Finding modules on graphs

Sushmita Roy
sroy@biostat.wisc.edu

Computational Network Biology
Biostatistics & Medical Informatics 826
https://compnetbiocourse.discovery.wisc.edu

Nov 1st 2018

Goals for today

- Finding modules on graphs/Community structure on graphs/Graph clustering
- Girvan-Newman algorithm
- Hierarchical agglomerative clustering on graphs
  - Stochastic block model
  - Hierarchical random graph model
  - Combining the two

RECAP

- Network structure is often analyzed for different topological properties.
  - Degree distribution
  - Diameter
  - Clustering coefficient/Modularity measures
  - Network motifs
- Network modularity questions
  - How to measure modularity
  - How to find modules
- Algorithms for clustering

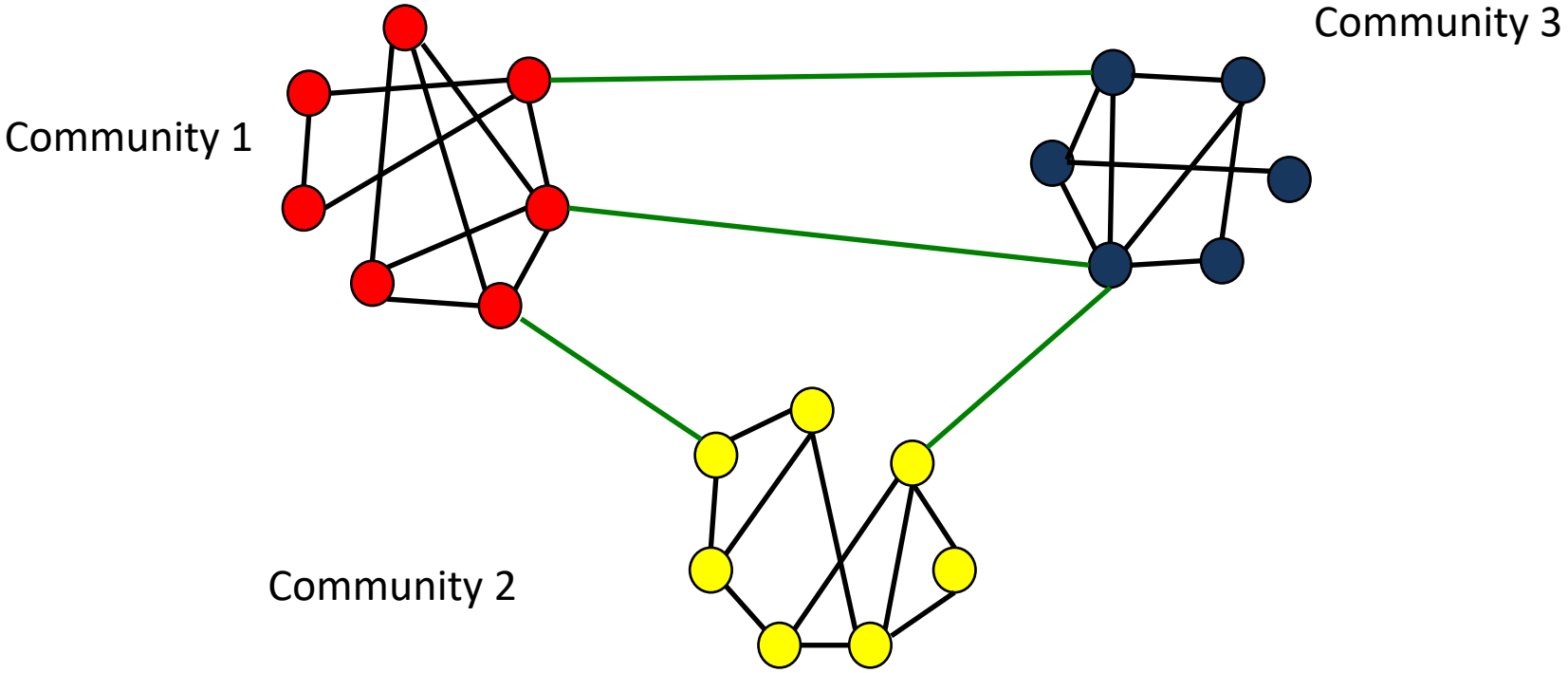
RECAP contd

- Flat clustering
  - Cluster objects into K clusters
  - K : number of clusters is a user defined argument
  - Popular algorithms
    - K-means
    - Gaussian mixture models
- Hierarchical clustering
  - Instead of the number of clusters, it requires us to specify how much dissimilarity we will tolerate between groups
  - Can be top-down (divisive) or bottom up (agglomerative)

Graph clustering enables community structure detection

Graph clustering partitions vertices into groups based on their interaction patterns

Such partitions could be indicative of the specific properties of the vertices/form communities



