) = \left[\frac{1}{14} \right] \frac{1}{1} = 0.071$$

Note that this definition matches our definition of closeness centrality when a graph is connected since $R(L) = N - 1$



Disconnected Nodes

How to measure the closeness centrality of a node when it cannot reach all other nodes?

What is the closeness centrality of node L?

```
In: closeCent = nx.closeness_centrality(G, normalized =  
False)
```

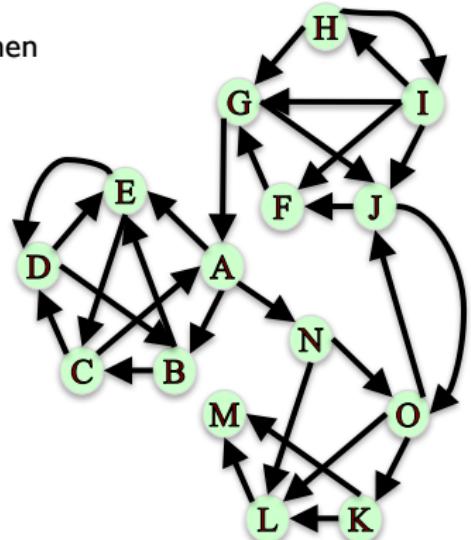
```
In: closeCent['L']
```

```
Out: 1
```

```
In: closeCent = nx.closeness_centrality(G, normalized =  
True)
```

```
In: closeCent['L']
```

```
Out: 0.071
```



Summary

Centrality measures identify the most important nodes in a network:

Degree Centrality

Assumption: important nodes have many connections.

$$C_{deg}(v) = \frac{d_v}{|N| - 1}$$

```
nx.degree_centrality(G)  
nx.in_degree_centrality(G)  
nx.out_degree_centrality(G)
```

Closeness Centrality

Assumption: important nodes are close to other nodes.

$$C_{close}(L) = \left[\frac{|R(L)|}{|N-1|} \right] \frac{|R(L)|}{\sum_{u \in R(L)} d(L,u)}$$

```
nx.closeness_centrality(G, normalized =  
True)
```

Betweenness Centrality

Assumption: important nodes connect other nodes.

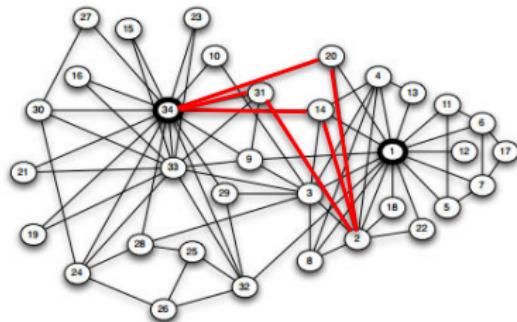
Recall: the distance between two nodes is the length of the shortest path between them.

Ex. The distance between nodes 34 and 2 is 2:

Path 1: 34 – 31 – 2

Path 2: 34 – 14 – 2

Path 3: 34 – 20 – 2



Friendship network in a 34-person karate club
[Zachary 1977]

Nodes 31, 14, and 20 are in a shortest path of between nodes 34 and 2.

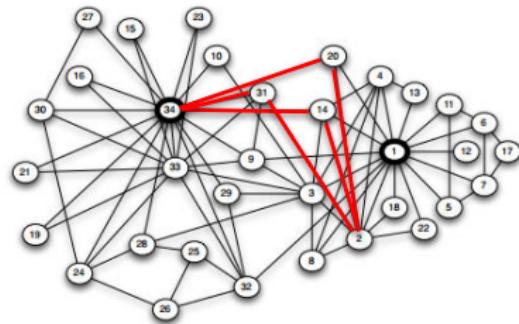
Betweenness Centrality

Assumption: important nodes connect other nodes.

$$C_{btw}(v) = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}, \text{ where}$$

$\sigma_{s,t}$ = the number of shortest paths between nodes s and t .

$\sigma_{s,t}(v)$ = the number shortest paths between nodes s and t that pass through node v .



Friendship network in a 34-person karate club
[Zachary 1977]

Betweenness Centrality

Assumption: important nodes connect other nodes.

$$C_{btw}(v) = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

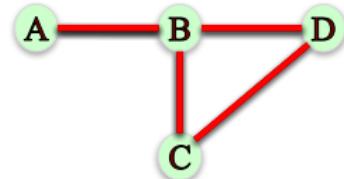
Endpoints: we can either include or exclude node v as node s and t in the computation of $C_{btw}(v)$.

