Hierarchical Structure and Predicting Missing Links

Cristopher Moore
University of New Mexico
and the Santa Fe Institute

joint work with
Aaron Clauset (UNM/SFI)
and Mark Newman (Michigan)

Three Goals

#1: Inferring network structure from observed data

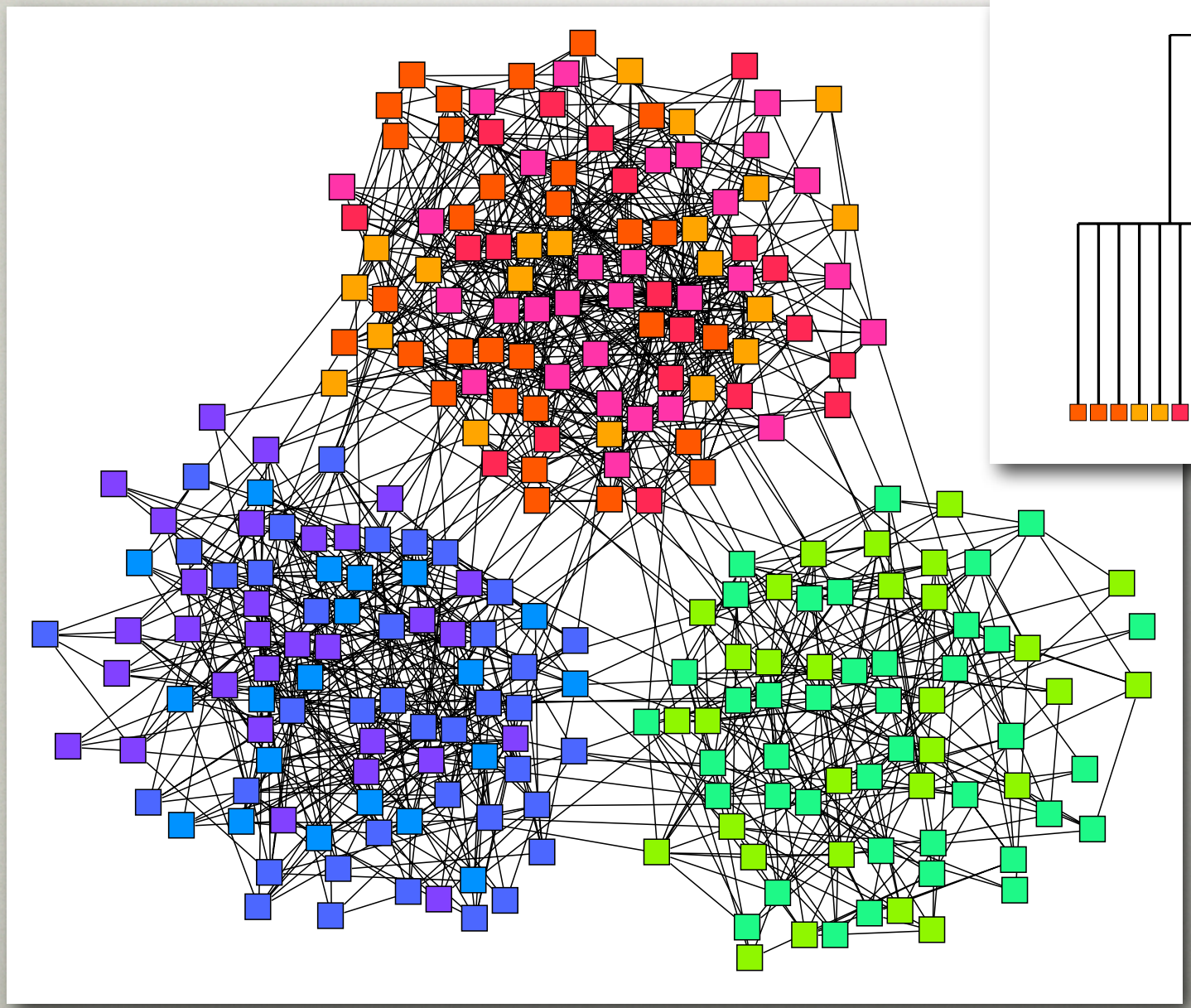
#2: Generating random graphs which are statistically similar to real ones

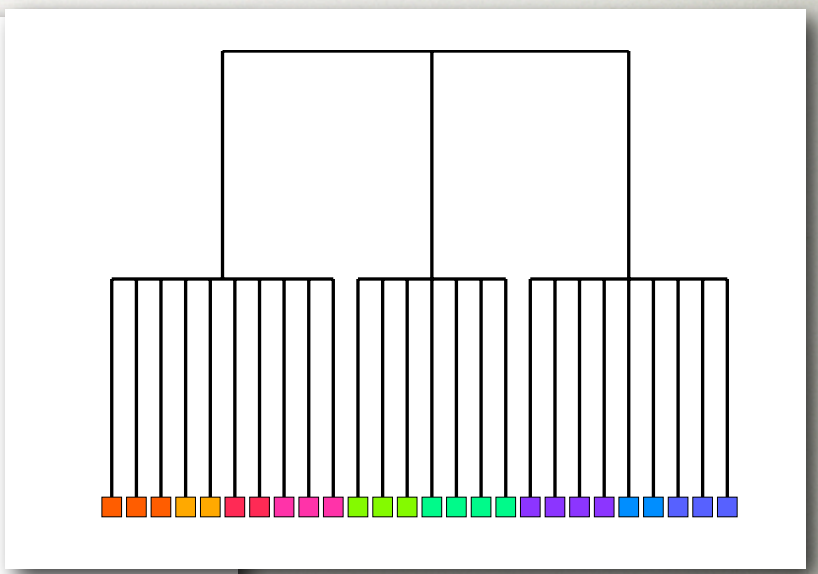
#3: Predicting missing links

Two Extremes

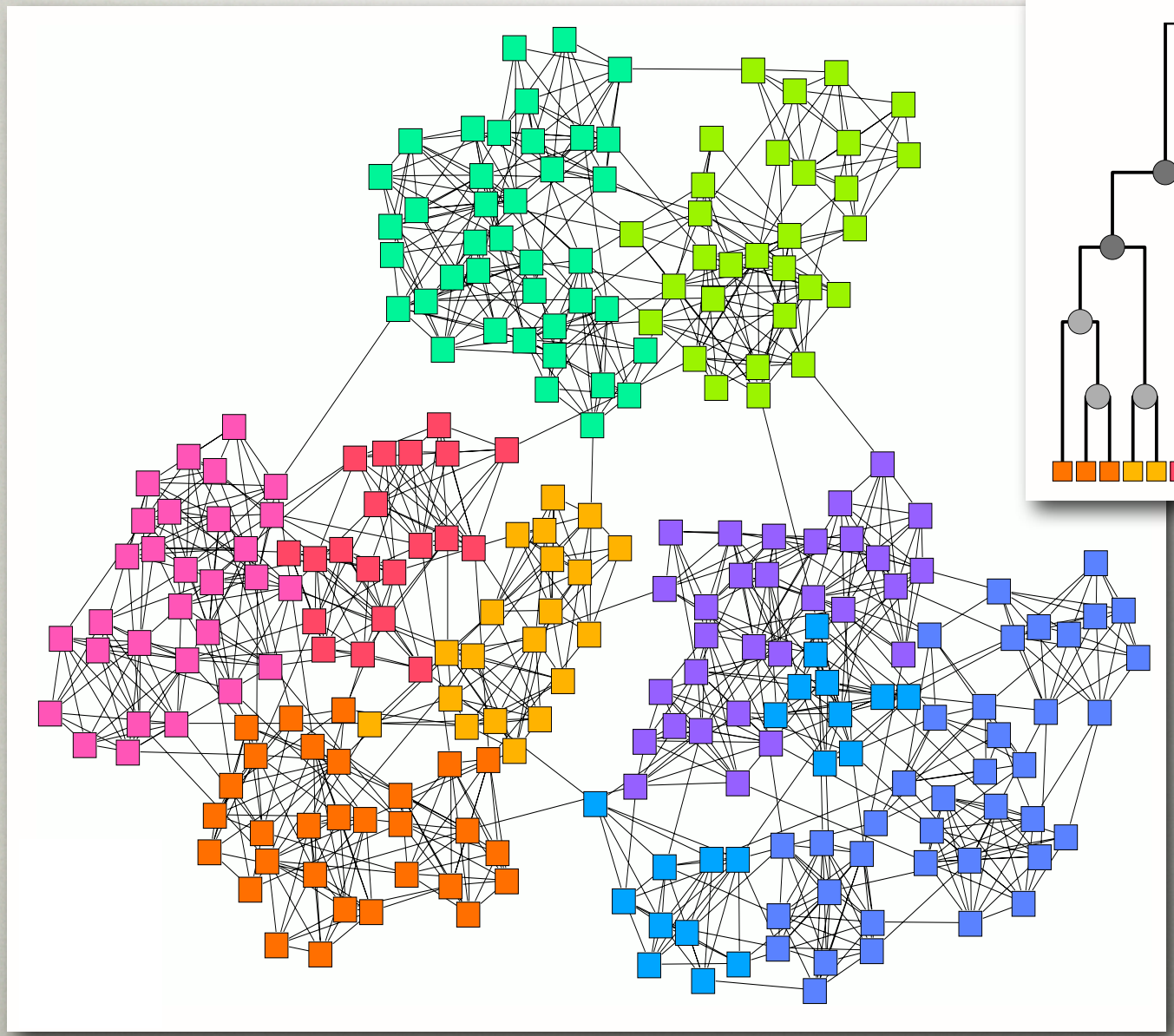
- Proving theorems about simple models
  - *Captures some aspect of network structure,*
  - *But leaves lots of other structure out.*
- Observing patterns in real networks
  - *Good "natural history"*
  - *But often ad-hoc.*
- One possible bridge: Machine Learning.

