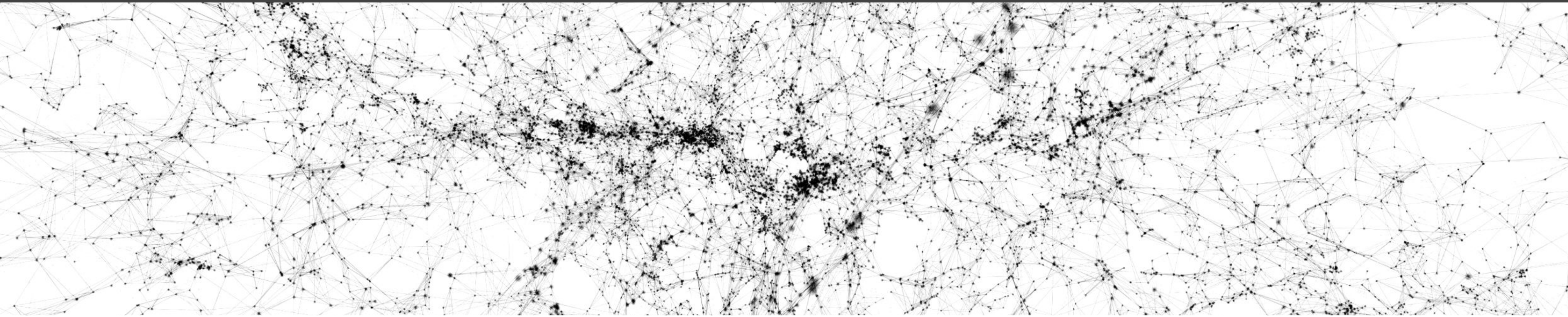
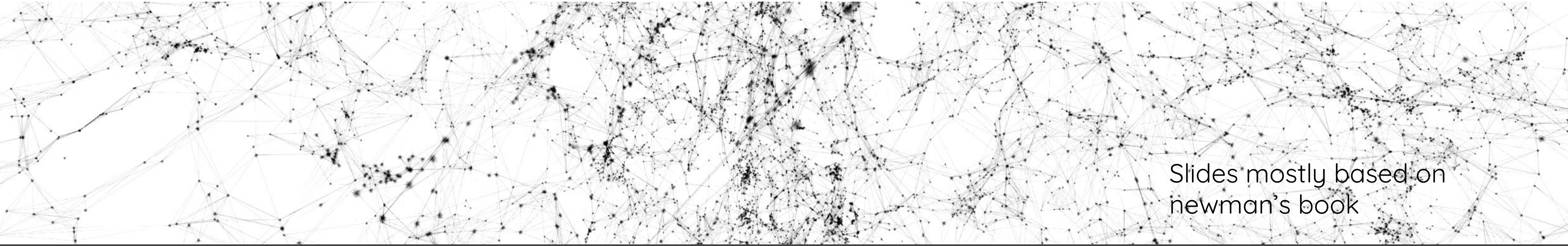


Modules

Analysis of complex interconnected data



Slides mostly based on newman's book

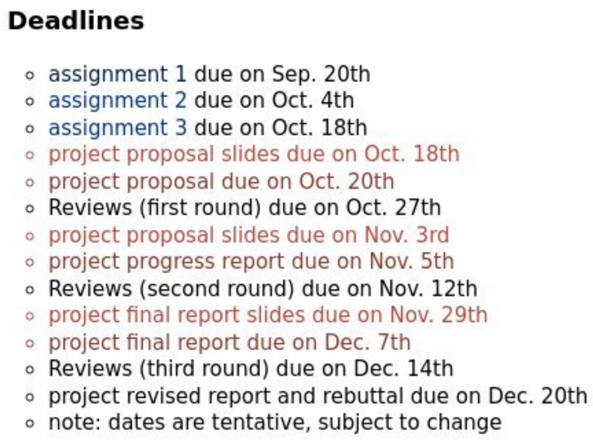
Outline

- Quick Notes
- Quick Recap of Centrality Measures
- Modules
  - Real graphs are modular
  - Spectral clustering
  - Objectives for quality of a module
  - TopLeaders
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - Resolution limits of Modularity
  - Link clustering
  - Evaluating clustering results

Quick Notes

- Second assignment is out, due on Oct 4th
  - http://www.reirab.com/Teaching/NS21/Assignment_2.pdf
  - Submit single entry as a Group (triad, dyad, or individual) in Mycourses

- Use slack for easier communications
- Let's have a designated question helper
- Any questions/suggestion?



Deadlines

- assignment 1 due on Sep. 20th
- assignment 2 due on Oct. 4th
- assignment 3 due on Oct. 18th
- project proposal slides due on Oct. 18th
- project proposal due on Oct. 20th
- Reviews (first round) due on Oct. 27th
- project proposal slides due on Nov. 3rd
- project progress report due on Nov. 5th
- Reviews (second round) due on Nov. 12th
- project final report slides due on Nov. 29th
- project final report due on Dec. 7th
- Reviews (third round) due on Dec. 14th
- project revised report and rebuttal due on Dec. 20th
- note: dates are tentative, subject to change

Quick Recap of Centrality Measures

$$x_i = \sum_{j\in N(i)} 1$$

$$x_i = c\sum_{j\in N(i)} x_j, \quad c = \frac{1}{\lambda^*(A)}$$

$$x_i = c\sum_{j\in N(i)} x_j + 1, \quad c < \frac{1}{\lambda^*(A)}$$

$$x_i = c\sum_{j\in N(i)} \frac{x_j}{\sum_k x_{kj}} + 1, \quad c < \frac{1}{\lambda^*(AD^{-1})}$$

$$x_i = c\sum_{j\in N(i)} y_i, \quad y_i = c'\sum_{j\in N(i)} x_i, \quad cc' = \frac{1}{\lambda^*(AA^T)}$$

$$x_i = \frac{1}{n-1}\sum_j \frac{1}{s_{ij}}, \quad s_{ij}\text{: length of shortest path from } i \text{ to } j$$

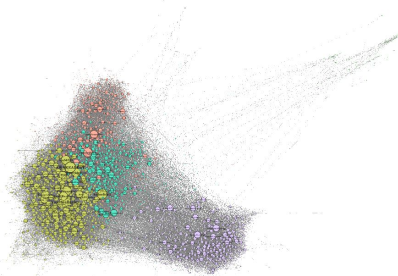
$$x_i = \frac{1}{n^2}\sum_{jk} \frac{|i\in s_{jk}|}{|s_{jk}|} \quad s_{ij}\text{: set of shortest paths from } i \text{ to } j$$

$N(i) = \{j|A_{ij} = 1\}, \quad \lambda^*(A)$: largest eigenvalue of A

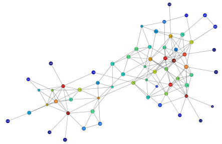
- Degree Centrality
  - count the number of neighbours, ignores their importance
- Eigenvalue Centrality
  - consider importance of connections, gives zero to nodes not in scc or its out component, in extreme case of an acyclic networks, e.g. citation networks, all nodes get zero score
- Katz Centrality
  - avoid zeros by giving everyone a basic importance
- PageRank
  - divide importance on how many connections it is passed over to
- HITS
  - consider two types of importance, hubs and authorities
- Closeness centrality
  - average how close you are to the rest
- Betweenness centrality
  - count what fraction of shortest paths pass through you

Outline

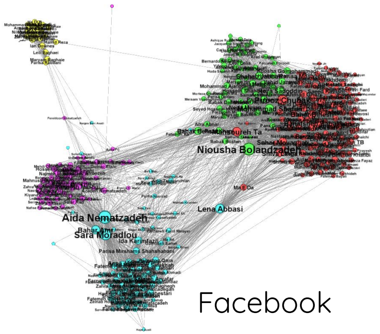
- Quick Notes
- Quick Recap of Centrality Measures
- **Modules**
    - Real graphs are modular
    - Spectral clustering
    - Objectives for quality of a module
    - TopLeaders
    - Using Betweenness Centrality
    - Modularity Optimization, FastModularity & Louvain
    - Resolution limits of Modularity
    - Link clustering
    - Evaluating clustering results



