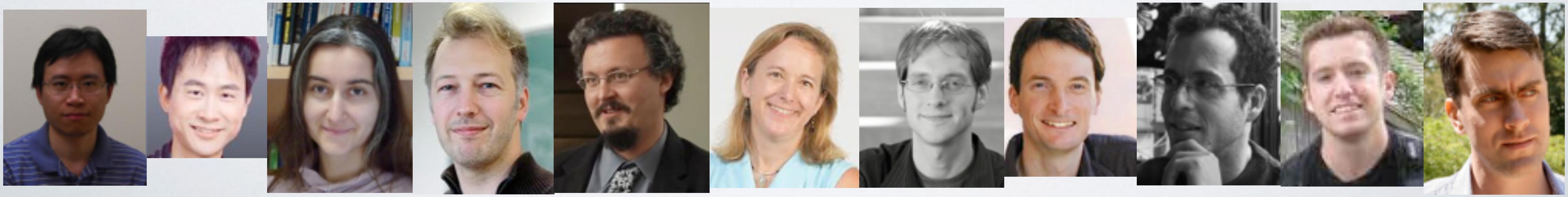
# Big Data Needs Smart Algorithms: Or, How Physics can Help Us Explore Social Networks

Cristopher Moore, Santa Fe Institute

joint work (over the years) with
Xiaoran Yan,Yaojia Zhu, Lenka Zdeborová, Florent Krzakala, Aurelien Decelle, Pan Zhang, Cosma Shalizi, Jacob Jensen, Jean-Baptiste Rouquier, Tiffany Pierce, Lise Getoor, Aaron Clauset, Mark Newman, Elchanan Mossel, Joe Neeman, and Allan Sly



# Big data needs big algorithms

We're swimming in data: the challenge is doing something with it

Finding simple trends isn't enough: we need to find structures and patterns in this data, that let us

understand it

predict it

generalize from what we know to what we don't

We need algorithms that do this automatically (often with a human in the loop)

Algorithms need to be scalable: $n^2$ or $n^3$ time on a data set of size $n$ is too slow if $n$=1 million

Not just computing power! Moore's law isn't enough—we need mathematical insight to avoid/simplify the search

# What is structure?

Structure is that which...

makes data different from noise: makes a network different from a random graph, from a background “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

# Statistical inference

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

Use whatever (partial, noisy) information we have to constrain the search...

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)

...and make good guesses about the information we don't have:

label unknown nodes

predict missing links

identify anomalies

# The stochastic block model

*k* types of nodes

we know the links between the nodes, but not their types

assumption: probability that two nodes are linked depends only on their types

assortativity / homophily: nodes connect more to others of the same type

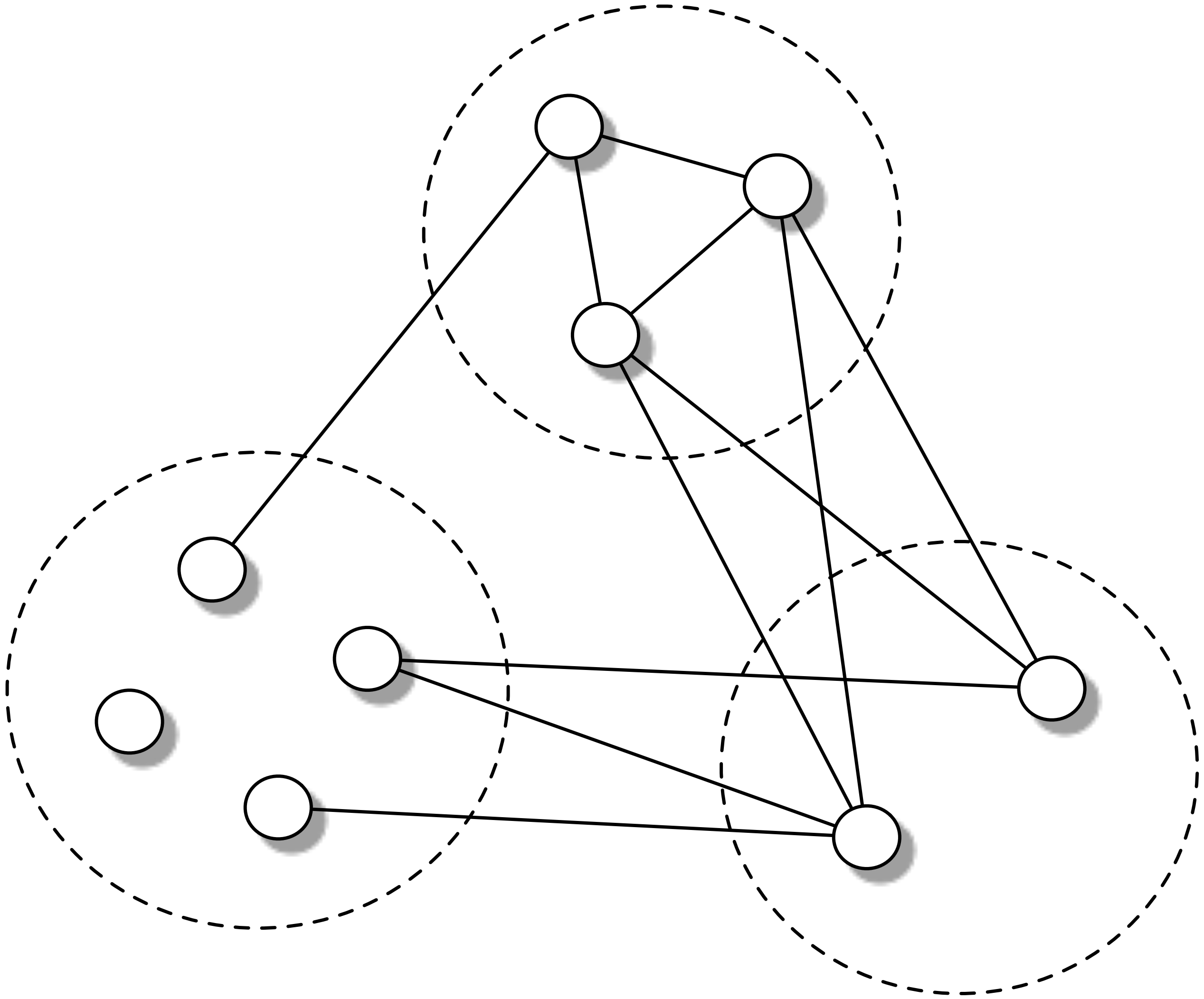
disassortativity / heterophily: links between types instead of within