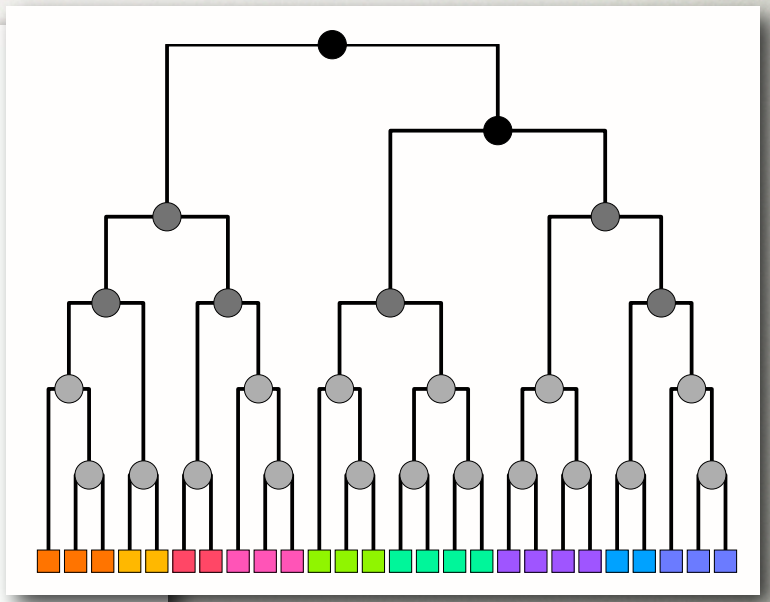
Clustering: one level





Hierarchy: many levels

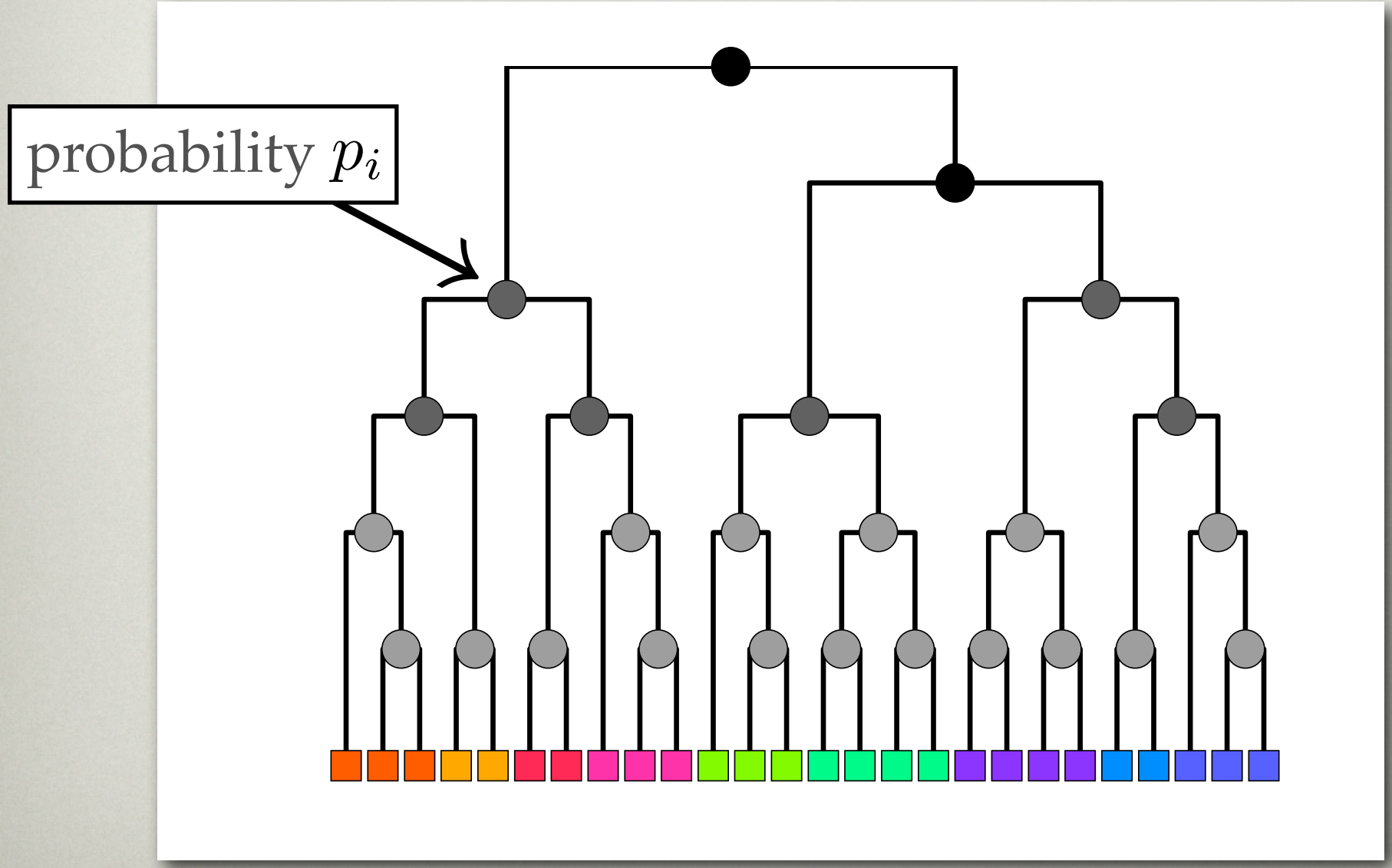




A Probabilistic Model (with lots of parameters!)

- A binary tree T: leaves are original vertices, internal nodes represent communities
- Each internal node has a probability p_i
- Two vertices are connected with probability p_i where i is their lowest common ancestor
- Allows *assortative* or *disassortative* structure, or any mix across scales and subtrees

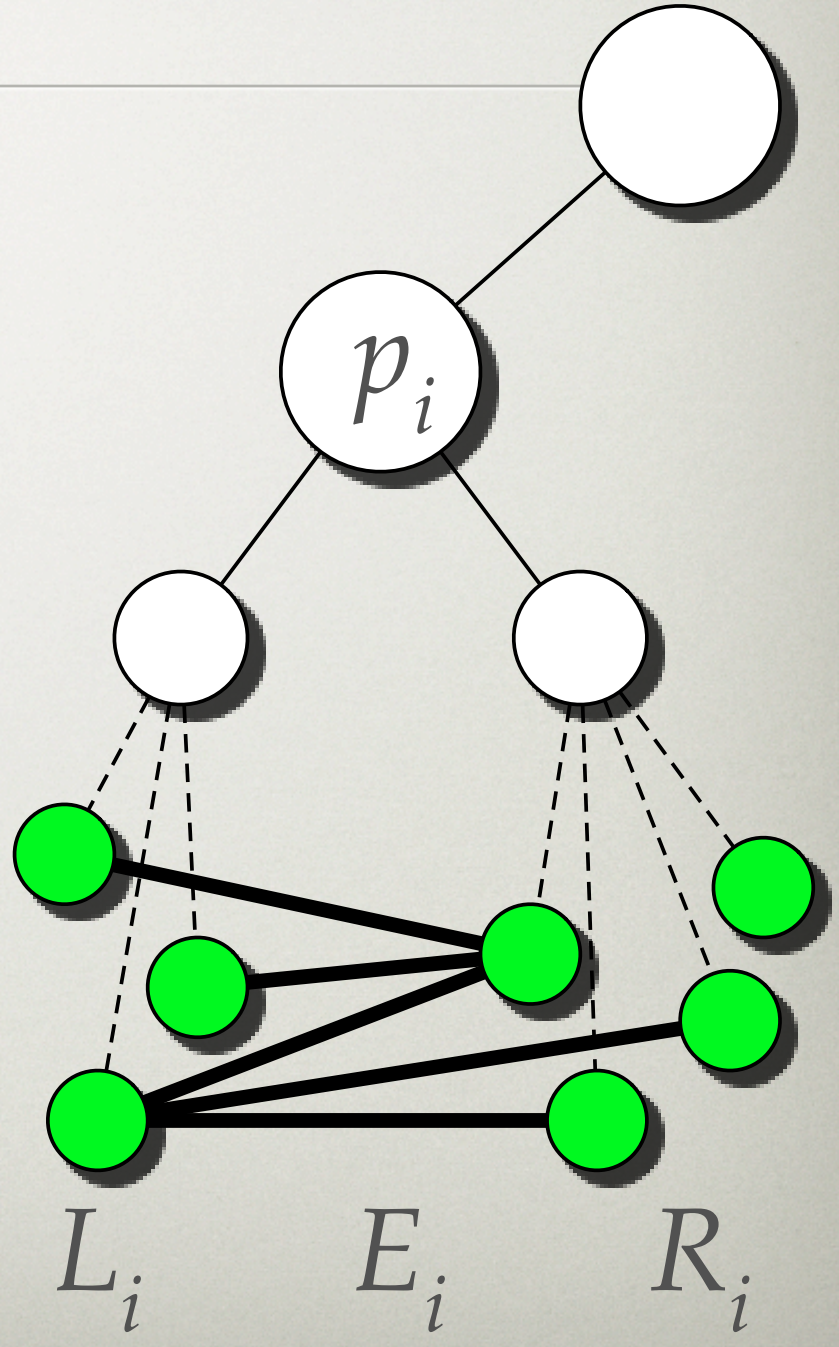
A Probabilistic Model (with lots of parameters!)



Maximum Likelihood

- For each internal node i,
  - L_i and R_i = # descendants
  - E_i = # edges between them
- Likelihood these edges exist, and not others, is

$$\mathcal{L}_i = p_i^{E_i}(1-p_i)^{L_i R_i - E_i}$$



Maximum Likelihood

- Each $\mathcal{L}_i$ is maximized by $p_i = E_i / L_i R_i$
- The likelihood of the entire tree is then

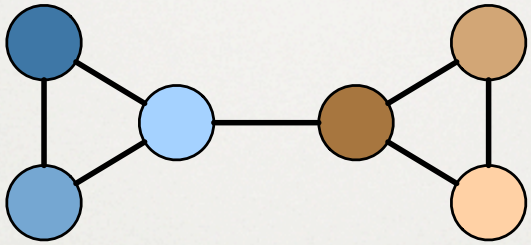
$$\mathcal{L}(T) = \prod_i \mathcal{L}_i$$

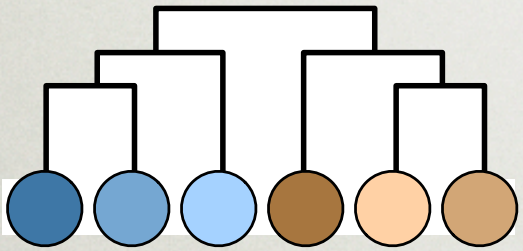
- The *log-likelihood* is

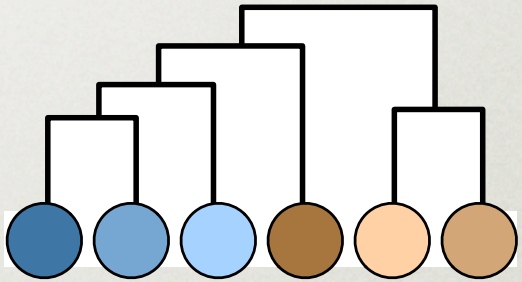
$$\ln \mathcal{L}(T) = -\sum_i L_i \, R_i \, h(E_i / L_i R_i)$$

where $h(p) = -p \ln p - (1-p) \ln(1-p)$

Maximum Likelihood







$$\mathcal{L} = \left(\frac{1}{9}\right)\left(\frac{8}{9}\right)^{8}$$

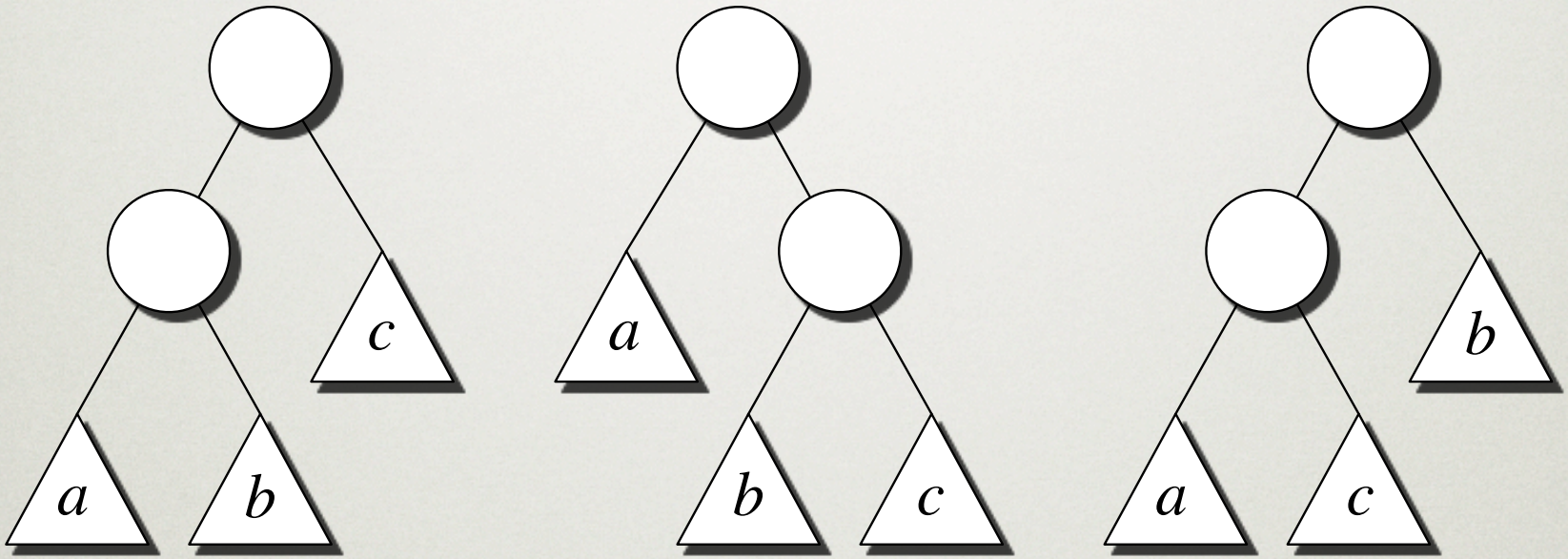
$$= 0.0433$$

$$\mathcal{L} = \left[\left(\frac{1}{3}\right)\left(\frac{2}{3}\right)^{2}\right] \cdot \left[\left(\frac{2}{8}\right)^{2}\left(\frac{6}{8}\right)^{6}\right]$$

$$= 0.0016$$

A Markov Chain

- We update the tree T with rotations:

