HUDK 4051: LEARNING ANALYTICS: PROCESS & THEORY

2/6/20 2:52 PM

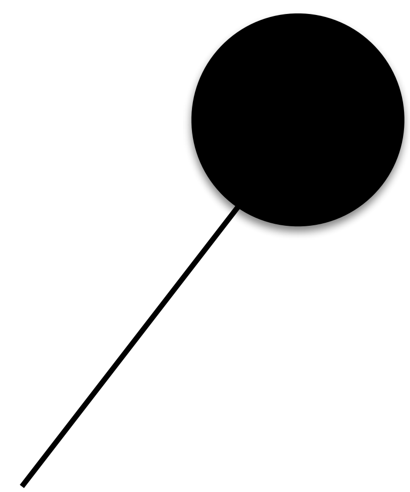
Today

- Social Network Analysis
- Recommender System Unit Due Feb 10
- Should be invited to the class Github

Social Network Analysis Recap

Networks

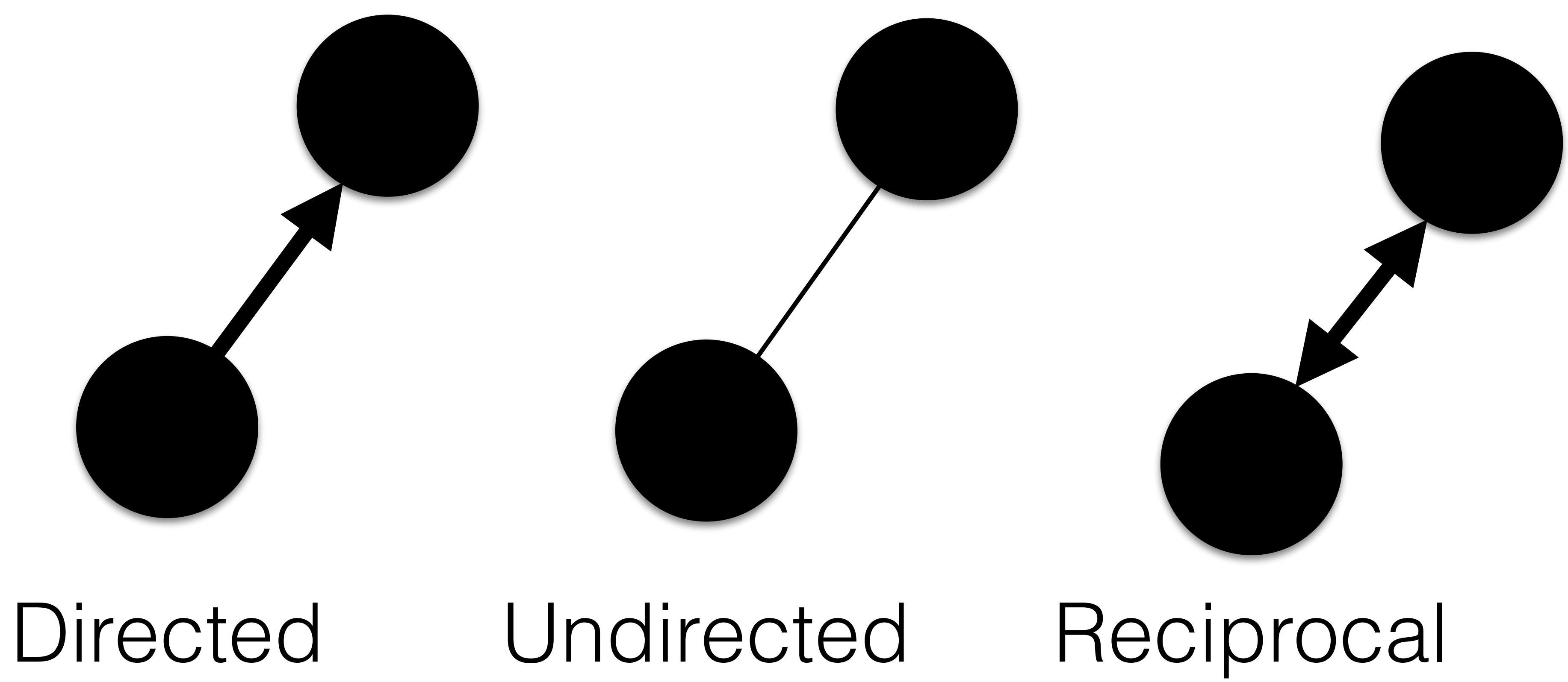
(Graphs)



