
Community Structure Detection

Amar Chandole
Ameya Kabre
Atishay Aggarwal

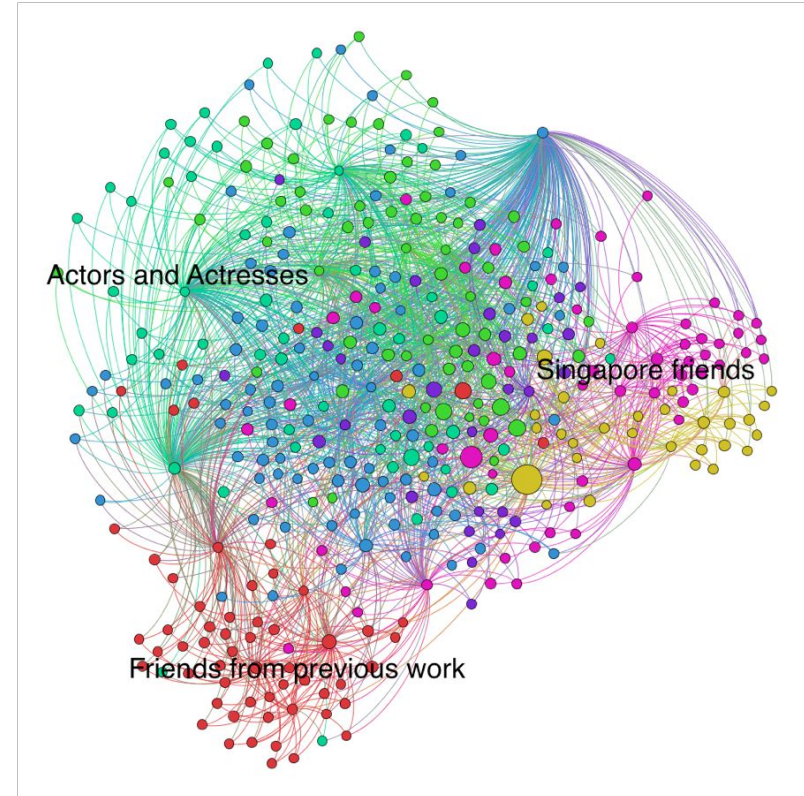
What is a network?

- Group or system of interconnected people or things
- Ways to represent a network:
 - Matrices
 - Sets
 - Sequences
 - Time series
 - **Graphs**



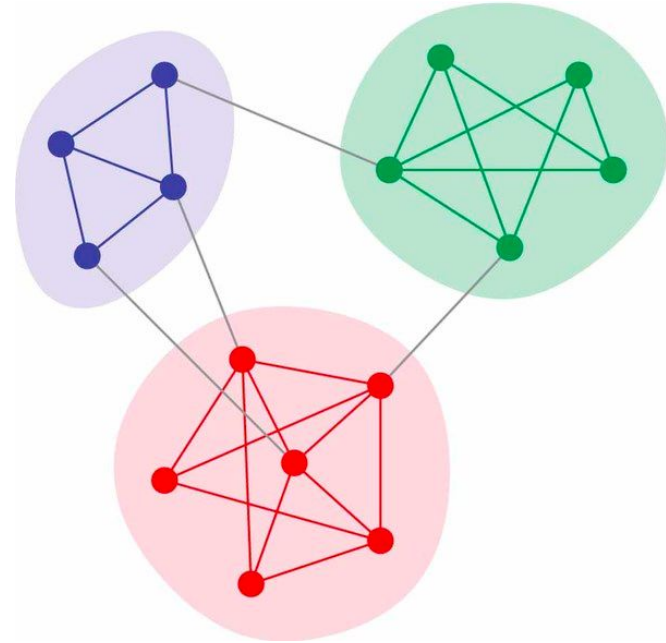
Networks as Graphs

- Entities represented as **nodes**
- Connections between entities represented as **edges**
- Example
 - Social Network
 - People: Nodes
 - Friendship: Edges
 - Citation Network
 - Documents: Nodes
 - Citation: Edges



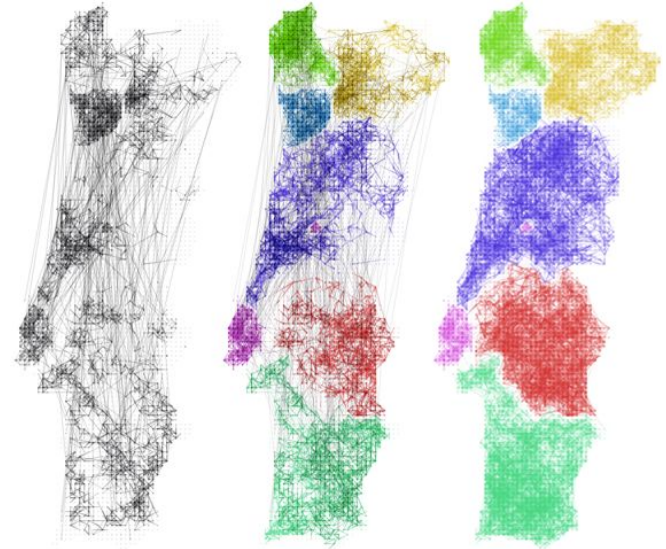
Community Structures

- Group of vertices **within** which connections are dense but **between** which they are sparser
- Can be found in the Internet, the world wide web, citation networks, transportation networks, email networks, food webs, social and biochemical networks



Need to study Community Structures

- Networks of interest found to naturally divide into communities or modules
- Communities generally have similar behavior, decision making characteristics
- Properties of communities may be quite different from average properties of the network



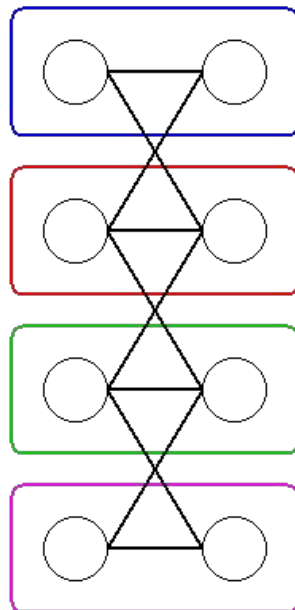
Detection of Community Structures

- Computer Science Approach
 - Graph Partitioning
- Social Science Approach
 - Block Modeling or Hierarchical Clustering or Community Structure Detection

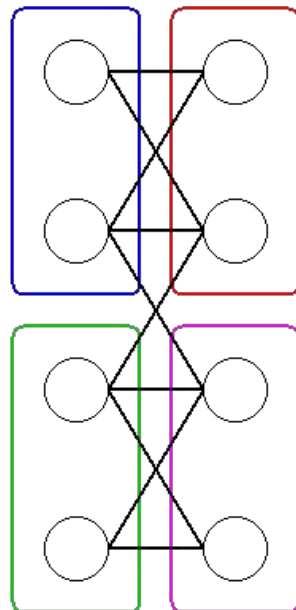
Graph Partitioning

- Goal: Find division of network irrespective of existence of a good division
- Condition: Number and size of groups is generally known in advance
- Example: Division of tasks between different cores of a parallel computer