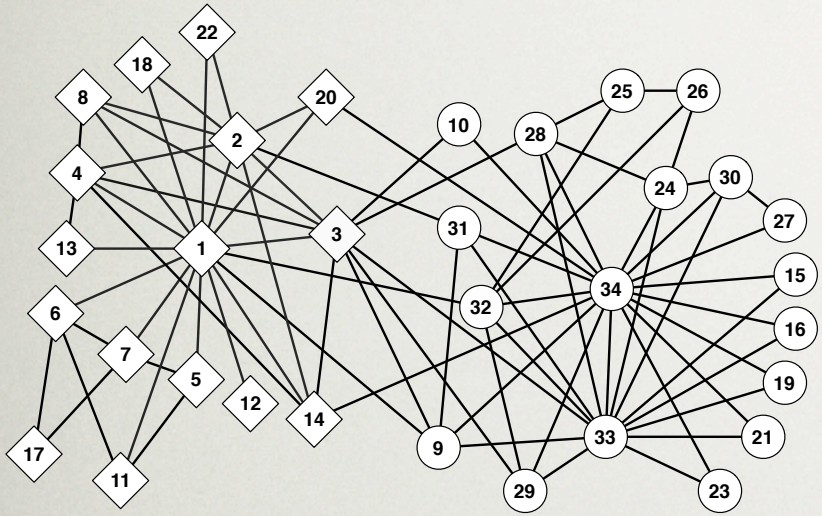


- We move with probability 1 if $\Delta \ln \mathcal{L} \geq 0$ and probability $\exp(\Delta \ln \mathcal{L})$ otherwise

Some Nice Properties

- Easy to calculate $\Delta \ln \mathcal{L}$: just "local" terms
- Moves entire chunks of the tree, while keeping their internal structure the same
- Allows us to sample trees with probability proportional to $\mathcal{L}$, instead of just the one with max likelihood (helps avoid overfitting)