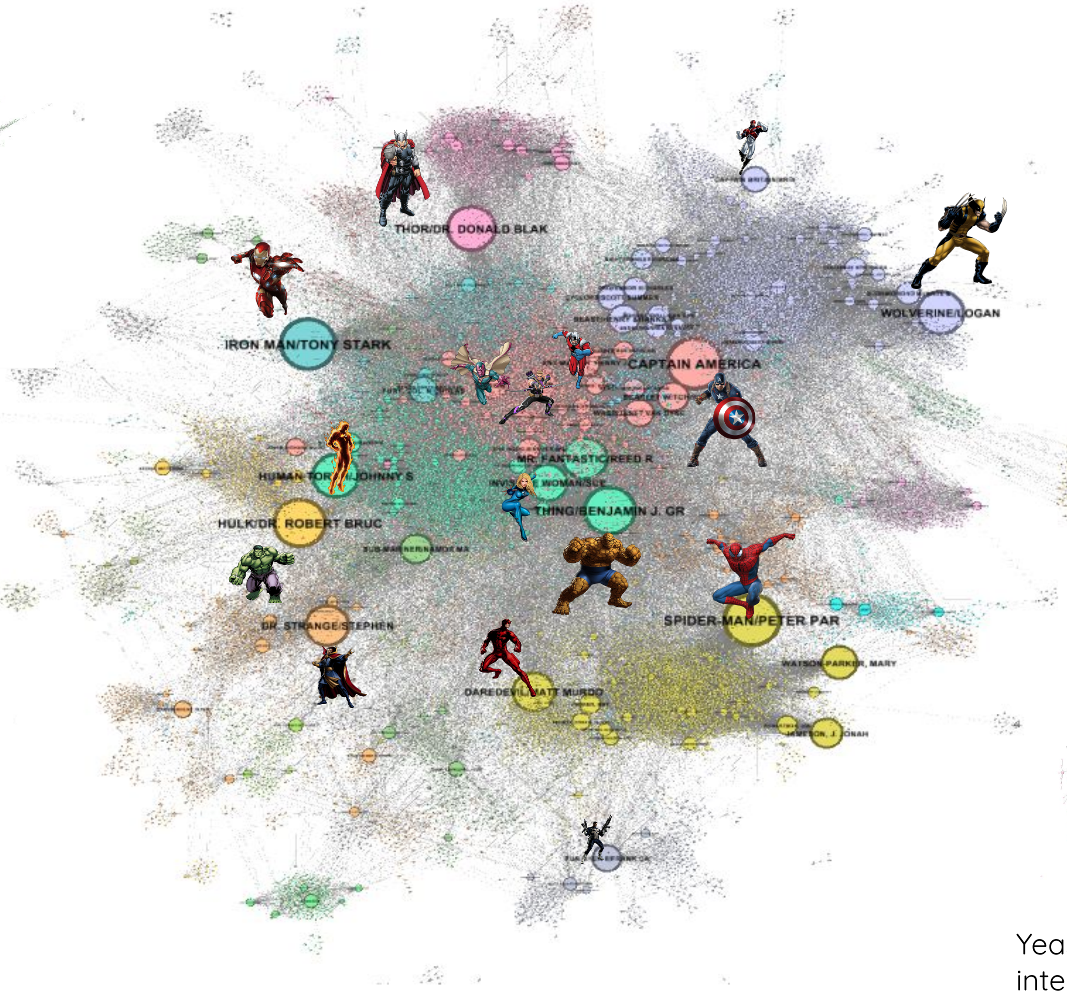
Twitter

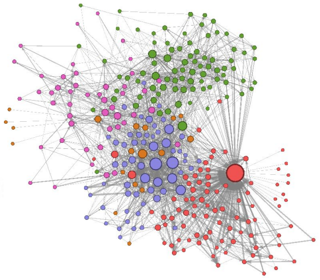


Dolphins

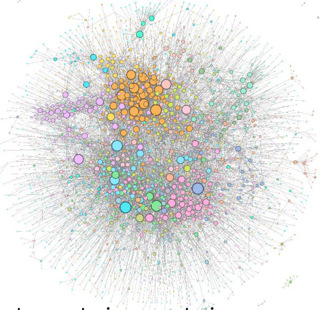


Facebook



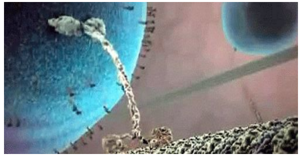


C. elegans neural network



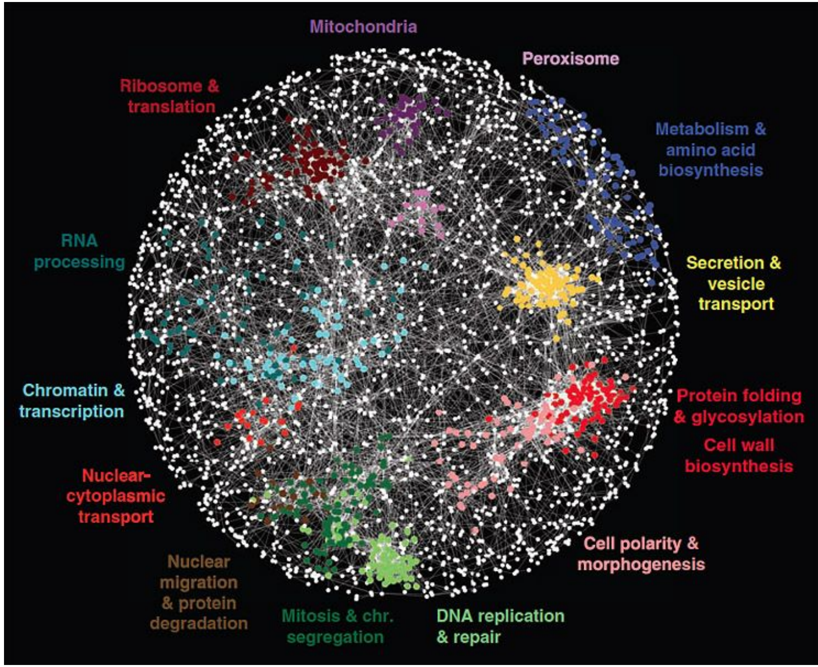
Yeast protein protein interaction networks

Example Applications



Module identification in biological networks

- Protein complexes and functional modules in PPI networks (Spirin & Mirny, PNAS 2003)
  - protein complexes: proteins that interact to carry out a task as a single complex unit, e.g., RNA splicing
  - functional units: proteins that bind at different time to participate in a cellular process, e.g., communicating a signal from the surface of the cell to the nucleus
- Representation of the metabolic networks (R Guimerà & Amaral, Nature 2005)
  - ultra-peripheral metabolites (that have all their connections inside their modules) have the highest evolutionary loss rate, whereas connector hubs (that connect to most of the other modules) are the most conserved across the species

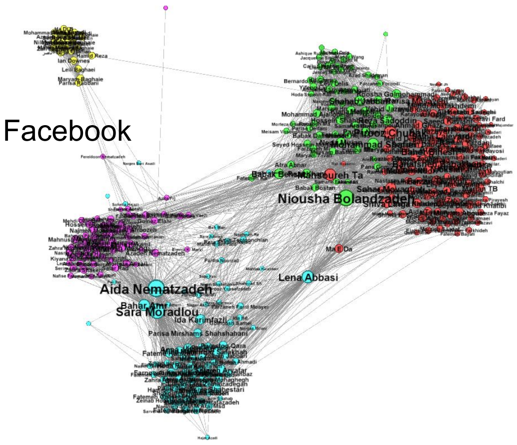


Mitochondria
Peroxisome
Ribosome & translation
Metabolism & amino acid biosynthesis
RNA processing
Secretion & vesicle transport
Chromatin & transcription
Protein folding & glycosylation
Cell wall biosynthesis
Nuclear-cytoplasmic transport
Cell polarity & morphogenesis
Nuclear migration & protein degradation
Mitosis & chr. segregation
DNA replication & repair

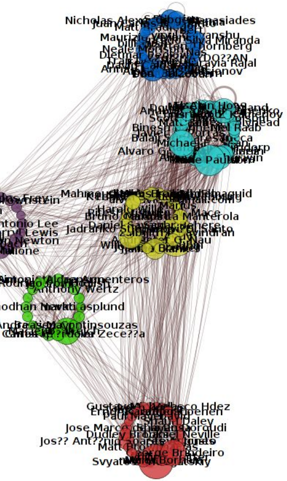
Modules as Coarse Representation

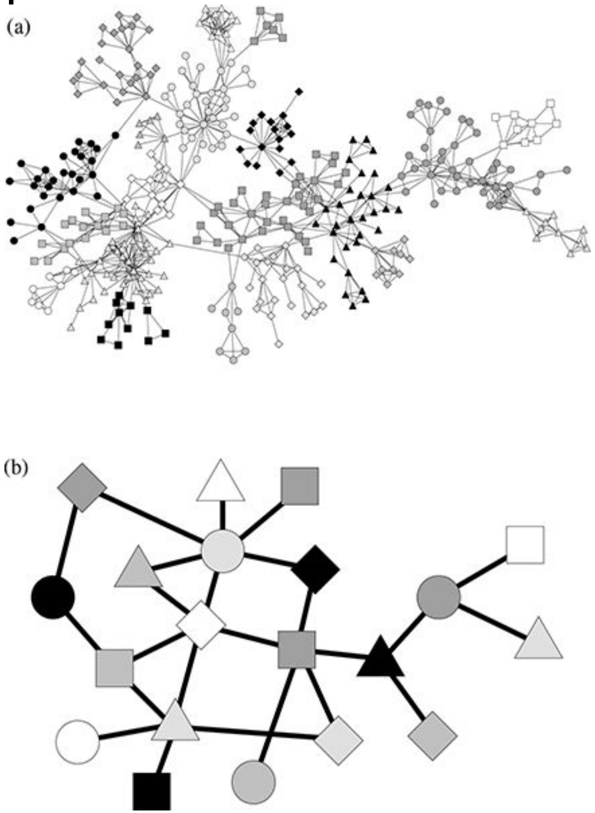
Modules give a coarse-grained representation of the structure

Also referred to as meso-scale, cluster, communities, etc.


Facebook

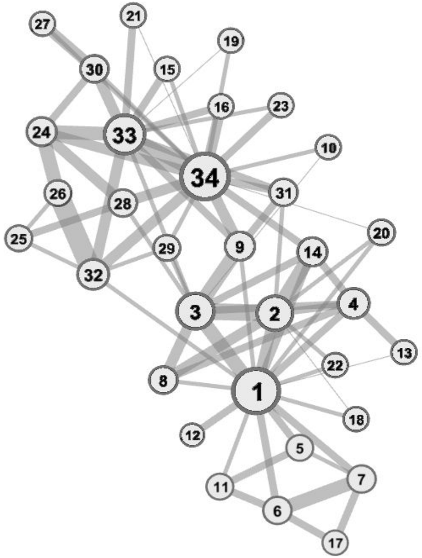
Students



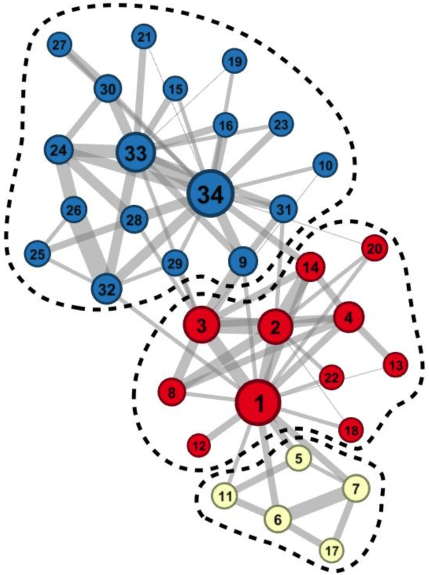


Clustering a.k.a Community Detection

Given a graph, how to cluster the nodes into modules?



algorithm



Outline

- Quick Notes
- Quick Recap of Centrality Measures
- Modules
    - Real graphs are modular
    - **Spectral clustering**
    - Objectives for quality of a module
    - TopLeaders
    - Using Betweenness Centrality
    - Modularity Optimization, FastModularity & Louvain
    - Resolution limits of Modularity
    - Link clustering
    - Evaluating clustering results

Spectral clustering

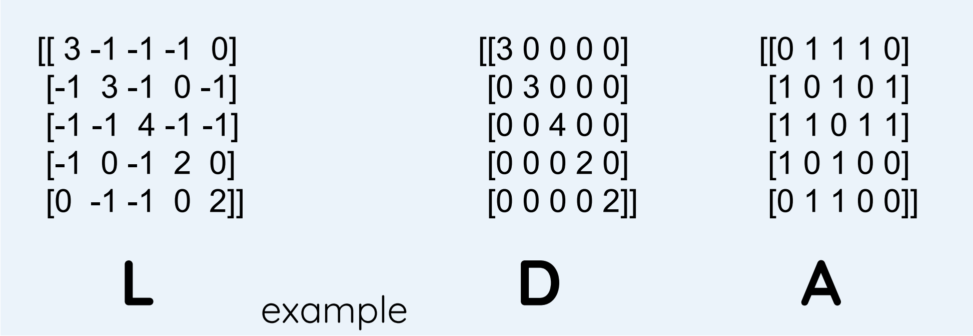
Uses the relation between connectivity & Laplacian matrix

Recall:

Laplacian Matrix: **L = D - A**

A: adjacency matrix

D: diagonal matrix of degrees



```
[[ 3 -1 -1 -1  0]
 [-1  3 -1  0 -1]
 [-1 -1  4 -1 -1]
 [-1  0 -1  2  0]
 [0  -1 -1  0  2]]
```