Graph Partitioning



Edge Crossings = 6



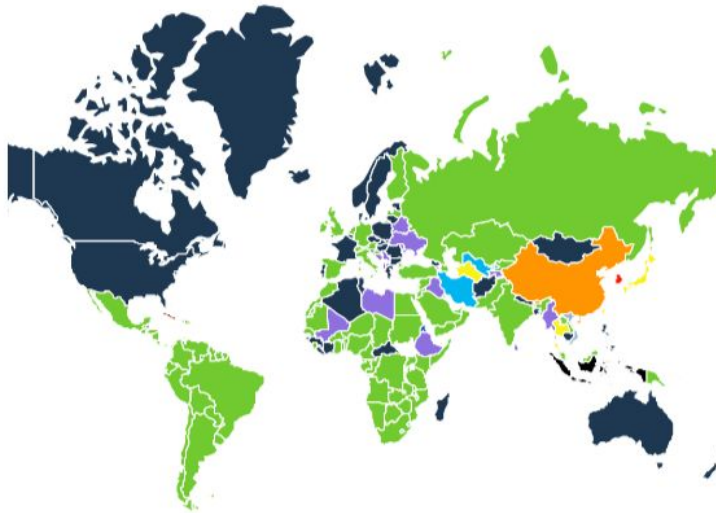
Edge Crossings = 10

Community Structure Detection

- Goal: Find a good division of the network provided that it exists
- Condition: Number and size of groups are determined by the network itself
- Example: Find most popular messaging application in each country

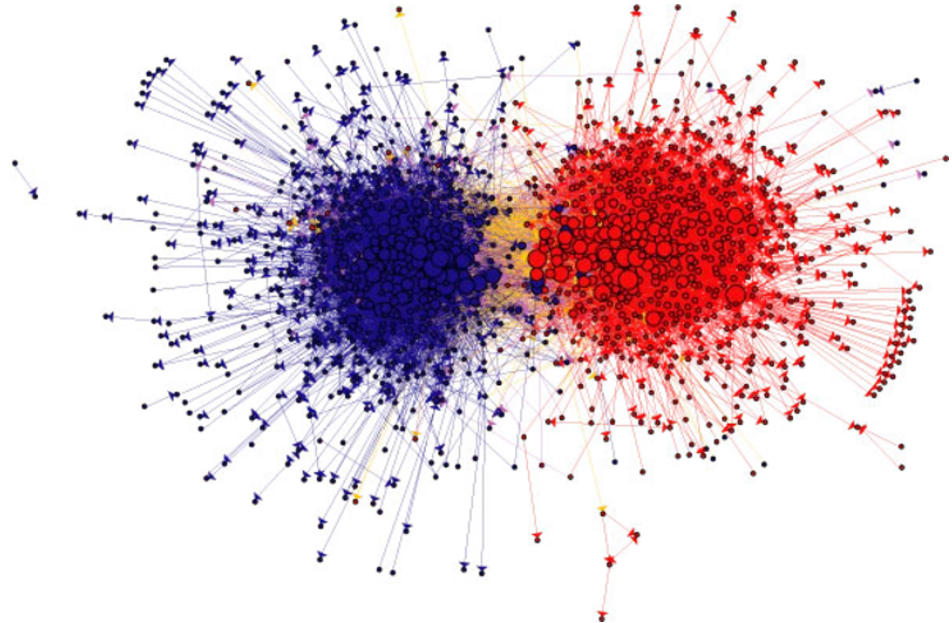
Community Structure Detection

Most Popular Messaging App in Every Country



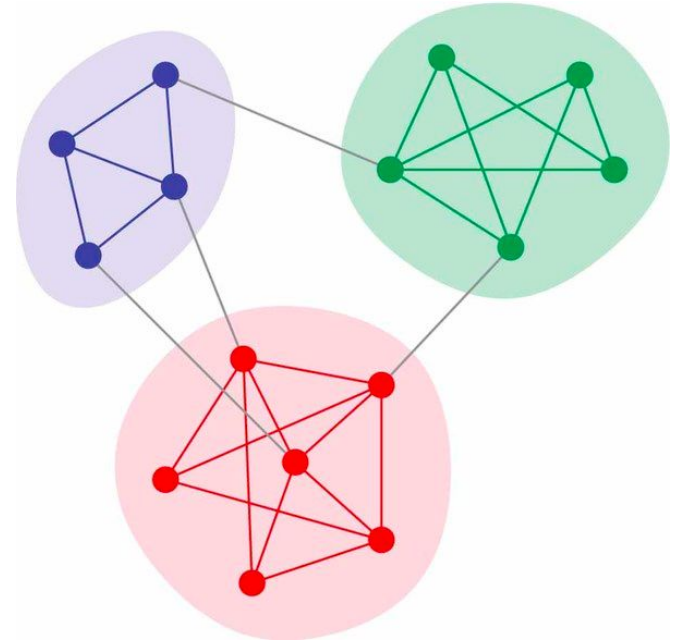
WhatsApp Messenger Viber Line WeChat Telegram KakaoTalk
imo Zalo BBM ChatOn

Political Orientation Across Blogs on the Internet



Girvan and Newman (GN) Algorithm

- Edge betweenness
 - Number of shortest paths that run between a pair of nodes
- Each edge assigned a betweenness score
- Edges with high betweenness scores removed
- Betweenness recalculated



Performance of GN Algorithm

- Computationally very expensive
- For m edges and n nodes:
 - Complexity of $O(m^2n)$ for regular graphs
 - $O(n^3)$ for sparse graphs

What do the papers that we
reviewed propose?

Paper 1 :

Fast algorithm for detecting community structure in networks (arXiv'03)

Author: M. E. J. Newman

University of Michigan, Ann Arbor

Drawbacks of graph partitioning and GN algorithm

- Computation effort required is immense
- Feasible even upto thousand nodes
- Infeasible for present day's interactive networks containing millions and billions nodes

Overcoming drawbacks

- Use of modularity
- Optimizing modularity



1. Modularity

Number of edges falling within groups minus the expected number in an equivalent network with edges placed at random

→ **Positive**

Magnitude indicates the probability of a community structure

→ **Zero**

Edges are simply a random distribution

→ **Negative**

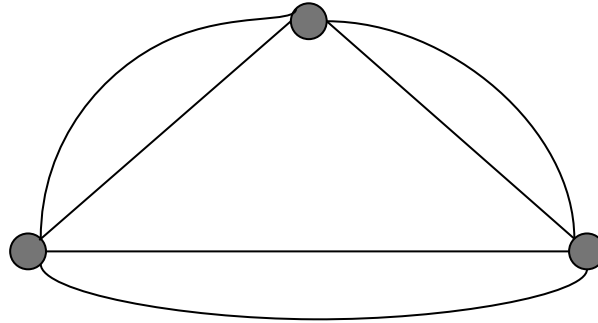
Indicates that a community structure must be absent

Example scenario for Modularity

Suppose we have 3 vertices and 6 edges.

Then by random distribution of edges, the network is expected to look like:

EXPECTED:

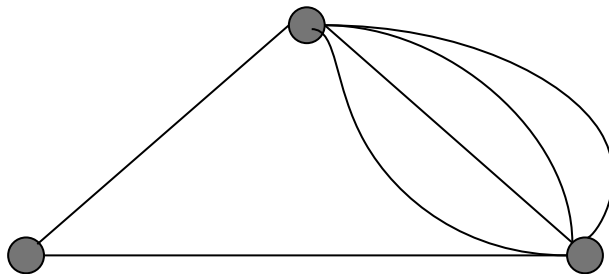


Example scenario for Modularity

Suppose we have 3 vertices and 6 edges.

