

Network Analysis:

The Hidden Structures behind the Webs We Weave

17-213 / 17-668

Communities

Thursday, September 28, 2023

Patrick Park & Bogdan Vasilescu

2-min Quiz, on Canvas

Quick Recap – Last Tuesday's Lecture

Carefully examine assumptions of network measures:

- Centrality metrics do not accurately predict “power” in negatively connected exchange networks (zero-sum)

Make the hidden assumptions explicit:

- Bonacich power centrality explicates the assumption with the beta parameter
- Better prediction of power use in experimental data (higher resource gains)

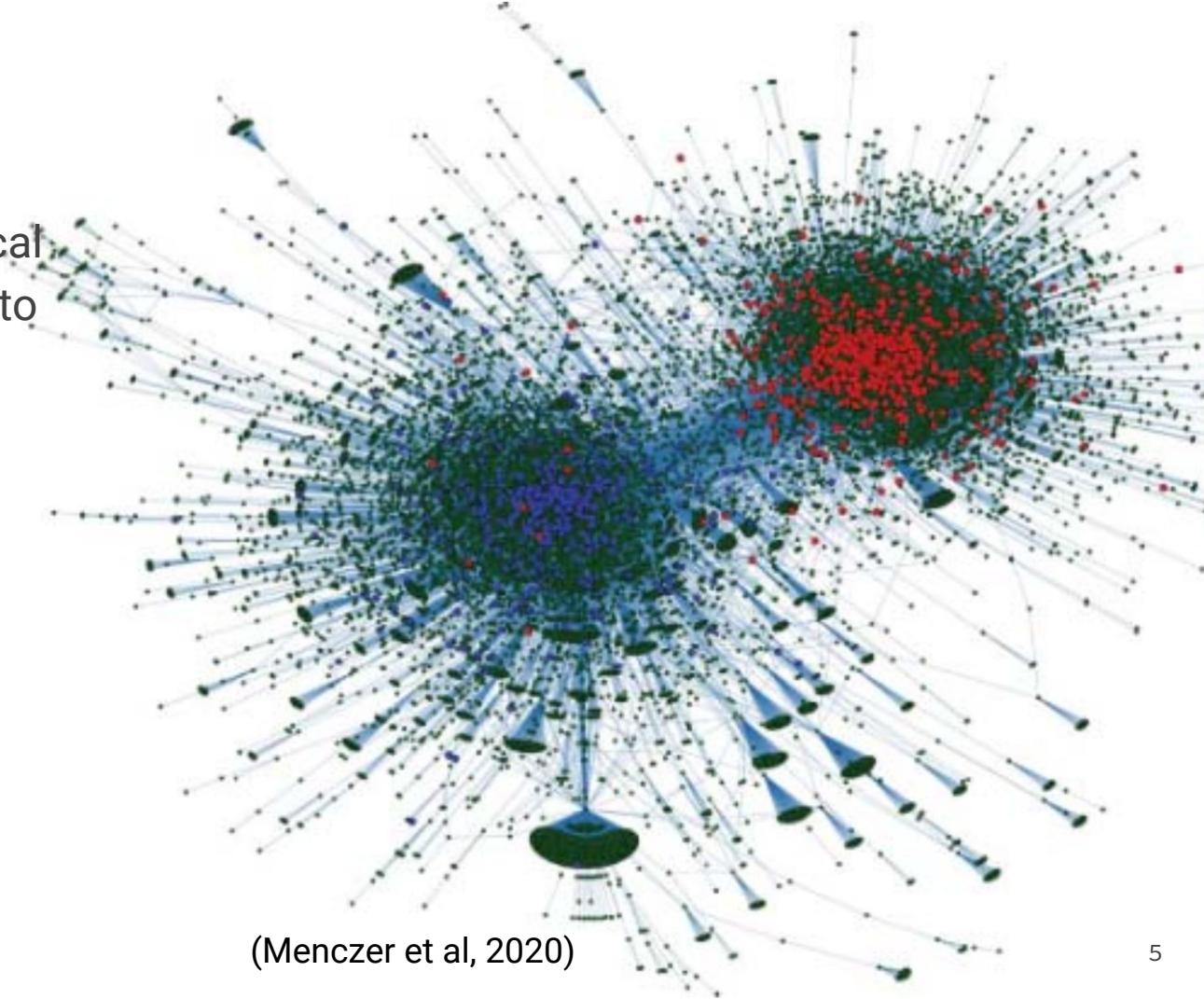
Further creative extensions:

- Building on the eigenvector-like centrality measure
- Insight: Fragility of neighbors increases my fragility
- Used Herfindahl index of concentration instead of node degree

Social networks are full of easy to spot
“communities” (cohesive subgroups)

Twitter users

Retweet network of political hashtags on Twitter prior to the 2010 US election.



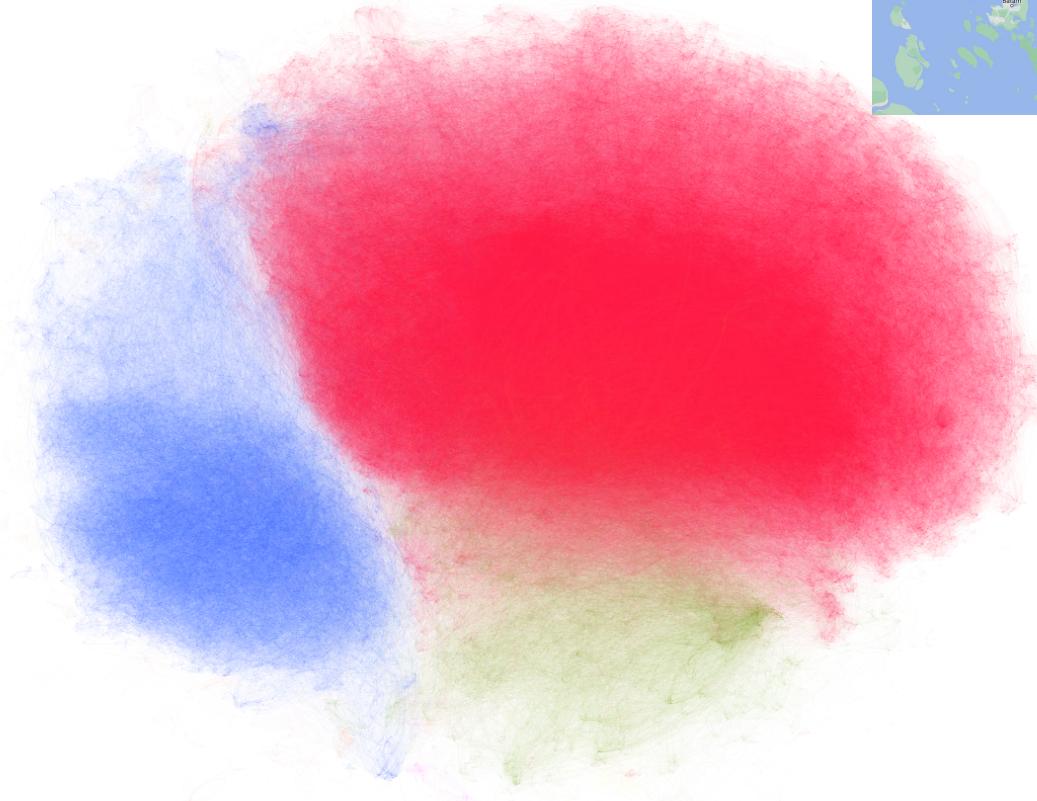
(Menczer et al, 2020)

Twitter users

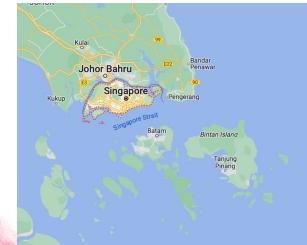
Bidirected @mention network
among Singapore Twitter
users

Colors based on community
detection

Q: What attribute do you think
the colors correspond to?



