Community Detection

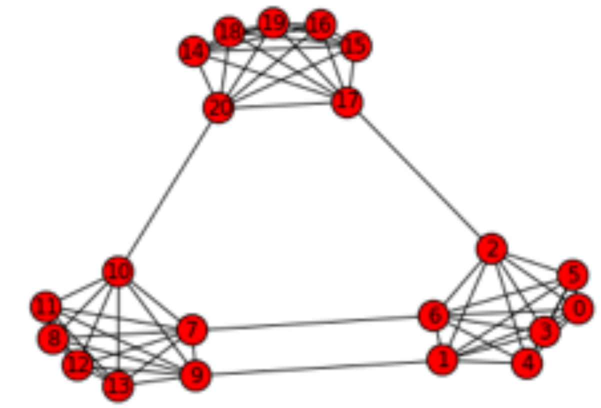
Rik Sarkar

Communities

- Groups of friends
- Colleagues/collaborators
- Web pages on similar topics
- Biological reaction groups
- Similar customers/users …

Other applications

- A coarser representation of networks
  - One or more meta-node for each community
- Identify bridges/weak-links
- Structural holes



Different definitions of communities

- General idea: **Dense subgraphs:** More links within community, few links outside

- Some types and considerations:

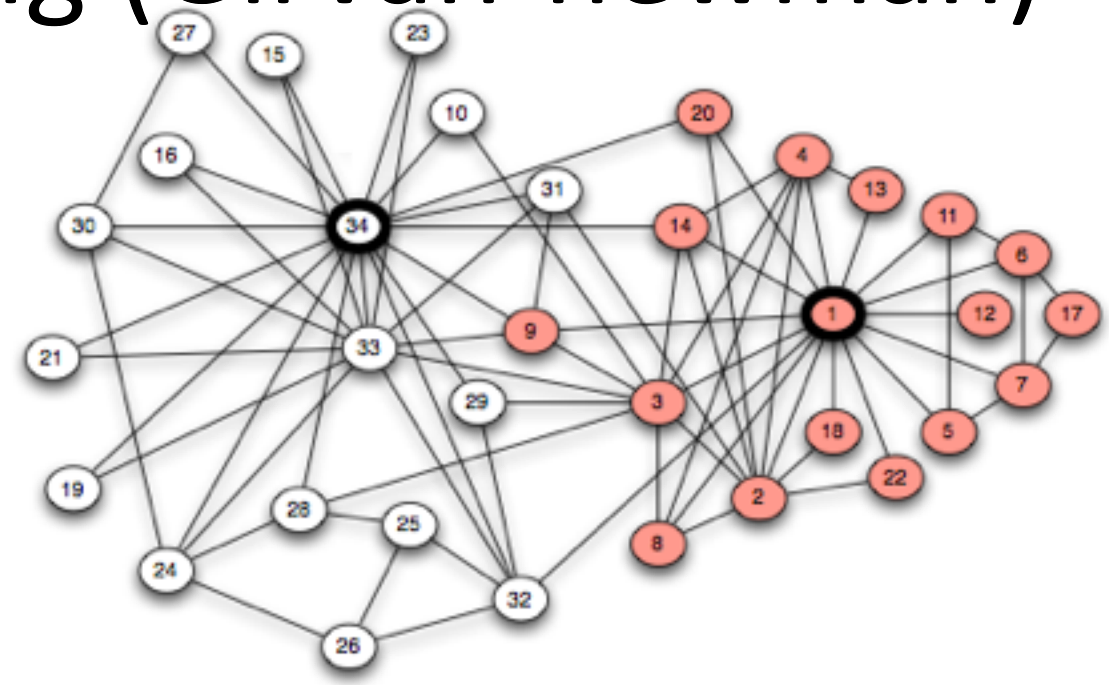
1. Partitions: Each node in exactly one community