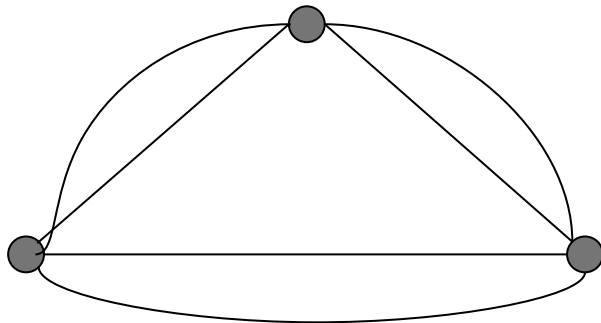
Then by random distribution of edges, the network is expected to look like:

ACTUAL:

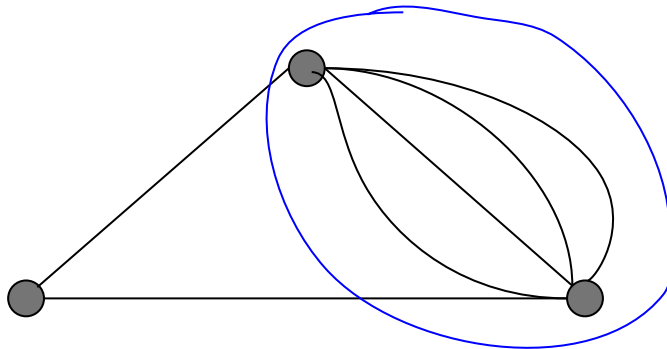


Example scenario for Modularity

EXPECTED:



ACTUAL:



Too many connections than expected. Something is fishy here!

Modularity will find out this deviation from expected value using different expressions (explained in later slides) and help us divide network into communities.



2. Ways to Optimize Modularity

- Simulated Annealing
- Genetic Algorithms
- Greedy Algorithms
- Spectral Techniques

What purpose does Modularity serve in this paper?

- Values of Q indicative of presence of community structure
- $Q = 0$ indicates random presence
- $Q \geq 0.3$ indicates strong presence

Mathematical Definitions

- Modularity Q is defined as:

$$Q = \sum (e_{ii} - a_i^2) \quad \forall i \quad \text{where,}$$

- e_{ij} is the fraction of edges in network that connect vertices in group i to those in group j .
 - $a_i = \sum e_{ij} \quad \forall j$
- $\Delta Q = e_{ij} + e_{ji} - 2 a_i a_j = 2 (e_{ij} - a_i a_j)$

Optimizing modularity

- Ways to divide n vertices into g non-empty groups is $S_n^{(g)}$
- Number of distinct community divisions:

$$\sum S_n^{(g)} \text{ for } g=1 \text{ to } n$$

- Maximum 20 - 30 vertices

Greedy Algorithm

- Each vertex is considered as a community
- Join 2 vertices such that it results in the greatest increase or smallest decrease in the value of modularity
- Process can be represented as a dendrogram and cuts through it give communities

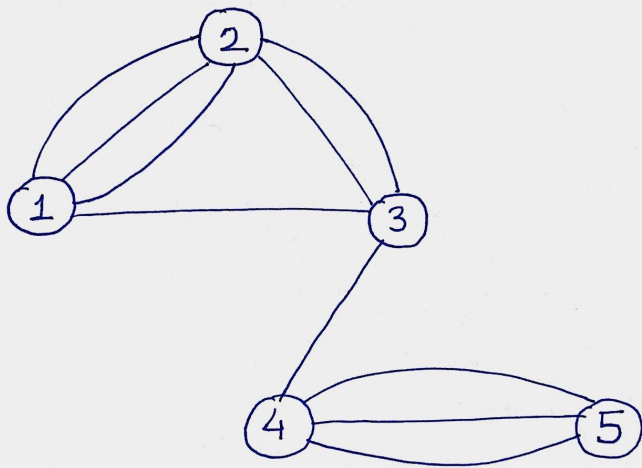
Steps to follow

1. Calculate e_{ij} matrix for the graph
2. Calculate ΔQ for every possible combination of 2 nodes
3. Join vertices with highest value of ΔQ
4. Calculate Q after the join and update dendrogram
5. Recompute e_{ij} matrix

Community Structures

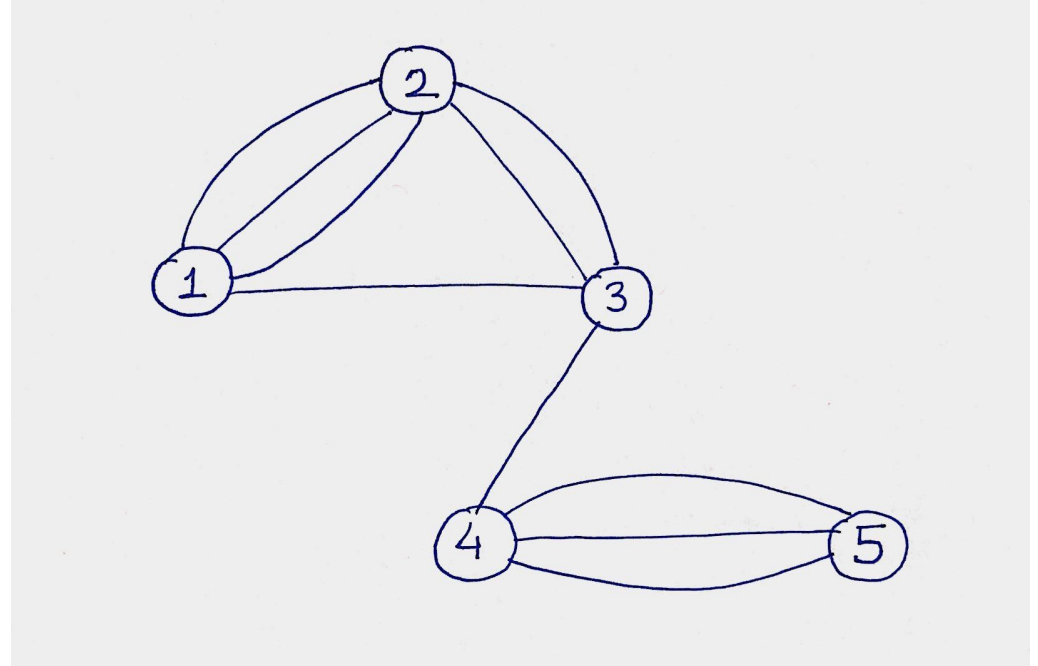
1. We have the dendrogram with values of Q after each join
2. Starting from the highest value of Q , if it is greater than 0.3, cut through it
3. We obtain the final community structure when no value > 0.3 left to be cut

Example



The eij matrix :

$$\begin{bmatrix} 0 & 0.3 & 0.1 & 0 & 0 \\ 0.3 & 0 & 0.2 & 0 & 0 \\ 0.1 & 0.2 & 0 & 0.1 & 0 \\ 0 & 0 & 0.1 & 0 & 0.3 \\ 0 & 0 & 0 & 0.3 & 0 \end{bmatrix}$$



All combinations

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

$$(1,2) \Rightarrow 2(0.3 - 0.4 * 0.5) = 0.2$$

$$(2,3) \Rightarrow 2(0.2 - 0.4 * 0.5) = 0.0$$

$$(1,3) \Rightarrow 2(0.0 - 0.4 * 0.4) = -0.32$$

$$(3,4) \Rightarrow 2(0.1 - 0.4 * 0.4) = -0.12$$

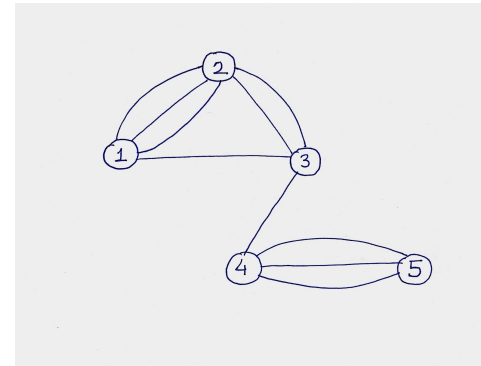
$$(4,5) \Rightarrow 2(0.3 - 0.3 * 0.4) = 0.36$$

$$\text{Max } \Delta Q = 0.36$$

Thus, join 4 & 5 and calculate $Q = -0.28$

The eij matrix :