directed: links from i→j but not j→i

given a network, we want to simultaneously...

> label the nodes with their types
>
> learn how the types affect the probability of a link

# Assortative and disassortative



functional groups, not just clumps

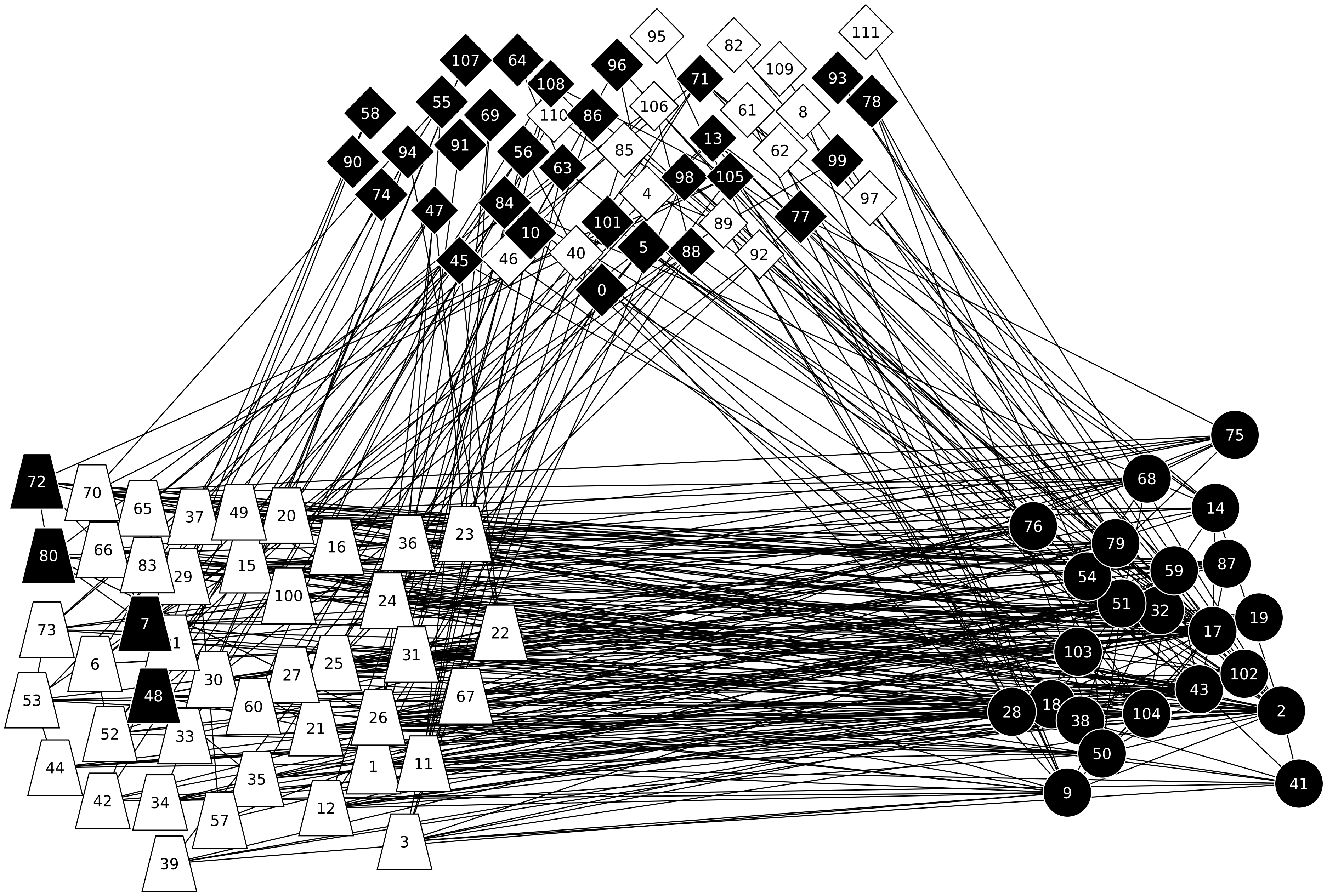
food webs: predators and prey

economics: suppliers and customers

word adjacencies: adjectives and nouns

social: leaders and followers

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