Ex. If we exclude node v , we have:

$$C_{btw}(B) = \frac{\sigma_{A,D}(B)}{\sigma_{A,D}} + \frac{\sigma_{A,C}(B)}{\sigma_{A,C}} + \frac{\sigma_{C,D}(B)}{\sigma_{C,D}} = \frac{1}{1} + \frac{1}{1} + \frac{0}{1} = 2$$

If we include node v , we have:

$$C_{btw}(B) = \frac{\sigma_{A,B}(B)}{\sigma_{A,B}} + \frac{\sigma_{A,C}(B)}{\sigma_{A,C}} + \frac{\sigma_{A,D}(B)}{\sigma_{A,D}} + \frac{\sigma_{B,C}(B)}{\sigma_{B,C}} + \frac{\sigma_{B,D}(B)}{\sigma_{B,D}} + \frac{\sigma_{C,D}(B)}{\sigma_{C,D}} = \frac{1}{1} + \frac{1}{1} + \frac{1}{1} + \frac{1}{1} + \frac{0}{1} = 5$$



Disconnected Nodes

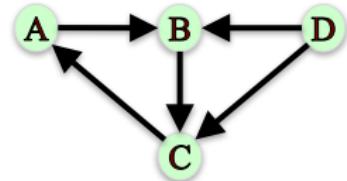
Assumption: important nodes connect other nodes.

$$C_{btw}(v) = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

What if not all nodes can reach each other?

Node D cannot be reached by any other node.

Hence, $\sigma_{A,D} = 0$, making the above definition undefined.



When computing betweenness centrality, we only consider nodes s, t such that there is at least one path between them.

Disconnected Nodes

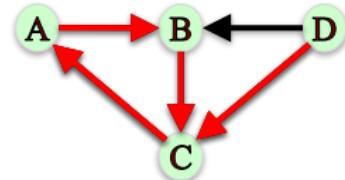
Assumption: important nodes connect other nodes.

$$C_{btw}(v) = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

What if not all nodes can reach each other?

Node D cannot be reached by any other node.

Hence, $\sigma_{A,D} = 0$, making the above definition undefined.



Ex. What is the betweenness centrality of node B, without including it as endpoint?

$$C_{btw}(B) = \frac{\sigma_{A,C}(B)}{\sigma_{A,C}} + \frac{\sigma_{C,A}(B)}{\sigma_{C,A}} + \frac{\sigma_{D,C}(B)}{\sigma_{D,C}} + \frac{\sigma_{D,A}(B)}{\sigma_{D,A}} = \boxed{1} + \boxed{0} + \boxed{0} + \boxed{0} = 1$$

Disconnected Nodes

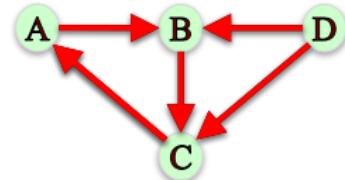
Assumption: important nodes connect other nodes.

$$C_{btw}(v) = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

What if not all nodes can reach each other?

Node D cannot be reached by any other node.

Hence, $\sigma_{A,D} = 0$, making the above definition undefined.



Ex. What is the betweenness centrality of node C, without including it as endpoint?

$$C_{btw}(C) = \frac{\sigma_{A,B}(C)}{\sigma_{A,B}} + \frac{\sigma_{B,A}(C)}{\sigma_{B,A}} + \frac{\sigma_{D,B}(C)}{\sigma_{D,B}} + \frac{\sigma_{D,A}(C)}{\sigma_{D,A}} = \frac{0}{1} + \frac{1}{1} + \frac{0}{1} + \frac{1}{1} = 2$$

Betweenness Centrality – Normalization

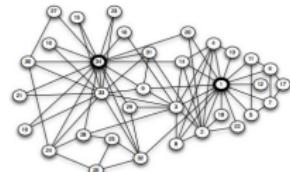
Assumption: important nodes connect other nodes.

$$C_{btw}(v) = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

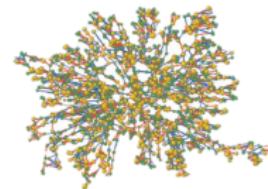
Normalization: betweenness centrality values will be larger in graphs with many nodes. To control for this, we divide centrality values by the number of pairs of nodes in the graph (excluding v):

$\frac{1}{2}(|N| - 1)(|N| - 2)$ in undirected graphs

$(|N| - 1)(|N| - 2)$ in directed graphs



Friendship network in a 34-person karate club
[Zachary 1977]



