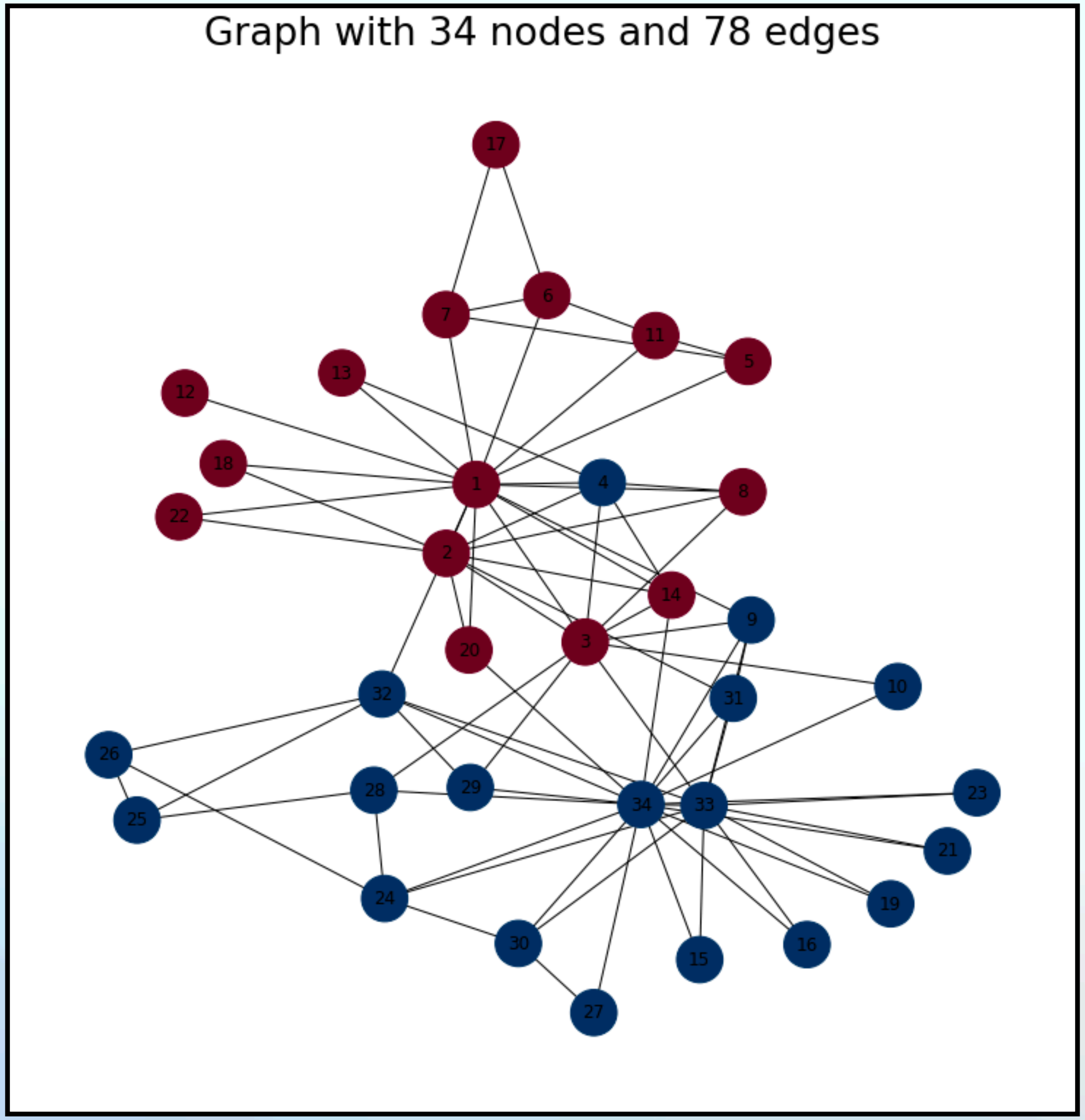
Community Detection And Community Optimization