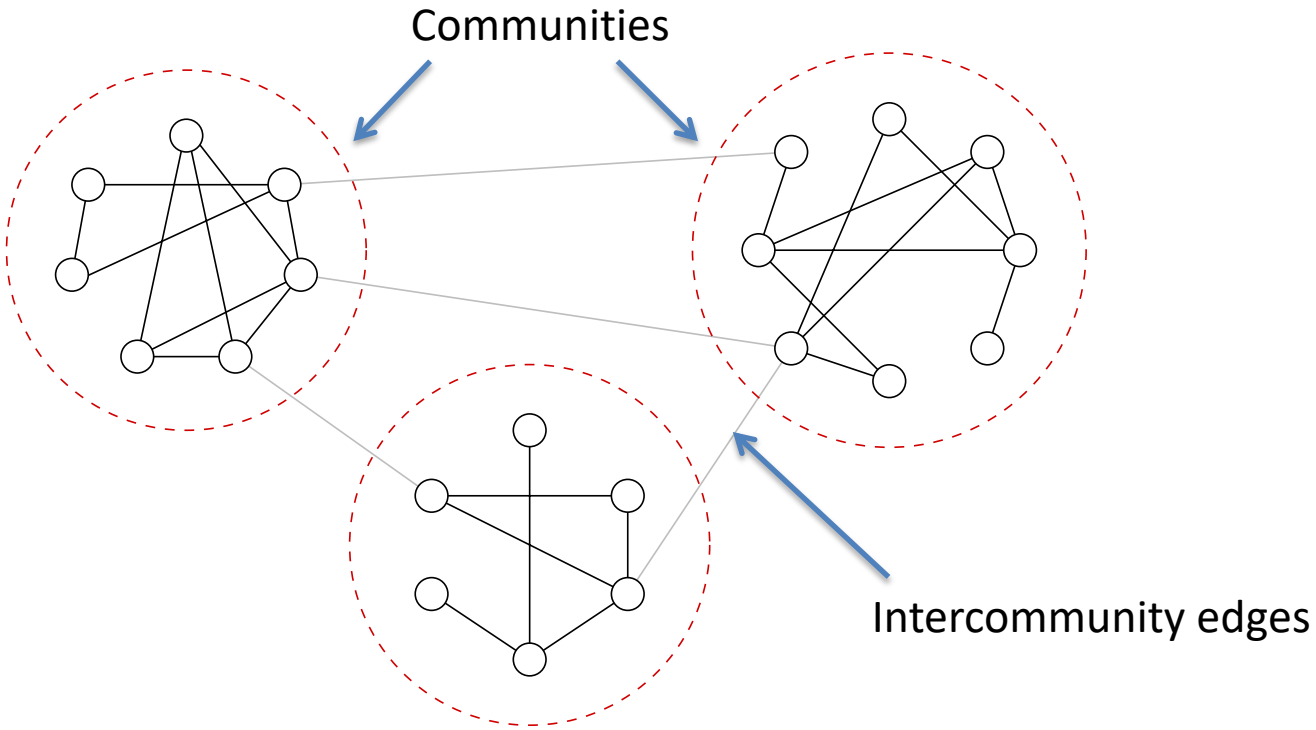
Common graph clustering algorithms

- Hierarchical or flat clustering using a notion of similarity between nodes
- Girvan-Newman algorithm
- Hierarchical Agglomerative clustering
- Spectral clustering
- Markov clustering algorithm
- Affinity propagation

Goals for today

- Finding modules on graphs/Community structure on graphs/Graph clustering
- Girvan-Newman algorithm
- Hierarchical agglomerative clustering on graphs
  - Stochastic block model
  - Hierarchical random graph model
  - Combining the two

Motivation of the Girvan-Newman algorithm



- A graph that has a grouping (community) structure is going to have few intercommunity edges.
- Community structure can be revealed by removing such intercommunity edges

Detecting community structure in networks, M. E. J. Newman

Girvan-Newman algorithm

- General idea: “If two communities are joined by only a few inter-community edges, then all paths through the network from vertices in one community to vertices in the other must pass along one of those few edges.”
- Community structure can be revealed by removing edges that with high betweenness
- Algorithm is based on a divisive clustering idea

M. E. J. Newman and M. Girvan. Finding and evaluating community structure

Betweenness of an edge

- Betweenness of an edge e is defined as the number of shortest paths that include e
- Edges that lie between communities tend to have high betweenness

$$B(e) = \frac{\text{Shortest path including } e}{\text{Number of total shortest paths}}$$

Girvan-Newman algorithm

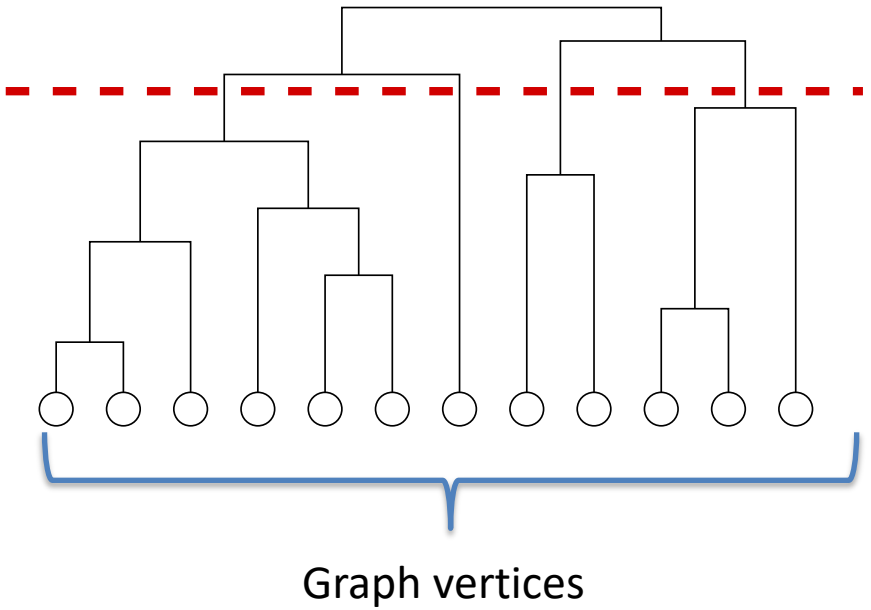
- Initialize
  - Compute betweenness for all edges
- Repeat until convergence criteria
  1. Remove the edge with the highest betweenness
  2. Recompute betweenness of remaining edges

- Convergence criteria can be
  - No more edges
  - Desired modularity

M. E. J. Newman and M. Girvan. Finding and evaluating community structure

Girvan-Newman algorithm as a hierarchical clustering algorithm

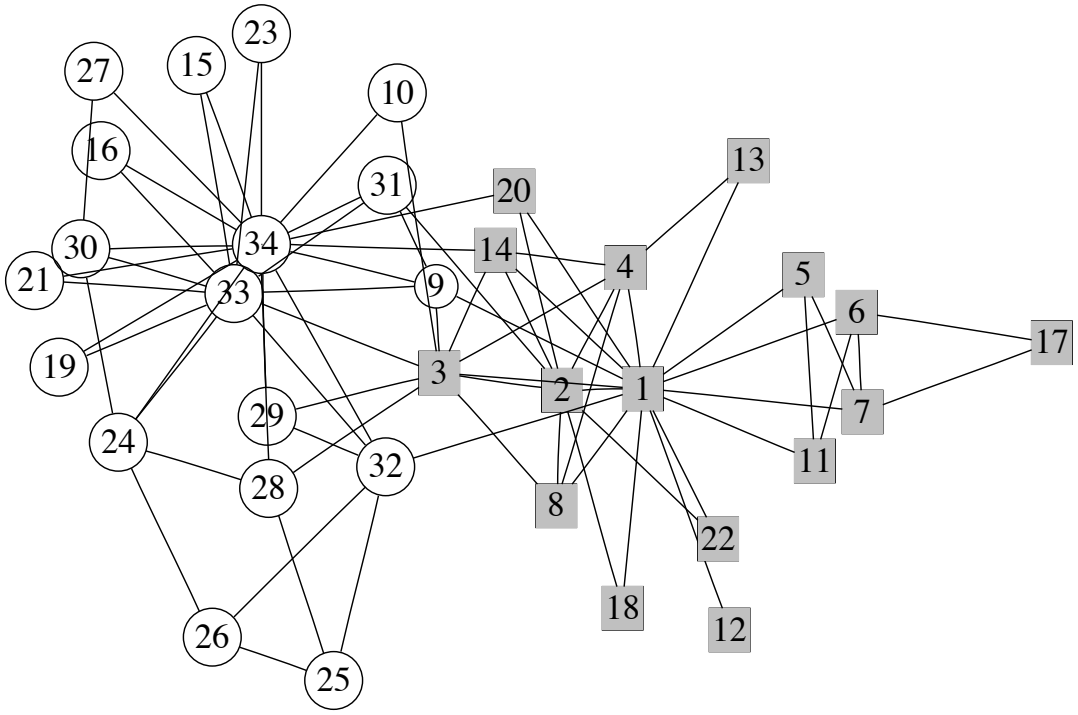
- One can view this algorithm as a top-down (divisive) hierarchical clustering algorithm
- The root of the dendrogram groups all nodes into one community
- Each branch of the tree represents the order of splitting the network as edges are removed