Let's Try it Out

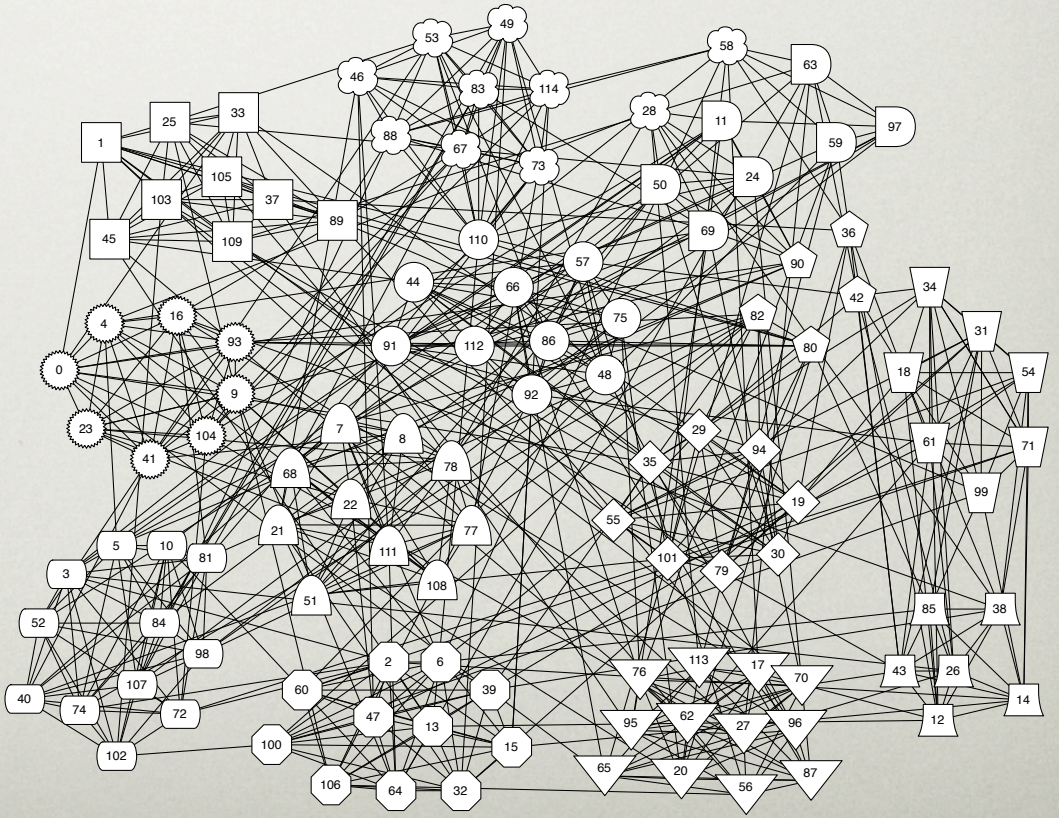


Zachary's Karate Club

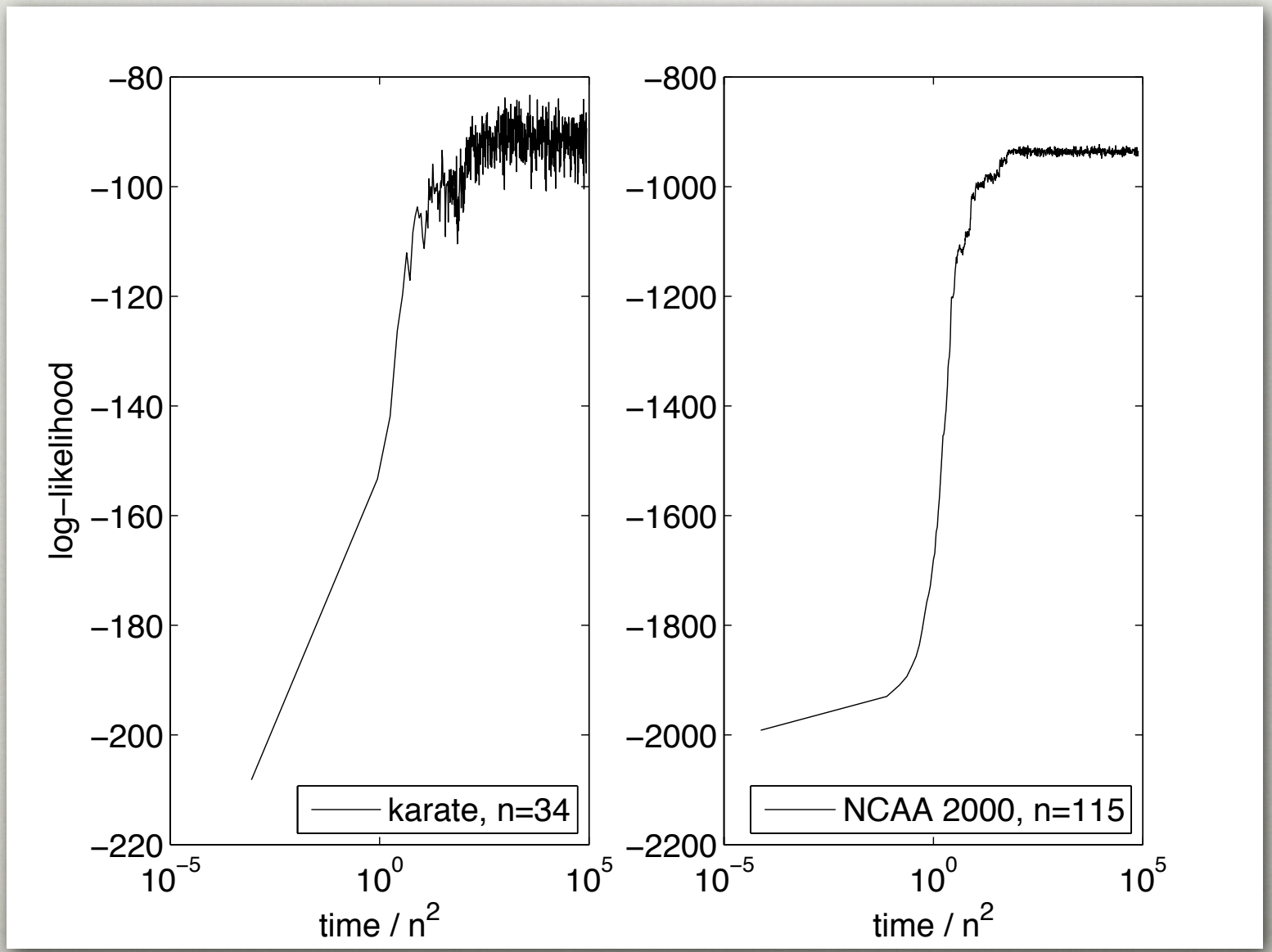
$n = 34 \quad m = 78$

NCAA Schedule 2000

$n = 115 \quad m = 613$



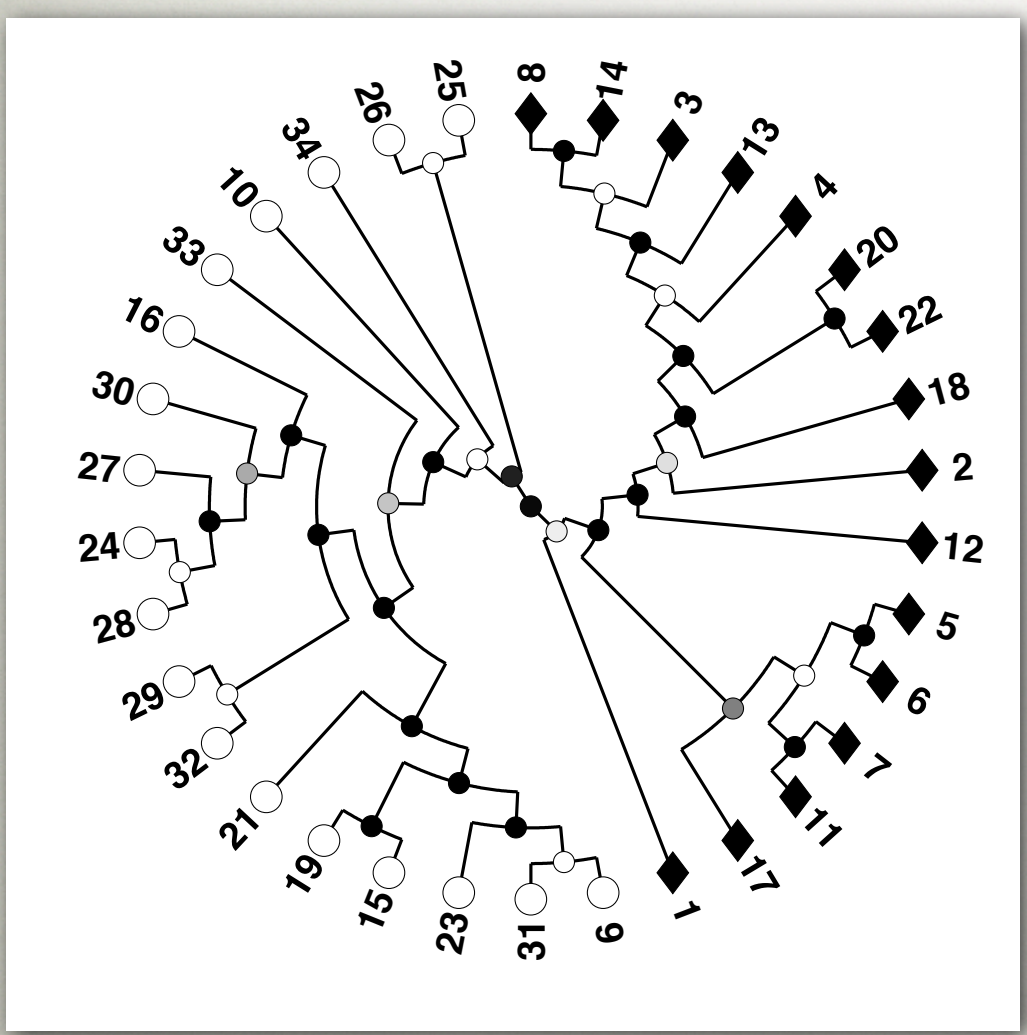
Experiments



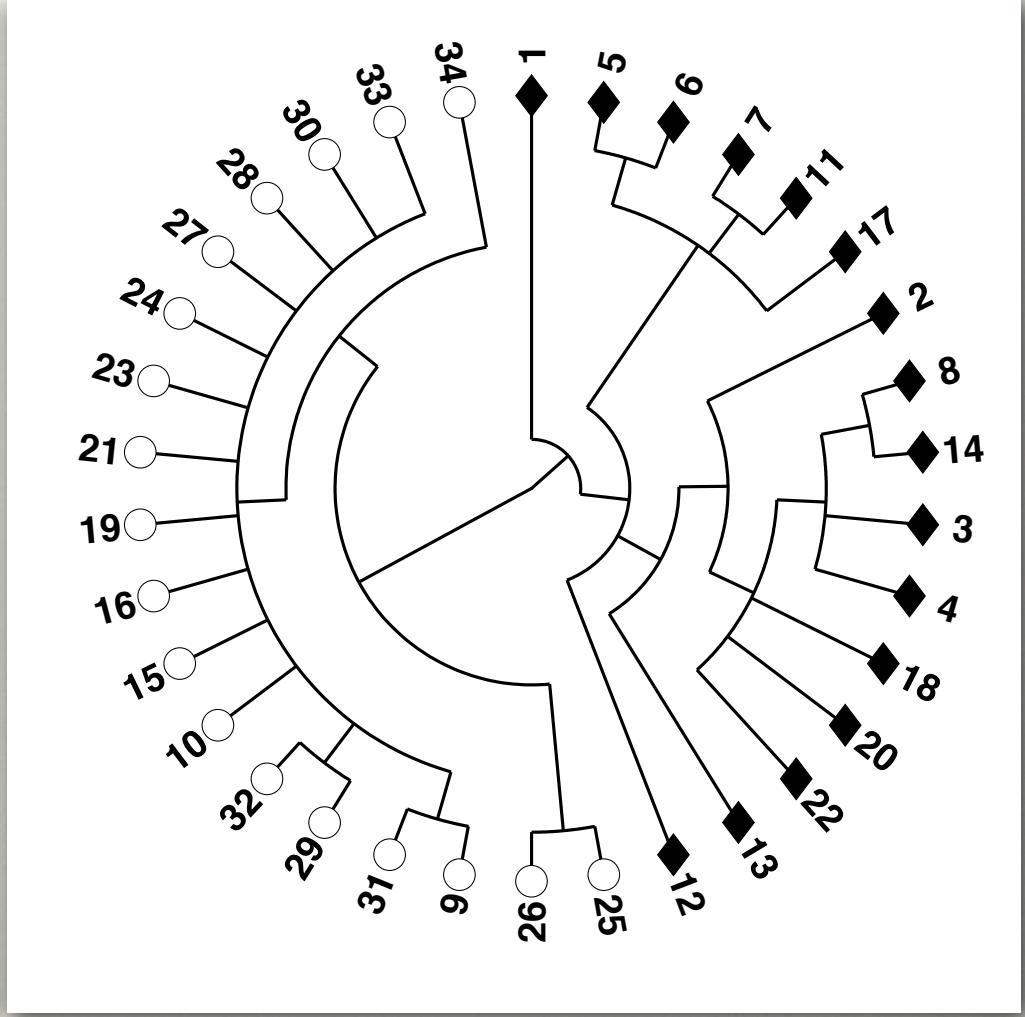
Consensus Dendrograms

- Let's sample many trees instead of one.
- From phylogeny construction: combine these into a *consensus* hierarchy, which includes the (weighted) majority of splits
- More appropriate than any single tree (even the most likely one)

Consensus Dendrogram: Karate Club



point estimate

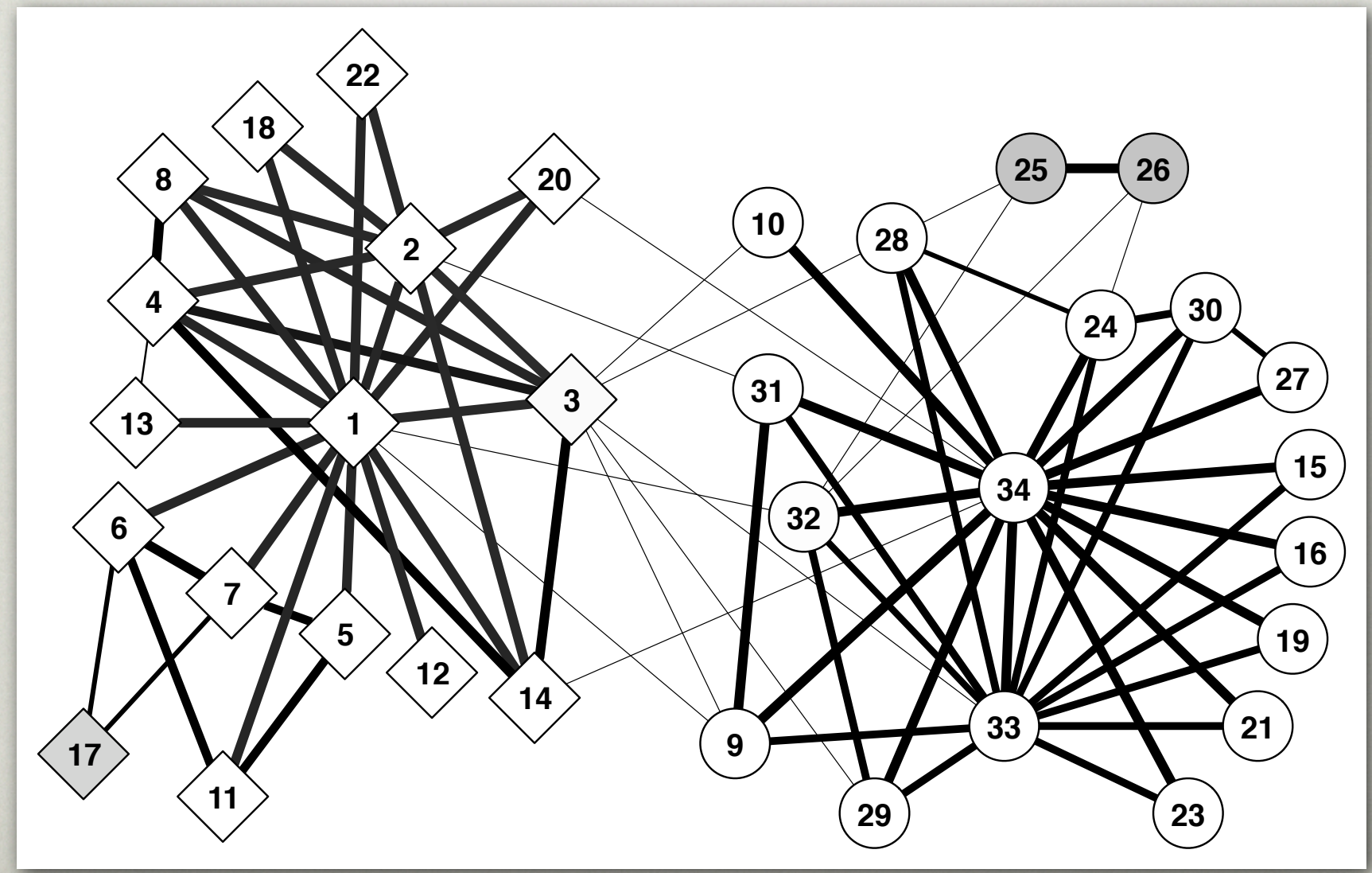


consensus hierarchy

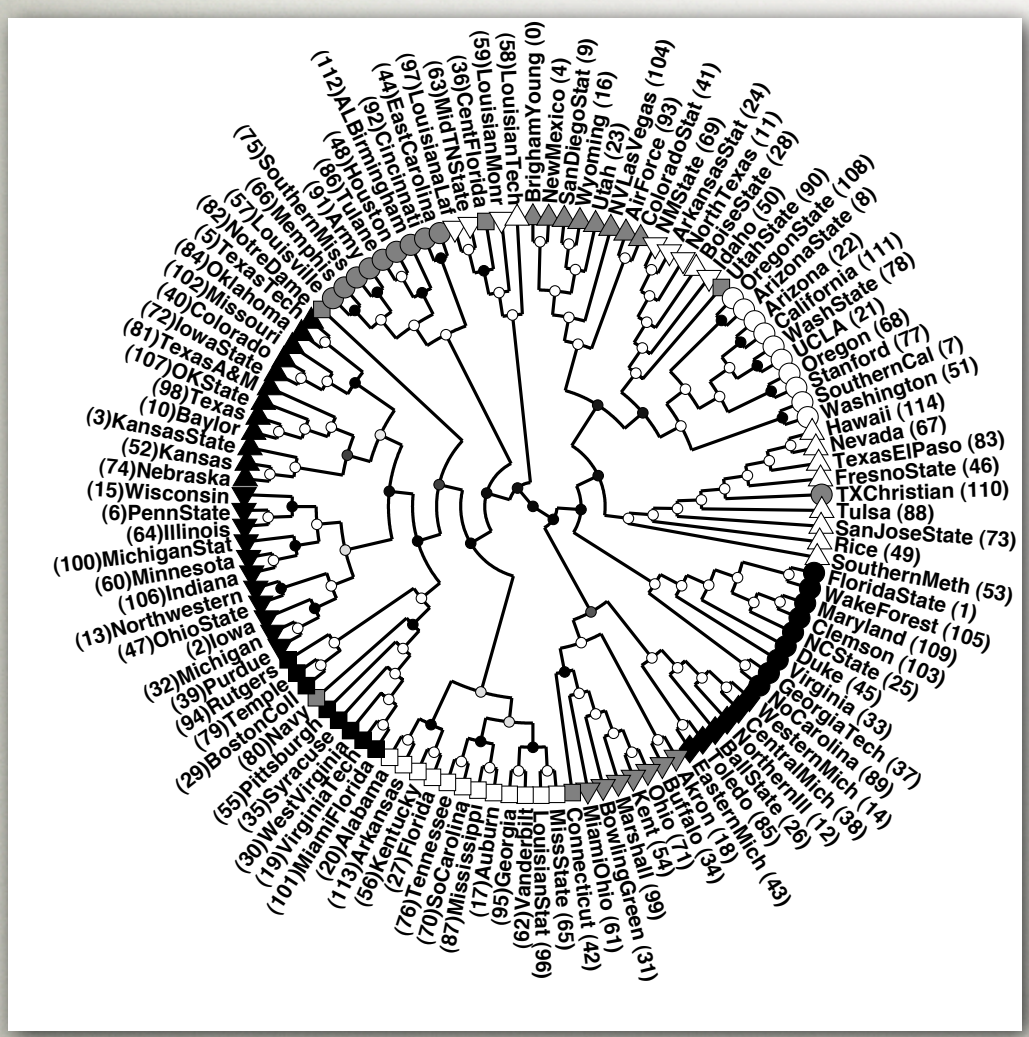
Sampling the Distribution

- Again, sample many trees instead of one.
- Calculate the (weighted) average probability that two vertices are connected, or that a vertex is part of a given community.
- Lets us classify how strongly a vertex is part of a group, or how “surprising” an edge is.