# A little statistical physics

each labeling of the nodes is a “state”, like orientations of atoms in a magnet

If there are *k* types, there are $k^n$ possible states

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

$$P \propto \mathrm{e}^{-E}$$

so the “energy” of a state in the block model is

$$E = -\log P$$

most likely labeling = “ground state” with lowest energy.

“free energy” = log of total likelihood of the model

entropy = how uncertain we are about the labeling

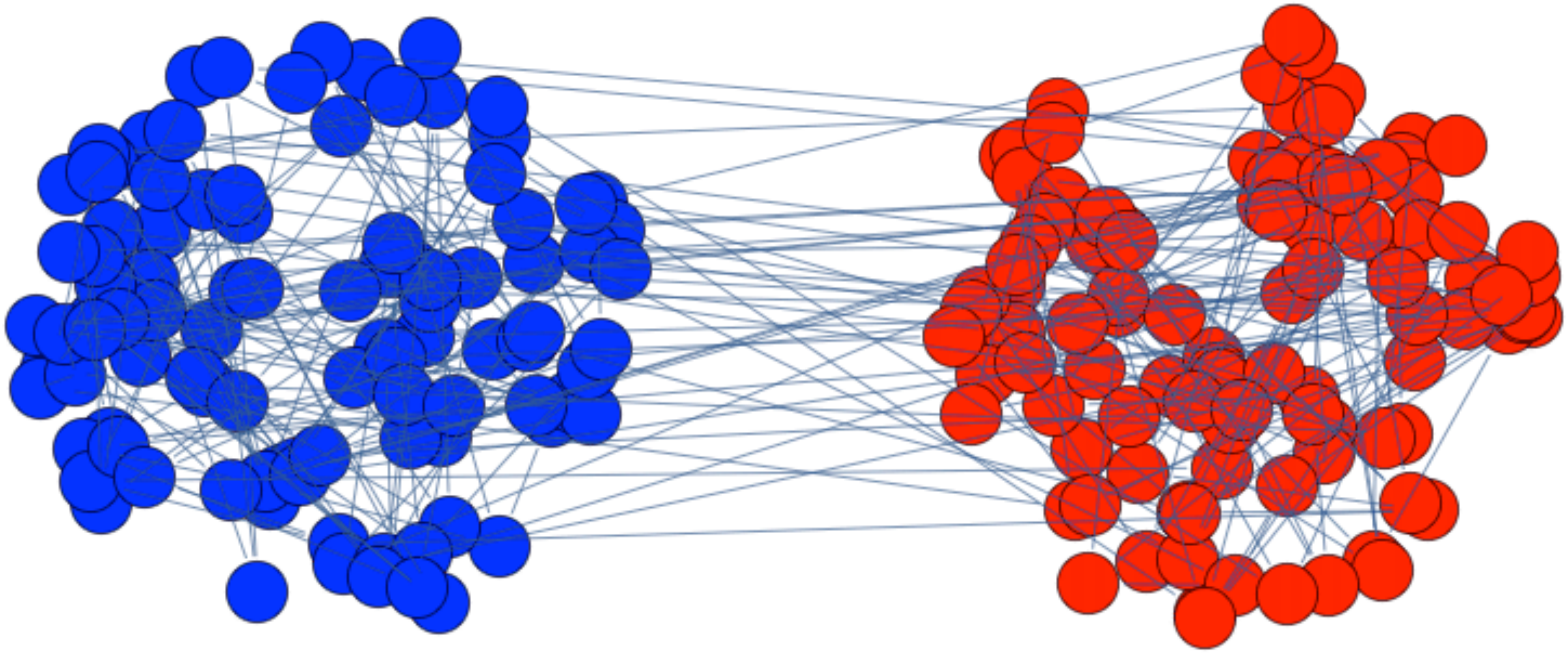
# What’s the best labeling?

