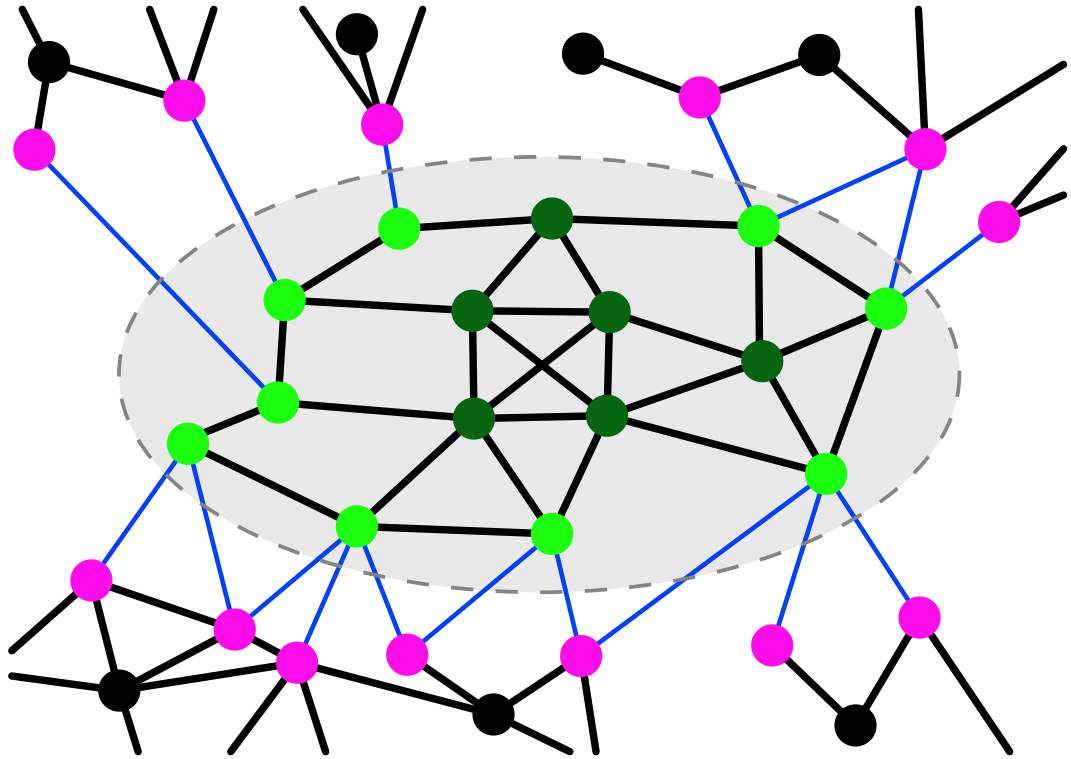
Complex networks

Communities

János Török

Department of Theoretical Physics

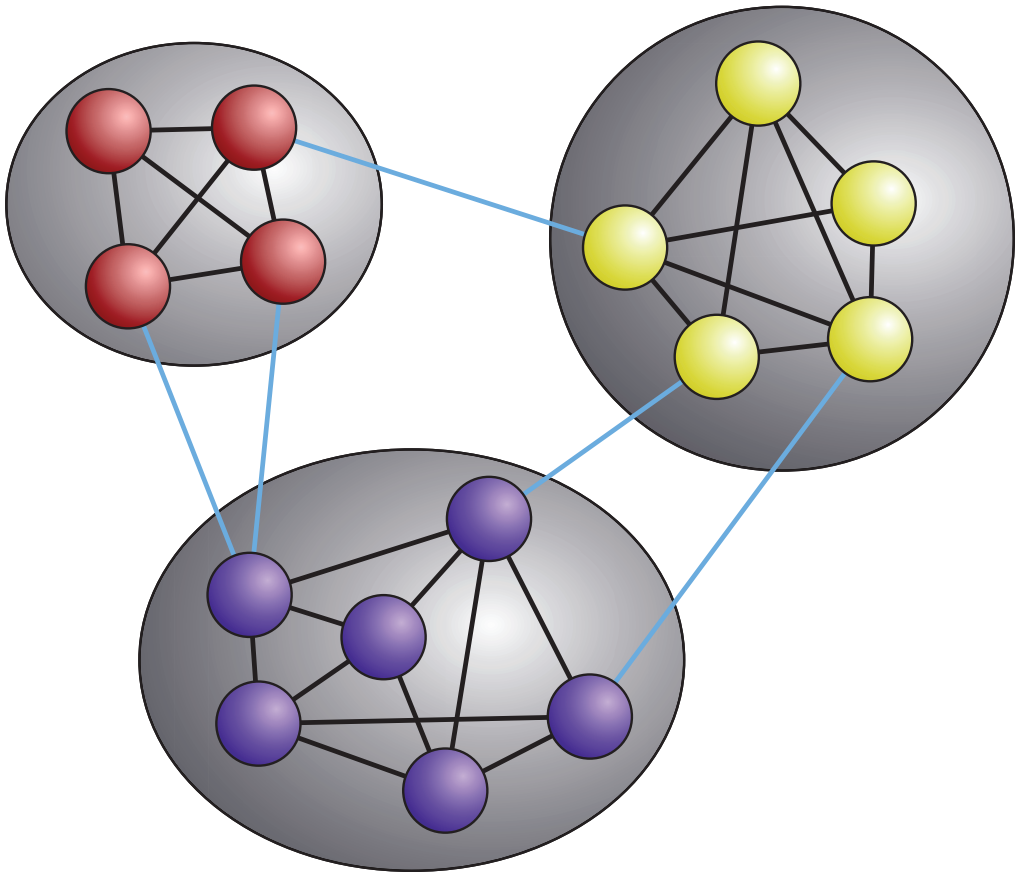
May 4, 2022



- Definition of a community?
- More connected to itself than to the rest

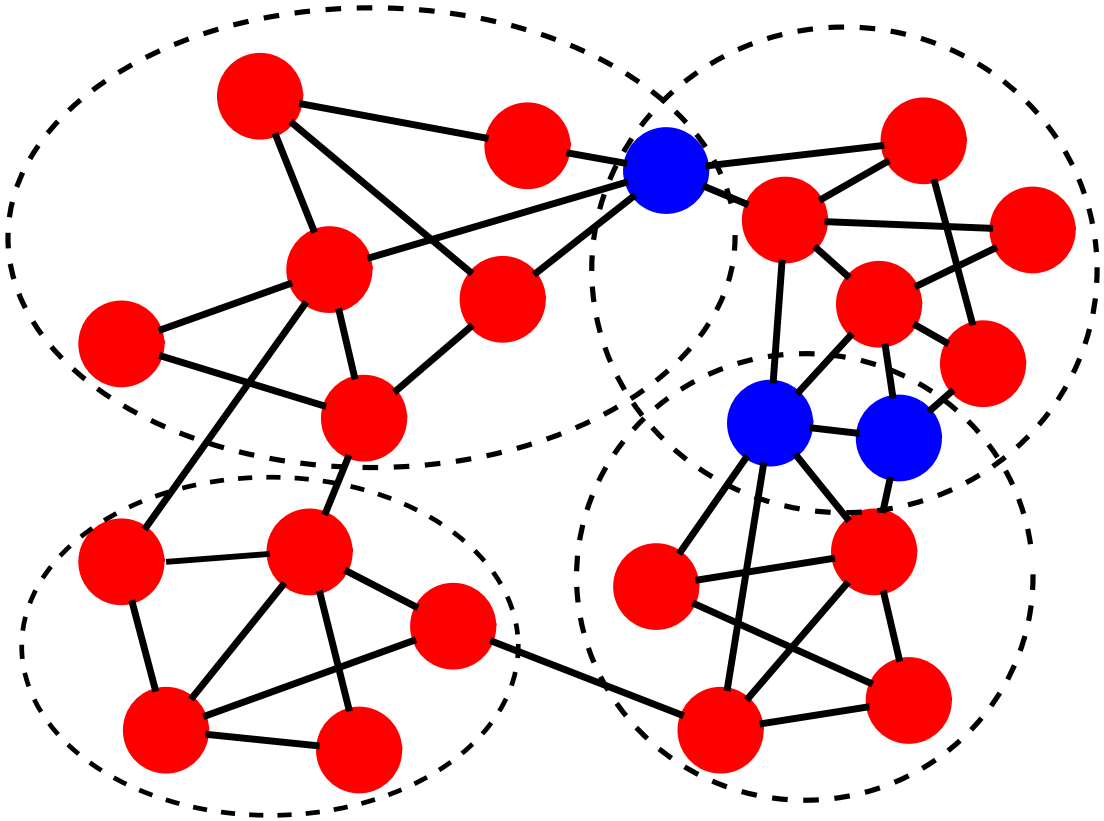
S. Fortunato, Phys. Rep. (2010)

Communities: Classic view



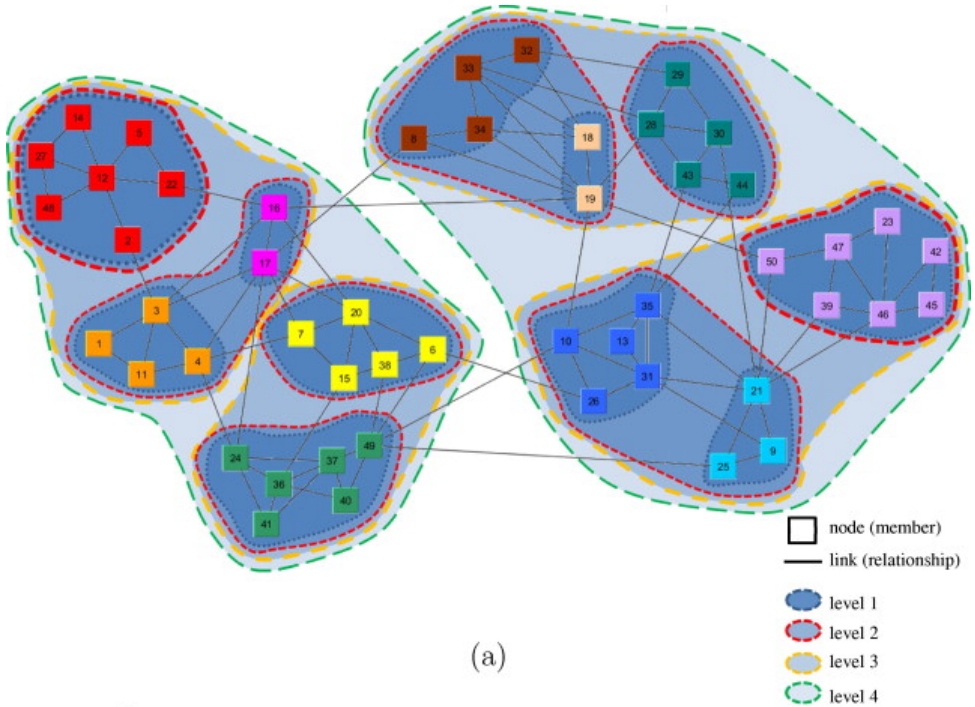
- Disjunct communities
- Each node is assigned to a single community

Communities: Classic view

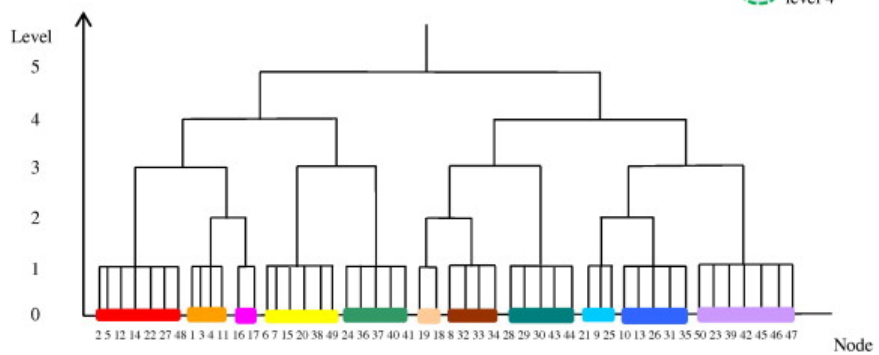


- Overlapping communities
- A node can be part of different communiteis

Communities: Classic view



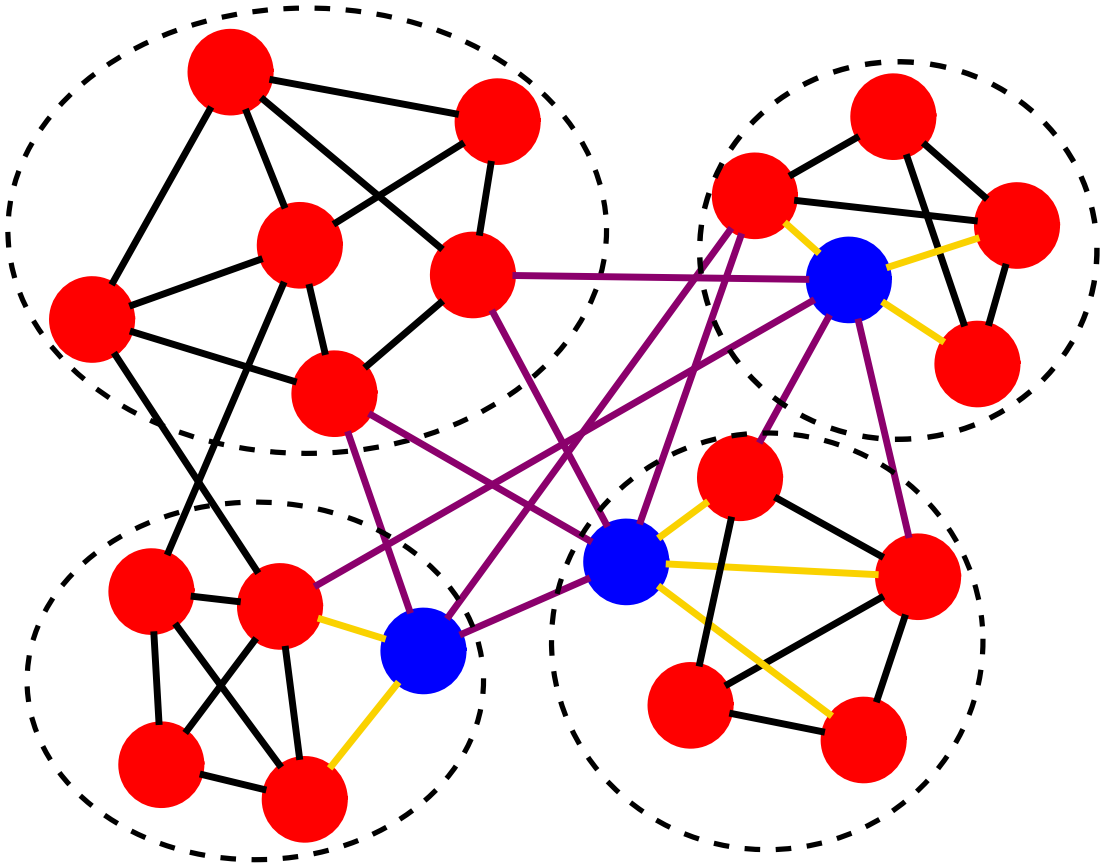
(a)



(b)

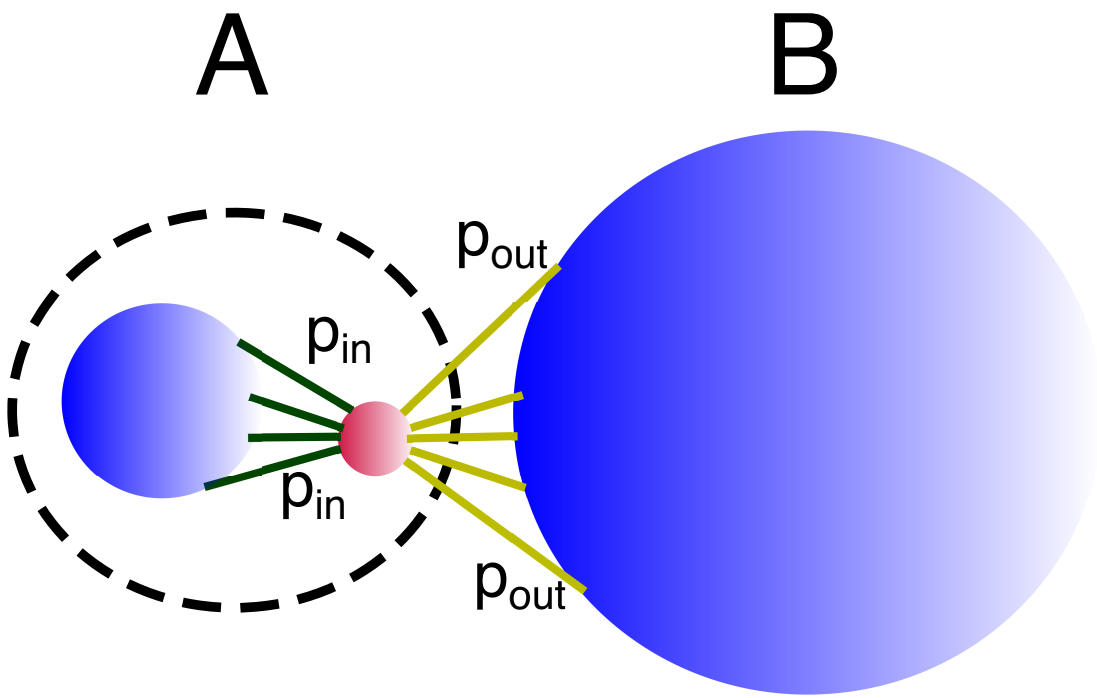
- Hierarchical communities
- Different level of communities

Strong communities



- Internal degree $>$ extrenal degree for all member nodes
- Internal degree $>$ extrenal degree to all other communities

Weak communities



- Internal degree of subgraph $>$ extrenal degree

Modern view

- **Strong community**: is a subgraph each of whose vertices has a higher probability to be linked to every vertex of the subgraph than to any other vertex of the graph.
- **Weak community**: is a subgraph such that the average edge probability of each vertex with the other members of the group exceeds the average edge probability of the vertex with the vertices of any other group

Alternative definitions

- Similarity, distance, *e.g. hierarchical clustering*
- Random walk, or diffusion (time spent in groups), *e.g. infomap*
- Presence of motifs, *e.g. clique percolation*
- Algorithms...