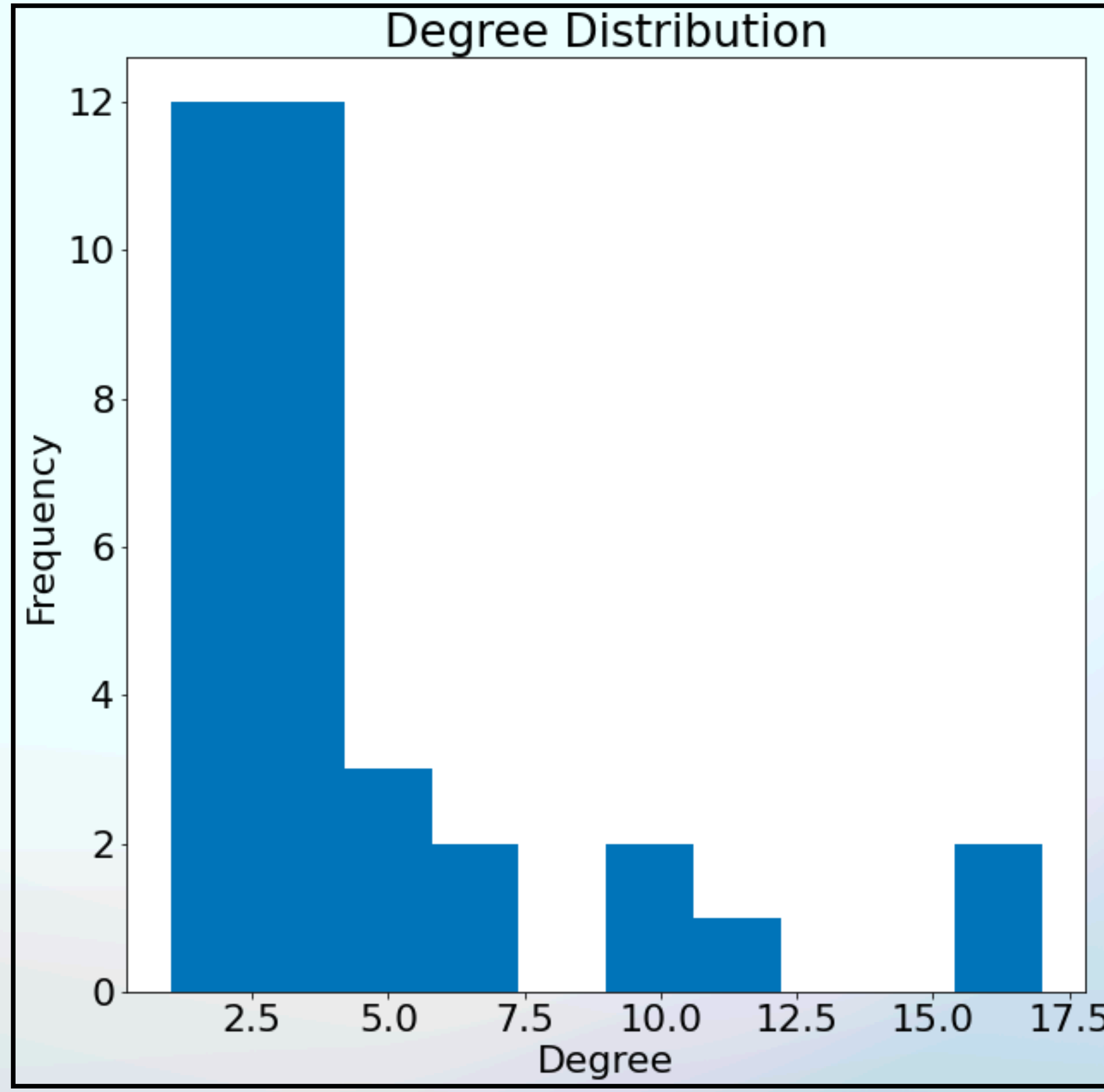
Content

- Modularity
- Modularity Maximation:
    - Newman's FastGreedy Algorithm
    - Louvain Algorithm
- Resolution Limit
- Conductance