the most likely group assignment, or MAP (maximum a posteriori) estimate, is the *ground state*: it maximizes $P(G|t)$

but there are good-looking ground states even when there no real communities!

e.g. random 3-regular graphs have bisections with only 11% of the edges crossing the cut [Zdeborová & Boettcher]

indeed, there are many such bisections, and they have nothing in common!

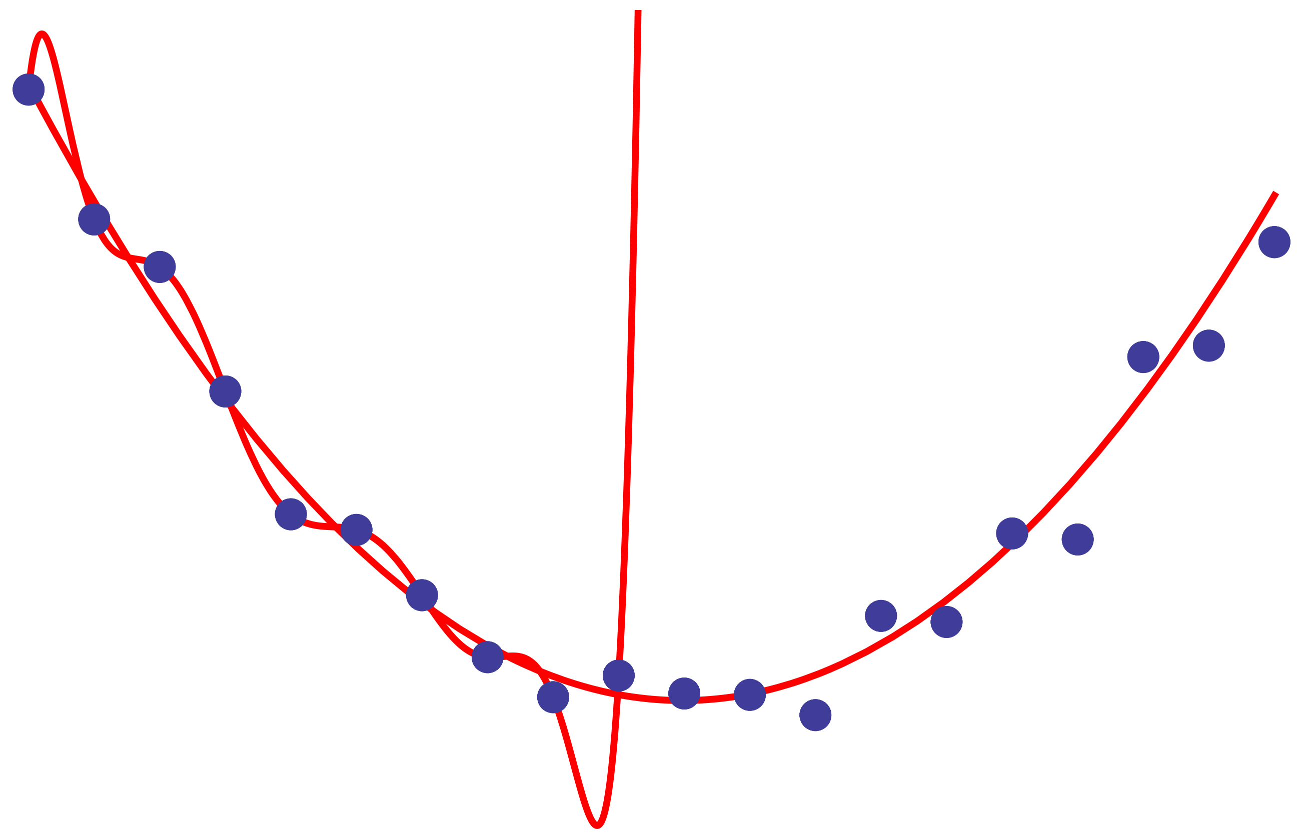


# Statistical significance

we don’t just want the best fit!

