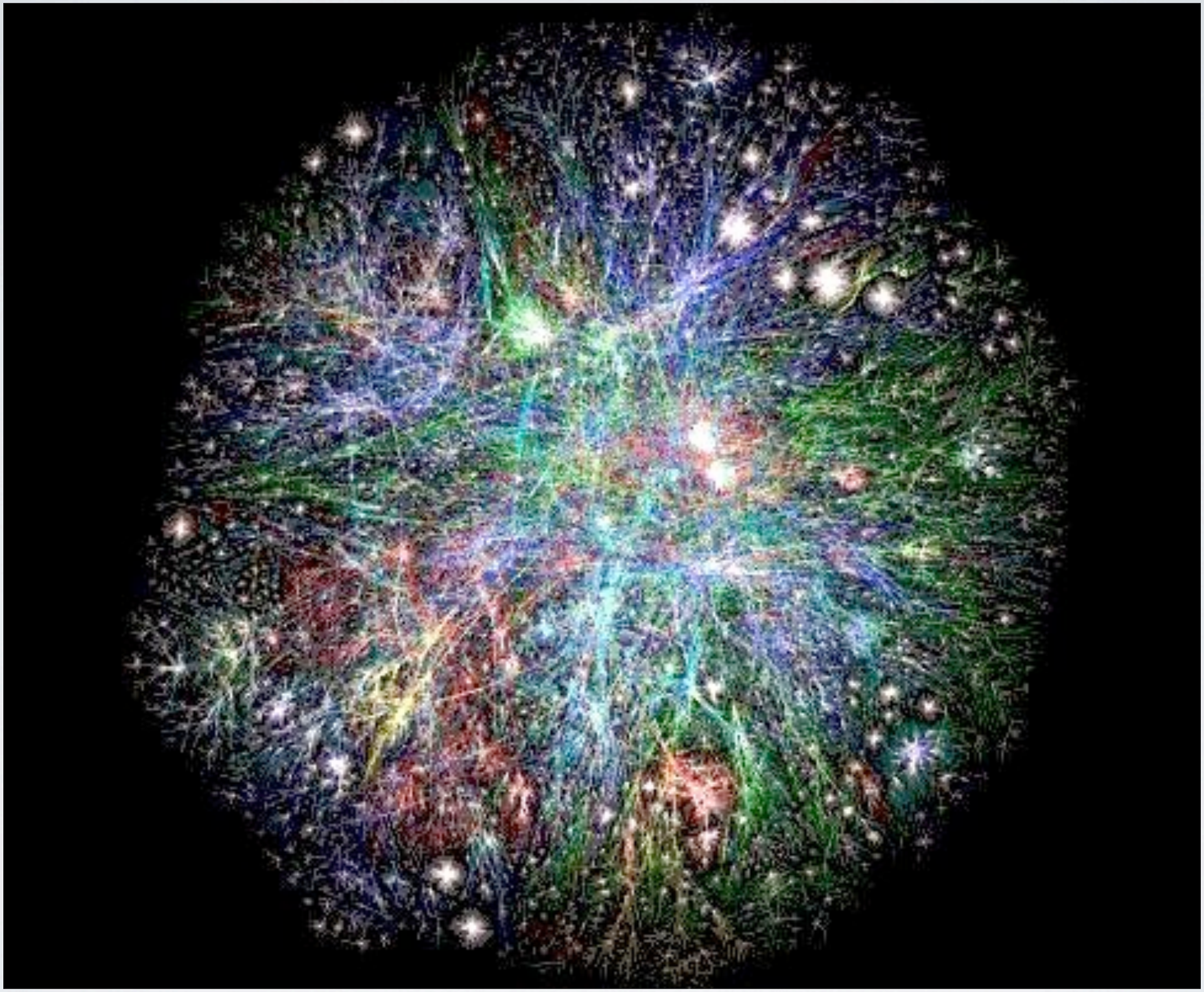
Finding Hidden Structure in Networks

Cristopher Moore, Santa Fe Institute

joint work with

Xiaoran Yan, Yaojia Zhu, Lenka Zdeborová, Florent Krzakala, Aurelien Decelle, Pan Zhang, Jean-Baptiste Rouquier, Tiffany Pierce, Cosma Shalizi, Jacob Jensen, Lise Getoor, Aaron Clauset, and Mark Newman



What is structure?

Structure is that which...

makes data different from noise: makes a network different from a random graph, or from a null model

helps us compress the data: describe the network succinctly, giving a human-readable summary of important structures

helps us generalize from data we've seen from data we haven't seen: e.g. predict missing links from the links we know about

helps us understand what multiple networks have in common: e.g. structure of food webs, from the Cambrian to today

helps us coarse-grain the dynamics, reducing the number of variables: e.g. compartmentalized models in epidemiology

The Bayesian approach

Imagine that the network is created by a *generative model*, and fit the parameters of this model to the data

We can gracefully incorporate partial information: e.g. if

attributes of some nodes are known, or known with some confidence

some links are known, others not observed yet (e.g. food webs)

some links might be false positives (e.g. gene regulatory networks, protein interactions)

Use the inferred model to generalize from what we do know to what we don't: label unknown nodes, predict missing links, mark false positives

Statistical inference

imagine that our graph *G* is drawn from an ensemble, or "generative model": some probability distribution $P(G|\theta)$ with parameters θ

θ can be continuous or discrete: represents the structure of the graph, properties of nodes and edges, etc.

maximum likelihood: given *G*, find the θ that maximizes $P(G|\theta)$

Bayes: compute, or sample from, the posterior distribution $P(\theta|G)$

if *G* is partly known, we can infer θ and use $P(G|\theta)$ to generate the rest of *G*: e.g. infer θ from known links, and predict missing links

if some parts of θ are known, can constrain the search and infer the rest of θ: e.g. if we know attributes of some nodes, can guess attributes of others

The Erdős-Renyí model

every pair of vertices *i, j* is connected independently with the same probability *p*

degree distribution is Poisson with mean *d=np*

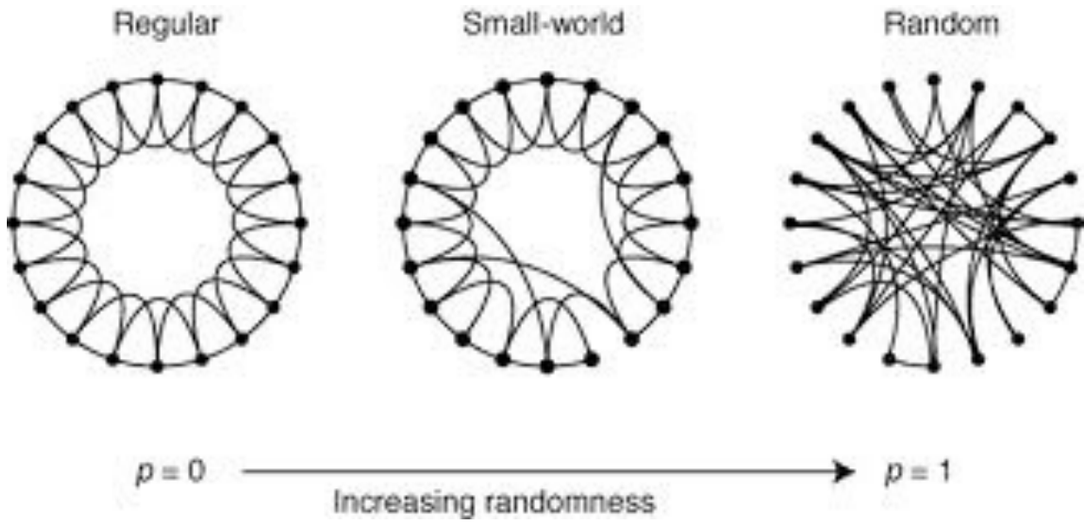
if *d* < 1, almost all components are trees, and max component has size O(log n)

if *d* > 1, a unique giant component appears

at *d* = ln *n*, completely connected

ring + Erdős-Renyí = Watts-Strogatz

but still pretends all nodes are the same...



The stochastic block model

nodes have discrete attributes: k types of nodes

each node i has type $t_i \in \{1,...,k\}$, with prior distribution $q_1,...,q_k$

$k \times k$ matrix p of connection probabilities

if $t_i = r$ and $t_j = s$, there is a link $i \to j$ with probability p_{rs}

p is not necessarily symmetric, and we don't assume that $p_{rr} > p_{rs}$

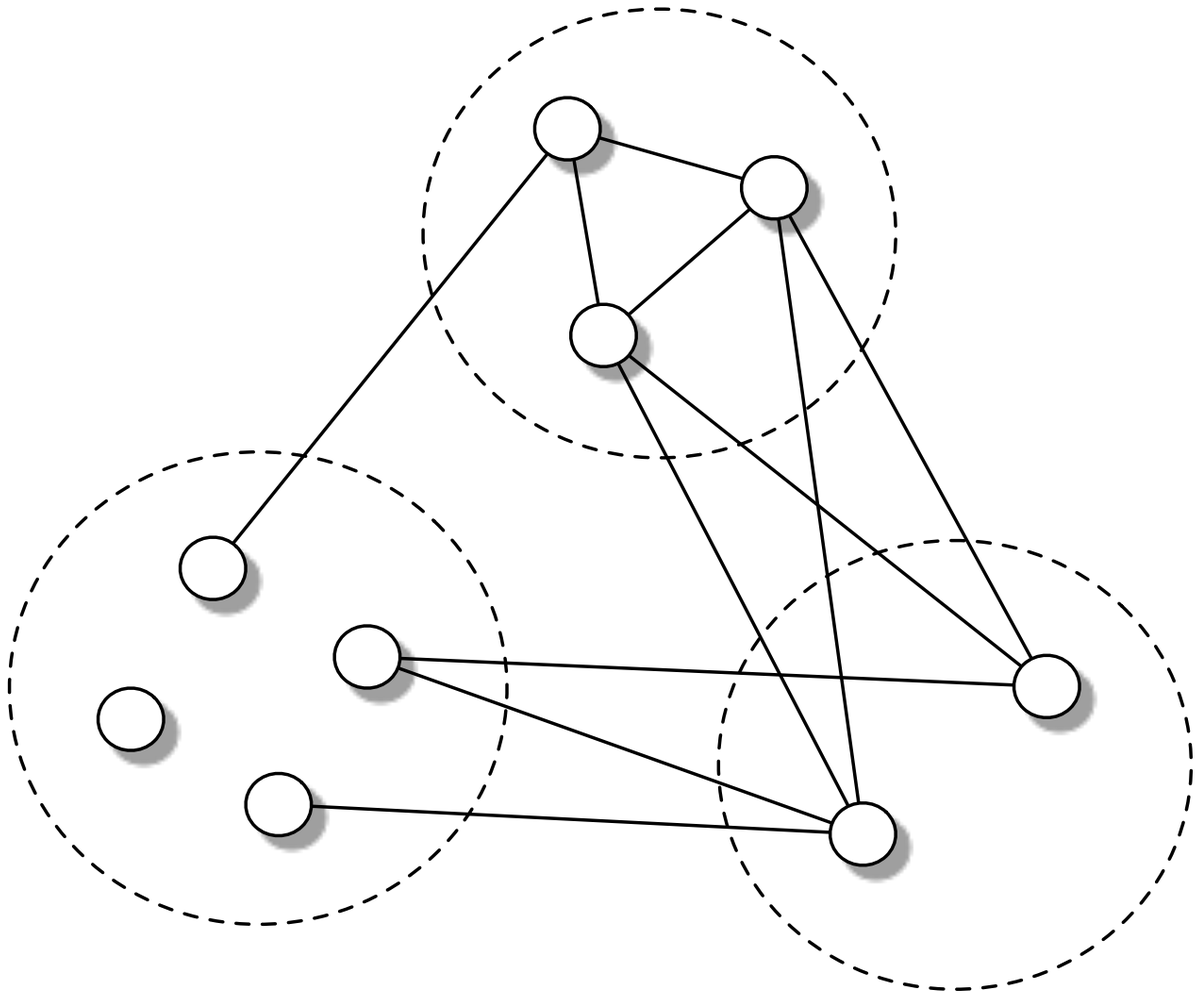
given a graph G, we want to simultaneously...

label the nodes, i.e., infer the type assignment $t : V \to \{1,...,k\}$

learn how types affect link probabilities, i.e., infer the matrix p

how do we get off the ground?

Assortative and disassortative



functional groups, not just clumps

food webs: predators and prey

economics: suppliers and customers

word adjacencies: adjectives and nouns

social: leaders and followers

The likelihood

the probability of *G* given the types *t* and parameters θ=(*p*,*q*) is a product

$$P(G \mid t, \theta) = \prod_{(i,j) \in E} p_{t_i,t_j} \prod_{(i,j) \notin E} (1 - p_{t_i,t_j})$$

so (after normalizing) the probability of *t* given *G* is

$$P(t \mid G, \theta) = \frac{P(t \mid \theta) P(G \mid t, \theta)}{\sum_{t' \in \{1,\ldots,k\}^n} P(G \mid t', \theta)}$$
$$\propto \prod_{i \in V} q_{t_i} \prod_{(i,j) \in E} p_{t_i,t_j} \prod_{(i,j) \notin E} (1 - p_{t_i,t_j})$$

A little statistical physics

the Boltzmann distribution: thermal equilibrium at temperature $T=1/\beta$

each state t is a set of "spins", or labels in our case

if a state t has energy E(t), then its probability is proportional to

$$P(t) \propto \mathrm{e}^{-\beta E(t)}$$

so (with β =1) the "energy" of a state in the block model is

$$E(t) = -\log P(G \mid t, \theta) = \sum_{(i,j)\in E} \log p_{t_i,t_j} + \sum_{(i,j)\notin E} \log(1 - p_{t_i,t_j})$$

like an Ising or Potts model (except non-neighbors also interact, since non-edges are informative)

Ground states vs. free energy

the most likely group assignment is a *ground state*: it maximizes

$$P(G \mid t, \theta)$$

and –log $P(G|t,\theta)$ is the *ground state energy*

one approach: find the $\theta=(p,q)$ that minimizes the ground state energy, i.e., maximize $P(G|t,\theta)$ as a function of t and θ

but this overfits! good ground states even when there no real communities

for instance, random 3-regular graphs have bisections with only about 15% of the edges crossing from one side to the other

there are communities in the graph but not the model

[Preview: it can be the other way around too!]