Modularity

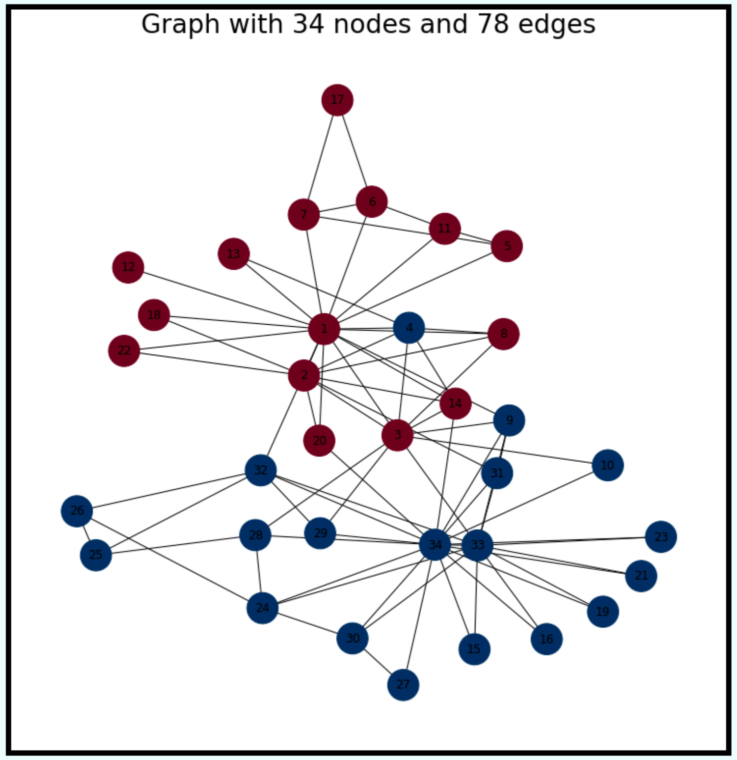
Demo: Zachary Karate graph and its ground-truth communities

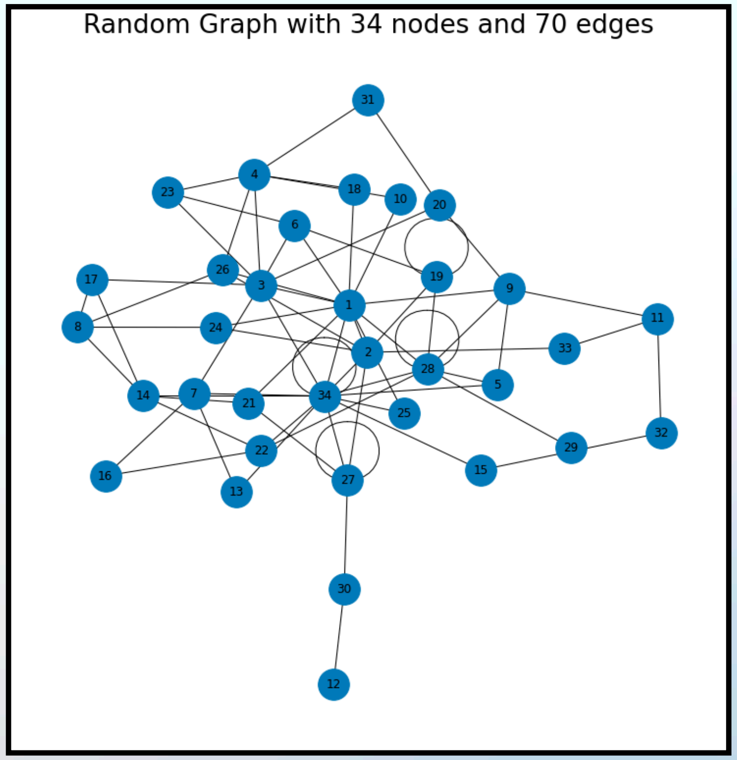




Modularity

- How do we measure modularity?
    - Compare the topology of the random network to the given network
    - how condensed is the graph?
- measures the density of links inside communities compared to links between communities
- Basic Hypothesis:
    - Random network lacks inherent community structure
    - Measure how clustered our network is relative to what's randomly expected
- Usually used for optimization or relative comparison





Modularity

Computing Modularity

$$M = \frac{1}{2m} \sum_{\forall u,v \in C} \left(A_{uv} - \frac{d(u)d(v)}{2m} \right)$$

- Where,
    - $M \rightarrow$ Modularity
    - $m \rightarrow |E(G)|$, number of Edges
    - $\forall u, v \in C \rightarrow$ vertex pairs in the community
    - $A \rightarrow$ Adjacency Matrix
    - $\frac{d(u)d(v)}{2m} \rightarrow$ expected # of edges between u and v in a random graph
- Value in between: $[-1/2, 1]$