random graphs have illusory communities, that only exist because of noise

sometimes the patterns we find aren’t really there:



we want to understand the coin, not the coin flips

# What’s the best labeling, redux

a better approach: for each vertex, compute its *marginal distribution*, i.e., the probability that it is of each type

assign each node to its most-likely label

this achieves a higher “overlap” with the true labeling than the ground state:
it maximizes the expected fraction of nodes labeled correctly
(as opposed to the probability that they are all correct)

if communities are really there, marginals represent clusters of many solutions that agree on most nodes...

if they’re not really there, likely labelings are uncorrelated with each other,
and the marginals are uniform (or they fail to converge, à la spin glasses)

**the consensus of many likely solutions is better than the most-likely one**

# Optimization vs. robust fits

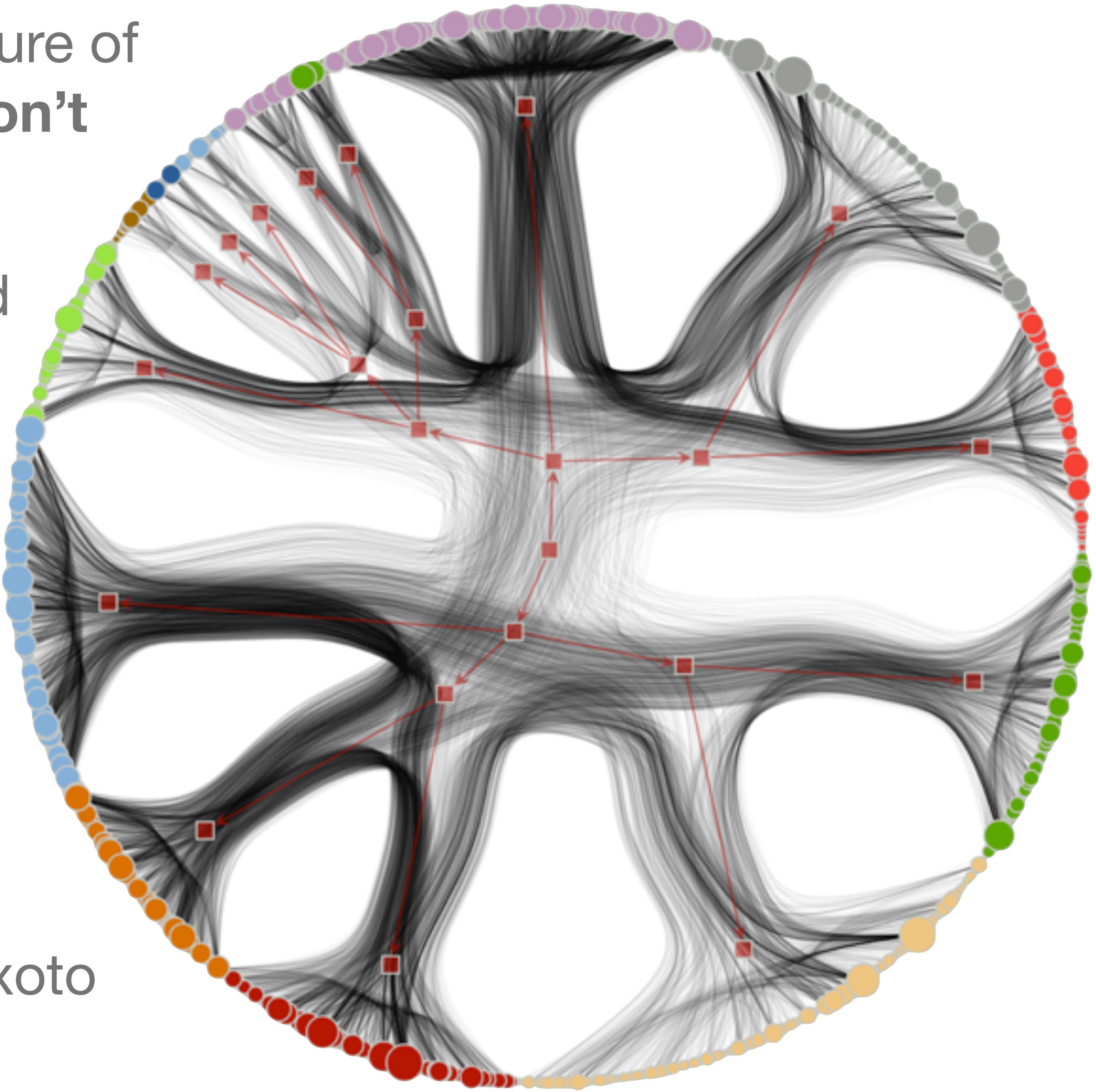
Modularity is a popular measure of community structure... **but don't optimize it!**