(Patrick Park, unpublished)

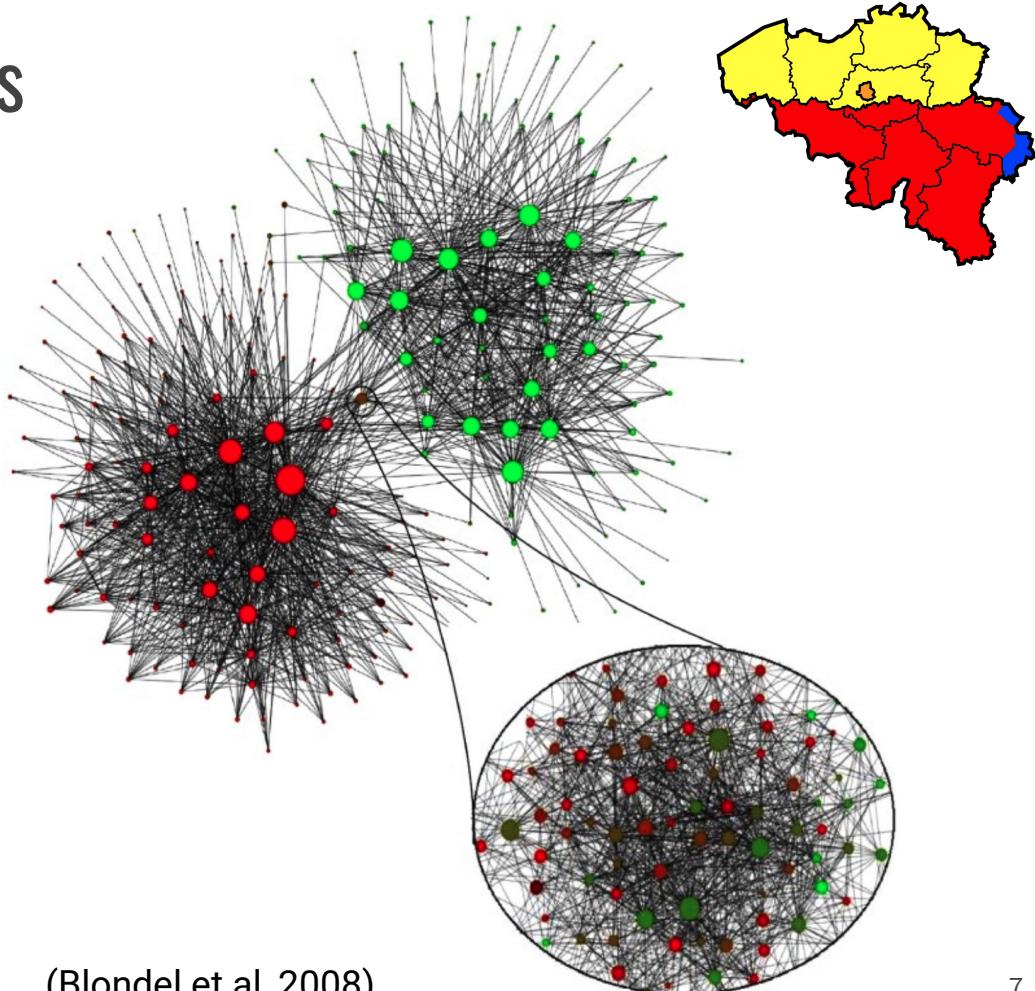


Belgian mobile phone users

The nodes correspond to communities.

The color represents the language spoken in the particular community: red for French and green for Dutch.

Bridge communities (Brussels) show less obvious language separation.



Subgroups: easy to spot, but tricky to define

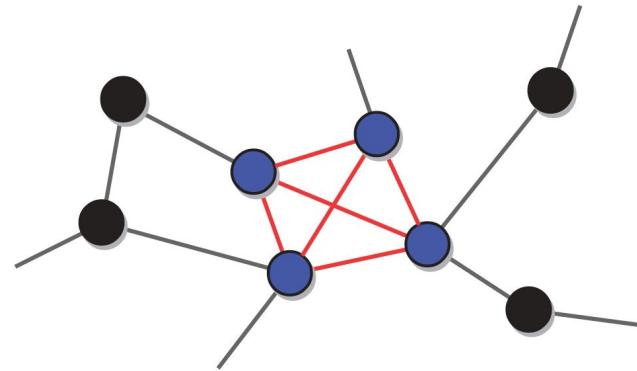
Social group is fundamental to humans

Yet a “group” lacks formal definition

- Too “obvious” to define
- But what is a group?

They come in all size, shapes, and forms

- Size (family, nation state)
- Intimacy (private vs. professional)
- Language
- Geography
- Means of production (capitalist vs. proletariat)
- ...

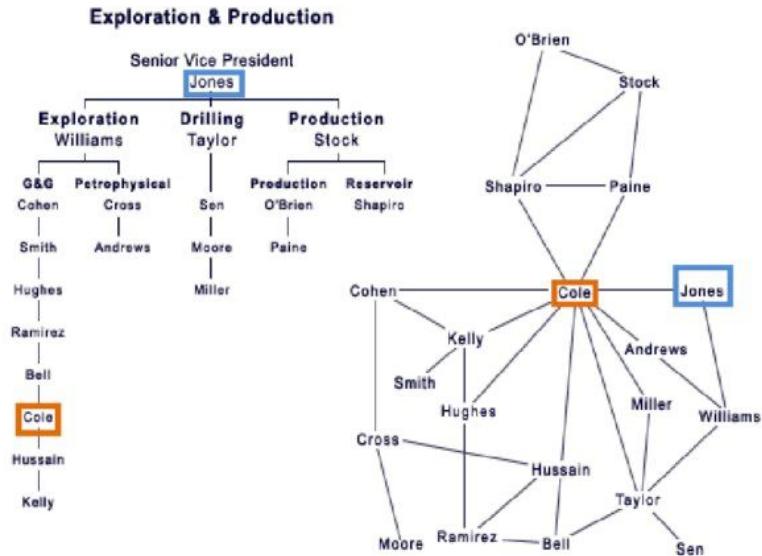


Subgroups: easy to spot, but tricky to define

The difficulty is apparent when you try to define groups **top-down**

- University C, Department X, Unit Z
- Member overlap: Department X and Y can share common members
- Informal groups: Some members in X have stronger ties to members in Y

Actual cohesion does not always form along formal groups

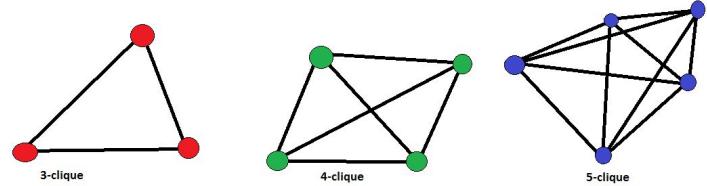


Source: Rob Cross, What is ONA? http://www.robcross.org/network_ona.htm

Subgroups: easy to spot, but tricky to define

The network approach develops **Bottom-up** approach to quantifying subgroups based on:

- Direct connections:
 - Clique: maximal subset of nodes with direct ties to one another

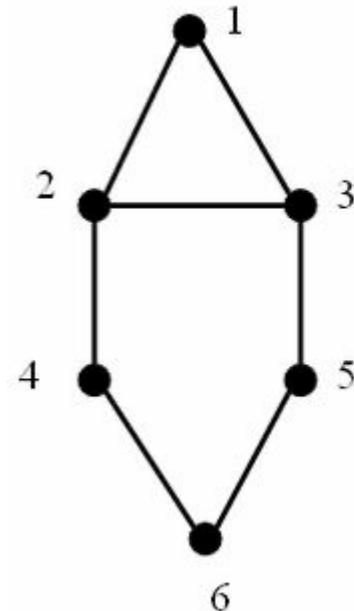


Subgroups: easy to spot, but tricky to define

The network approach develops **Bottom-up** approach to quantifying subgroups based on:

- Direct connections:
 - Clique: maximal subset of nodes with direct ties to one another
- Distance:
 - n -clan: Maximal subgraph of nodes that are within a path length n only through the nodes in that subset

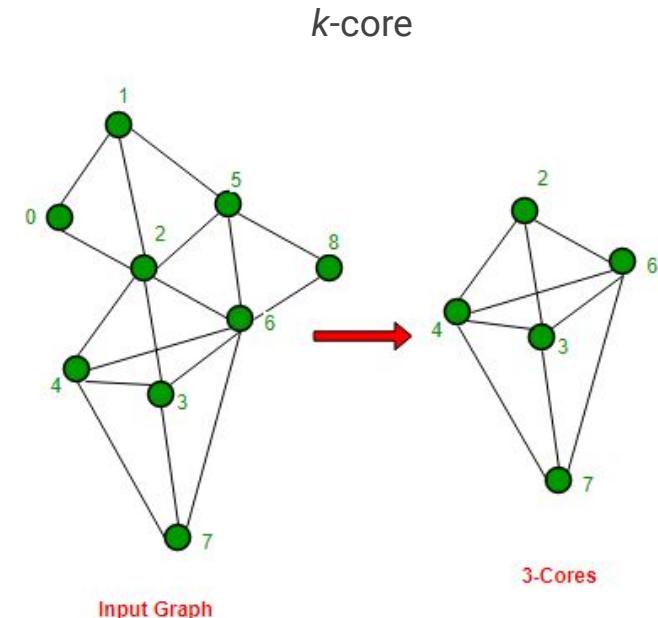
2-clan: {2, 3, 4, 5, 6}



Subgroups: easy to spot, but tricky to define

The network approach develops **Bottom-up** approach to quantifying subgroups based on:

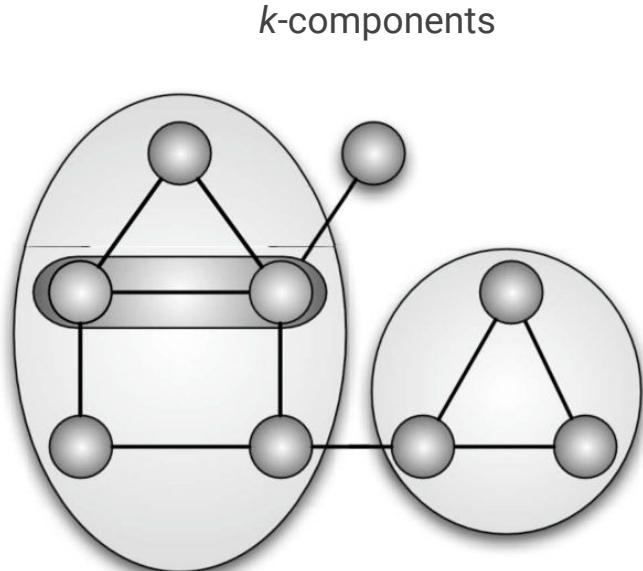
- Direct connections:
 - Clique: maximal subset of nodes with direct ties to one another
- Distance:
 - n -clan: Maximal subgraph of nodes that are within a path length n only through the nodes in that subset
- Redundancy (many ways to reach others):
 - k -core: Maximal subgraph in which every node has edges to at least k other nodes in the subgraph



Subgroups: easy to spot, but tricky to define

The network approach develops **Bottom-up** approach to quantifying subgroups based on:

- Direct connections:
 - Clique: maximal subset of nodes with direct ties to one another
- Distance:
 - n -clan: Maximal subgraph of nodes that are within a path length n only through the nodes in that subset
- Redundancy (many ways to reach others):
 - k -core: Maximal subgraph in which every node has edges to at least k other nodes in the subgraph
 - k -component: Every node has at least k non-overlapping paths to every node in the subgraph



Basic concepts

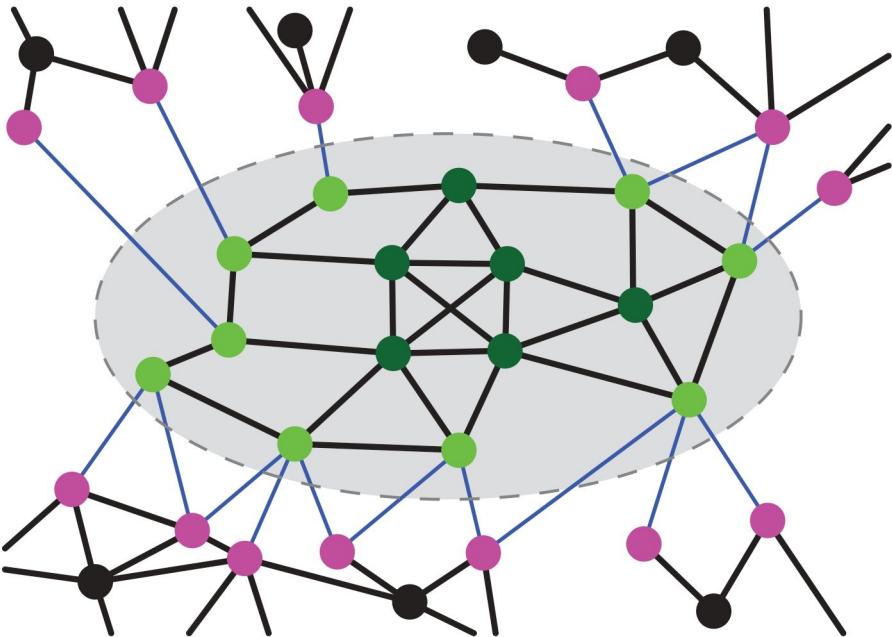
Internal links (black links)

Internal (black links) & external (blue links) degree of a node in the community (green nodes)

$$k_i^{int} \quad k_i^{ext}$$

Community degree (sum of num neighbors of each internal node)

$$k_C = \sum_{i \in C} k_i.$$



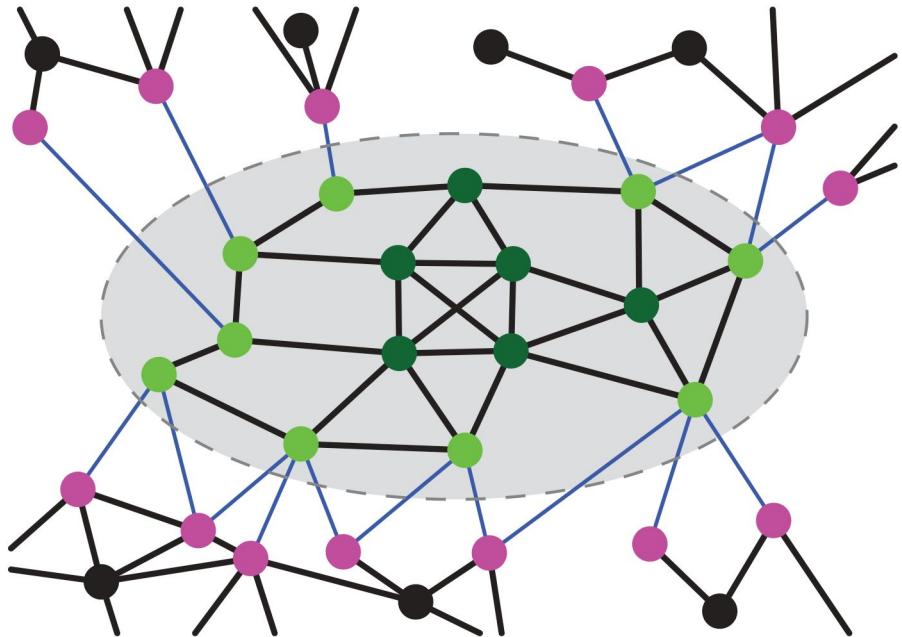
Recall

The maximum number of links in an undirected network with N nodes:

?

The density of a network with N nodes and L links:

?



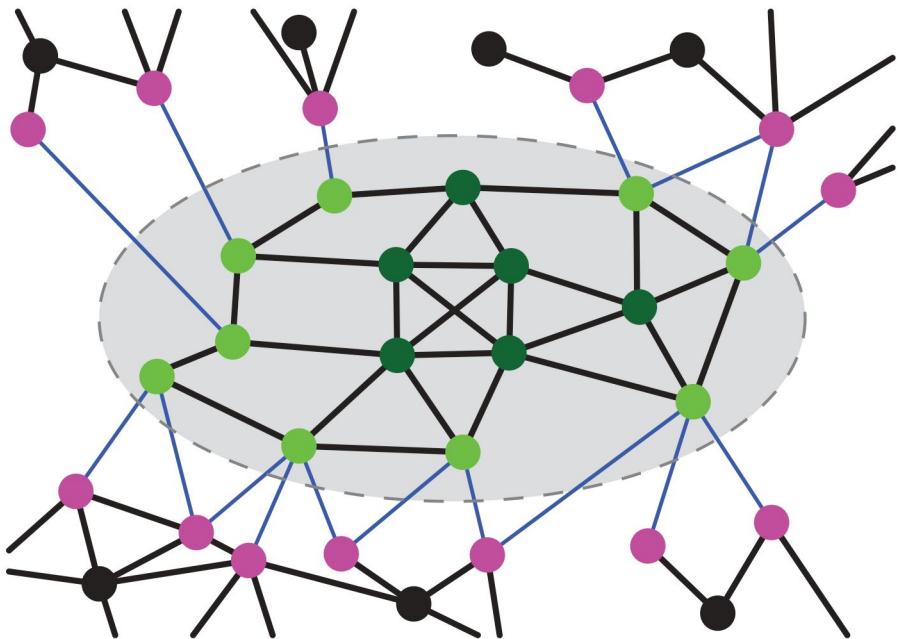
Recall

The maximum number of links in an undirected network with N nodes:

$$L_{max} = \binom{N}{2} = N(N - 1)/2.$$

The density of a network with N nodes and L links:

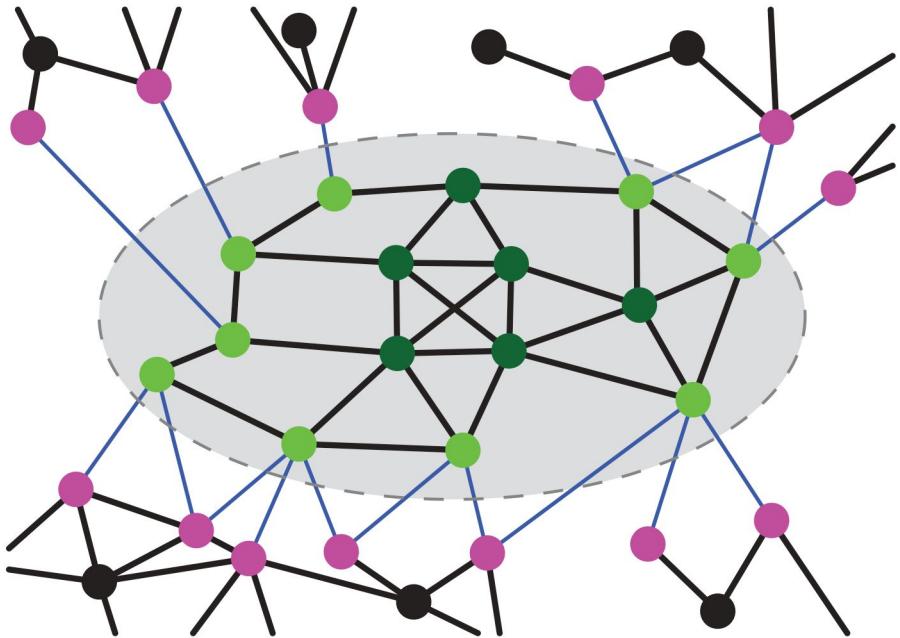
$$d = L/L_{max} = \frac{2L}{N(N - 1)}$$



Basic concepts

Internal link density:

$$\delta_C^{int} = \frac{L_C}{\binom{N_C}{2}} = \frac{2L_C}{N_C(N_C - 1)}.$$



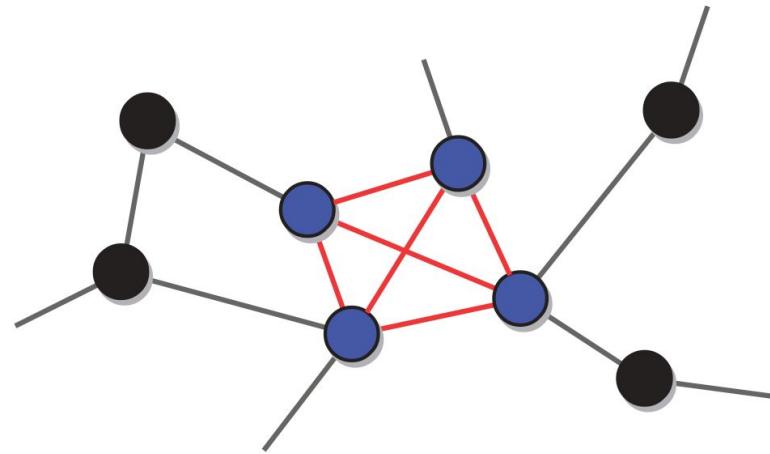
Intuition: Nodes within a “community” have higher likelihood of connecting to each other than to nodes from other “communities.”
→ high “cohesion,” high “separation”

Aside: Cliques have high cohesion, but aren't realistic communities

Real communities aren't as dense as cliques.

In real communities some nodes are more important than others.

Better: The number of internal links should be larger than the number of external links.

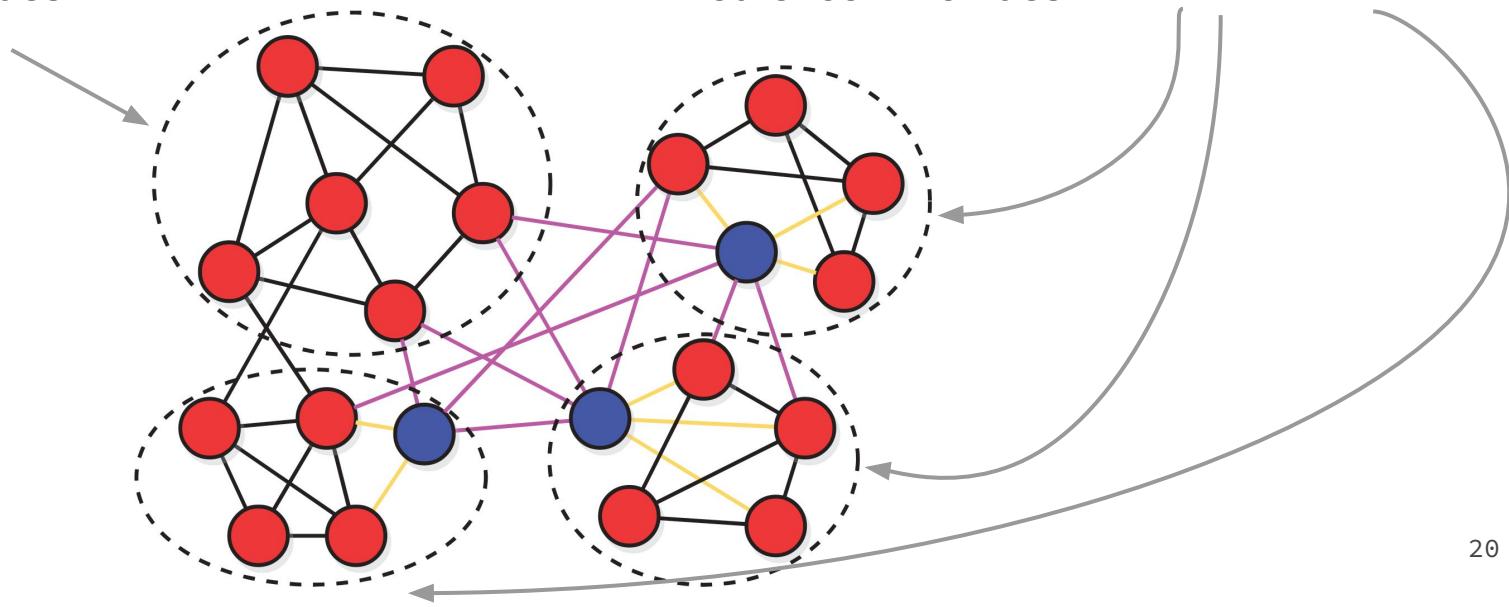


“Strong” vs “weak” communities

Strong

The internal degree of each node exceeds its external degree towards other communities.