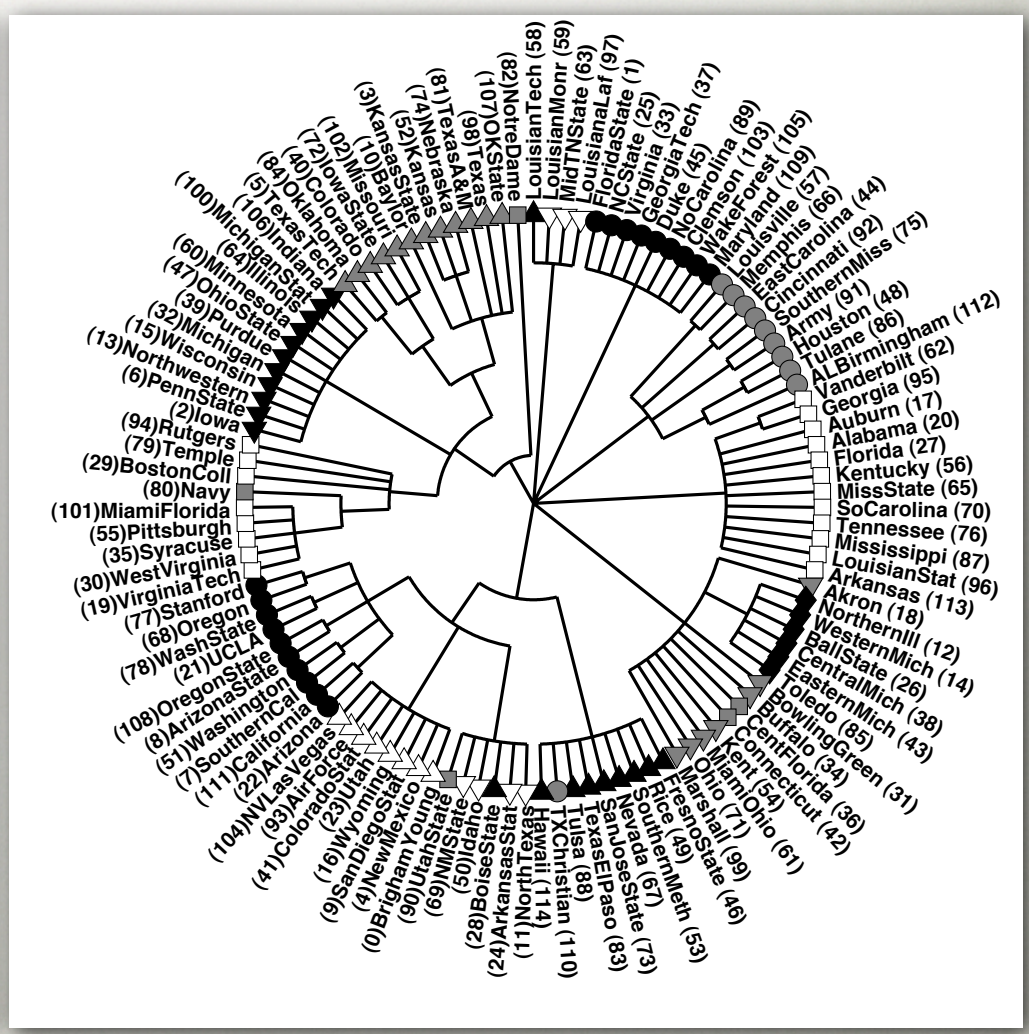
Average Probability of Edges and Vertices



Consensus Dendrogram: NCAA

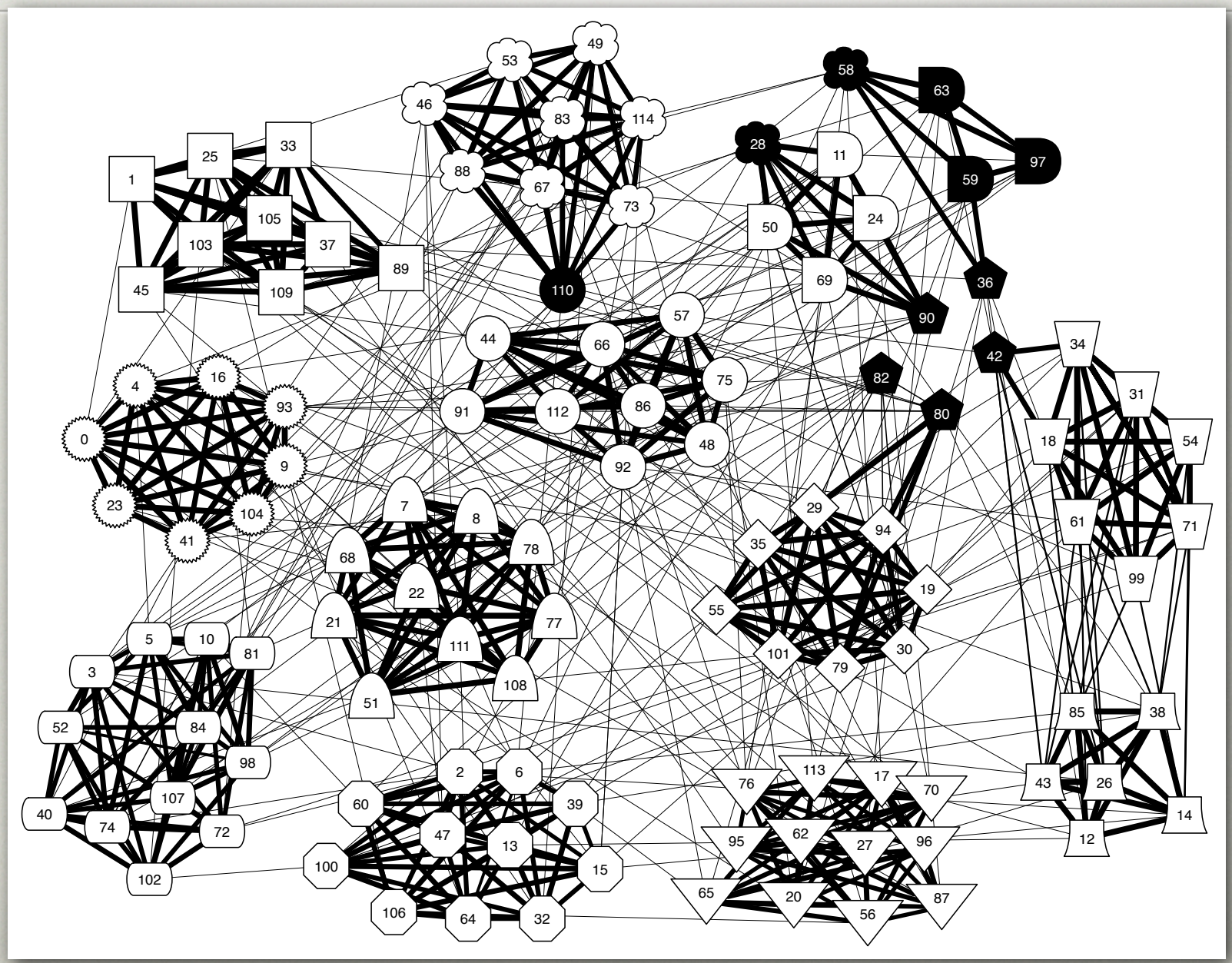


point estimate



consensus hierarchy

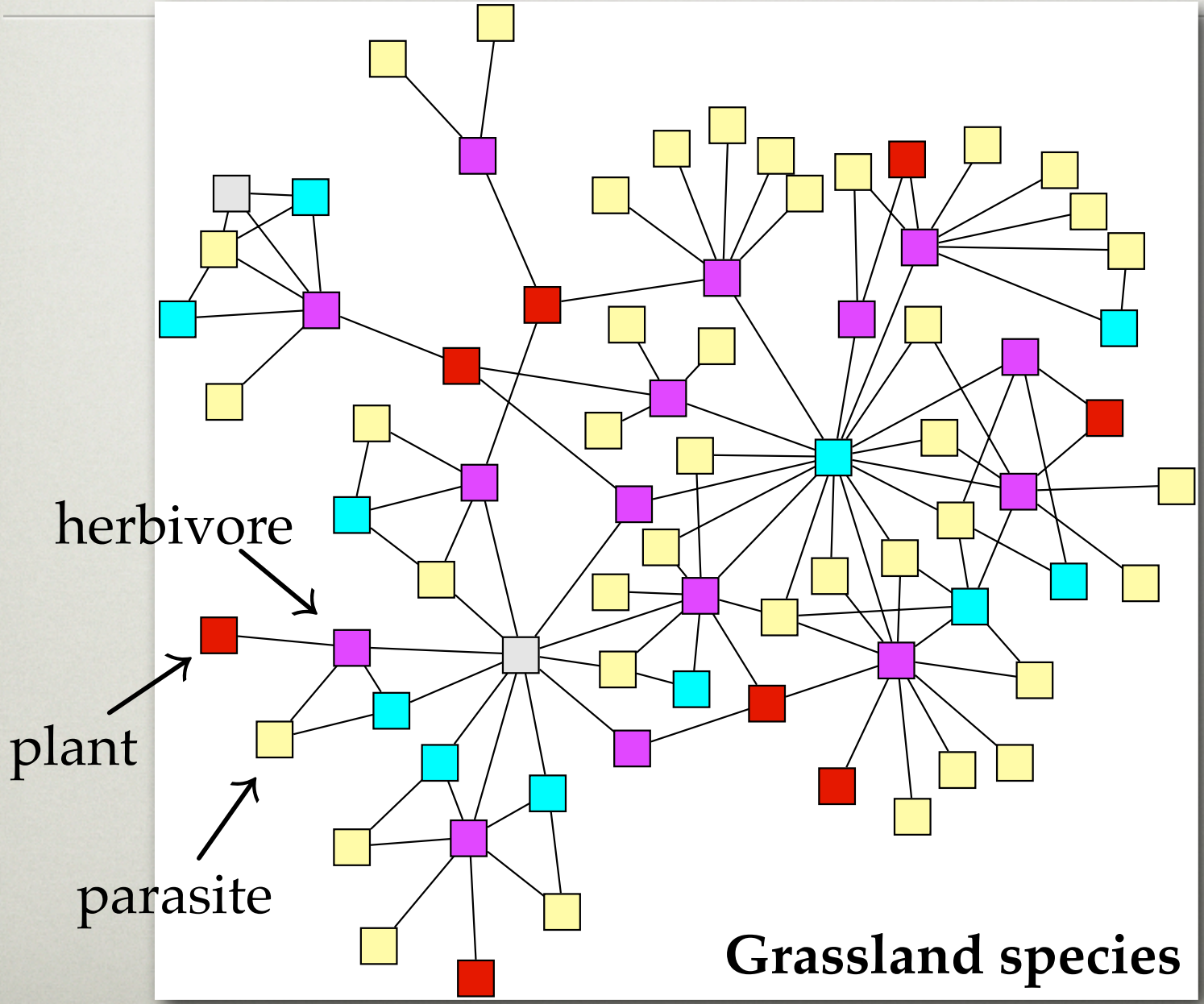
Average Probability of Edges and Vertices