Benchmarks

- Real network
  - Known partition: (e.g. Zachary karate club)
  - Metadata of a real network (e.g. orkut)
- Artificial benchmarks
  - Stochastic block model (Girvan and Newman, with single P_{in} and P_{out}, and $\mu = k^{\text{ext}}/(k^{\text{ext}} + k^{\text{int}})$ as mixing parameter)
  - Lancichinetti-Fortunato-Radicchi (LFR) Benchmark:
    - Configuration model, with power law degree distribution
    - Predefined community structure (originally power law distribution)
    - Node i has $(1-\mu)k_i$ internal and μk_i external links, with μ being the mixing parameter.

Similarity measures

Let a_{11}, (a_{00}) be the number of nodes pairs (not) in the same community in both partitions, a_{10}, a_{01} be the number of nodes pairs present only in one of the partitions. Other words: a_{11} true positive, a_{00} true negative, a_{01}, a_{10} false prediction (pairs here not pairs there)

- Rand index:

$$\frac{a_{11} + a_{00}}{a_{11} + a_{00} + a_{01} + a_{10}}$$

- Jaccard index

$$\frac{a_{11}}{a_{11} + a_{01} + a_{10}}$$

- Normalized mutual information (NMI)
- Omega index

Similarity measures

- Rand index:
- Jaccard index
- Normalized mutual information (NMI):

$$I_{\text{norm}} = \frac{2I(X,Y)}{H(X)+H(Y)}$$

where $H(X) = -\sum_x P(x)\log P(x)$ Shannon entropy. $H(X) = 0$ for perfect clustering, and maximal for random graph, and $P(x) = n_x^X/n$ is the correctly assigned nodes in cluster X, and
$I(X,Y) = \sum_{X,Y} P(X,Y)\log\left(P(X,Y)/[P(X)P(Y)]\right)$

- Omega index

Similarity measures

- Rand index:
- Jaccard index
- Normalized mutual information (NMI):
- Omega index

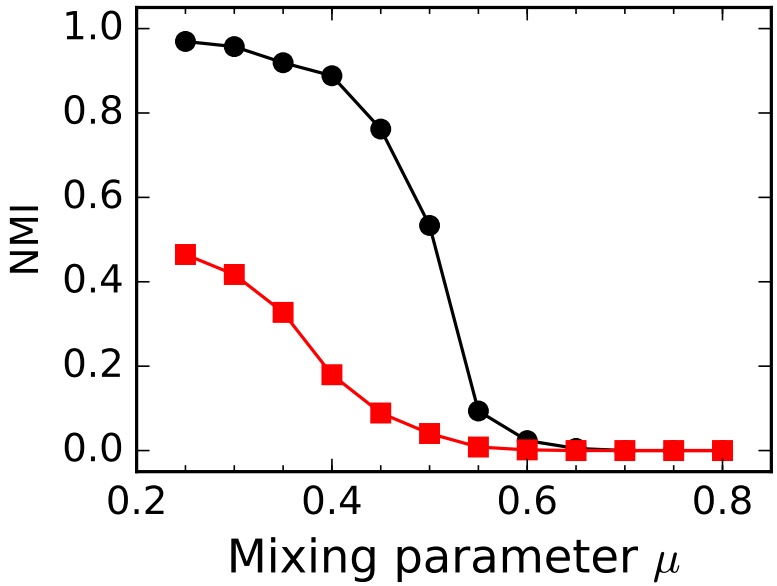
$$o(X, Y) = \frac{2}{n(n-1)} \sum_j a_{jj}$$

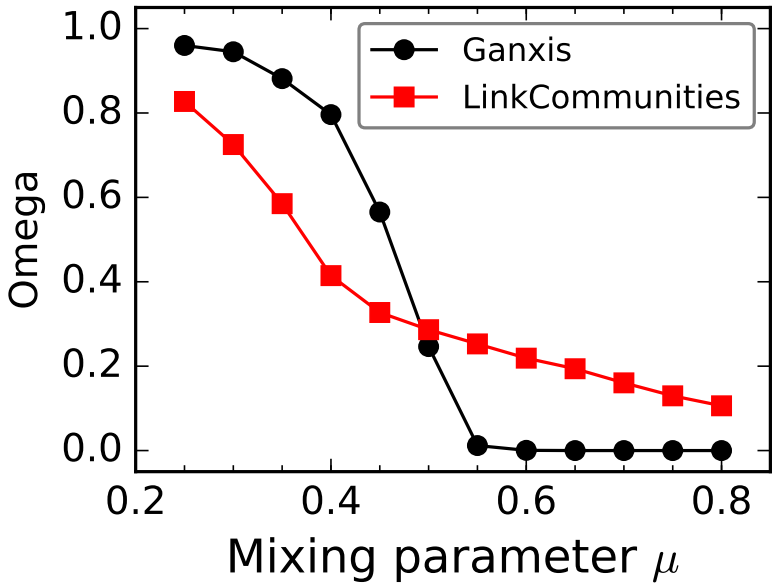
and

$$\Omega(X, Y) = \frac{o(X, Y) - o_e(X, Y)}{1 - o_e(X, Y)}$$

where $o_e(X, Y)$ is the expected value of $o(X, Y)$ according to a null model.

Similarity measures: no uniqness





Computation time

Name	Nature	Comp.
Ravasz	Hierarchical Agglomerative	$O(N^2)$
Girvan-Newman	Hierarchical Divisive	$O(N^2)$
Greedy Modularity	Modularity Optimization	$O(N^2)$
Greedy Modularity (Optimized)	Modularity Optimization	$O(N\log^2 N)$
Louvain	Modularity Optimization	$O(L)$
Infomap	Flow Optimization	$O(N\log N)$
Clique Percolation (CFinder)	Overlapping Communities	$Exp(N)$
Link Clustering	Hierarchical Agglomerative; Overlapping Communities	$O(N^2)$

Methods

- Infomap
- Girvan-Newman
- Link community

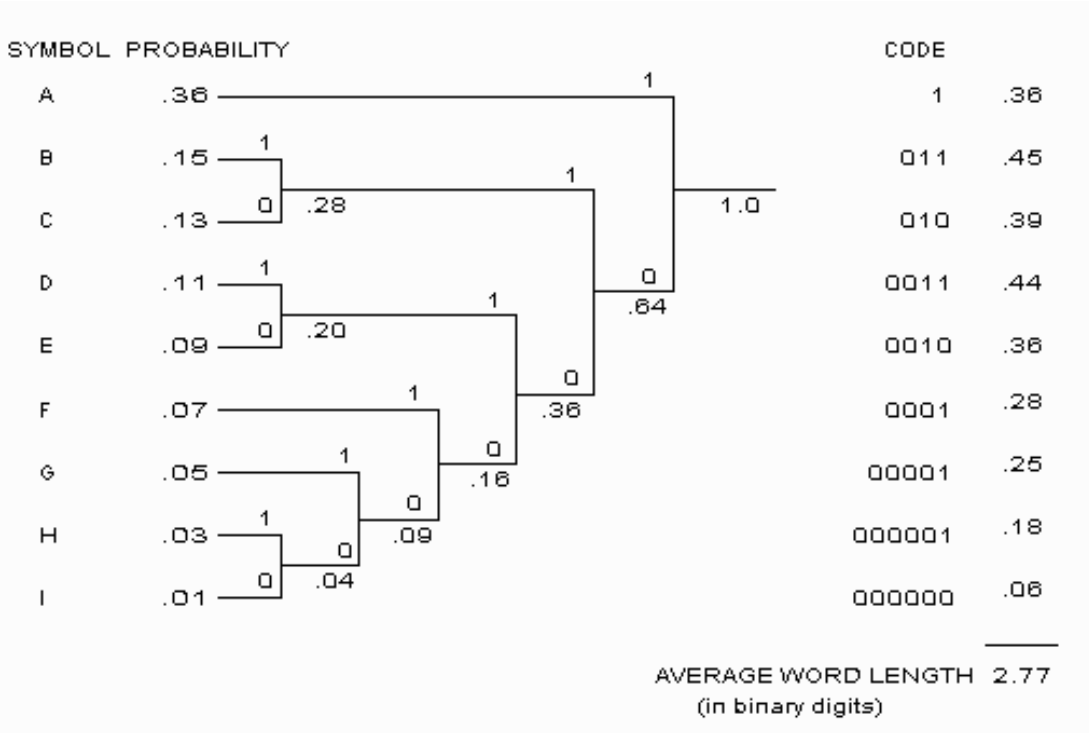
Random Walks on Graphs

- Nodes in a community have higher probability for internal than for external link.
- Random walker has a higher probability of remaining inside a community than passing to an other.
- Use this feature for community detection.
- Infomap

Infomap idea

- Take a (long) path of a random walker
- Encode it efficiently by giving unique address to each node
- Compress the encoding by assuming two level structure
- Give two level codes: Top ones (unique for each group), local (can be the same in different groups). Ex:
  - addresses in real life: Countries, Cities (there is also a Budapest in the USA), Streets (you may find Main street in many cities)
  - domain names: .hu, .de; lower domains, e.g. notebook, weather

Huffman coding



- Compress data in the most efficient general way

Huffman coding

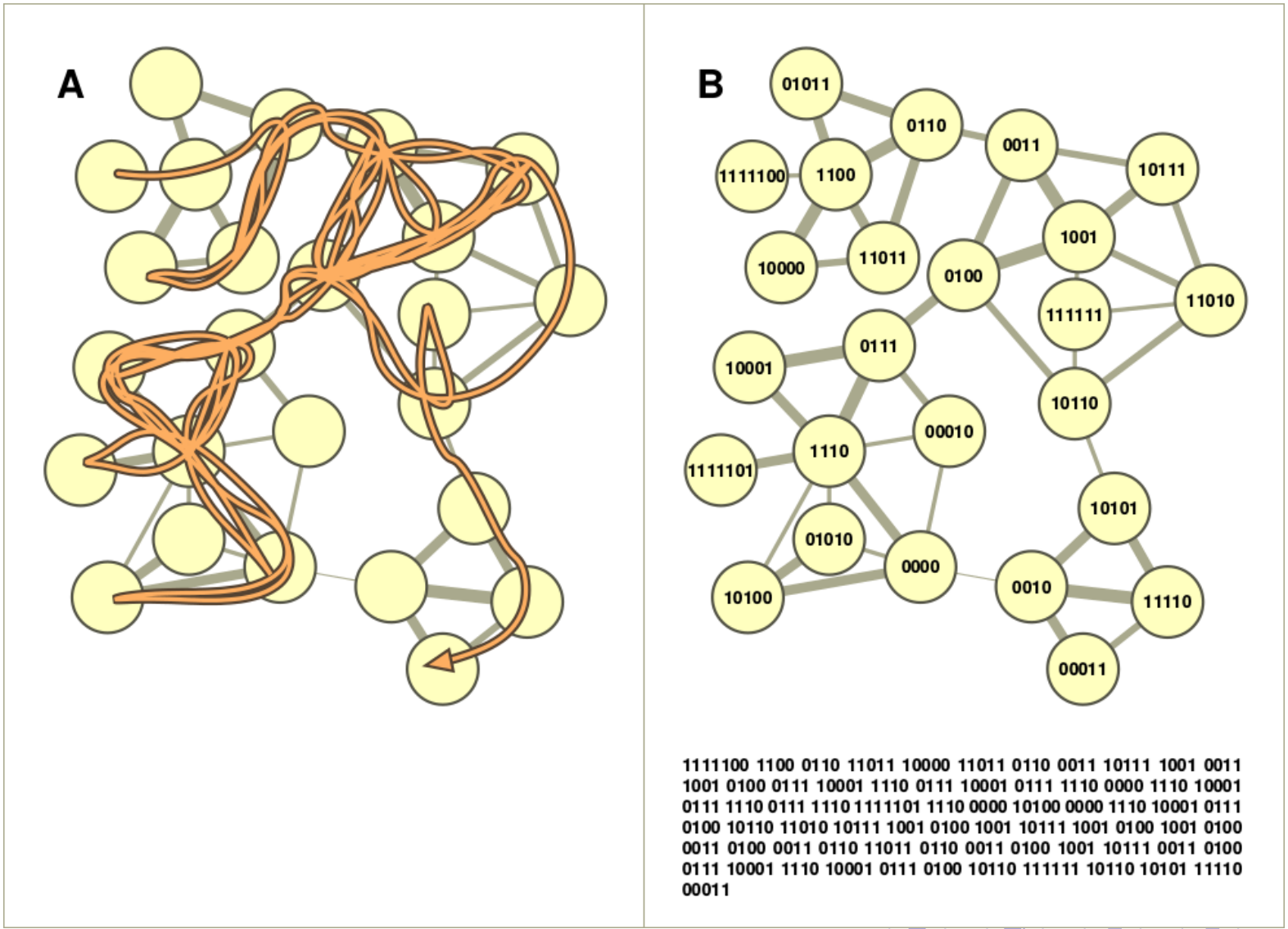
1. Create a leaf node for each symbol and add it to the priority queue.
2. While there is more than one node in the queue:
   - 2.1 Remove the two nodes of highest priority (lowest probability) from the queue
   - 2.2 Create a new internal node with these two nodes as children and with probability equal to the sum of the two nodes' probabilities.
   - 2.3 Add the new node to the queue.
3. The remaining node is the root node and the tree is complete.