2. Overlapping: Each node can be in multiple communities

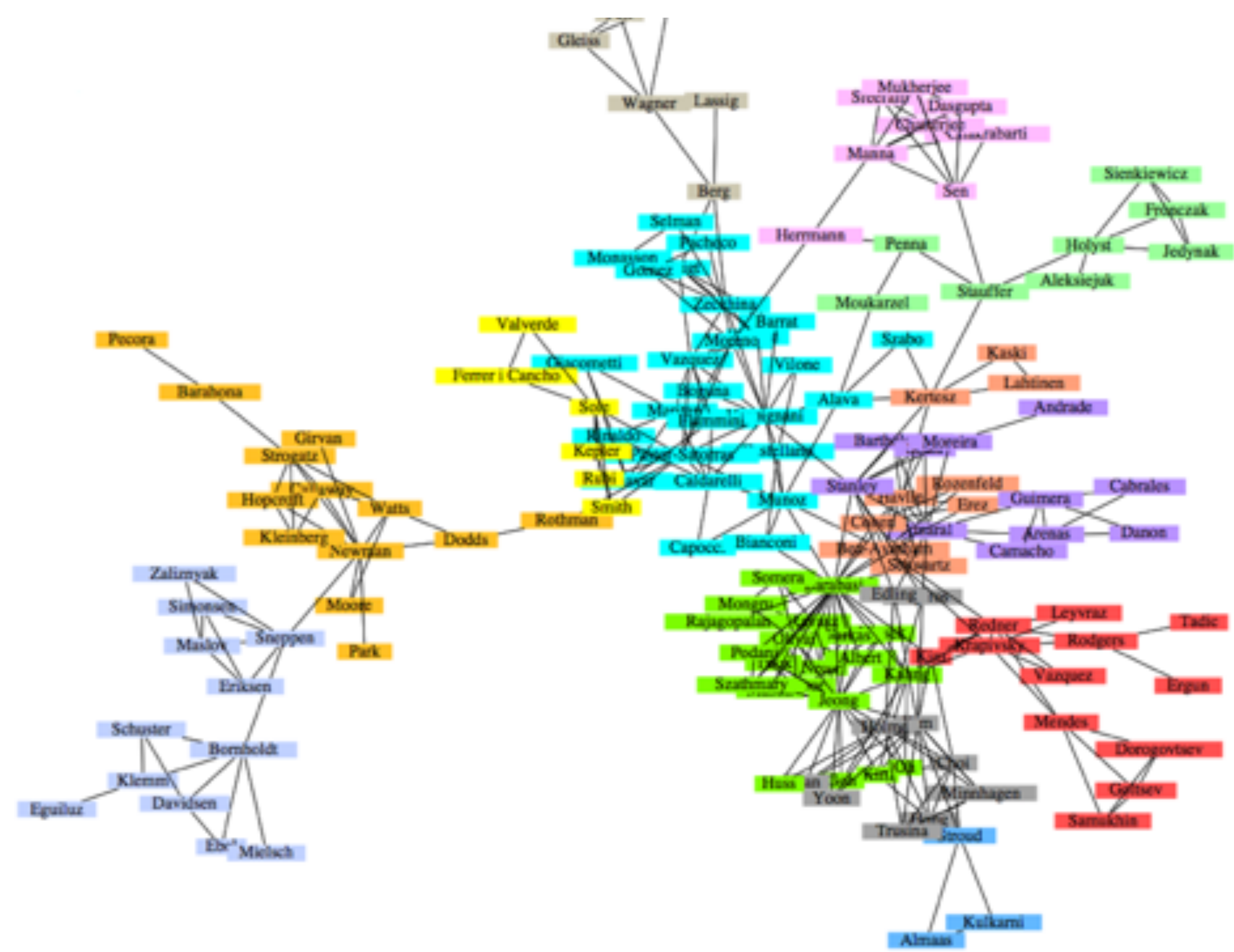
From last class: Partitioning (Girvan-newman)

Repeat:

- Find edge e of highest betweenness
- Remove e



- Produces a hierarchic paritioning structure as the graph decomposes into smaller components