0	0.3	0.1	0	0
0.3	0	0.2	0	0
0.1	0.2	0	0.1	0
0	0	0.1	0	0.3
0	0	0	0.3	0

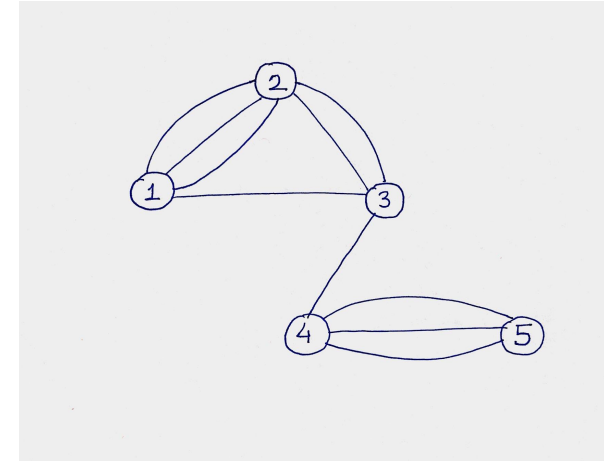


The updated eij matrix :

$$\begin{bmatrix} 0 & 0.3 & 0.1 & 0 \\ 0.3 & 0 & 0.2 & 0 \\ 0.1 & 0.2 & 0 & 0.1 \\ 0 & 0 & 0.1 & 0.3 \end{bmatrix}$$

The original eij matrix :

$$\begin{bmatrix} 0 & 0.3 & 0.1 & 0 & 0 \\ 0.3 & 0 & 0.2 & 0 & 0 \\ 0.1 & 0.2 & 0 & 0.1 & 0 \\ 0 & 0 & 0.1 & 0 & 0.3 \\ 0 & 0 & 0 & 0.3 & 0 \end{bmatrix}$$



All combinations

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

$$(1,2) \Rightarrow 2(0.3 - 0.4 * 0.5) = 0.2$$

$$(2,3) \Rightarrow 2(0.2 - 0.4 * 0.5) = 0.0$$

$$(1,3) \Rightarrow 2(0.0 - 0.4 * 0.4) = -0.32$$

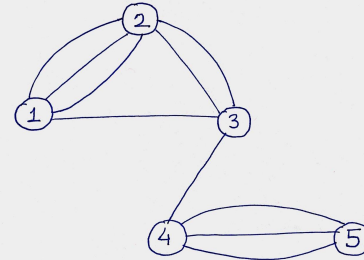
$$(3, 4-5) \Rightarrow 2(0.1 - 0.4 * 0.4) = -0.12$$

$$\text{Max } \Delta Q = 0.2$$

Thus, join 1 and 2 and calculate $Q = -0.41$

The updated eij matrix :

$$\begin{bmatrix} 0 & 0.3 & 0.1 & 0 \\ 0.3 & 0 & 0.2 & 0 \\ 0.1 & 0.2 & 0 & 0.1 \\ 0 & 0 & 0.1 & 0.3 \end{bmatrix}$$

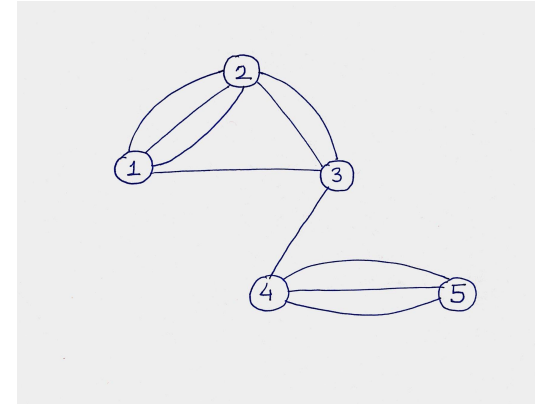


The updated eij matrix :

$$\begin{bmatrix} 0.3 & 0.3 & 0 \\ 0.3 & 0 & 0.1 \\ 0 & 0.1 & 0.3 \end{bmatrix}$$

The previous eij matrix :

$$\begin{bmatrix} 0 & 0.3 & 0.1 & 0 \\ 0.3 & 0 & 0.2 & 0 \\ 0.1 & 0.2 & 0 & 0.1 \\ 0 & 0 & 0.1 & 0.3 \end{bmatrix}$$



All combinations

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

$$(1-2, 3) \Rightarrow 2(0.3 - 0.4 * 0.6) = 0.12$$

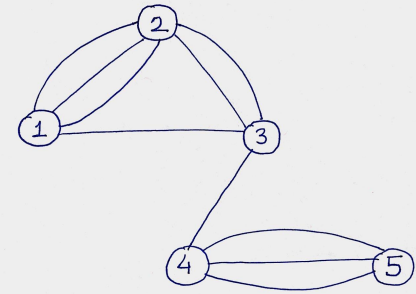
$$(3, 4-5) \Rightarrow 2(0.1 - 0.4 * 0.4) = -0.12$$

$$\text{Max } \Delta Q = 0.12$$

Thus, join 1-2 with 3 and calculate $Q = 0.05$

The updated eij matrix :

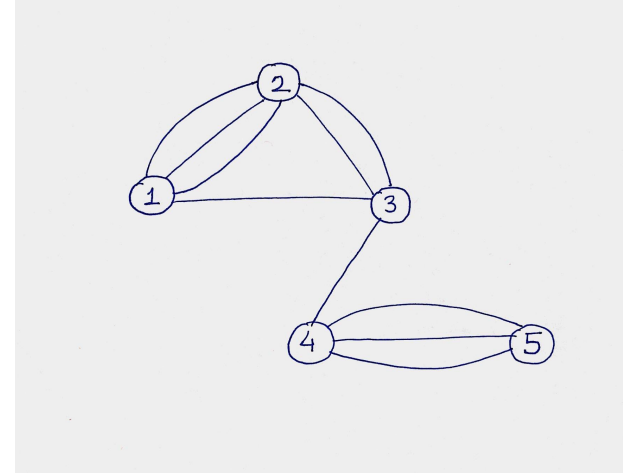
$$\begin{bmatrix} 0.3 & 0.3 & 0 \\ 0.3 & 0 & 0.1 \\ 0 & 0.1 & 0.3 \end{bmatrix}$$



The updated eij matrix: The previous eij matrix:

$$\begin{bmatrix} 0.6 & 0.1 \\ 0.1 & 0.3 \end{bmatrix}$$

$$\begin{bmatrix} 0.3 & 0.3 & 0 \\ 0.3 & 0 & 0.1 \\ 0 & 0.1 & 0.3 \end{bmatrix}$$



All combinations

Join the 2 communities:

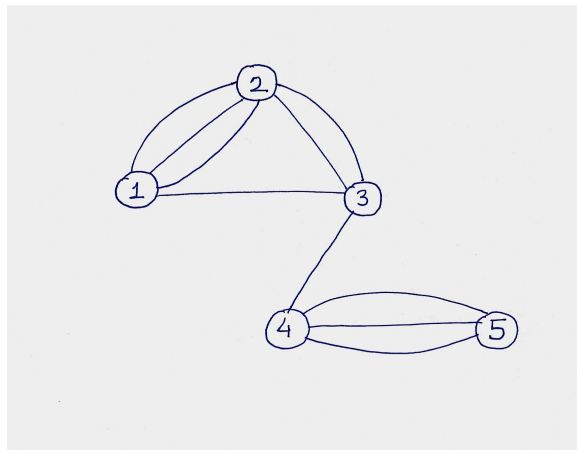
1-2-3 and 4-5

As there is just one single component now,
we stop.