Ground states vs. free energy

better to use the total probability of *G* given θ, summed over all k^n labelings of the vertices:

$$P(G|\theta) = \sum_{t \in \{1,\ldots,k\}^n} P(G,t|\theta)$$

$$= \sum_{t \in \{1,\ldots,k\}^n} P(G|t,\theta)\,P(t|\theta)$$

this is a *partition function,* and –log *P*(G|θ) is a *free energy*

goal: find θ=*(p,q*) that minimizes the free energy, i.e., maximizes *P*(G|θ)

Expectation-Maximization

Gradient ascent (or descent) in parameter space

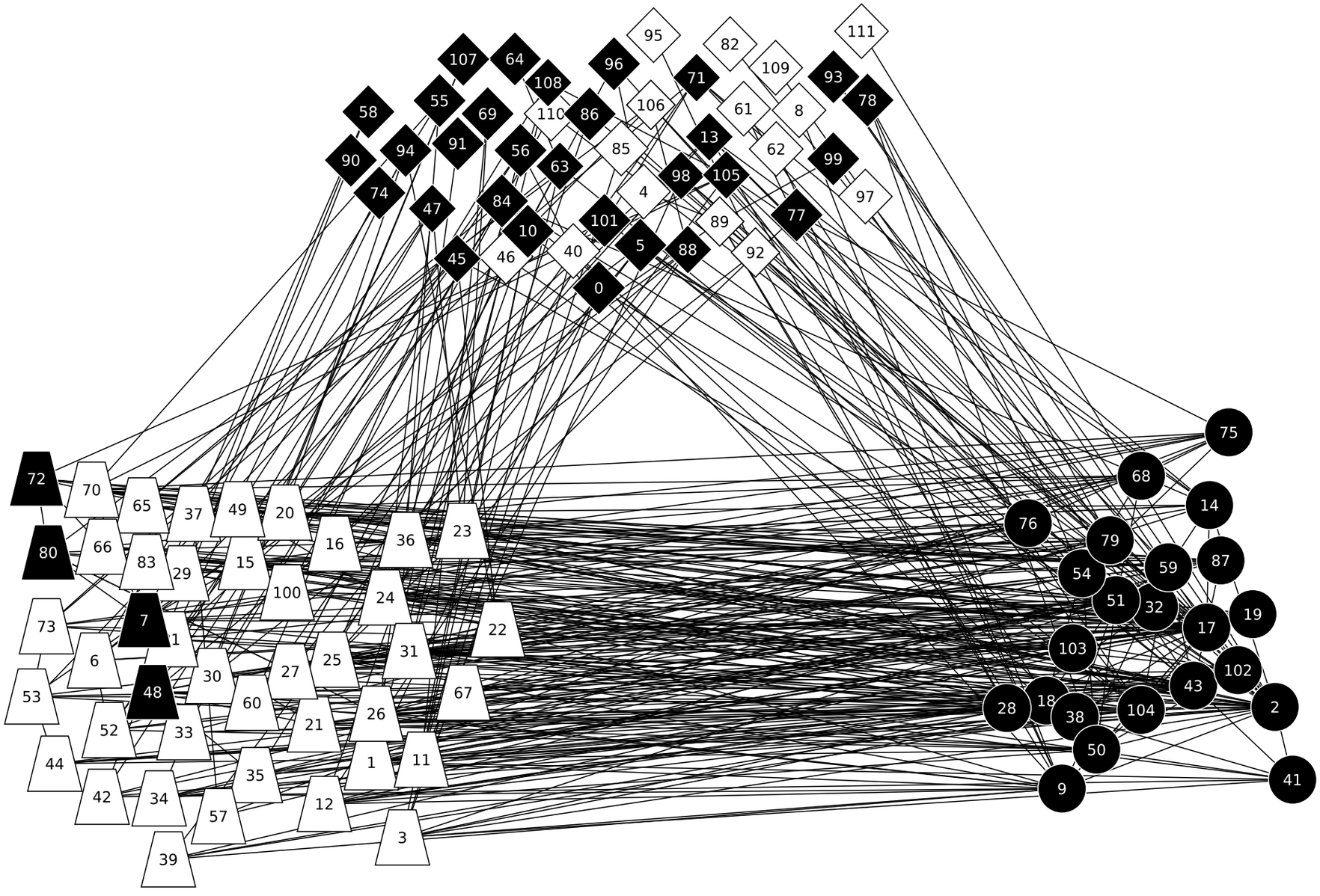
(E step) given the current θ=*(p,q)*, estimate one- and two-point marginals of the Gibbs distribution

(M step) update θ=*(p,q)* to their most likely values

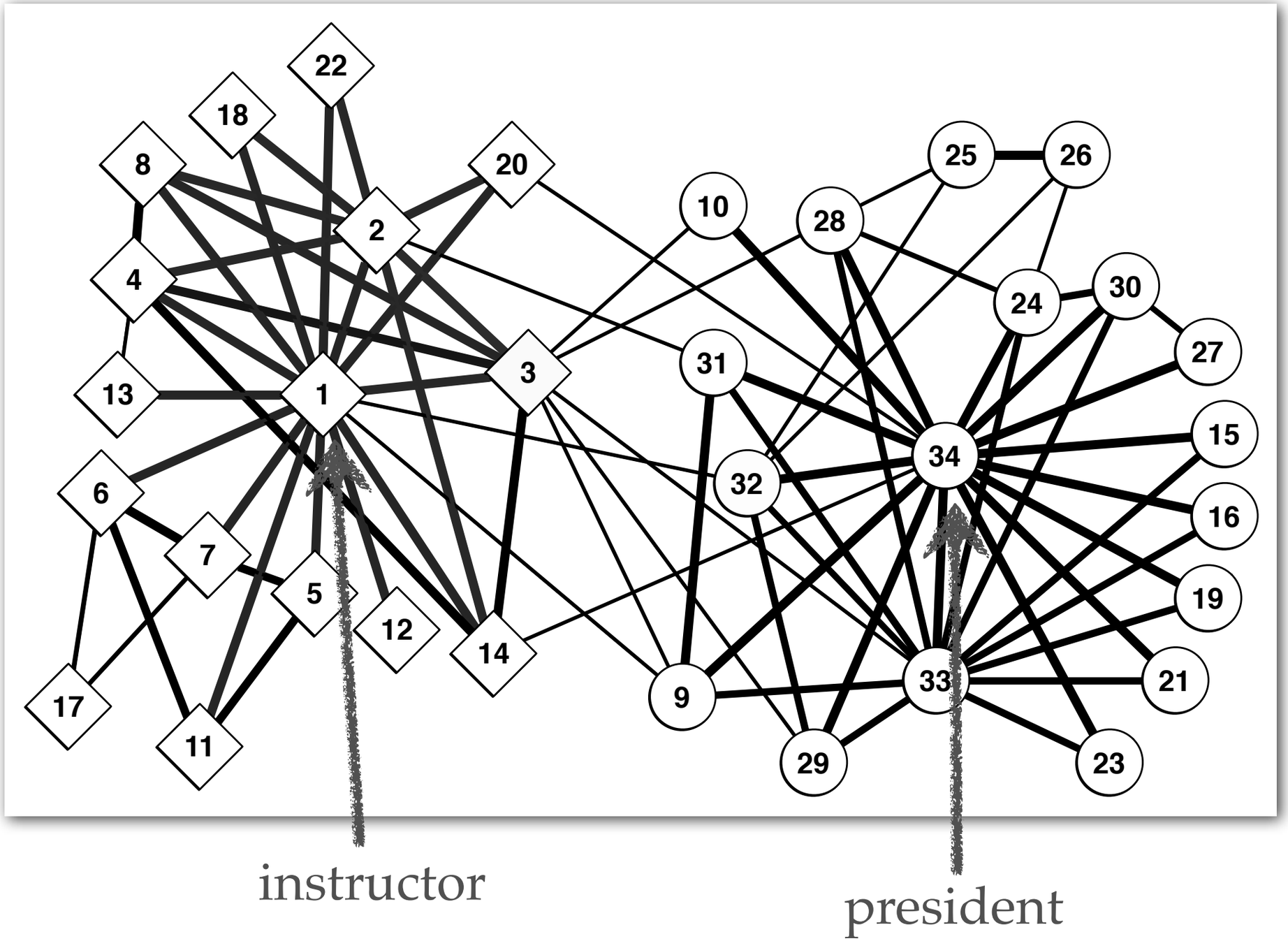
$$\mu_r^i = \Pr[t_i = r] \qquad \mu_{rs}^{ij} = \Pr[t_i = r \text{ and } t_j = s]$$

$$q_r = \frac{1}{N}\sum_i \mu_r^i \qquad p_{rs} = \frac{\sum_{(i,j)\in E} \mu_{rs}^{ij}}{q_r q_s N^2}$$

Classifying words with a ground state:
I record that I was born on a Friday



Classifying (softly) by Gibbs sampling: The Karate Club



Method #1:
Markov Chain Monte Carlo

computing $P(t|G,\theta)$ is hard, but it's a product of local terms

can compute ratios between $P(t|G,\theta)$ and $P(t' \mid G,\theta)$ if t and t' differ at one node

heat-bath dynamics: choose a random node v, fix types of all other nodes, update v's type according to its marginal distribution

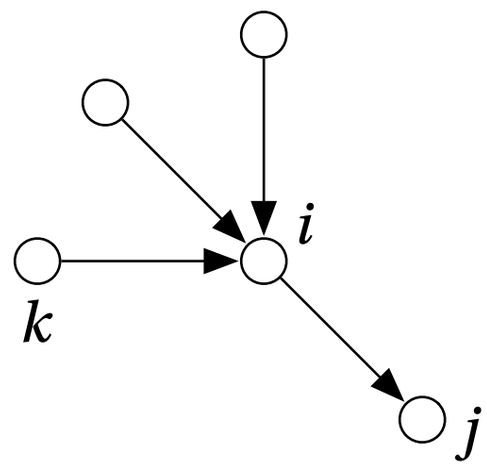
pretty good for finding ground states, but can get stuck in local optima

can speed up by introducing a temperature parameter:

- simulated annealing
- population annealing
- parallel tempering

but there's no free lunch

Method #2:
Belief propagation (a.k.a. the cavity method)

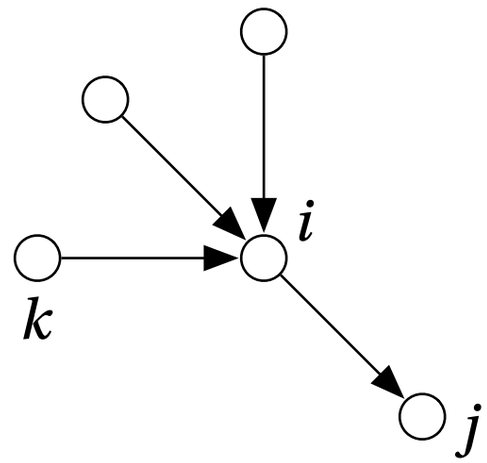


each node *i* sends a “message” to each of its neighbors j, giving *i*’s marginal distribution based on its other neighbors *k*

denote this message $\mu_r^{i\to j} = \text{estimate of } \Pr[t_i = r] \text{ if } j \text{ were absent}$

how do we update it?

Updating the beliefs



conditional independence

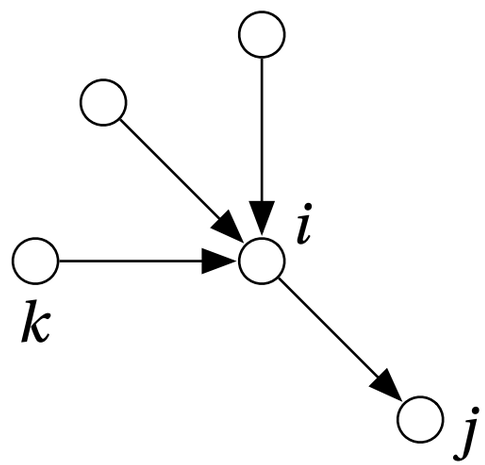
$$\mu_s^{i\to j} = \frac{1}{Z^{i\to j}} q_s \prod_{\substack{k\neq j \\ (i,k)\in E}} \sum_r \mu_r^{k\to i} p_{rs} \times \prod_{\substack{k\neq j \\ (i,k)\notin E}} \sum_r \mu_r^{k\to i} (1-p_{rs})$$

a complete graph of messages—takes O(n^2) time to update

can simplify by assuming that $\mu_r^{k\to i} = \mu_r^k$ for all non-neighbors *i*

each node *k* applies an "external field" $\sum_r \mu_r^k (1-p_{rs})$ to all vertices of type *s*

Making belief propagation scalable



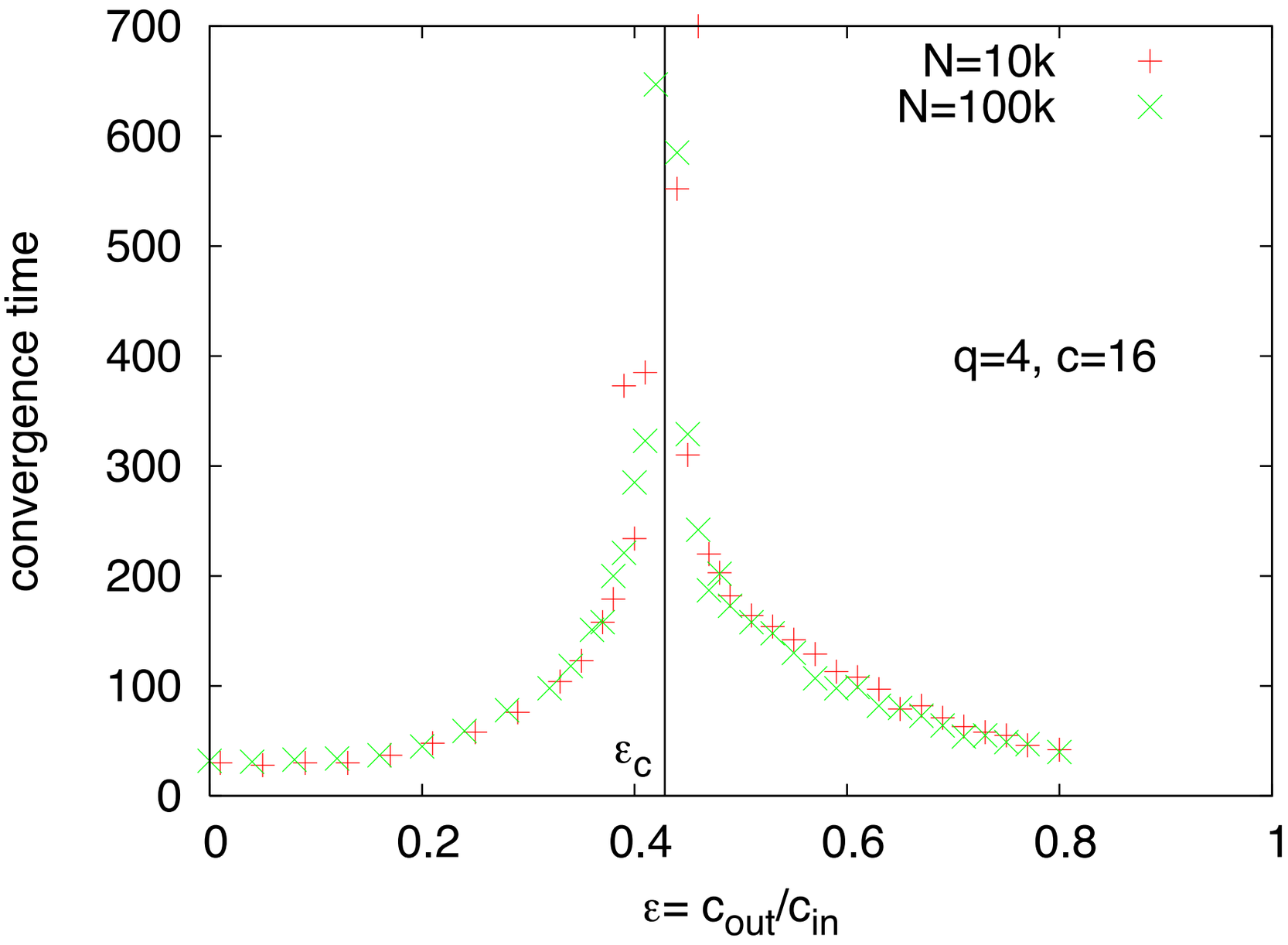
$$\mu_s^{i\to j} = \frac{1}{Z^{i\to j}} q_s \prod_{\substack{k\neq j \\ (i,k)\in E}} \sum_r \mu_r^{k\to i} p_{rs} \times \frac{\prod_{k\neq j} \sum_r \mu_r^k (1-p_{rs})}{\prod_{(i,k)\in E} \sum_r \mu_r^{k\to i} (1-p_{rs})}$$

each update now takes O(*n*+*m*) time: scalable!

update until the messages reach a fixed point

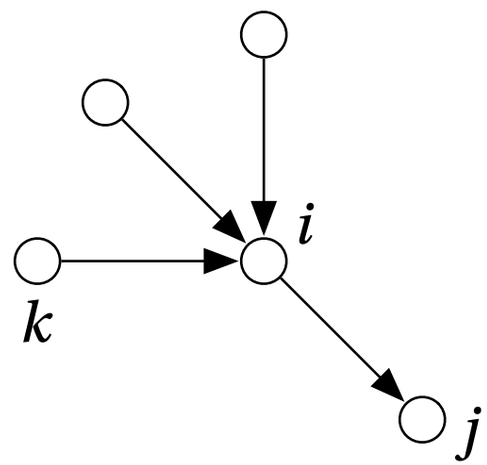
like Monte Carlo, can get stuck: try different initial messages

BP converges in a small number of iterations on many networks: finite correlation length



[Decelle, Krzakala, Moore, Zdeborová, PRL 2011]