Finding dense subgraphs is hard in general

- Finding largest clique
  - NP-hard
  - Computationally intractable
  - Polynomial time (efficient) algorithms unlikely to exist
- Decision version: Does a clique of size k exist?
  - NP-complete
  - Computationally intractable
  - Polynomial time (efficient) algorithms unlikely to exist

Few preliminary definitions

- For S, T subgraphs of V
- e(S,T): Set of edges from S to T
  - e(S) = e(S,S): Edges within S
- $d_S(v)$: number of edges from v to S
- Edge density of S : |e(S)|/|S|
  - Largest for complete graphs or cliques

Dense subgraph

- The subgraph with largest edge density
  - There also exists a decision version:
    - Is there a subgraph with edge density > α
- Can be solved using Max Flow algorithms
    - $O(n^2m)$: inefficient in large datasets
    - Finds the one densest subgraph
- Variant: Find densest S containing given subset X
- Other versions: Find subgraphs size k or less
  - NP-hard

Efficient approximation for finding dense S containing X

Let $G_n \leftarrow G$.
for $k = n$ downto $|X| + 1$ **do**
  Let $v \notin X$ be the lowest degree node in $G_k \setminus X$.
  Let $G_{k-1} \leftarrow G_k \setminus \{v\}$.
Output the densest subgraph among $G_n, \ldots, G_{|X|}$.

- Gives a 1/2 approximation
  - Edge density of output S set is at least half of optimal set S*
  - See Kempe 2011 for proof.

Modularity

- We want to find the many communities, not just one
- Clustering a graph
- Problem: What is the right clustering?
- Idea: Maximize a quantity called *modularity*

Modularity of subset S

- Given graph G
- Consider a random G' graph with same node degrees (remember configuration model)
  - Number of edges in S in G: $|e(S)|_G$
  - Expected number of edges in S in G': $E[|e(S)|_{G'}]$
  - Modularity of S: $|e(S)| - E[|e(S)|_{G'}]$
  - More coherent communities have more edges inside than would be expected in a random graph with same degrees
  - Note: modularity can be negative

Modularity of a clustering

- Take a partition (clustering) of V: $\mathcal{P} = \{S_1, \ldots, S_k\}$
- Write $d(S_i)$ for sum of degrees of all nodes in S_i
- Can be shown that $E[|e(S)|_{G'}] \sim d(S_i)^2$
- Definition: Sum over the partition:

$$q(\mathcal{P}) \quad = \quad \frac{1}{m} \sum_i |e(S_i)| - \frac{1}{4m} d(S_i)^2$$

Modularity based clustering

- Finding clustering with highest modularity is NP-hard
- Heuristic:
  - Use modularity matrix
  - Take its first eigen vector
- Note: Modularity is a relative measure of community structure.
- Not entirely clear in which cases it may or may not give good results
- A threshold of 0.3 or more is sometimes considered to give good clustering

Modularity

- Can be used as a stopping criterion (or finding right level of partitioning) in other methods
    - Eg. Girvan-newman