Modularity Maximization

- Maximize modularity as a community detection algorithm
- Usually: Greedy Agglomerative
  - Each observation starts in its cluster, and greedily, pairs of clusters are merged as one moves up the hierarchy.
    - Newman Algorithm
    - Louvain Algorithm

Modularity Maximization

Newman's FastGreedy Algorithm

- Greedy Agglomerative Algorithm:
    - Initially: all vertices in unique communities
    - Iterate while # communities > 1:
        - Merge community pair that maximizes modularity
- Pros:
    - Encapsulates the hierarchy
- Cons:
    - Issues with modularity calculations in practice

Modularity Maximization

Louvain Algorithm

- Similar to the Newman Algorithm
    - But with explicit edge contractions
- Initially, each node in the network is assigned a community
- Edge contraction:
    - For each node, compute the difference in modularity if it is placed in its neighbors' community. Move if there is any gain.
    - Contract all the nodes within the communities to a "super-node"
- Repeat the edge contraction method until modularity does not increase

Issues with Modularity

Resolution Limit

- It cannot resolve relatively small communities!

$$M = \frac{1}{2m} \sum_{\forall u,v \in C} \left(A_{uv} - \frac{d(u)d(v)}{2m} \right)$$

- the expected number of edges between nodes u and v in a random graph does not adequately scale for detecting smaller communities in large networks

- become insensitive to communities smaller than a certain scale

Issues with Modularity

Resolution Limit

- How small?

- Change in modularity by combining communities A and B

$$\Delta M = \frac{l_{AB}}{m} - \frac{k_A k_B}{2m^2}$$

  - $l_{AB} \rightarrow$ edges between A and B

  - $k_A, k_B \rightarrow$ sum of degrees of vertives in A or B

- Consider:

$$\frac{l_{AB}}{m} = \frac{k_A k_B}{2m^2} \implies l_{AB} = \frac{k_A k_B}{2m}$$

Issues with Modularity

Resolution Limit

- Assuming we gain from merging A and B

$$l_{AB} > \frac{k_A k_B}{2m},\ k_A = k_B = k, \text{ and } \ l_{AB} = 1$$

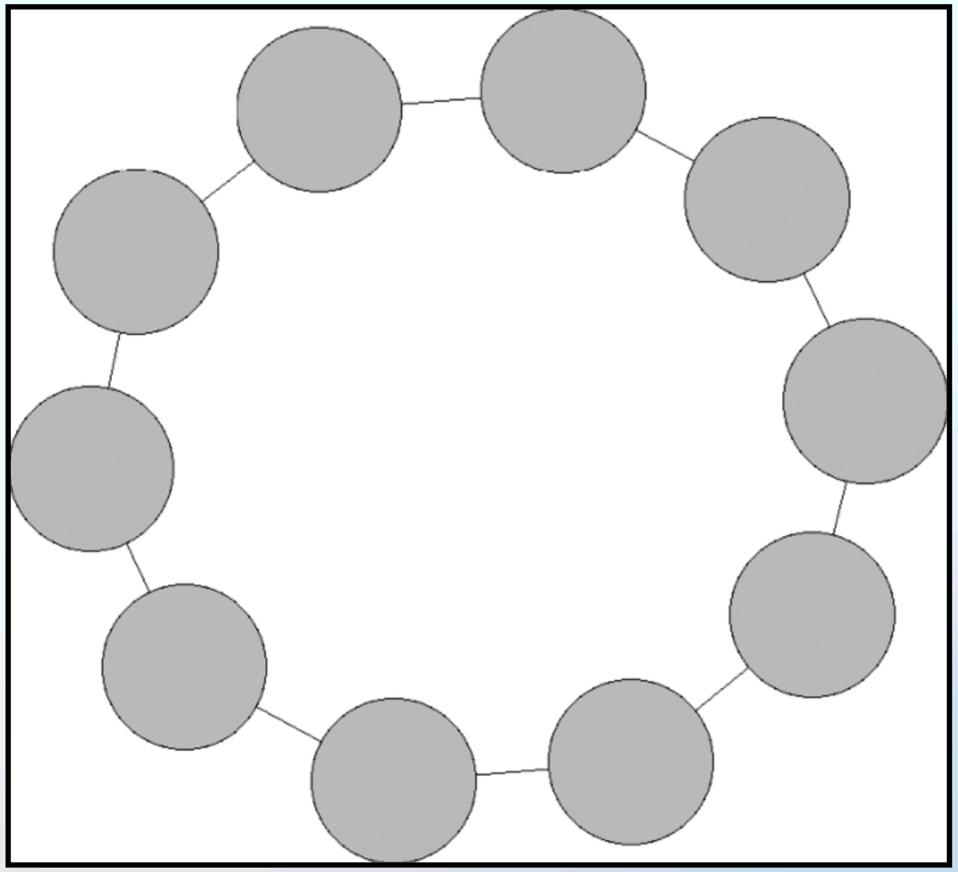
- Then,

$$1 > \frac{k^2}{2m}$$

$$2m > k^2$$

$$\sqrt{2m} > k$$

- Hence, the lower bound on the size of the community that modularity optimization can find is $k \leq \sqrt{2m}$.



Example: Ring of Cliques

Issues with Modularity

Other Issues:

- The expected density of a random network:

$$\frac{d(u)d(v)}{2m}$$

- Different types of networks have different degree distribution:
    - Dense graphs, skewed, or dense subgraphs
- Takeaways: Our modularity value can be meaningless in skewed or dense networks
- Potential Fix: Use the actual attachment probabilities

Conductance

- A measure of how quickly a random walk converges to a stationary state

- Lower Conductance $\implies$ More defined communities

- Defined in terms of edge cut, S and $\bar{S}$:

- Conductance (S) = $\dfrac{\text{cut}(S)}{\min(K_S, K_{\bar{S}})}$

    - $\text{cut}(S) \rightarrow$ number of edges in cut

    - $K_S \rightarrow$ sum of degrees in S

Conductance

- A measure of how quickly a random walk converges to a stationary state
- Lower Conductance $\Longrightarrow$ More defined communities
- Defined in terms of edge cut, S and $\bar{S}$:
- Conductance (S) = $\dfrac{\text{cut}(S)}{\min(K_S, K_{\bar{S}})}$
    - $\text{cut}(S) \rightarrow$ number of edges in cut
    - $K_S \rightarrow$ sum of degrees in S

Review

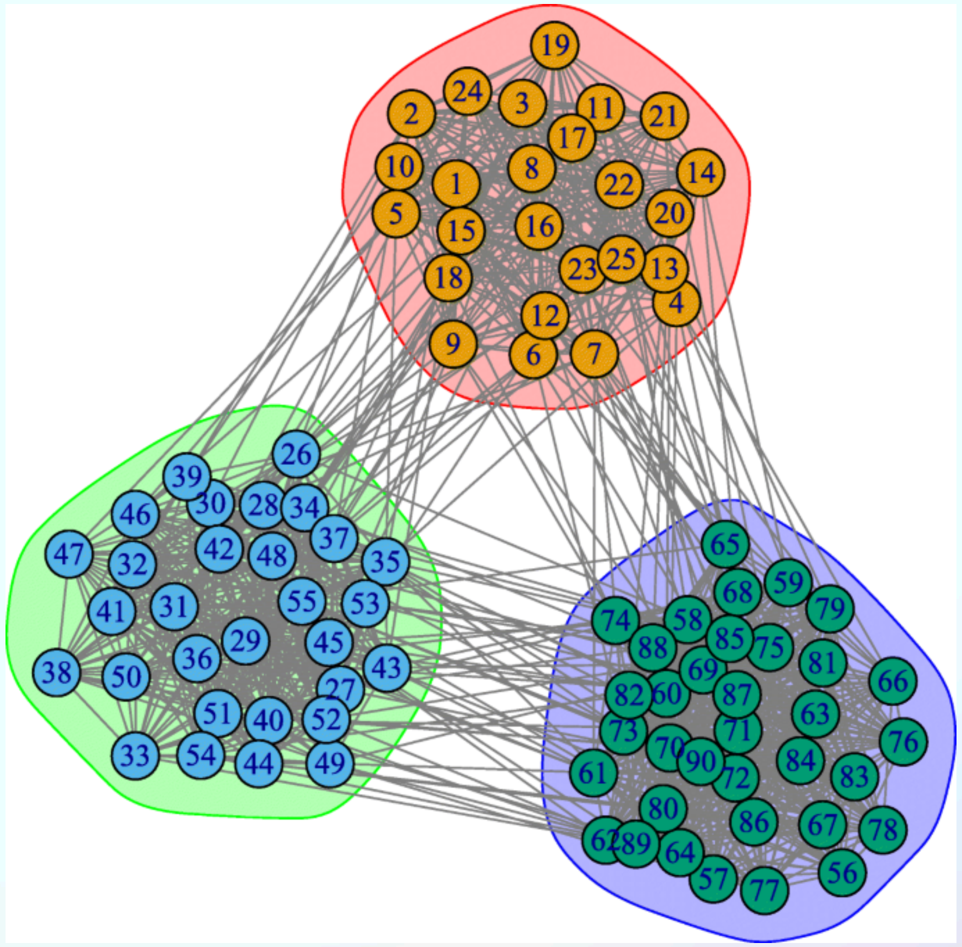
- Evaluating community detection algorithms using
  - **modularity calculation**
    - Issues with modularity maximization
    - Resolution limit
  - **edge cuts, edge cut ratio,**
    - Number of edges between different categories
    - Take into account for the size of the community
  - **conductance**
    - How quickly a random walk traverses through the graph

