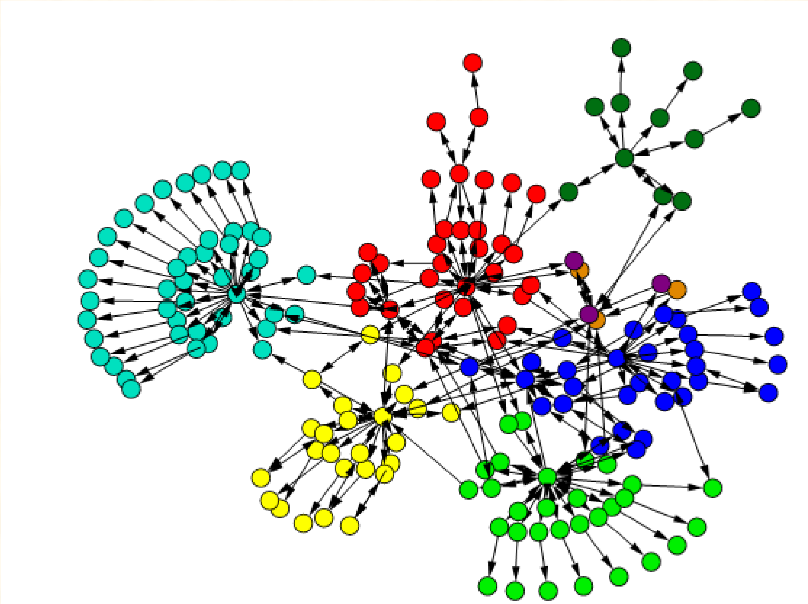
Complex Dynamical Networks:

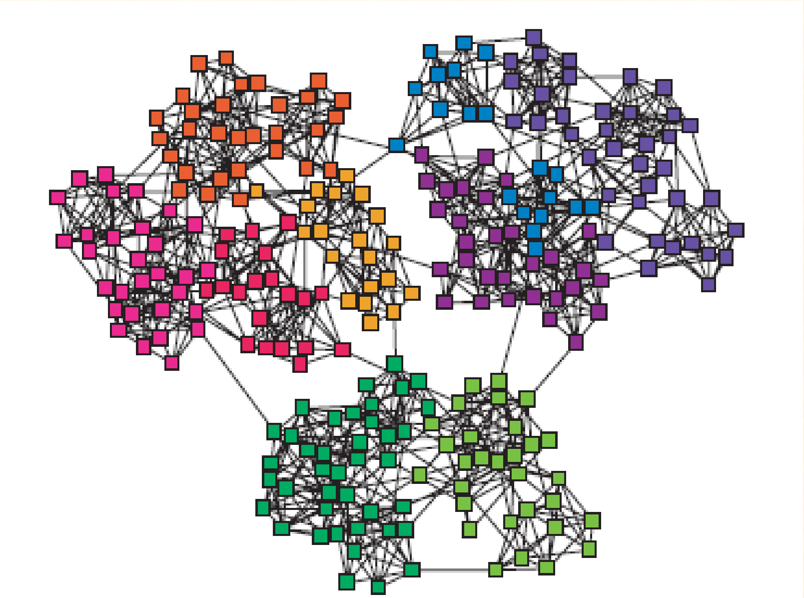
Lecture 6a: Community Structures

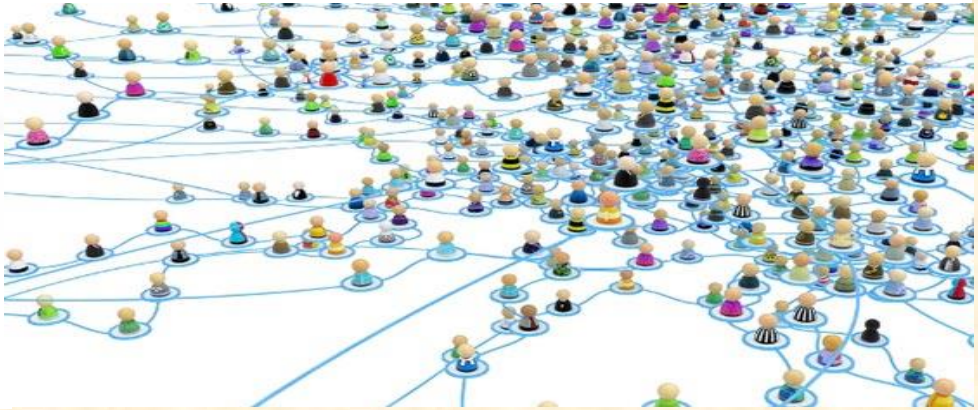
EE 6605

Instructor: G Ron Chen

Community Structure in Complex Networks



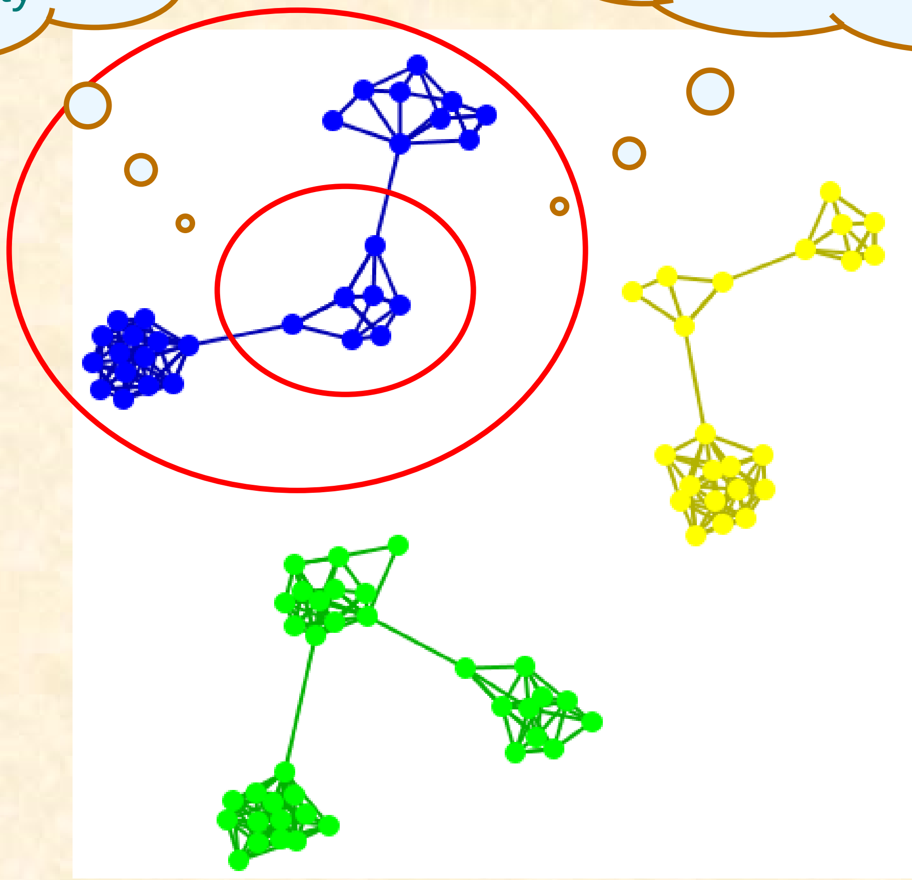




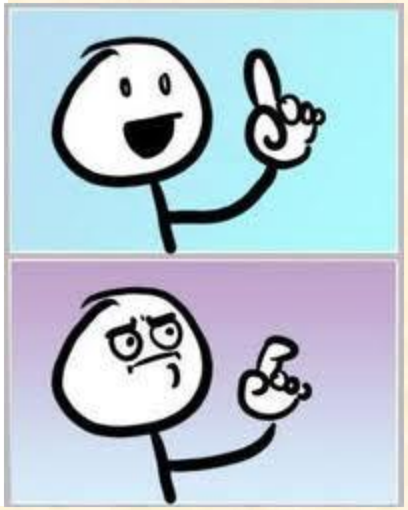
Community

Each densely-linked group is a community

Each symmetrical group is a community



Definition of a community can be subjective



Community Detection

- Common Definition: A community is a set of nodes among which the connections are relatively dense and strong, or interactions are relatively frequent
- A network has a community structure if the network can naturally be divided into clusters of nodes with dense internal connections and sparse external connections
- Community Detection (Clustering, Grouping)

  To find cohesive subgroups from a given graph
- Applications

  Understanding interactions between people (or systems)

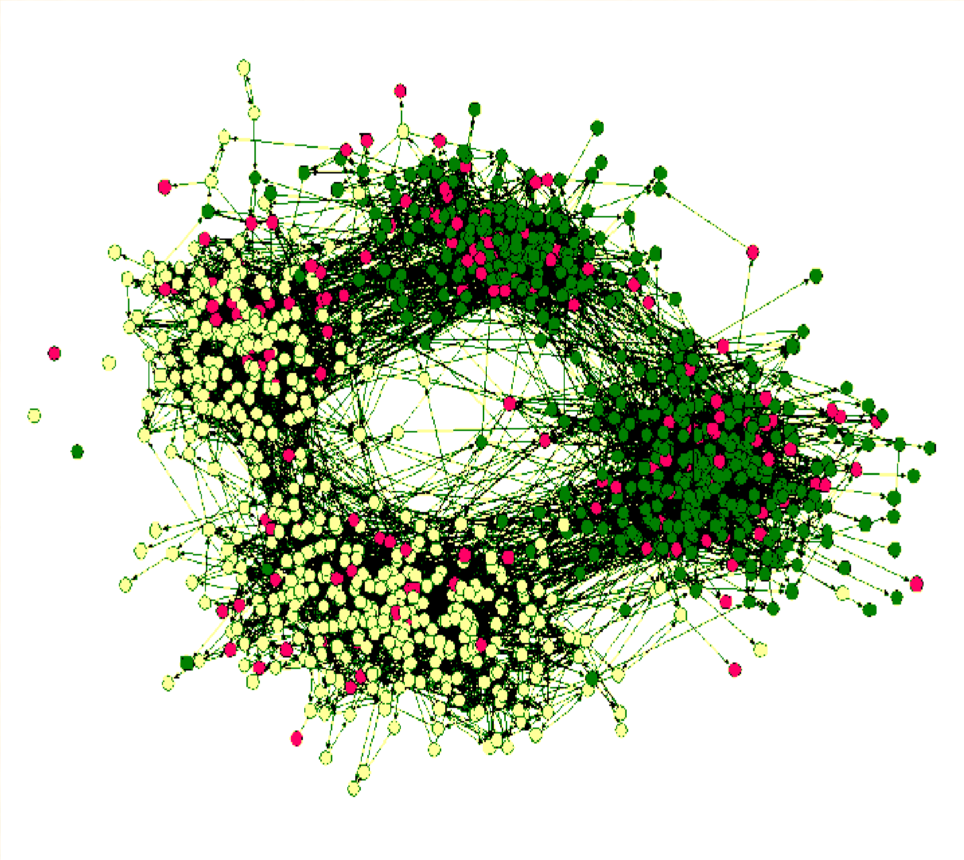
  Visualizing and navigating on large-scale networks

  Forming bases for other tasks such as data mining

  …

Wiki

Example: USA school integration and friendship segregation



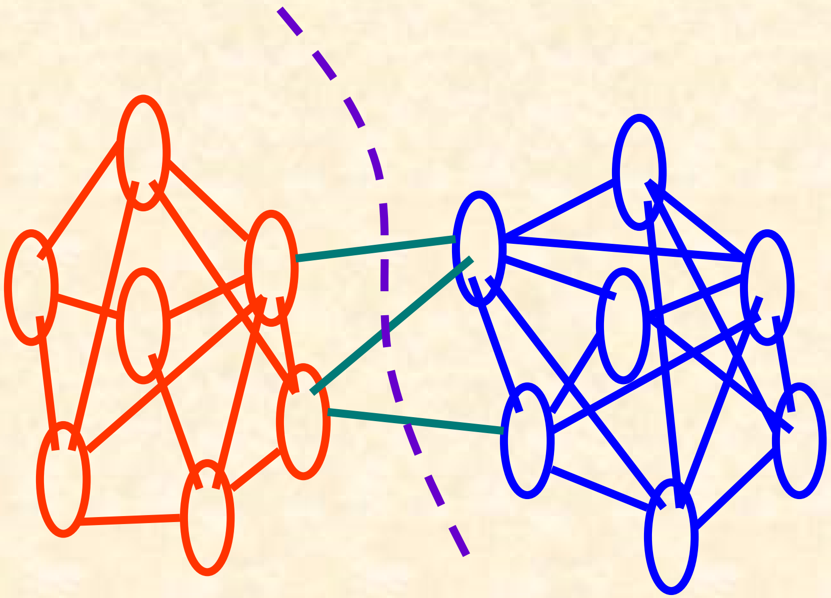
J. Moody, Amer. J. of Sociology, 2001

Race: **left (white) to right (black)**

Grade: **up (junior) to down (senior)**

Community Detection

Many real networks have a natural community structure, and we want to discover this structure



For a large-scale network, how can we discover community structure in an automated way?