Belief propagation: scalability, learning, marginals, free energy



total running time is nearly linear: can handle millions of nodes on a laptop

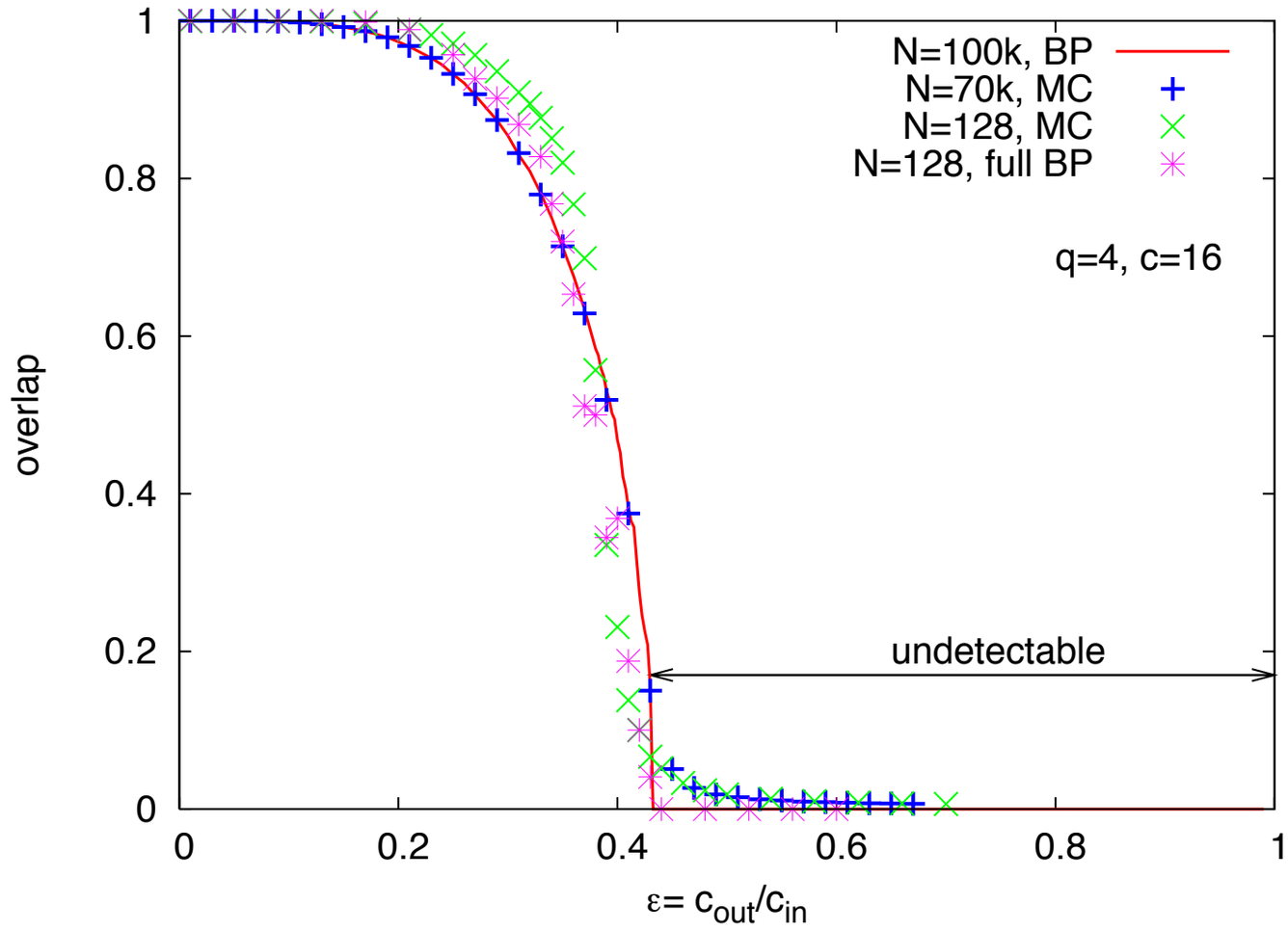
for each setting of the parameters θ, can compute the *Bethe free energy*: a good approximation even for graphs with loops

can explore free energy landscape as a function of θ

Expectation-Maximization (EM) algorithm: find θ that maximizes $P(G|\theta)$ (minimizes free energy)

returns marginals, i.e. soft clustering, and two-point correlations

A phase transition: detectable to undetectable communities



when the rows of p_{ij} are different enough, BP can recover the communities

but there is a transition where it can't — and no algorithm can!

the ensemble of graphs "knows" the communities, but a typical graph doesn't

[Decelle, Krzakala, Moore, Zdeborová, PRL 2011; Mossel, Neeman, Sly 2012]

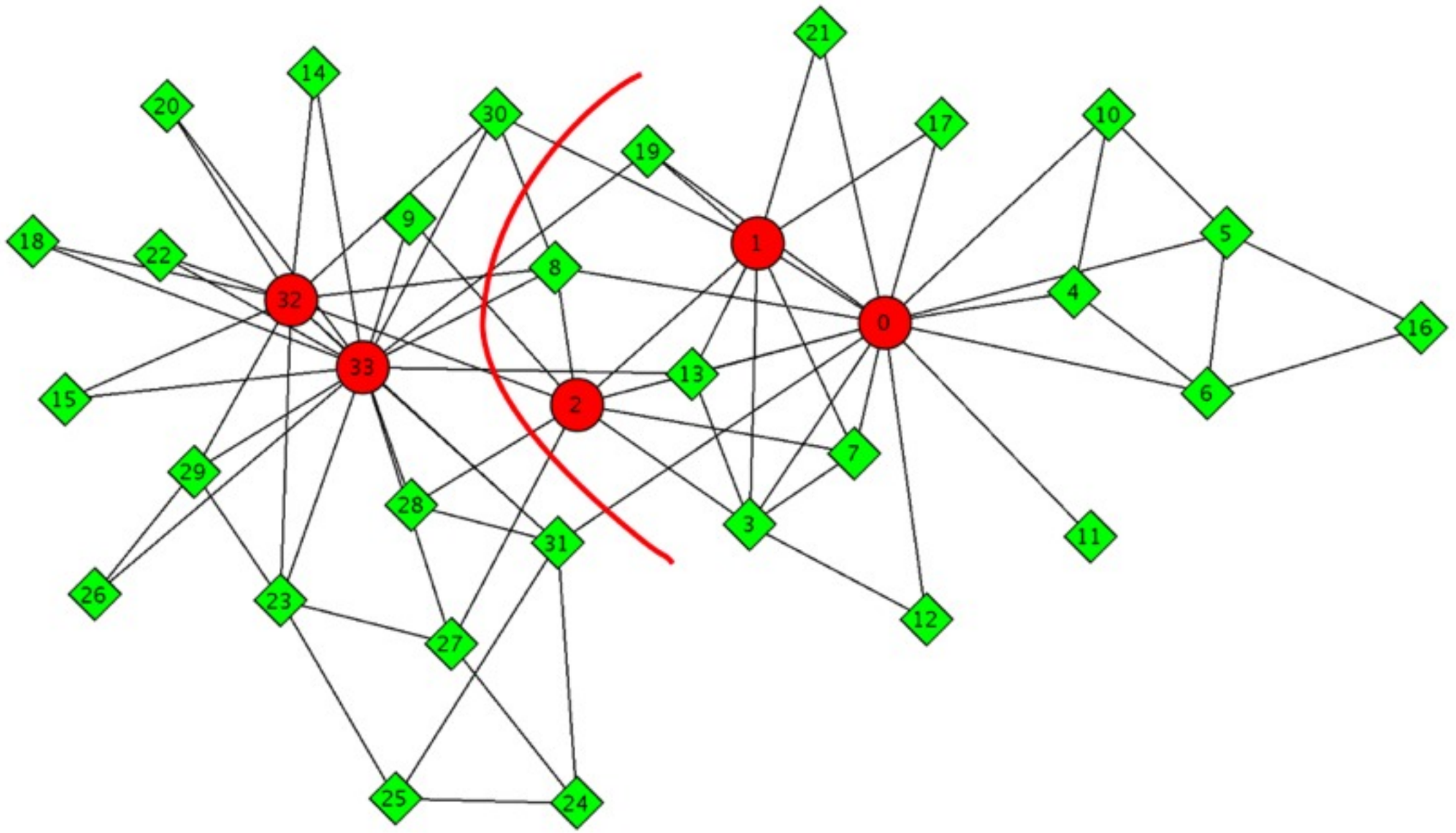
What kind of community do you want?

different models give different answers for the communities

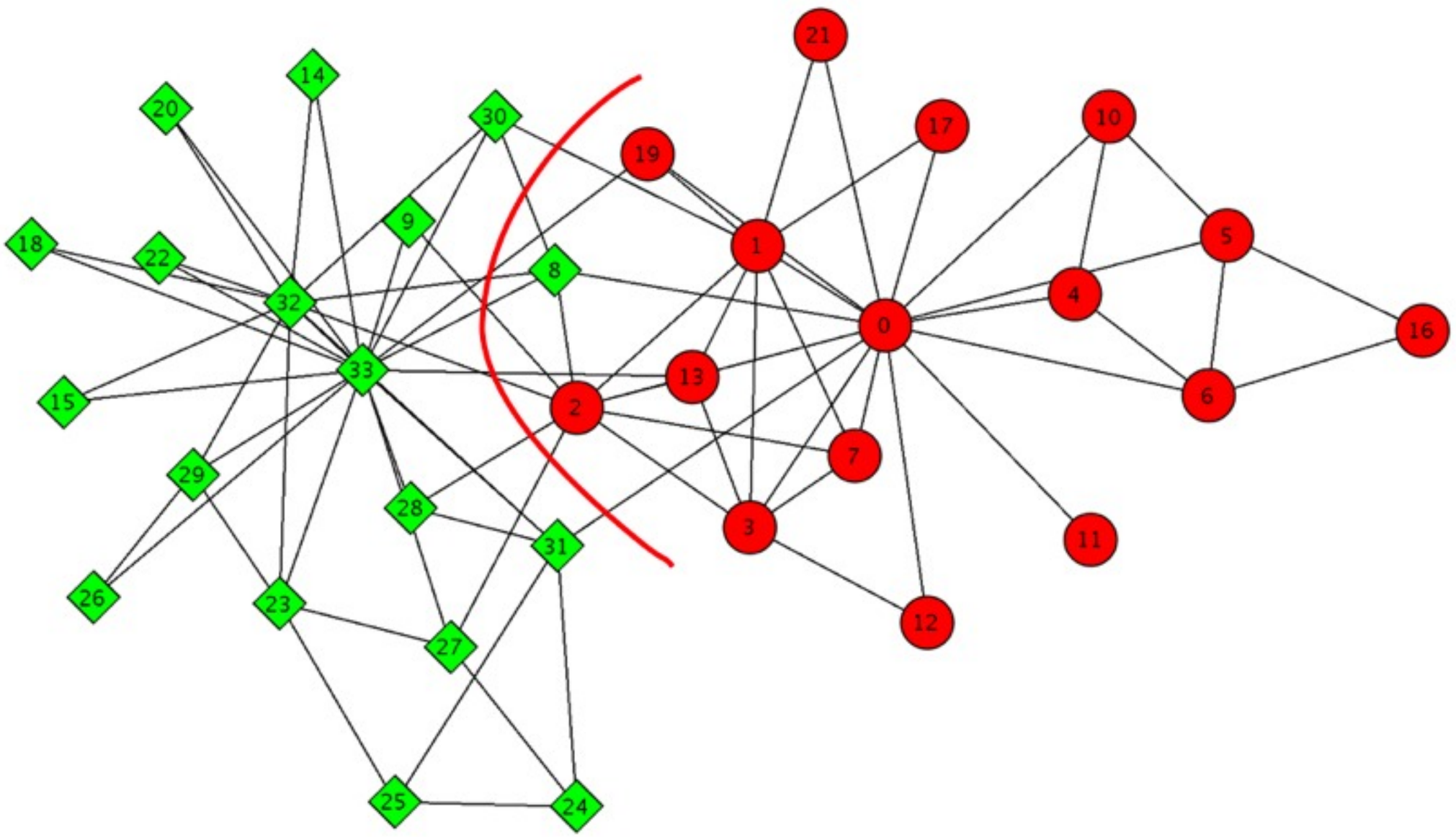
we can compare each one to “ground truth” and judge its accuracy...

...or embrace the fact that they are sensitive to different kinds of structure

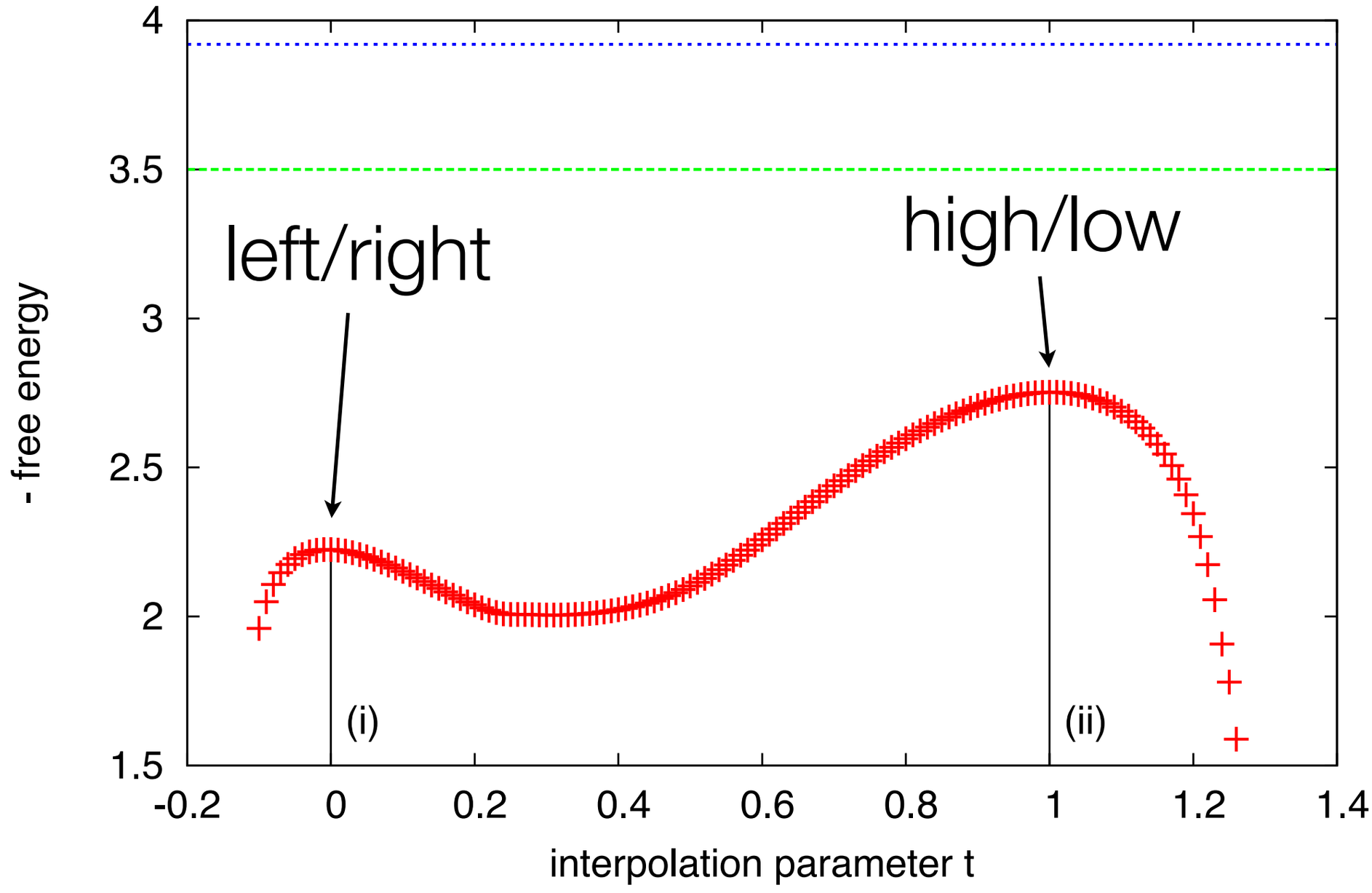
The Karate Club again: Leaders vs. followers