Huffman coding, vs. infomap

- Can a coding be more efficient than Huffman coding?
- If we know more about the data yes!
- Answer: Two level coding (Of course it would be stupid for text)

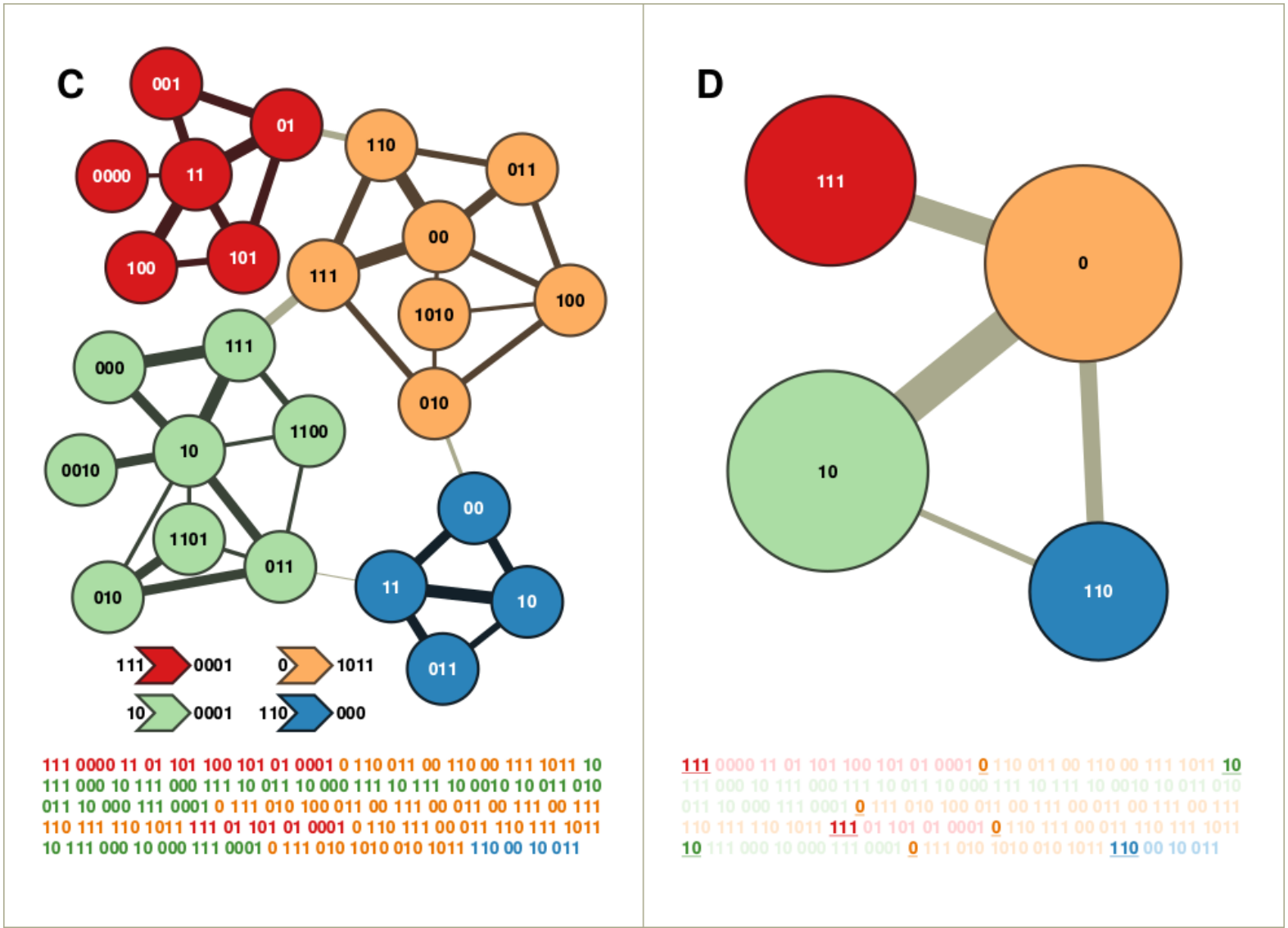
Sample random path and Huffman coding

Path length: 314 bits



Sample random path and Huffman coding

Path length: 243 bits



Infomap: Algorithm

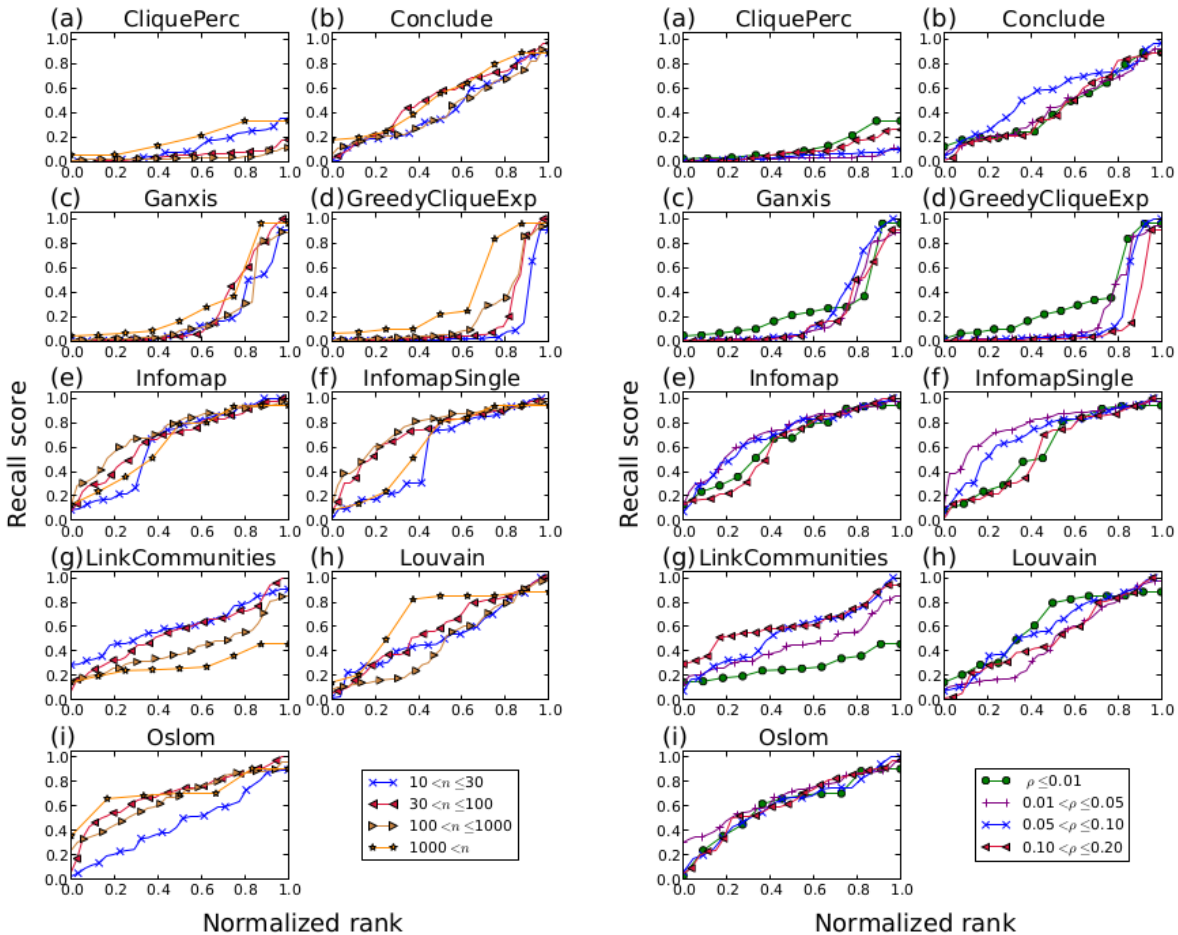
- Start with Huffman coding
- Optimize coding to minimize the *map equation*:

$$L = q_{\curvearrowright} H(Q) + \sum_{i=1}^{n_c} p^i_{\circlearrowright} H(P^i),$$

  where $H(Q)$ is the frequency-weighted average length of codewords for inter group jumps, $H(P^i)$ is frequency-weighted average length of codewords for group i.
- Implementation: Start with all nodes as different communities
- Merge them if L decreases

Infomap

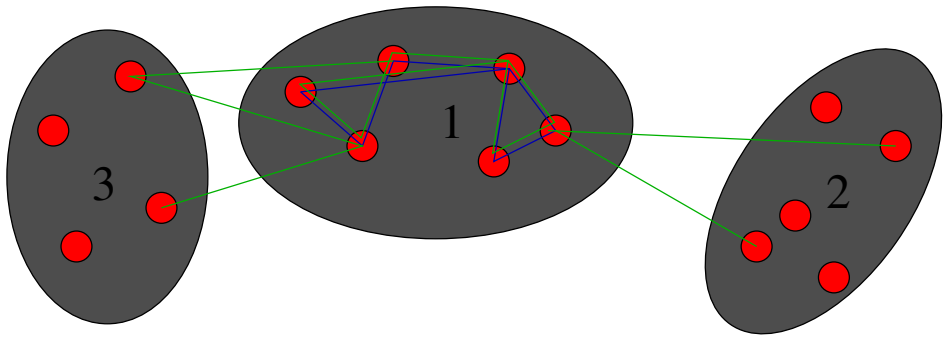
- One of the most popular
- Fast for large networks
- Relability is comparable to more complex methods



Modularity

Global method

- $e_{\alpha\beta}$ percentage of edges between modules (clusters) α and β
  probability edge is in module α is $e_{\alpha\alpha}$
- a_α percentage of edges with at least 1 end in module α
  probability a random edge would fall into module α



- Modularity is

$$Q = \sum_{\alpha=1}^{k} (e_{\alpha\alpha} - a_\alpha^2)$$

- $a_\alpha = \sum_\beta e_{\alpha\beta}$
- Try to maximize Q

Modularity algorithm

- Rewrite Q:

$$Q = \frac{1}{2m} \sum_{i,j \in \text{same}} \left[A_{ij} - \frac{k_i k_j}{2m} \right]$$

  $2m = \sum_i k_i$
- Only two modules
- $s_i = \pm 1$: 1 if node i is in module 1; -1 otherwise

$$Q = \frac{1}{4m} \sum_{\{i,j\}} \left[A_{ij} - \frac{k_i k_j}{2m} \right] (s_i s_j + 1)$$

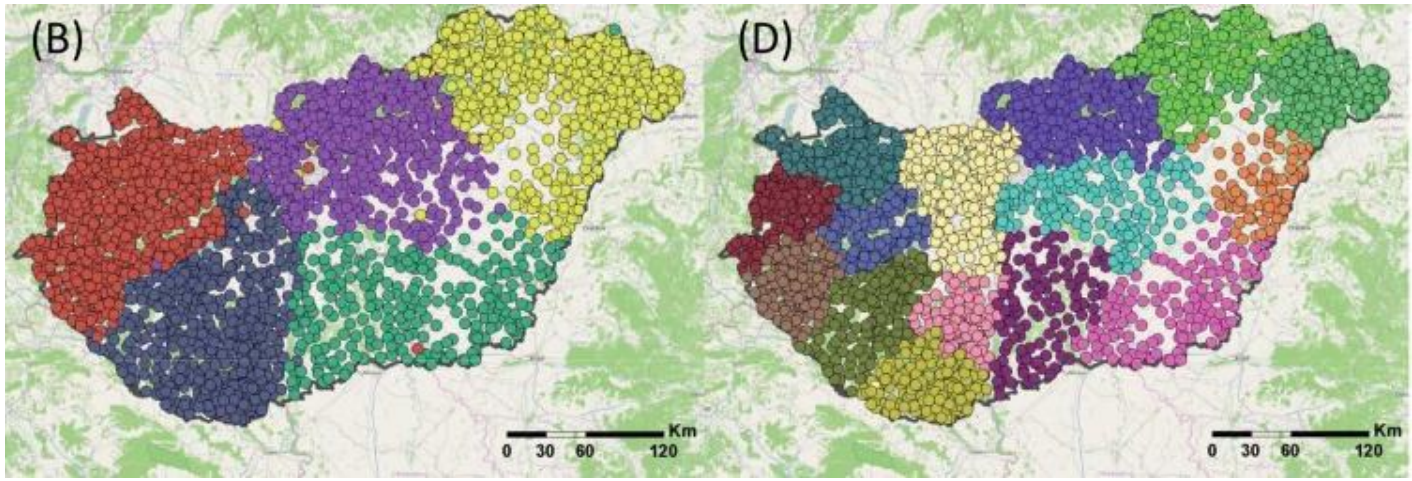
- $+1$ is a constant can be omitted
- Change the vector s_i to maximize Q

Modularity algorithm

$$Q = \frac{1}{4m} \sum_{\{i,j\}} \left[A_{ij} - \frac{k_i k_j}{2m} \right] s_i s_j$$

- Try to find ± 1 vector s_i that maximizes the modularity.
- Start with two groups
- Then split one of the two groups using the same technique
- Very similar to spin glass Hamiltonian
- Generally a np-complete problem, we can use the same techniques.
- Often steepest descent is used, (greedy method): change the site that would increase the modularity the most.

iWiW vs. counties: aggregate connections between cities

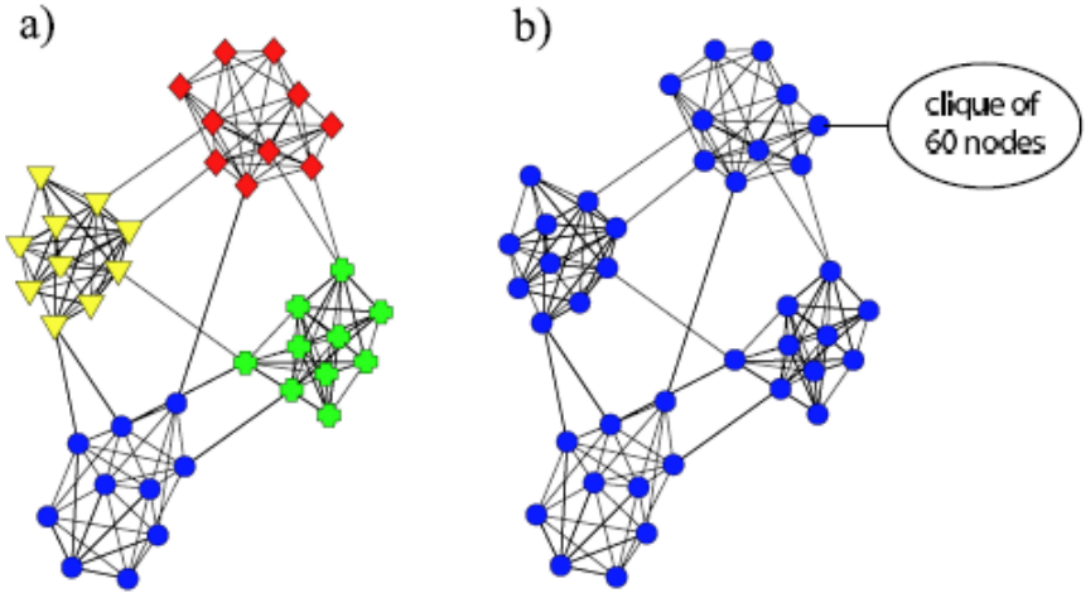


Problems with modularity

Resolution

$$Q = \frac{1}{4m} \sum_{\{i,j\}} \left[A_{ij} - \frac{k_i k_j}{2m} \right] s_i s_j$$

- On large networks normalization factor m can be very large
- (It relies on random network model)
- The expected edge between modules decreases and drops below 1
- A single link is a strong connection.
- Small modules will not be found

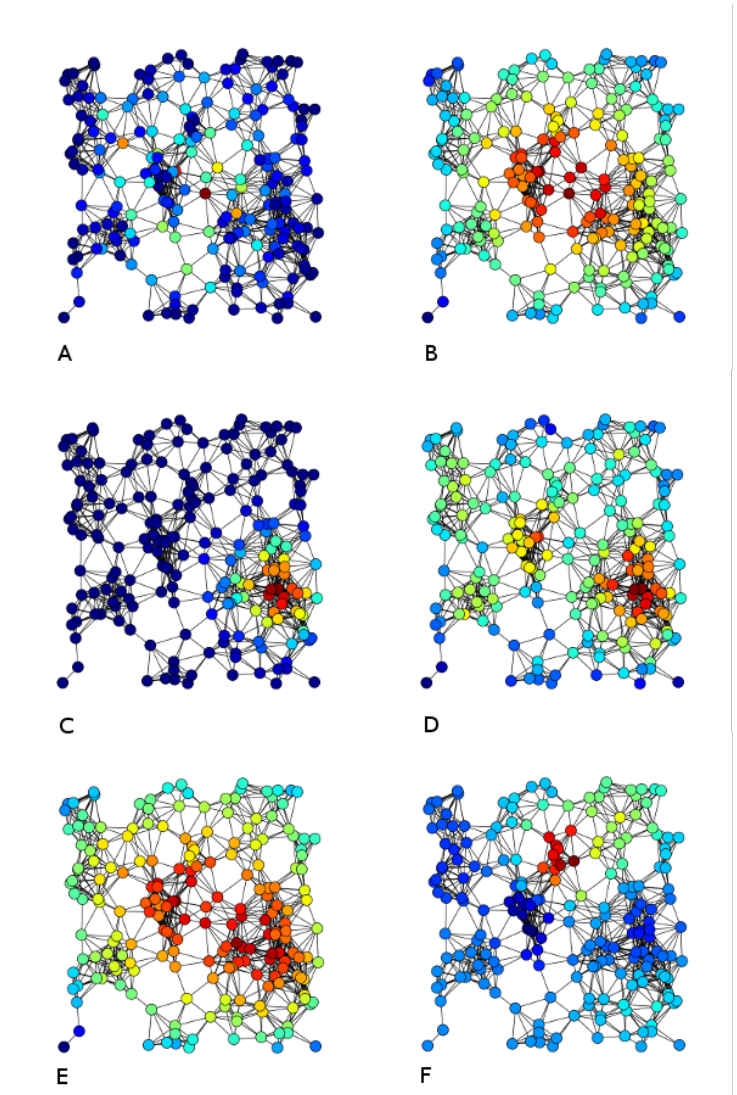


Girvan-Newman method

- Hyerarchical method
- Global decisive procedure
- Based on centrality
- Cut the edge with highest centrality
- Recalculate centrality after each cut