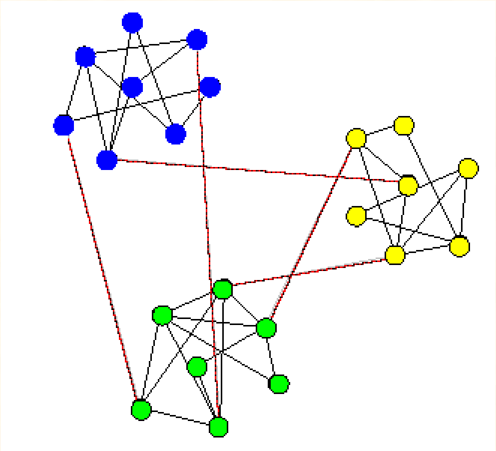
Criteria: Which partition is better **?**

- Quality Measures:

  How to evaluate an algorithm's performance while the community structure is unknown?

- Benchmarks:

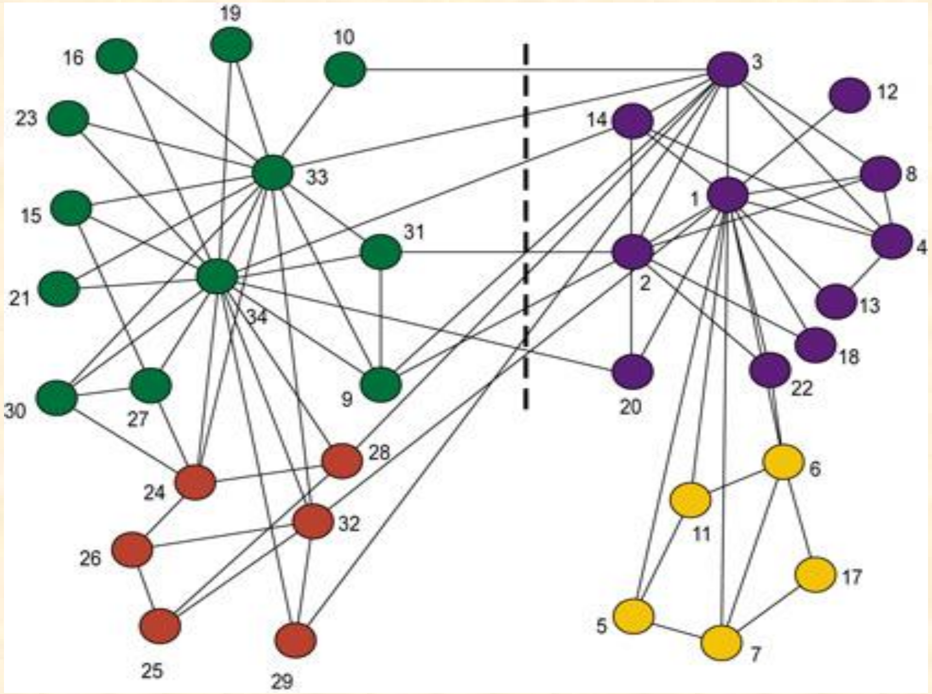
  Which algorithm is the best to characterize a given network with a known community structure?



Benchmark Examples

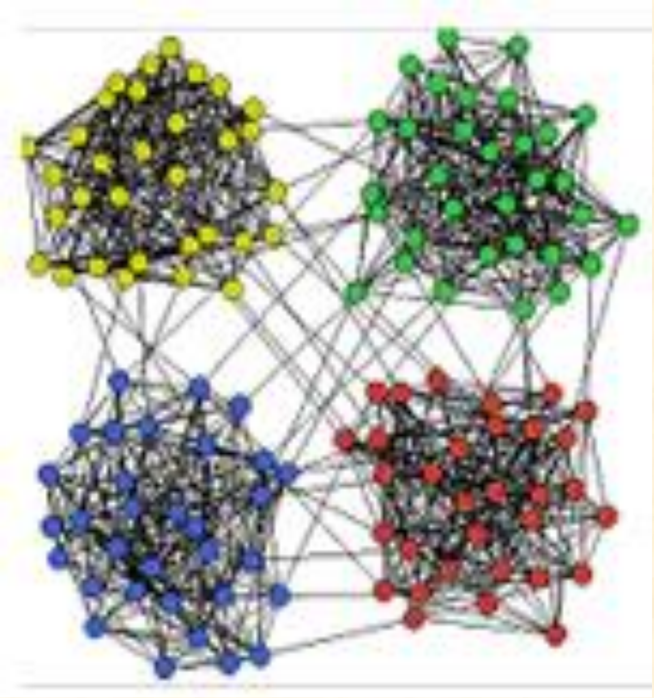
A real network:

Zachary's Karate Network



An artificial network:

Planted L-partition model



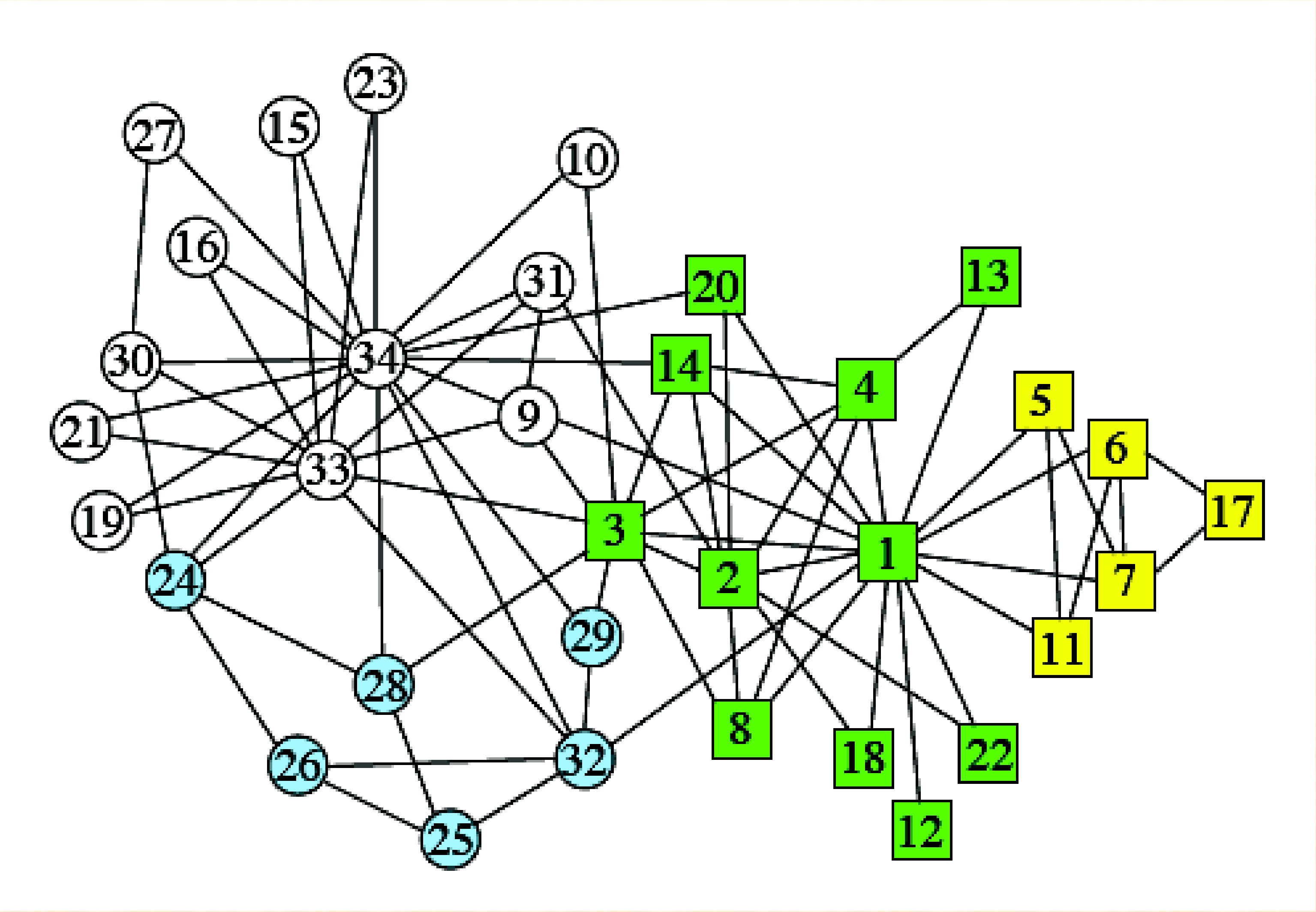
W. W. Zachary, J. Anthropological Research 33: 452-473, 1977