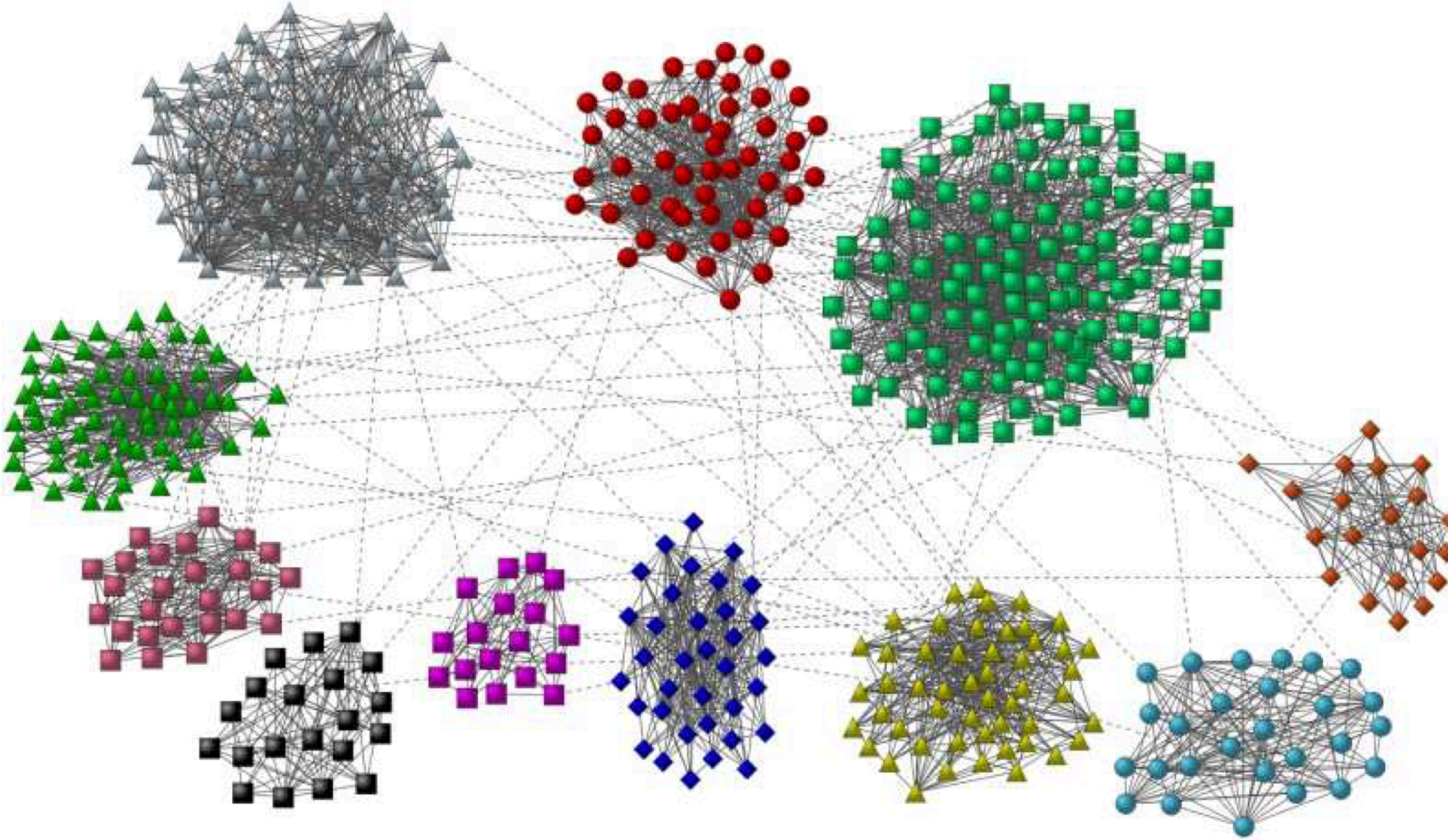
Evaluating Community Detection Algorithms