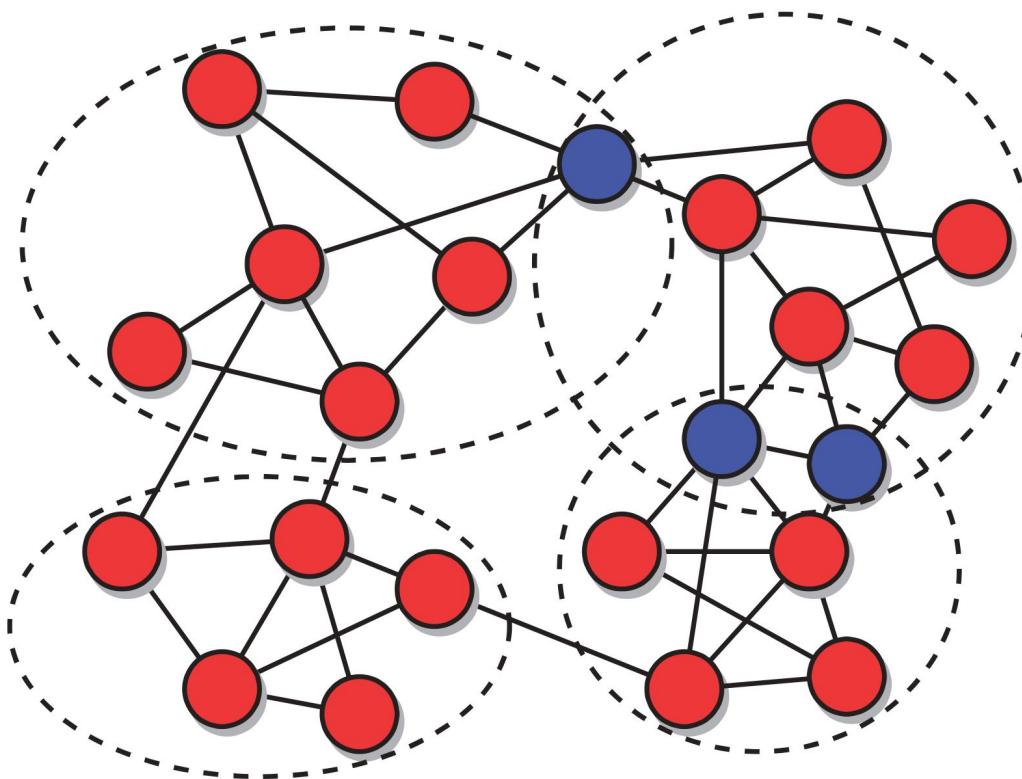
strong → weak



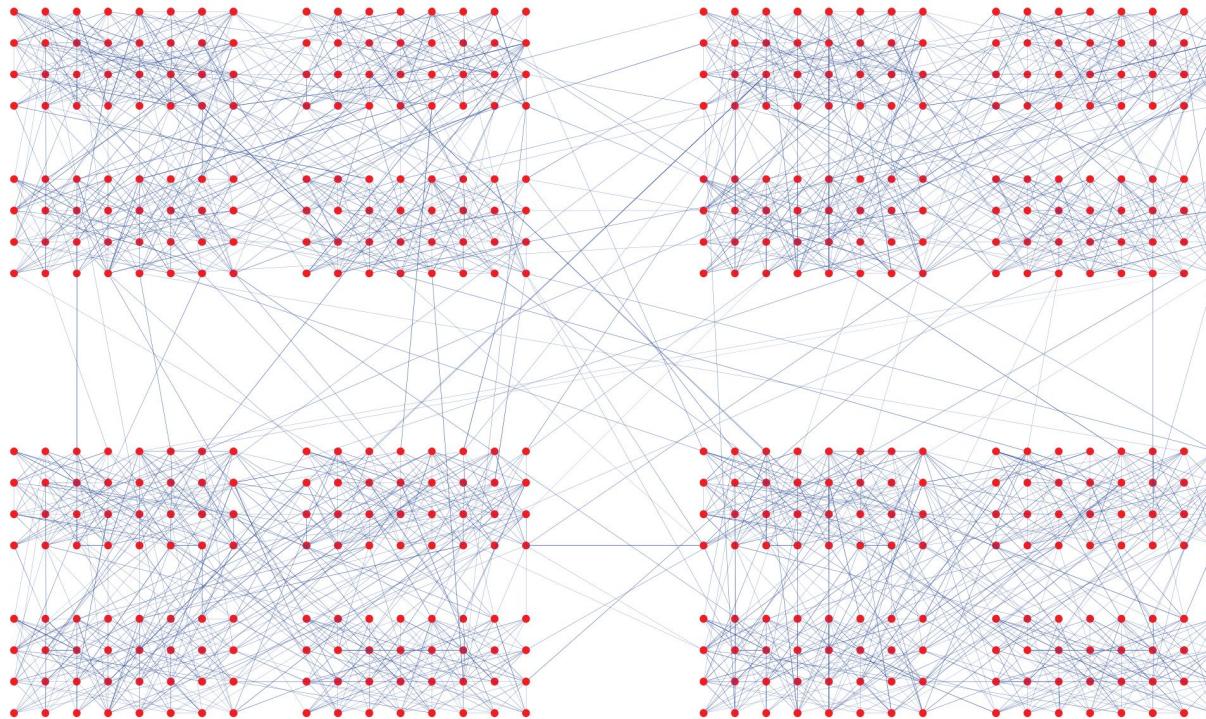
Weak

The sum of internal degrees of all nodes exceeds the sum of their external degrees in other communities.

The communities in many real-world networks overlap



Partitions can be hierarchical when the network has multiple levels of organization



So, how to find the communities?

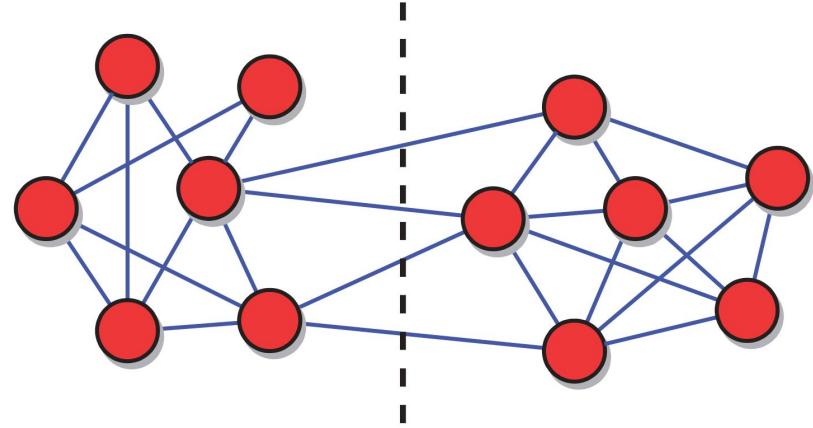
1. Graph partitioning – old problem

Min-cut graph bisection doesn't quite work

Trivial solution to minimizing cut size:
single cluster containing the entire
network gives cut size of zero.

→ Need to specify the number of
clusters beforehand.

(Also need to specify size of each cluster beforehand. Example: one leaf vs
all other nodes)

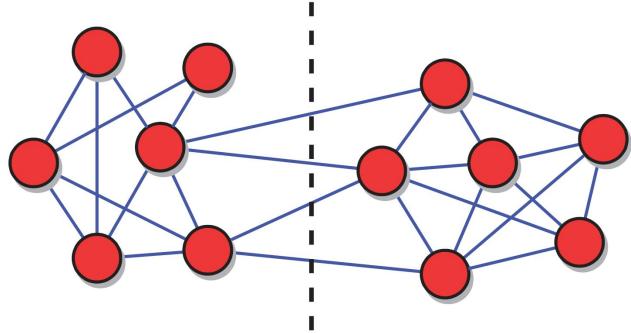


Kernighan-Lin graph bisection algorithm

We start from an arbitrary partition P of the network into two clusters A and B . For instance, we can select half of the nodes at random and put them in one cluster, and the rest in the other cluster. Each iteration of the algorithm consists of the following steps:

1. For each pair of nodes i, j , with $i \in A$ and $j \in B$, compute the variation in cut size between the current partition and the one obtained by swapping i and j .
2. The pair of nodes i^* and j^* yielding the largest decrease in the cut size is selected and swapped. This pair of nodes is locked; they will not be touched again during this iteration.
3. Repeat steps 1 and 2 until no more swaps of unlocked nodes yield a decrease in the cut size. This yields a new partition P' , that is used as a starting configuration for the next iteration.

The procedure ends when the cut size of partitions obtained after consecutive iterations is the same, meaning that the algorithm is unable to improve the result. The Kernighan–Lin algorithm can easily be extended to partitions with more than two clusters, by swapping nodes between pairs of clusters.

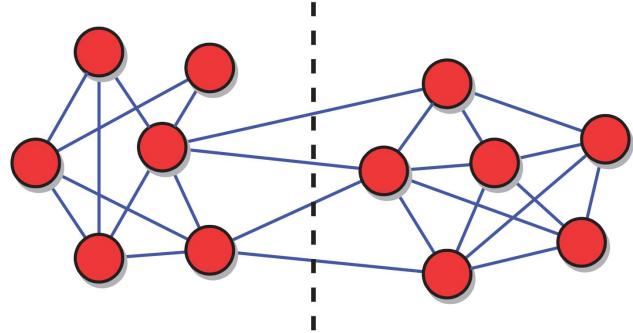


Kernighan-Lin graph bisection algorithm

We start from an arbitrary partition P of the network into two clusters A and B . For instance, we can select half of the nodes at random and put them in one cluster, and the rest in the other cluster. Each iteration of the algorithm consists of the following steps:

1. For each pair of nodes i, j , with $i \in A$ and $j \in B$, compute the variation in cut size between the current partition and the one obtained by swapping i and j .
2. The pair of nodes i^* and j^* yielding the largest decrease in the cut size is selected and swapped. This pair of nodes is locked; they will not be touched again during this iteration.
3. Repeat steps 1 and 2 until no more swaps of unlocked nodes yield a decrease in the cut size. This yields a new partition P' , that is used as a starting configuration for the next iteration.