L

```
[[3 0 0 0 0]
 [0 3 0 0 0]
 [0 0 4 0 0]
 [0 0 0 2 0]
 [0 0 0 0 2]]
```

D

```
[[0 1 1 1 0]
 [1 0 1 0 1]
 [1 1 0 1 1]
 [1 0 1 0 0]
 [0 1 1 0 0]]
```

A

example

L is symmetric & positive-semidefinite

Spectral clustering

Uses the relation between connectivity & Laplacian matrix

- **$Lu = \lambda u$:** Eigenvalues of Laplacian Matrix
- We have n eigenvalues which we call **Laplacian Spectrum**:

  $$0 = \lambda_0 \leq \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$$

- λ_0 is always zero since we have **$L(1,1 \ldots 1) = 0$** : why?
- $\lambda_0 = \lambda_1 \ldots = \lambda_k = 0 \Rightarrow k$ is number of connected components
- Largest is bounded by twice the maximum degree in G
- $E = ½ \Sigma d_i = ½ \, Tr(L) = ½ \Sigma \lambda_i$
- Spectral gap: smallest nonzero eigenvalue
- Fiedler vector: eigenvector corresponding to the spectral gap
- Spectral ordering: Fiedler vector sorted
- Laplacian Spectrum relates to graph connectivity & clustering

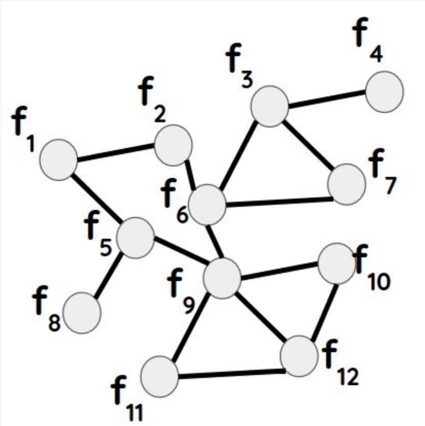
Laplacian Matrix & Smoothness

- **$f = (f_1, \ldots, f_n)$** function on Graph
  - $\mathbf{f \in R^n \Rightarrow f^T L f = \frac{1}{2} \Sigma_{ij} A_{ij}(f_i - f_j)^2}$

Measures how much the value of f is smooth over edges, i.e. the difference of values for connecting nodes

How to find modules?

Consider function **f** that maps vertices to a value



$$\mathbf{f^T L f = f^T D f - f^T A f = \Sigma_i d_i f_i^2 - \Sigma_{ij} f_i f_j A_{ij}}$$

$$\mathbf{= \frac{1}{2} [\Sigma_i d_i f_i^2 - 2 \Sigma_{ij} f_i f_j A_{ij} + \Sigma_i d_i f_i^2]}$$

$$\mathbf{= \frac{1}{2} \Sigma_{ij} A_{ij}(f_i - f_j)^2}$$

See this for more details.

Laplacian Matrix & Smoothness

- $\mathbf{f} = (\mathbf{f}_1, \ldots, \mathbf{f}_n)$ function on Graph
  - $f \in R^n \Rightarrow f^T L f = ½ \Sigma_{ij} A_{ij}(f_i - f_j)^2$

Measures how much the value of f is smooth over edges, i.e. the difference of values for connecting nodes

How to find modules? Find f that give smoothest results, i.e, minimizes this

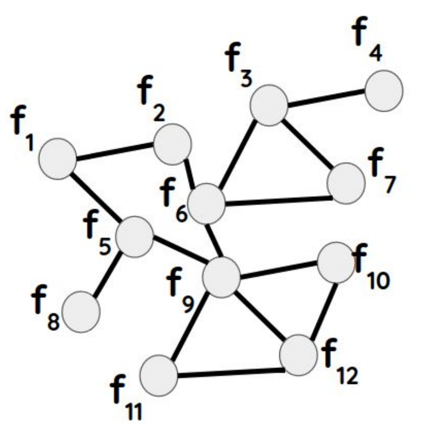
$$f^T L f = f^T D f - f^T A f = \Sigma_i d_i f_i^2 - \Sigma_{ij}\ f_i f_j A_{ij}$$

$$= ½ [\Sigma_i d_i f_i^2 - 2\ \Sigma_{ij}\ f_i\, f_j A_{ij} + \Sigma_i d_i f_i^2]$$

$$= ½ \Sigma_{ij} A_{ij}(f_i - f_j)^2$$

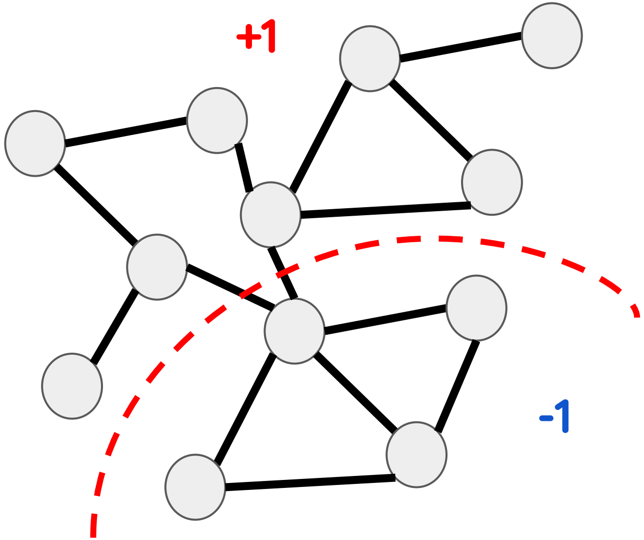
See this for more details.

Consider function **f** that maps vertices to a value

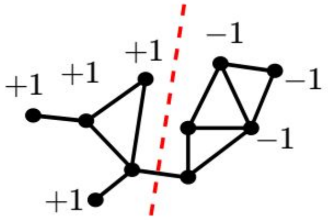


Spectral Clustering

- **f = (f_1, …, f_n)** function on Graph
  - $f \in R^n \Rightarrow f^T L f = ½ \Sigma_{ij} A_{ij} (f_i - f_j)^2$
- Cut edges = **¼ $x^T L$ x**, why?
- Find best balanced cut
  - Minimize given $x_i \in \{+1,-1\}$ & $\Sigma_i x_i = 0$
  - What does it mean?



x_i = +1 if i ∈ 1 else -1



Spectral Clustering

- $f = (f_1, \ldots, f_n)$ function on Graph
  - $f \in R^n \Rightarrow f^T L f = \frac{1}{2} \Sigma_{ij} A_{ij} (f_i - f_j)^2$

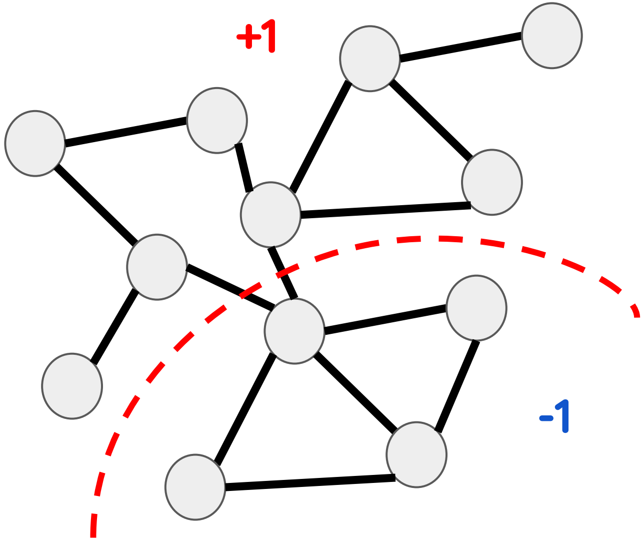
- Cut edges = $\frac{1}{4}\, x^T L\, x$
- Find best balanced cut
  - Minimize given $x_i \in \{+1,-1\}$ & $\Sigma_i x_i = 0$
  - What does it mean? Balanced partitions

- $x_i \in \{+1,-1\} \Rightarrow x_i \in R$ & $\Sigma_i x_i^2 = n$ $(x^T x = n)$
  - $\text{Min } \frac{1}{4}\, x^T L\, x = \frac{1}{4}\, n\, v_1^T\, L\, v_1 = \frac{1}{4}\, n\, \lambda_1$

Courant Fischer Minimax Theorem

- Second smallest **eigenvalue**
  ⇒ sparsest ratio cut

- Signs of corresponding **eigenvector**



$x_i = +1 \text{ if } i \in 1 \text{ else } -1$

Normalized Graph Laplacian

- **Symmetric normalization**
  - $L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2}$
- **Random walk normalization**
  - $L_{rw} = D^{-1} L = I - D^{-1} A$

- $\mathbf{L_{sym}}$ and $\mathbf{L_{rw}}$ are positive semi-definite and have n non-negative real-valued eigenvalues

- Multiplicity of zero eigenvalues in both still gives the number of connected components

Further reading? See this

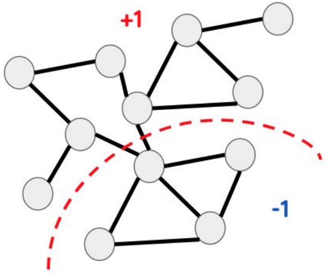
Spectral Clustering with Normalized Graph Laplacian

- **Symmetric normalization** used for spectral clustering by **Ng, Jordan, and Weiss** (2002)
- **Random walk normalization** used for spectral clustering by **Shi and Malik** (2000)
- Spectral clustering with unnormalized Laplacian optimizes the **RatioCut**

$$\mathrm{RatioCut}(A_1,\ldots,A_k) = \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i,\overline{A}_i)}{|A_i|}$$

- Spectral clustering with normalized Laplacian optimizes **Normalized cut (Ncut)**

$$\mathrm{Ncut}(A_1,\ldots,A_k) = \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i,\overline{A}_i)}{\mathrm{vol}(A_i)}. \quad \mathrm{vol}(A) := \sum_{i\in A} d_i$$