Node
Vertex

Link
Edge
Tie
Arc

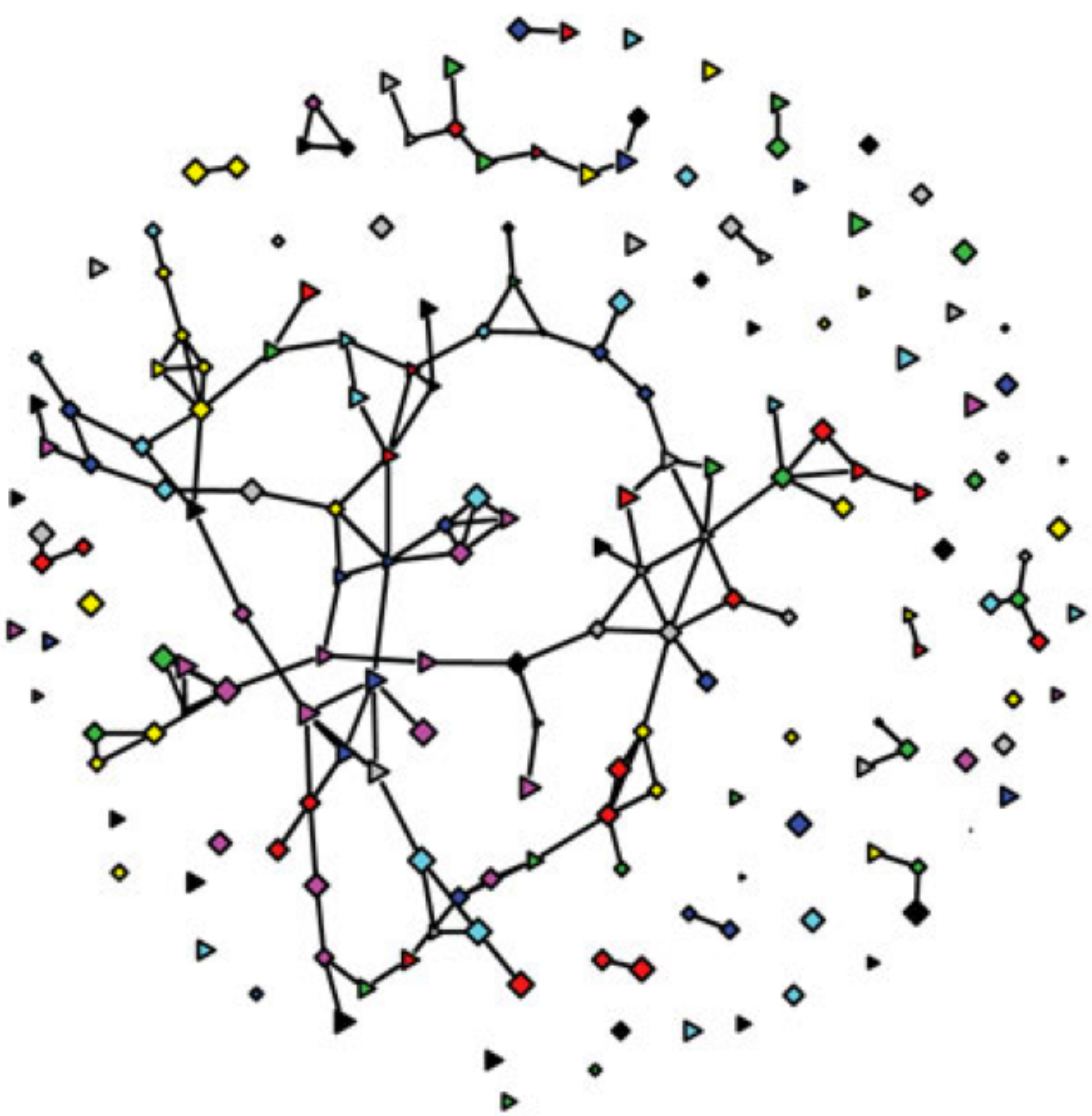
Networks



Degree

The number of links to other nodes in the network

Undirected

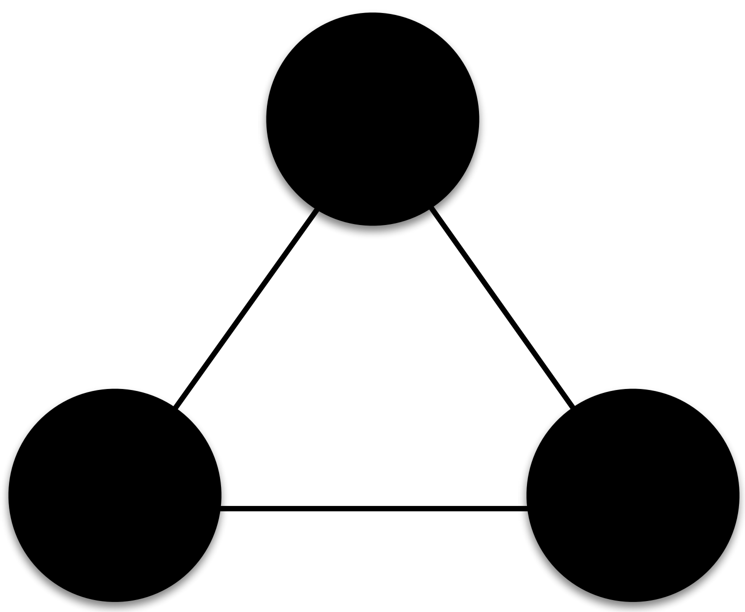


Directed

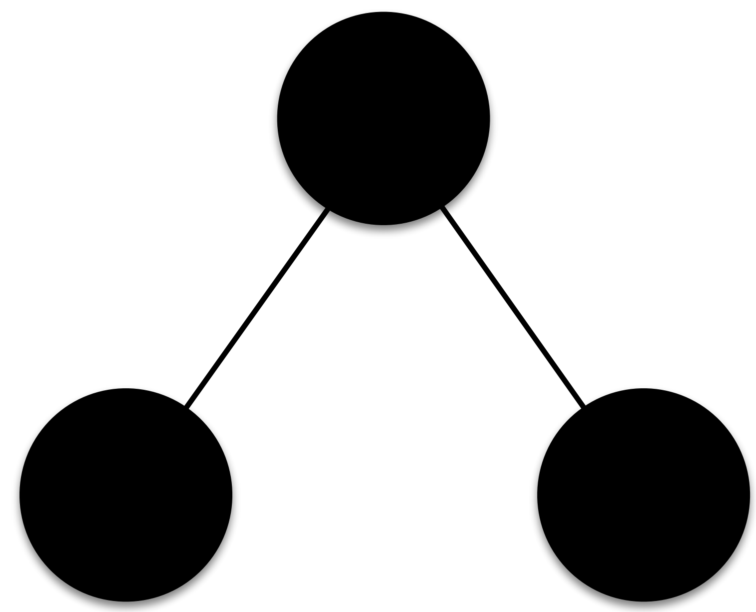
Indegree = Popularity
Outdegree = No shame