$$Q = 0.88$$

The updated eij matrix :

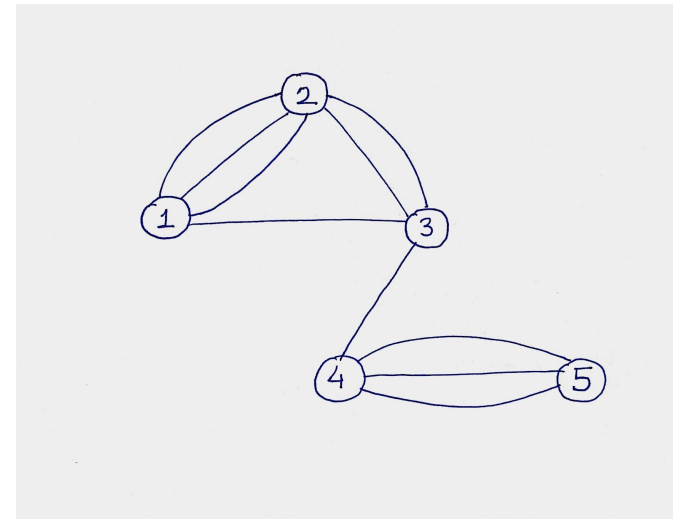
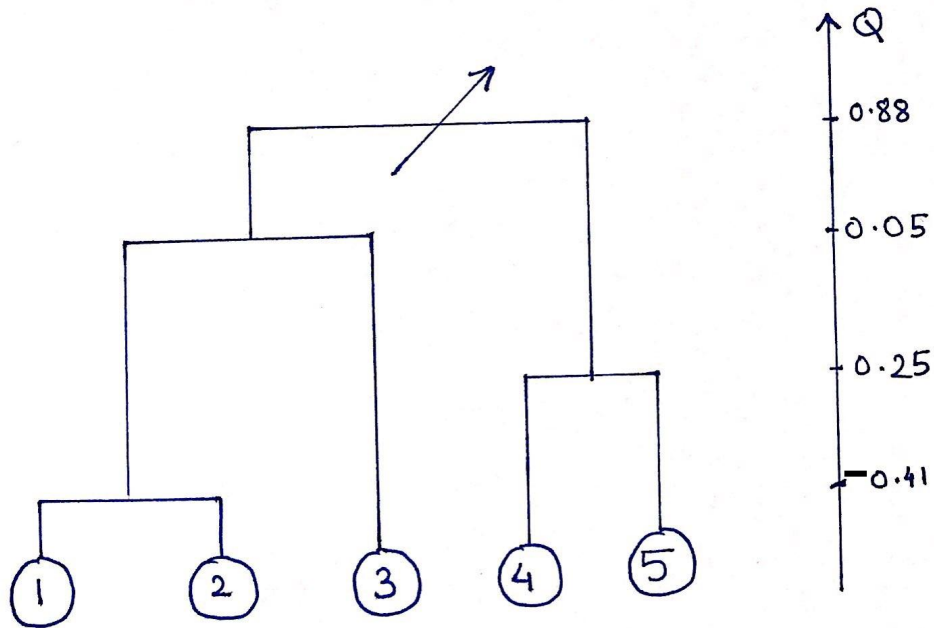
$$\begin{bmatrix} 0.6 & 0.1 \\ 0.1 & 0.3 \end{bmatrix}$$



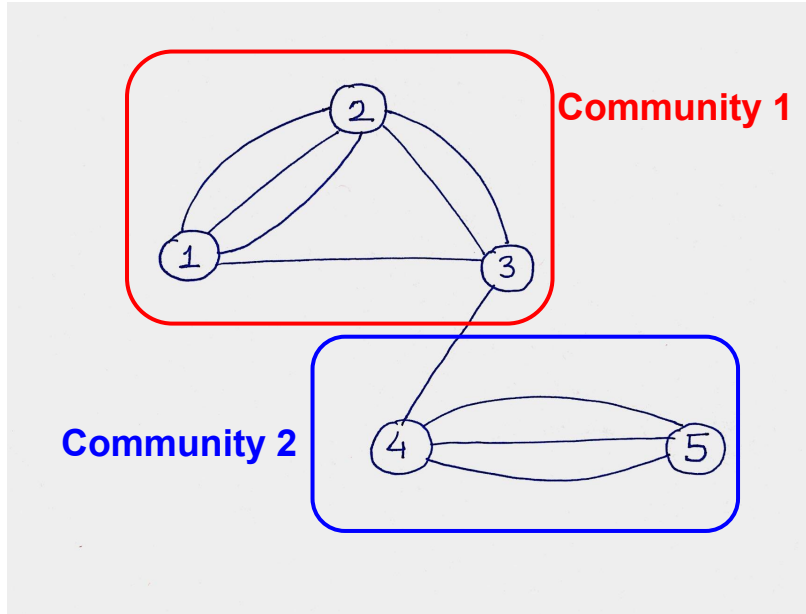
Splitting into communities

1. Now, we observe that the maximum value of Q is 0.88
2. As, this is ≥ 0.3 , we cut through it
3. We obtain 2 communities - (1-2-3) and (4-5)
4. No other value of $Q \geq 0.3$
5. The final community structure: (1-2-3) and (4-5)

The dendrogram



Resulting community structure division



Performance of Greedy Algorithm

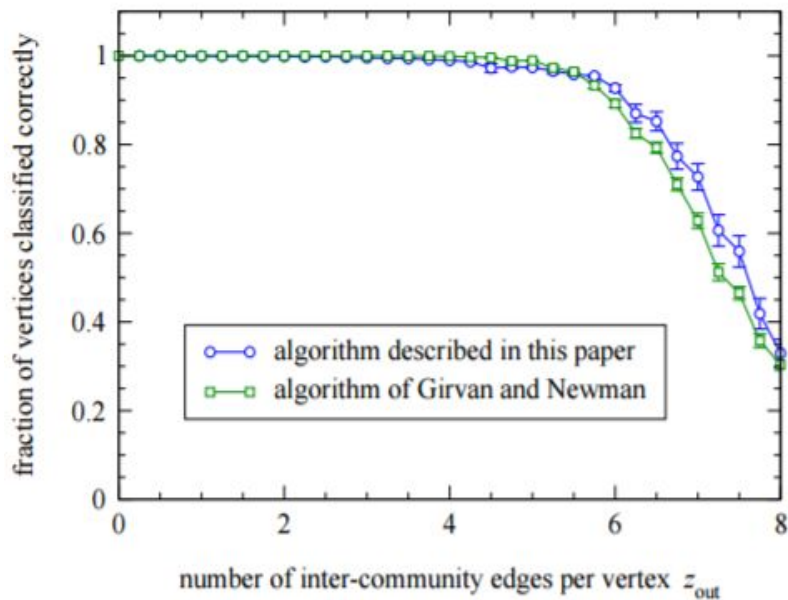
- Computationally less expensive
- For m edges and n nodes
 - Complexity of $O(m+n)$ for each join operation
 - $O(n(m+n))$ for $n-1$ join operations

Computer generated graph

- $n=128$ vertices. 4 groups of 32.
- $Z_{in} + Z_{out} = 16$

Observations:

- 90% correct for $Z_{out} \leq 6$
- For $Z_{out}=5$,
Greedy algo : 97.4% vertices
GN algo : 98.9% vertices.
- For greater Z_{out} , Greedy performs better.

