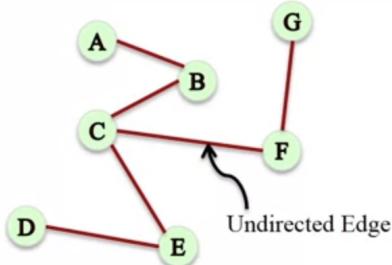


Applied Social Network Analysis in Python

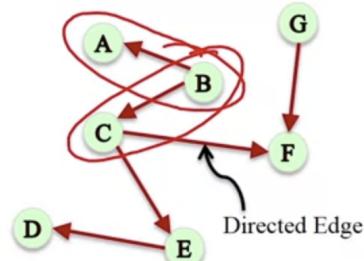
<input checked="" type="checkbox"/> Completed	<input type="checkbox"/>
Tags	Data Science Python

Edge Direction



Undirected network:
edges have no direction

```
G=nx.Graph()  
G.add_edge('A','B')  
G.add_edge('B','C')
```



Directed network:
edges have direction

```
G=nx.DiGraph()  
G.add_edge('B', 'A')  
G.add_edge('B', 'C')
```

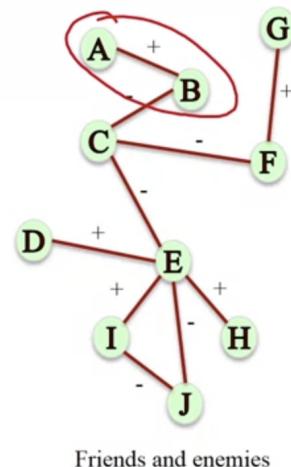
Signed Networks

Some networks can carry information about friendship and antagonism based on conflict or disagreement.

Ex: In Epinions and Slashdot people can declare friends and foes.

Signed network: a network where edges are assigned positive or negative sign.

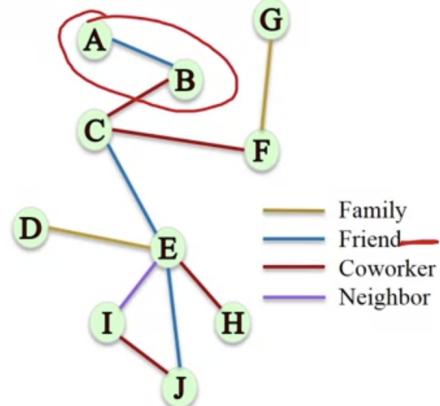
```
G=nx.Graph()  
G.add_edge('A','B', sign= '+')
```



Other Edge Attributes

Edges can carry many other labels or attributes

```
G=nx.Graph()  
G.add_edge('A','B', relation= 'friend')  
G.add_edge('B','C', relation= 'coworker')  
G.add_edge('D','E', relation= 'family')
```



Edge Attributes in NetworkX

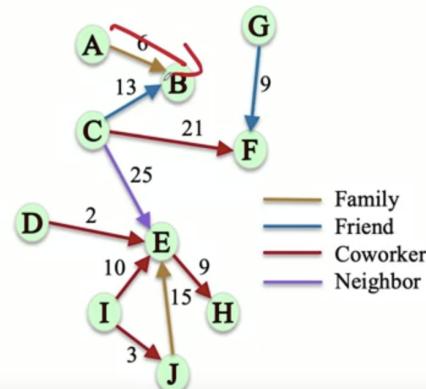
Directed, weighted network:

```
G=nx.DiGraph()  
G.add_edge('A','B', weight= 6, relation = 'family')  
G.add_edge('C', 'B', weight= 13, relation = 'friend')
```

Accessing edge attributes:

In: G.edges['C','B']['weight']
Out: 13

In: G.edges['B','C']['weight'] # directed graph, order matters
Out: KeyError: 'C'



?

Directed graph, callout must be in order in order to access attributes

Output of Multi-Edge graph access

- Array form of storing and accessing
- Likewise order matters if it is directed

Node Attributes

role

Node Attributes in NetworkX

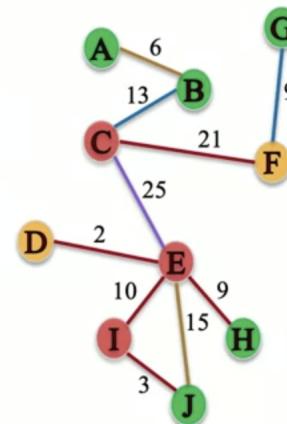
```
G=nx.Graph()  
G.add_edge('A','B', weight= 6, relation = 'family')  
G.add_edge('B','C', weight= 13, relation = 'friend')
```