The consensus of many good solutions is better than the "best" single solution

If no consensus exists, there's nothing really there

Reveals substructure in a network of political blogs

[Zhang and Moore, 2014]
Image by Tiago de Paula Peixoto



# Method #1: Markov Chain Monte Carlo

update the labels one node at a time

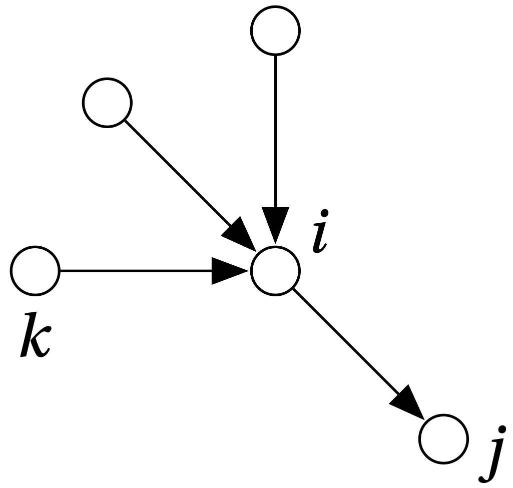
- choose a random node *v*
- fix types of all other nodes
- update *v*'s type according to its neighbors and the link probabilities

can speed up by introducing a temperature parameter:

- simulated annealing
- parallel tempering

but to obtain free energies or soft labels requires many independent samples: too slow!

# 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 probability distribution of types based on its other neighbors *k*

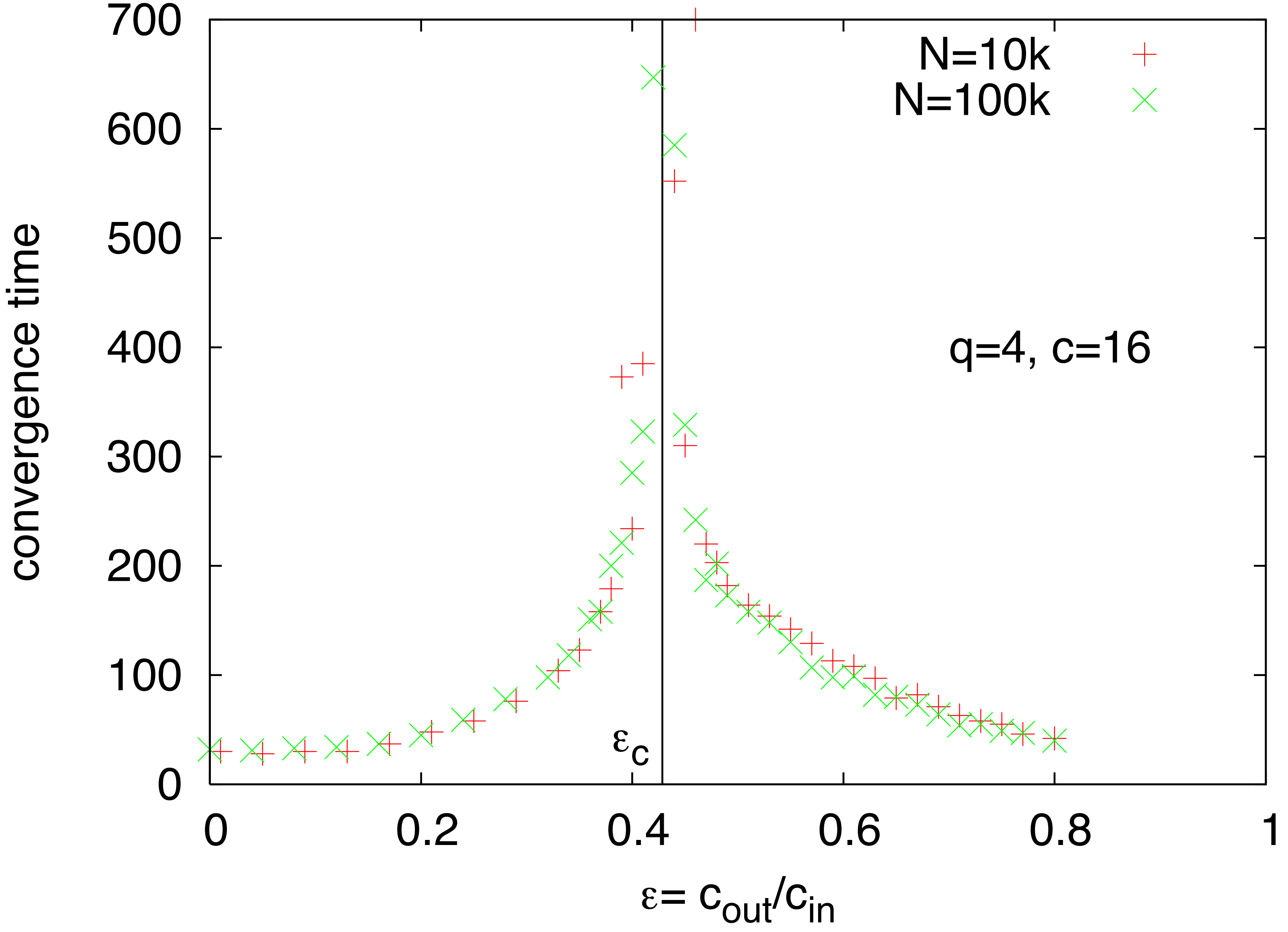
avoids the “echo chamber”