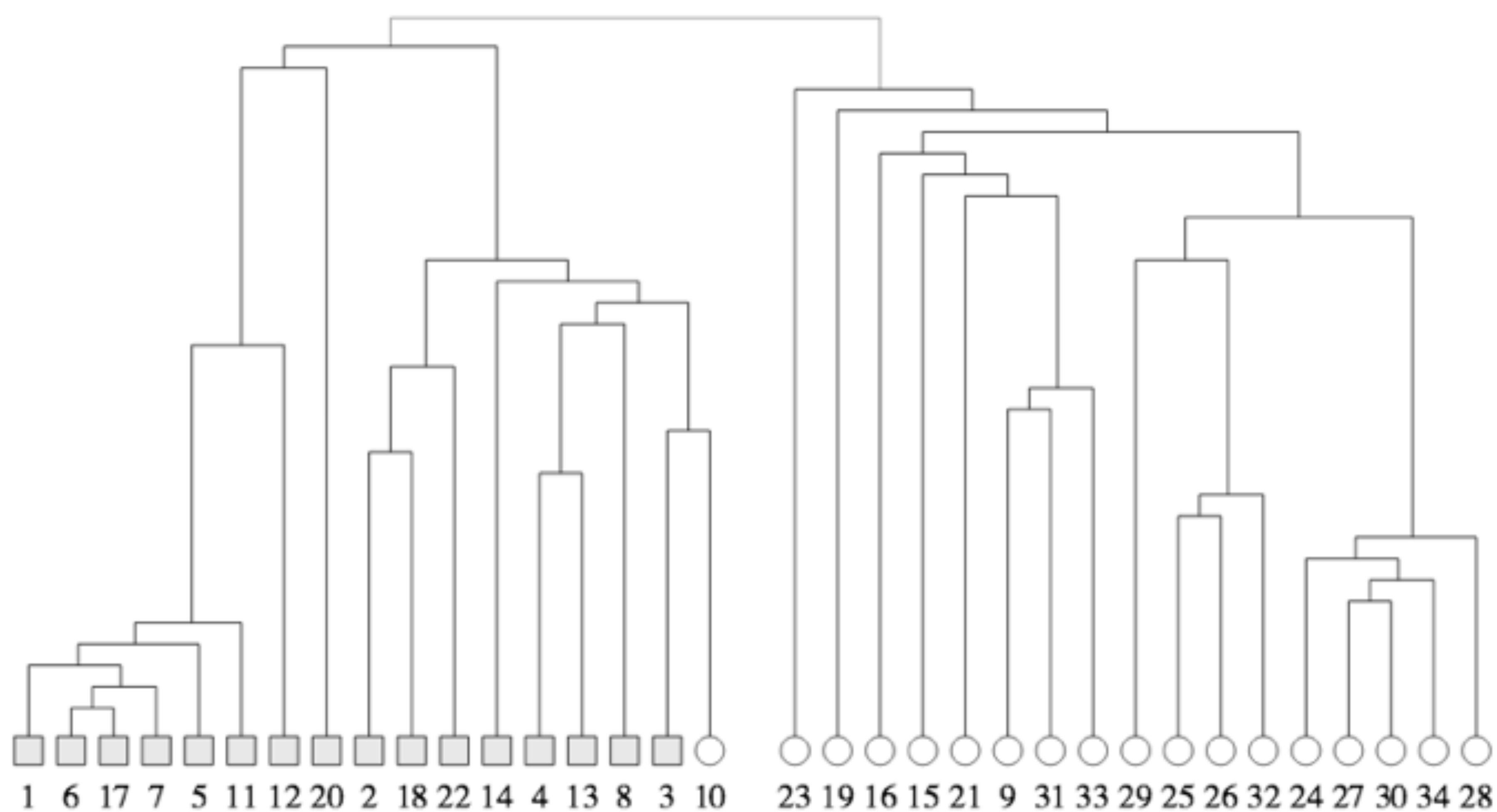
Faster modularity clustering

- Start with all nodes as their own community and proceed by merging
- In every round,
  - Consider merging every pair of current communities
  - Merge the pair giving largest Δq : increase in modularity
  - Keep store modularity after each round
- Take the set of clusters in round with max modularity
- $O((m+n)n)$
- General technique for hierarchic clustering, except using modularity