The Karate Club again:
Two factions



Two local optima in free energy



Degree-corrected block models

the “vanilla” block model expects vertices of the same type to have roughly the same degree:

account for “intrinsic” degree, or popularity, of nodes [Karrer & Newman, 2010]

each node *i* has an expected degree d_i

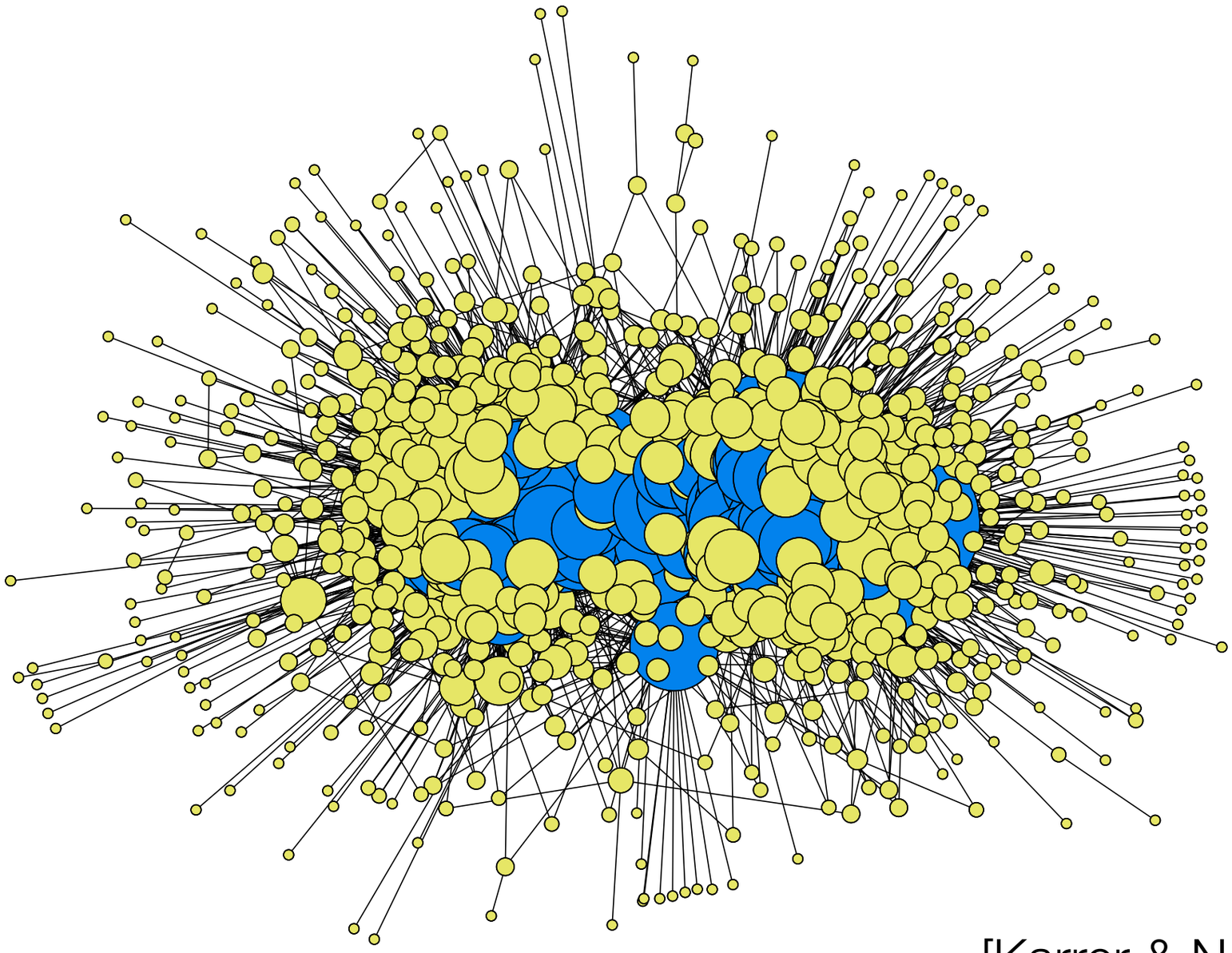
for nodes *i*, *j* of types *r*, *s*, number of edges A_{ij} is Poisson-distributed:

$$A_{ij} \sim \mathrm{Poi}(d_i d_j w_{rs})$$

now the degrees are parameters, not data to be explained

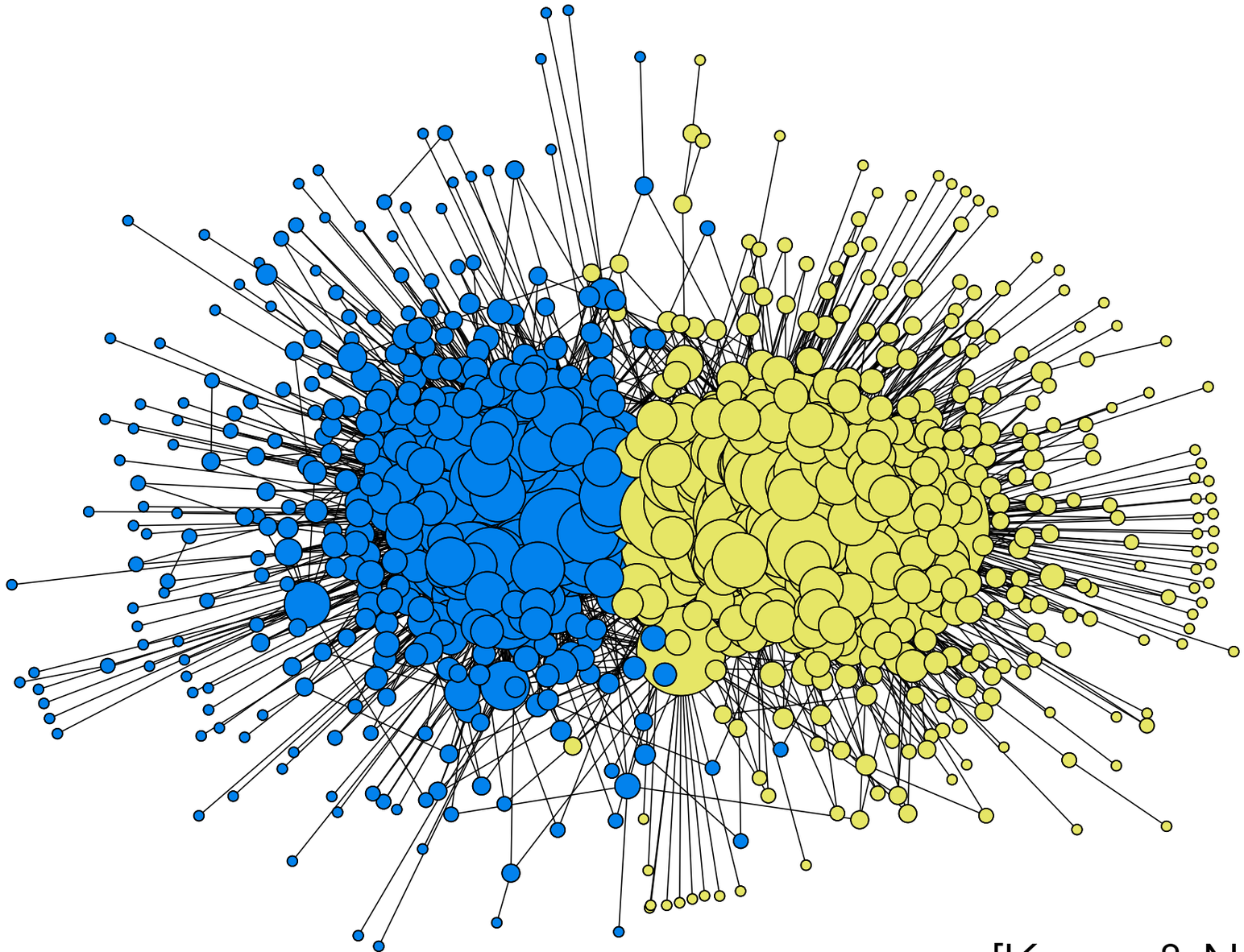
can again write down the BP equations, and use them in an EM algorithm

Blogs: vanilla block model



[Karrer & Newman, 2010]

Blogs: degree-corrected block model



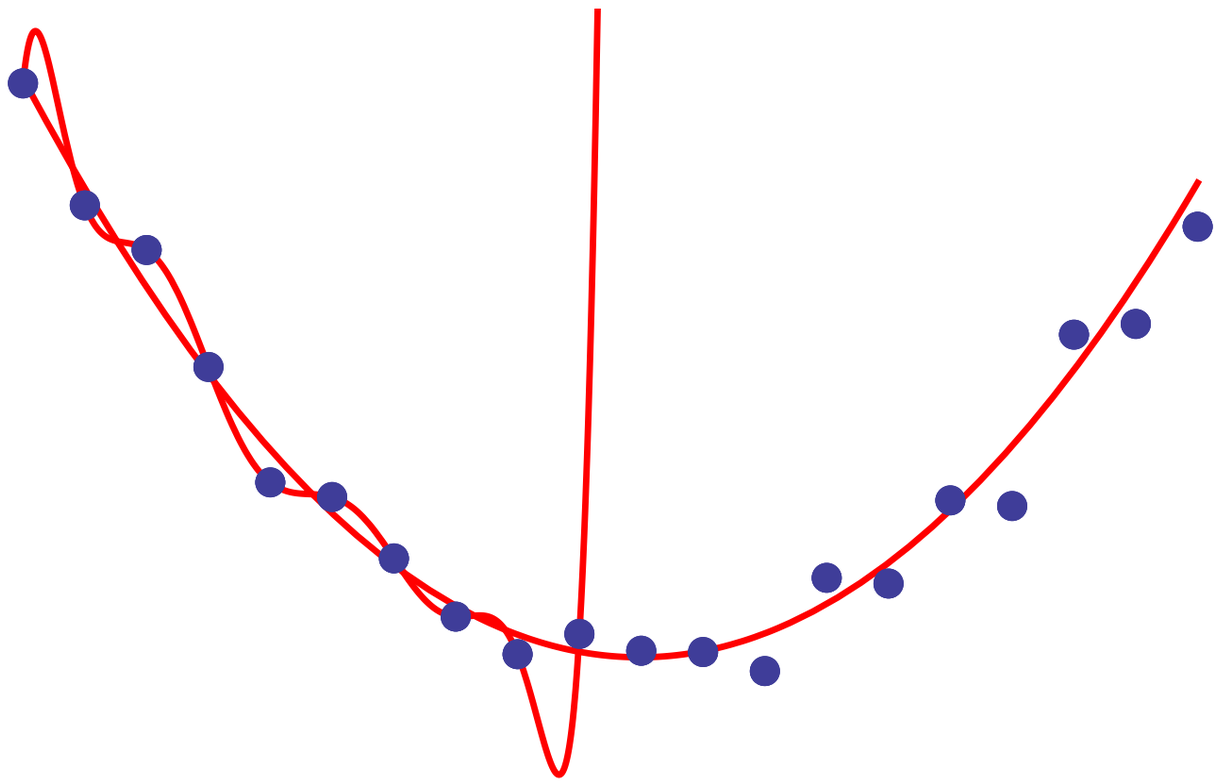
[Karrer & Newman, 2010]

Model selection

when the “vanilla” stochastic block model disagrees with the degree-corrected one, which one should we use?

the vanilla model is a special case of the degree-corrected model, so the degree-corrected model always gets a better fit (higher likelihood)

but is this just overfitting? are the extra parameters worth it?



Likelihood-based hypothesis testing

when the “vanilla” stochastic block model disagrees with the degree-corrected one, which one should we use?

likelihood ratio test: how large is

$$\frac{\max_\theta P_{\mathrm{DC}}(G|\theta)}{\max_\theta P_{\mathrm{SBM}}(G|\theta)}$$

log is difference between two free energies

Q: how large does the difference need to be to justify the fancy model?

A: larger than it would be if *G* were actually generated by the simple model (with a small *p*-value)

only then can we reject the null hypothesis, i.e., the simple model

Beyond χ^2 and the AIC

classical result: if the fancy model has *n* more parameters, the log-likelihood ratio follows a χ^2 distribution, with mean *n*/2

but this relies on a "large data limit": assumption that the likelihood, and posterior distribution of parameters, has a Gaussian peak

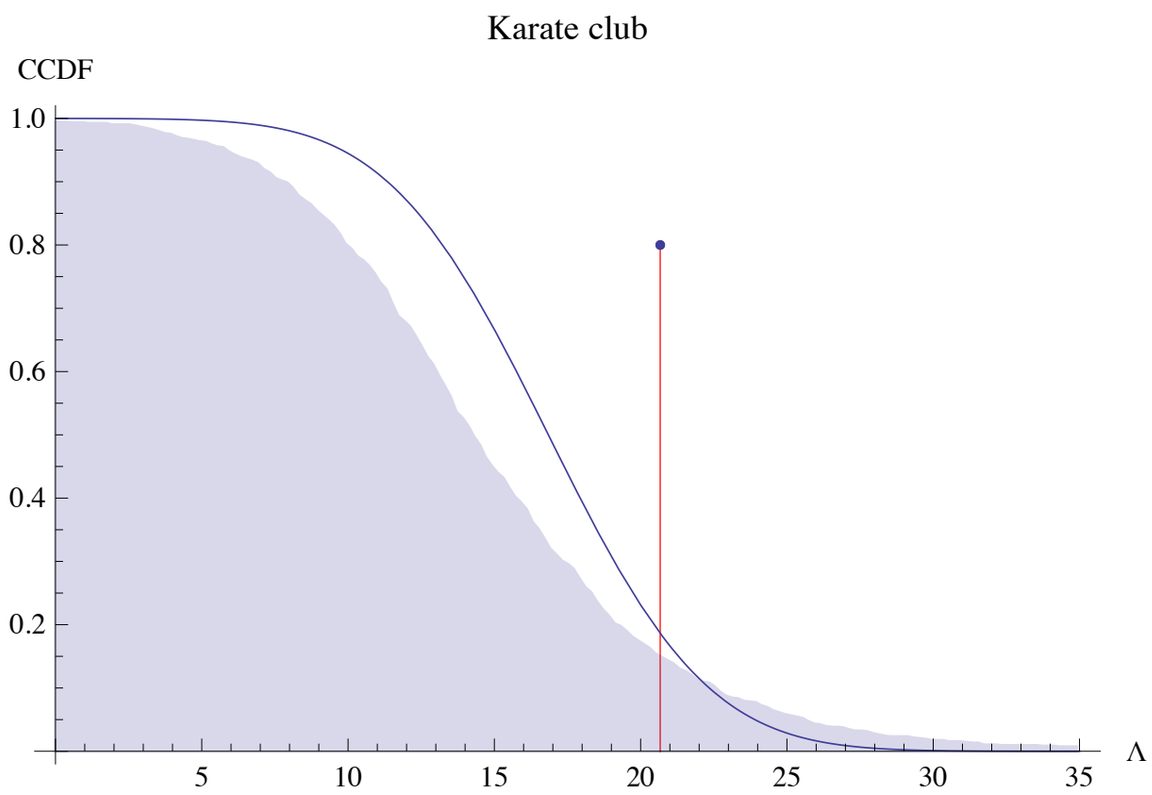
holds for i.i.d. data, but network data is highly correlated

in the degree-corrected model, degree of each node *v* is Poisson with mean μ_v and we get just one observation of this Poisson