The procedure ends when the cut size of partitions obtained after consecutive iterations is the same, meaning that the algorithm is unable to improve the result. The Kernighan–Lin algorithm can easily be extended to partitions with more than two clusters, by swapping nodes between pairs of clusters.



Greedy, risks getting stuck
in local optima.

Not bad, but we can do better.

Clusters identified via network partitioning are well-separated but not necessarily cohesive.

2. Clustering – also old problem

The main ingredient is a similarity measure between nodes

A classic example is structural equivalence, which expresses the similarity between the neighborhoods of a pair of nodes.

$$S_{ij}^{SE} = \frac{\text{number of neighbors shared by } i \text{ and } j}{\text{total number of nodes neighboring only } i, \text{ only } j, \text{ or both}}.$$

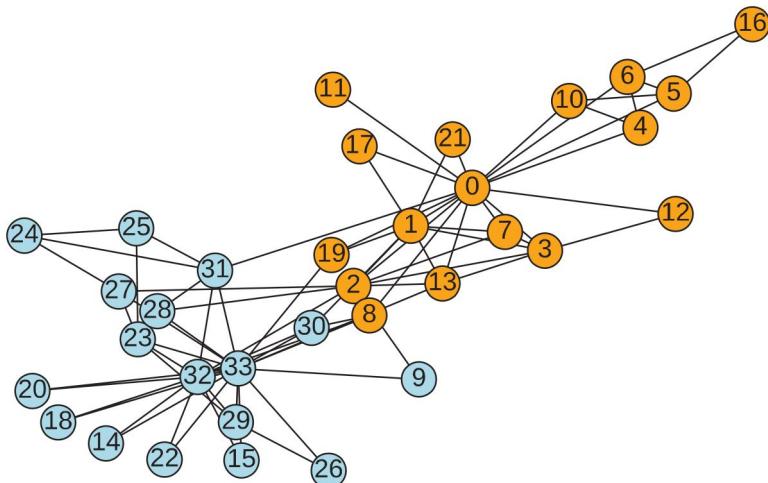
We also need to define similarity for groups of nodes

Given a node similarity measure S and two groups of nodes G_1 and G_2 :

- Single linkage uses the maximum pairwise similarity: $S_{G_1 G_2} = \max_{i,j} S_{ij}$.
- Complete linkage uses the minimum pairwise similarity: $S_{G_1 G_2} = \min_{i,j} S_{ij}$.
- Average linkage uses the average pairwise similarity: $S_{G_1 G_2} = \langle S_{ij} \rangle_{i,j}$.

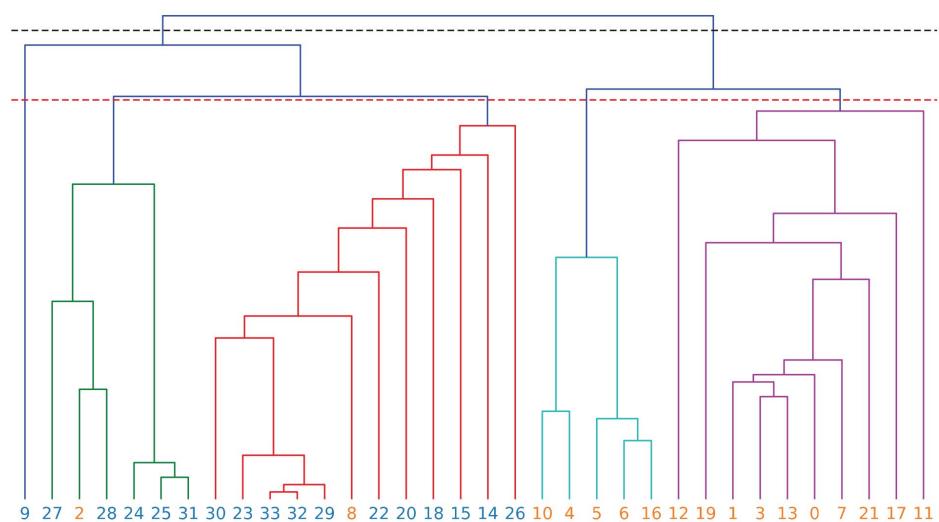
Next we can apply agglomerative hierarchical clustering

Start from the trivial partition into N groups. At each step, merge the pair of groups with the largest similarity. Repeat until all nodes are in the same group.



Zachary's karate club network.

Node 0: instructor. Node 33: club president

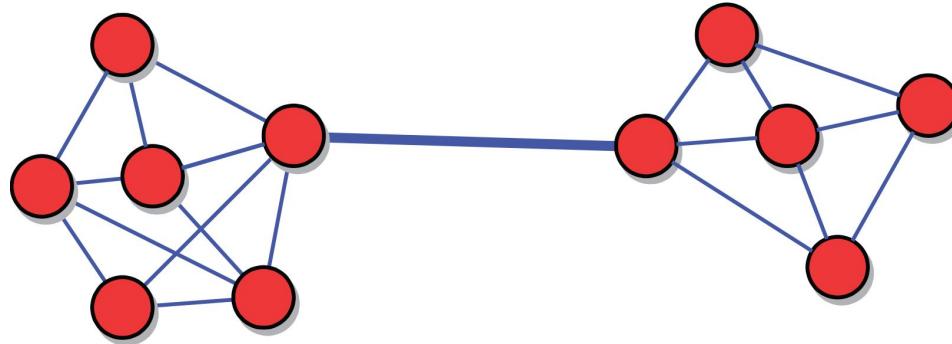


As many partitions as there are nodes → Unclear which partition is meaningful for the given network. Plus, rather slow.

3. Community detection

3.1. Bridge removal

Key idea: Find links with high betweenness and remove them.



Link betweenness defined similarly to node betweenness centrality in previous lecture – fraction of shortest paths that run through that link.

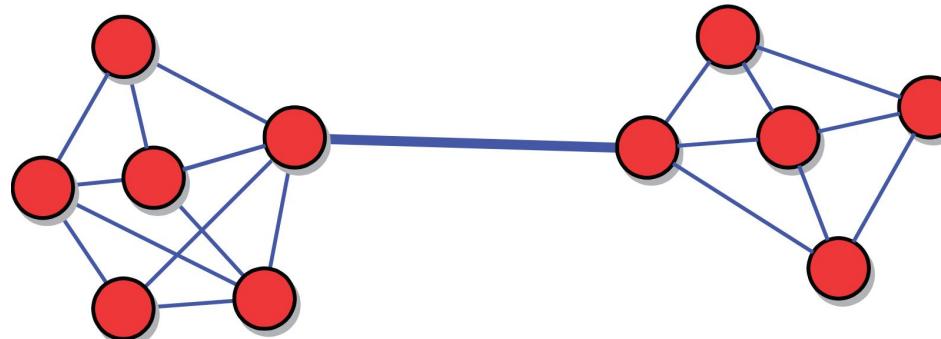
Link betweenness should be higher for bridges than for links inside a cluster.