- **Prior methods only take the topology and Community Detection output into consideration
  - Ideally => Compare against the ground truth
  - Problem:
    - Hard to find within Real-world data
    - Big Companies: Google, Facebook
  - Solution:
    - Generate datasets
    - Generate a graph with a known structure
      - # of communities
      - Size of communities
      - Community coherence

Solution

- Generate a graph with a known structure
  - # of communities
  - Size of communities
  - Community coherence
    - mixing parameter
    - ratio of $\mu = \dfrac{\text{external edges}}{\text{total edges}}$





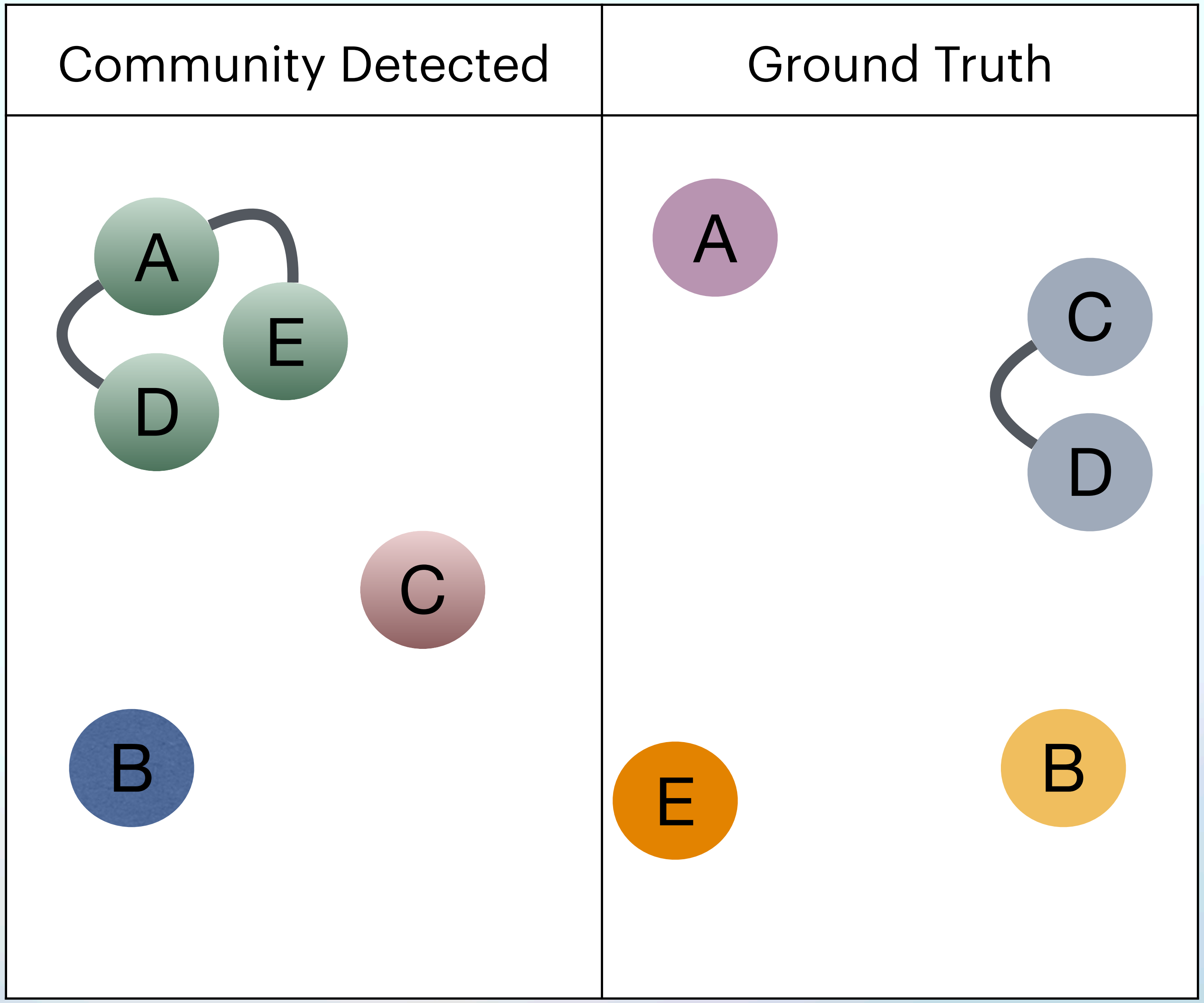
Lancichinetti–Fortunato–Radicchi (LFR) Benchmark