Centrality

Business as usual



(A) Betweenness centrality

(B) Closeness centrality

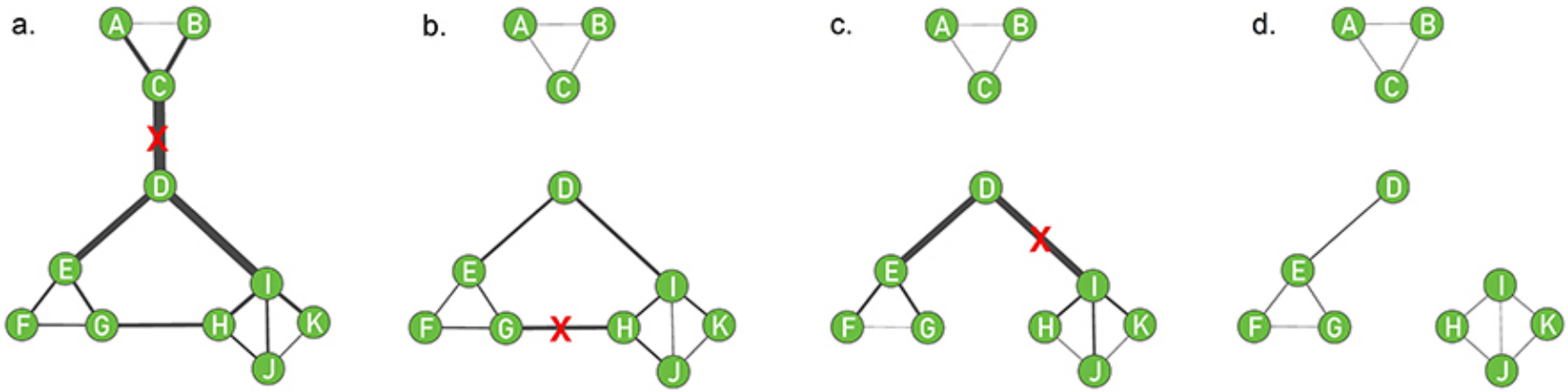
(C) Eigenvector centrality

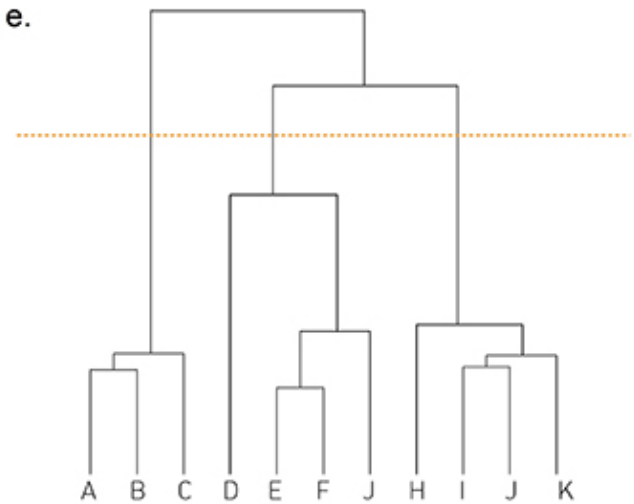
(D) Degree centrality

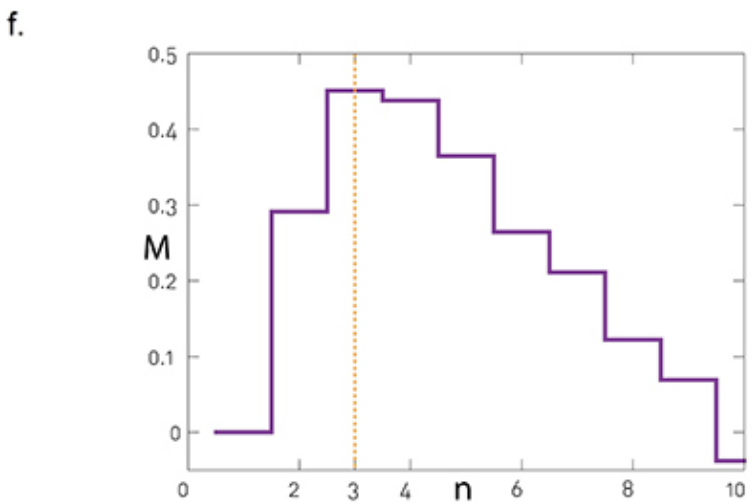
(E) Harmonic Centrality

(F) Katz centrality

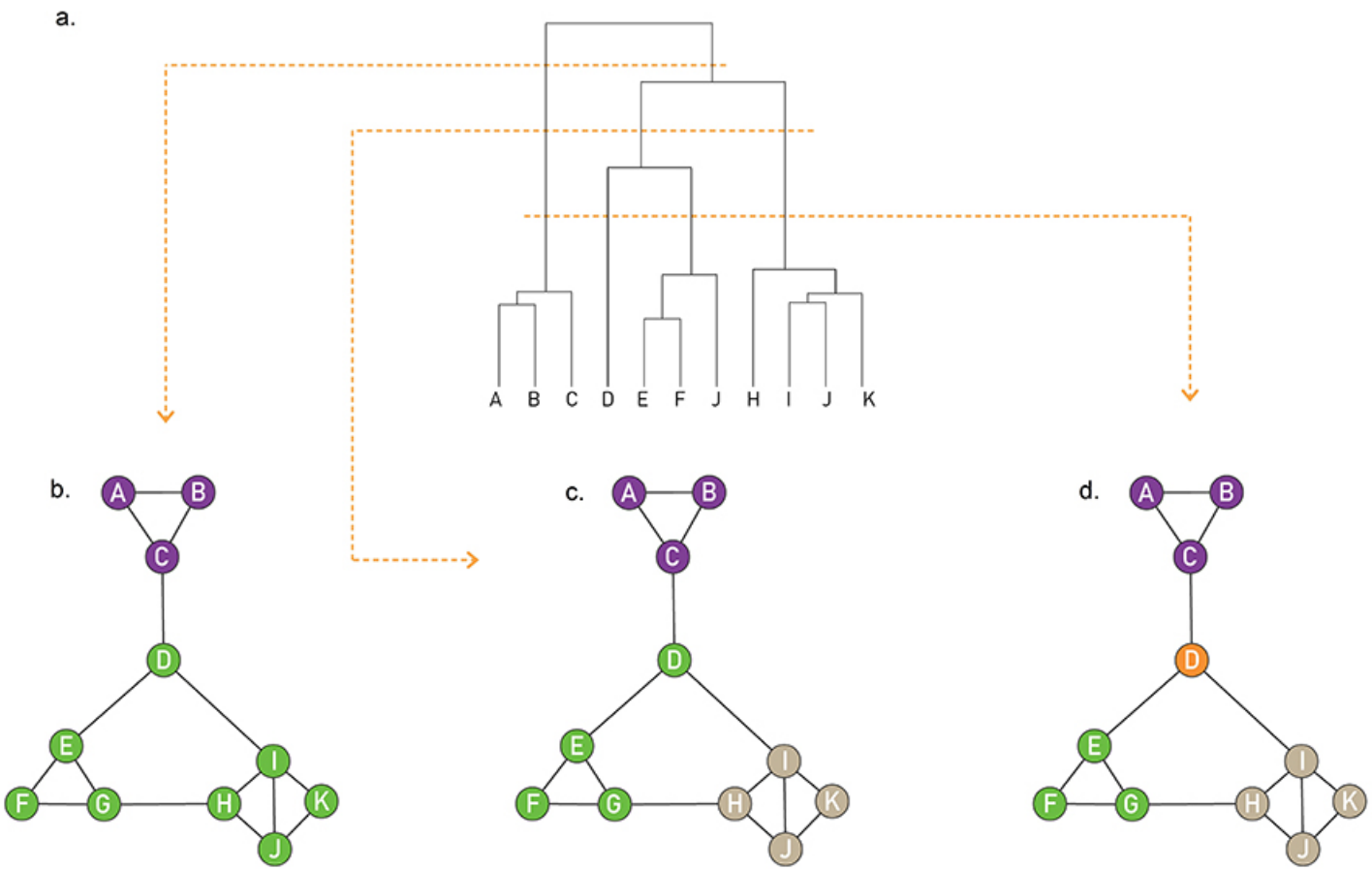
Girvan-Newman method: example





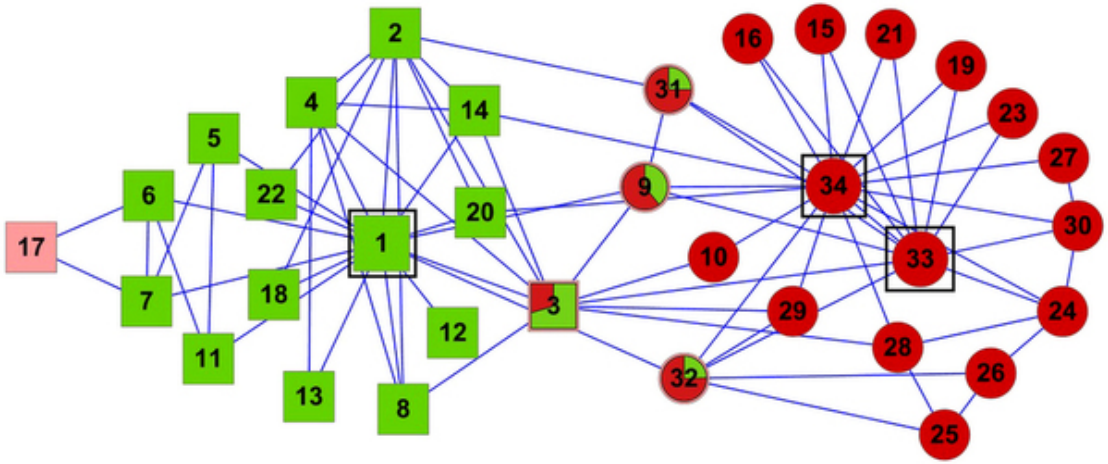


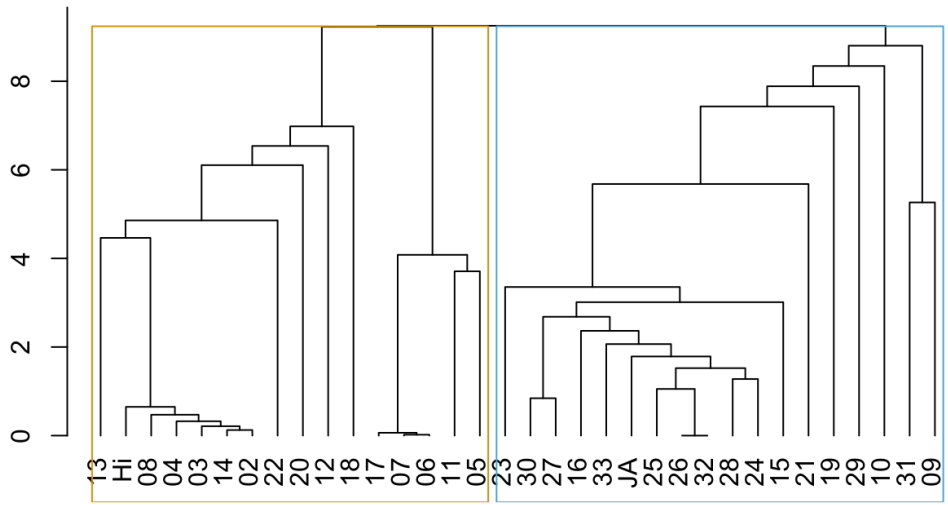
Girvan-Newman method: where to cut



- Girvan-Newman: Modularity
- Community structure almos always hierarchical
  - Kinship: family, grandparents, cousins, etc.
  - Social contacts: e.g. School, year, class, friends in the class

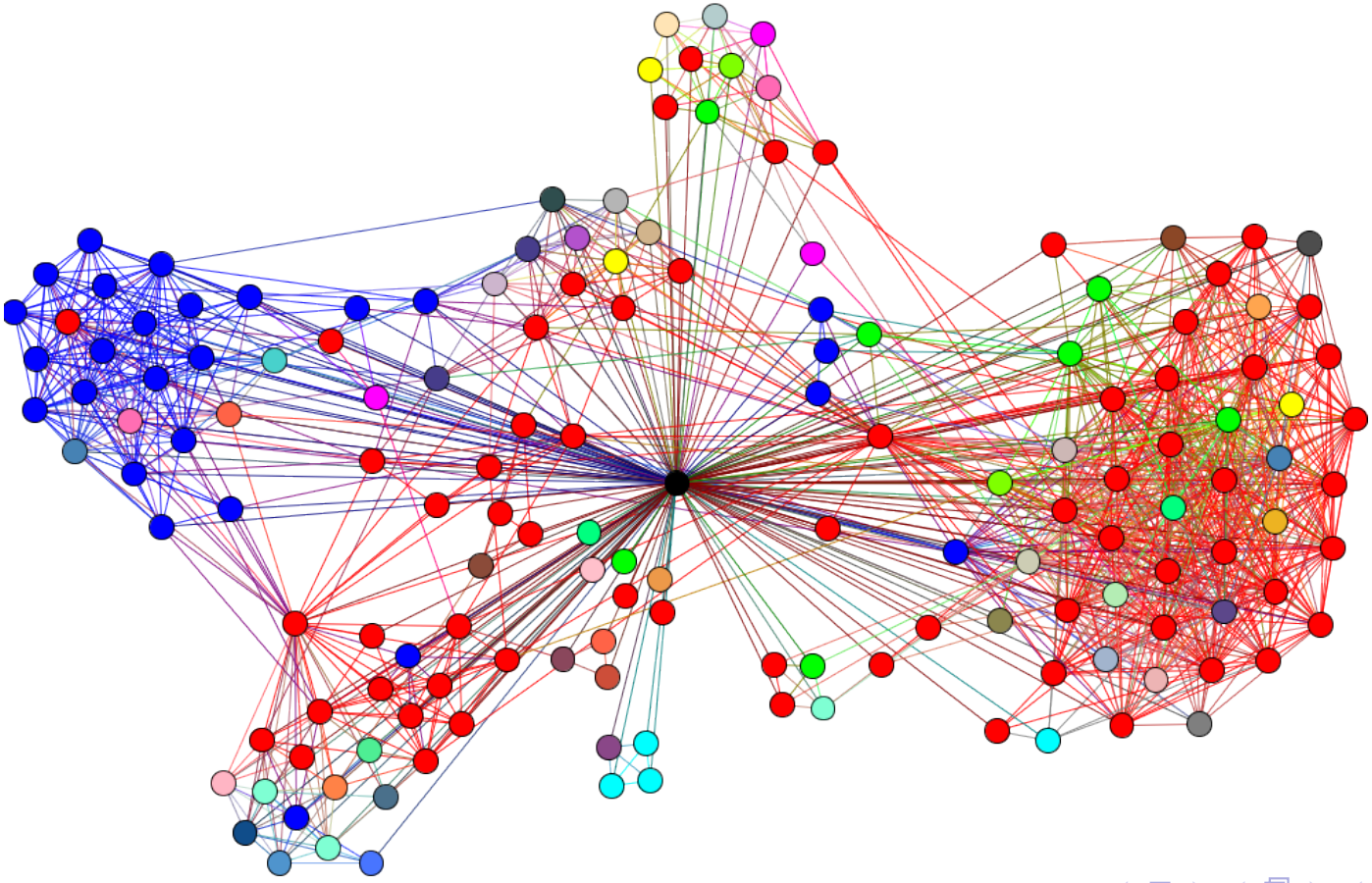
Girvan-Newman example: Zakhary





Link hierarchical method

- Aka: Link communities, Ahn method
- Nodes may belong to multiple communities
- Cluster links not nodes
- It is much rarer that a link is shared between two communities (e.g. a relative is also a collegue, or a collegue is also a friend)