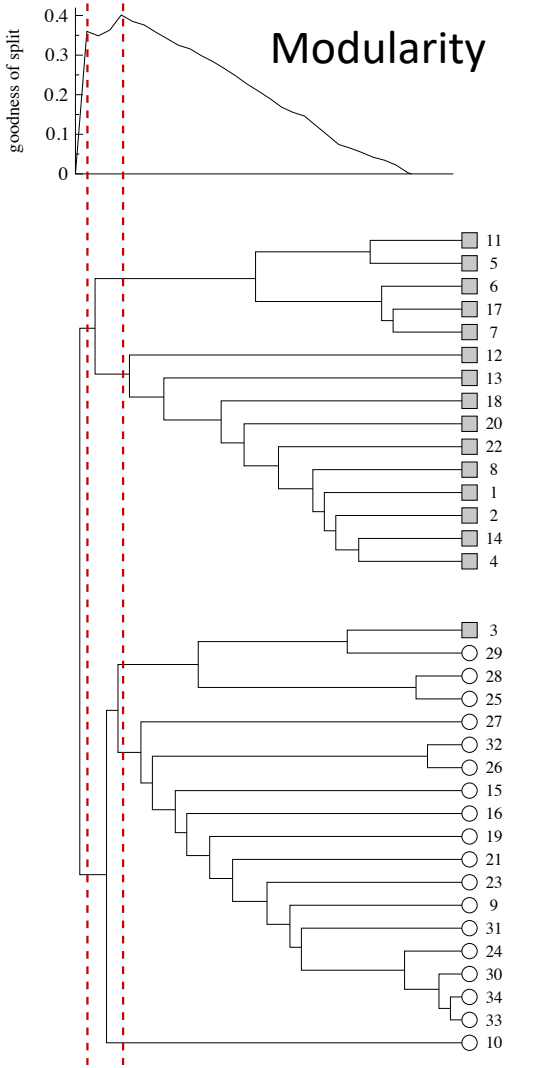


Applying the Girvan-Newman algorithm to Zachary's karate club network

- Dataset collected by Wayne Zachary over 2 years who observed social interactions among members of a karate club
- Zachary's karate club network is a well-known example of a social network with community structure
- Network represents the friendships among members of a karate club
- Due to a dispute the club split into two factions
- Can a graph clustering/module detection algorithm predict the factions?

Zachary's karate club study





Node grouping based on betweenness

Each node is an individual and edges represent social interactions among individuals.
The shape and colors represent different groups.

Goals for today

- Finding modules on graphs/Community structure on graphs /Graph clustering
- Girvan-Newman algorithm
- Hierarchical agglomerative clustering on graphs
  - Stochastic block model
  - Hierarchical random graph model
  - Combining the two

Issues with existing graph clustering models

- Flat clustering (Stochastic block model)
  - Need to specify the number of clusters
  - Cluster structure might be hierarchical
  - Resolution limit: the presence of large clusters may prevent the recovery of smaller clusters.
- Hierarchical clustering
  - At the top level, only one cluster is found, which might group unrelated clusters
  - At the bottom level, clusters might be too small

Hierarchical Agglomerative Clustering (HAC)

- A hierarchical “agglomerative/bottom-up” clustering algorithm
- Starting from individual graph nodes it gradually finds clusters of increasing size but does not go all the way to the top
- HAC algorithm was developed to overcome issues of existing flat and hierarchical graph clustering strategies
- Main idea: Combines a **flat stochastic block model** and a **hierarchical random graph** model

Park and Bader BMC Bioinformatics 2011

Goals for today

- Finding modules on graphs/Community structure on graphs
- Girvan-Newman algorithm
- Hierarchical agglomerative clustering on graphs
  - Stochastic block model
  - Hierarchical random graph model
  - Combining the two

Stochastic block model (SBM)

- SBM is a probabilistic model for a flat community structure
- Let $G=\{V,E\}$ be a graph, with vertex set V and edge set E
- Suppose each vertex (node) v in V can belong to one of K groups
- Let M denote the edge and non-edge (*hole*) patterns on the graph where membership of v is in one of K groups
- SBM describes $P(M)$
- SBM is defined by parameters of within and between group interactions
  - θ_{ii} Probability of interactions between vertices of the same group i
  - θ_{ij} Probability of interactions between vertices of two different groups i and j