A. Condon and R. M. Karp, Random Structures & Algorithms 18(2): 116-140 , 2001

Zachary's Karate Network

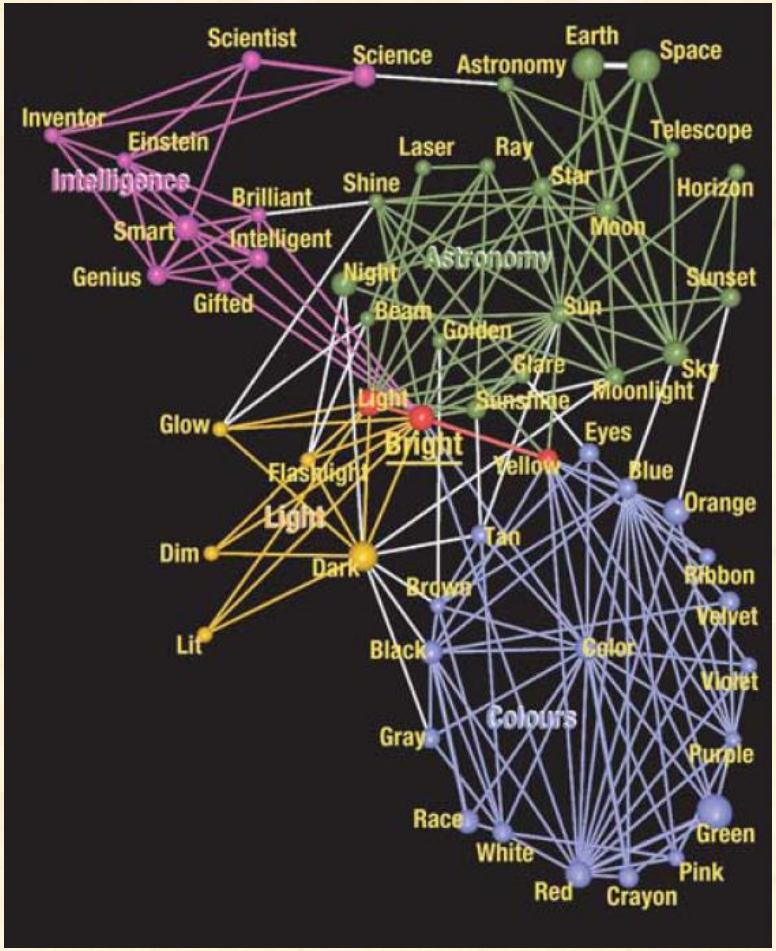
Story Background / Dataset

Zachary's Karate Network



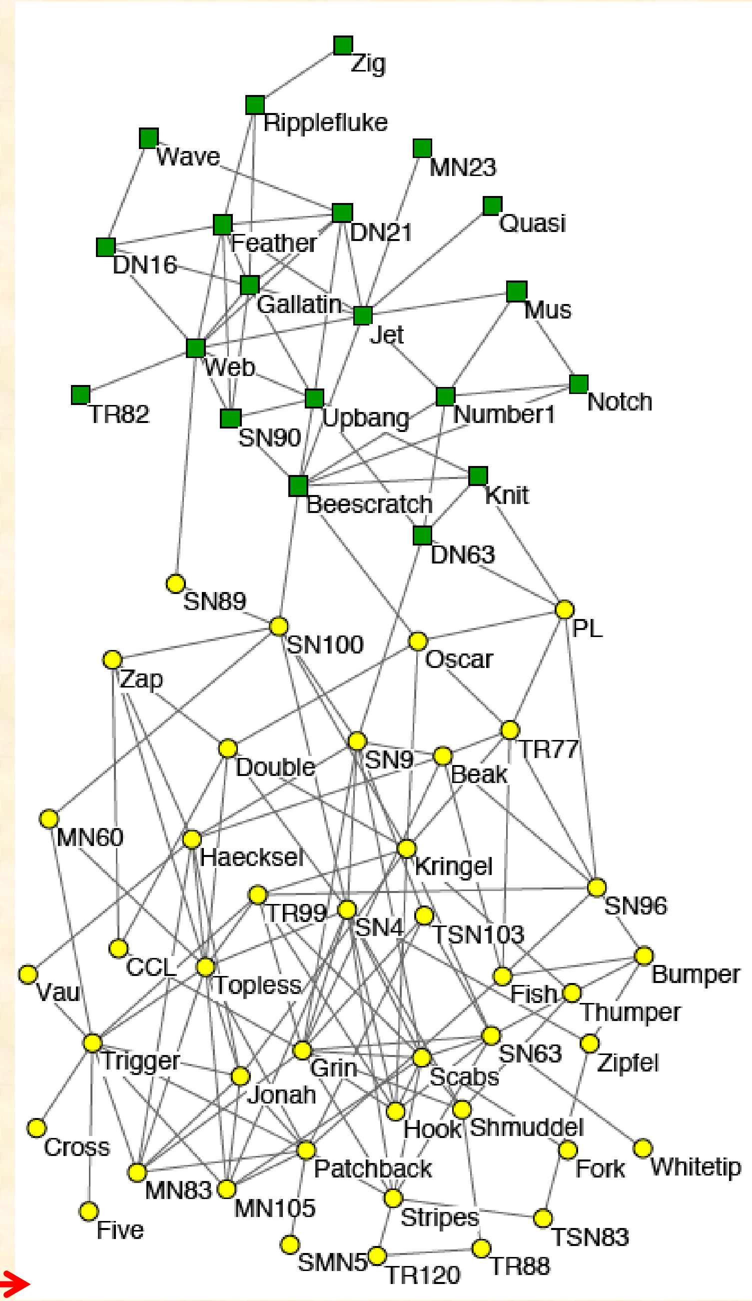
Other

Benchmark Examples

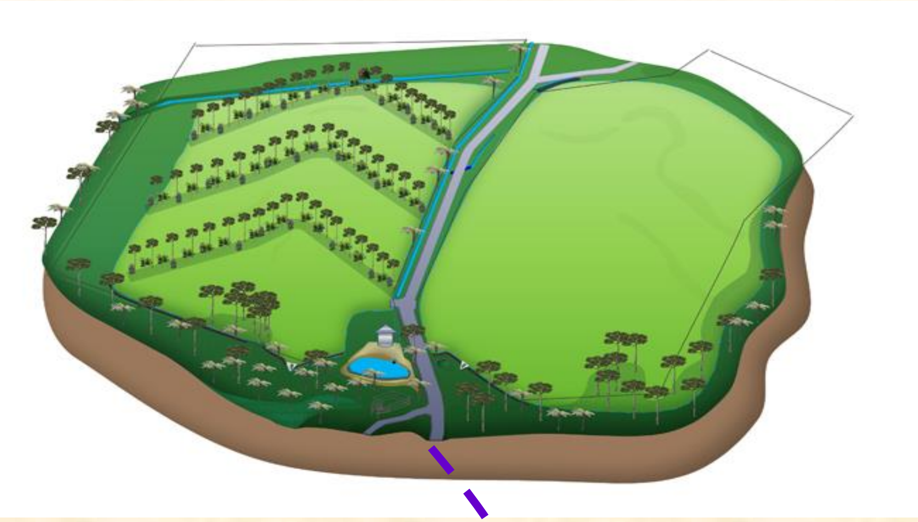


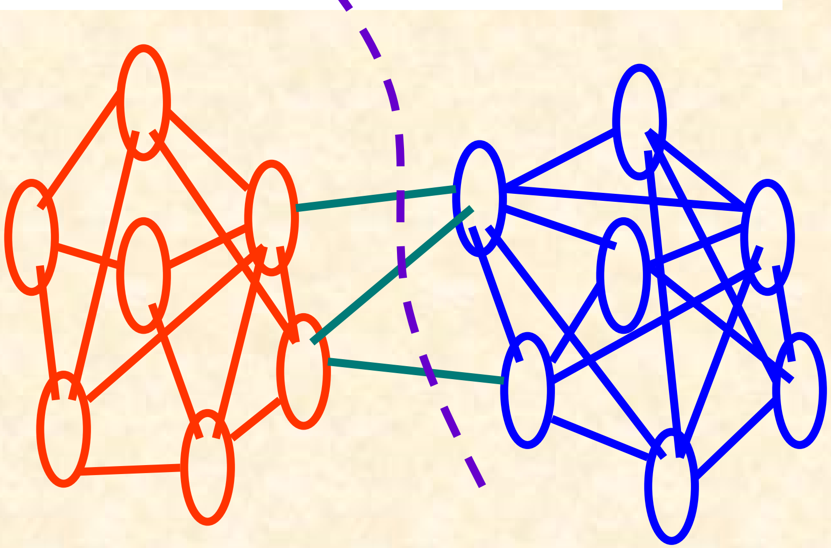
word network

Lusseau's network of bottlenose dolphins



Example: Planted *2*-Partition Model





$L = 2$

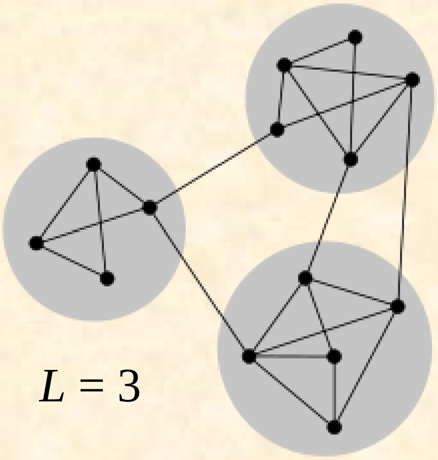
Planted L-Partition Model

Partition the graph into L parts (for example, $L = 3$)

Each node has a probability (or, degree) p_{in} of being connected to nodes inside its group and a probability p_{out} of being connected to nodes outside its group

If $p_{in} > p_{out}$ then the graph has a community structure

Otherwise, it is a homogeneous (e.g., random) graph



$L = 3$

Methods and Algorithms

1) Minimum-cut method

2) Hierarchical clustering

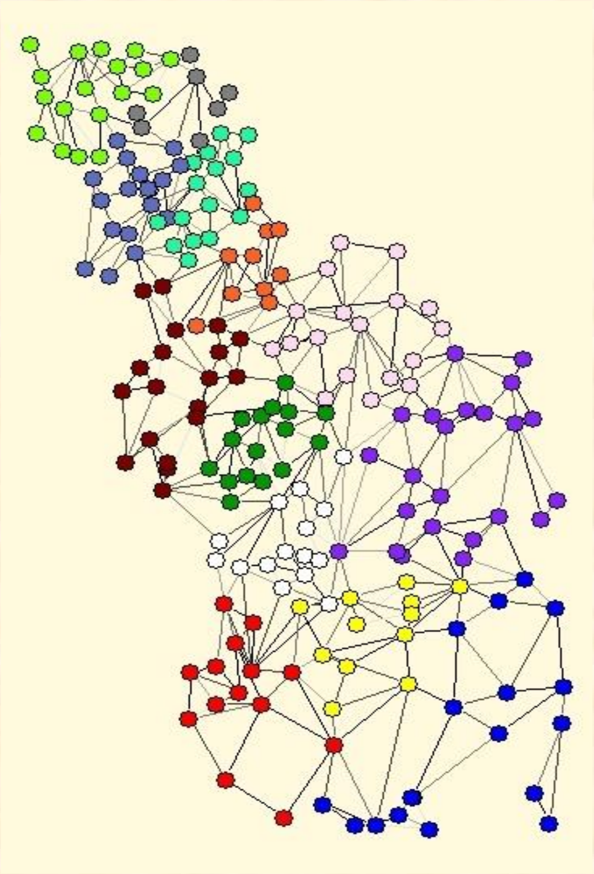
3) GN algorithm

4) Clique-based methods

5) Modularity maximization

6) Information-based algorithms

……



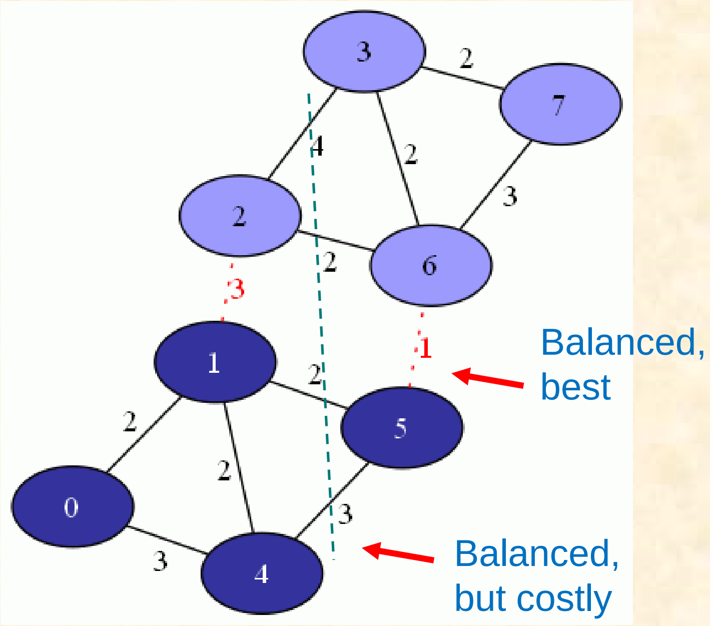
1) Minimum-Cut Method

One of the oldest algorithms for dividing networks into parts

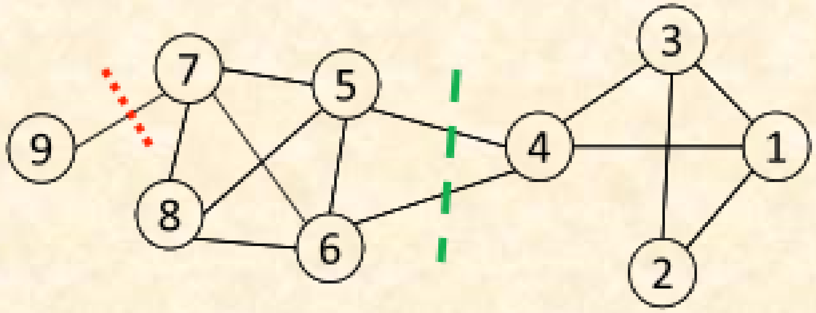
The network is divided into a pre-determined number of groups, in approximately the same size ("balanced"), chosen such that the number or cost (total weights) of edges between groups is minimized

Karger's algorithm

Example:



Ratio Cut and Normalized Cut



- If cost (total weights) of edges between groups are not chosen appropriately, the corresponding minimum-cost cut may yield an unbalanced partition (e.g., with one set being a singleton)

- Other objective functions:

$$\text{Ratio Cut}(\pi) = \frac{1}{k}\sum_{i=1}^{k}\frac{cut(C_i,\bar{C}_i)}{|C_i|},$$

$$\text{Normalized Cut}(\pi) = \frac{1}{k}\sum_{i=1}^{k}\frac{cut(C_i,\bar{C}_i)}{vol(C_i)}$$

C_i: community i

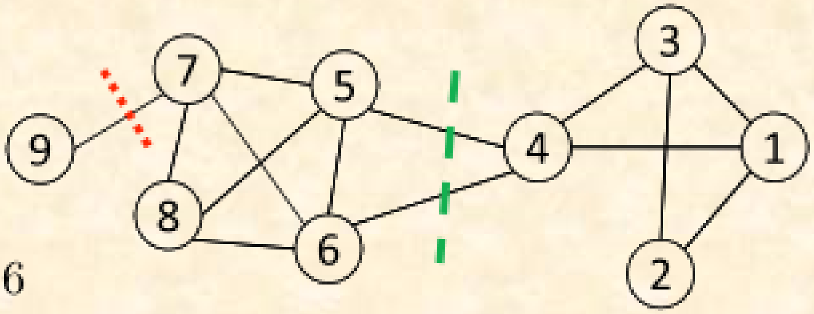
$\overline{C}_i$: complementary community

$|C_i|$: number of nodes in C_i

$vol(C_i)$: sum of degrees in C_i

$cut(C_i, \overline{C}_i)$ = degrees of the cut

Example



For partition by red: π_1

$$\text{Ratio Cut}(\pi_1) = \frac{1}{2}\left(\frac{1}{1} + \frac{1}{8}\right) = 9/16 = 0.56$$

$$\text{Normalized Cut}(\pi_1) = \frac{1}{2}\left(\frac{1}{1} + \frac{1}{27}\right) = 14/27 = 0.52$$

For partition by green: π_2

$$\text{Ratio Cut}(\pi_2) = \frac{1}{2}\left(\frac{2}{4} + \frac{2}{5}\right) = 9/20 = 0.45 < \text{Ratio Cut}(\pi_1)$$

$$\text{Normalized Cut}(\pi_2) = \frac{1}{2}\left(\frac{2}{12} + \frac{2}{16}\right) = 7/48 = 0.15 < \text{Normalized Cut}(\pi_1)$$

→ We should cut π_2 as comparing to π_1

$$\text{Ratio Cut}(\pi) = \frac{1}{k}\sum_{i=1}^{k} \frac{cut(C_i, \bar{C}_i)}{|C_i|},$$

$$\text{Normalized Cut}(\pi) = \frac{1}{k}\sum_{i=1}^{k} \frac{cut(C_i, \bar{C}_i)}{vol(C_i)}$$

$|C_i|$: number of nodes in C_i

$vol(C_i)$: sum of degrees in C_i

$cut(C_i, \bar{C}_i)$ = degrees of the cut

2) Node-Similarity-Based Clustering

Put all nodes with the same (or close) similarity into the same community

- For large-scale networks:
  - Consider the connections as features
  - Use Cosine similarity or Jaccard similarity to compute node similarity
  - Apply the classical K-means Clustering Algorithm
- K-means Clustering Algorithm
  - Each cluster is associated with a centroid (center point)
  - Each node is assigned to the cluster with the closest centroid