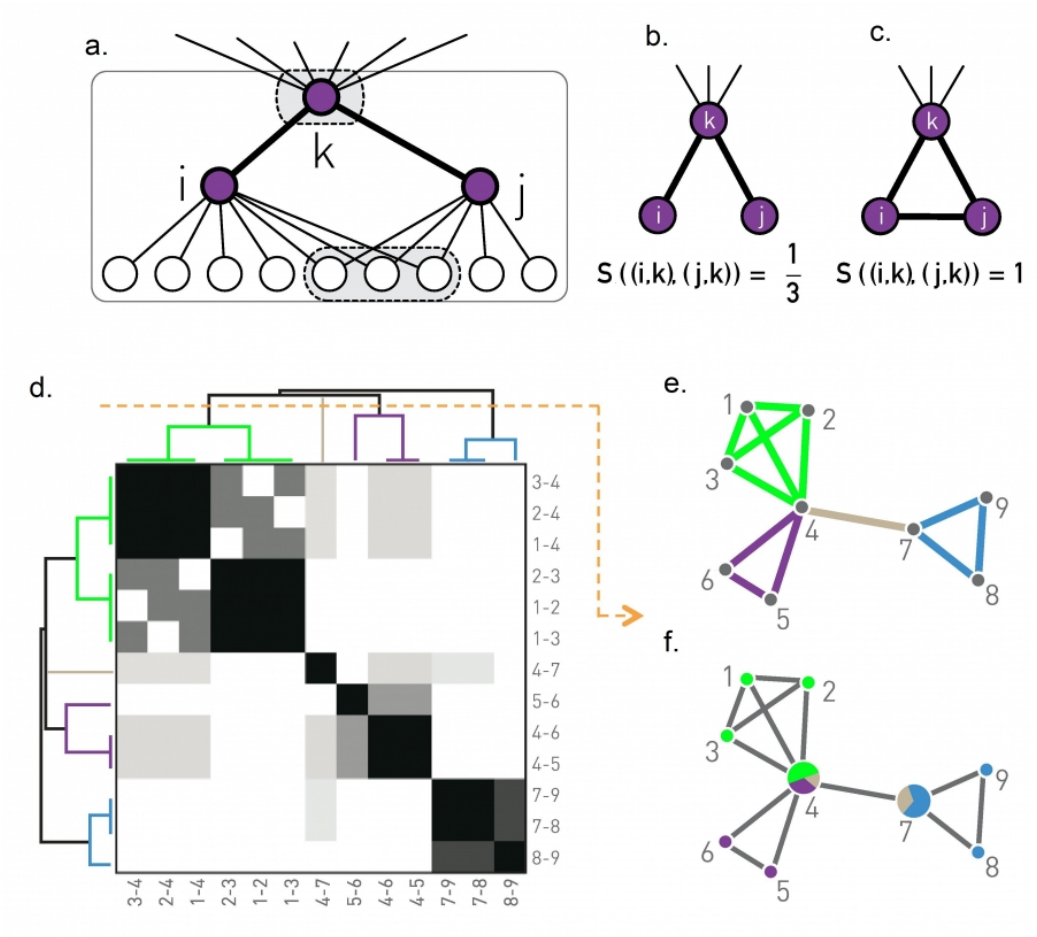


Ahn,

Ahn method

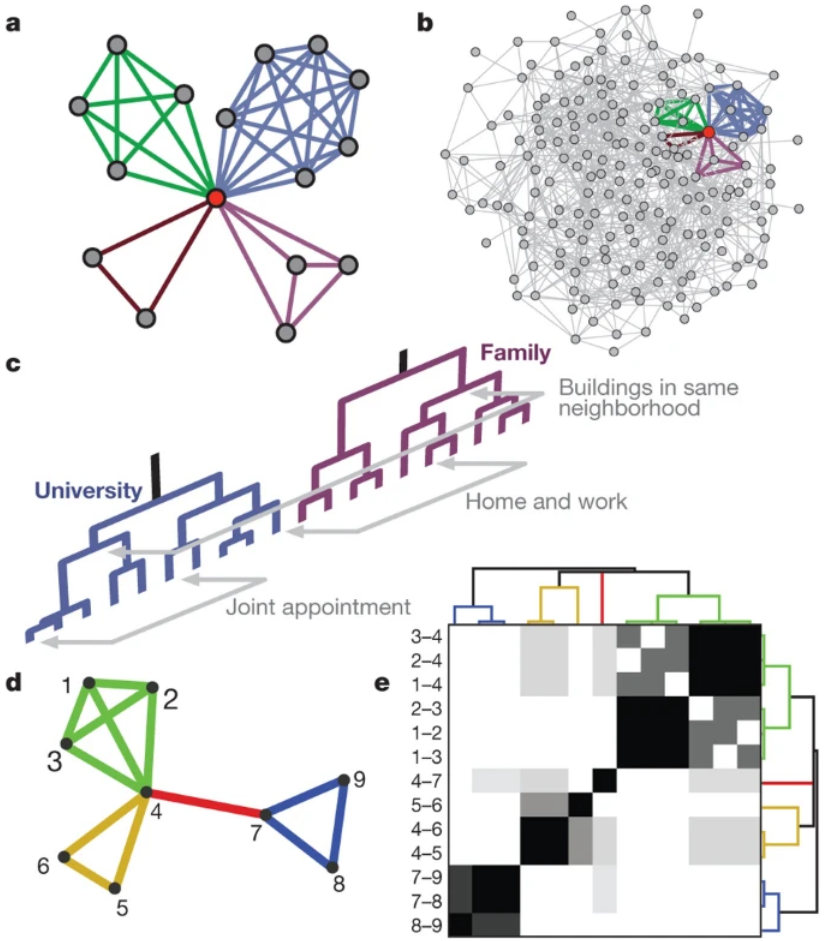
- $n_+(i)$ number of neighbors of node i including itself
- Similarity measure

$$S[(i,k),(j,k)] = \frac{|n_+(i) \cap n_+(j)|}{|n_+(i) \cup n_+(j)|}$$



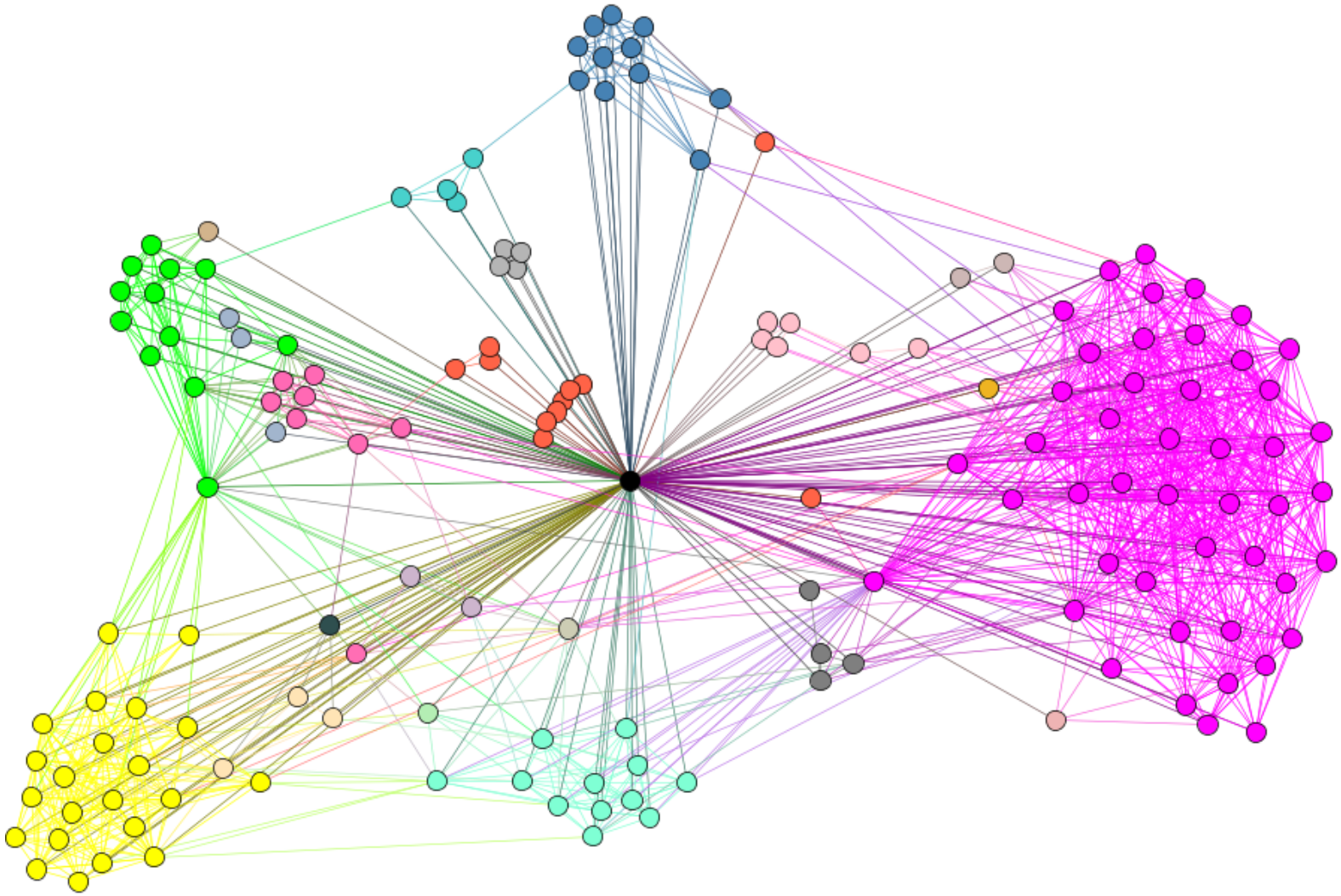
Ahn method

- Calculate link distances
- Merge clusters using single-linkage (minimal distance between two clusters)
- Cut the dendrogram according to average link density of the clusters



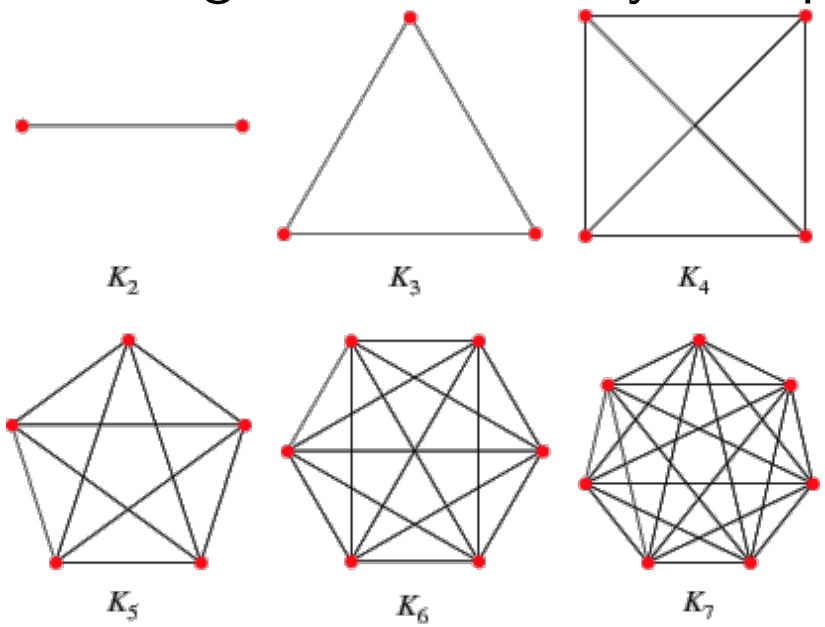
Ahn method

- Good for social networks



Clique percolation

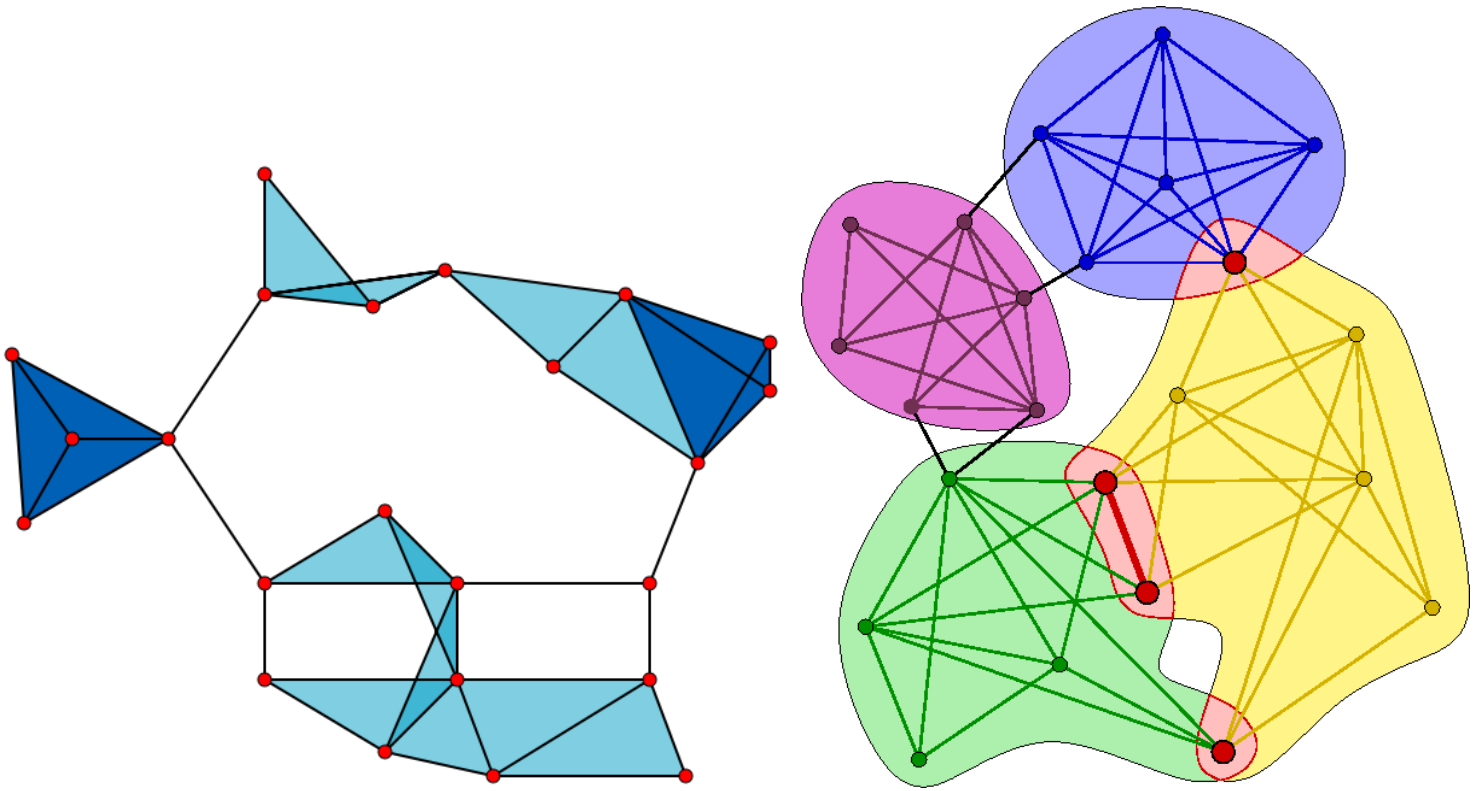
- Motivation: clusters are formed with at least triangles
- Can be generalized to any k-clique



- $k = 2$ normal percolation

Clique percolation

- It will definitely lead to overlapping communities, but overlap is limited to $k-1$ nodes
- k-clusters are included in $k-1$ clusters



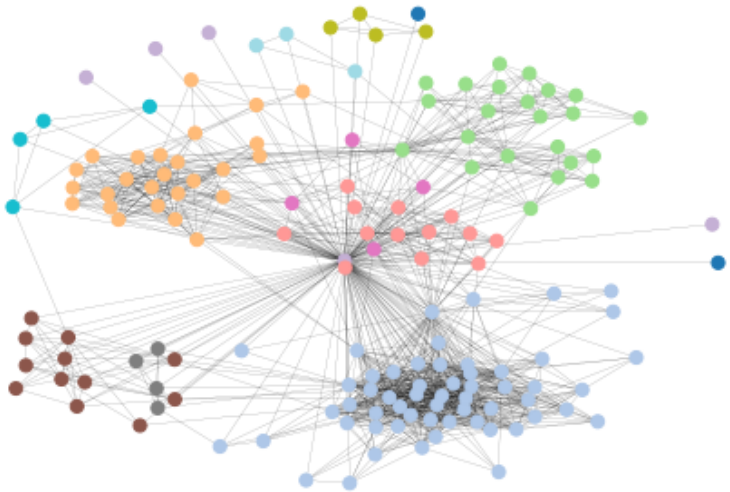
Clique percolation

- Algorithm
  - Similar to normal percolation on networks but with multiple loops
- Advantages
  - Different level of clusters
  - Clusters are generally relevant
  - No heuristics
- Disadvantages
  - Running time cannot be guessed (finding the maximal clique is an np-complete problem)
  - Code may run for ages