Network of friendship, marital tie, and family tie among 2200 people
[Christakis & Fowler 2007]

Betweenness Centrality

```
In: btwnCent = nx.betweenness_centrality(G,  
normalized = True, endpoints = False)
```

```
In: import operator
```

```
In: sorted(btwnCent.items(),  
key=operator.itemgetter(1), reverse = True)[0:5]
```

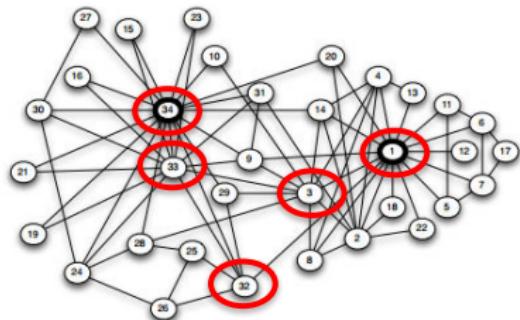
```
Out: [(1, 0.43763528138528146),
```

```
(34, 0.30407497594997596),
```

```
(33, 0.14524711399711399),
```

```
(3, 0.14365680615680618),
```

```
(32, 0.13827561327561325)]
```

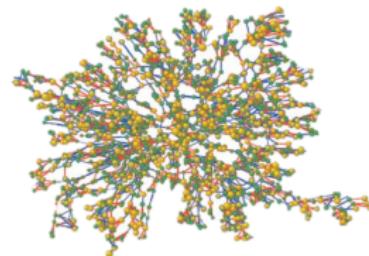


Friendship network in a 34-person karate club
[Zachary 1977]

Betweenness Centrality – Complexity

Computing betweenness centrality of all nodes can be very computationally expensive.

Depending on the algorithm, this computation can take up to $O(|N|^3)$ time.



Network of friendship, marital tie, and family tie among 2200 people
[Christakis & Fowler 2007]

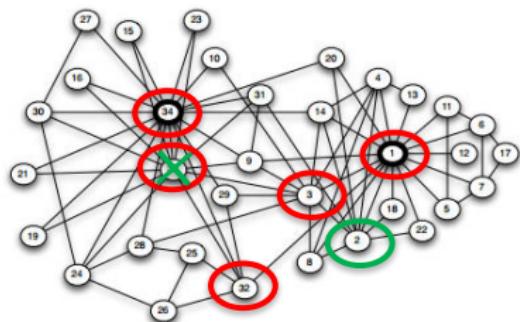
Approximation: rather than computing betweenness centrality based on all pairs of nodes s, t , we can approximate it based on a sample of nodes.

$N = 2200$ nodes →
~4.8 million pairs of nodes

Betweenness Centrality – Approximation

In: `btwnCent_approx = nx.betweenness_centrality(G,
normalized = True, endpoints = False, k = 10)`

In: `sorted(btwnCent_approx.items(),
key=operator.itemgetter(1), reverse = True)[0:5]`
Out: [(1, 0.4826939033189033),
(34, 0.27564694564694564),
(32, 0.20863636363636362),
(3, 0.1697598003848004),
(2, 0.13194624819624817)]



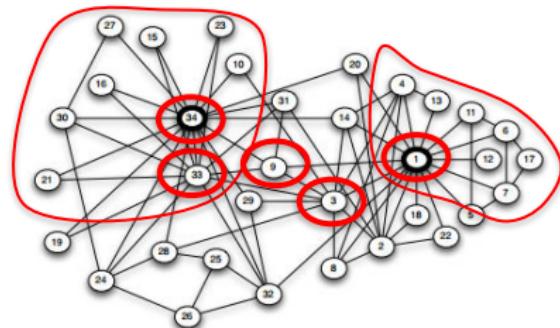
Friendship network in a 34-person karate club
[Zachary 1977]

Betweenness Centrality – Subsets

```
In: btwnCent_subset =  
nx.betweenness_centrality_subset(G, [34, 33, 21, 30,  
16, 27, 15, 23, 10], [1, 4, 13, 11, 6, 12, 17, 7],  
normalized=True)
```

```
In:sorted(btwnCent_subset.items(),key=operator.item  
getter(1), reverse=True)[0:5]
```

```
Out: [(1, 0.04899515993265994),  
(34, 0.028807419432419434),  
(3, 0.018368205868205867),  
(33, 0.01664712602212602),  
(9, 0.014519450456950456)]
```



Friendship network in a 34-person karate club
[Zachary 1977]