Algorithm 1 Basic K-means Algorithm.

```
Select K points as the initial centroids.
repeat
    Form K clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change
```

Node Similarity

Adjacency matrix

Structurally equivalent

	1	2	3	4	5	6	7	8	9	10	11	12	13
5		1				1							
8	1					1							1
9	1					1							1

Others omitted

Cosine Similarity:

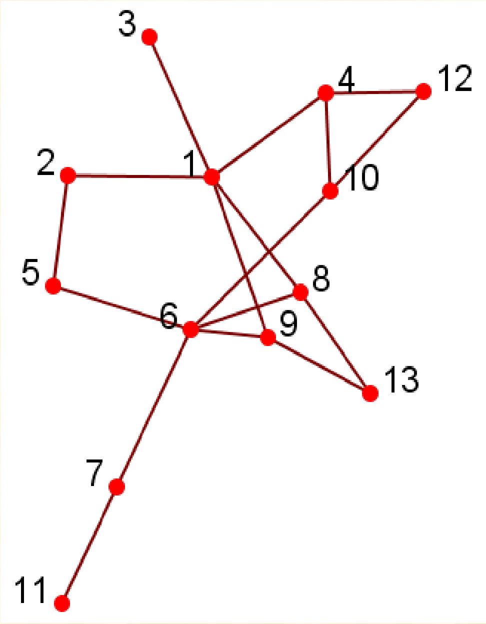
$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}.$$

$$sim(5,8) = \frac{0\times1+1\times0+1\times1+0\times1}{\sqrt{2}\times\sqrt{3}} = \frac{1}{\sqrt{6}}$$

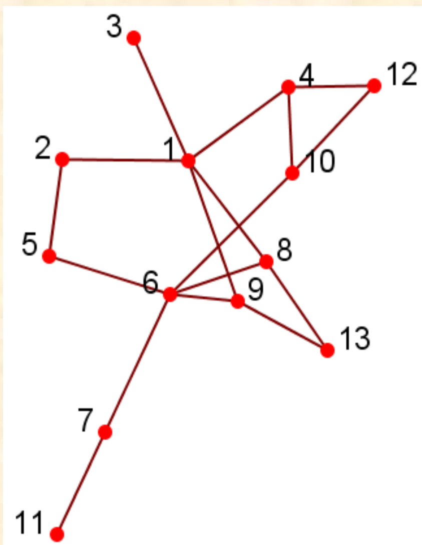
Jaccard Similarity:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}.$$

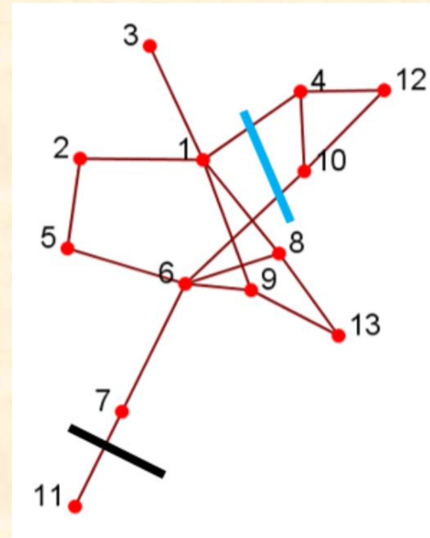
$$J(5,8) = \frac{|\{6\}|}{|\{1,2,6,13\}|} = \frac{1}{4}$$



Node-Similarity-Based Clustering



Cosine,
Jaccard



K-means

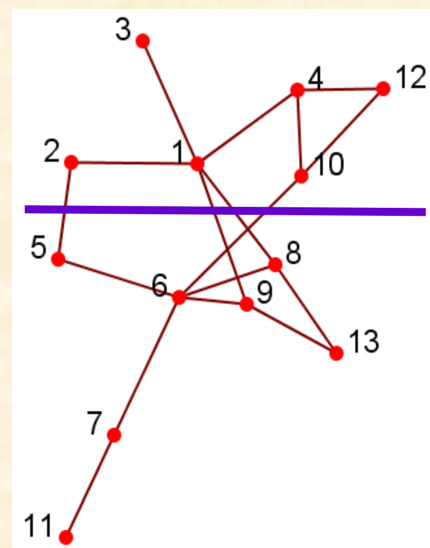
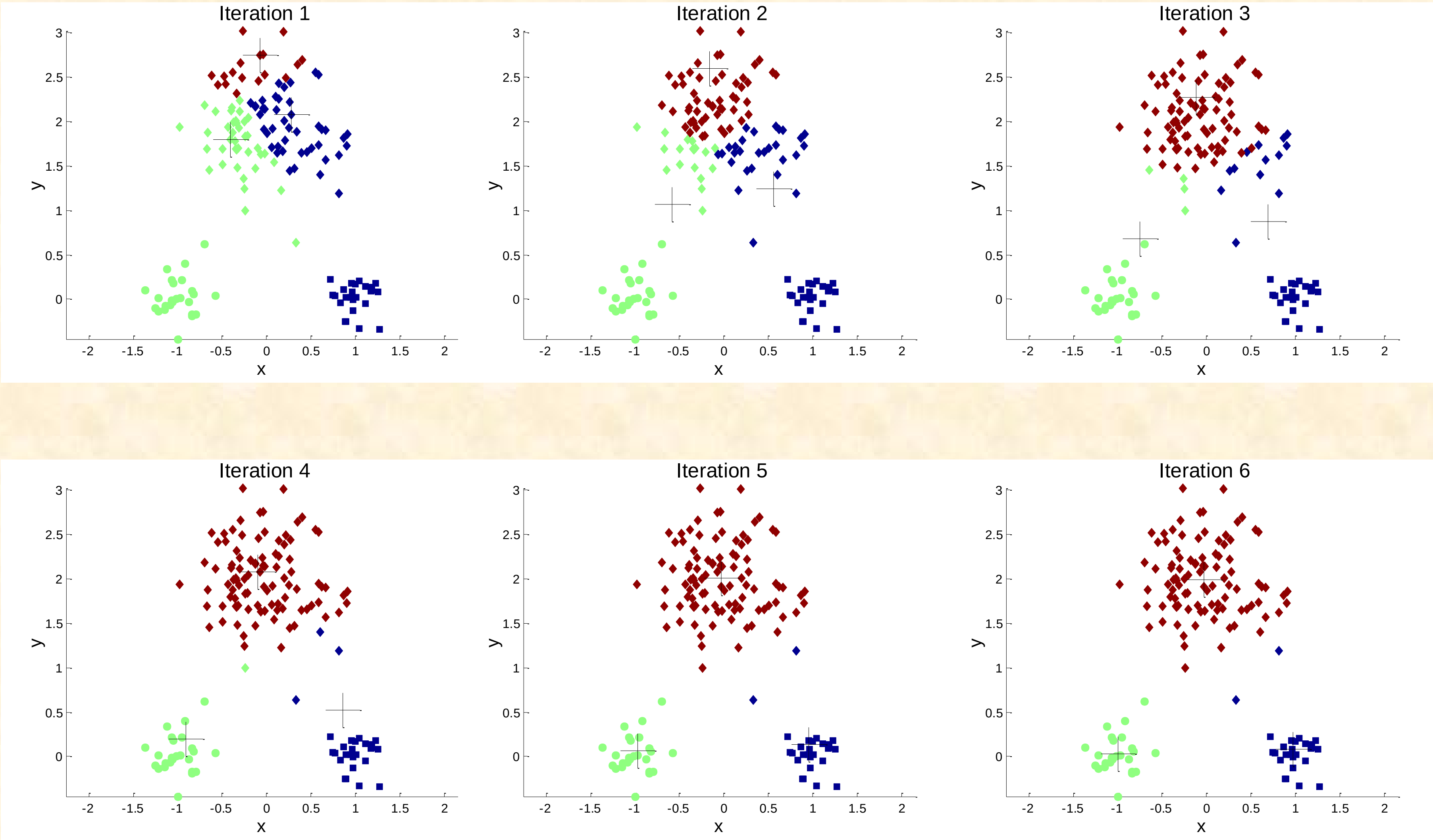


Illustration of K-means Clustering



2) Hierarchical Clustering Method

A similarity measure is used to quantify node pairs

Commonly used measures include: cosine similarity, Jaccard similarity, and Hamming distance between rows of the adjacency matrix, etc.

Then, according to any of such measures, similar nodes are grouped into communities

Example: Hamming distance

between: toned and roses

Is: 3

Example: Hamming distance

between: 100100 and 011011

Is: 6

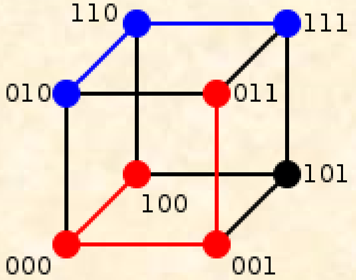
2) Hierarchical Clustering Method

In information theory, the **Hamming distance** between two strings of equal length is the number of positions at which the corresponding symbols are different

Hamming distance:

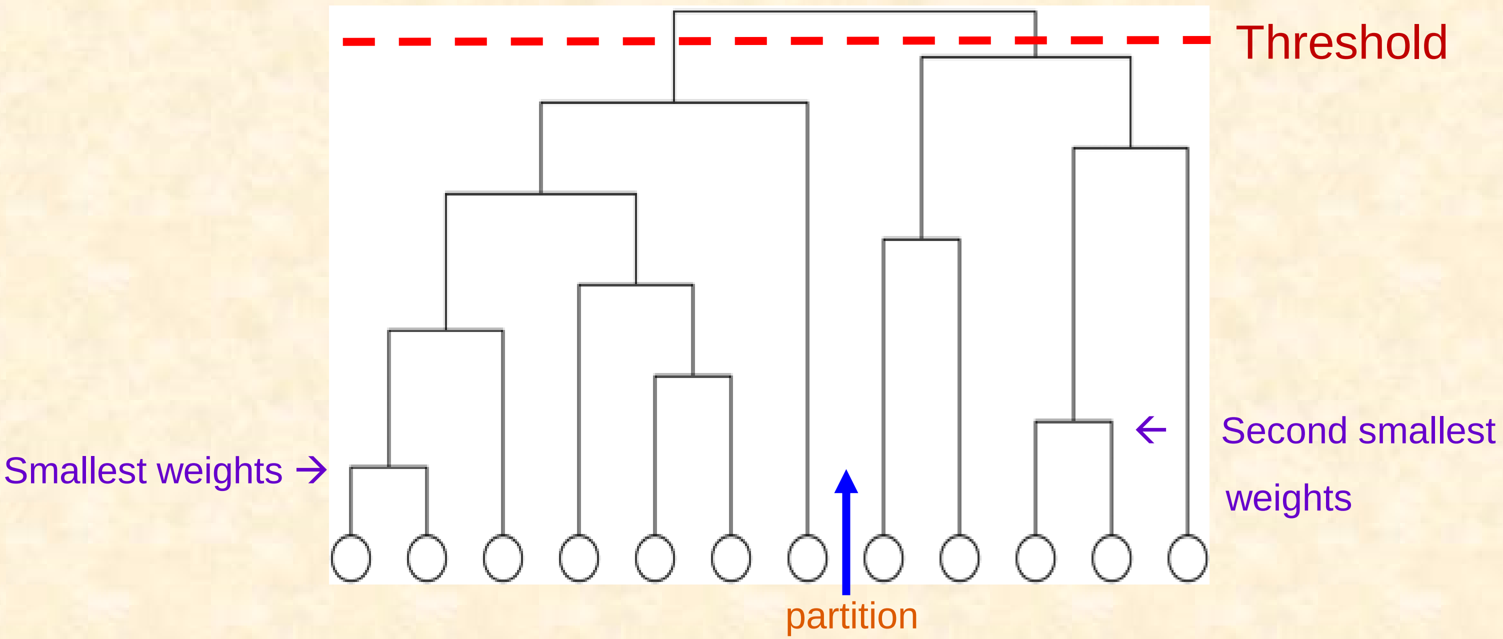
100 → 011 has distance 3 (red path)

010 → 111 has distance 2 (blue path)



Hierarchical Clustering Method

- Calculate a "weight" (e.g., similarity value) for every pair of nodes, which represents how closely connected this pair of nodes is
- Starting with all disconnected nodes, add edges between pairs, one by one, in increasing order of their weights
- Result: Some nested components, where one can take a "slice" (threshold) at any level of the tree



3) GN Algorithm

Girvan–Newman (GN) algorithm identifies "heavy" edges in a network that lie between communities and then removes them, leaving only the communities

This is a betweenness-based clustering method: Identification is performed by employing the edge-betweenness, yielding results of reasonable quality

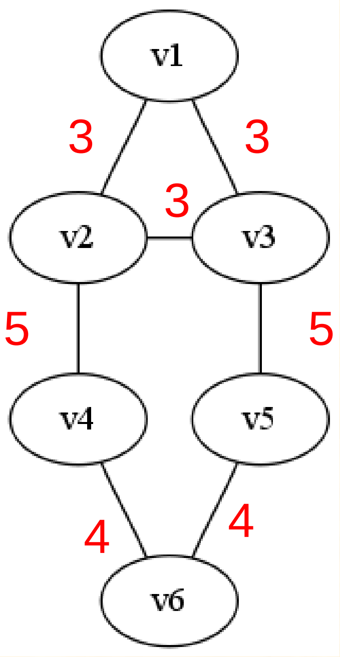
Drawbacks: High computational complexity