Family
Friend
Coworker
Neighbor

Accessing node attributes:

```
In: list(G.nodes()) # list of all nodes  
Out: ['A', 'C', 'B']  
In: list(G.nodes(data= True)) #list of nodes w/ attributes  
Out: [('A', {'role': 'trader'}), ('C', {'role': 'manager'}),  
, ('B', {'role': 'trader'})]  
In: G.nodes['C']['role']  
Out: 'manager'
```

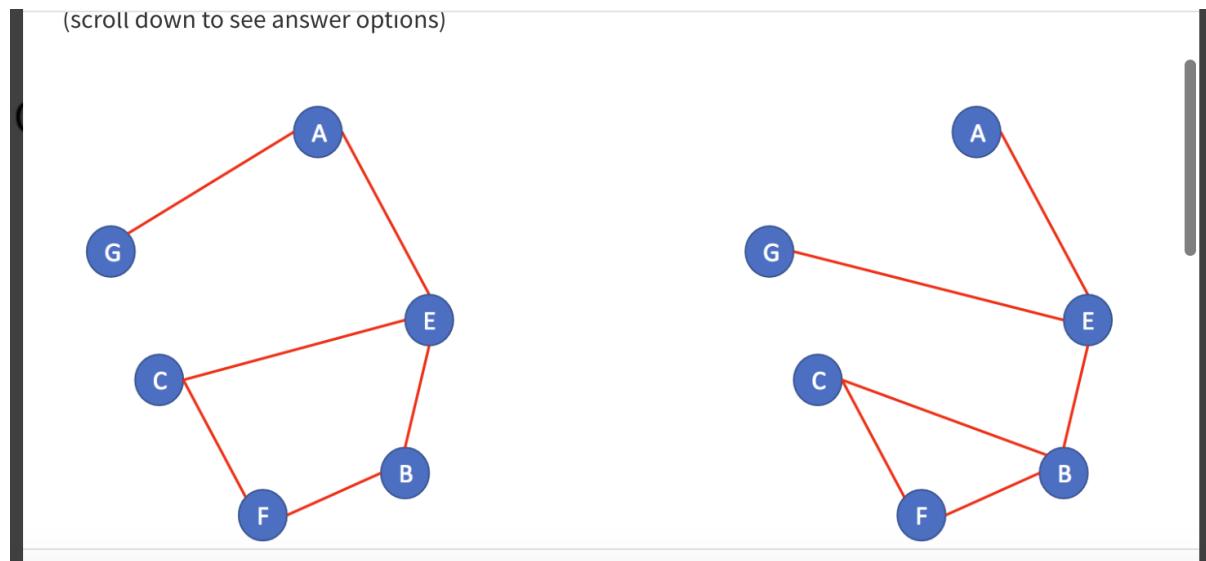
Manager
Trader
Analyst



Number of times coworkers had lunch together in one year

Bipartite Graph

Graph A is bipartite: the two sets of nodes are {A,B,C} and {E,G,F}. All edges connect a node in one set to a node in the other set. Graph B is not bipartite: note that nodes C, B, and F form a triangle, so it is not possible to assign each of these nodes to a side without having edges connecting nodes that were assigned to the same side. In fact, for the same reason, a bipartite graph cannot contain a cycle of an odd number of nodes.



to be bipartite :

within the set cannot have internal edges, which will be against the properties

- Good and Suitable for Recommender System

Projected Graph to show relationship between node

Distance Measures

Summary

Distance between two nodes: length of the shortest path between them.

Eccentricity of a node n is the largest distance between n and all other nodes.

Characterizing distances in a network:

Average distance between every pair of nodes.

Diameter: maximum distance between any pair of nodes.

Radius: the minimum eccentricity in the graph.

Identifying central and peripheral nodes:

The **Periphery** is the set of nodes with eccentricity = diameter.

The **center** is the set of nodes with eccentricity = radius.

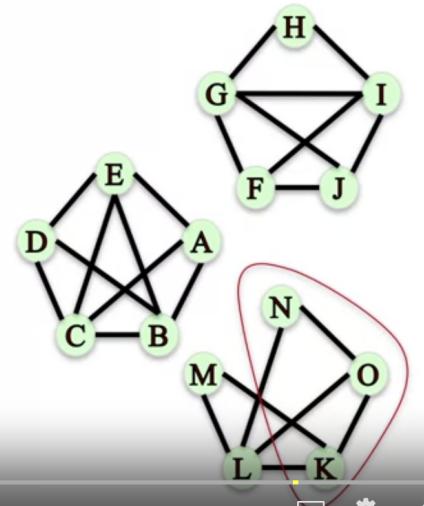
Graph Components

Connected component:

A subset of nodes such as:

- i. Every node in the subset has a path to every other node.
- ii. No other node is has a path to any node in the subset.