Girvan-Newman algorithm (similar to hierarchical clustering)

We start by calculating the betweenness for all links. Then, each iteration of the algorithm consists of two steps:

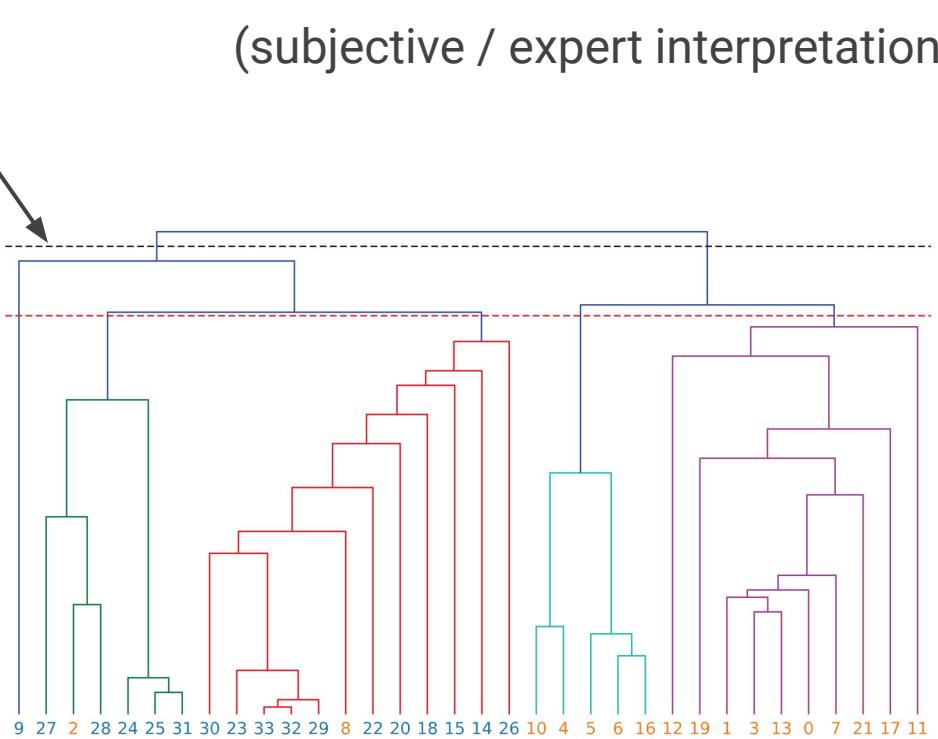
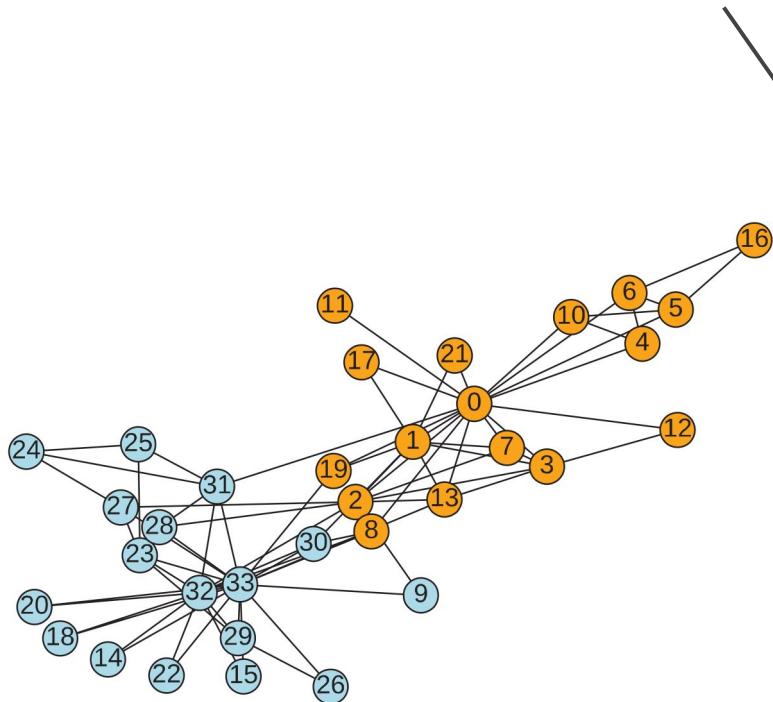
1. Remove the link with largest betweenness; in case of ties, one of them is picked at random.
2. Recalculate the betweenness of the remaining links.

The procedure ends when all links are removed and the nodes are isolated.



Girvan-Newman algorithm on Zachary's karate club: 2 clusters

(subjective / expert interpretation)



Girvan-Newman algorithm - reflections

Slow – must recompute the betweenness of all links each iteration.

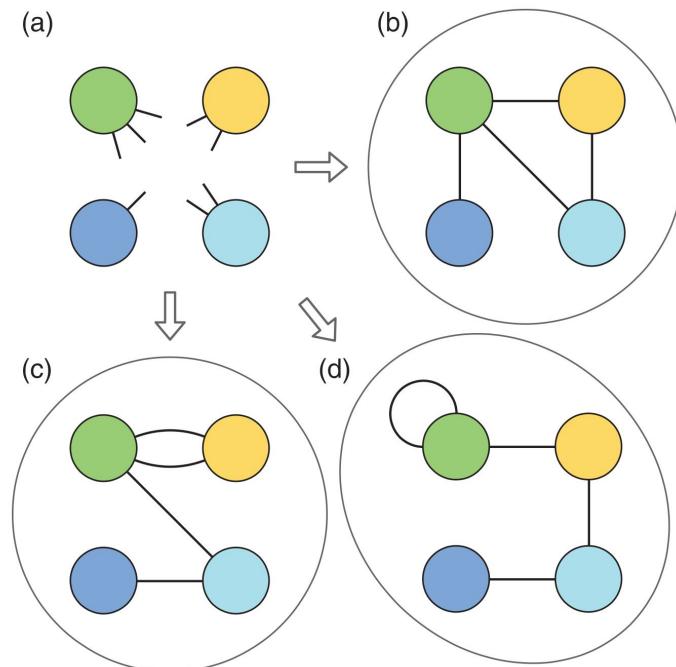
Improvement: recompute betweenness only within the connected component including the last removed link.

We still need a measure of the quality of a partition.

Modularity

The difference between the number of links internal to all clusters and the expected equivalent number in a randomized network.

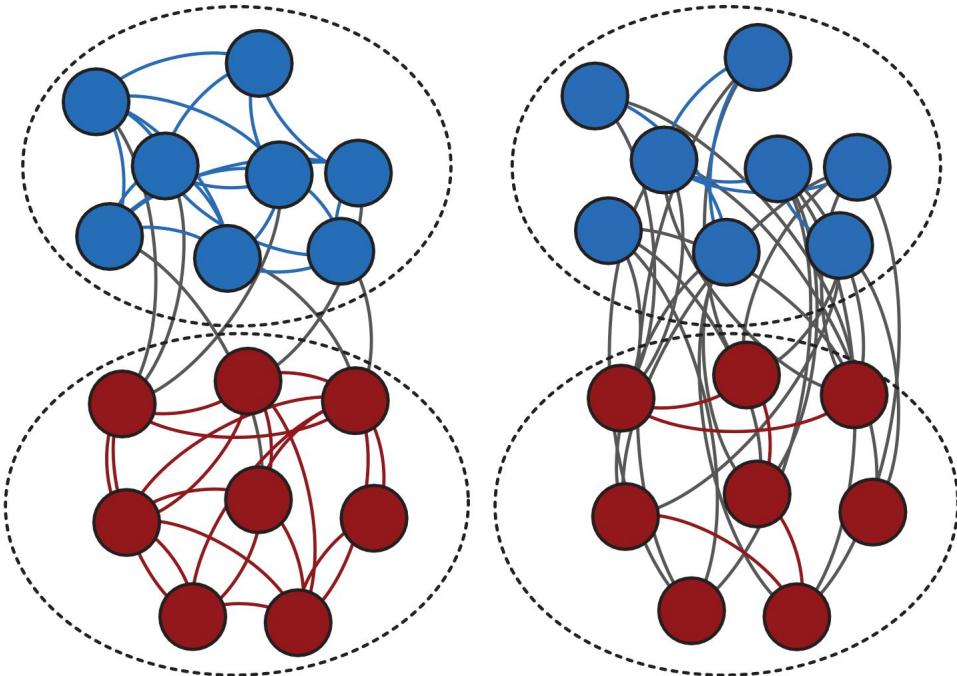
Randomization strategy: maintain number of nodes and degree sequence, shuffle links.



Modularity

Left network: visible community structure (high modularity).

Right network: degree-preserving randomization – fewer internal links and more links between the subnets.



Modularity

The modularity of a partition in an undirected, unweighted network:

$$Q = \frac{1}{L} \sum_C \left(L_C - \frac{k_C^2}{4L} \right)$$

L_C is the number of internal links in cluster C, k_C is the total degree of nodes in C.

k_C (total num stubs attached to nodes in C) stays constant in each randomization, by construction.

The probability of selecting one of these stubs at random is: $k_C/2L$

The probability of picking a pair of stubs from C at random is: $\frac{k_C}{2L} \cdot \frac{k_C}{2L} = \frac{k_C^2}{4L^2}$

Modularity

The modularity of a partition in an undirected, unweighted network:

$$Q = \frac{1}{L} \sum_c \left(L_c - \frac{k_c^2}{4L} \right)$$

What happens when there is a single cluster?

Can $Q \geq 1$?

Can $Q < 0$?