GN Algorithm

M. Girvan and M. E. J. Newman, PNAS, **99**(12): 7821-7826, 2002

GN Algorithm

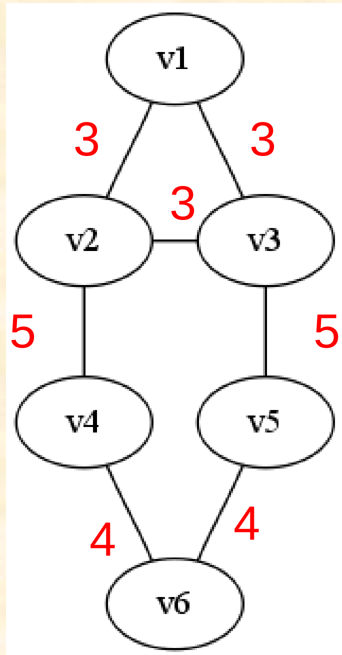
- Calculate all the edge-betweenness in the network
- Remove the edge with the highest betweenness
- Re-calculate all the edge-betweennesses for the resulting (smaller) network
- Repeat the above, until no edge is left



Remove (5) v2-v4, v3-v5

Remove (4) v4-v6, v5-v6

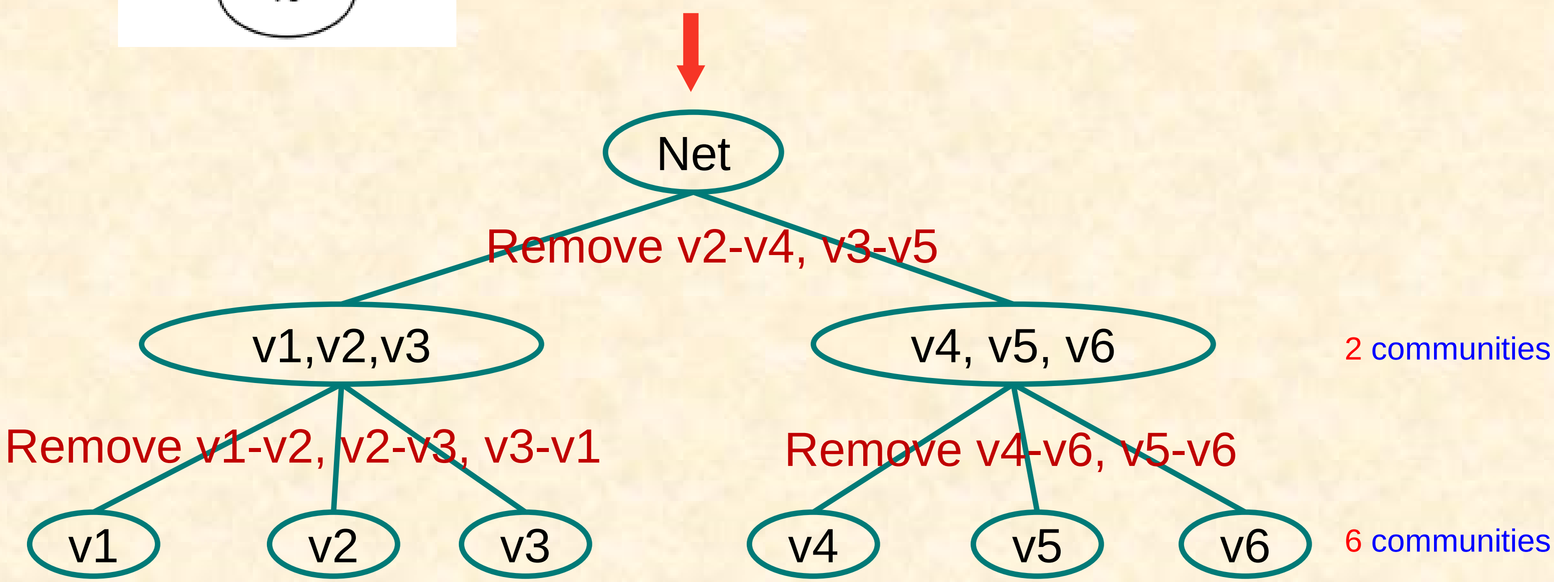
Remove (3) v1-v2, v2-v3, v3-v1



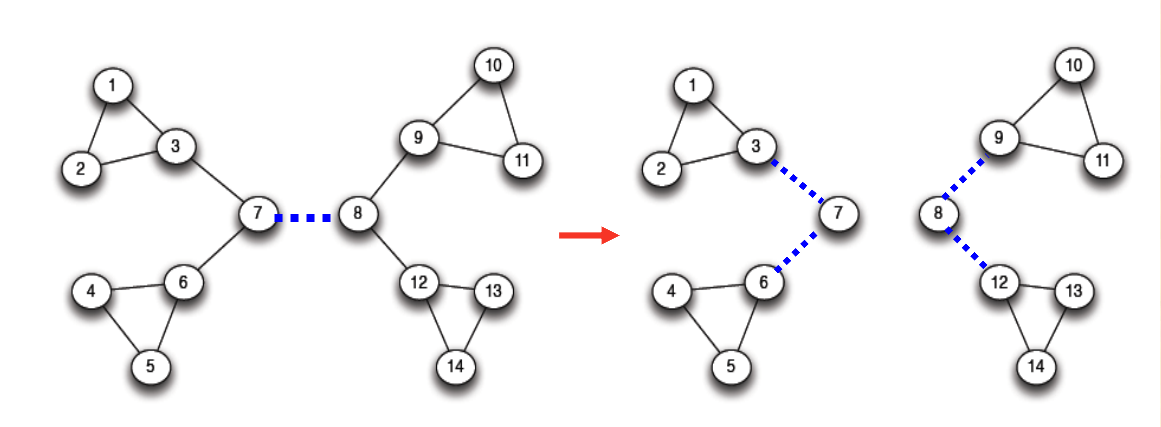
Remove (5) v2-v4, v3-v5

Remove (4) v4-v6, v5-v6

Remove (3) v1-v2, v2-v3, v3-v1

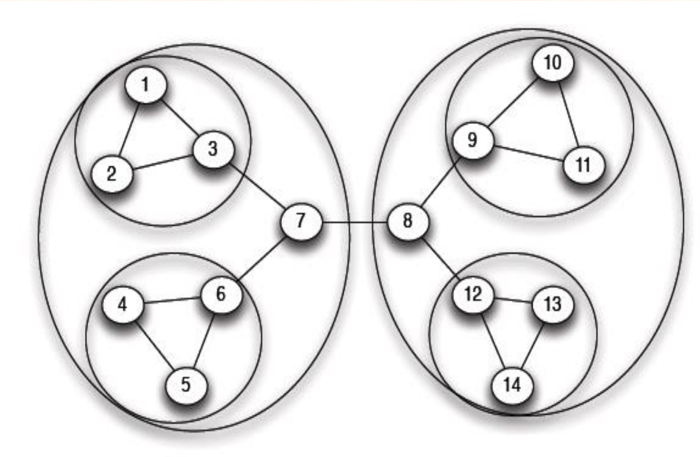


Example:



(a) *Step 1* (b) *Step 2*

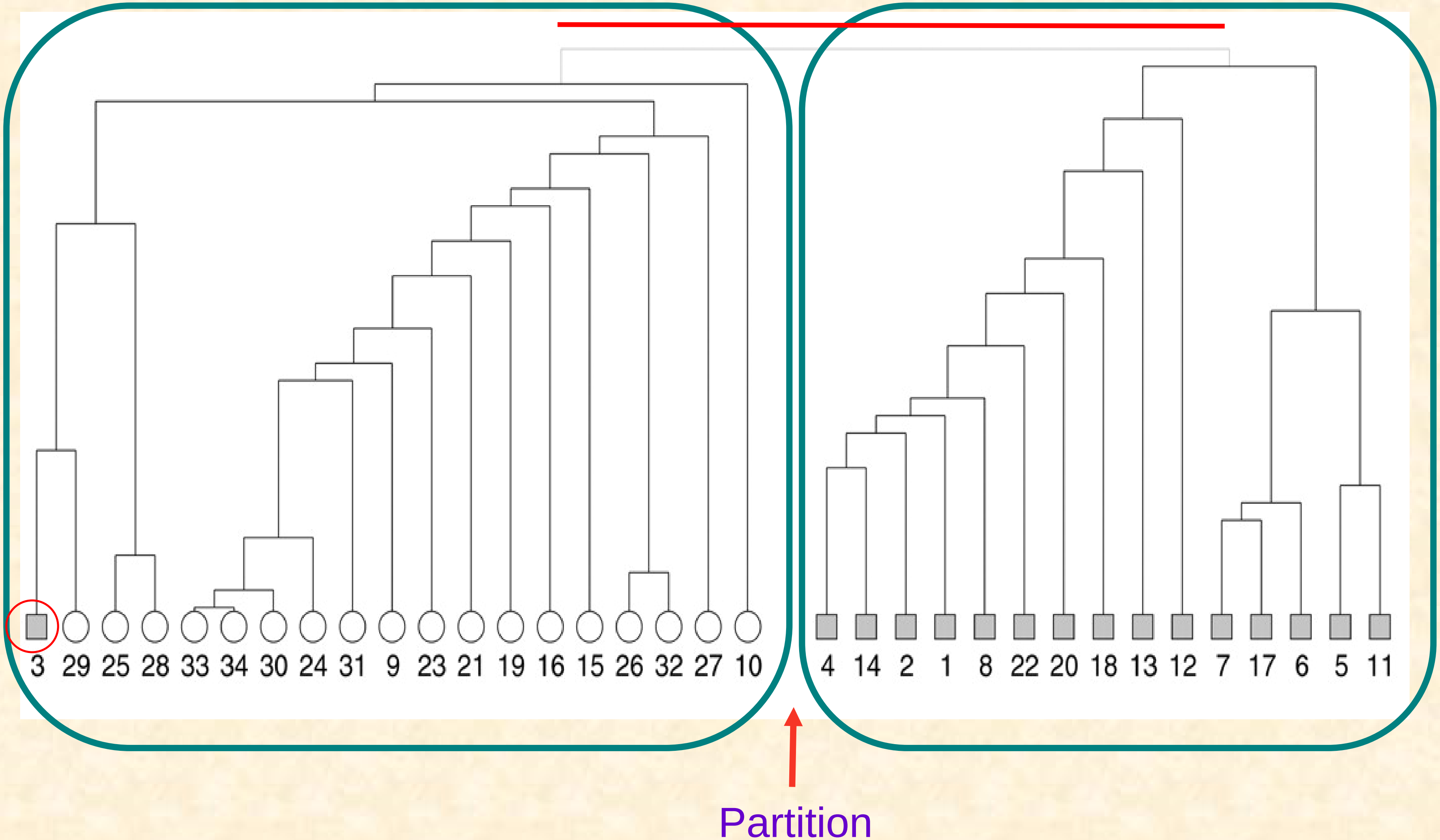
Step 1: remove 7-8 Step 2: remove 3-7, 6-7; 8-9, 8-12