Probability of a grouping structure from the stochastic block model

Probability of edge and hole patterns between group i and j

$$P_{ij} = \theta_{ij}^{e_{ij}} (1 - \theta_{ij})^{h_{ij}}$$

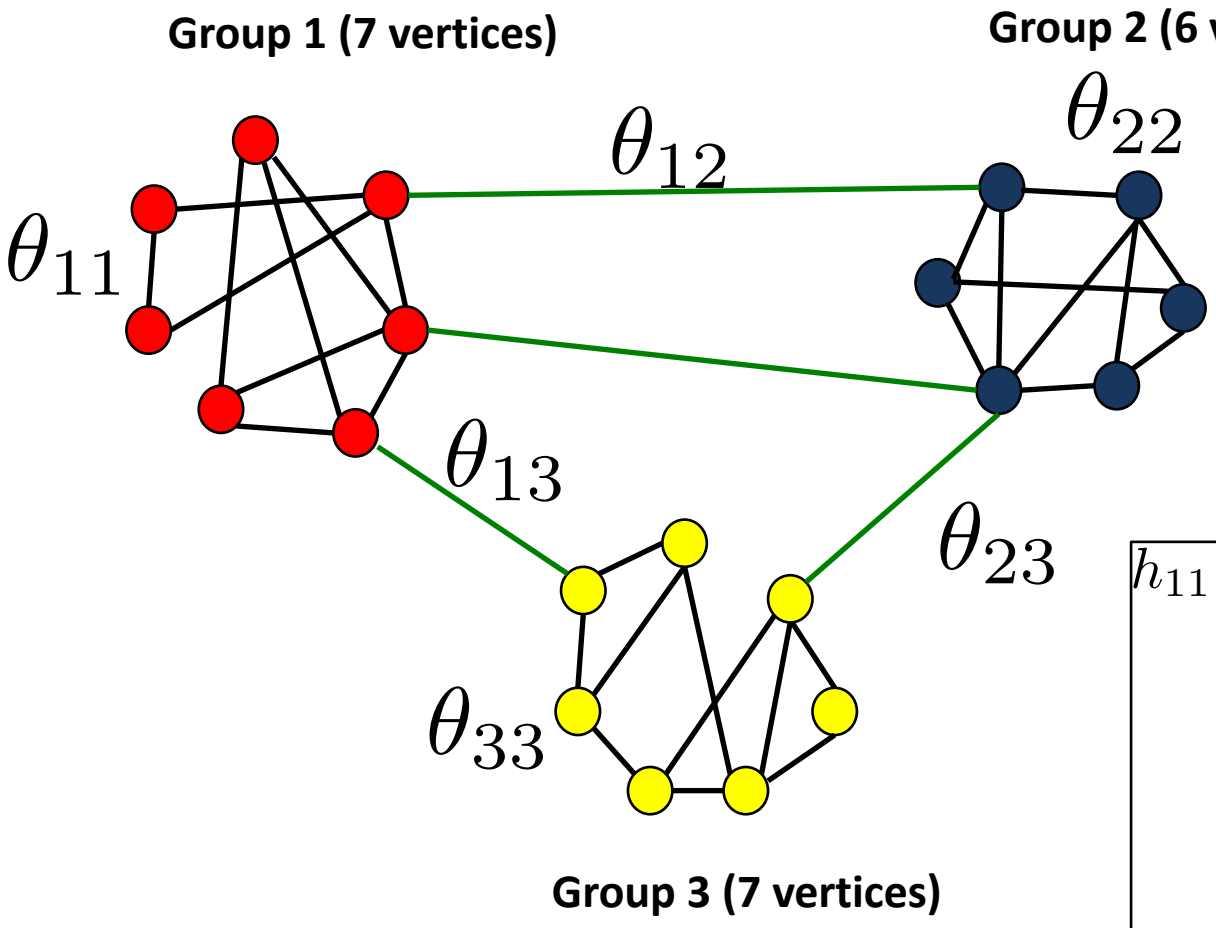
e_{ij}: Number of edges between groups i and j
t_{ij}: Number of possible edges between groups i and j
h_{ij}: Number of holes between groups i and j
$h_{ij} = t_{ij} - e_{ij}$

Probability of edge-hole pattern (*M*) for all groups

$$P(M) = \prod_{i \leq j}^{K} P_{ij} = \prod_{i \leq j}^{K} \theta_{ij}^{e_{ij}} (1 - \theta_{ij})^{h_{ij}}$$

Groups

Stochastic block model for flat clustering for K=3 groups

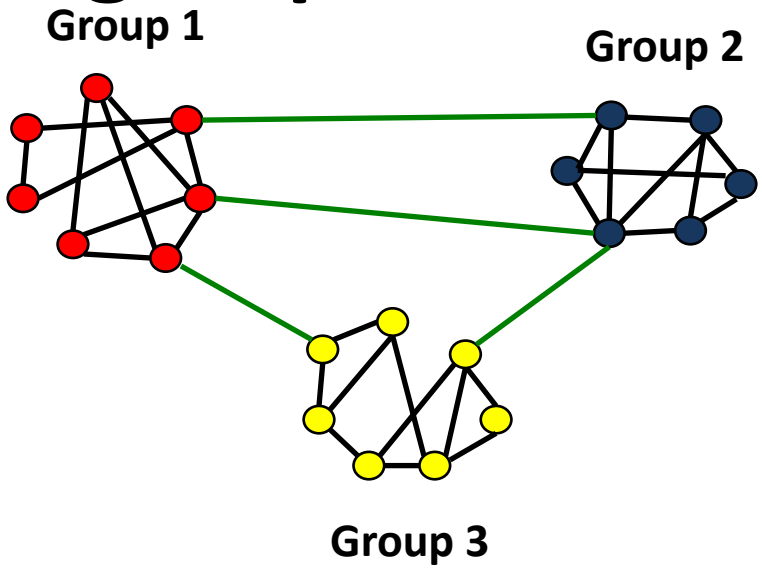


$$
\begin{aligned}
e_{11} &= 10 \\
e_{22} &= 10 \\
e_{33} &= 10 \\
e_{12} &= 2 \\
e_{13} &= 1 \\
e_{23} &= 1
\end{aligned}
$$

$$
\begin{aligned}
h_{11} &= (7*6)/2 - 10 = 11 \\
h_{22} &= 15 - 10 = 5 \\
h_{33} &= 21 - 10 = 11 \\
h_{12} &= 42 - 2 = 40 \\
h_{13} &= 49 - 1 = 48 \\
h_{23} &= 42 - 1 = 41
\end{aligned}
$$

Stochastic block model for flat clustering for K=3 groups

$$e_{11} = 10 \qquad h_{11} = (7*6)/2 - 10 = 11$$
$$e_{22} = 10 \qquad h_{22} = 15 - 10 = 5$$
$$e_{33} = 10 \qquad h_{33} = 21 - 10 = 11$$
$$e_{12} = 2 \qquad h_{12} = 42 - 2 = 40$$
$$e_{13} = 1 \qquad h_{13} = 49 - 1 = 48$$
$$e_{23} = 1 \qquad h_{23} = 42 - 1 = 41$$



$$P(M) = (\theta_{11})^{10}(1-\theta_{11})^{11}(\theta_{12})^{2}(1-\theta_{12})^{40}(\theta_{13})^{1}(1-\theta_{13})^{48}$$

$$\left(\prod_{i \leq j=2}^{3} \theta_{ij}^{e_{ij}} (1-\theta_{ij})^{h_{ij}} \right)$$

Maximum likelihood Parameter estimation in HAC

- Parameters of HAC are θ_{ij}
- Suppose we know what the cluster memberships are
- Maximum likelihood estimate of parameters is $\theta_{ij} = \frac{e_{ij}}{t_{ij}}$
- The probability of edges/holes between groups *i* and *j*

$$P_{ij} = \theta_{ij}^{e_{ij}} (1 - \theta_{ij})^{h_{ij}}$$

- can be re-written as

$$P_{ij} = \left(\frac{e_{ij}}{t_{ij}} \right)^{e_{ij}} \left(\frac{h_{ij}}{t_{ij}} \right)^{h_{ij}}$$