Density

How close is the graph to the maximal number of links



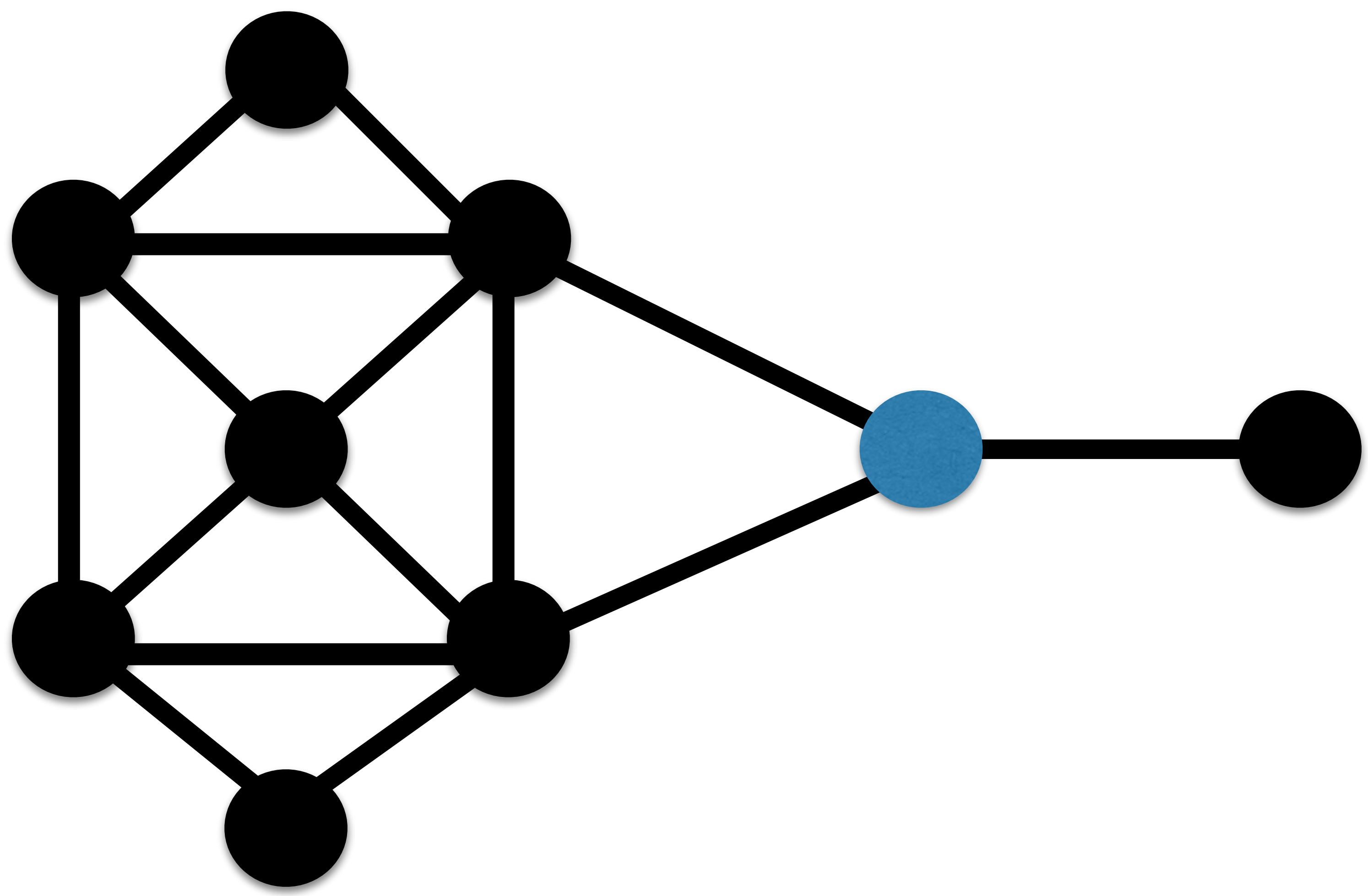
3 actual
3 possible
Density = 1



2 actual
3 possible
Density = 0.67

Betweenness Centrality

The extent to which a node lies between other nodes



It is equal to the number of shortest paths from all nodes to all others that pass through that node

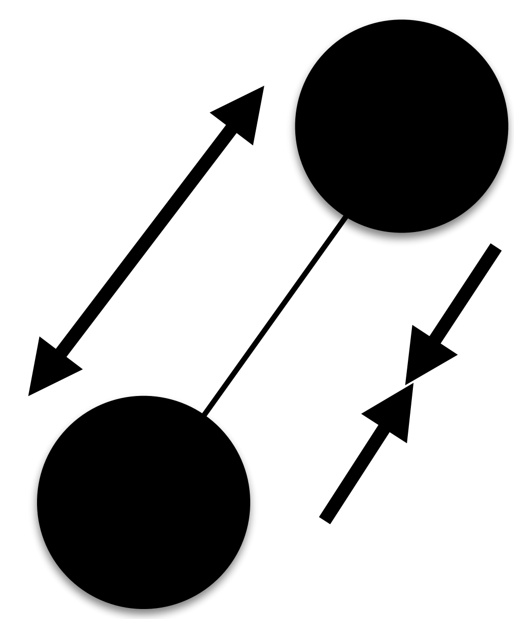
How do we make the network look nice?