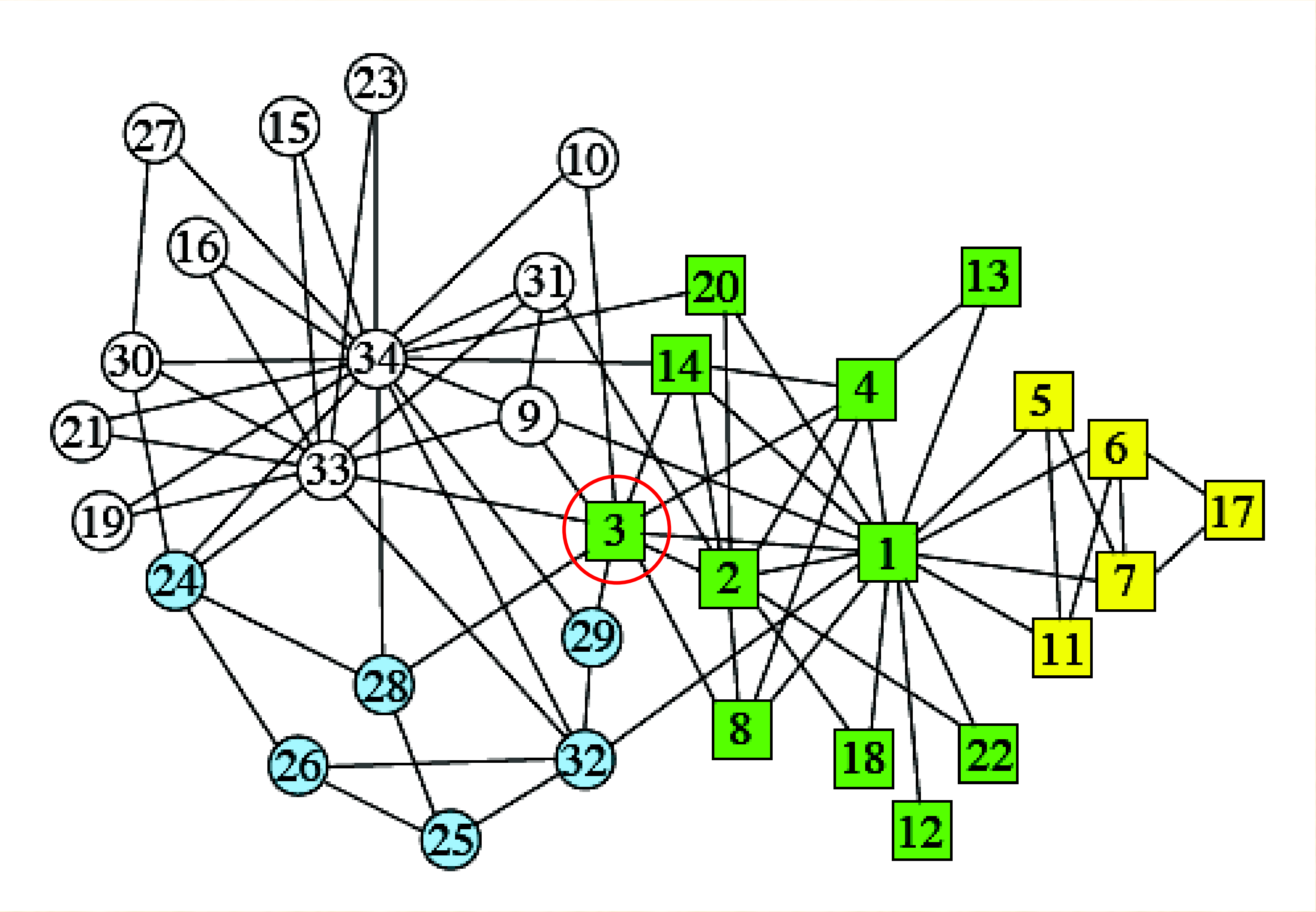
Result

GN Algorithm

Applied to the Karate Club Dataset

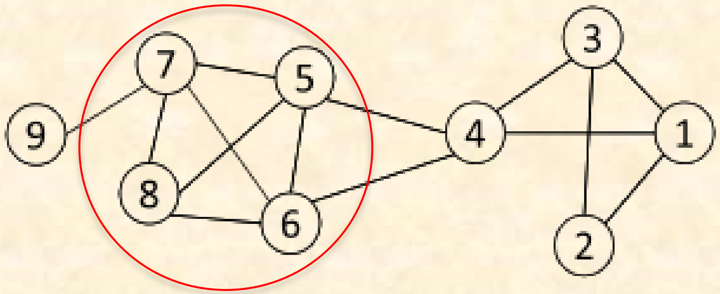


Zachary's Karate Network



4) Clique-Based Methods

- A clique is a fully-connected subgraph
- There are several clique-based community detection algorithms (computationally NP-hard)
- Since a node can be a member of more than one clique, these methods usually yield overlapping community structures



For example:
Nodes (5, 6, 7, 8) form a clique

Nodes (1, 2, 3), (1, 3, 4), (4, 5, 6),
also form a clique, respectively

Clique-Based Methods

- One approach is to find all the maximal cliques, which are cliques that are not a subgraph of any other clique
- To find maximum cliques: Bron-Kerbosch Algorithm

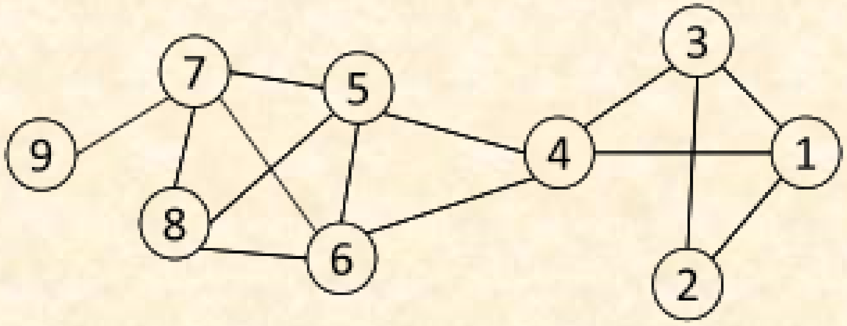
Recursively apply the following pruning procedure:

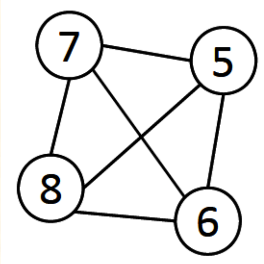
Sample a (large) subgraph from the given network, and find a clique of size k in it (say, by a greedy algorithm)

To find a larger clique, all nodes with degree $\leq k-1$ are removed from the whole network

Repeat the above, until the network is small enough

Example





- First, suppose we sampled a sub-network with nodes {1, 2, 3, 4, 5} and found a clique {1, 2, 3} of size 3

  [This is used as a reference to search for a possible larger clique]

- Next, to find a clique of size > 3, remove all nodes with degrees 1 and 2
  - Remove nodes 2, 9
  - Remove nodes 1, 3, 4

- Result is a larger clique: {5, 6, 7, 8}

- If the network is huge, continue: size > 4, size > 5, …

Clique Percolation Method

- A node can be a member of more than one clique. These methods usually yield overlapping community structures

- **CPM** is a method to find overlapping communities

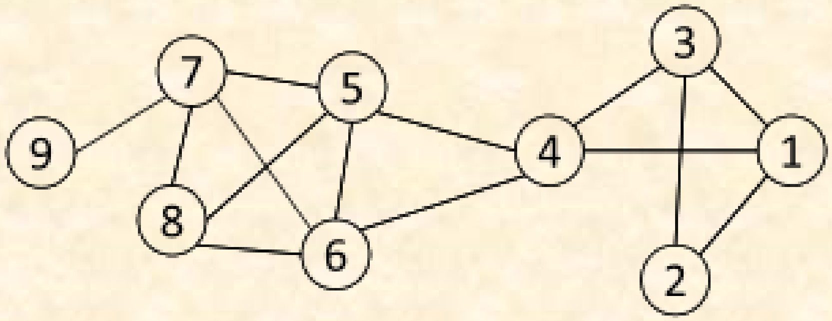
  Input: A network, and a parameter k

  Procedure:

  - Find all cliques of size k in the given network
  - Construct a clique graph:

    Two cliques are adjacent if they share $k-1$ nodes
  - Each connected cluster in the clique graph is a community

Example

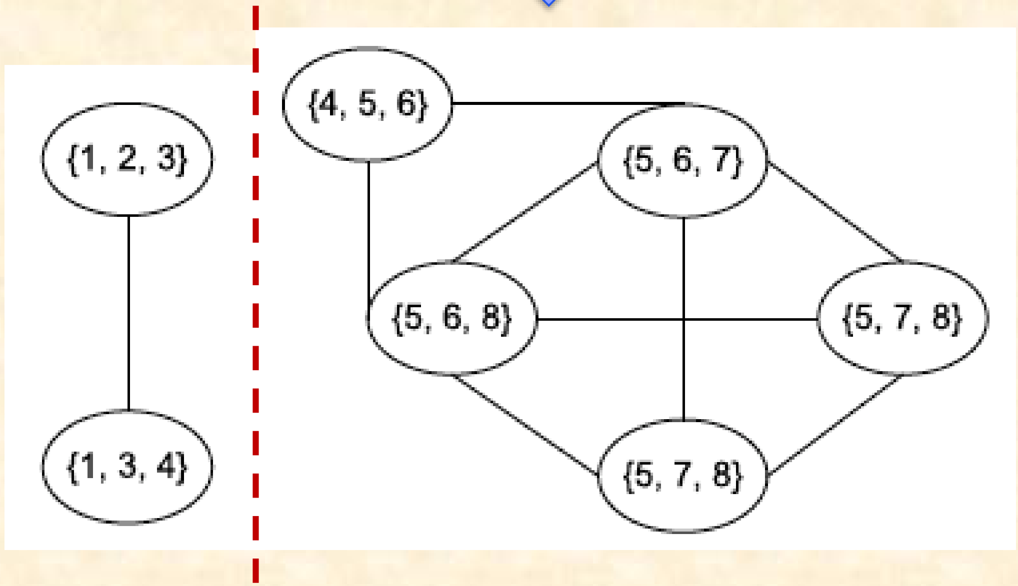


Cliques of size 3:

{1, 2, 3}, {1, 3, 4},
{4, 5, 6}, {5, 6, 7},
{5, 6, 8}, {5, 7, 8},
{6, 7, 8}

$k = 3$

$k - 1 = 2$ shared nodes



Communities:

{1, 2, 3, 4}
{4, 5, 6, 7, 8}
{9}

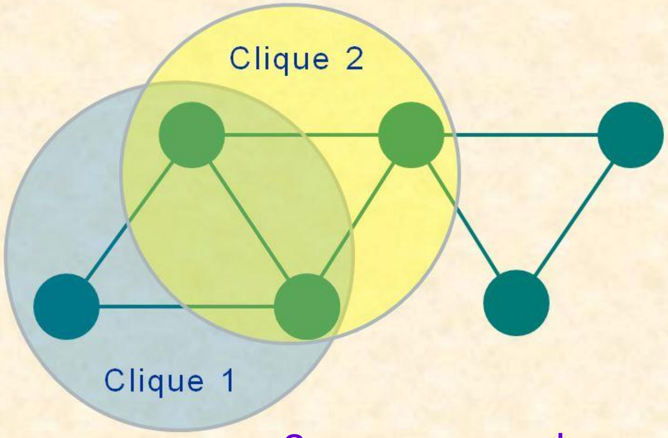
overlapping: node 4

k-Clique Communities

Union of all k-cliques that can be reached from each other through a series of adjacent k-cliques

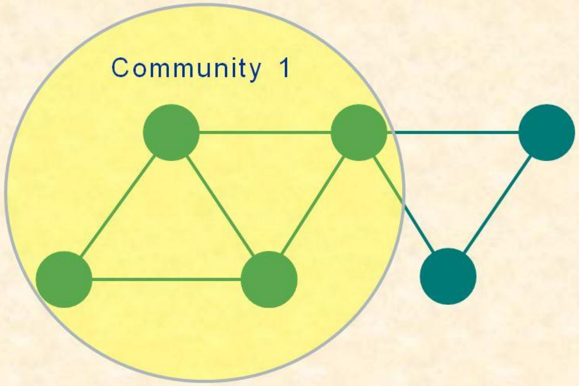
Example:

k = 3

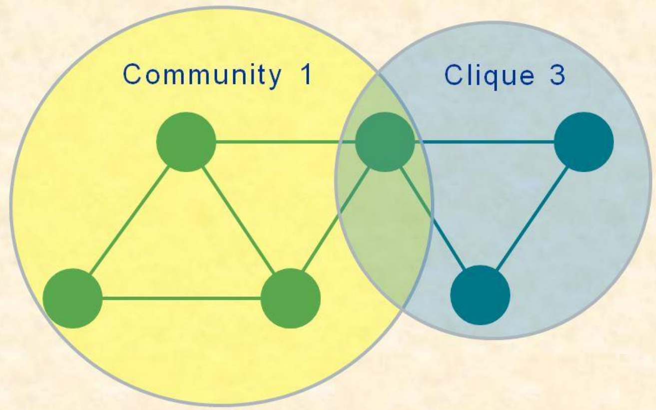


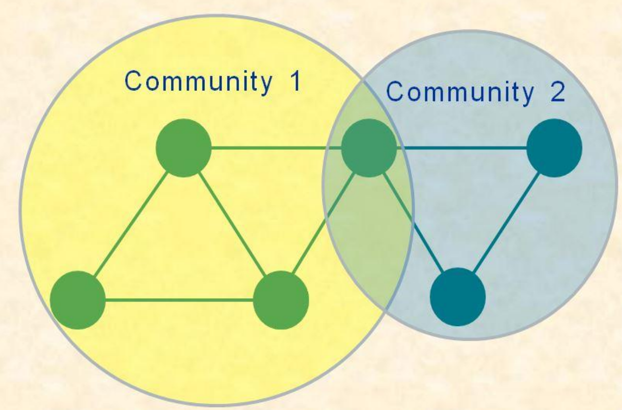
2 common nodes

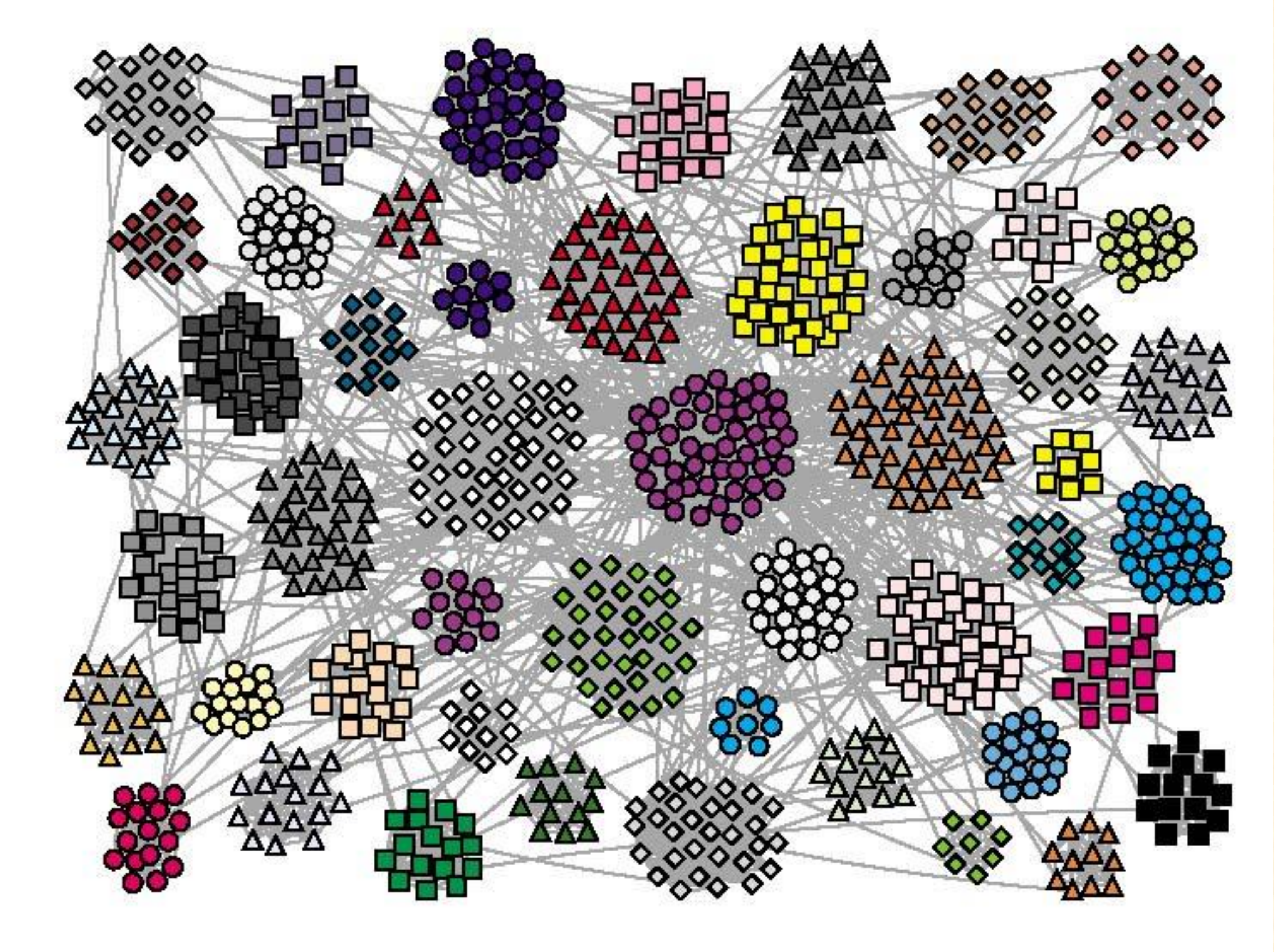
→ They belong to the same community



1 common node

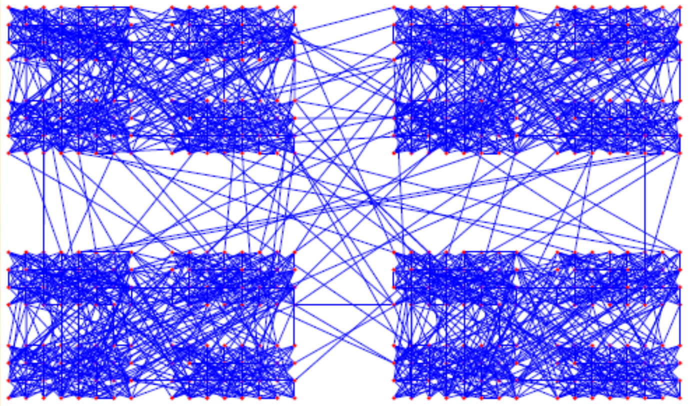


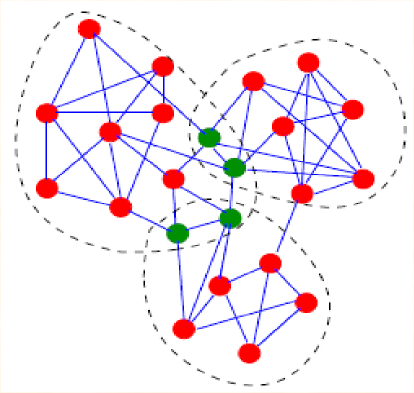




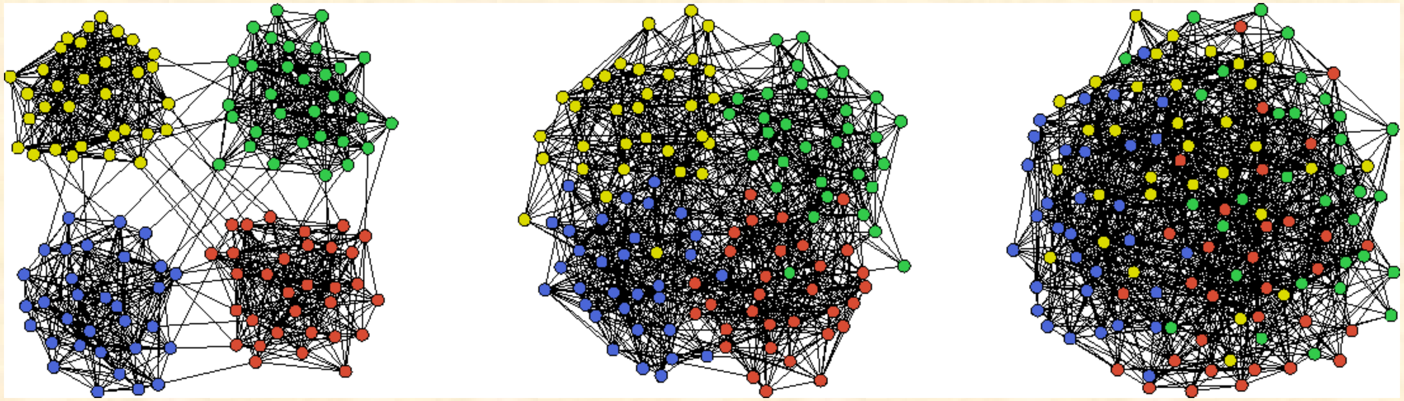
A. Lancichinetti, S. Fortunato, and F. Radicchi, Phys. Rev. E 78, 046110 (2008)

Detecting Community Structure: Challenges

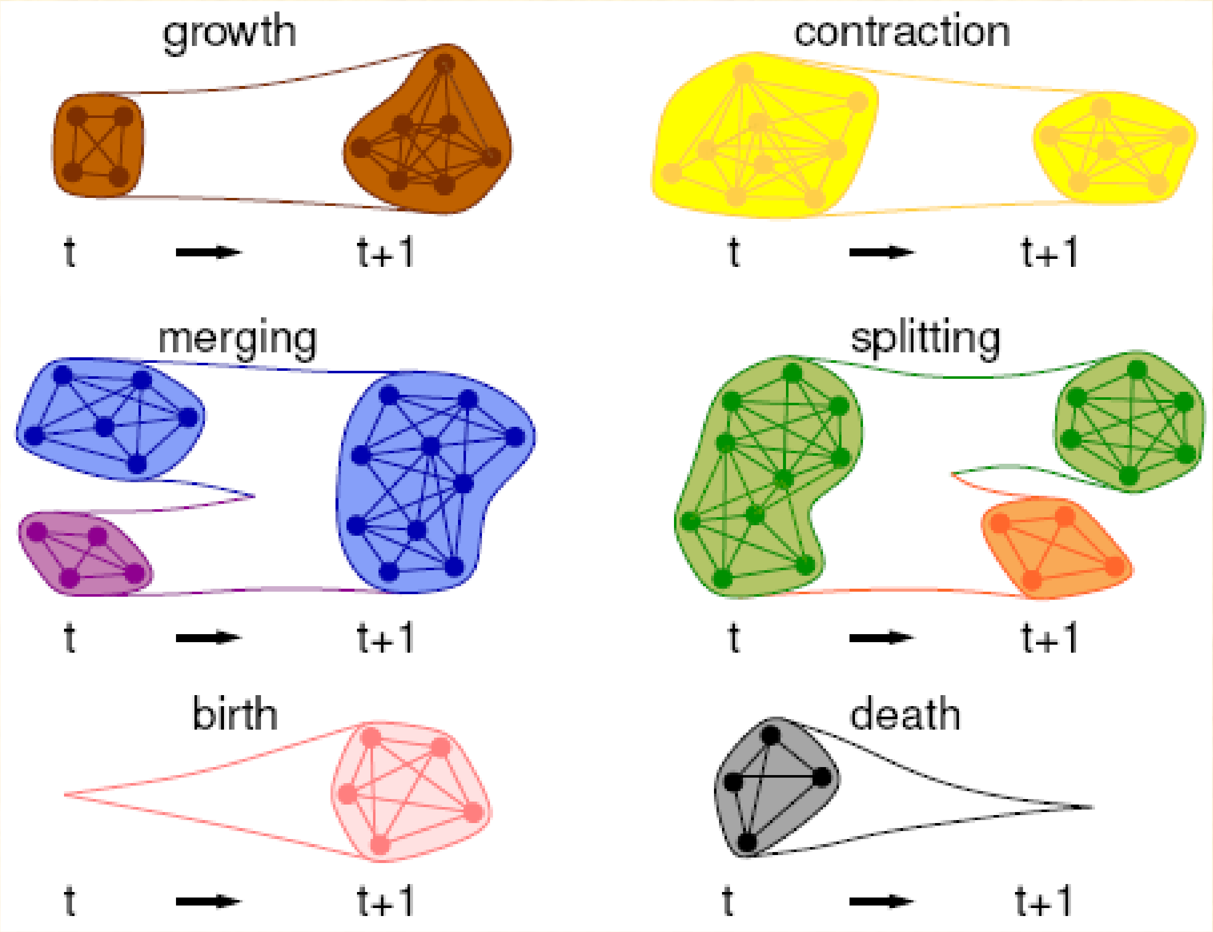




- Hierarchical
- Overlapping



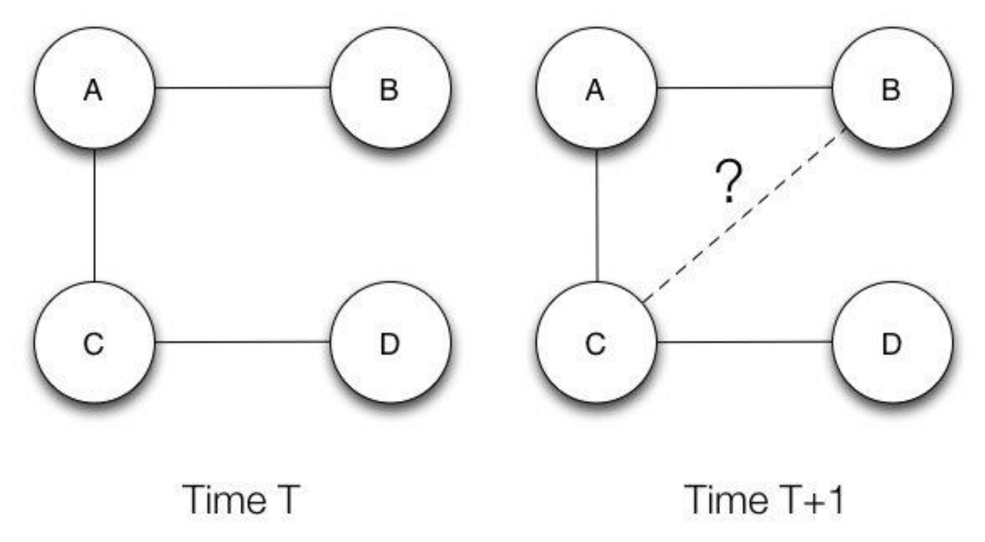
Detecting Community Structure: Challenges



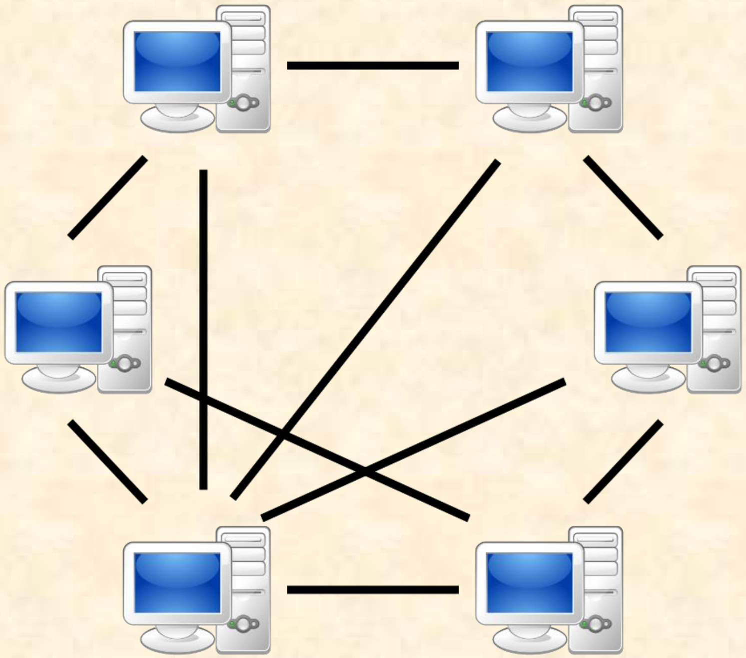
- Evolution, Emergence

Link Prediction

An emerging community ?