update messages, assuming that *i*’s neighbors are independent of each other...

each update takes O(*n*+*m*) time: iterate until we reach a fixed point

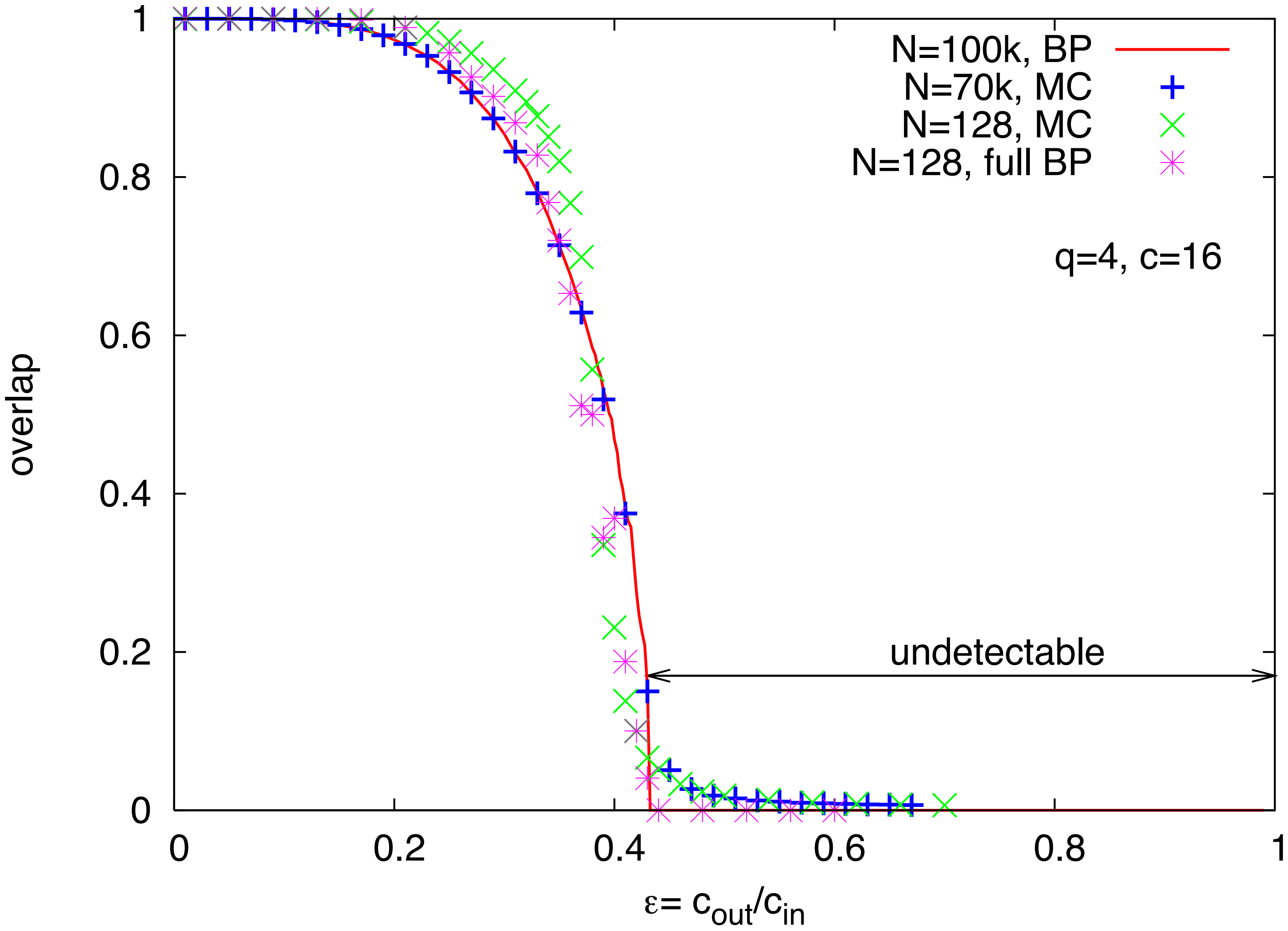
how long does it take to converge? does it give us good info when it does?