Karate club hierarchic clustering

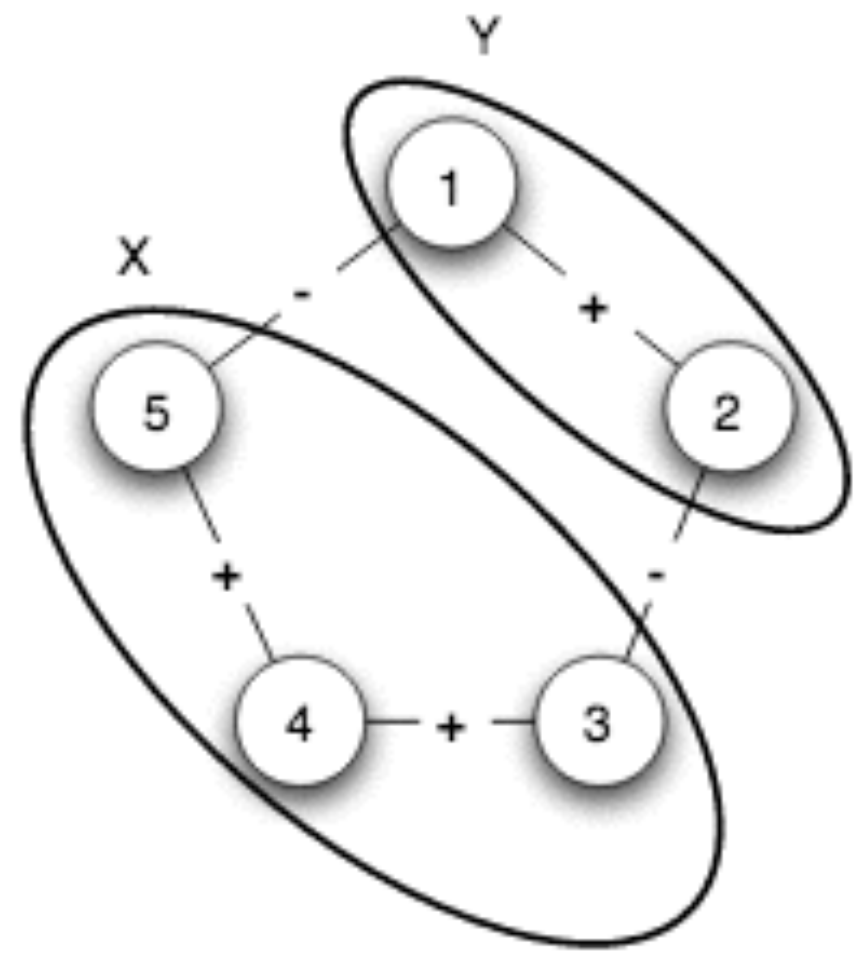
- Shape of nodes gives actual split in the club due to internal conflicts

Newman 2003. Fast algorithm for detecting community structure in networks



Correlation clustering

- Some edges are known to be similar/ friends/trusted
- marked "+"
- Some edges are known to be dissimilar/enemies/distrusted
- marked "-"
- Maximize the number of + edges inside clusters and
- Maximize the number of - edges between clusters



Applications

- Community detection based on similar people/ users
- Document clustering based on known similarity or dissimilarity between documents

Features

- Clustering without need to know number of clusters
  - k-means, medians, clusters etc need to know number of clusters or other parameters like threshold
  - Number of clusters depends on network structure
- Actually, does not need any parameter
- NP hard
- Note that graph may be complete or not complete
  - In some applications with unlabeled edges, it may be reasonable to change edges to “+” edges and non-edges to “-” edges

Approximation

- Naive 1/2 approximation (not very useful):
  - If there are more + edges
    - Put them all in 1 cluster
  - If there are more - edges
    - Put nodes in n different clusters

Better approximations

- 2 ways of looking at it:
  - Maximize agreement or Minimize disagreement
  - Same idea, but we know different approximation algorithms
- Nikhil Bansal et al. develop PTAS (polynomial time approximation scheme) for maximizing agreement:
  - (1-ε) approximation, running time $O(n^2 \mathrm{e}^{O(1/\epsilon)})$

Approximation

- Min-disagree:
  - 4-approximation

Projects

- Some people are looking for teammates (P1: lastfm, P5: Entropy, others..)
- Please post and use the piazza forum to find teammates

Next

- (Possibly) A bit more on clustering
- Diffusion, Spread of epidemics, cascades, finding influential nodes
- Other suggestions?