Link Prediction



Predict:

Which computer is likely to connect to which computer

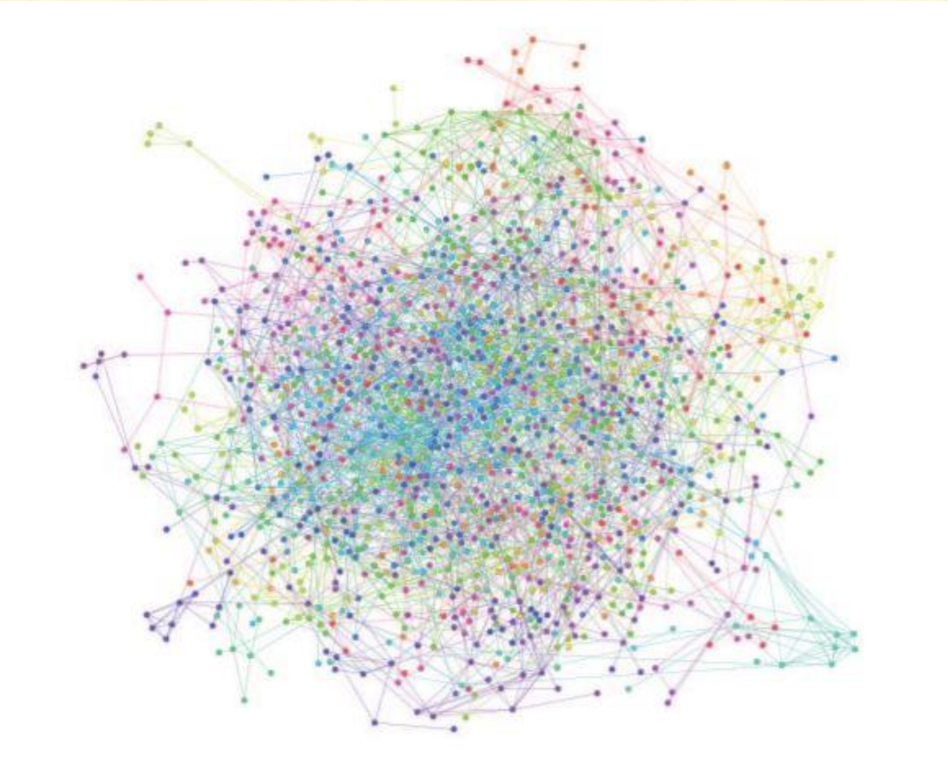
Link Prediction

Predict:

Which person is likely to connect to which friend

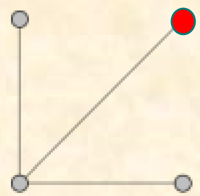
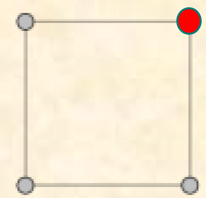
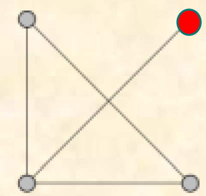
Link Prediction

For a large-scale network, this could be challenging



Link Prediction

Simple Cases:

Degree sequences: (3,1,1,1) (2,2,2,2) (3,2,2,1)

Criteria:

Similarity: Similar degrees, properties, importance, …

Commonality: Common friends, features, …

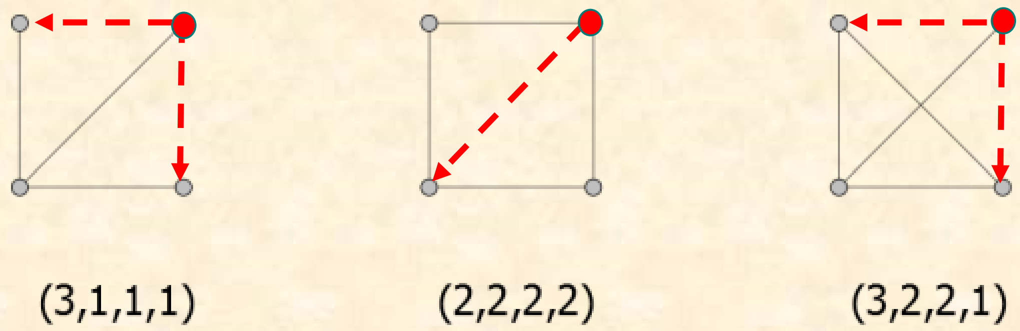
Closeness: Closeness, distances, …

Link Prediction

Based on:

Degree similarity

Closeness/Distance



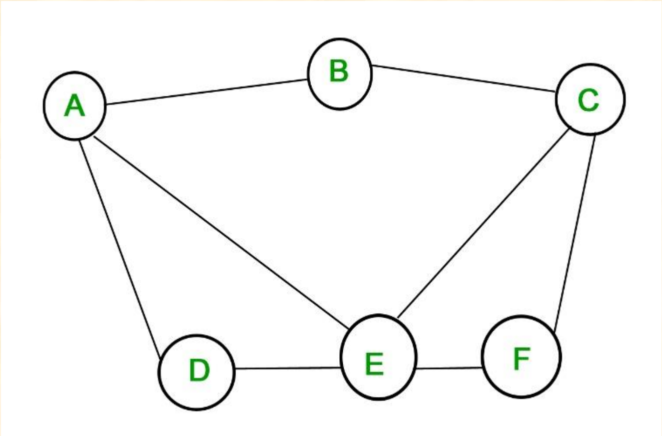
Criteria:

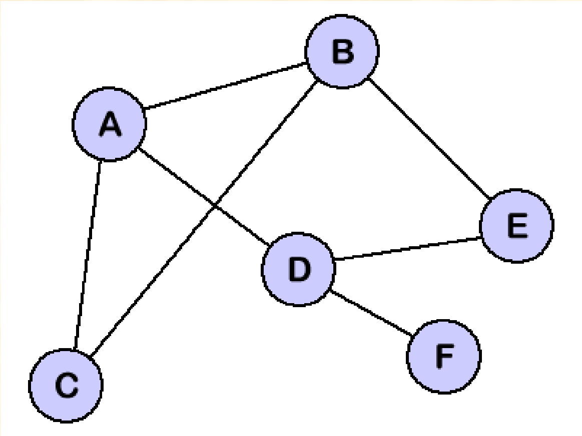
Similarity: Similar degrees, properties, importance, …

Commonality: Common friends, features, …

Closeness: Closeness, distances, …

Examples





Predict link(s) from node "B"

Predict link(s) from node "F"

Answer: B – D and B – F

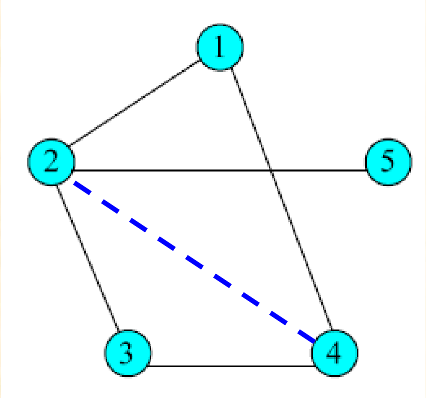
(They have degree 2)

Answer: F – E and A – F

(There is no other degree-1 node)

(They have a shortest distance)

Link Prediction



Criterion:

Based on node-degree average

- Give every node-pair (i, j) a value (weight):

  $(i, j) = \sqrt{k_i k_j}$ (node i has degree k_i)

- Predict a link between two un-connected nodes with a highest node-pair value:

- $(1,2) = \sqrt{6}$, $(1,3) = \sqrt{4}$, $(1,4) = \sqrt{4}$, $(1,5) = \sqrt{2}$

  $(2,3) = \sqrt{6}$, $(2,4) = \sqrt{6}$, $(2,5) = \sqrt{3}$, $(3,4) = \sqrt{4}$, $(3,5) = \sqrt{2}$, $(4,5) = \sqrt{2}$

→ Predict a new link: 2 - 4

Link Prediction

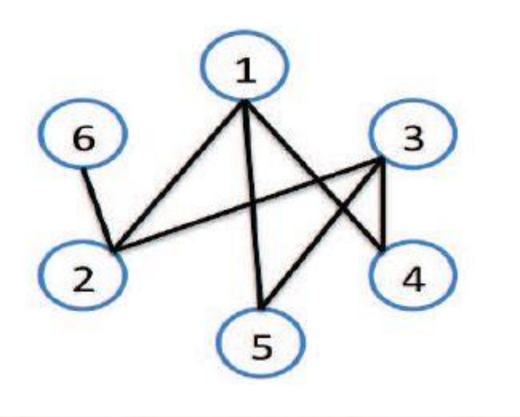
Based on Commonality

Neighborhood of node x : $N(x) = \{i\text{: } i \text{ connects to } x\}$

Neighborhood of node y : $N(y) = \{i\text{: } i \text{ connects to } y\}$

Intersect: $N(x) \cap N(y) = \{i\text{: } i \text{ belongs to both neighborhoods}\}$

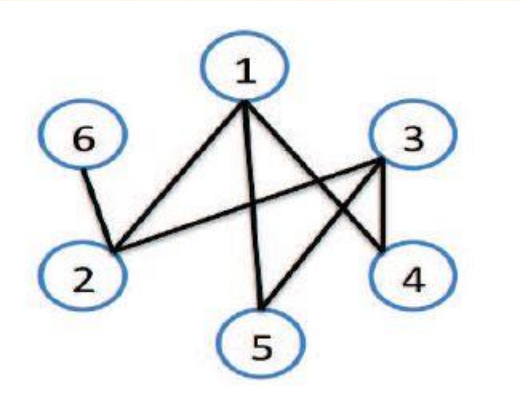
Cardinality: $|N(x) \cap N(y)|$ = number of nodes in the intersect



Example:

$N(1) = \{2, 4, 5\}$
$N(6) = \{2\}$
→ $N(1) \cap N(6) = \{2\}$
→ $|N(1) \cap N(6)| = 1$

Link Prediction



Based on Commonality

→

Predict 1st new link: 1 – 3

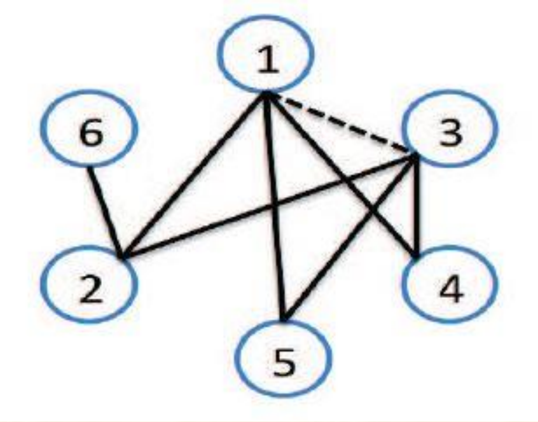
Predict 2nd new links:

2 – 4 or 2 – 5 or 4 – 5

and so on

- $|N(1) \cap N(3)| = 3$
- $|N(2) \cap N(4)| = 2$
- $|N(2) \cap N(5)| = 2$
- $|N(4) \cap N(5)| = 2$
- $|N(1) \cap N(6)| = 1$
- $|N(3) \cap N(6)| = 1$
- $|N(4) \cap N(6)| = 0$
- $|N(5) \cap N(6)| = 0$
- … … = 0

→



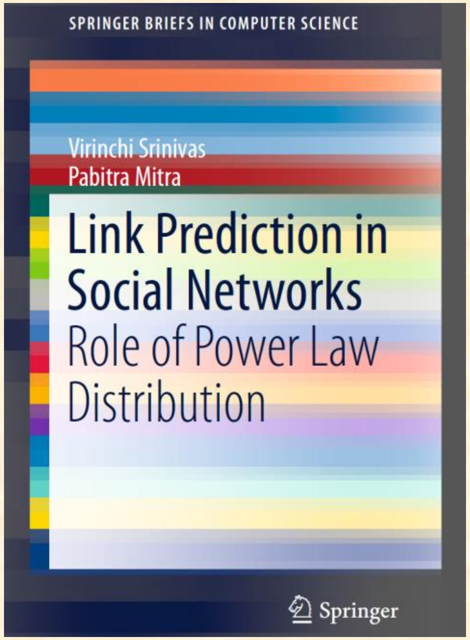
Other Link Prediction Methods

- Node-similarity-based methods
- Topology-similarity-based methods

  CN, AA, RA, LP, Katz, LRW, SRW, RWR, …
- Maximum-likelihood analytic methods

  Layer-structure modeling,

  random partitioning, …
- Machine Learning
- … …



BREAK

10 minutes