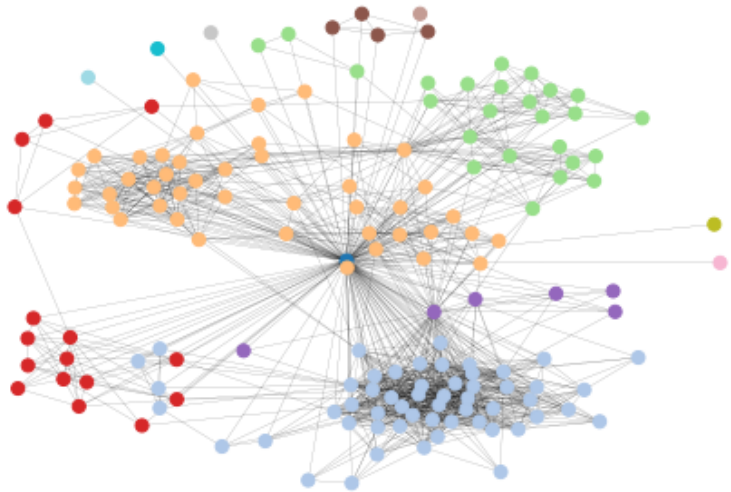
Comparison

Layout

Ahn (biggest first)



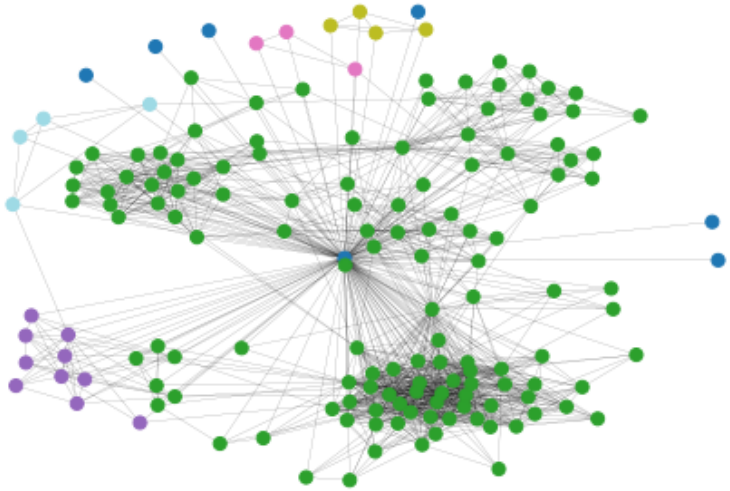
Infomap



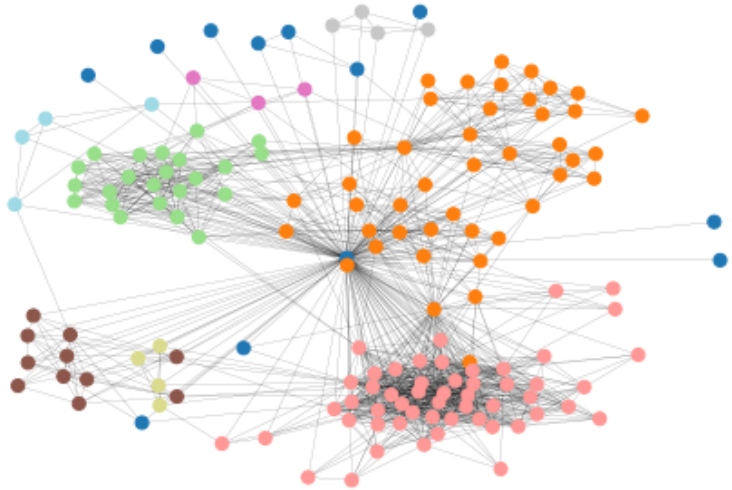
Modularity (greedy)

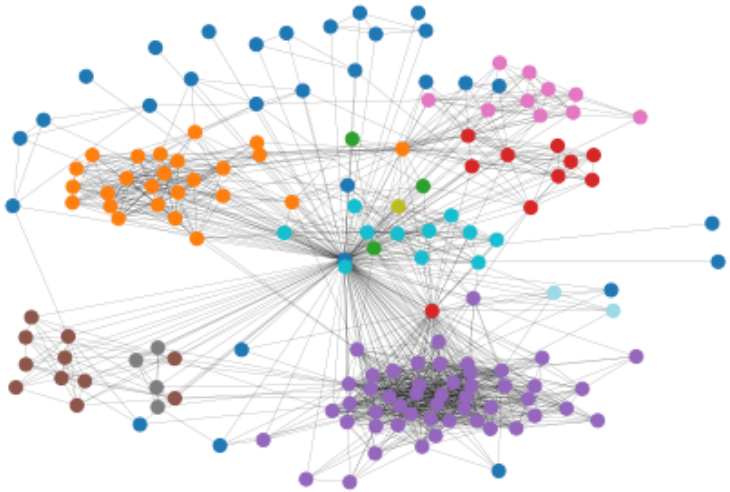
Comparison

Clique3



Clique4





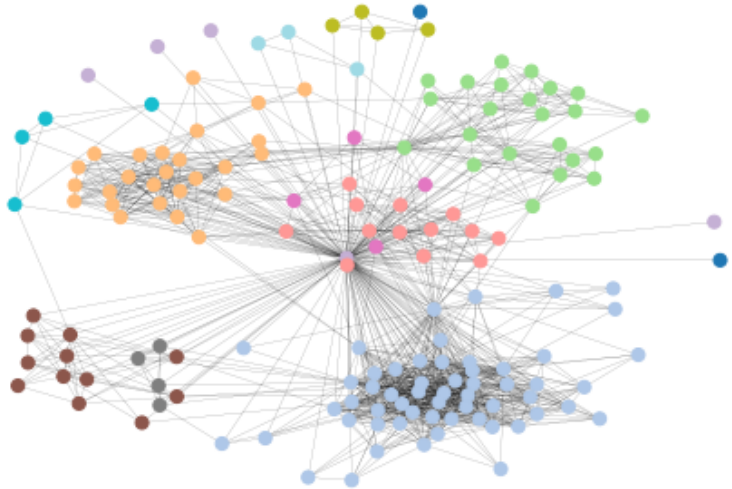
Clique5

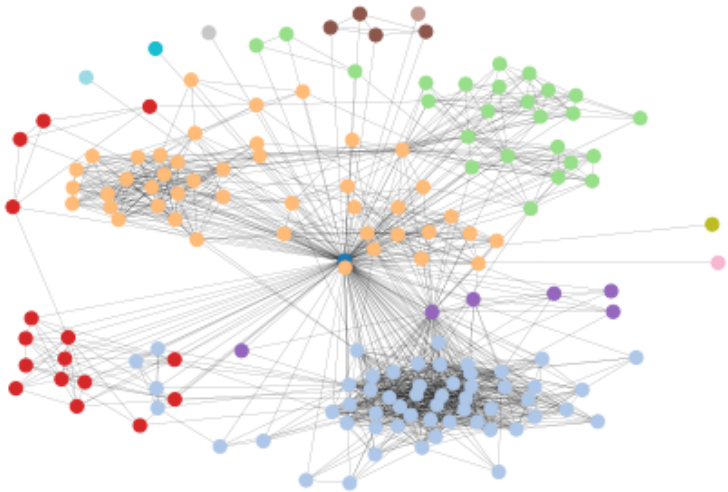
Clique6

Comparison

Ahn

Infomap



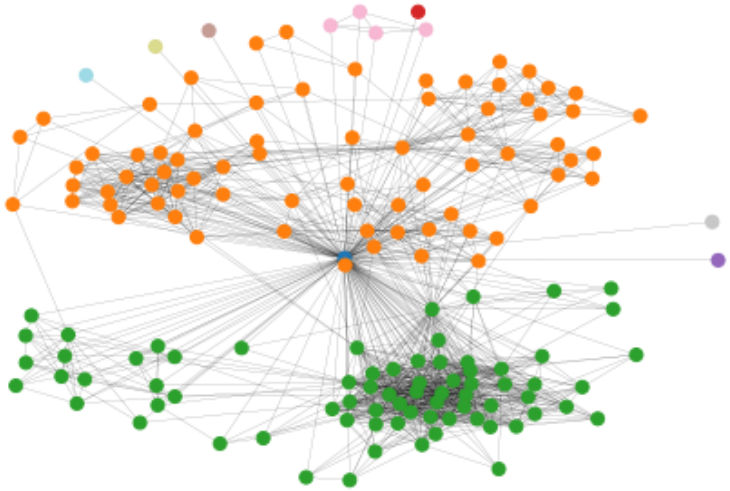


Modularity

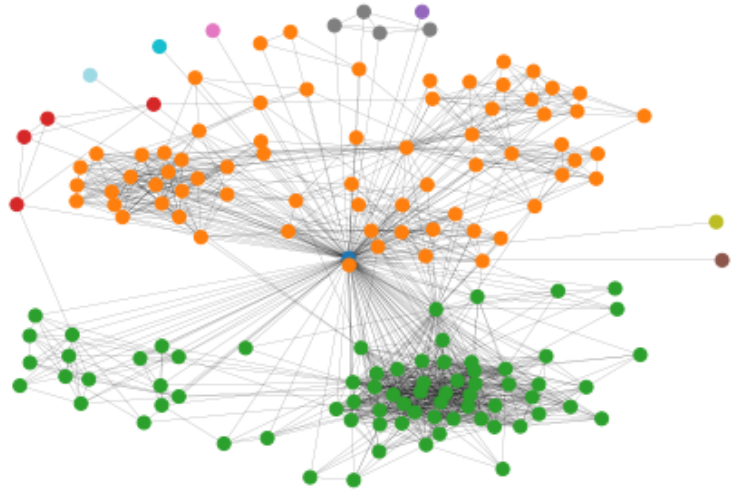
Clique 6

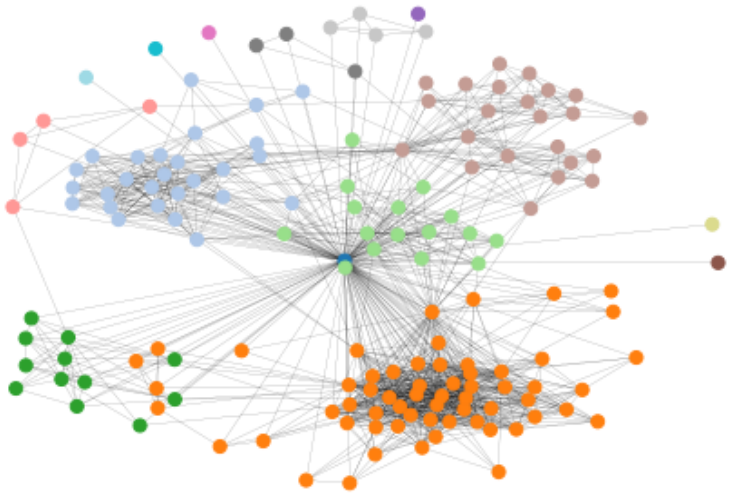
Comparison: Newman-Girvan

Layer 0

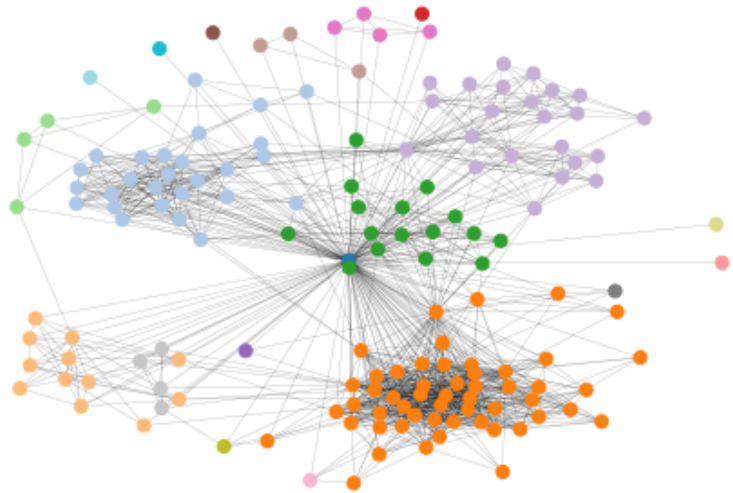


Layer 1





Layer 5

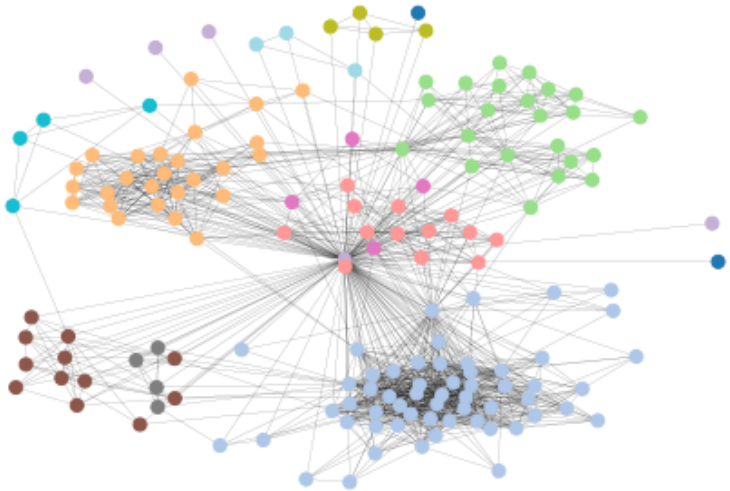


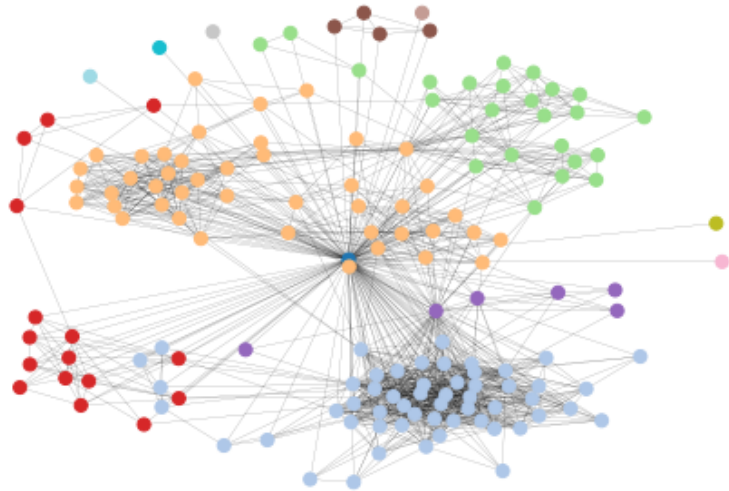
Layer 10

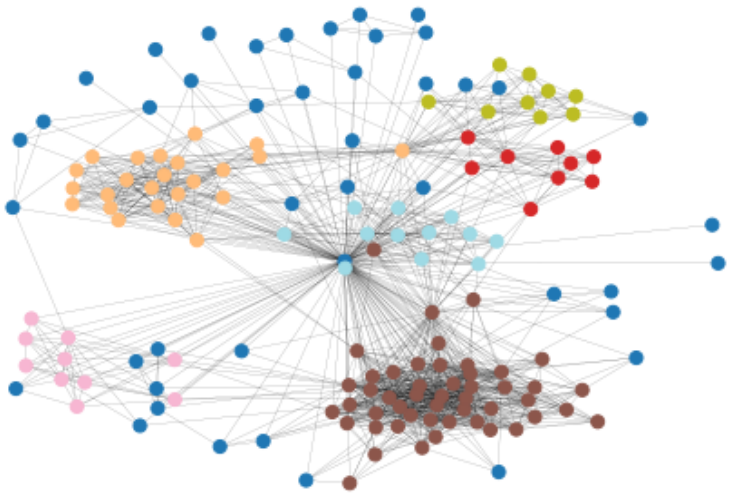
Comparison

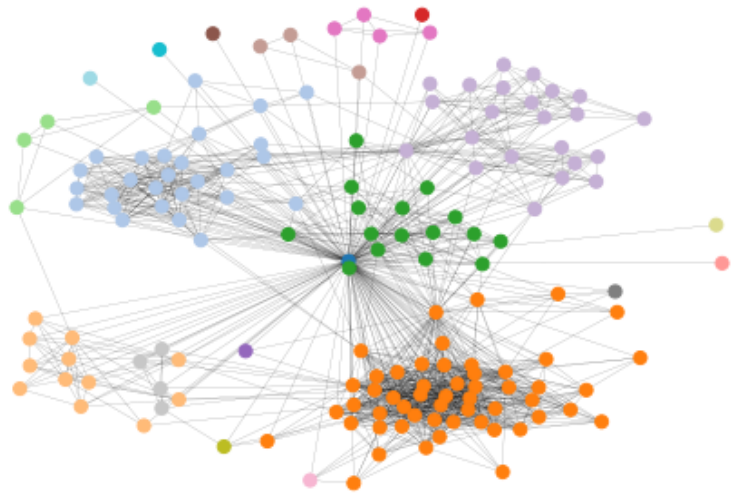
Infomap

Modularity (greedy)