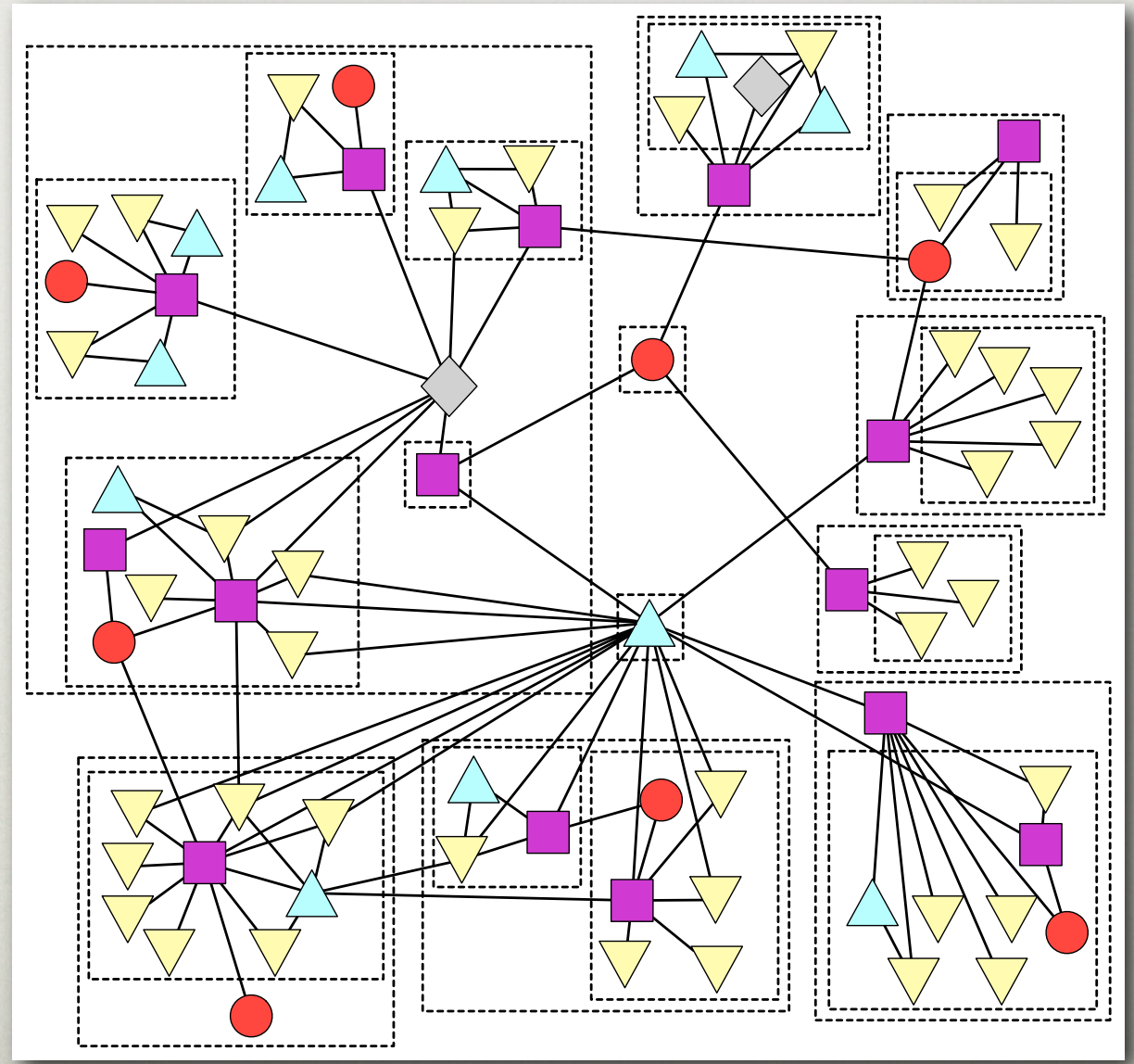
Goal #2: Generating Similar Random Graphs

- Idea: once we infer a hierarchical structure (or a distribution of them) we can use it to generate new random graphs.
- If these graphs are "similar" to the original (*n.b.: application-dependent!*) then we may have captured part of its structure. Generalizing from a single example!
- If they aren't, we have falsified our model, which is a good thing to be able to do...

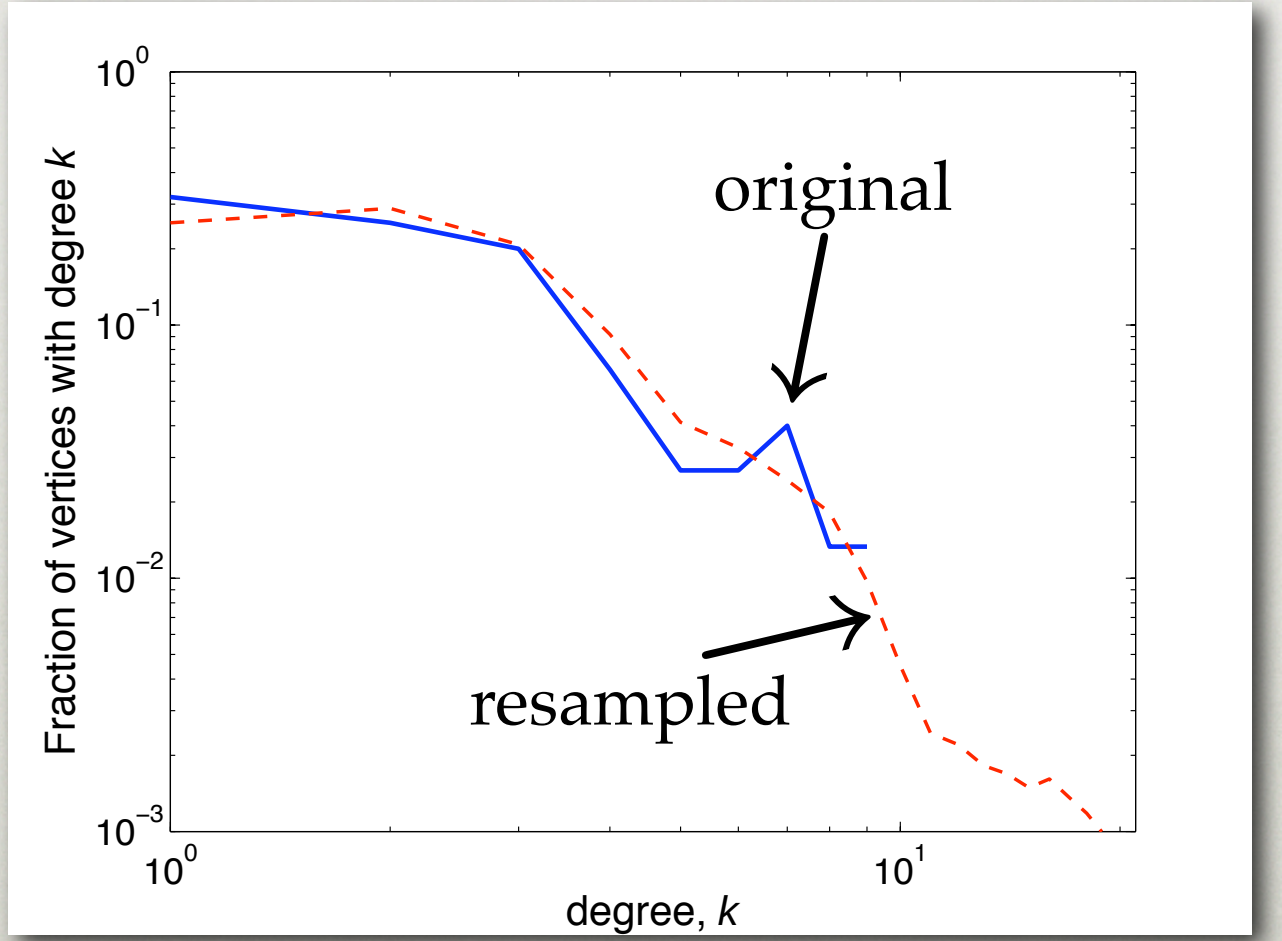
A Food Web



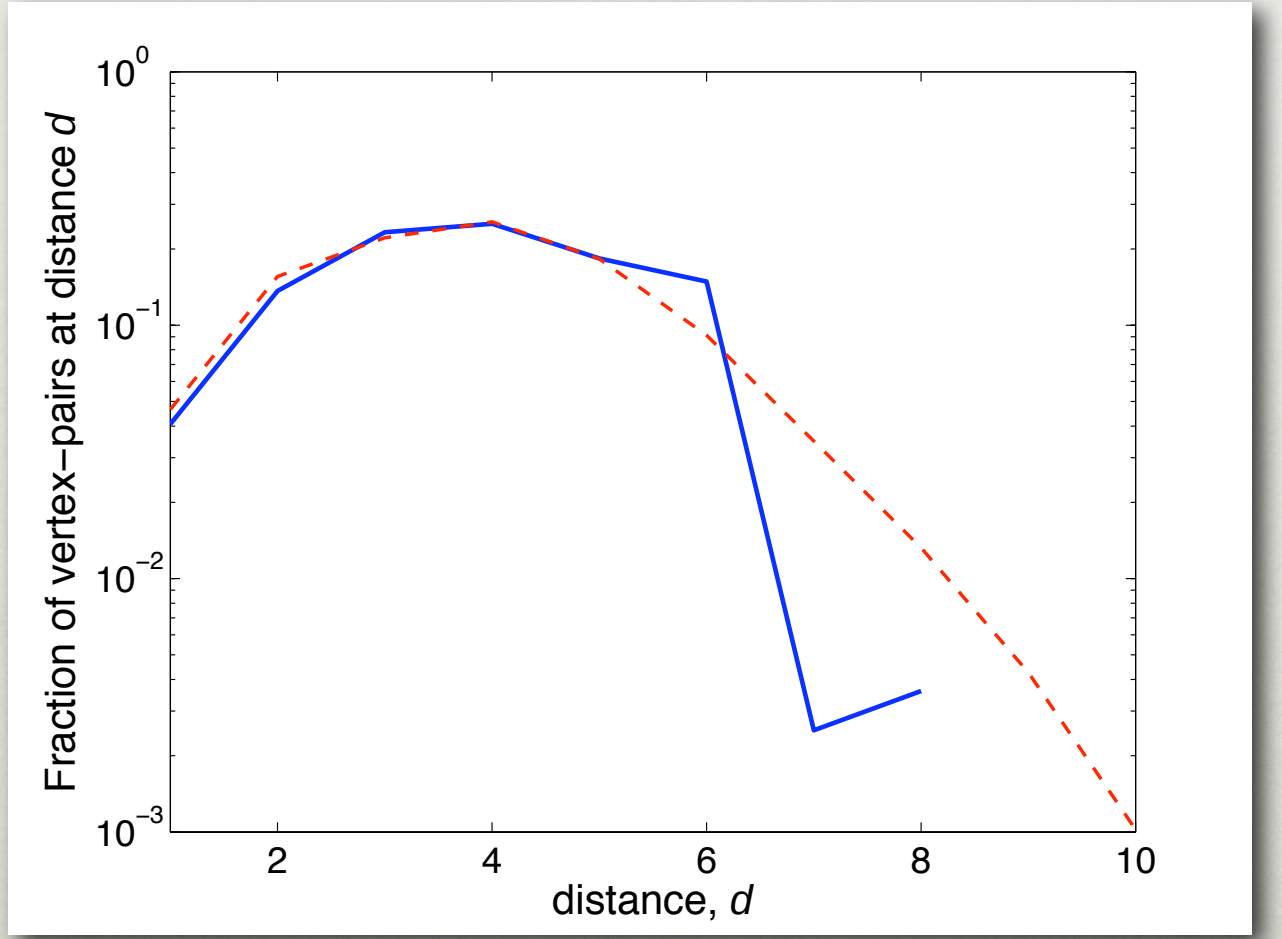
An Inferred Hierarchy



Resampling the Ensemble: Degree Distribution



Resampling the Ensemble: Distance Distribution



Goal #3: Link Prediction

- For many networks, edges are discovered one at a time, using difficult work in the field or laboratory
- Given the edges observed so far, can we predict missing ones better than chance?
- If so, we can focus our attention on pairs of vertices likely to be connected.