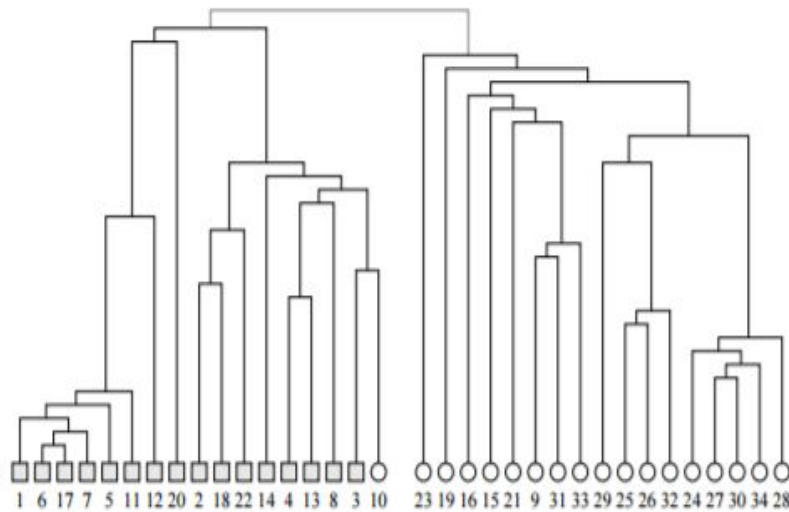


Karate Club Network

- Friendships between 34 members of a club at a US university over a 2-year period.
- The club naturally split into 2 due to some disputes.

Observations:

- Peak modularity $Q = 0.381$
- Vertex 10 classified wrongly.
- GN algorithm classified vertex 3 wrongly.



Games Network

- Schedule of games between American college football teams in a single season.
- Teams divided into groups or conferences.
- Intra-conference games more frequent than inter-conference games.

How much faster does a
greedy algorithm perform
in comparison to the
Girvan and Newman
algorithm?

Some comparisons:

- For the Games Network:
Greedy algorithm : Less than 100th of a second.
GN algorithm: Little over a second
- 1275-node Jazz musicians network:
Greedy algorithm : 1 second
GN algorithm : 3 hours.

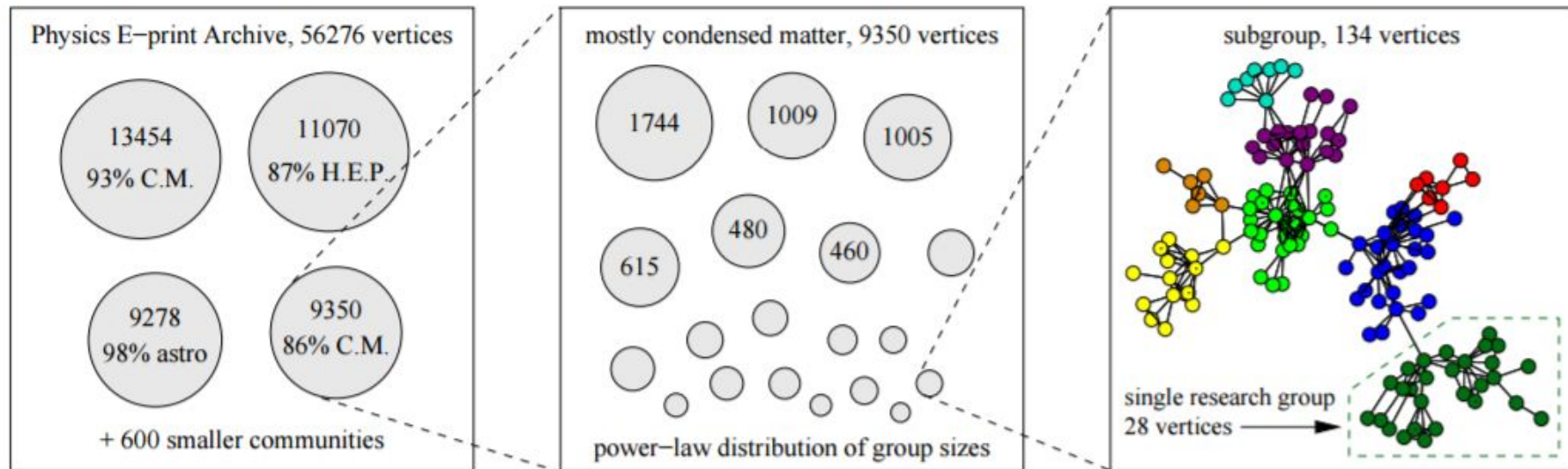
42 minutes! vs. 3-5 years!

Physicists network

- Network of collaborations between physicists as documented by papers posted on the widely-used Physics E-print Archive at arxiv.org
- 56,276 scientists in all branches of physics
- Scientists are considered connected if they have coauthored one or more papers posted on the archive

Observations:

- 600 communities
- Peak modularity of $Q = 0.713$
- 4 large communities - 77% vertices
- Astrophysics, High-energy physics, 2 of condensed matter physics



Greedy algorithm conclusions:

- The author uses a bottom-up approach
- The algorithm can be recursively applied to identify intricate groups and communities.
- The algorithm is computationally feasible
- It is possible to study larger networks now.

Paper 2 :

Modularity and community structure in networks (PNAS'06)

Author: M. E. J. Newman

University of Michigan, Ann Arbor

What is a good division for community detection?

A good division **is not** merely the one that has **fewer** edges between communities;

A good division **is** the one that has **fewer than expected** edges between communities!

What purpose does Modularity serve here?

Modularity quantifies the idea that, a true community structure in a network corresponds to a statistically surprising arrangement of edges

The paper reformulates modularity **in terms of the spectral properties** of the network.

Spectral properties?

Properties of a graph in relationship to the eigenvalues and eigenvectors of matrices associated to the graph, such as its adjacency matrix.

Eigenvalues and Eigenvectors

- If A is a matrix, then the **eigenvalues** are given by the solution of λ such that

$$|A - \lambda I| = 0 \quad \text{---(1)}$$

where I is identity matrix.

- For each value of λ obtained from eq. 1, the following equation gives an **eigenvector**:

$$Av = \lambda v \quad \text{---(2)}$$

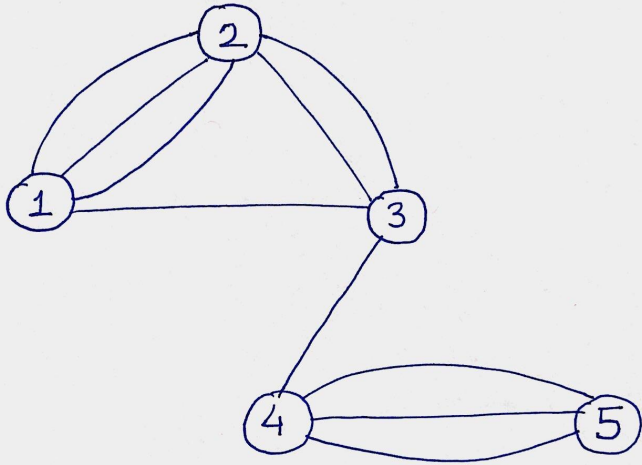
Spectral Algorithm

- Start with the adjacency matrix A of the network
- Convert A into modularity matrix B
- Find all eigenvalues and eigenvectors of B
- Consider the eigenvector corresponding to the highest eigenvalue and use the signs of elements in this vector to decide the group.