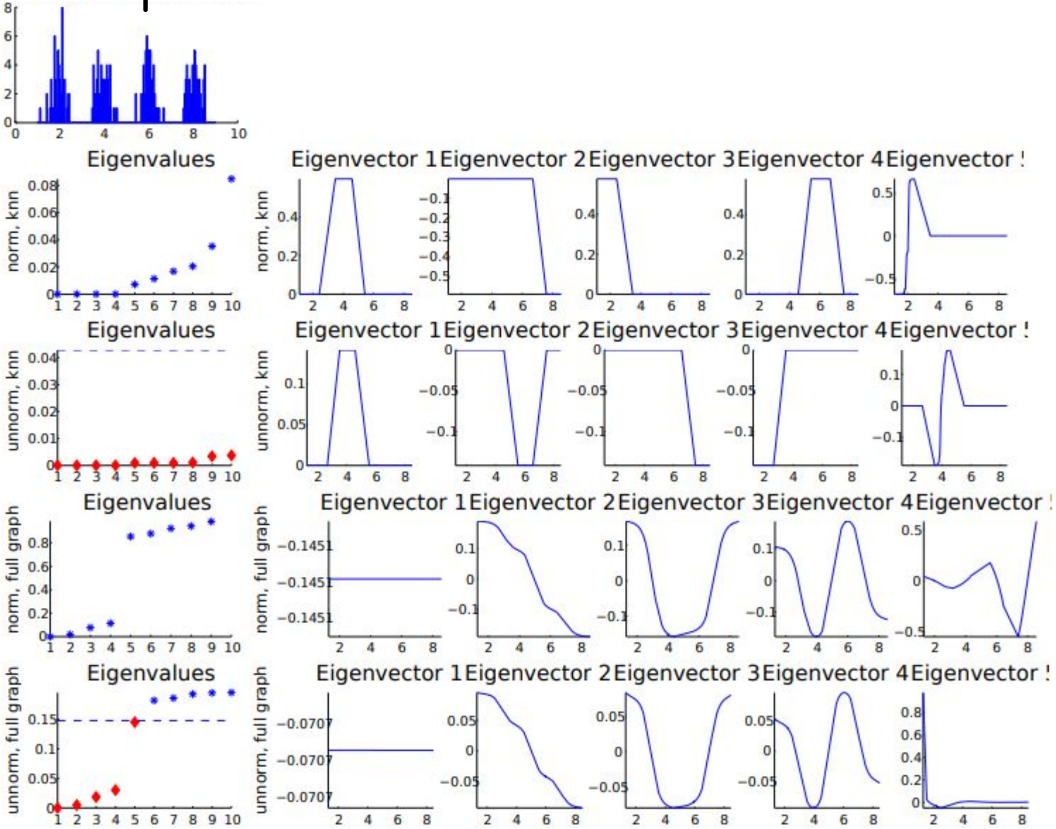


x_i = +1 if i ∈ 1 else -1

Cut = 2
RatioCut = 2 / 4
NCut = 2 (1/12 + 1/18)

Further reading?
See this

Example



a random sample of 200 points drawn according to a mixture of four Gaussians: $x_1, \ldots, x_{200} \in R \Rightarrow$ similarity graph with knn or complete graph

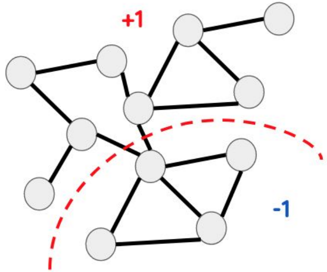
Knn: the first four eigenvalues are 0, and the corresponding eigenvectors are cluster indicator vectors since clusters form disconnected parts in the k-nearest neighbor graph

fully connected graph: the first eigenvector is the constant vector. The following eigenvectors carry the information about the clusters.

Further reading? See this

Normalized Graph Laplacian

- **Symmetric normalization** {used for spectral clustering by Ng, Jordan, and Weiss (2002)}
  - $L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2}$
- **Random walk normalization** {used for spectral clustering by Shi and Malik (2000)}
  - $L_{rw} = D^{-1} L = I - D^{-1} A$
  - ⇒ **Normalized cut (Ncut),** by the number of **edges** in the clusters



x_i = +1 if i ∈ 1 else -1

K Clusters? Use k-means on first (nontrivial) k eigenvectors

(each node is represented with k features)

Many successful applications including image segmentation but not the best choice for finding modules in real world graph.

How can we define a better objective for finding modules in real world graph ?

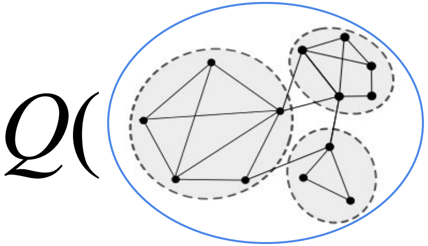
Further reading? See this

Outline

- Quick Notes
- Quick Recap of Centrality Measures
- Modules
  - Real graphs are modular
  - Spectral clustering
  - **Objectives for quality of a module**
  - TopLeaders
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - Resolution limits of Modularity
  - Link clustering
  - Evaluating clustering results

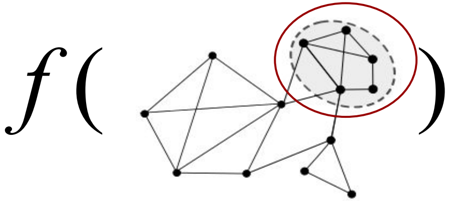
Objectives for quality of a community

We can define the community detection either **globally** or **locally** and have global or local algorithms. Local algorithms are the choice when really with graphs that do not fit in memory.



Globally-defined quality function to partition the whole network

Gives sets of sets, set of all clusters, usually disjoint and covering the full data

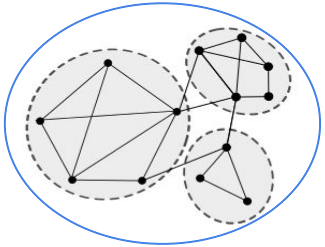


Locally defined quality function for one subset of nodes in a network

Gives one set of nodes belonging to the same cluster

Objectives for quality of a community

$Q($



C: sets of sets, set of all clusters, usually disjoint and covering the full data

Globally-defined quality function to partition the whole network

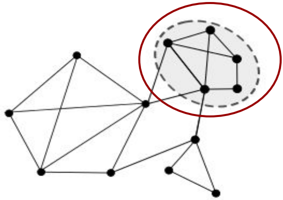
- **Q-modularity** (Newman 2003)

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j).$$

m = total edges in the graph
i, j: node indexes
k_i : degree of nodes i
C_i : cluster index that node i belongs to

Most common representatives

$f($



$)$

S: a set of nodes in one cluster

Locally defined quality function for one subset of nodes in a network

- **Conductance** (Sinclair & Jerrum 1989)

$$f(S) = \frac{c_S}{2m_S + c_S}$$

- Normalized Cut (Shi & Malik 2000)

$$f(S) = \frac{c_S}{2m_S + c_S} + \frac{c_S}{2(m - m_S) - c_S}$$

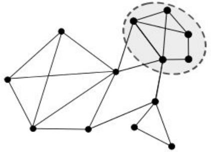
In the example above

= 3/(2*7+3)

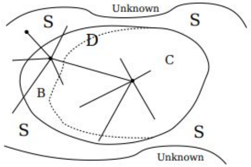
= 3/(2*7+3)
+ 3/(2*(12)+3)

c_S = cut size: number of edges going out of module
m_S = module size: number of edges inside module
$2m_s + c_s$ = vol (S) = sum of degrees for nodes in S

Locally defined objectives

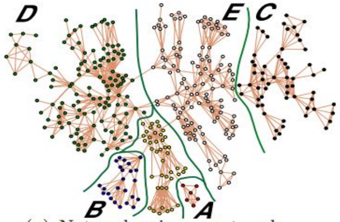


Defining and evaluating network communities based on ground-truth (Yang, J., Leskovec, J., Knowledge and Information Systems, 2015)

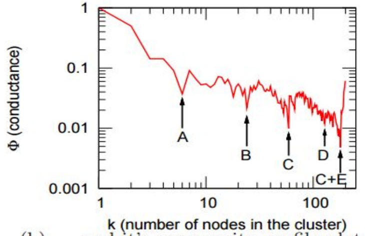


- Community detection from a seed node
  - Score proximity of nodes from seed using random walk
  - Expand from the closest node, and compute the objective
  - Local optima of objective (e.g. conductance) correspond to detected communities

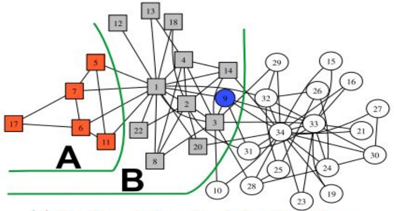
shutterstock.com · 1399192832



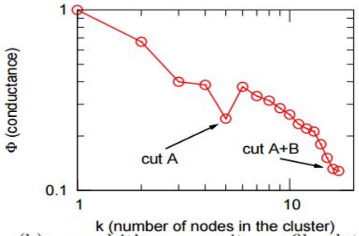
(g) Network science network . . .



(h) . . . and it's community profile plot



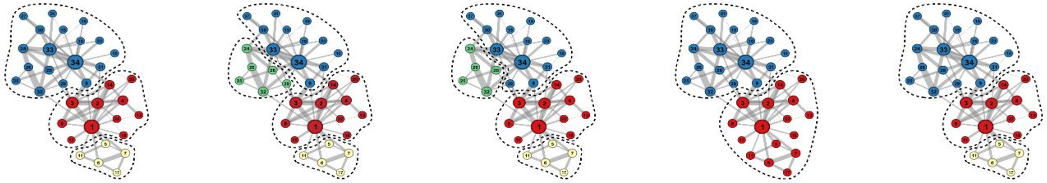
(a) Zachary's karate club network . . .



(b) . . . and it's community profile plot

Defining the Global Modular Structure of Networks

- **Number of links between them is more than chance**
  - Modularity Q (Newman & Grivan, Phys Rev E, 2004)
    - FastModularity (Clauset, Phys Rev E 2005); Louvain (Blondel et al., J Stat Mech Theory Exp, 2008)
- **Within them a random walk is more likely to trap**
  - Walktrap (Pons & Latapy, ISCIS 2005)
- **Coding gives efficient compression of any random walk**
  - Infomap (Rosvall & Bergstrom, PNAS 2008; PloS One 2010)
- **Follow their closest leader**
  - TopLeader



FastModularity $Q = 0.434$

Louvain $Q = 0.445$

Walktrap $Q = 0.44$

TopLeader(2) $Q = 0.403$

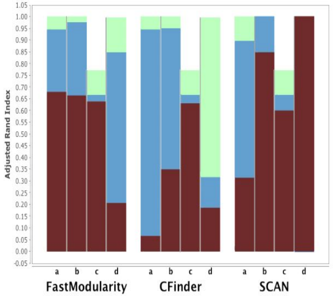
Infomap $Q = .434$

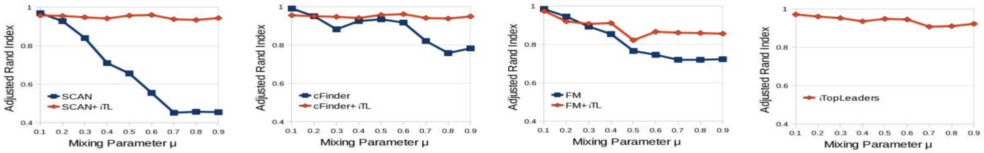
Outline

- Quick Notes
- Quick Recap of Centrality Measures
- Modules
    - Real graphs are modular
    - Spectral clustering
    - Objectives for quality of a module
    - **TopLeaders**
    - Using Betweenness Centrality
    - Modularity Optimization, FastModularity & Louvain
    - Resolution limits of Modularity
    - Link clustering
    - Evaluating clustering results