Is the subset {N, O, K} a connected component?



Connectivity in Directed Graph

Strongly Connected

directed path from u to v , and v to u

Weakly Connected

all undirected edges , because there will be a path from u to v, without order

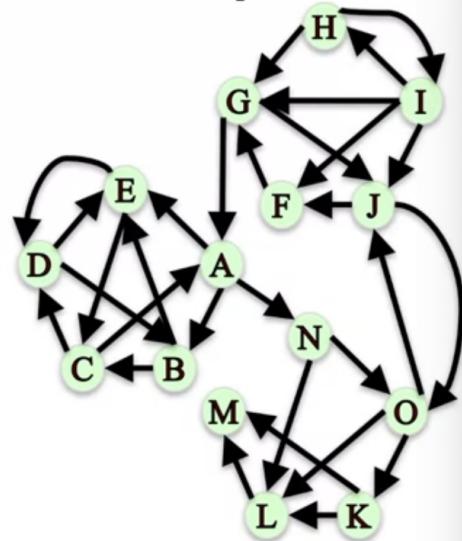
Connectivity in Directed Graphs

Strongly connected component:

A subset of nodes such as:

- . Every node in the subset has a **directed** path to every other node.
- i. No other node has a **directed** path to and from every node in the subset.

What are the strongly connected components in this graph?



Connectivity and Robustness

How well are the nodes connected

If certain edge is removed, can i still access the same node via alternate path

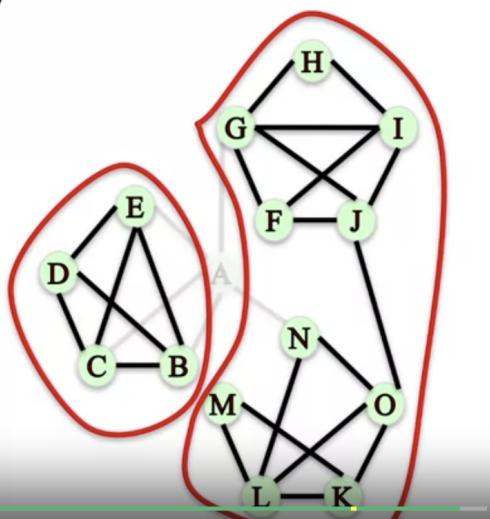
– Item navigation

Summary

Node connectivity: Minimum number of nodes needed to disconnect a graph or pair of nodes.
`nx.node_connectivity(G, 'G', 'L')`
`nx.minimum_node_cut(G, 'G', 'L')`

Edge connectivity: Minimum number of edges needed to disconnect a graph or pair of nodes.
`nx.edge_connectivity(G, 'G', 'L')`
`nx.minimum_edge_cut(G, 'G', 'L')`

Graphs with large node and edge connectivity are more robust to the loss of nodes and edges.



Python Lib

▼ pickle

Python Pickle is used to serialize and deserialize a python object structure.

▼ nx_read_edgelist

https://networkx.org/documentation/stable/reference/readwrite/generated/networkx.readwrite.edgelist.read_edgelist.html
`read_edgelist(path, comments='#', delimiter=None, create_using=None, nodetype=None, data=True, edgetype=None, 8')[source]#`

pathfile or string

File or filename to read. If a file is provided, it must be opened in 'rb' mode. Filenames ending in .gz or .bz2 will be uncompressed.

comments: string, optional

The character used to indicate the start of a comment. To specify that no character should be treated as a comment, use `comments=None`.

delimiter: string, optional

The string used to separate values. The default is whitespace.

create_usingNetworkX graph constructor, optional (default=nx.Graph)

Graph type to create. If graph instance, then cleared before populated.

nodetypeint, float, str, Python type, optional

Convert node data from strings to specified type

databool or list of (label,type) tuples

Tuples specifying dictionary key names and types for edge data

edgetype: int, float, str, Python type, optional OBSOLETE

Convert edge data from strings to specified type and use as 'weight'

encoding: string, optional

Specify which encoding to use when reading file.

▼ weakly_connected_components

```
wcc = max(weakly_connected_components(G), key=len)
len(wcc)
```

weakly_connected_components — NetworkX 3.1 documentation

https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.components.weakly_connected_components.html#networkx.algorithms.components.weakly_connected_components

▼ strong_connected_components

strongly_connected_components — NetworkX 3.1 documentation

https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.components.strongly_connected_components.html#strongly-connected-components