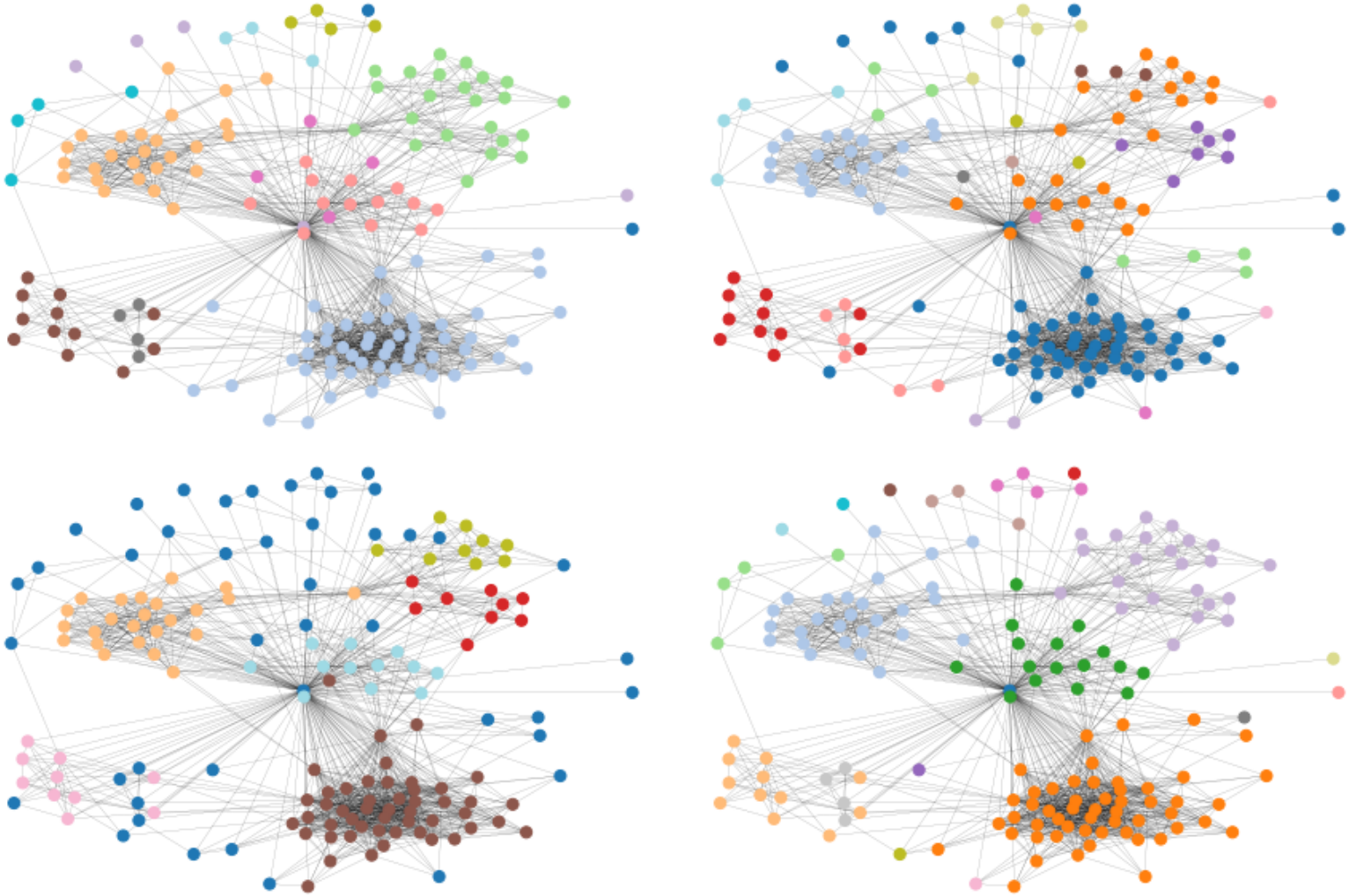
Clique 6

Newman-Girvan layer 10

Comparison

Infomap

Link hierarchical

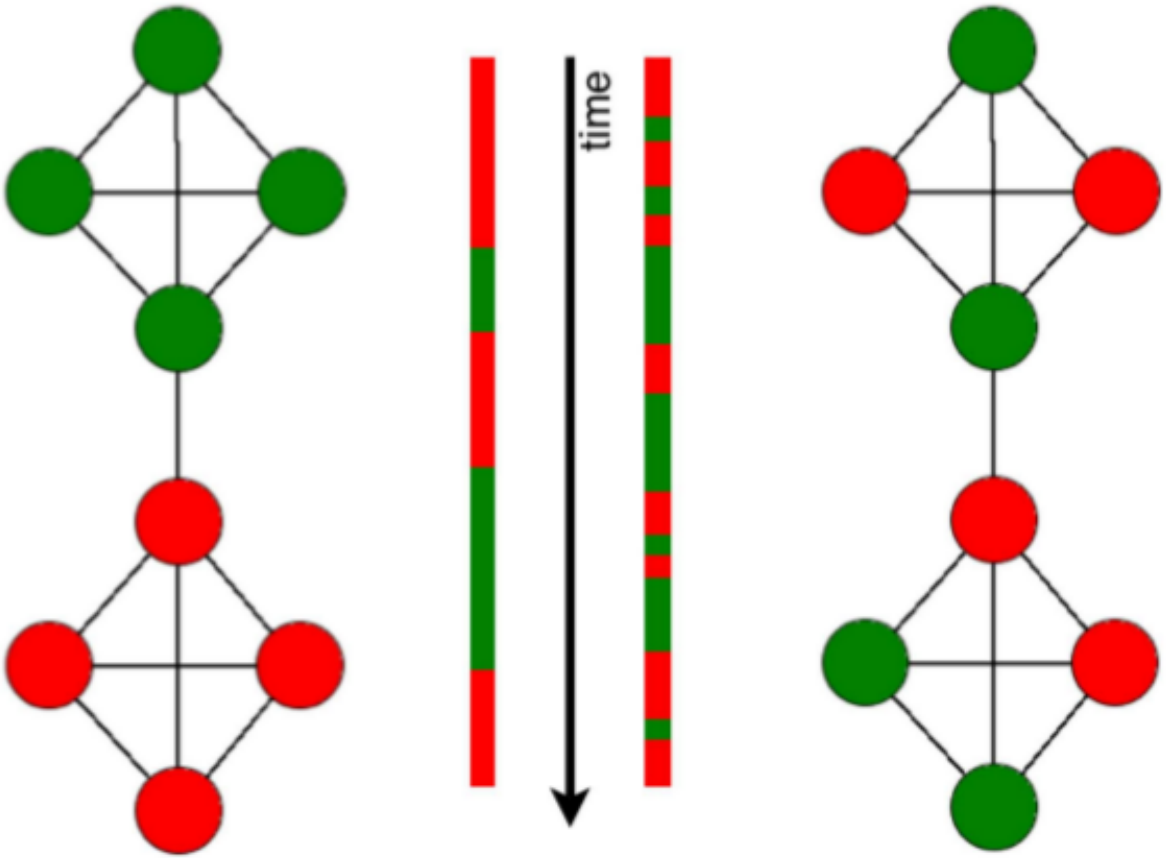


Clique 6

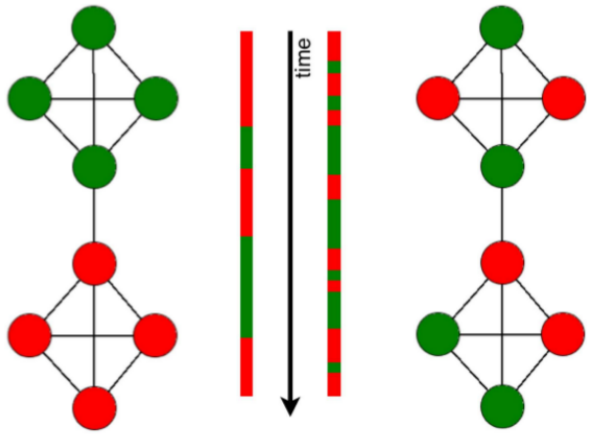
Newman-Girvan layer 10

Dynamical networks

- Almost all networks evolve in time
- How is the dynamics related to the structure?
- Let us study a random walker:



Dynamical networks



- Let $P(C,t)$ e the probability that a walker is in community C at time 0 and t.
- Partition stability measure:

$$R = \sum_C [P(C,t) - P(C,\infty)]$$

- $P(C,\infty)$ is the probability to find two independent random walkers in the same community

Dynamical networks: Stability of partitions

- Let $P(C,t)$ e the probability that a walker is in community C at time 0 and t.
- Partition stability measure:

$$R(t) = \sum_C [P(C,t) - P(C,\infty)]$$

- $P(C,\infty)$ is the probability to find two independent random walkers in the same community
- If the partition is stable $R(t)$ decays slowly
- It can be used as a resolution parameter

Dynamical networks: Stability of partitions

- Use an undirected graph
- Define a discrete random walk
- On a given node i the density of the random walkers is

$$p_i(t+1) = \sum_j \frac{A_{ji}}{k_j^{\text{out}}} p_j(t)$$

  with $k_j^{\text{out}} = \sum_i A_{ji}$
- For unweighted, undirected network $A_{ij} = A_{ji}$ is the adjacency matrix
- The stationary density is

$$p_i^* = \frac{k_i}{2L}$$

Dynamical networks: Stability of partitions

- The probability that a walker is in community C is

$$\sum_{j \in C} \frac{k_j}{2L}$$

- The probability of a random walker to be in C during two successive time steps is

$$\sum_{i,j \in C} \frac{A_{ij}}{k_j} \frac{k_j}{2L}$$

- The stability at time $t = 1$:

$$R(t{=}1) = \sum_{C} \sum_{i,j \in C} \left[\frac{A_{ij}}{k_j} \frac{k_j}{2L} - \frac{k_i}{2L} \frac{k_j}{2L} \right] = Q$$

Dynamical networks: Stability of partitions

- The stability at time $t = 1$:

$$R(t{=}1) = \sum_C \sum_{i,j \in C} \left[\frac{A_{ij}}{k_j} \frac{k_j}{2L} - \frac{k_i}{2L} \frac{k_j}{2L} \right] = Q$$

- New, independent introduction of modularity
- assumption of the configuration model as null model naturally appears
- For weighted network $R(t{=}1)$ takes a different form

Dynamical networks: Limits of $R(t)$

- Continuous timescale, with $B_{ij} = A_{ij}/k_j$:

$$\dot{p}_i(t) = \sum_j \frac{A_{ij}}{k_j} p_j - p_i$$

$$\dot{\mathrm{p}}(t) = (\mathbf{B\text{-}I})\mathrm{p}(t)$$

- Solution is an exponential

$$\mathrm{p}(t) = e^{(B-I)t}\mathrm{p}(0)$$

$$R(t) = \sum_C \sum_{i,j\in C} \left[e^{(\mathbf{B\text{-}I})t} \frac{k_j}{2L} - \frac{k_i}{2L}\frac{k_j}{2L} \right]$$

- For t=0

$$R(t) = 1 - \sum_C \sum_{i,j\in C} \frac{k_i}{2L}\frac{k_j}{2L}$$

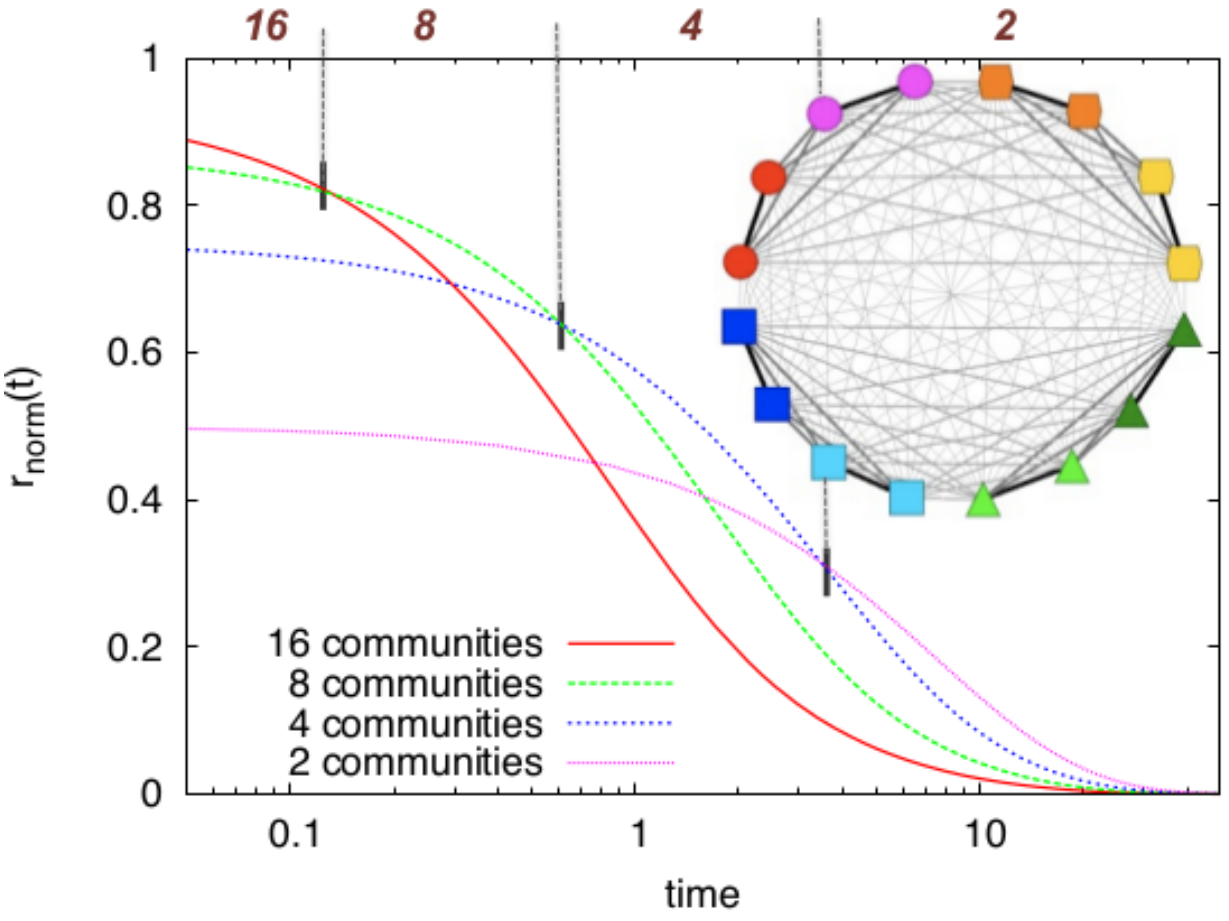
- Maximum with every node as separate community

Dynamical networks: Limits of $R(t)$

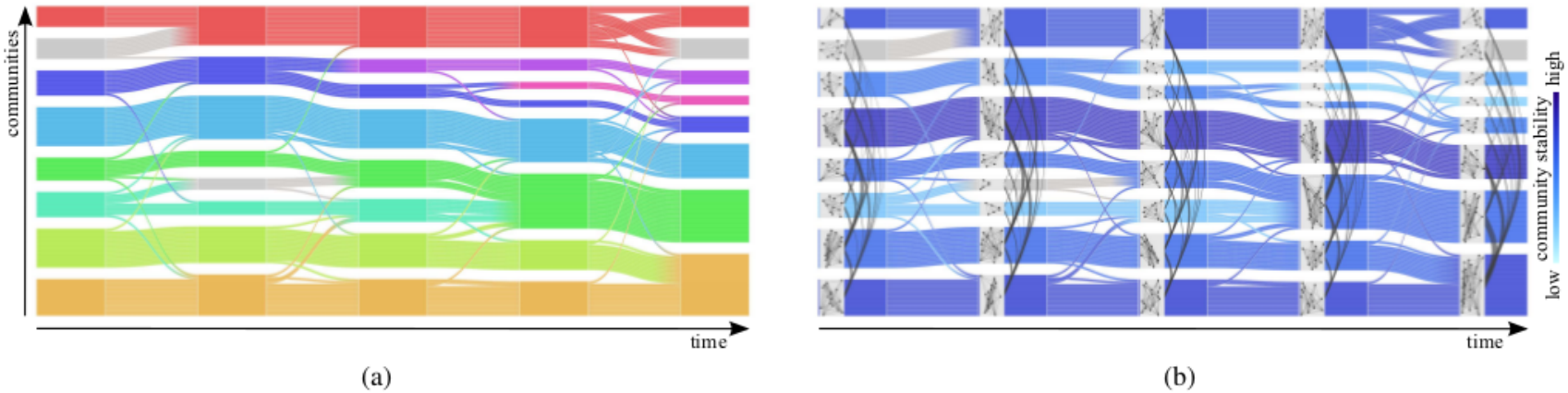
- For t=0

$$R(t) = 1 - \sum_{C} \sum_{i,j \in C} \frac{k_i}{2L} \frac{k_j}{2L}$$

- Maximum with every node as separate community
- For larger times two communities