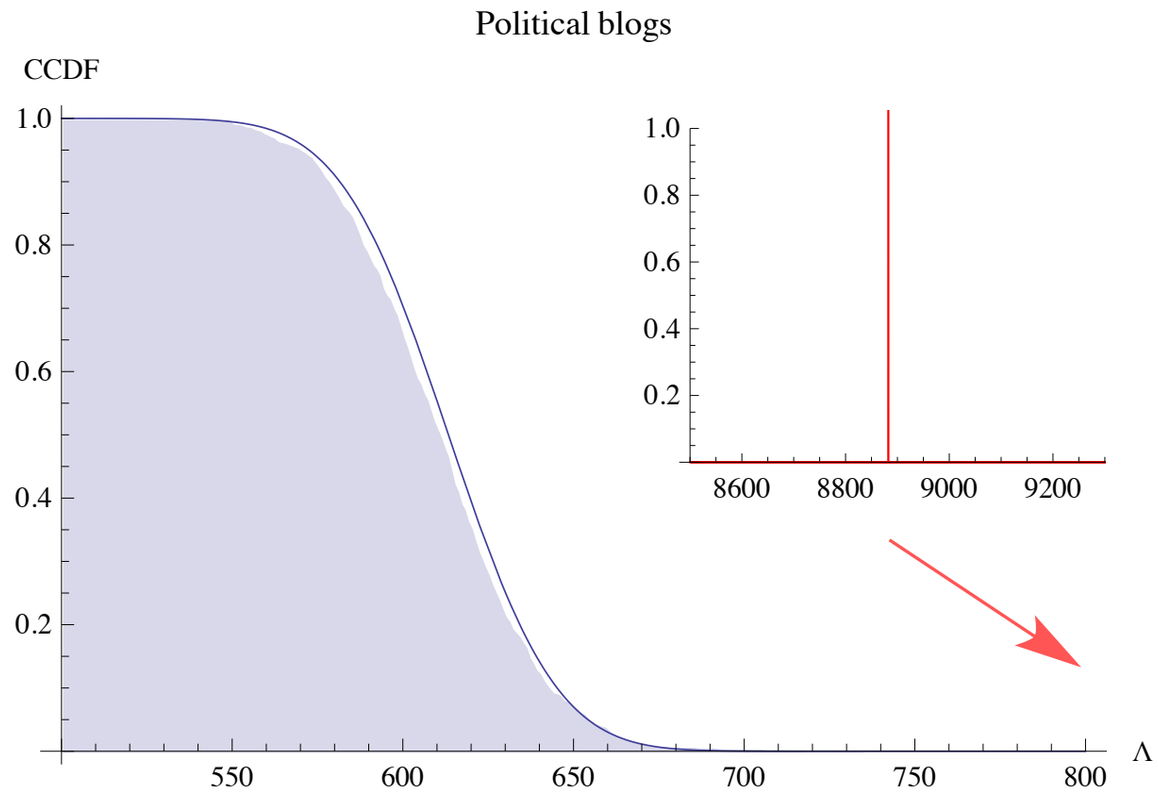
if *G* is dense and μ_v is large, the Poisson distribution looks Gaussian; but for sparse networks, it has a different shape

do the math!

Trying it out



p=0.15 (bootstrap)
p=0.19 (theory)



p=0.0

overwhelming evidence for the blog network; for the Karate Club, less so

[Yan, Jensen, Krzakala, Moore, Shalizi, Zdeborová, Zhang, Zhu 2012]

Dealing with uncertainty #1: predicting missing links

for many networks, links are discovered one at a time, using difficult work and limited resources in the field or laboratory

given the links observed so far, can we predict missing links?

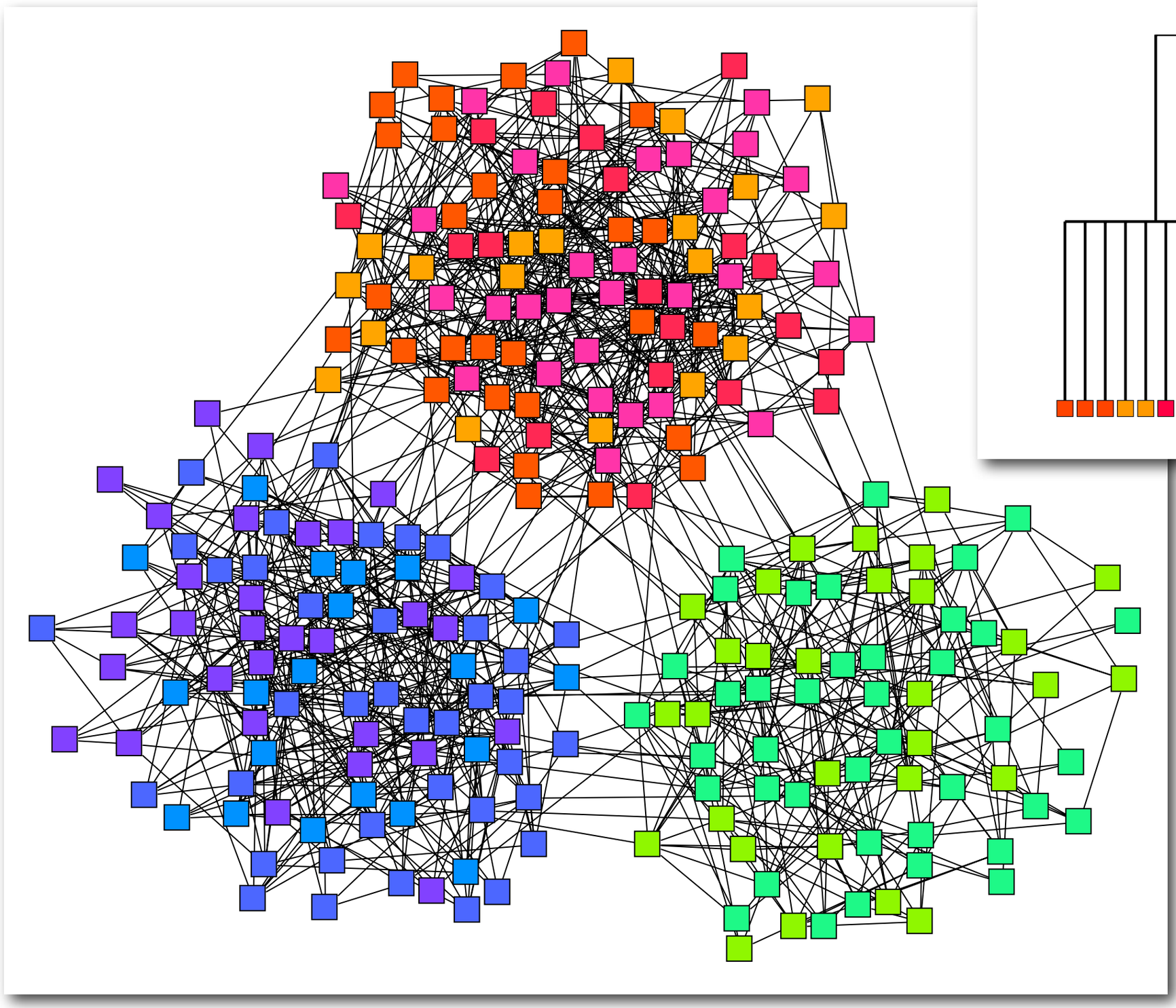
if there are spurious edges (false positives), can we identify them?

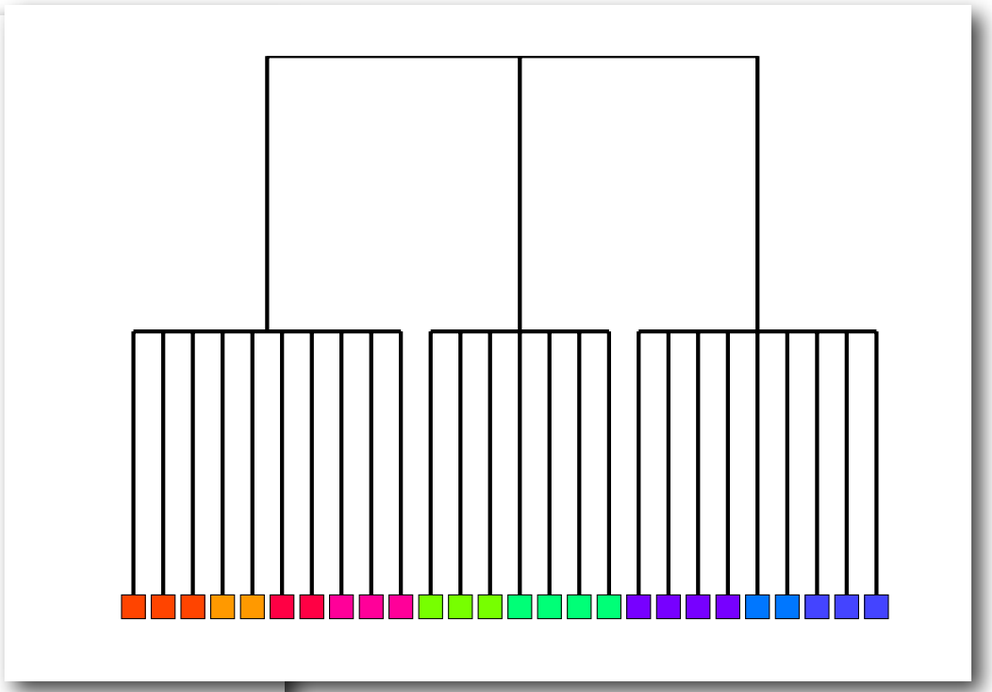
test the algorithm by hiding a random subset of edges from it, and ask it to rank possible missing links according to probability

can use the accuracy of prediction as another method of model selection

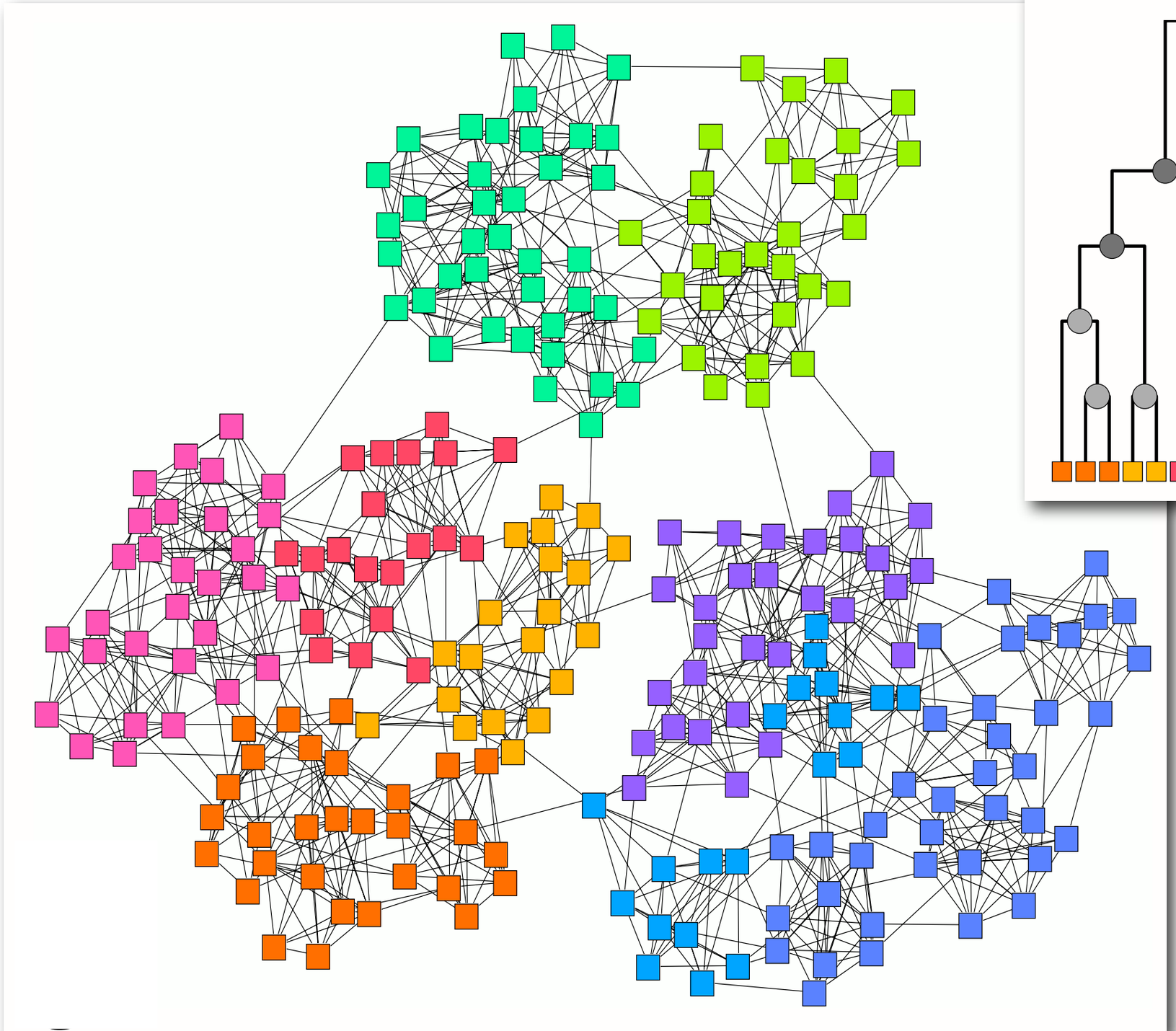
let’s try a particular model...