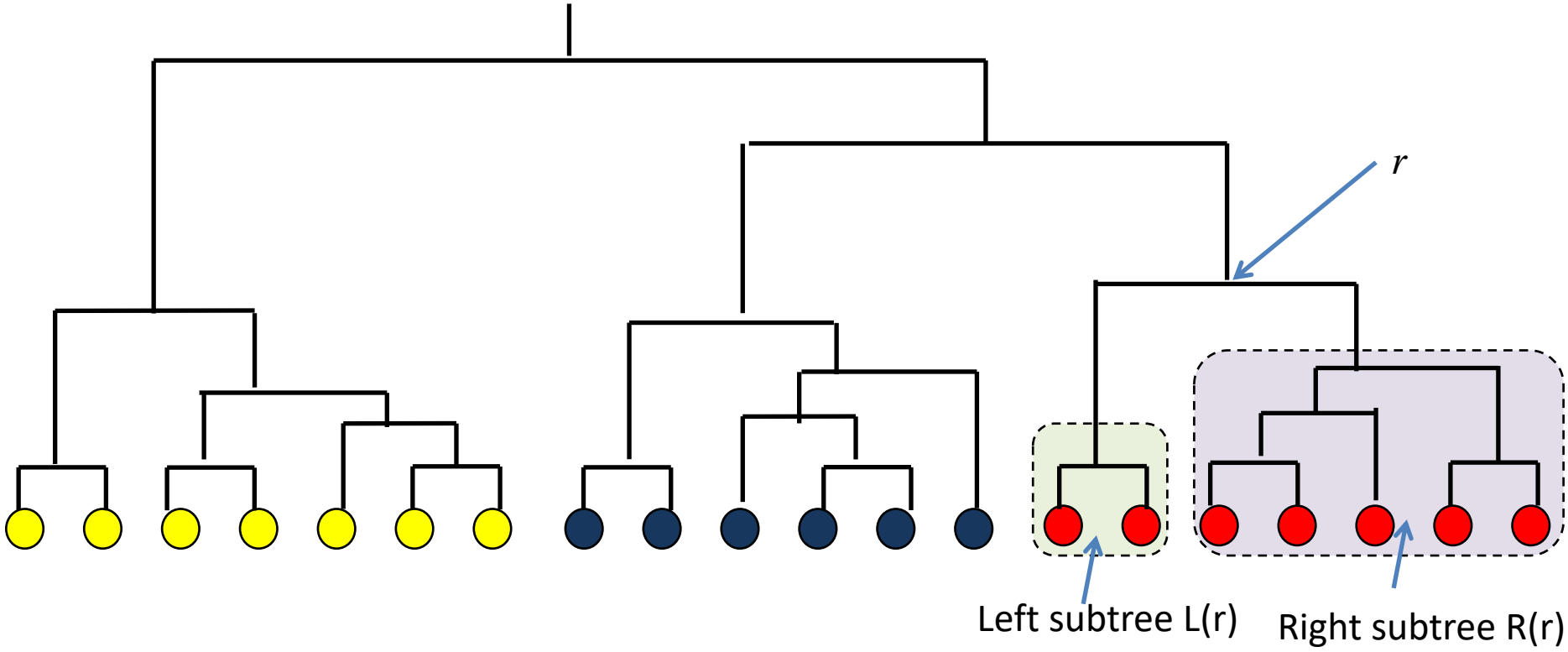
$$P_{ij} = \frac{e_{ij}^{e_{ij}} h_{ij}^{h_{ij}}}{t_{ij}^{t_{ij}}}$$

Hence, everything can be expressed in terms of the edge presence and absence pattern

Goals for today

- Finding modules on graphs/Community structure on graphs
- Girvan-Newman algorithm
- Hierarchical agglomerative clustering on graphs
  - Stochastic block model
  - Hierarchical random graph model
  - Combining the two

Hierarchical Random Graph model



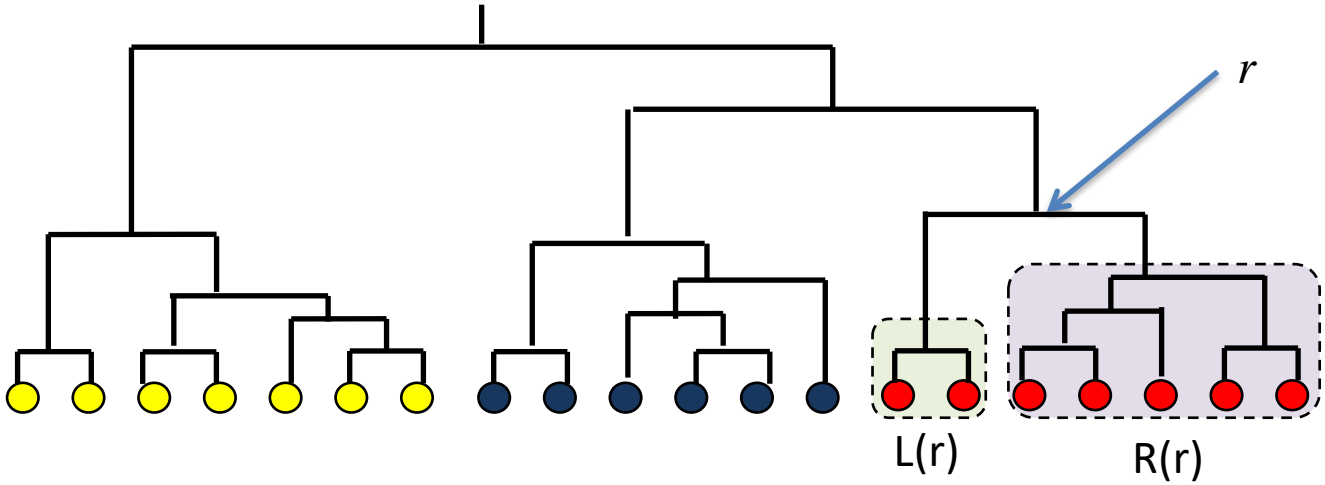
Probability of edges (e_r) and holes (h_r) between nodes in

$$P_r = \theta_r^{e_r}(1-\theta_r)^{h_r}$$

e_r: Number of edges crossing L(r) and R(r)

h_r: Number of holes between L(r) and R(r)

Hierarchical Random Graph model



The HRG also describes a probabilistic model over the entire hierarchy

$$P(M) = \prod_{r} P_r = \prod_{r} \theta_r^{e_r} (1 - \theta_r)^{h_r}$$

All internal nodes

What if we did not want to merge up to the very top?