TopLeaders: K-medoid for graphs

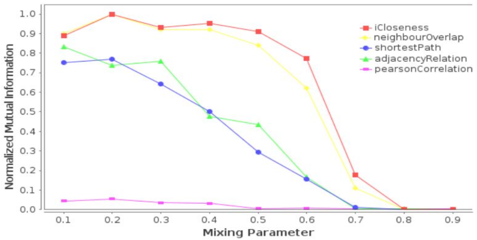
- Iteratively assigns nodes to leaders, selects leaders
  - Leader: central member in community
  - Community: set of followers surrounding a leader
  - Assigning followers to closest leader based on neighbourhoods
- Initialization requires k (central nodes with few neighbours in common)

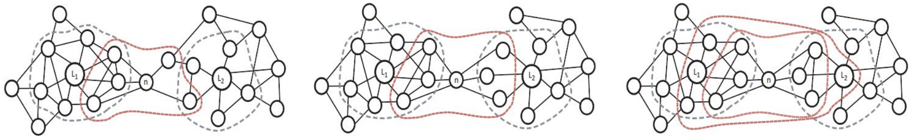




SCAN CFinder FastModularity GroundTruth

- Also identities outliers and hubs in the network
- Closeness measure based on diffusion of innovation





Outline

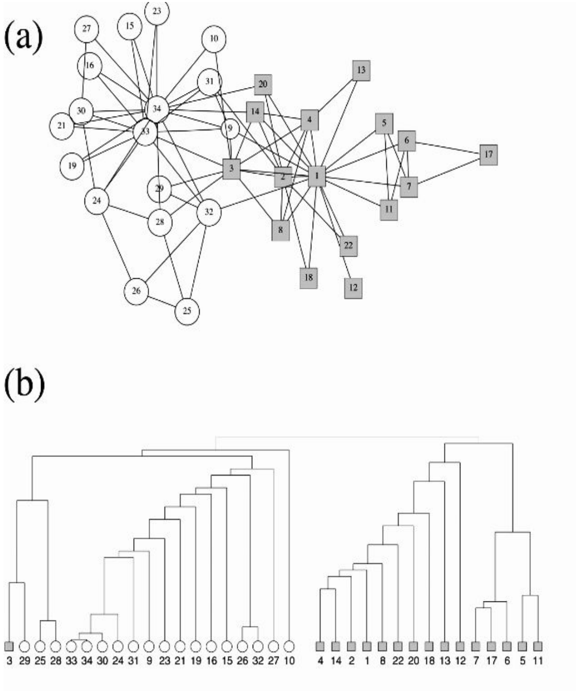
- Quick Notes
- Quick Recap of Centrality Measures
- Modules
  - Real graphs are modular
  - Spectral clustering
  - Objectives for quality of a module
  - TopLeaders
  - **Using Betweenness Centrality**
  - Modularity Optimization, FastModularity & Louvain
  - Resolution limits of Modularity
  - Link clustering
  - Evaluating clustering results

First community detection algorithm

A divisive hierarchical clustering

(Girvan and Newman, PNAS 2002)

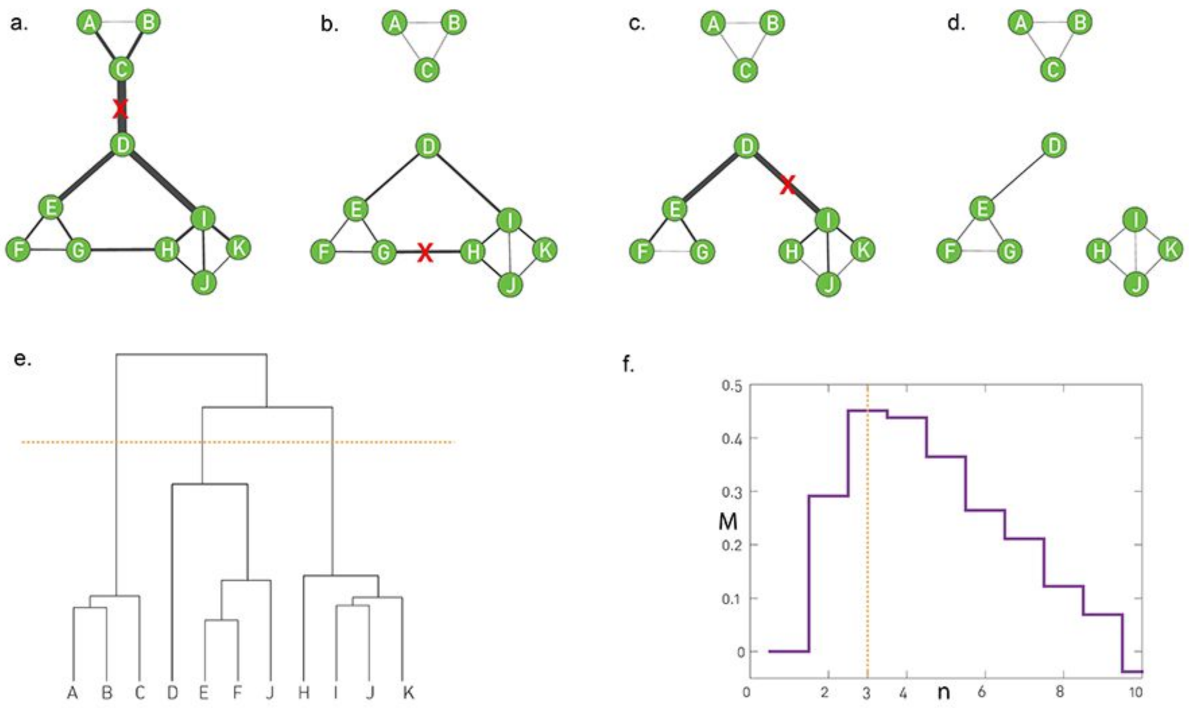
1. Calculate the betweenness for all edges in the network
2. Remove the edge with the highest betweenness
3. Recalculate betweennesses for all edges affected by the removal
4. Repeat from step 2 until no edges remain
5. Where to cut?



Side note: the infamous Karate club dataset https://networkkarate.tumblr.com/

Example

A divisive hierarchical clustering



Recursively remove **bridges**, edges with high edge-betweenness

In the resulted dendrogram, evaluate M for flat modules obtained at different levels

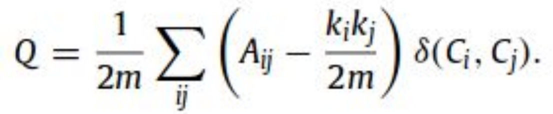
How to define M?

How good is a clustering of a network?

Originally proposed to know where to cut the dendrogram, but we optimize this directly in practice

Measure the difference between the fraction of edges that are within the clusters and the expected such fraction if the edges were randomly distributed when degrees are fixed

Use configuration model as the null model



$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(C_i, C_j).$$

Sum over all pairs of nodes in the same cluster:

A_{ij} - $(k_i/2m) * (k_j/2m)$

p_{ij} in configuration model

m = total edges in the graph
i, j: node indexes
k_i : degree of nodes i
C_i : cluster index that node i belongs to

Kronecker delta: 1 only if i & j are in the same cluster, $C_i = C_j$

Q-modularity

Sums over all pairs of nodes in the same clusters

$$Q = \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, C_j).$$

k_i = degree of node i
M = total edges

Kronecker delta: 1 only if i & j are in the same cluster, $C_i = C_j$

We can reformulate this to sum over clusters

$$Q = \sum_i (e_{ii} - a_i^2) \quad = \mathrm{Tr}[e] - ||e^2||$$

here $||e^2||_1 = \sum_{ij} e_{ij}^2$

e_{ij} : fraction of edges between cluster i and j

$a_i = e_{i.} = \Sigma_j\, e_{ij}$

$$\begin{array}{c|ccc} & \text{🔴} & \text{🟢} & \text{🔵} \\ \hline \text{🔴} & 0.25 & 0 & 0 \\ \text{🟢} & 0 & 0.25 & 0 \\ \text{🔵} & 0 & 0 & 0.25 \end{array}$$

Q**<=**1

Rule of thumb: Q>**0.3** indicates strong communities

Outline

- Quick Notes
- Quick Recap of Centrality Measures
- Modules
    - Real graphs are modular
    - Spectral clustering
    - Objectives for quality of a module
    - TopLeaders
    - Using Betweenness Centrality
    - **Modularity Optimization, FastModularity & Louvain**
    - Resolution limits of Modularity
    - Link clustering
    - Evaluating clustering results

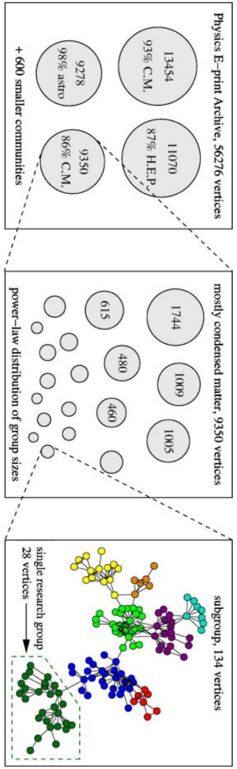
An agglomerative hierarchical clustering

(Newman, Phys. Rev. E 2004)

1. Start from every node a cluster
2. Initialize **e** as the adjacency matrix
3. Merge two cluster that give the highest gain in Q:

$$\Delta Q = 2(e_{ij} - a_i a_j)$$

4. Update the **e** by merging together the rows and columns corresponding to the joined communities
5. Go to step 3 until no increase in Q


Physics E-print Archive, 56276 vertices
13454 93% C.M.
9278 98% astro
11070 87% H.E.P.
9350 86% C.M.
+ 600 smaller communities
mostly condensed matter, 9350 vertices
1744
615
1009
480
460
1005
power-law distribution of group sizes
subgroup, 134 vertices
single research group 28 vertices

Modularity optimization

- Divisive hierarchical clustering (Girvan and Newman, PNAS 2002)
  - Removes the edge with highest betweenness
  - All pairs shortest paths: expensive to compute
  - can be approximated but still not scalable

- Agglomerative hierarchical clustering (Newman, Phys. Rev. E 2006)
  - Start from every node a cluster, and merge
  - O(n(m+n)) : n, m: number of nodes and edges
  - With heap based data structure ⇒ O(m log n) (Clauset et al., 2004)

    ⇒ **FastModularity**

Louvain, another agglomerative method

Agglomerative method tends to produce super-communities

$$Q = \sum_i (e_{ii} - a_i^2) \qquad Q = \sum_i \sum_{u,v \in i} (w_{uv} - w_{u.} w_{v.})$$

w_{uv} : normalized weight of the edge from node u to node v

Gain of adding node u to community i is: $\Delta Q = 2 \sum_{v \in i} (w_{uv} - w_{u.} w_{v.})$.