Force directed graphing

- Attractive forces

  Springs

  Hooke's Law: $F = kX$



- Repulsive forces

  Electrons

  Coulomb's Law: $|\mathbf{F}| = k_e \frac{|q_1 q_2|}{r^2}$

https://youtu.be/YGDvR6CRwEc

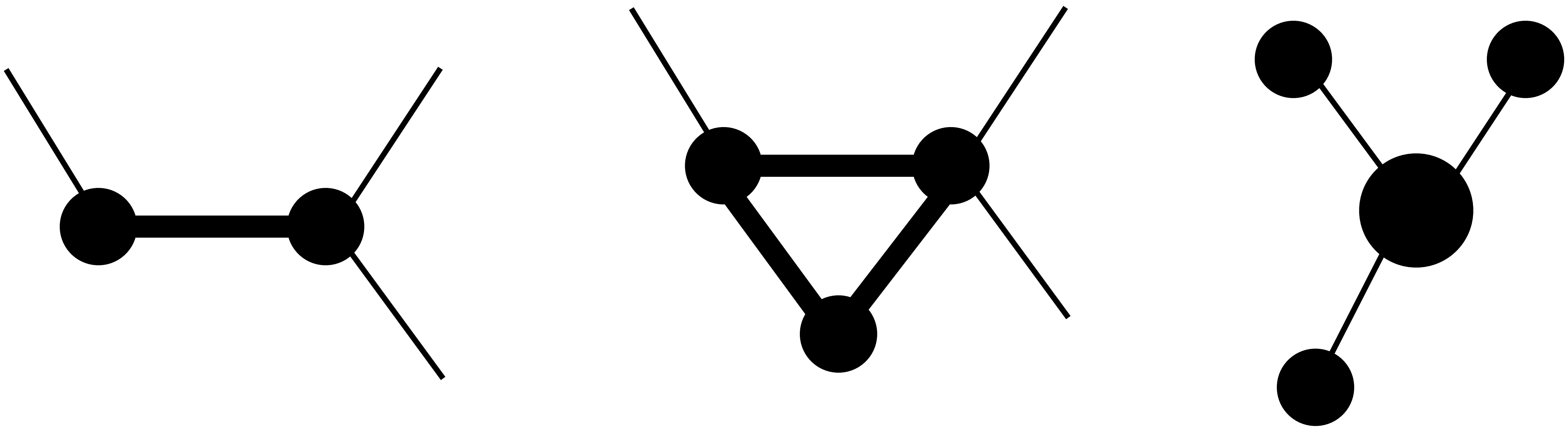
Modularity

The fraction of the edges that fall within the given groups minus the expected such fraction if edges were distributed at random

$$
\begin{aligned}
Q_{\mathrm{S}} &= \frac{1}{2\bar{w}} \sum_i \sum_j \left(\bar{w}_{ij} - \frac{\bar{w}_i \bar{w}_j}{2\bar{w}} \right) \delta(C_i, C_j) \\
&= \frac{1}{4w} \sum_i \sum_j \left(w_{ij} + w_{ji} - \frac{(w_i^{\mathrm{out}} + w_i^{\mathrm{in}})(w_j^{\mathrm{out}} + w_j^{\mathrm{in}})}{4w} \right) \delta(C_i, C_j) \\
&= \frac{1}{4w} \sum_i \sum_j \left[\left(w_{ij} - \frac{w_i^{\mathrm{out}} w_j^{\mathrm{in}}}{2w} \right) + \left(w_{ji} - \frac{w_i^{\mathrm{in}} w_j^{\mathrm{out}}}{2w} \right) \right] \delta(C_i, C_j) \\
&= -\frac{1}{(4w)^2} \sum_i \sum_j (w_i^{\mathrm{out}} - w_i^{\mathrm{in}})(w_j^{\mathrm{out}} - w_j^{\mathrm{in}}) \delta(C_i, C_j) \\
&= Q_{\mathrm{D}} - \frac{1}{(4w)^2} \sum_i \sum_j (w_i^{\mathrm{out}} - w_i^{\mathrm{in}})(w_j^{\mathrm{out}} - w_j^{\mathrm{in}}) \delta(C_i, C_j).
\end{aligned}
$$

SNA: Cliques & Subnetworks

Dyads, Triads, Ego-Centric Networks



- Ego - node of interest
- Neighborhood - nodes connected to ego at some path length (usually 1)
- Neighborhood can be In/Out, weak/strong, N-step

Larger Structures

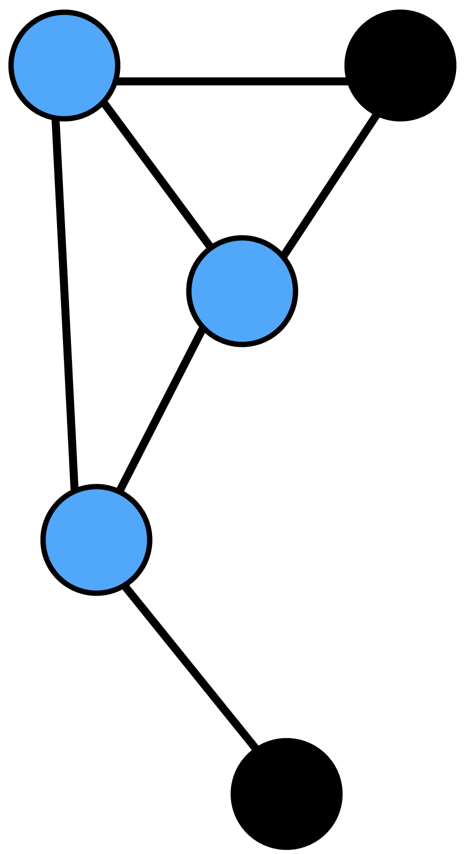
- Need to make an argument that nodes *belong together*
- Maximal clique = a clique that is not included in a larger clique
- Lots of ways to do this (density, Strength of ties, etc)
- Two main groups of methods: Bottom Up or Top Down

Bottom Up

- How does the macro emerge from the micro?
- Dyad = simplest group
- Clique/Complete Subgraph = maximum number of actors who have all possible ties present among themselves (all members are tied to all other members in the group)
- *Build* the network up from the bottom, comparing cliques sizes and relationships with each other