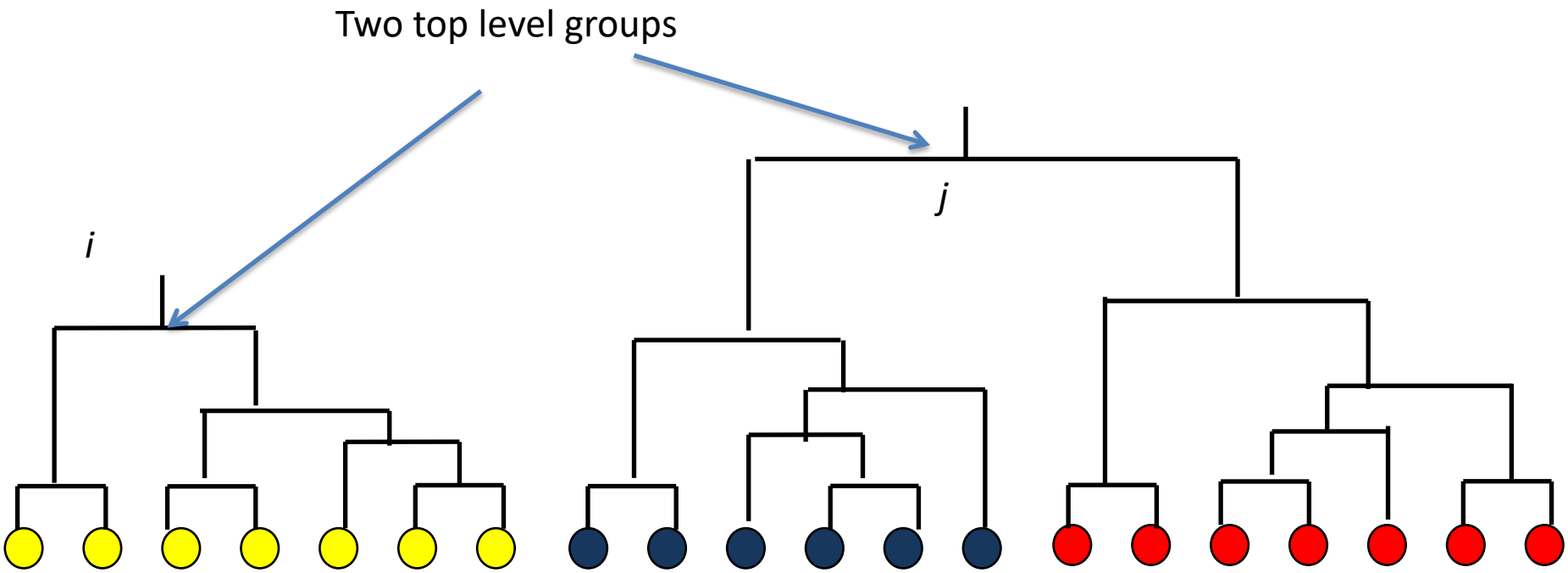
- This is what the HAC model tries to do

$$P(M) = \prod_{rr' \in \text{top}} P_{rr'} \prod_{r} P_r$$

Defined in the same way as the flat model for two groups, *r* and *r'*

Defined in the same way as the hierarchical model for tree node *r*

Combining the HRG and the SB model



$$P(M) = P_{ij} \prod_{r_i} P_{r_i} \prod_{r_j} P_{r_j}$$

Agglomerative clustering of HAC

- The idea of the HAC algorithm is to use agglomerative clustering:
  - build the hierarchy bottom up, but stop if a merge of trees is not good enough
- We consider each pair of clusters and merge them if the corresponding model score improves
- What is a good enough merge?

Define the score of a merge

- Assume there are K top level clusters
- If two nodes 1 and 2 are merged into 1', the change in likelihood before and after the merge is scored as:

$$\lambda_{12} = \frac{P(M')}{P(M)} = \prod_{k=3}^{K} \frac{P_{1'k}}{P_{1k} P_{2k}}$$

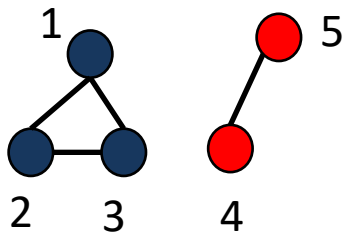
- This score enables us to develop a greedy agglomerative algorithm

HAC greedy algorithm

- Input: Graph $G=(V,E)$
- Initialize top-level clusters top={{v}, *in* V}
- Initialize $\mathrm{K} = |\mathrm{V}|$
- **While** $\mathrm{K}>1$ **do**
  - Find top-level clusters i and j with max λ_{ij}
  - Merge i and j into new top level cluster r
  - Add r to top
  - $\mathrm{L}(r) = \{i\}$ and R(r)={j}
  - Remove i and j from top
  - K=K-1

HAC algorithm execution

- Let's see one iteration of HAC with a small example graph with 5 nodes V={1,2,3,4,5} and following edges



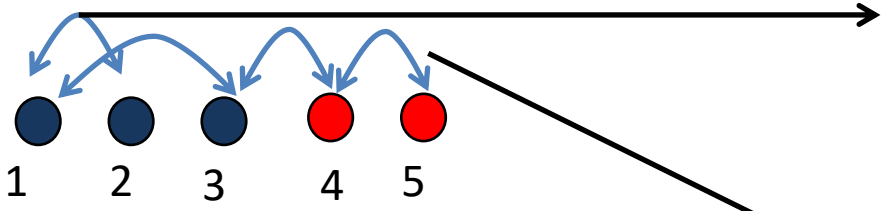
Example graph

Initialization

- *top*={1,2,3,4,5}
- K=5

HAC algorithm iteration 1

- In iteration one

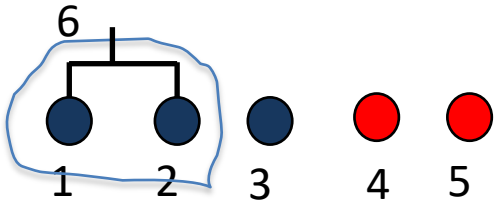


- Compute scores for all potential merges, into a new node 6.