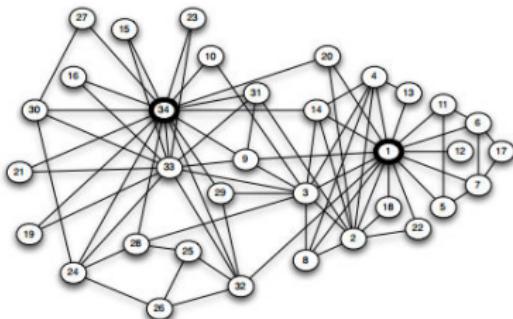
Betweenness Centrality – Edges

We can use betweenness centrality to find important edges instead of nodes:

$$C_{btw}(e) = \sum_{s,t \in N} \frac{\sigma_{s,t}(e)}{\sigma_{s,t}}, \text{ where}$$

$\sigma_{s,t}$ = the number of shortest paths between nodes s and t .

$\sigma_{s,t}(e)$ = the number shortest paths between nodes s and t that pass through edge e .

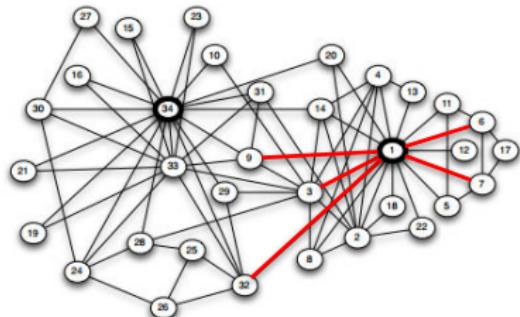


Friendship network in a 34-person karate club
[Zachary 1977]

Betweenness Centrality – Edges

```
In: btwnCent_edge =  
nx.edge_betweenness_centrality(G,  
normalized=True)
```

```
In: sorted(btwnCent_edge.items(),  
key=operator.itemgetter(1), reverse = True)[0:5]  
Out: [((1, 32), 0.12725999490705373),  
((1, 7), 0.07813428401663694),  
((1, 6), 0.07813428401663694),  
((1, 3), 0.0777876807288572),  
((1, 9), 0.07423959482783014)]
```

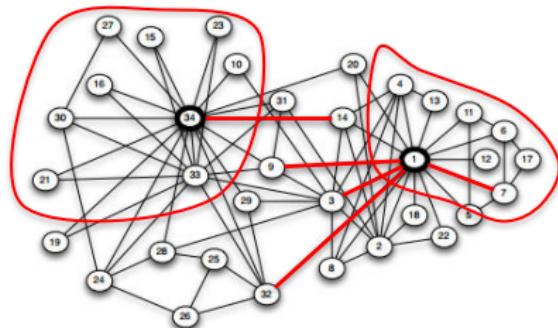


Friendship network in a 34-person karate club
[Zachary 1977]

Betweenness Centrality – Edges

```
In: btwnCent_edge_subset =  
nx.edge_betweenness_centrality_subset(G, [34, 33,  
21, 30, 16, 27, 15, 23, 10], [1, 4, 13, 11, 6, 12, 17,  
7], normalized=True)
```

```
In: sorted(btwnCent_edge_subset.items(),  
key=operator.itemgetter(1), reverse = True)[0:5]  
Out: [((1, 32), 0.01366536513595337),  
((1, 9), 0.01366536513595337),  
((14, 34), 0.012207509266332794),  
((1, 3), 0.01211343123107829),  
((1, 7), 0.012032085561497326)]
```



Friendship network in a 34-person karate club
[Zachary 1977]

Summary

Betweenness centrality assumption: important nodes connect other nodes.

$$C_{btw}(v) = \sum_{s,t \in N} \frac{\sigma_{s,t}(v)}{\sigma_{s,t}}$$

Normalization: Divide by number of pairs of nodes.

Approximation: Computing betweenness centrality can be computationally expensive. We can approximate computation by taking a subset of nodes.

Subsets: We can define subsets of source and target nodes to compute betweenness centrality.

Edge betweenness centrality: We can apply the same framework to find important edges instead of nodes.

Resources

- NetworkX Docs
<http://networkx.lanl.gov/tutorial/index.html>
- NetworkX Tutorial
<http://networkx.lanl.gov/contents.html>
- Matplotlib Docs
<http://matplotlib.sourceforge.net/contents.html>
- Matplotlib Tutorial
http://matplotlib.sourceforge.net/users/pyplot_tutorial.html
- Numpy Docs
<http://numpy.scipy.org/>
- MacPorts
<http://macports.org>