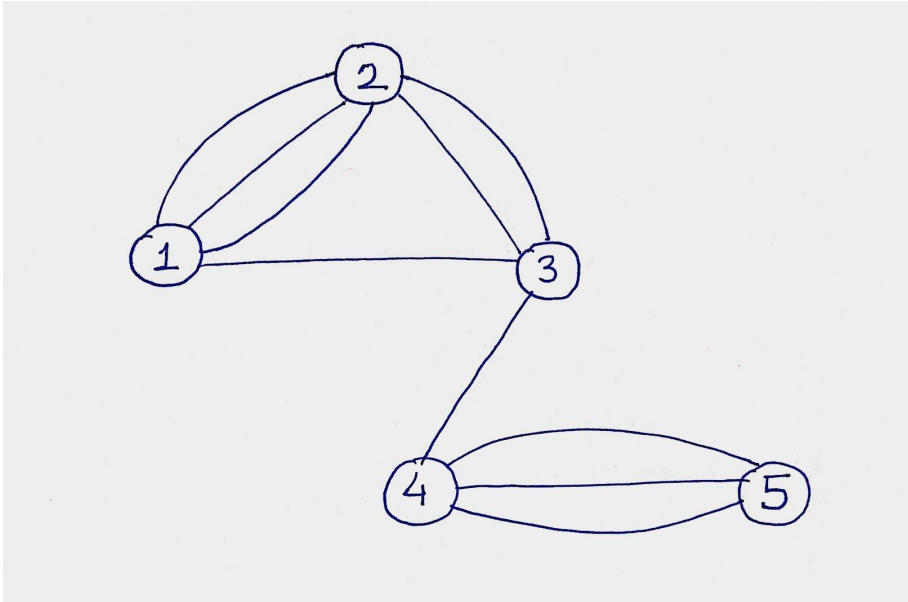
Steps to follow

1. Calculate the e_{ij} matrix for the graph
2. Calculate ΔQ for every possible combination of 2 nodes.
3. Join the vertices with highest value of ΔQ
4. Calculate Q after the join and update the dendrogram
5. Recompute the e_{ij} matrix

Example:



Example:



Adjacency matrix

$$A = \begin{bmatrix} 0 & 3 & 1 & 0 & 0 \\ 3 & 0 & 2 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 3 \\ 0 & 0 & 0 & 3 & 0 \end{bmatrix}$$

Calculating Modularity Matrix

Modularity matrix B is a real symmetric matrix with elements

$$B_{ij} = A_{ij} - k_i k_j / 2m \quad \text{where,}$$

- A_{ij} is the adjacency matrix
- k_i and k_j are the degrees of vertices
- m is $\frac{1}{2} \sum_i k_i$

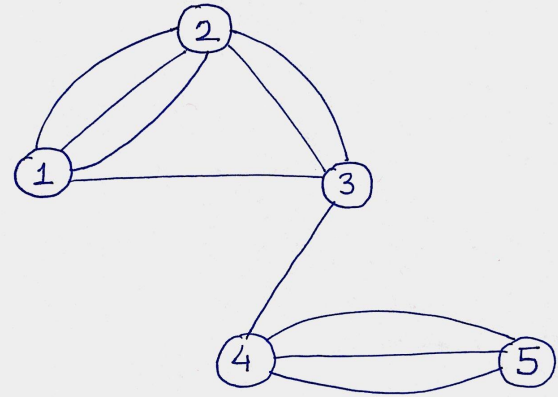
Calculating Modularity Matrix

$$B_{12} = A_{12} - k_1 k_2 / 2m$$

$$= 3 - 4 \cdot 5 / 2 \cdot 10$$

$$= 3 - 1$$

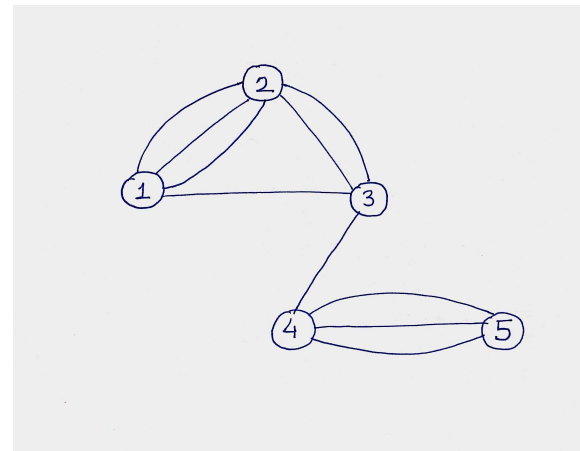
$$= 2$$





Calculating Modularity Matrix

Doing the same calculation for all B_{ij} , we get

$$B = \begin{bmatrix} -0.8 & 2 & 0.2 & -0.8 & -0.6 \\ 2 & -1.25 & 1 & -1 & -0.75 \\ 0.2 & 1 & -0.8 & 0.2 & -0.6 \\ -0.8 & -1 & 0.2 & -0.8 & 2.4 \\ -0.6 & -0.75 & -0.6 & 2.4 & -0.45 \end{bmatrix}$$



Eigenvalues and Eigenvectors for B

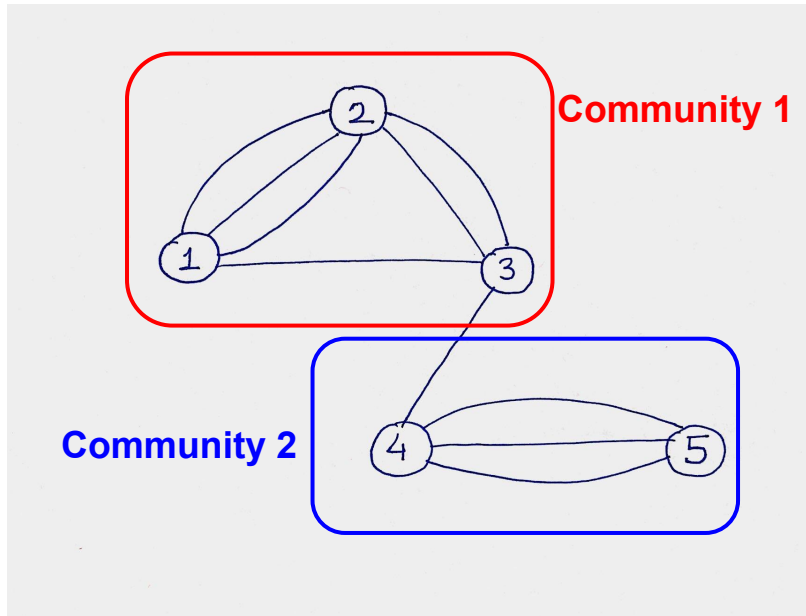
		Eigenvalues	Corresponding eigenvectors
Leading eigenvector		$\lambda_1 \cong 3.95139$	$v_1 \cong (0.442855, 0.499515, 0.199318, -0.511499, -0.502997)$
		$\lambda_2 \cong -2.61989$	$v_2 \cong (0.144319, -0.295586, 0.292573, -0.654337, 0.614854)$
		$\lambda_3 \cong -2.09078$	$v_3 \cong (0.629671, -0.664442, 0.152521, 0.237745, -0.286785)$
		$\lambda_4 \cong 0.82013$	$v_4 \cong (0.387237, 0.451396, 0.436317, 0.501464, 0.452162)$
Always present!		$\lambda_5 \cong 0$	$v_5 \cong (1, 1, 1, 1, 1)$

Splitting into communities

Leading eigenvector is the one with highest eigenvalue

Node number:		1	2	3	4	5
$\lambda_1 \approx 3.95139$	$v_1 \approx$	(0.442855, 0.499515, 0.199318,			-0.511499, -0.502997)	
		+ve	+ve	+ve	-ve	-ve
Community:		A	A	A	B	B

Resulting community structure division



What else do the eigenvectors suggest?