$$\lambda_{12} = \frac{\prod_{k=3}^{5} P_{6k}}{\prod_{k=3}^{5} P_{1k} P_{2k}}$$

$$\vdots$$

$$\lambda_{45} = \frac{\prod_{k=1}^{3} P_{6k}}{\prod_{k=1}^{3} P_{4k} P_{5k}}$$

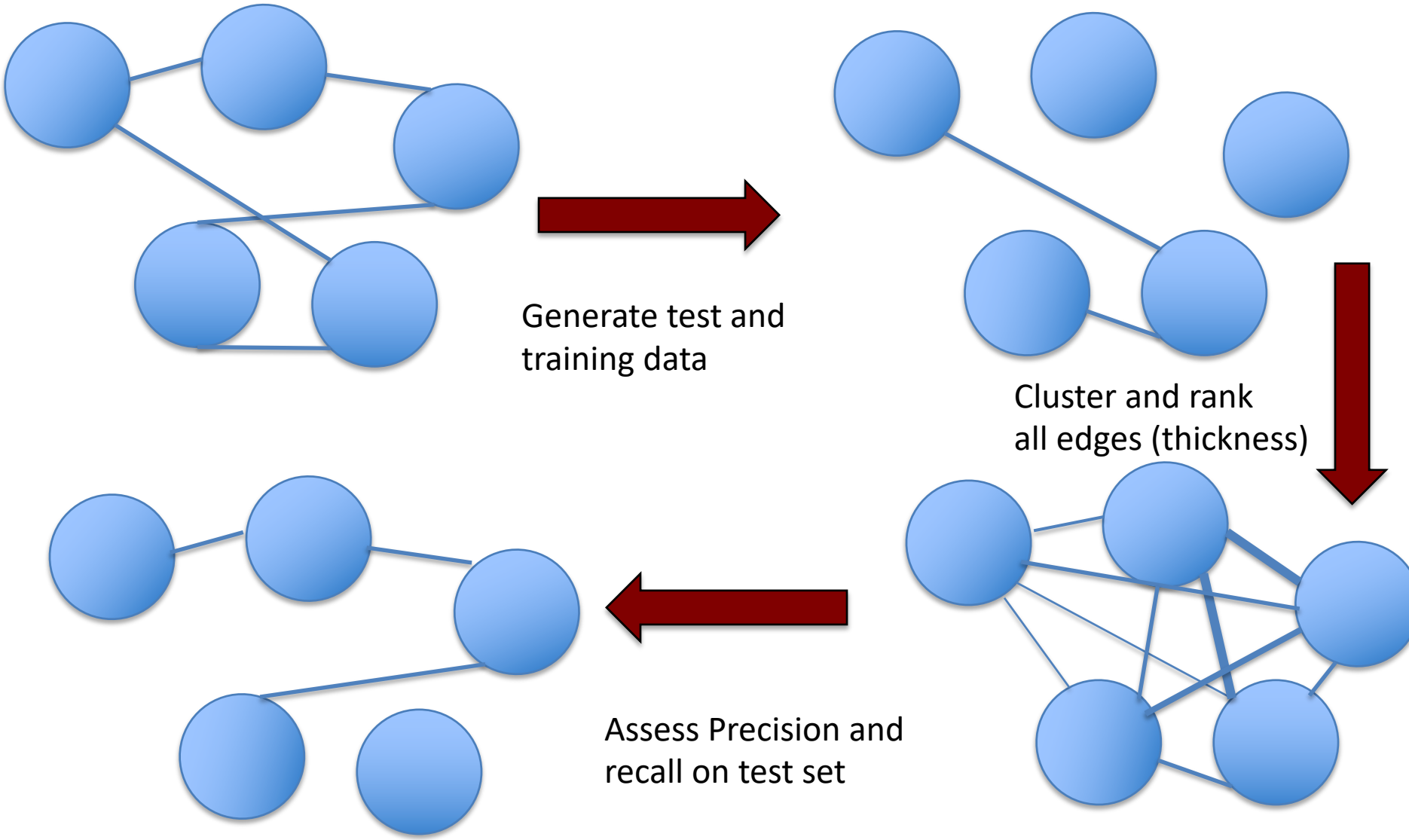


- Merge nodes with the highest score (example shows 1,2)
- Update hierarchy L(6)=1, R(6)=2
- Update top=$\{3, 4, 5, 6\}$
- K=4
- Repeat with new top

Experiments

- Compare HAC with other algorithms
  - Fast Modularity (CNM)
  - Variational Bayes Modularity (VBM)
  - Graph Diffusion Kernel (GDK)
- Assess performance based on link prediction
  - Randomly remove a set of test edges
  - Estimate within and between cluster interaction probabilities using the remaining edges
  - Generate confidence scores for all pairs of nodes within and between clusters using cluster structure from training data
  - Rank links based on confidence score and compute AUPR and F-score
    - Note, this assumes that the test nodes do not disappear from the training set.

Assessing performance based on link prediction



HAC-ML is able to outperform other algorithms

Link prediction performance of 85/15 cross validation (7.5% of observed edges held out).