Move nodes around (only through links), aggregate clusters, repeat

(Blondel et al. Journal of Statistical Mechanics, 2008)

O(n log n)

Outline

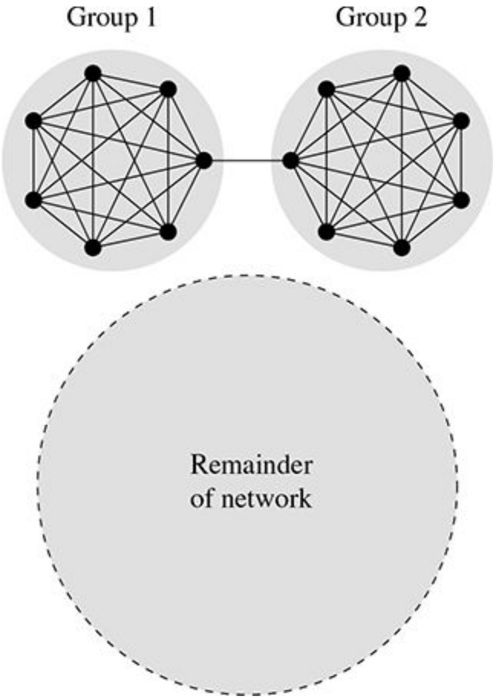
- Quick Notes
- Quick Recap of Centrality Measures
- Modules
  - Real graphs are modular
  - Spectral clustering
  - Objectives for quality of a module
  - TopLeaders
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - **Resolution limits of Modularity**
  - Link clustering
  - Evaluating clustering results

Q problems

1. very different divisions of the network but same Q
2. resolution limit, the inability to see communities in a network if they are too small, relative to the size of the network as a whole

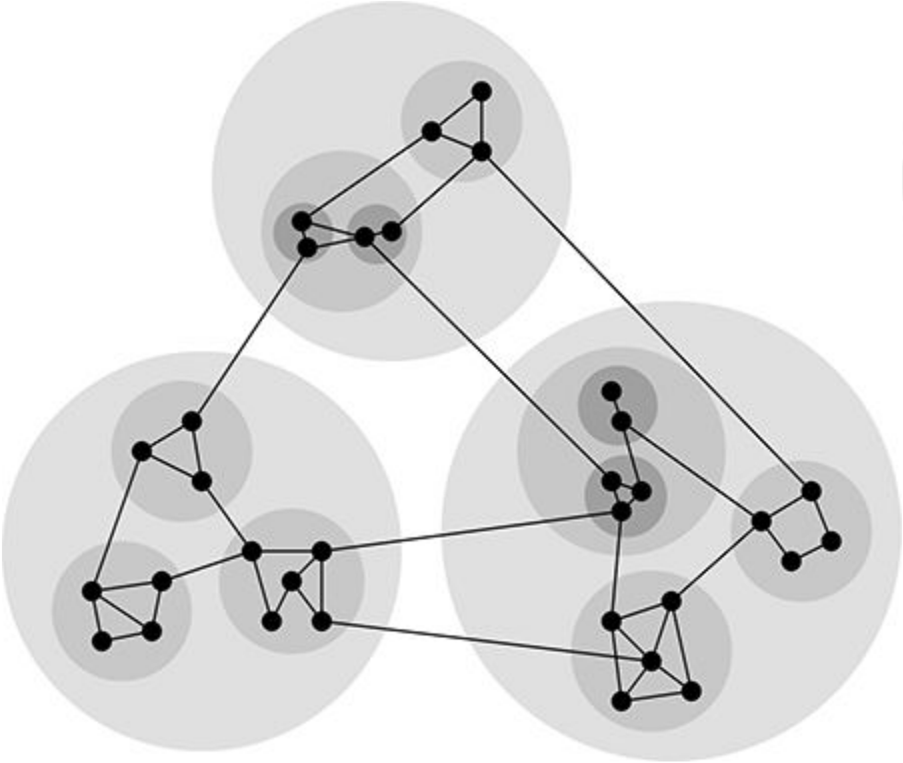
$$\Delta Q = \frac{1}{2m}\left(1 - \frac{\kappa_1 \kappa_2}{2m}\right), >0 \text{ iff } \quad \kappa_1 \kappa_2 < 2m.$$

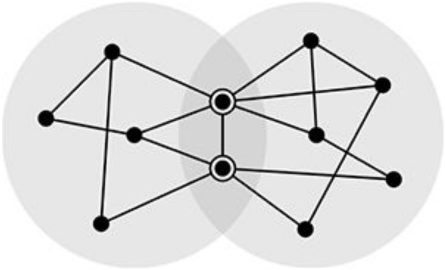
κ_1 and κ_2 be the sums of the degrees of the nodes in each of the two groups

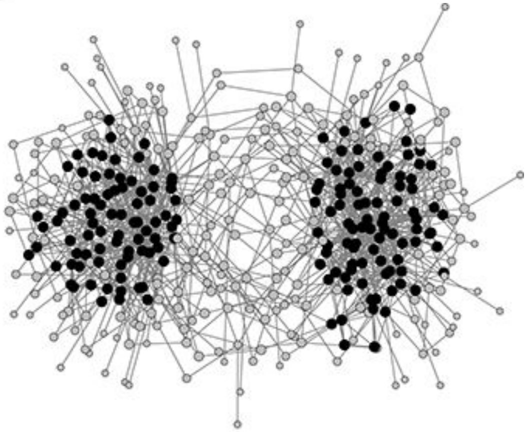


E.g. 5000 edges, can not detect degree sum less than 100

Overlap, hierarchy, periphery





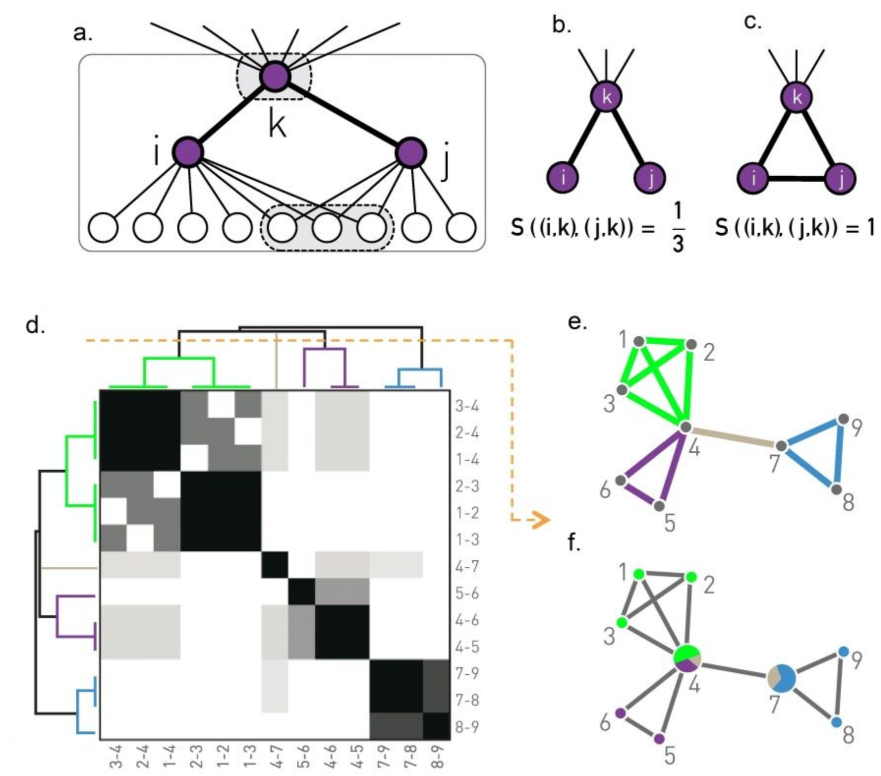


(b)

Link Clustering

Find overlapping clusters naturally by clustering edges instead of nodes

The similarity of a link pair is determined by the neighborhood of the nodes connected by them.



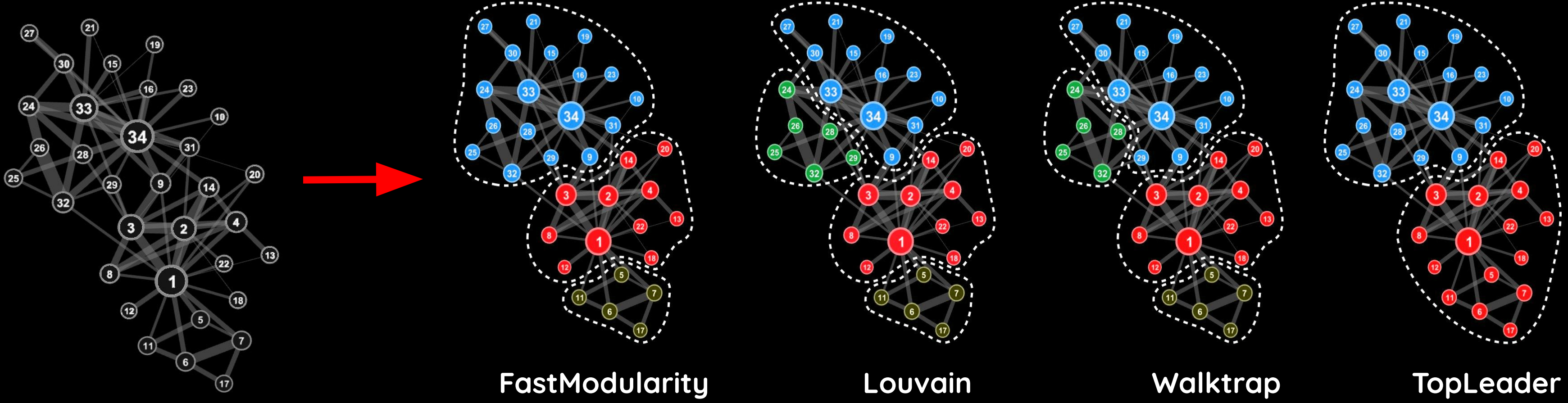
Ahn YY, Bagrow JP, Lehmann S. Link communities reveal multiscale complexity in networks. nature. 2010 Aug;466(7307):761-4.

Outline

- Quick Notes
- Quick Recap of Centrality Measures
- Modules
  - Real graphs are modular
  - Spectral clustering
  - Objectives for quality of a module
  - TopLeaders
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - Resolution limits of Modularity
  - Link clustering
  - **Evaluating clustering results**

Evaluating the Modular Structure of Networks

Given different algorithms **which one** is better?