Algorithms

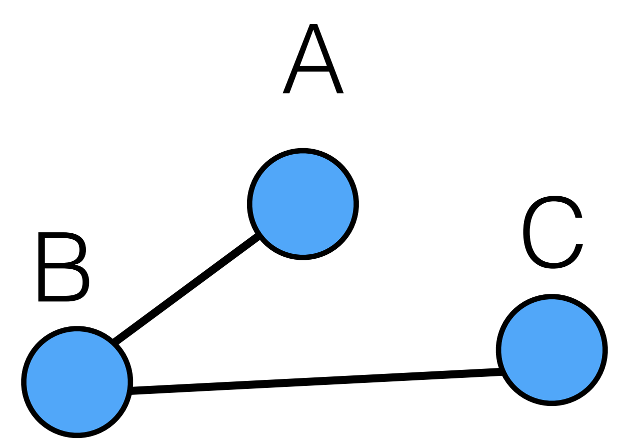
- Greedy algorithm: makes the locally best decision at each stage
    - Start with arbitrary node
    - Examine each node that it is connected to
    - If it is connected to every other node, keep, otherwise discard
    - Gotcha: Might not find the maximal clique only a local minima



Activity

- At your table
- Create a “friend matrix” of who is friends with who (both must agree)
- Draw the graph of friendships
- Stand up
- Choose one person at random and examine each person they are connected to
- Now choose one of those people and see if they are connected to the all the same people, if they are not then sit down if they are, then they can remain standing
- Go to the next person that the original person is connected to and repeat

	A	**B**	**C**
A		Y	N
B	Y		Y
C	N	Y	



Algorithms

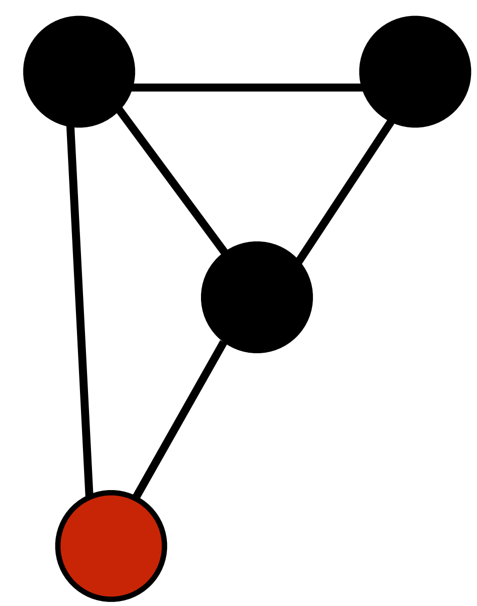
- Brute force algorithm: enumerate every possible combination of nodes

- Usually used to identify a clique of certain size (*k*)

Larger than cliques

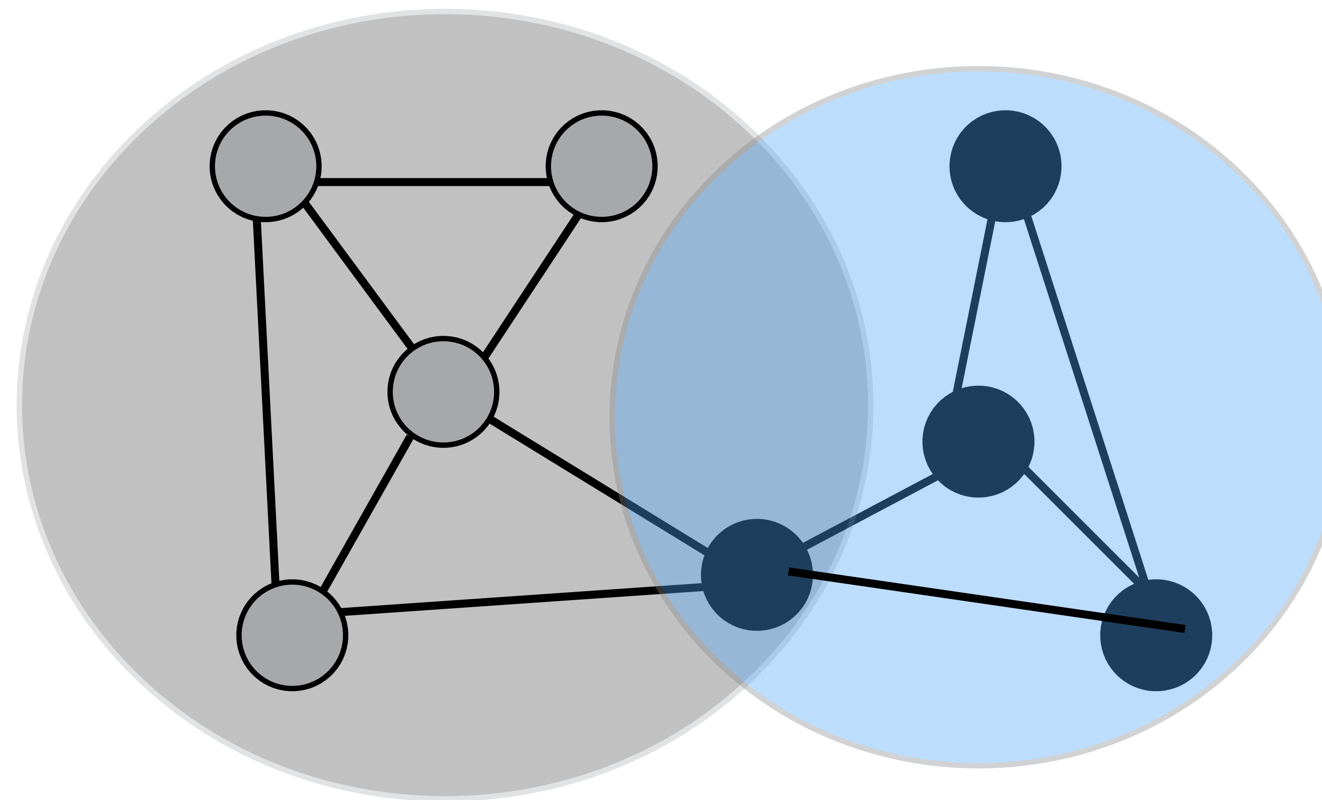
- Co-membership of cliques (overlap)
- <u>N-cliques</u>: Friend of a friend clique (can produce "stringy" groups)
- <u>N-clans</u>: Requires that all the ties among actors occur through other members of the group

- Are there any friends of friends in your graph?