Our Approach

- Sample hierarchies using observed edges
- Sort remaining pairs according to the average probability they are connected
- Predict the top few of these
- Cross-validation: remove some fraction of edges randomly, and try to re-predict the ones you removed

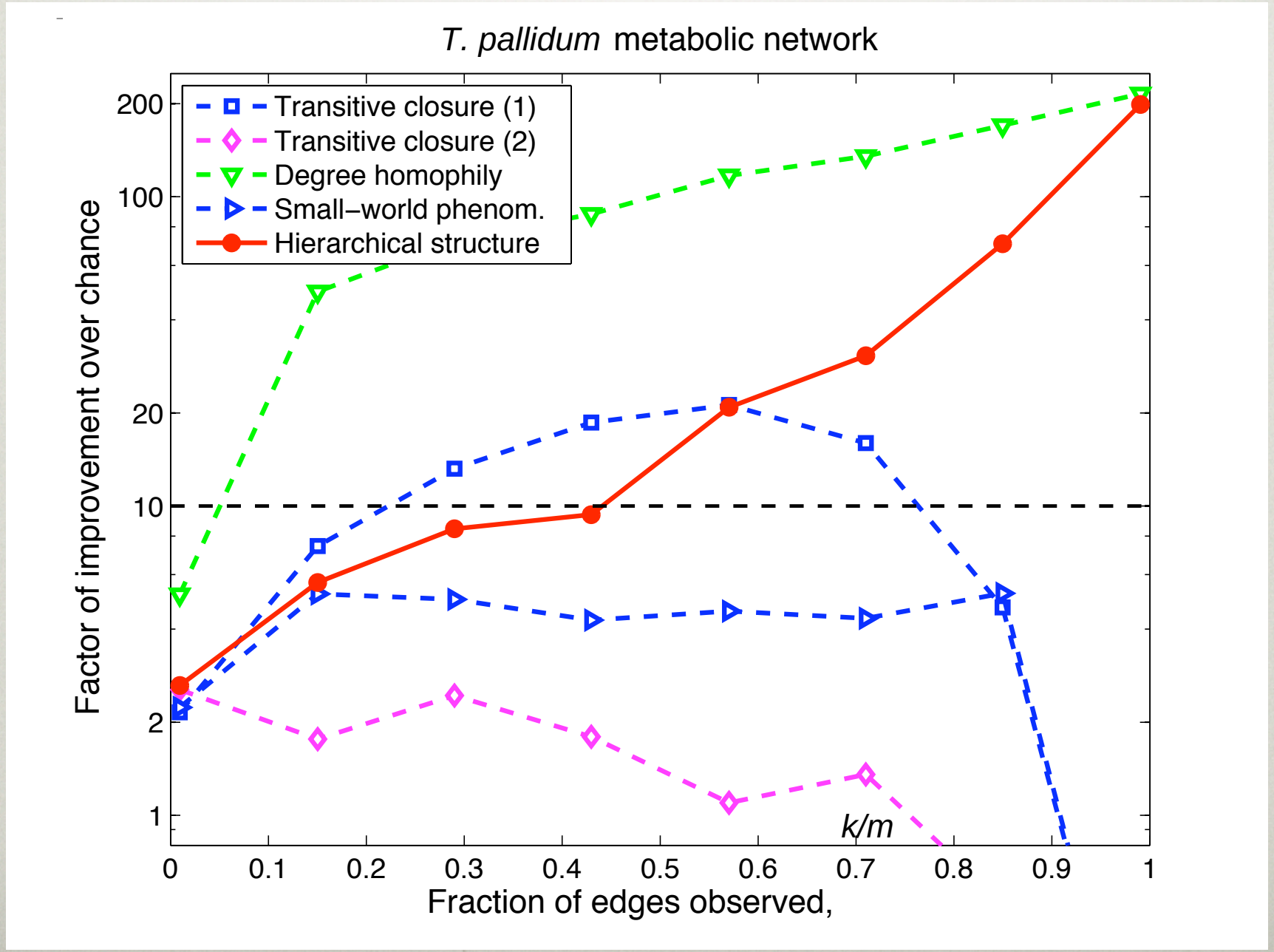
Some Simpler Methods

"Link Prediction" problem [Liben-Nowell and Kleinberg, 2003: predicting collaborations]

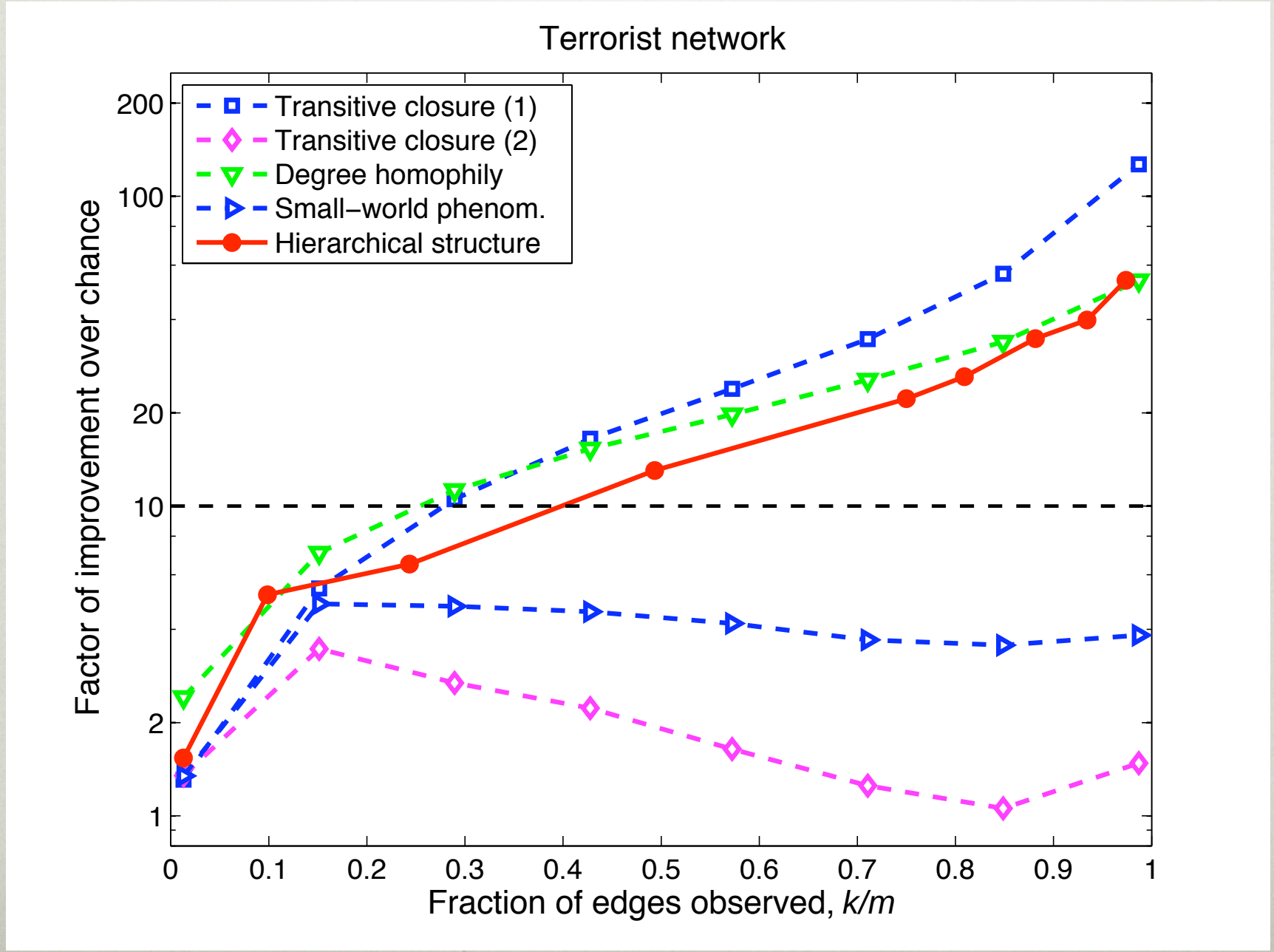
Guess that two vertices are connected if:

- They have many common neighbors
- They share a large fraction of their neighbors (Jaccard coefficient)
- The product of their degrees is large
- They have many short paths between them