FastModularity

Louvain

Walktrap

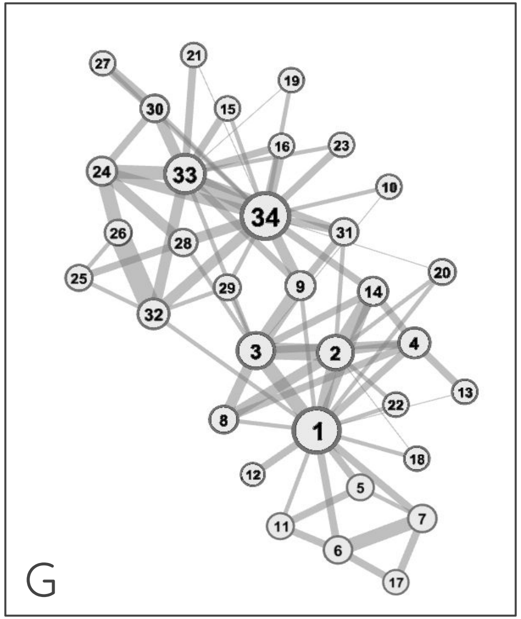
TopLeader

Evaluation of Community Detection

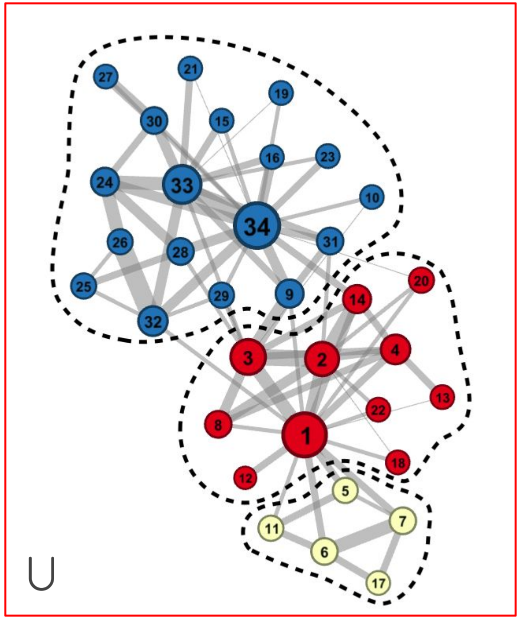
Validation on benchmarks for which we know the ground-truth, a common practice

(G_1 , U_1)
(G_2 , U_2)
(G_3 , U_3)

⋮

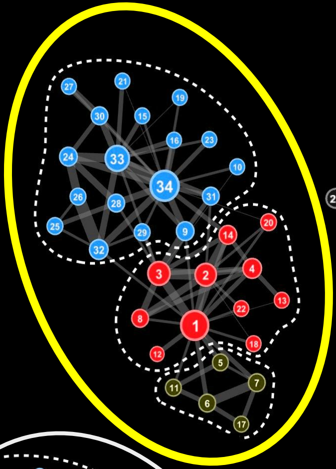


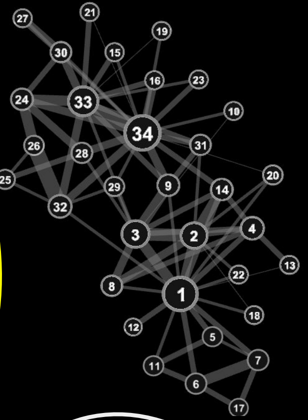
Ground-Truth

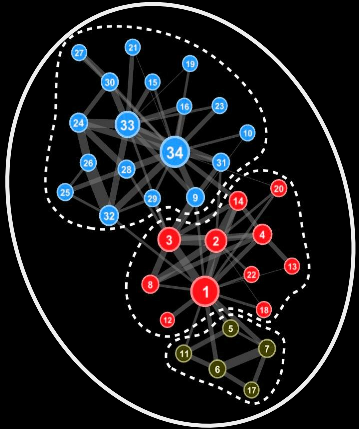


Validation on Benchmarks

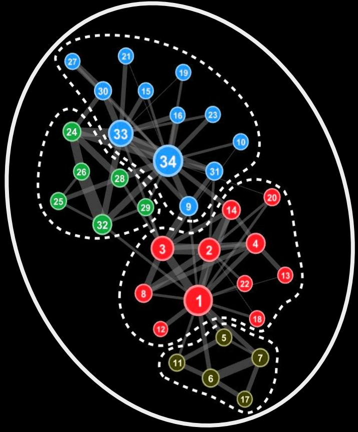
Compare with ground-truth



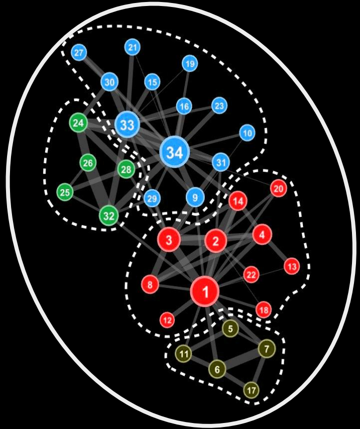




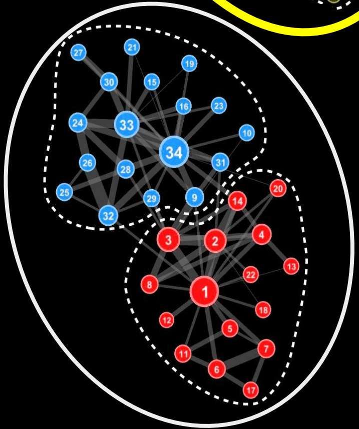
FastModularity



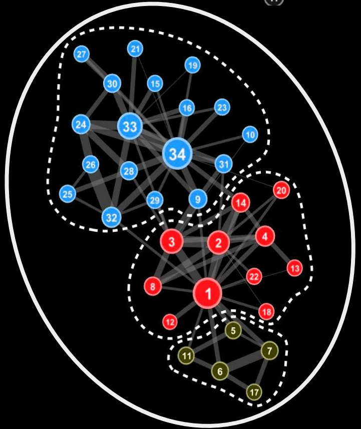
Louvain



Walktrap



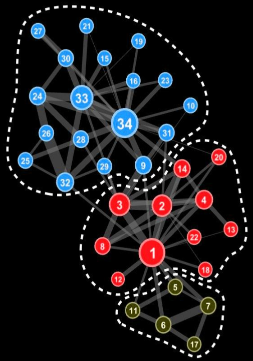
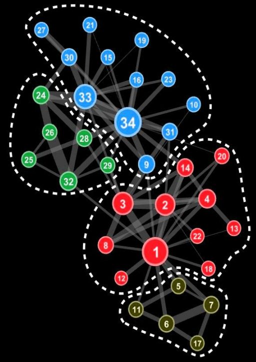
TopLeader

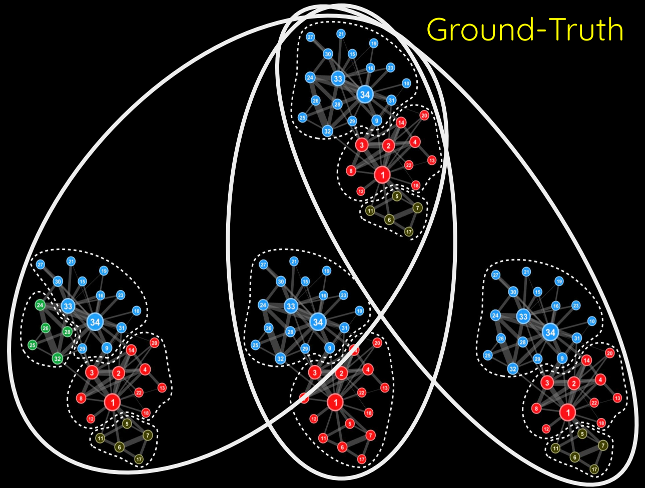


InfoMap

Validation on Benchmarks

Agreement measure

A( , )

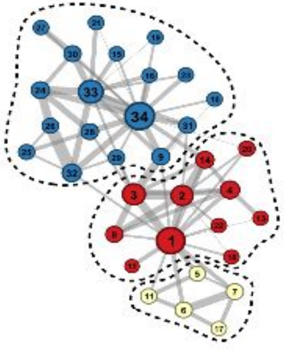


Clusterings Agreement Measures

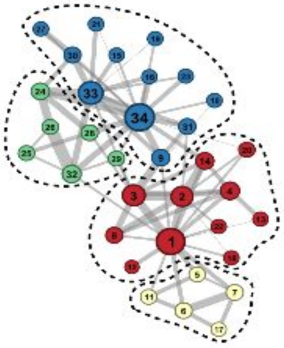
Current families, background

- Set matching
- Information theoretic
- Pair counting

A(



,



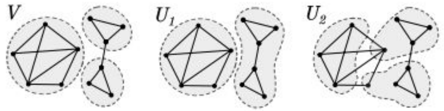
)

Clusterings Agreement Measures

Set matching, background

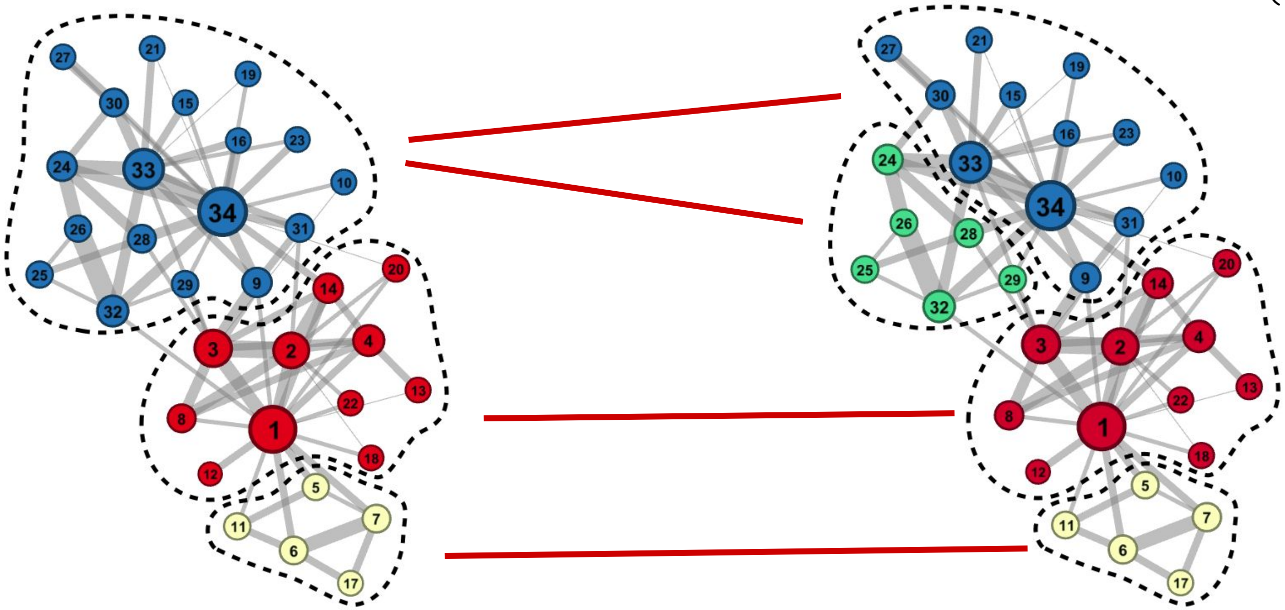
“problem of matching” since it only compares the best matched clusters

example:



$A(U_1,V)$ vs $A(U_2,V)$

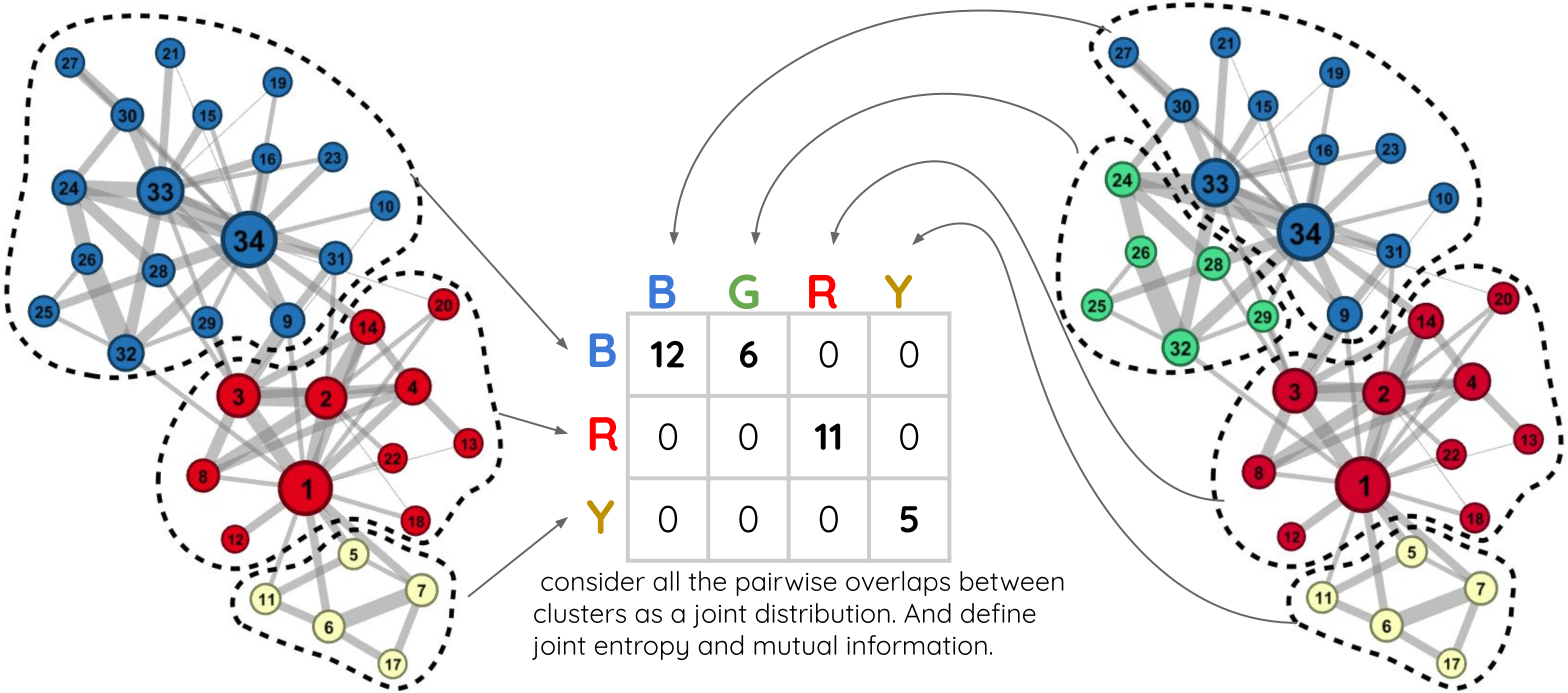
Read more here



Clusterings Agreement Measures

Information theoretic, background

Examples: Variation of Information (VI), Normalized Mutual Information (NMI)



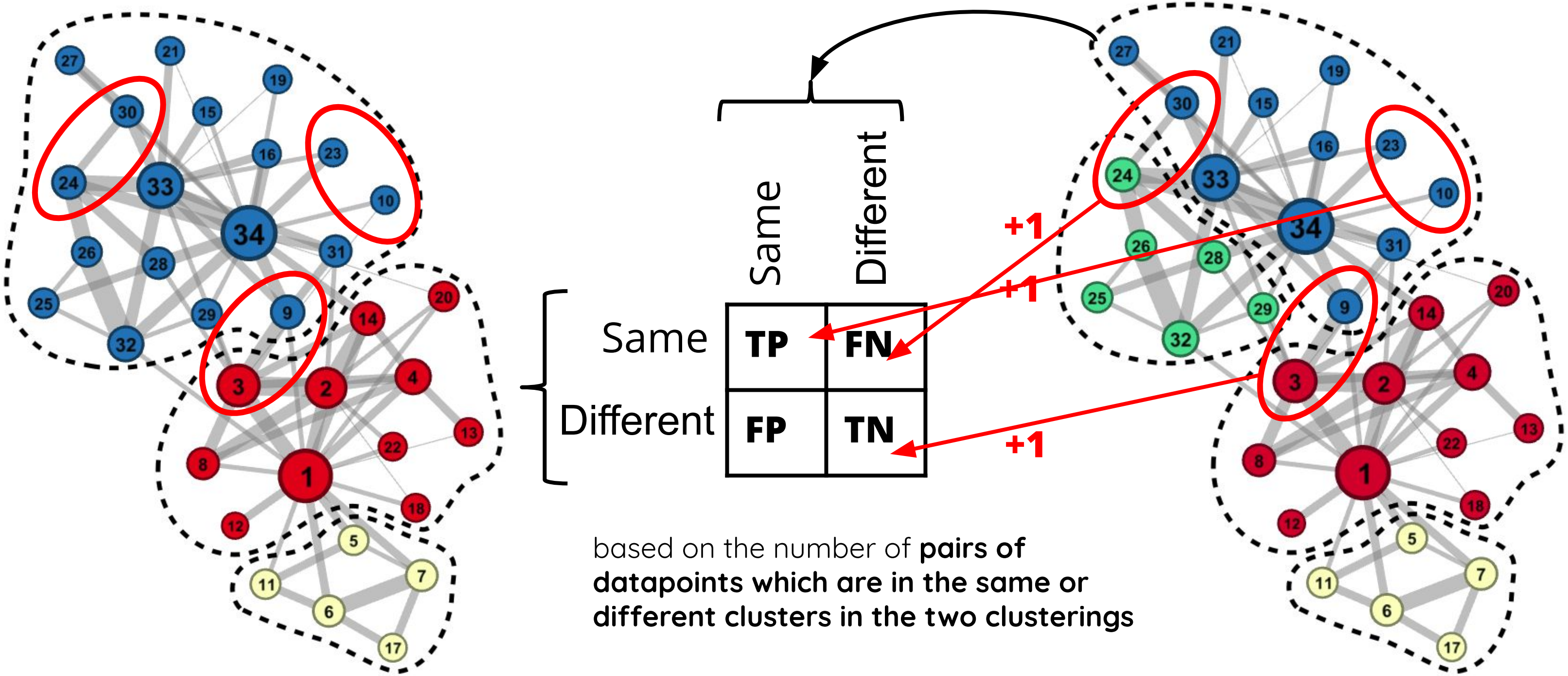
	B	G	R	Y
B	**12**	**6**	0	0
R	0	0	**11**	0
Y	0	0	0	**5**

consider all the pairwise overlaps between clusters as a joint distribution. And define joint entropy and mutual information.

Clusterings Agreement Measures

Pair counting, background

Examples: Jaccard, Rand Index, F-measure, Adjusted Rand Index (ARI)



	Same	Different
Same	**TP**	**FN**
Different	**FP**	**TN**

based on the number of **pairs of datapoints which are in the same or different clusters in the two clusterings**

Generalization

Linking the two families

- **Pair counting**
- **Information theoretic**

dispersion in the contingency table

	Same	Different
Same	**TP**	**FN**
Different	**FP**	**TN**

	B	G	R	Y
B	12	6	0	0
R	0	0	11	0
Y	0	0	0	5

$$\mathbf{TP} = \binom{12}{2}+\binom{6}{2}+\binom{11}{2}+\binom{5}{2}$$

Read more
here

Generalization

Measuring dispersion

	B	G	R	Y	Σ
B	12	6	0	0	18
R	0	0	11	0	11
Y	0	0	0	5	5
Σ	12	6	11	5	34

$$\left.\begin{array}{l} \varphi(18) - \varphi(12) - \varphi(6) \\ \varphi(11) - \varphi(11) \\ \varphi(5) - \varphi(5) \end{array}\right\} +$$

$$\frac{\varphi(18) - \varphi(12) - \varphi(6)}{\varphi(34)}$$

Read more
here

Generalization

Subsumes pair counting

	B	G	R	Y	Σ
B	12	6	0	0	18
R	0	0	11	0	11
Y	0	0	0	5	5
Σ	12	6	11	5	34

$$\varphi(x) = \binom{x}{2}$$

$$\frac{\varphi(18) - \varphi(12) - \varphi(6)}{\varphi(34)} = 1 - \underbrace{\frac{TP+TN}{TP+TN+FP+FN}}_{\textbf{Rand Index}}$$

Read more here

Generalization

Subsumes information theoretic

$$\begin{array}{cc|cccc|c} & & \multicolumn{4}{c}{V} & \\ & & B & G & R & Y & \Sigma \\ \hline U & B & 12 & 6 & 0 & 0 & 18 \\ & R & 0 & 0 & 11 & 0 & 11 \\ & Y & 0 & 0 & 0 & 5 & 5 \\ \hline & \Sigma & 12 & 6 & 11 & 5 & 34 \end{array}$$

$$\varphi(x) = x \log(x)$$

$$\frac{\varphi(18) - \varphi(12) - \varphi(6)}{\varphi(34)} = \frac{\overbrace{H(U, V) - I(U, V)}}{\log(34)}$$

Variation of Information

Read more here

Generalization

Second normalization

	B	G	R	Y	Σ
B	12	6	0	0	18
R	0	0	11	0	11
Y	0	0	0	5	5
Σ	12	6	11	5	34

$$\frac{\varphi(18) - \varphi(12) - \varphi(6)}{\varphi(34)}$$

$$\frac{\varphi(18) - \varphi(12) - \varphi(6)}{\varphi(\frac{18}{34} \times \frac{12}{34}) + \varphi(\frac{18}{34} \times \frac{6}{34}) + \varphi(\frac{18}{34} \times \frac{11}{34}) + \dots}$$

- ➢ $\varphi(x) = x^2 \Rightarrow$ **ARI** (Adjusted Rand Index)
- ➢ $\varphi(x) = x \log x \Rightarrow$ **NMI** (Normalized Mutual Information)

Read more
here

Generalization

Subsumes common measures

- Pair counting
- Information theoretic

A generalized distance with a plug-in function

- $\varphi(x) = x^2 \Rightarrow$ RI
- $\varphi(x) = x \log x \Rightarrow$ VI

Normalization 1

- $\varphi(x) = x^2 \Rightarrow$ ARI
- $\varphi(x) = x \log x \Rightarrow$ NMI

Normalization 2

Read more here