Larger than Clans

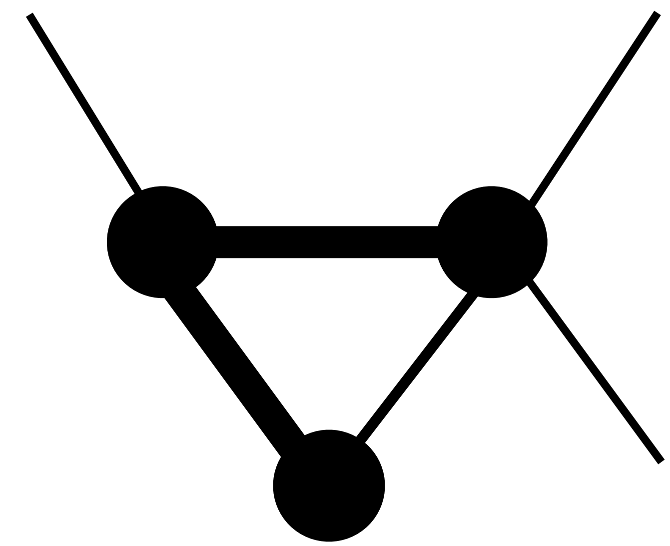
- K-plexes: A node is a member of a clique of size *n* if it has direct ties to *n*-*k* members of that clique
- K-cores: To be included in a core, an actor must be tied to all but *k* other actors in the group

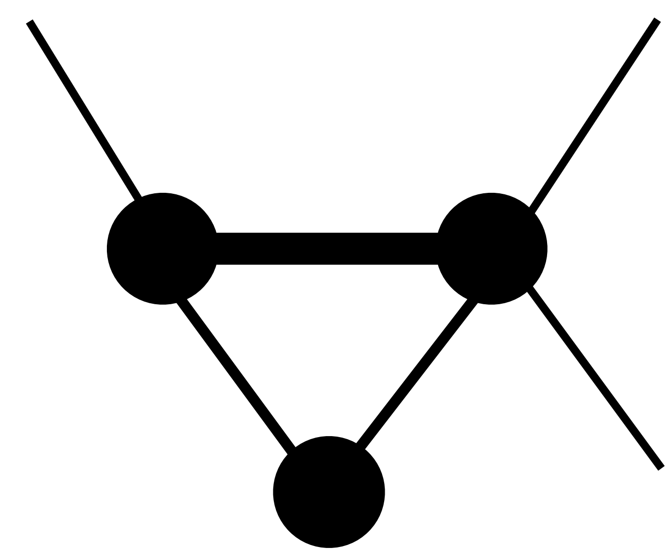


n >= 4, k = 2

F-Groups

- Incorporate strength of connection into grouping
- Equates grouping with balanced triads