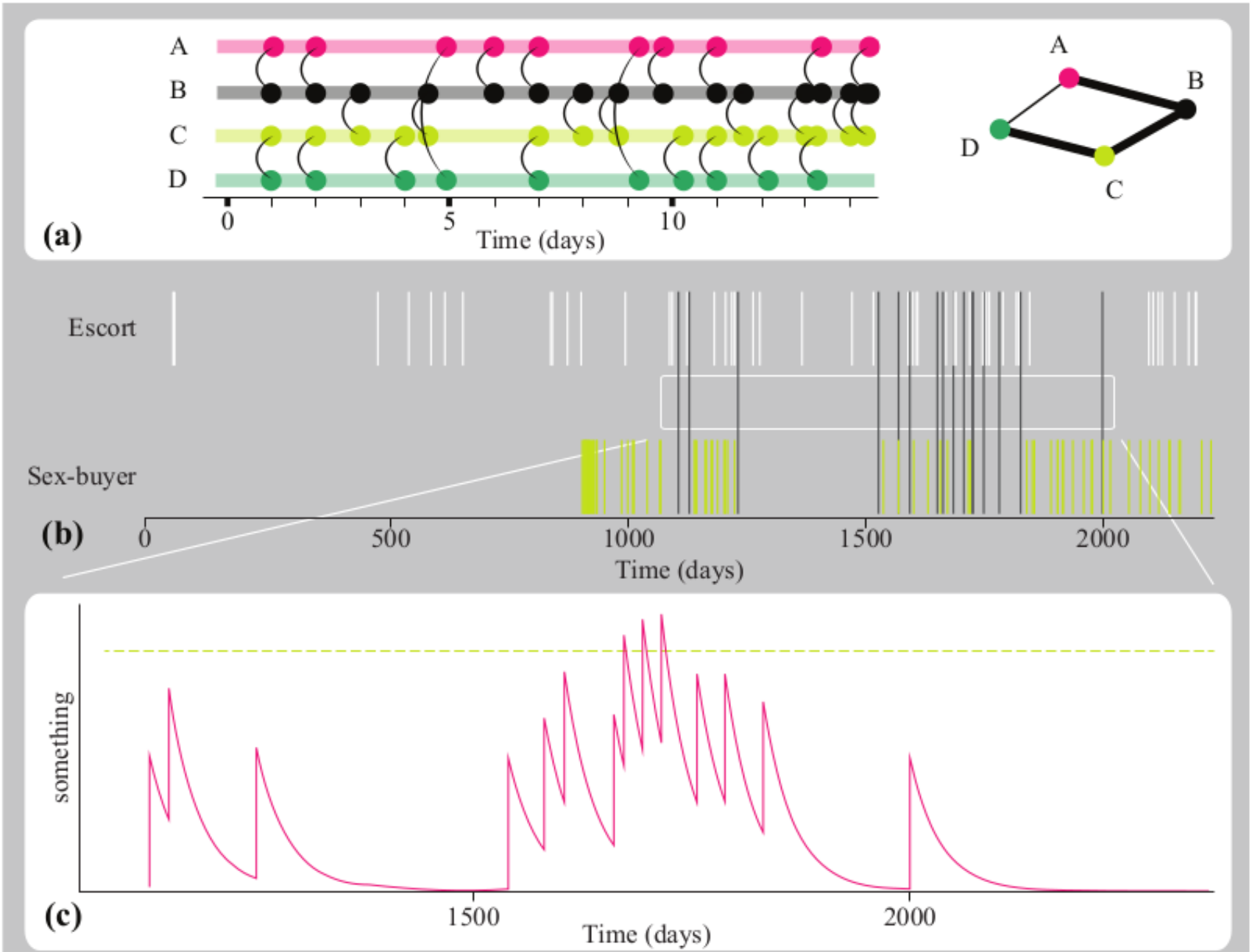


Temporal communities

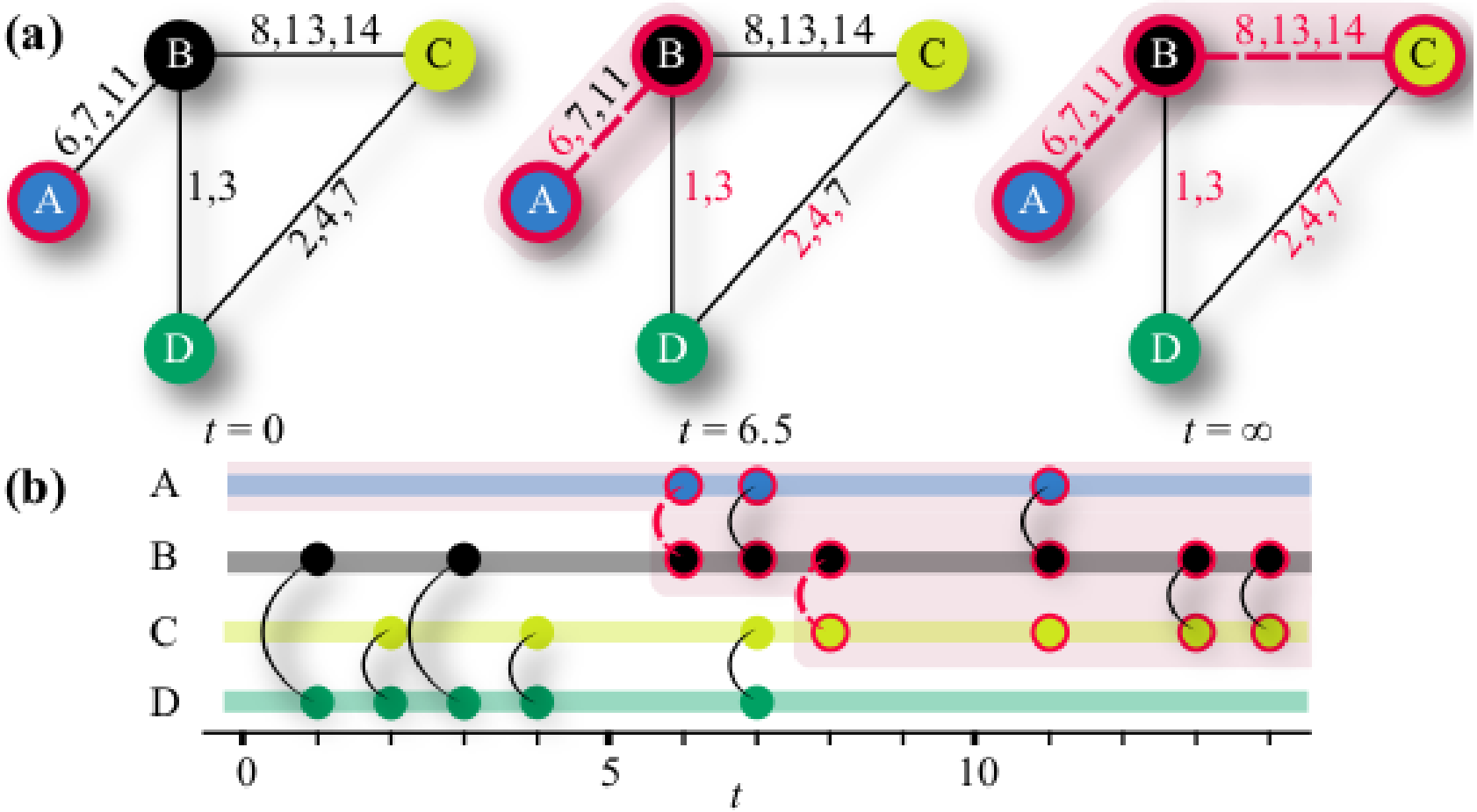


- Communities come and go
- Merge and split
- Connections fade and strengthen

Temporal data



Temporal data



- Slices in time s
- A_{ijs} coupling between nodes
- C_{jrs} coupling between slices

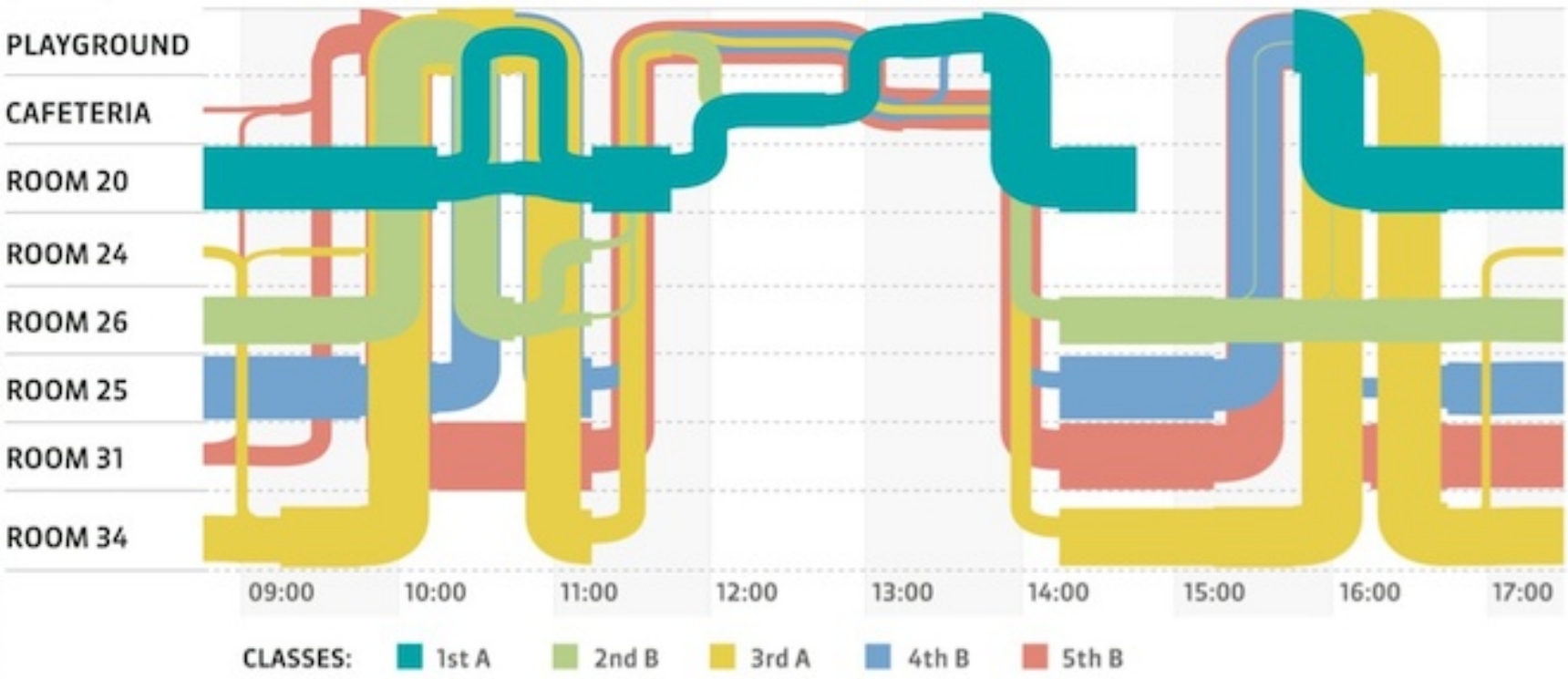
Temporal data

- Slices in time s
- A_{ijs} coupling between nodes
- C_{jrs} coupling between slices
- $k_{js} = \sum_i A_{ijs}$
- $c_{js} = \sum_r C_{jrs}$
- $\kappa_{js} = k_{js} + c_{js}$
- New quality function:

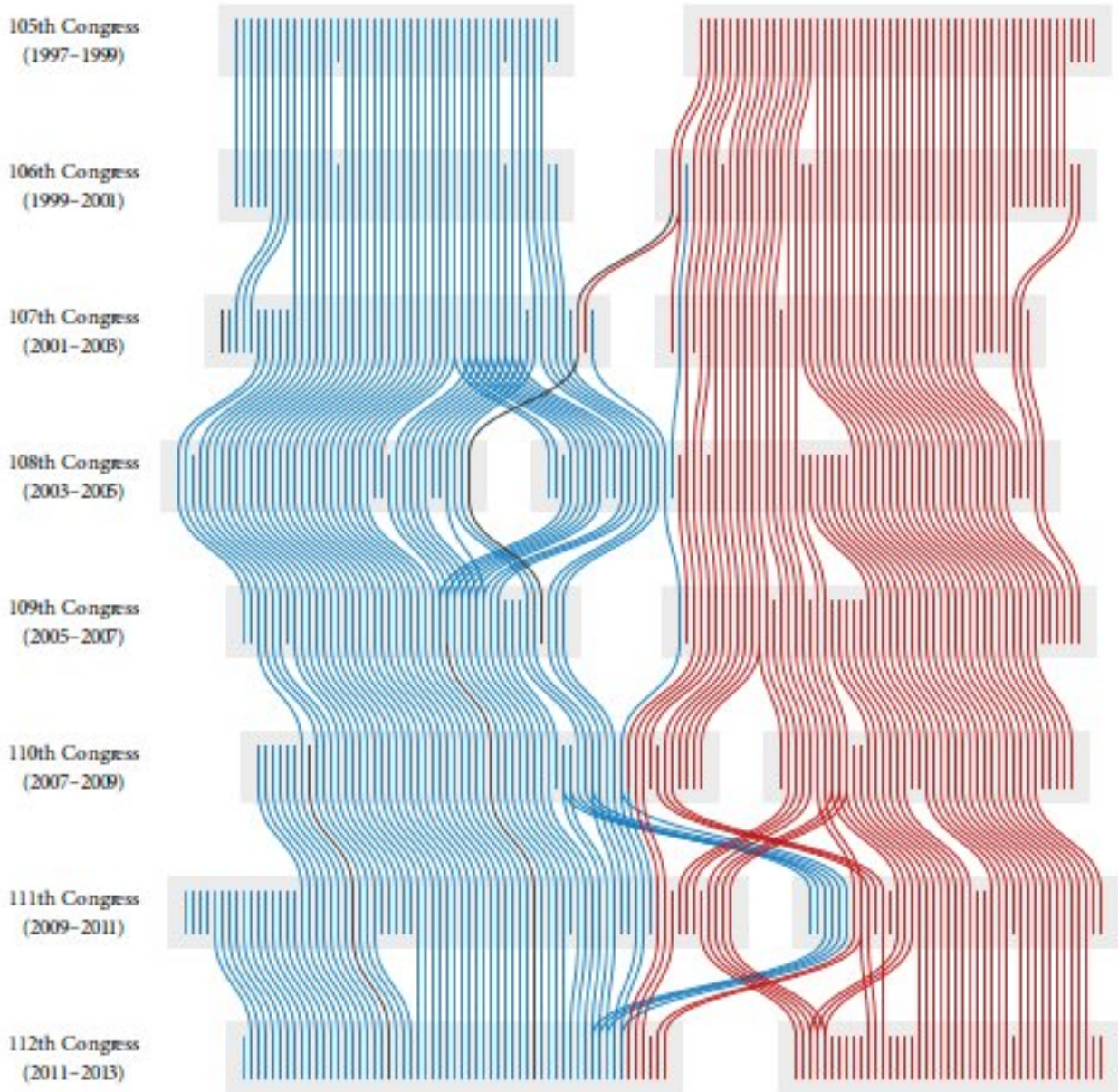
$$Q(t) = \frac{1}{2\mu} \sum_{ijsr} \left\{ \left(A_i js - \gamma_s \frac{k_{is} k_{js}}{2m_s} \right) \delta_{sr} + \delta_{ij} C_{jsr} \right\} \delta(g_{is}, g_{jr})$$

- where γ_s is the resolution parameter

Temporal data: Examples



Temporal data: Examples



Temporal data: Examples