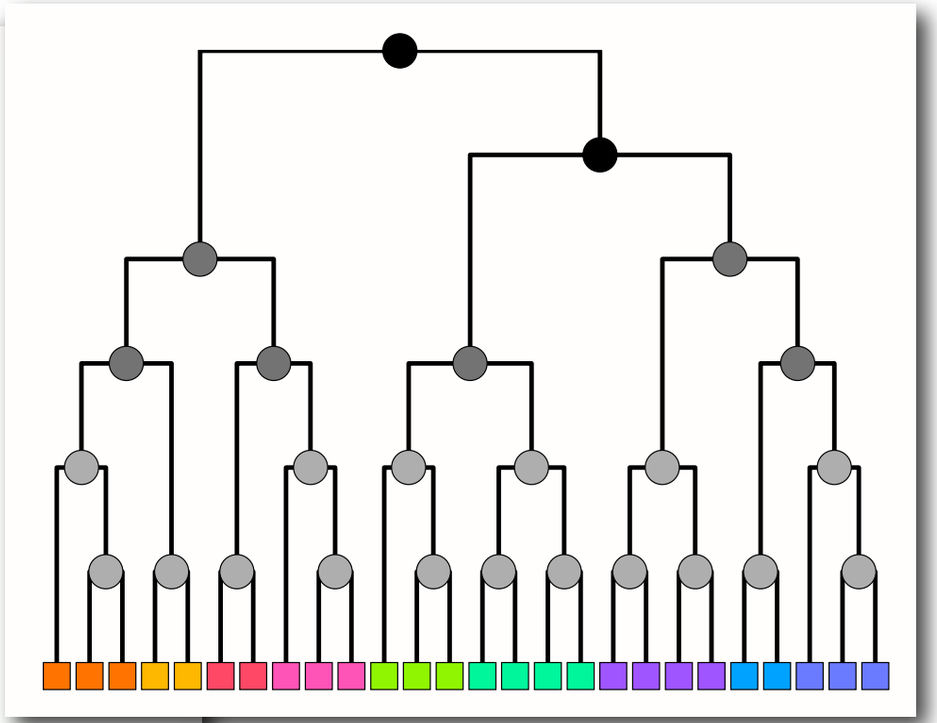
Clustering: one level



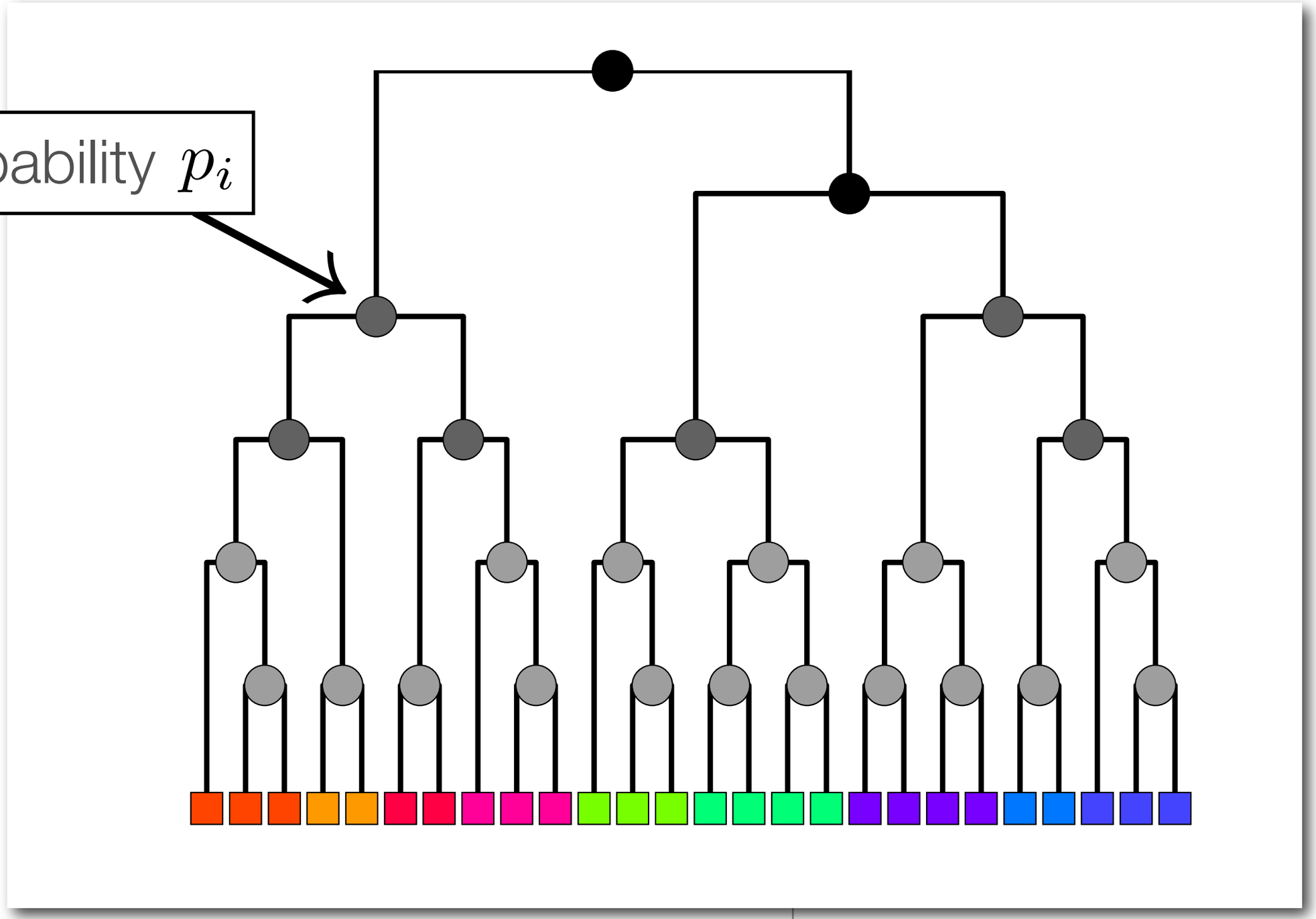


Hierarchy: many levels





A probabilistic model



Likelihood

For each internal node *i*, let

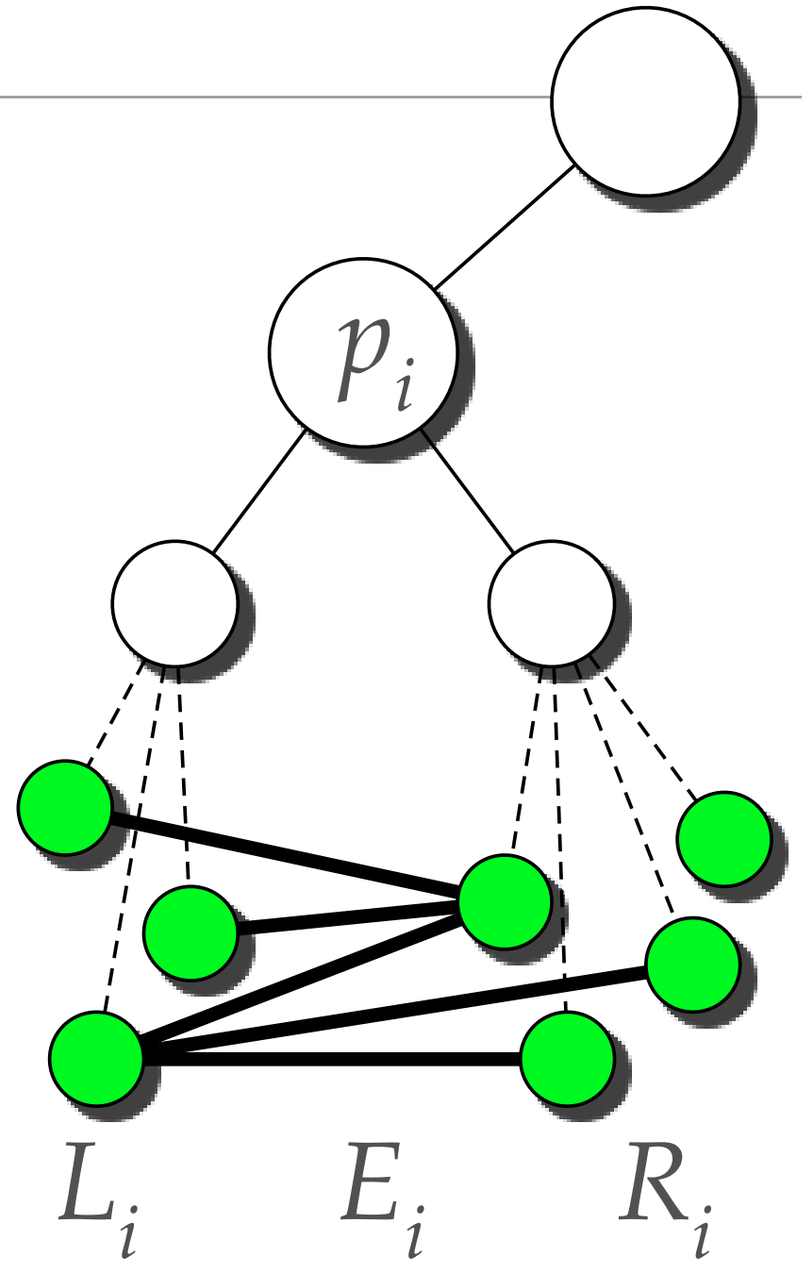
L_i and R_i = # of descendants

E_i = # of edges between them

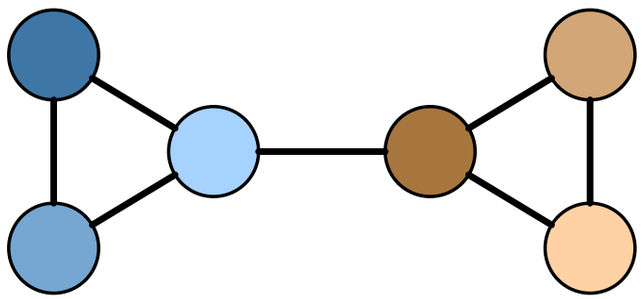
Likelihood these edges exist, and not others, is

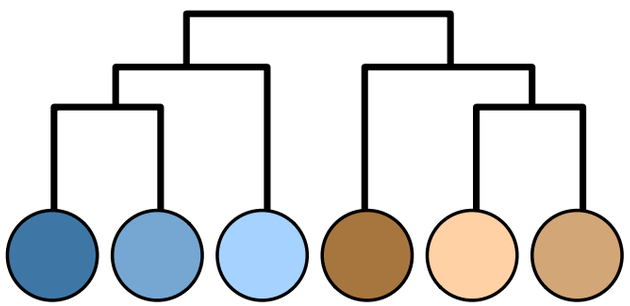
$$\mathcal{L}_i = p_i^{E_i}(1-p_i)^{L_i R_i - E_i}$$

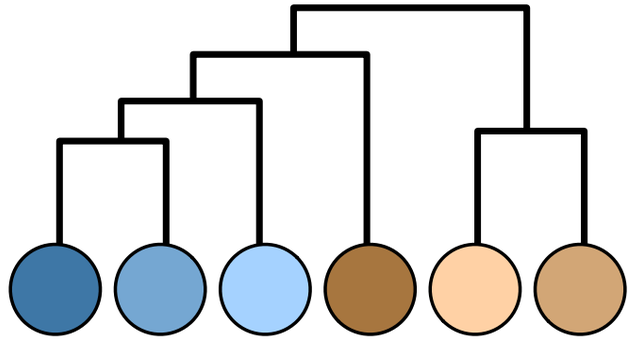
Overall likelihood is a product: $\mathcal{L}(T) = \prod_i \mathcal{L}_i$



Maximum likelihood trees







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

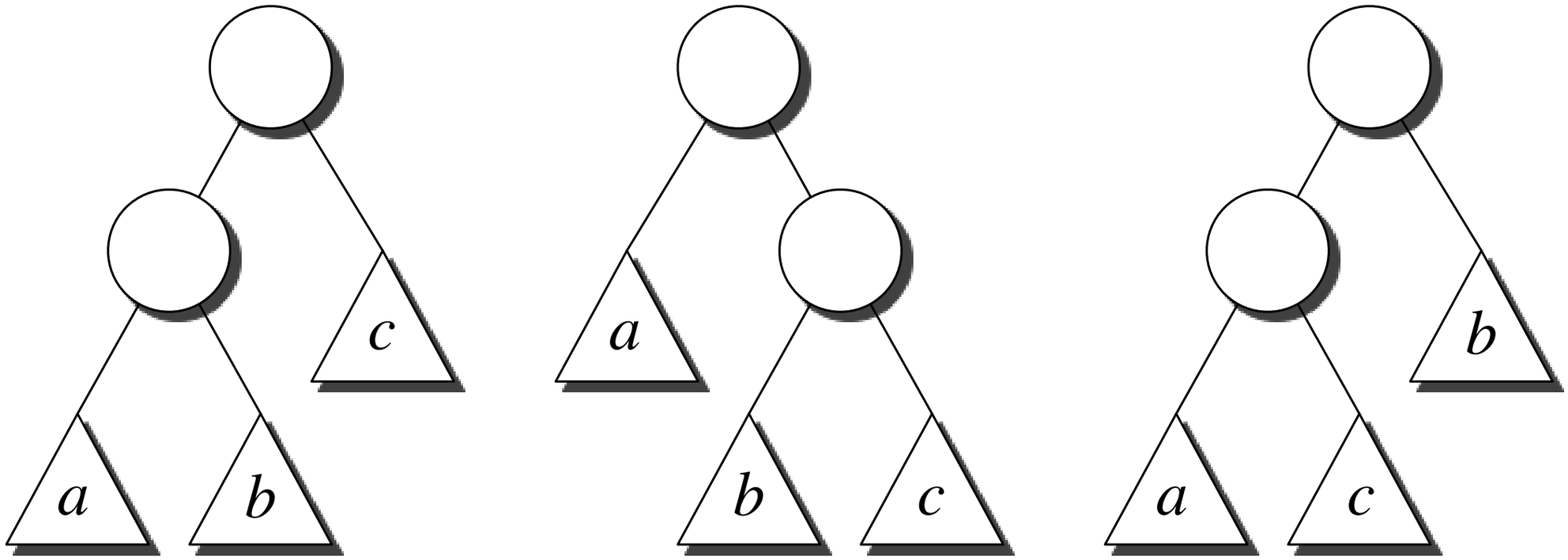
$$= 0.0433$$

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

$$= 0.0016$$

A Markov chain that explores the space of trees

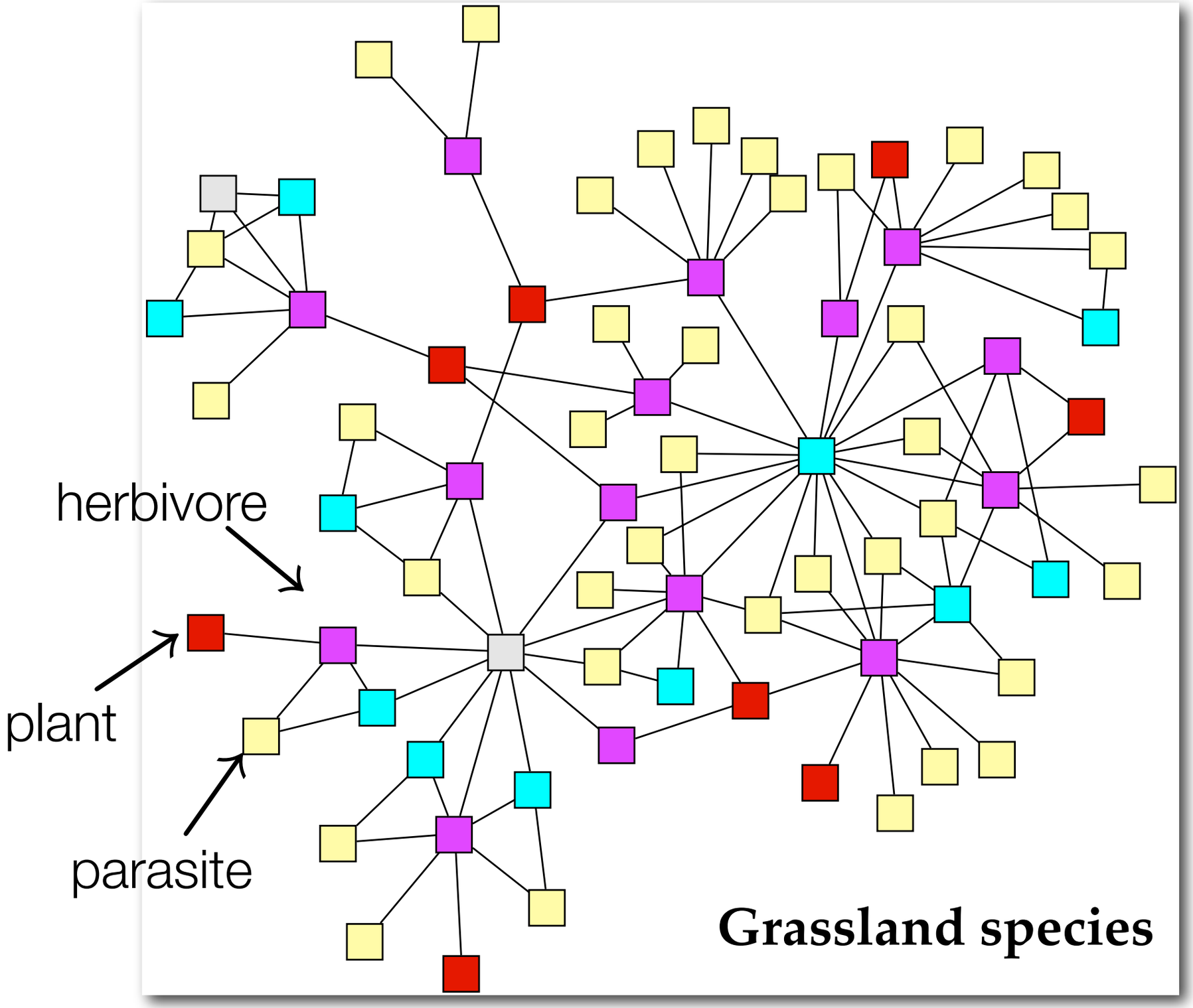
update the tree *T* with rotations, like in balanced tree data structures