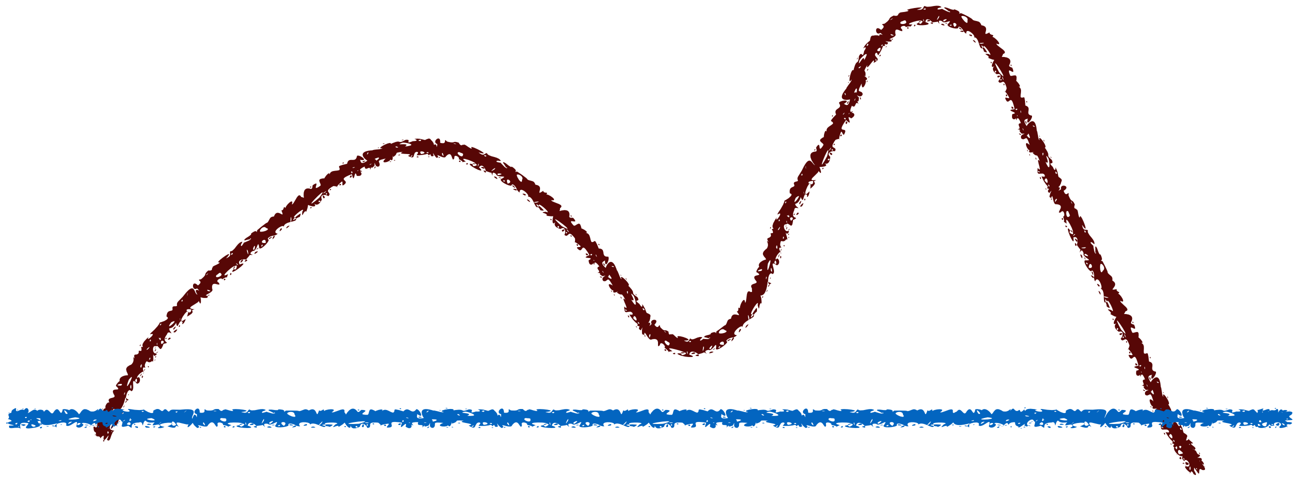


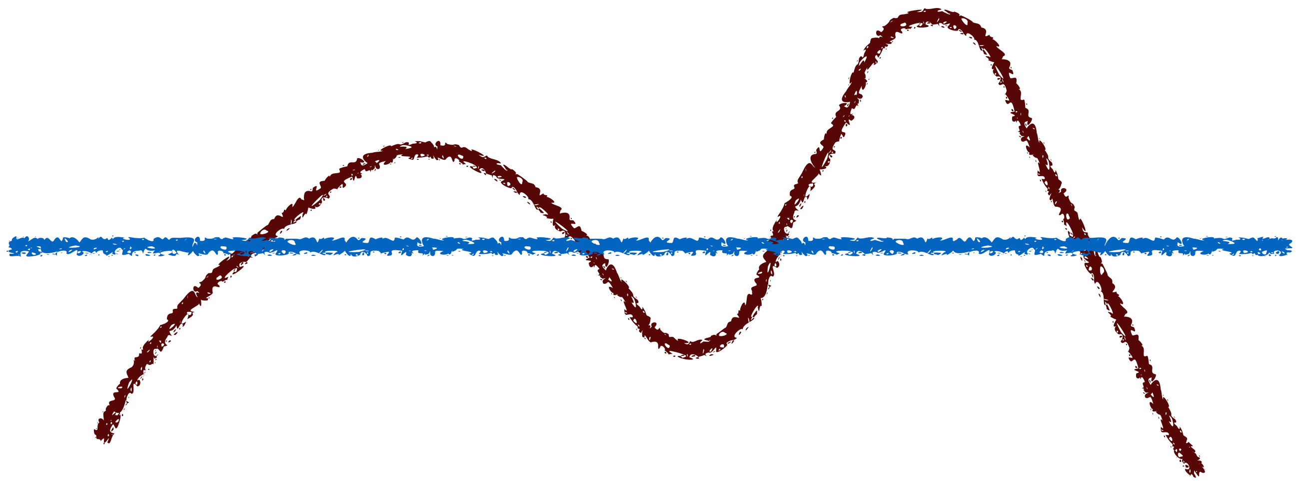
Top Down

- Looking for "holes" or "vulnerabilities" or "weak spots" in the overall structure
- Focuses on the constraints under which the network has developed
- Asks questions about overall system dynamics

Top Down Groupings

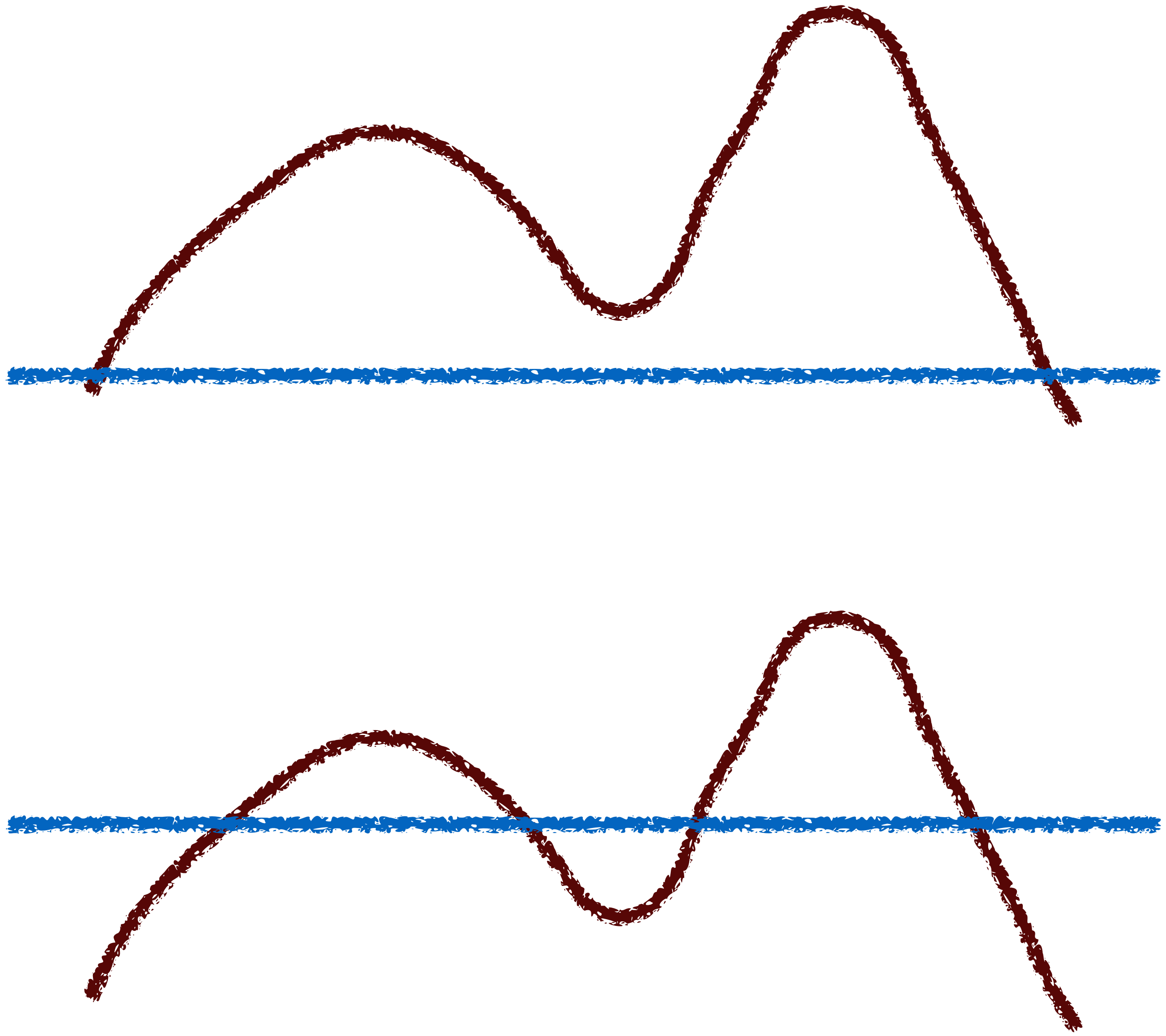
- Components: Disconnected parts of a graph
- Island Method: Take a measure of connectivity, remove nodes below a certain threshold. As if you are raising the water level around an island. What is left are the subgraphs
- Could also use connection weight or direction also