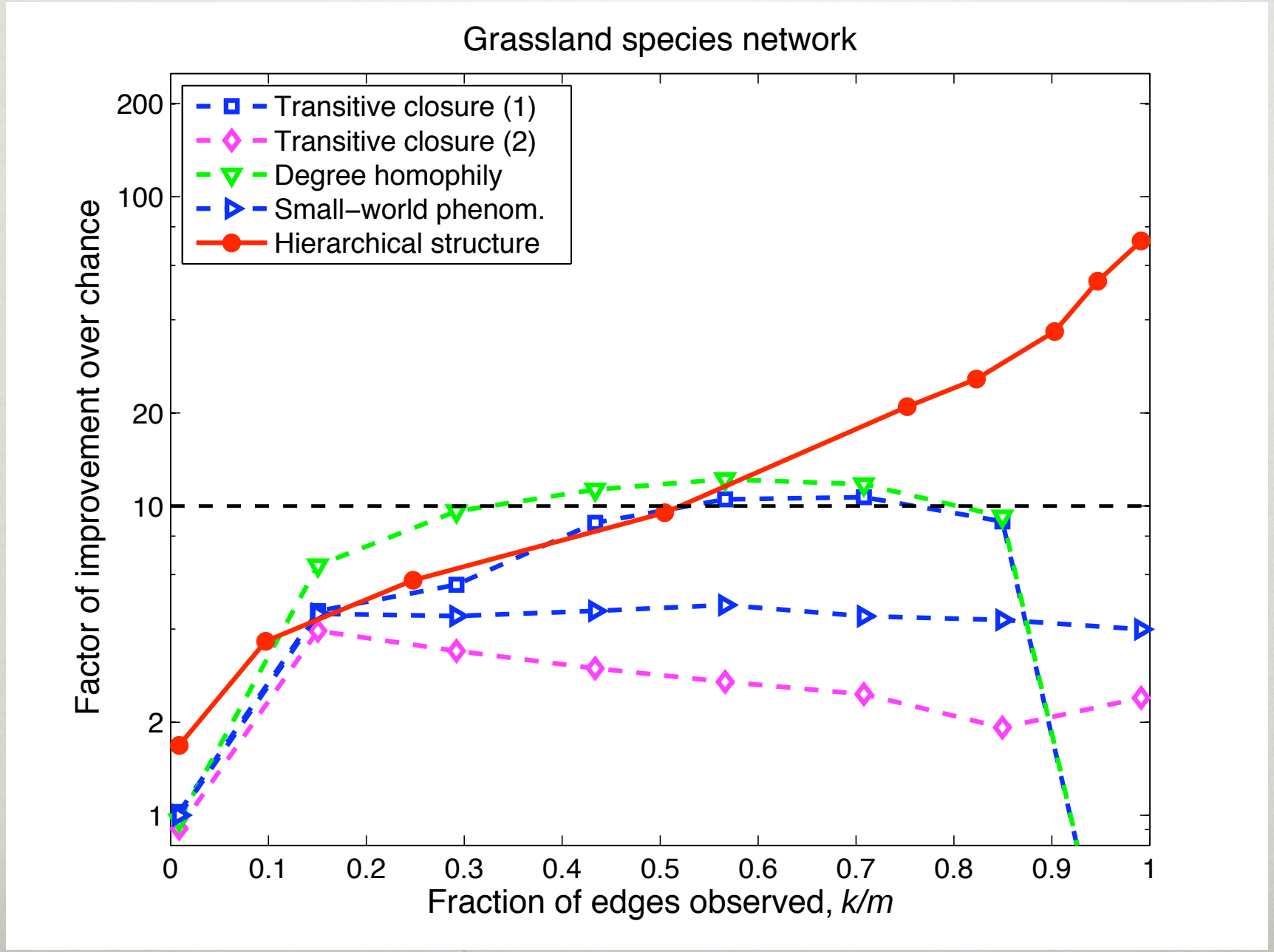
Other Methods: Metabolic



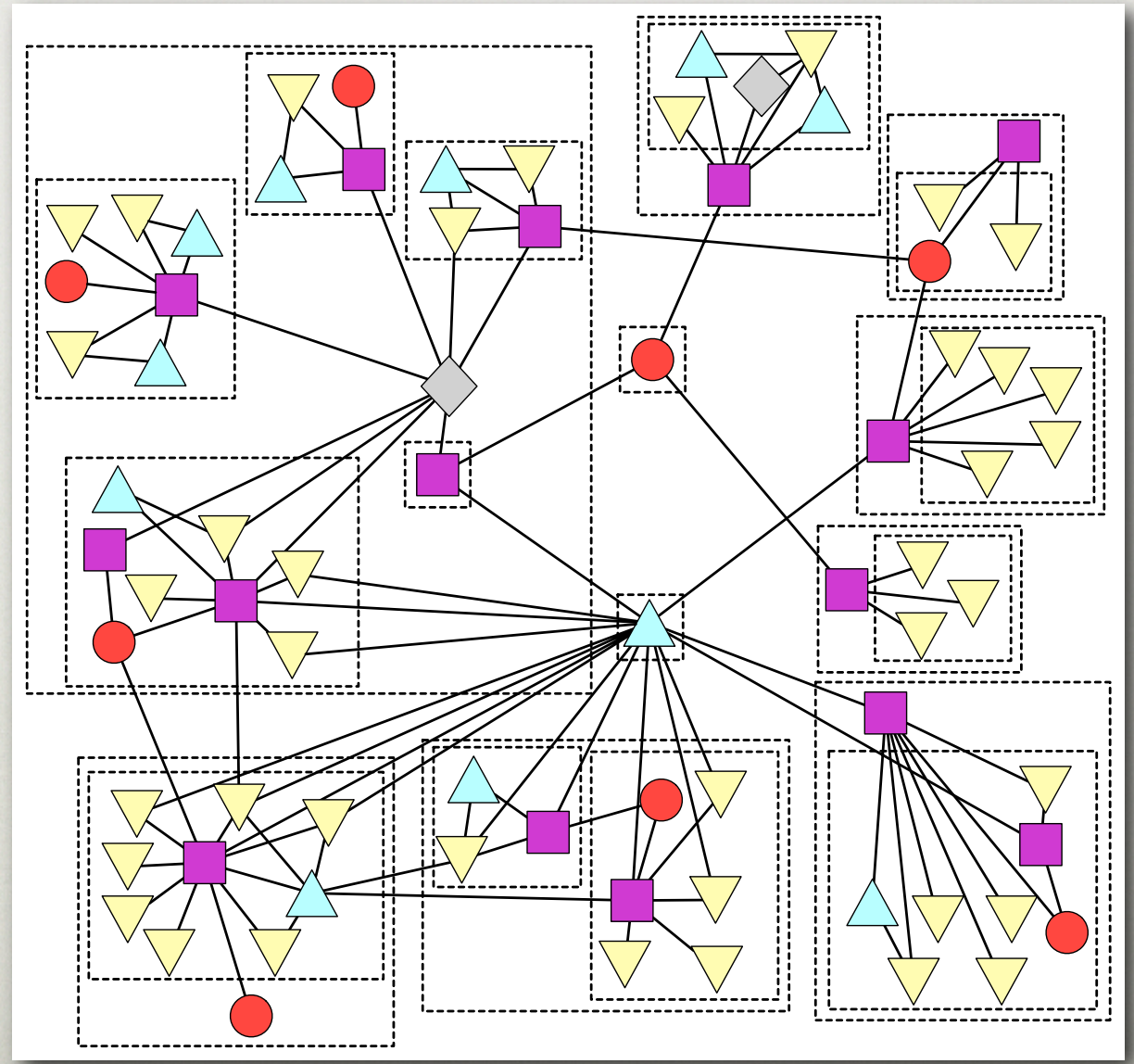
Other Methods: Bad Guys



Other Methods: Food Web



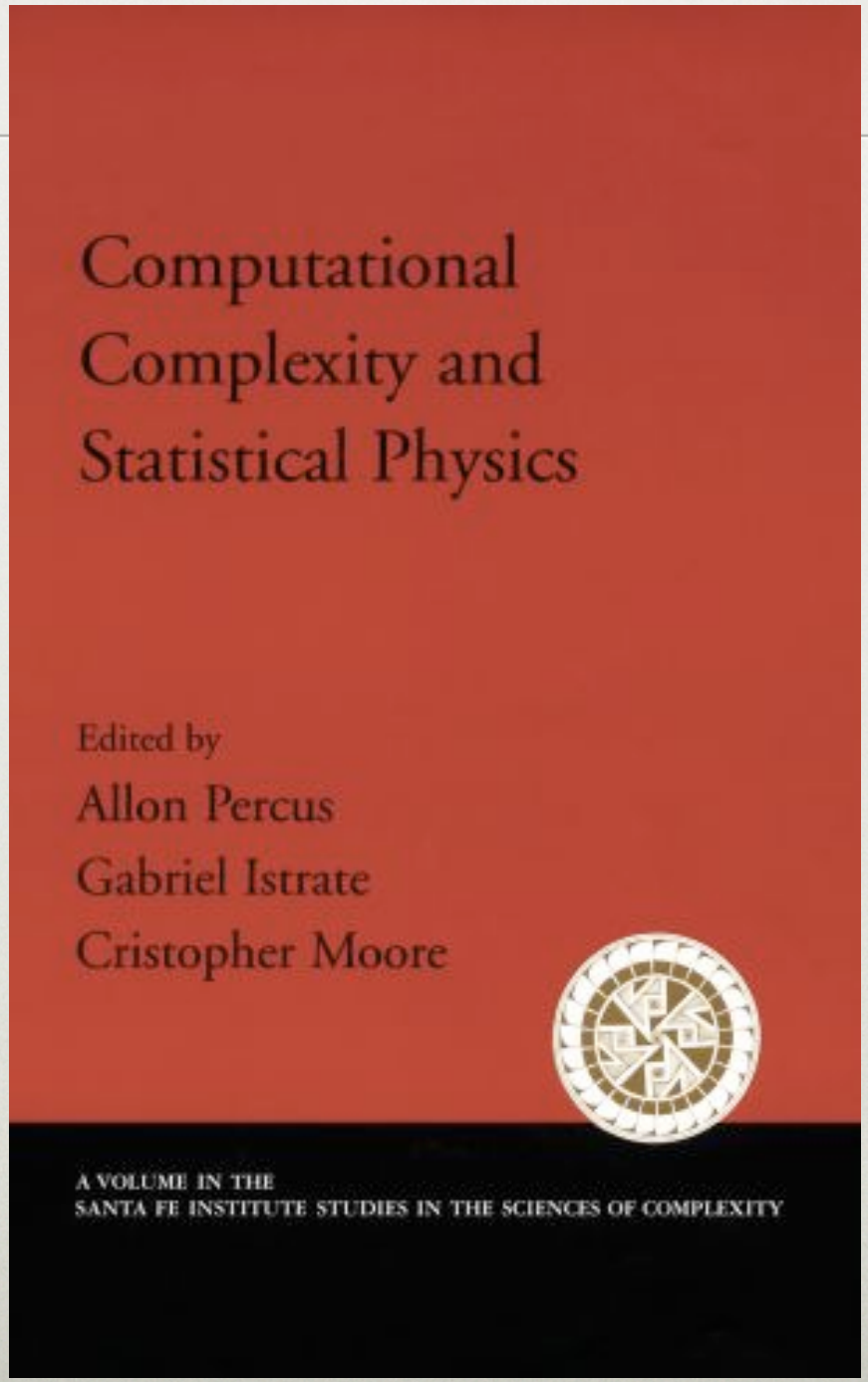
Disassortativity: Predators share Prey



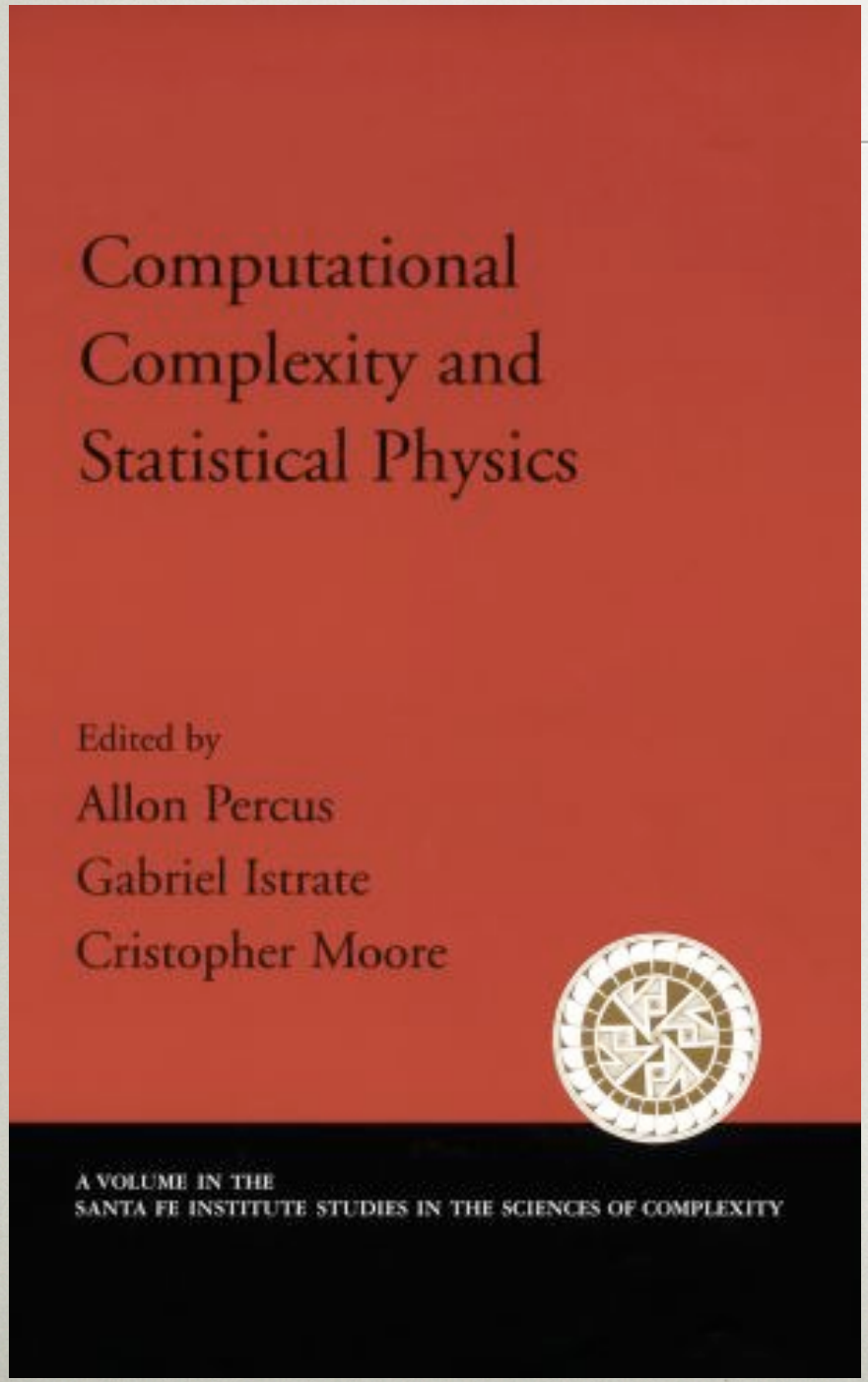
Conclusion

- We have a rich space of hierarchical random graph models, not just assortative
- We use a Markov chain to sample the likely models from this space based on data
- This lets us generate random graphs which are statistically similar to an observed one
- It also lets us predict missing links with probability much better than chance.

Shameless Plug



Shameless Plug



The Nature of Computation

Mertens and Moore

Acknowledgments