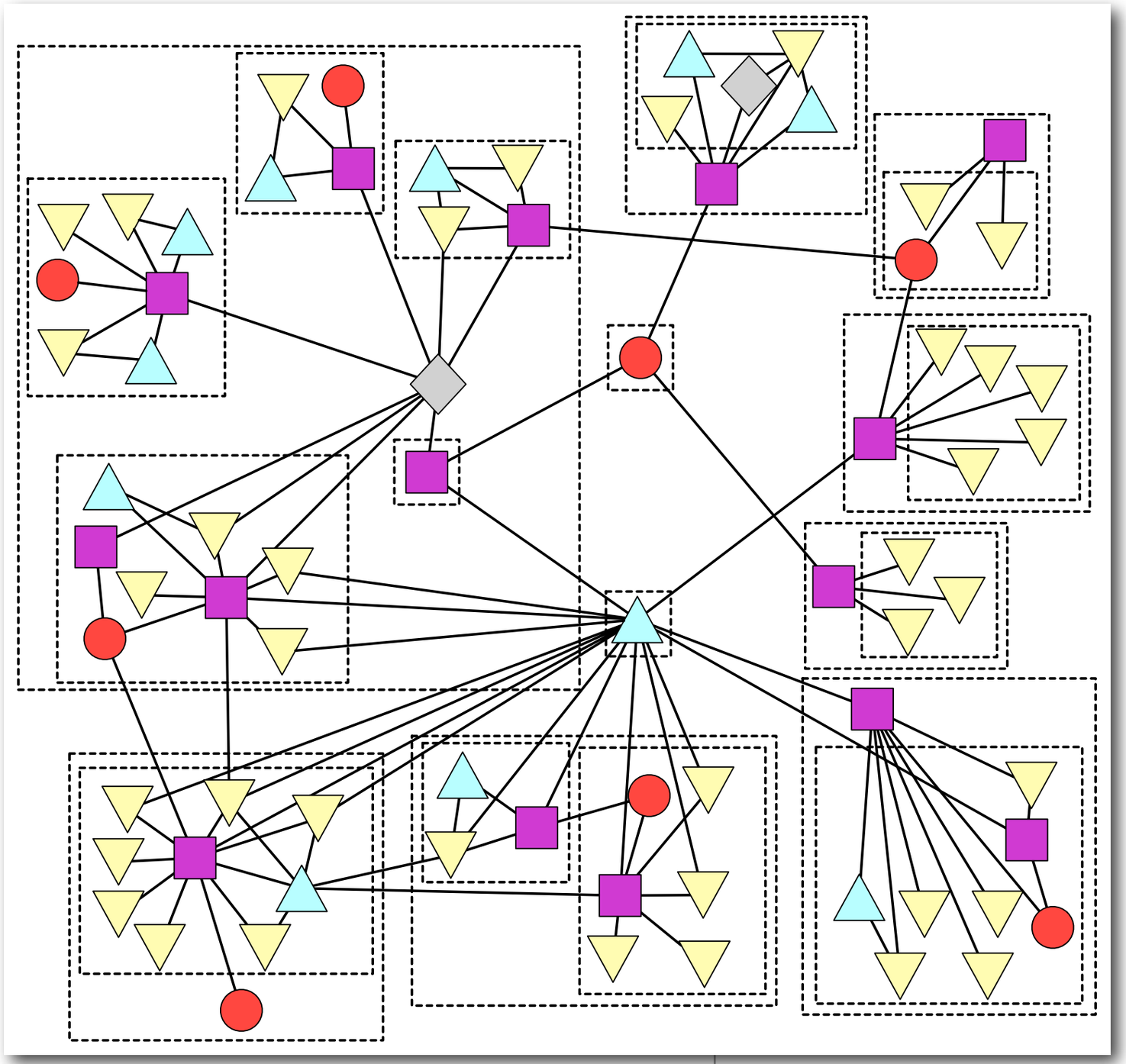
Metropolis Monte Carlo: move with probability 1 if $\Delta \log \mathcal{L} \geq 0$
and probability $\exp(\Delta \ln \mathcal{L}) = \mathcal{L}_{\text{new}}/\mathcal{L}_{\text{old}}$ otherwise

[Clauset, Moore, Newman, *Nature* 2008]

Functional roles in a food web

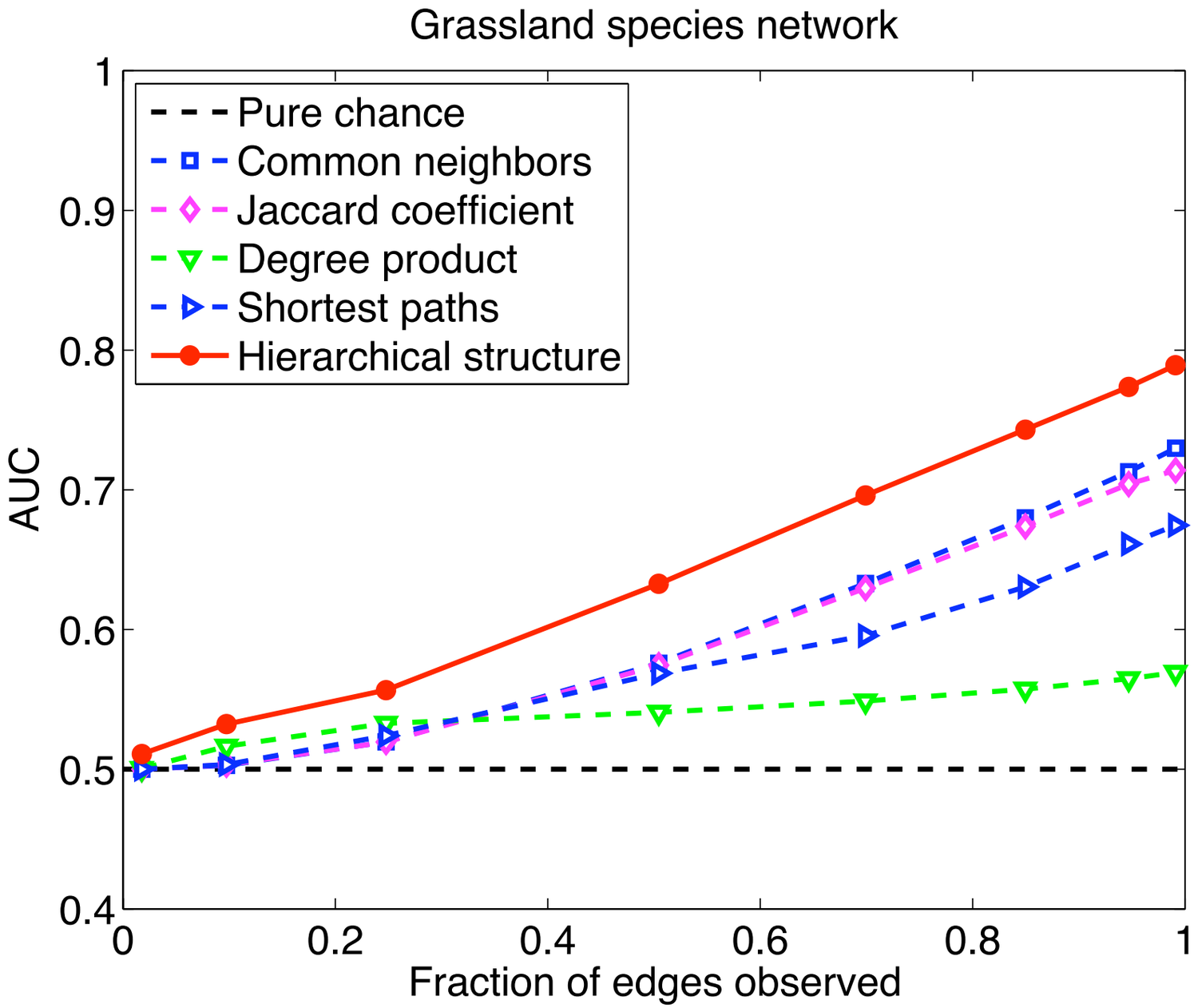


Functional roles in a food web



Predicting missing links: comparison with simple heuristics

AUC: probably a random true positive is ranked above a random true negative



Dealing with uncertainty #2: active learning of hidden node attributes

suppose we can learn a node's attributes, but at a cost

we want to make good guesses about most of the nodes, after querying just a few of them — but which which ones?

query the node with the largest *mutual information* between it and the others:

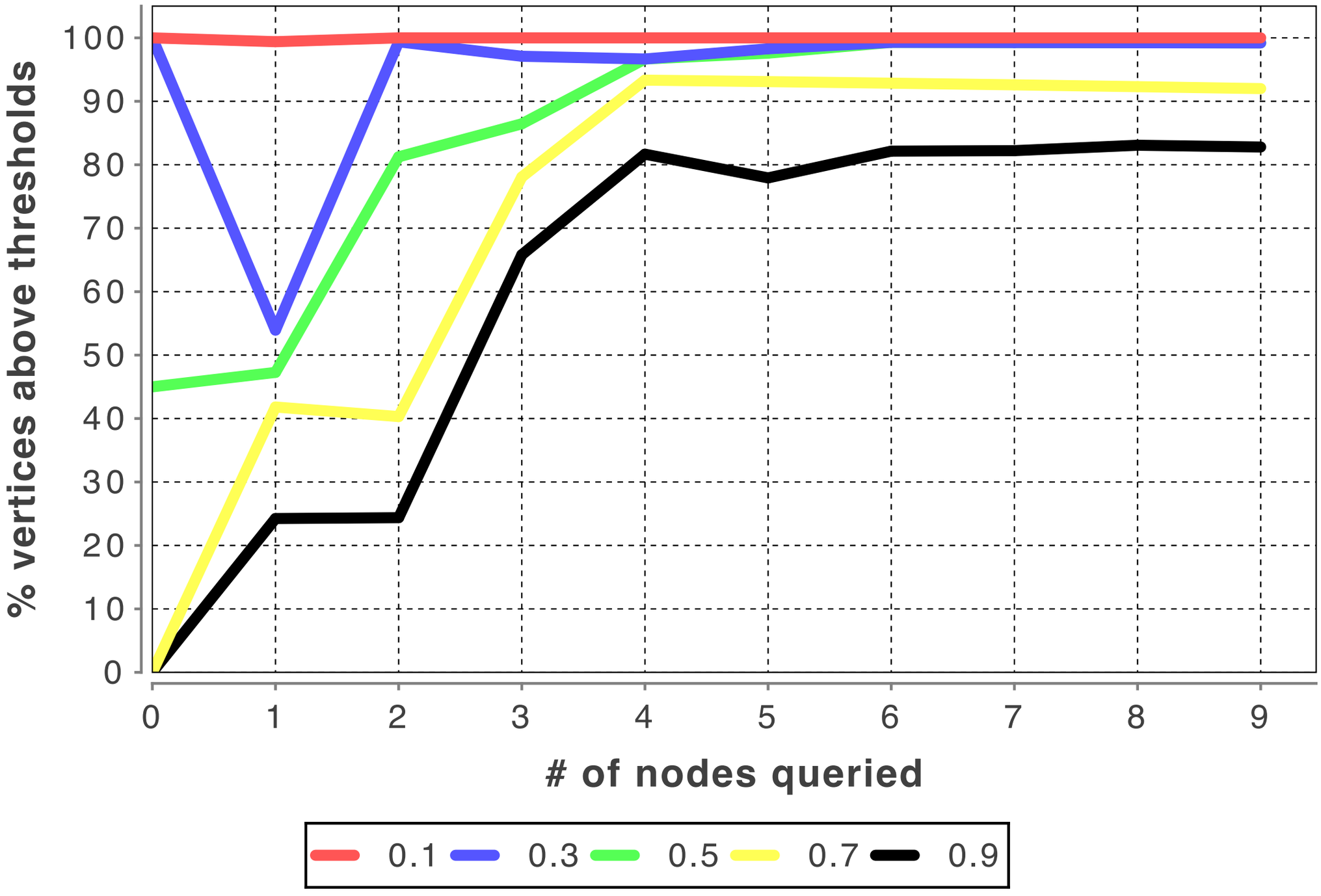
$$I(v, G-v) = H(v) - H(v \mid G-v)$$
$$= H(G-v) - H(G-v \mid v)$$

average amount of information we learn about *G–v* we learn by querying *v*

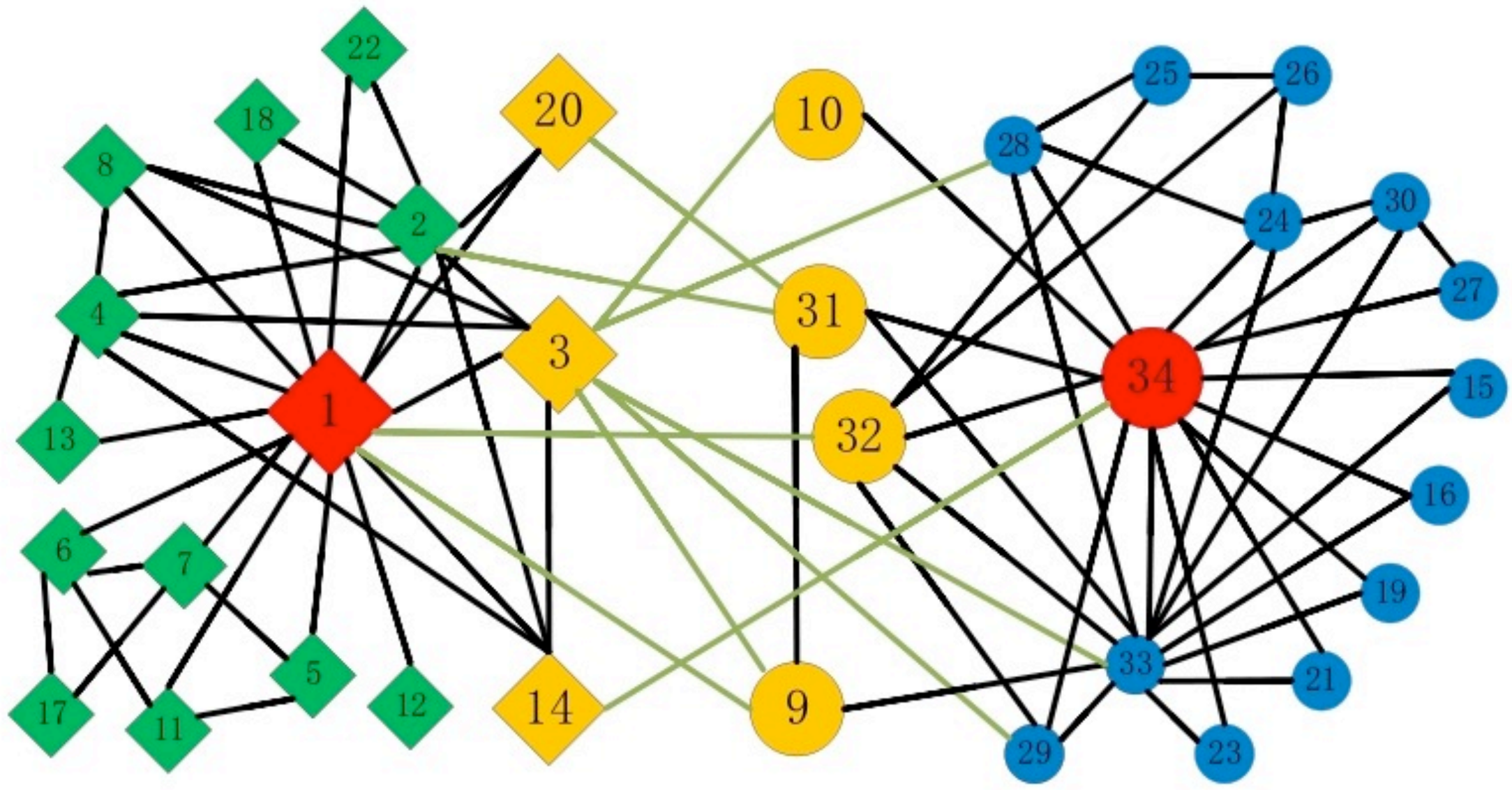
high when we're uncertain about *v*, and when *v* is highly correlated with others

[Moore, Yan, Zhu, Rouquier, Lane KDD 2011]