Modularity

The modularity of a partition in an undirected, unweighted network:

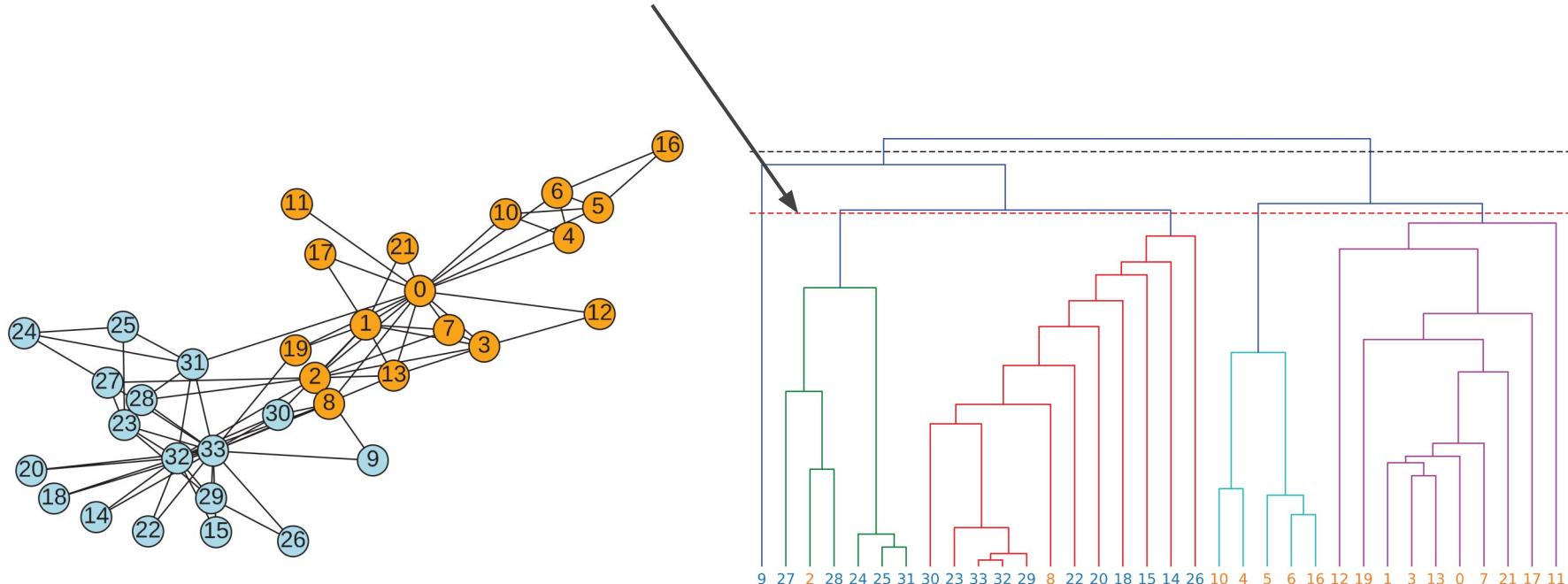
$$Q = \frac{1}{L} \sum_C \left(L_C - \frac{k_C^2}{4L} \right)$$

What happens when there is a single cluster? $\rightarrow Q = 0$

Can $Q \geq 1$? \rightarrow No

Can $Q < 0$? \rightarrow Yes (partition into N singletons)

Largest modularity value: five clusters



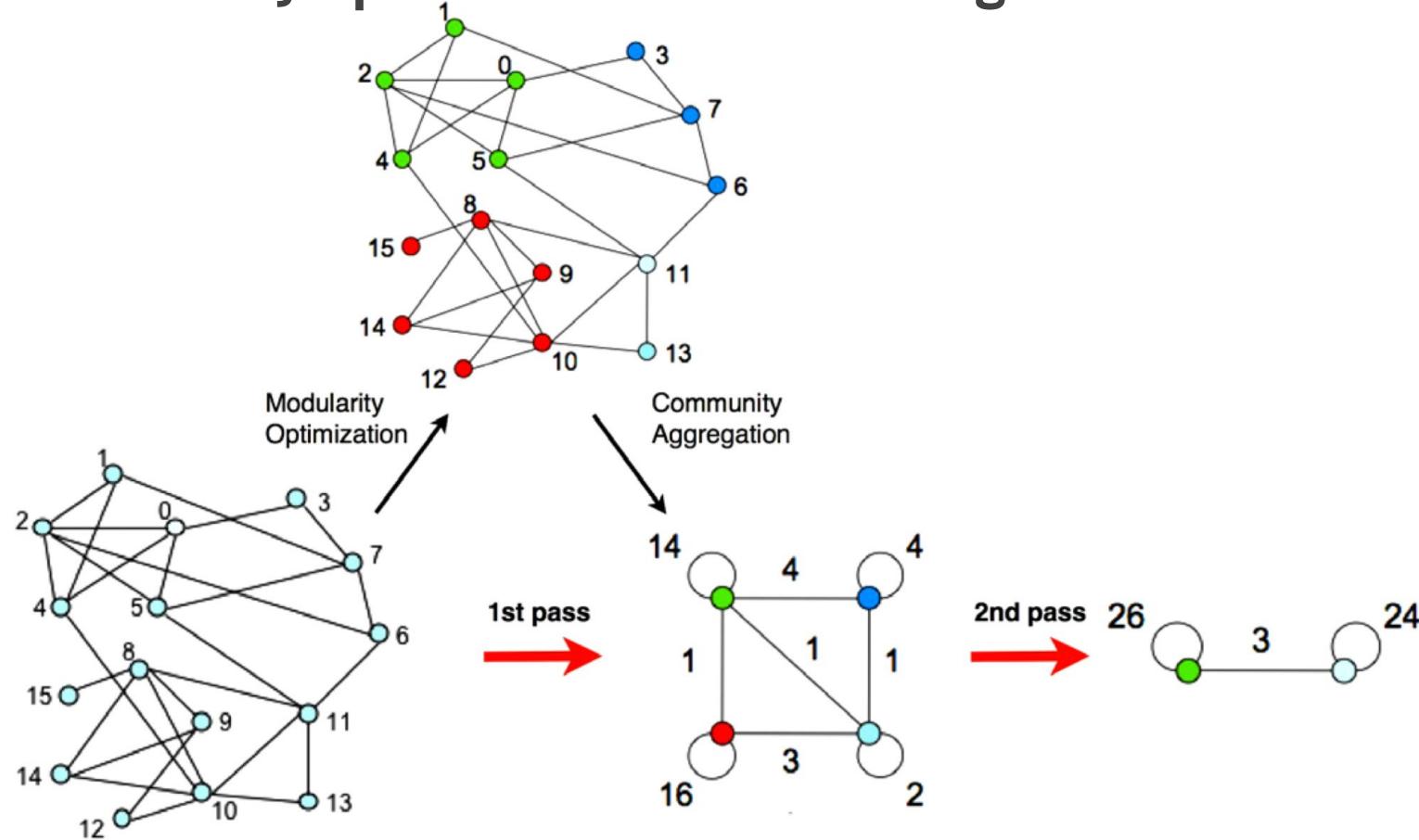
3.2. Modularity optimization - Louvain algorithm

The algorithm starts again from the partition into singletons. Each iteration consists of two steps.

1. Loop over the nodes: each node is put into the community of the neighbor that yields the largest modularity increase ΔQ with respect to the current partition. All nodes are revisited over and over until it is no longer possible to increase Q by moving a node to a different community.
2. The network is transformed into a weighted supernetwork, where each community in the partition from step 1 is replaced by a supernode, links between supernodes carry a weight corresponding to the number of links joining nodes in the corresponding groups, and links joining nodes in the same community are represented as a self-loop from the corresponding supernode to itself, with weight equal to the number of internal links.

Since we ultimately care about clustering the actual nodes and not the supernodes, modularity is always computed with respect to the original network. The procedure stops when no further grouping of the clusters in the current partition increases the modularity, and returns the partition with the largest modularity.

3.2. Modularity optimization - Louvain algorithm



Still greedy. And result depends on the order in which the nodes are visited. But fast → very commonly used in practice.

More later

3.3. Label propagation

3.4. Stochastic block modeling

Summary

Communities play a key role in the structure and function of networks.

But communities are not well-defined objects.

Network partitioning searches for well-separated subnetworks.

Hierarchical clustering groups nodes based on their similarity. Biggest drawback: lack of criterion for selecting meaningful partitions.

Bridge removal (same drawback).

Modularity optimization (Louvain) widely used in practice.