- Gold Standard for evaluating Community Detection Algorithm
- Uses power-law distributions for degrees and community sizes
- assigns vertices → communities
- Wires vertices together(edge) while maintaining μ

Lancichinetti–Fortunato–Radicchi (LFR) Benchmark

Normalized Mutual Information

Index	Node	Community Detected	Ground Truth
1	**A**	1	1
2	**B**	2	2
3	**C**	3	3
4	**D**	1	3
5	**E**	1	4



Lancichinetti–Fortunato–Radicchi (LFR) Benchmark

Normalized Mutual Information

- Set of N elements: $S = \{s_1, s_2, \ldots, s_N\}$ → vertex
- Two Partition:
  - $U = \{u_1, u_2, \ldots, u_a\}$ → Comm. Det.
  - $V = \{v_1, v_2, \ldots, v_b\}$ → Ground Truth

Index	Node	Comm. Det.	Ground
1	**A**	1	3
2	**B**	2	1
3	**C**	3	3
4	**D**	1	3
5	**E**	1	4

- Contingency Table, T
  - Overlapping for all possible $U_i V_j$ pairs
    - $T_{ij} = |U_i \cap V_i|$ → Contingency Table (T)

	1	**2**	**3**	**4**
1			2	1
2	1			
3			1	
4				

Contingency Table (T)

Lancichinetti–Fortunato–Radicchi (LFR) Benchmark

Normalized Mutual Information

Index	Node	Comm. Det. (U)	Ground (V)
1	**A**	1	3
2	**B**	2	1
3	**C**	3	3
4	**D**	1	3
5	**E**	1	4

	1	**2**	**3**	**4**
1			2	1
2	1			
3			1	
4				

Contingency Table (T)

- $P_u(i) = \dfrac{|U_i|}{N}$: random prob. that a node is in U partition
- $P_v(j) = \dfrac{|V_j|}{N}$: random prob. that a node is in V partition
- $P_{UV}(i, j) = \dfrac{T_{ij}}{N}$: random prob. that a node are in both U and V partition

Lancichinetti–Fortunato–Radicchi (LFR) Benchmark

Normalized Mutual Information

- $P_u(i) = \frac{|U_i|}{N}$, $P_v(j) = \frac{|V_j|}{N}$ and $P_{UV}(i,j) = \frac{T_{ij}}{N}$
- Entropy:
- $H(U) = \sum_{i=1}^{R} P_U(i)\log\left(P_U(i)\right)$ and $H(V) = \sum_{j=1}^{C} P_V(j)\log\left(P_V(j)\right)$
- Mutual Information:

$$MI(U,V) = \sum_{i=1}^{R}\sum_{j=1}^{C} P_{UV}(i,j)\log\left(\frac{P_{UV}(i,j)}{P_U(i)P_V(j)}\right) = H(U) - H(U\,|\,V)$$

- Normalized Mutual Information:

$$NMI = \frac{MI}{\frac{1}{2}\left(H(U) + H(V)\right)}$$

Lancichinetti–Fortunato–Radicchi (LFR) Benchmark

Normalized Mutual Information