- The elements of the leading eigenvector measure how firmly each vertex belongs to its assigned community.
- Large vector elements \Rightarrow strong central members of their communities.
- Small vector elements \Rightarrow more ambivalent

Special case!

- There is always an eigenvector $(1, 1, 1, \dots, 1)$ with eigenvalue 0 for every modularity matrix B
- This is a property of matrices having sum of rows and columns 0
- If all the eigenvectors have negative eigenvalues, then this special eigenvector acts as the leading eigenvector
- The network is said to be **INDIVISIBLE** in this case
- Big highlight point of this algorithm as many algorithms divide into communities even when it is not needed, based on sheer number of edges between nodes (if they are less)

More than two communities?

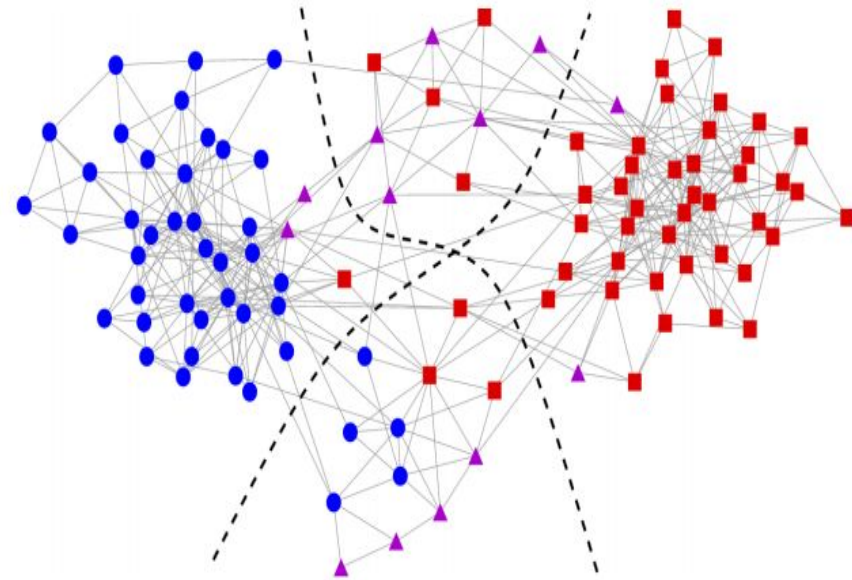
- **Possible** - because this algorithm knows when to stop dividing!
- Start the algorithm with original network
- Run the same algorithm over the newly formed communities
- Continue unless all the communities obtained are **indivisible**

Example verified in the paper

- A network of books about politics, compiled by V. Krebs.
- Vertices represent 105 recent books on American politics bought from the on-line bookseller Amazon.com
- Edges join pairs of books that are frequently purchased by the same buyer.
- Books were divided (by author) according to their stated or apparent political alignment, liberal or conservative, except for a small number of books that were explicitly bipartisan or centrist, or had no clear affiliation.

Result & Interpretation

- Two strong communities are formed - one consists almost entirely of conservative books and other liberal.
- Centrist books belong to their own communities and are not, in most cases, as expected
- Indicates that political moderates form their own purchasing community.
- Books appear to form communities of copurchasing that align closely with political views



● Liberal ■ Conservative ▲ Centrist

Fig. Dashed lines divide the **four communities** found by the algorithm, and shapes represent the political alignment of the books.

How better does leading
eigenvector method do
in comparison to the
Girvan and Newman
algorithm?

Context

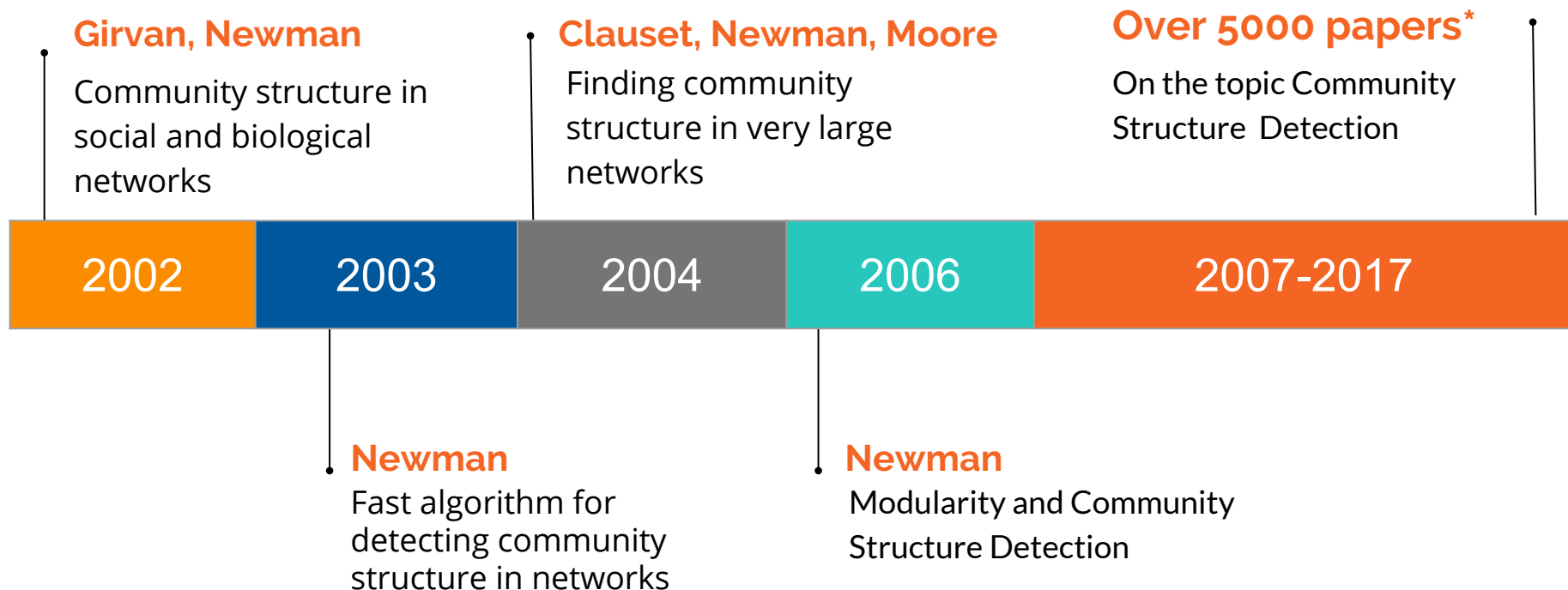
Collaboration
network of
27000 vertices

20 minutes! vs. 2-3 years!

Performance

- Fast and accurate
- Most time consuming part is evaluation of leading eigenvector of modularity matrix
- For n nodes
 - Complexity of $O(n^2 \log n)$
 - Better than all except Greedy - $O(n \log^2 n)$
 - But quality is better than Greedy, especially in large networks
- Takes ~20 min to process network with ~27,000 vertices on a standard PC (circa 2006)

Milestones in Community Structure Detection



Conclusion

Both greedy and spectral algorithm have their own advantages and shortcomings, but both of them have been strong foundational algorithms in the starting days of Community Structure Detection back in early 2000s.

While spectral has more accurate results, greedy is faster. Both make use of the concept of Modularity successfully to get meaningful & insightful results.

References

Community structure
in social and biological
networks (PNAS' 02)

Modularity and
community
structure in
networks.
(PNAS'06)

Finding community
structure in very
large networks
(arxiv'04)

Wikipedia

Fast algorithm for
detecting
community
structure in
networks
(arxiv'03)



Thank you!

~~We are so tired! :/~~

Questions? :)