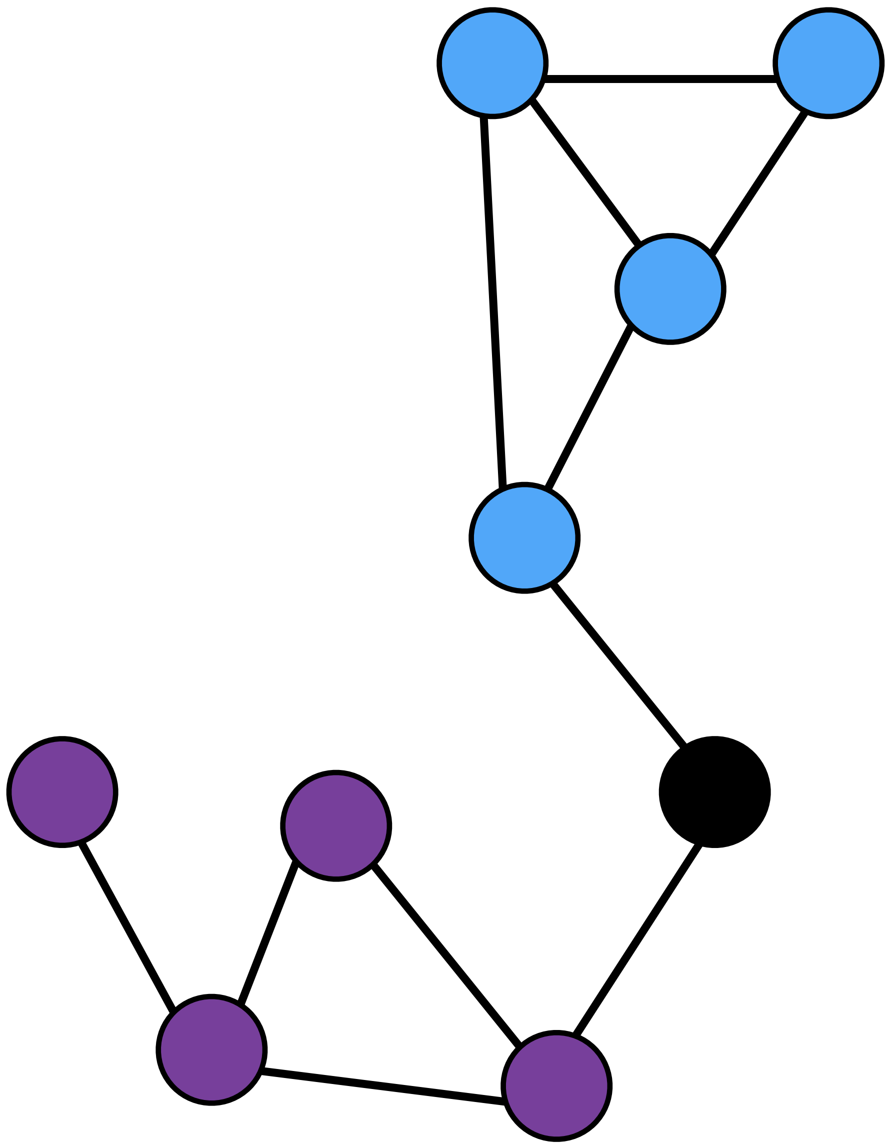
Activity

- Stand up
- Looking at your friend graph
- What is the degree centrality of each person?
- Start at the minimum degree
- Raise the degree by 1 unit increments
- As you raise degree, sit down once the level passes your degree
- At what point do components appear?



Blocks & Cut/Articulation points

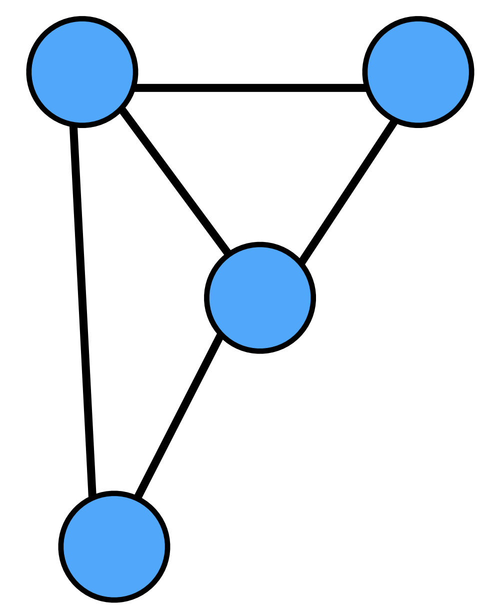
- Cutpoint: If node is removed it creates two separate graphs
- Block: group without cut-points (non-separable)
- Could also use connection weight or direction
- Are there any cutpoints in your friend graph?



Bridges & Lambda Sets

- Bridge: If edge is removed it creates two separate graphs
- Lambda Sets: Measures the flow through the network
- Rank sets with respect to importance (amount of flow)
- Highlights points at which the fabric of connection is most vulnerable to disruption

- Are there any bridges in your friend graph?



Factions

- Imagine an idealized graph made up of three unconnected groups
- These are factions
- We can compare this imaginary graph to an actual graph to get a sense of how factionalized a network is
- Algorithm compares adjacency matrices to find the arrangement of nodes within the real data that most closely matches a theoretical factionalized network

	A	B	C	D
A		1	0	0
B	1		0	0
C	0	0		1
D	1	0	1	