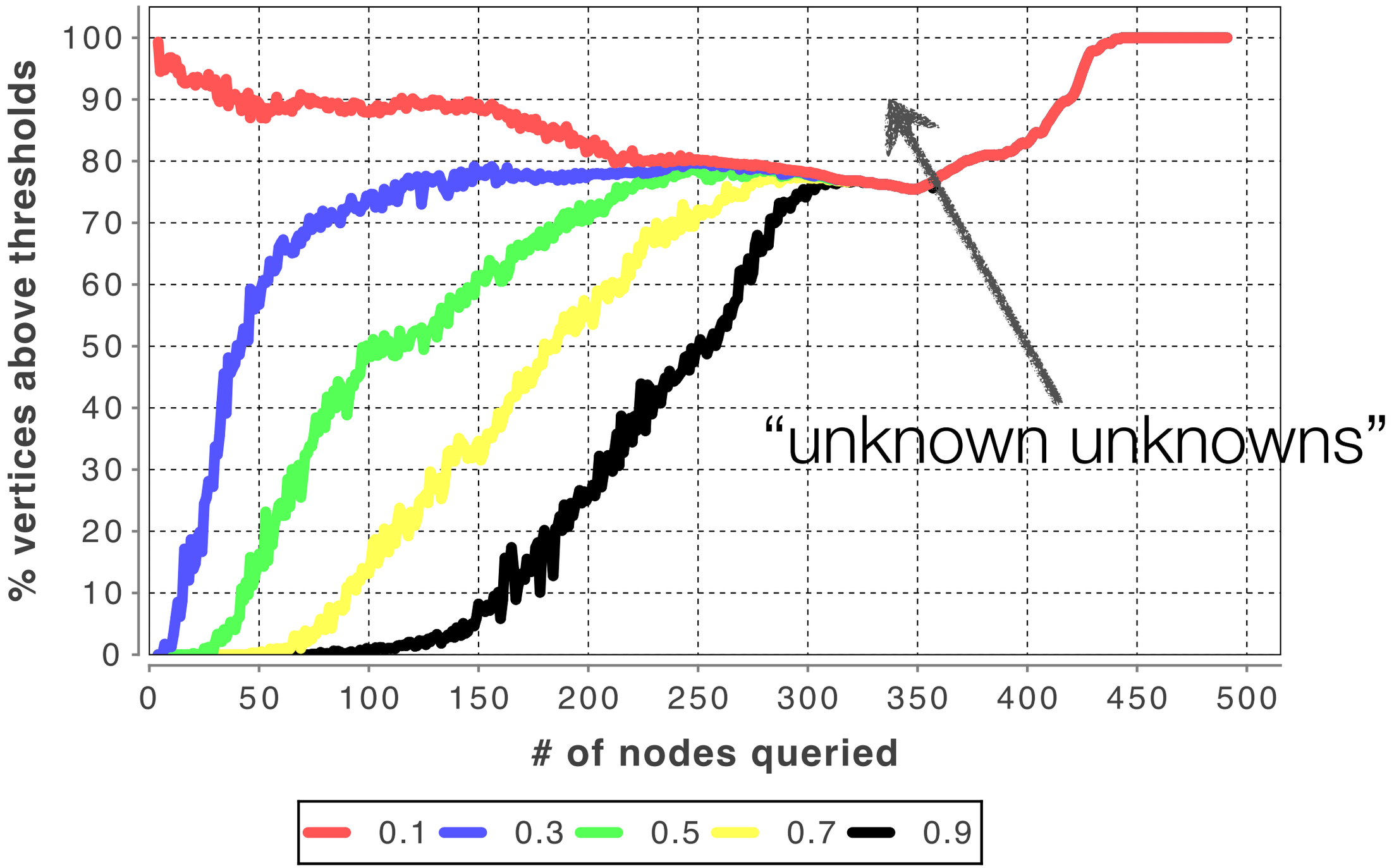
Learning factions in the Karate Club



Which vertices do we query first?



An antarctic food web



The story so far

Statistical inference using generative models of networks lets us detect communities, classify nodes, and predict missing links

Functional groups of nodes, not just assortative “clumps”

Belief propagation and expectation-maximization algorithms let us identify these groups, and learn model parameters, often in linear time: scalable!

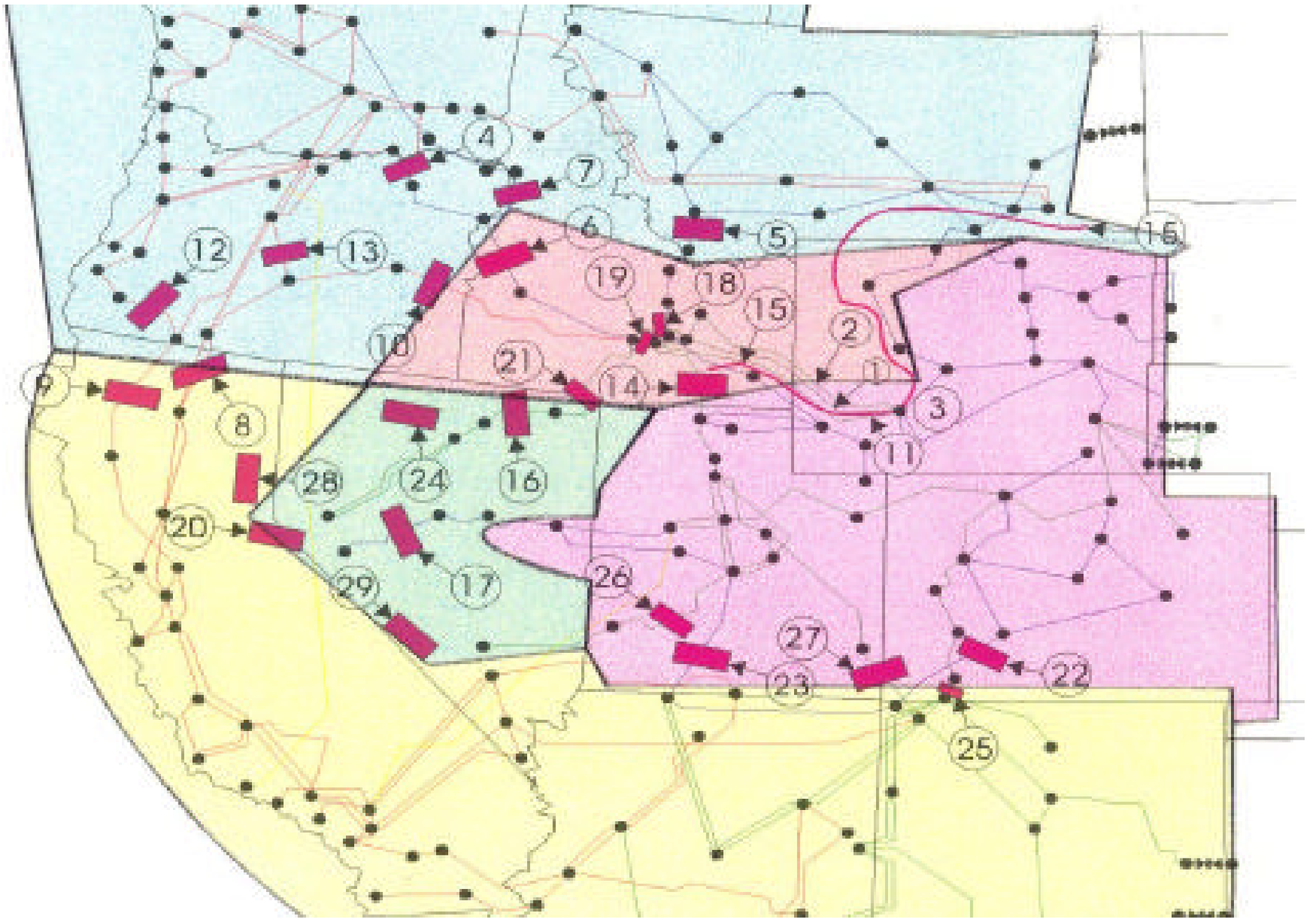
We can elaborate these models by adding discrete or continuous attributes: degree distributions, edge types, social status or niche positions, overlapping communities, hierarchy, signed edges, document content...

For instance, we can classify documents using their content and the links between them better than with content or links alone [Zhu, Yan, Getoor, Moore]

But a cautionary note...

A real cascade of line and generator failures

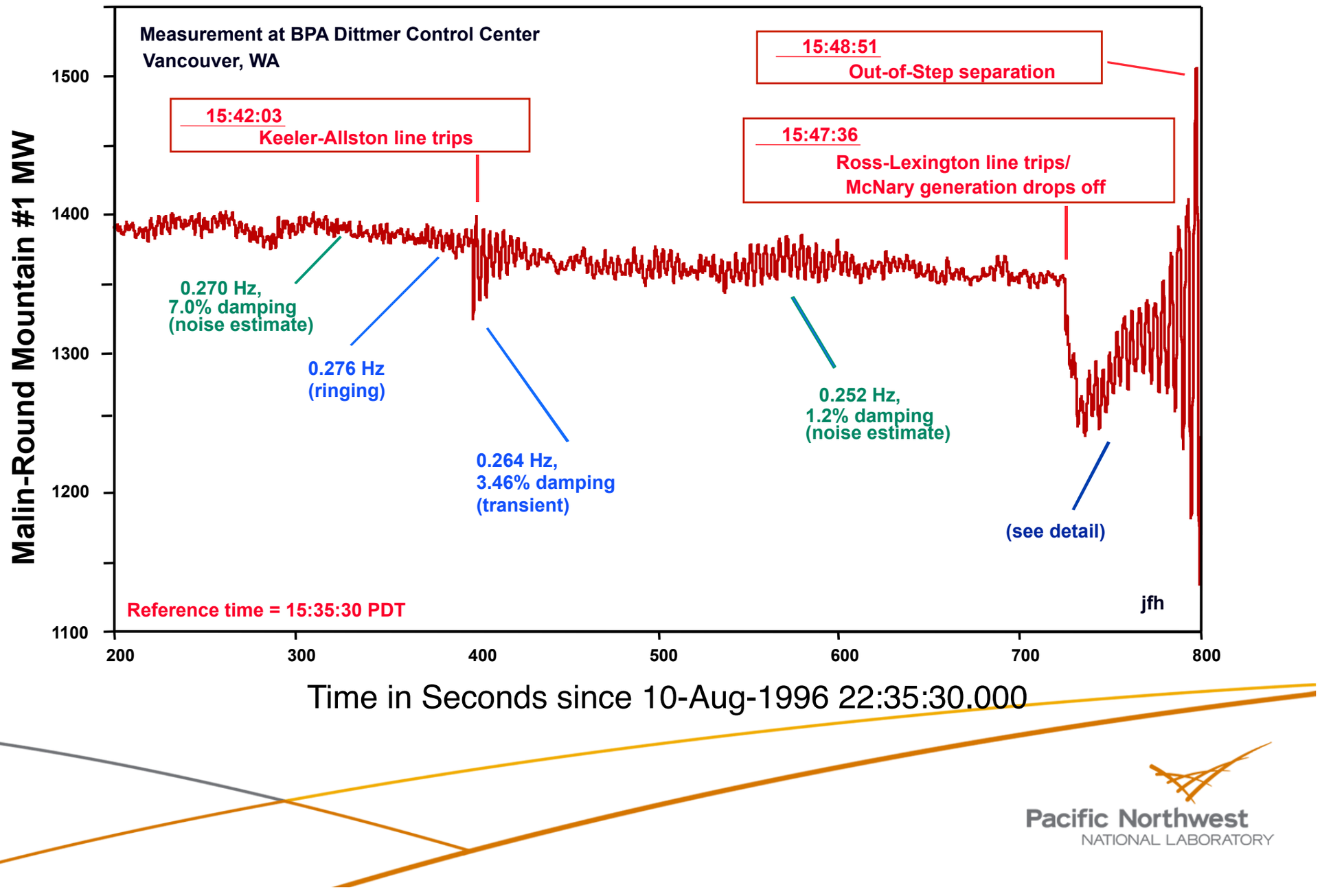
Sequence of outages in Western blackout, July 2 1996



from NERC 1996 blackout report

Rich dynamics of coupled, nonlinear oscillators

Sequence of Events



Pacific Northwest
NATIONAL LABORATORY

Beyond topology

We need a new network theory that doesn't focus on topology alone

Nodes and edges have rich attributes:

power grid: generators have nonlinear dynamics at many time scales, transmission lines have capacities, users have fluctuating demands...

cybersecurity: multiple types of links between computers (web fetches, SSH links) with timing, duration, packet size... and many links are unique

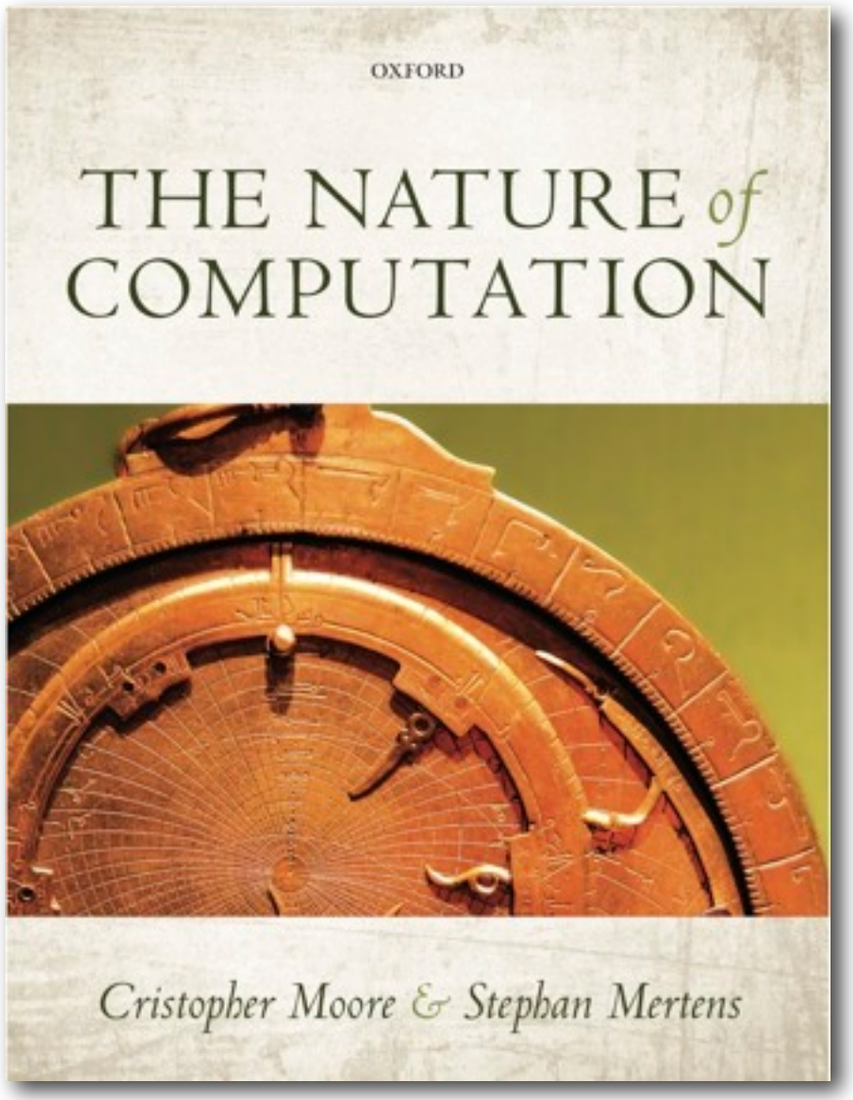
food webs: species have populations, links have nutrient flows.... dynamic response to climate change, species loss, invasive species

Networks are rich, dynamic data sets, not just lists of nodes and edges

Extending Bayesian inference to richer data is possible, but challenging

We need to be agnostic about what types of structure are important

Shameless Plug



www.nature-of-computation.org

To put it bluntly: this book rocks! It somehow manages to combine the fun of a popular book with the intellectual heft of a textbook.

Scott Aaronson, MIT

A creative, insightful, and accessible introduction to the theory of computing, written with a keen eye toward the frontiers of the field and a vivid enthusiasm for the subject matter.

Jon Kleinberg, Cornell

A treasure trove of ideas, concepts and information on algorithms and complexity theory. Serious material presented in the most delightful manner!

Vijay Vazirani, Georgia Tech

A fantastic and unique book, a must-have guide to the theory of computation, for physicists and everyone else.

Riccardo Zecchina, Politecnico de Torino

This is the best-written book on the theory of computation I have ever read; and one of the best-written mathematical books I have ever read, period.

Cosma Shalizi, Carnegie Mellon

Acknowledgments

and the McDonnell Foundation, DARPA/AFOSR, and the NSF