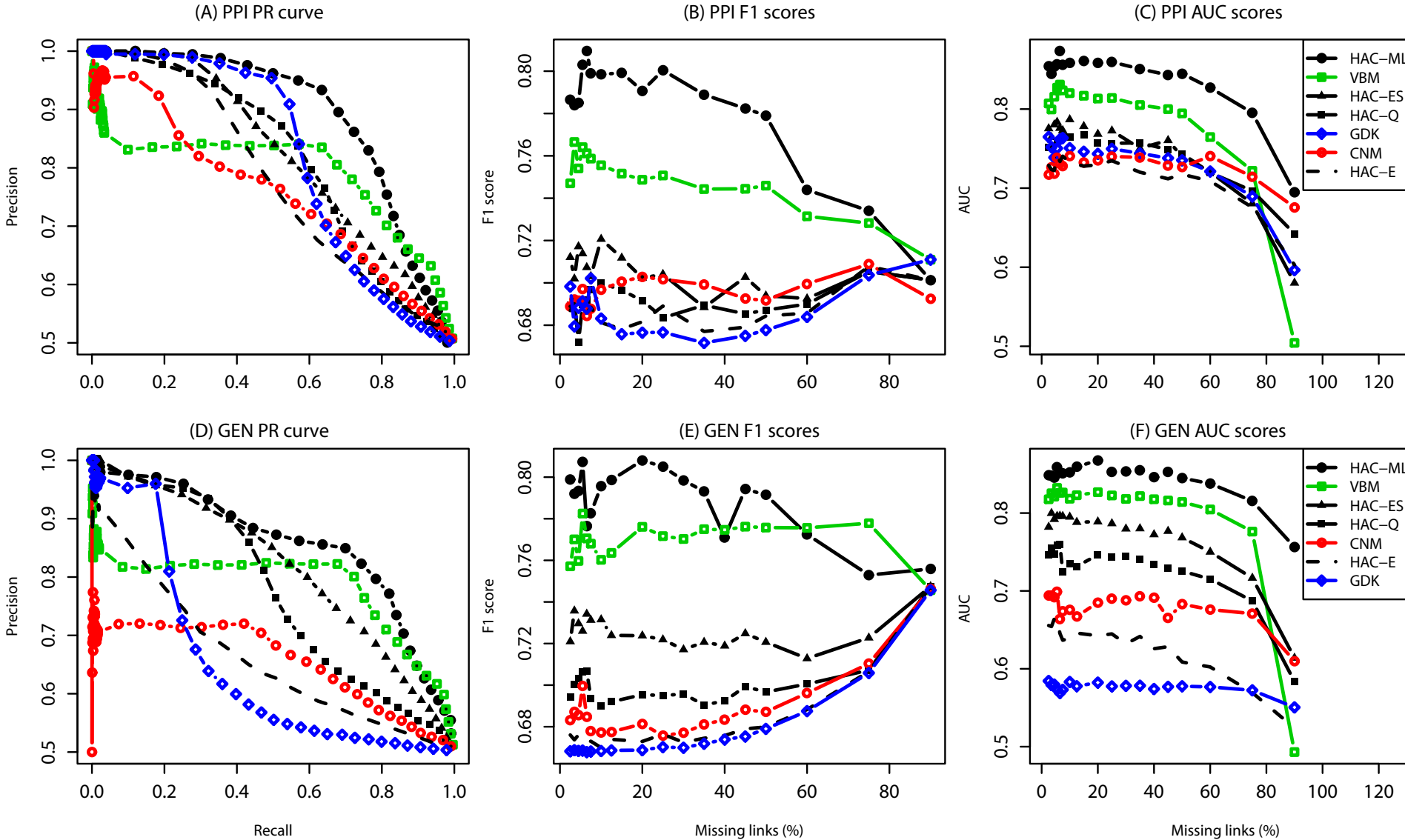
Physical interactions

Data	HAC-ML	GDK	CNM	VBM	HAC-ES	HAC-E	HAC-Q
Yeast-PPI	**0.79**±0.5	0.69±0.3	0.69±0.7	0.76±0.4	0.71±0.5	0.69±0.7	0.69±0.8
Drosophila	**0.73**±0.8	0.66±0.2	0.67±0.4	0.70±0.4	0.67±0.3	0.67±0.3	0.67±0.4
Human	0.73±0.9	**0.75**±0.7	0.71±0.5	0.70±0.6	0.67±0.4	0.68±0.5	0.69±1.0
Celegans	**0.68**±1.5	0.67±1.3	**0.68**±1.3	0.66±0.6	0.66±0.8	0.66±0.7	0.67±0.8
Arabidopsis	0.80±8.3	**0.92**±2.2	**0.92**±3.2	0.90±3.6	0.78±11.0	0.87±10.8	0.88±11.4

Genetic interactions

Data	HAC-ML	GDK	CNM	VBM	HAC-ES	HAC-E	HAC-Q
Yeast-GEN	**0.78**±2.3	0.67±0.0	0.69±0.7	0.74±6.0	0.73±0.8	0.67±0.1	0.69±0.7
SGA	**0.76**±1.5	0.67±0.0	0.67±0.2	**0.76**±0.3	0.70±0.2	0.67±0.0	0.69±0.2
SLAM	**0.92**±1.0	0.91±0.5	0.68±0.8	0.67±0.3	0.84±2.9	0.76±1.0	0.67±0.3

HAC-ML is able to outperform competing methods



Multi-resolution analysis of yeast PPI network

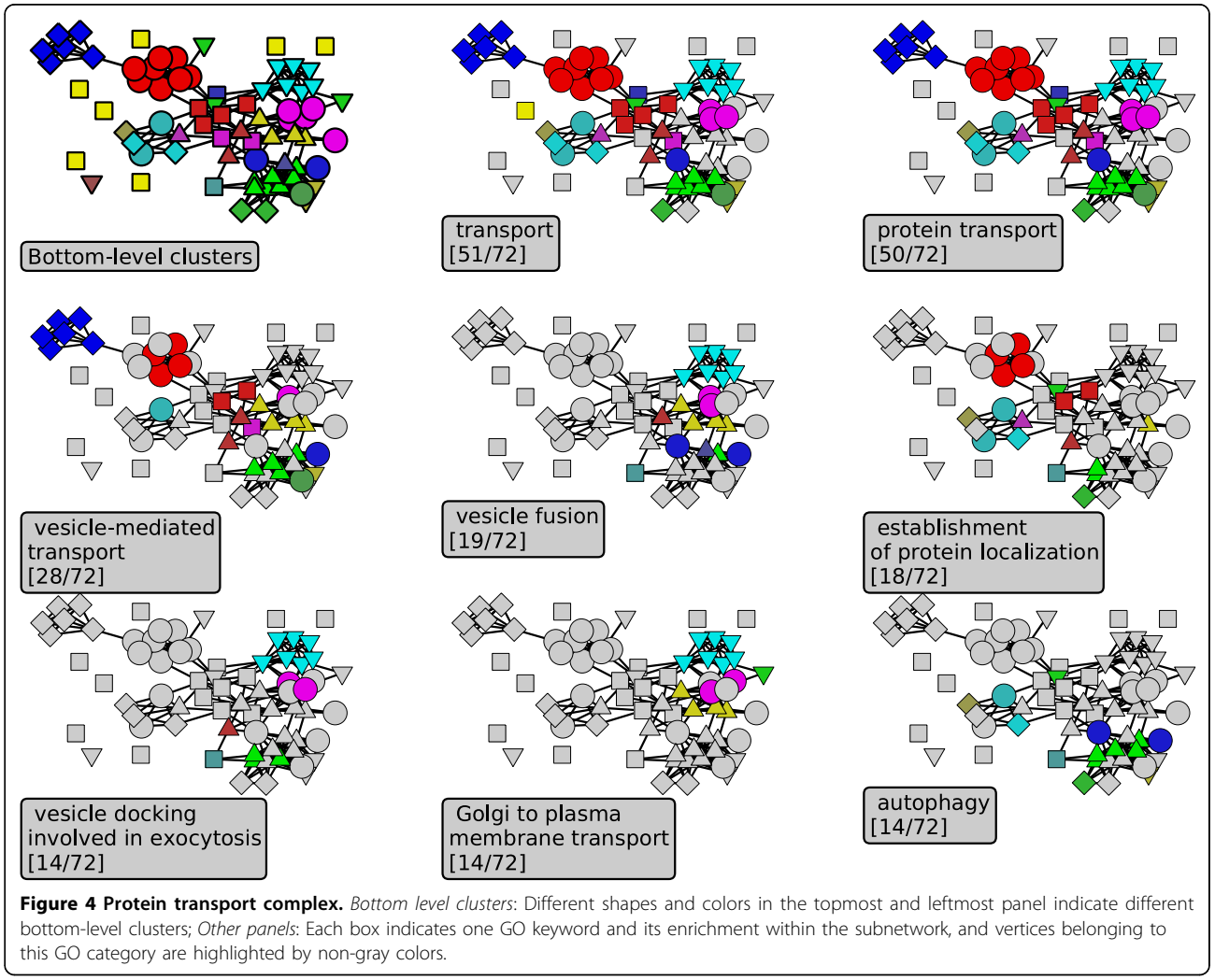


Figure 4 Protein transport complex. *Bottom level clusters*: Different shapes and colors in the topmost and leftmost panel indicate different bottom-level clusters; *Other panels*: Each box indicates one GO keyword and its enrichment within the subnetwork, and vertices belonging to this GO category are highlighted by non-gray colors.

Bottom level clusters could provide greater resolution to the GO terms

Take away points

- Biological networks are modular
- Modules can be topological or functional
- We have seen two clustering algorithms
  - Girvan-Newman algorithm
    - based on edge-betweenness
    - Can be viewed as top-down/divisive clustering algorithm
  - Hierarchical Agglomerative clustering
    - Combines SBM and HRG
    - Enables one to find clusters and also their relationships
    - Greedy probabilistic score for merging subtrees is fast
    - Good performance on link prediction