# BP convergence: nearly size-independent, but with critical slowing down at a phase transition



[Decelle, Krzakala, Moore, Zdeborová]

# A phase transition: undetectable communities

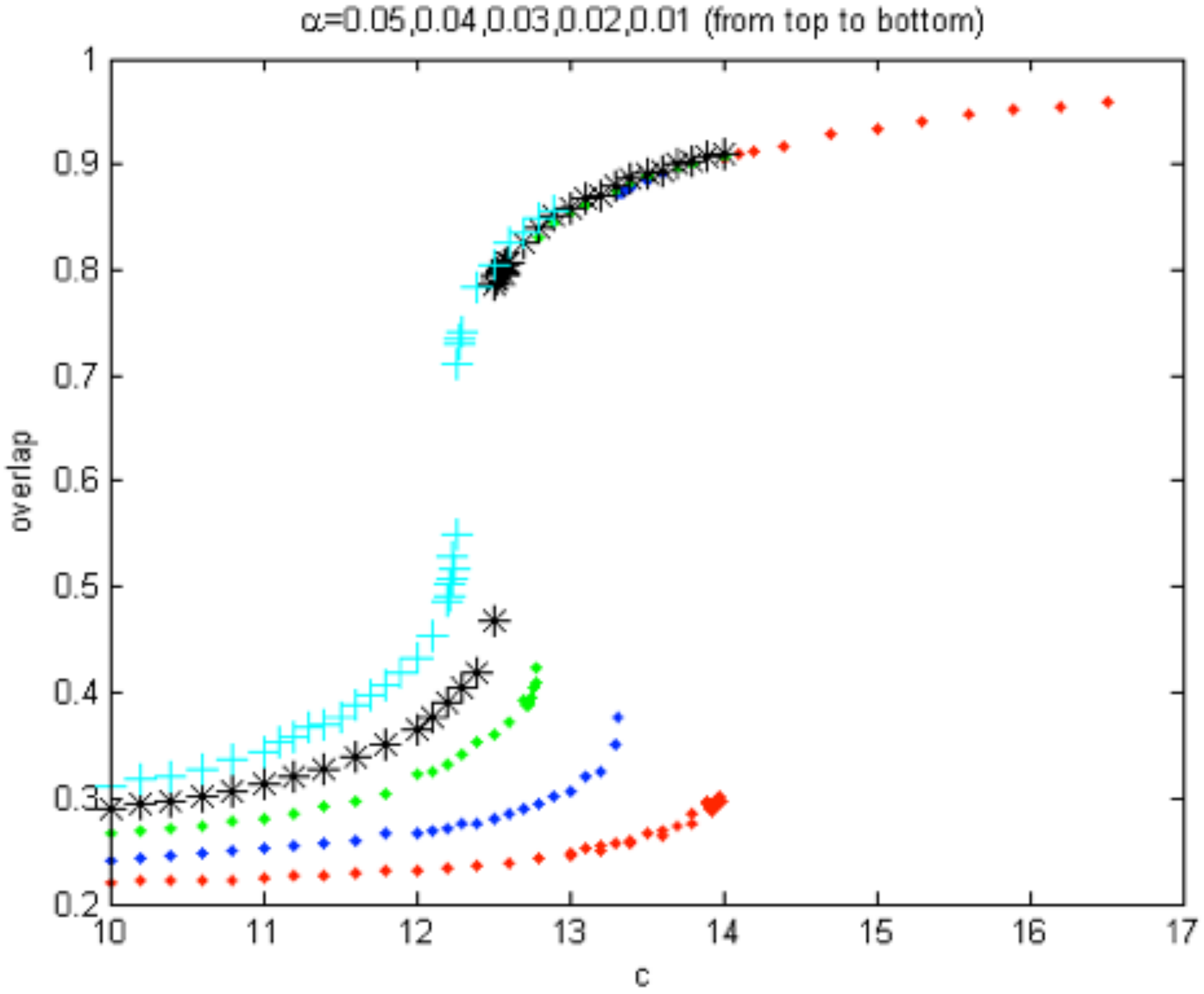


if the link probabilities are different enough (e.g. more links within groups than between groups) then we can find the communities, efficiently and accurately...

but there is a point beyond which no algorithm can!

[Decelle, Krzakala, Moore, Zdeborová; Mossel, Neeman, Sly]

# Another phase transition: Generalizing from known nodes to unknown ones

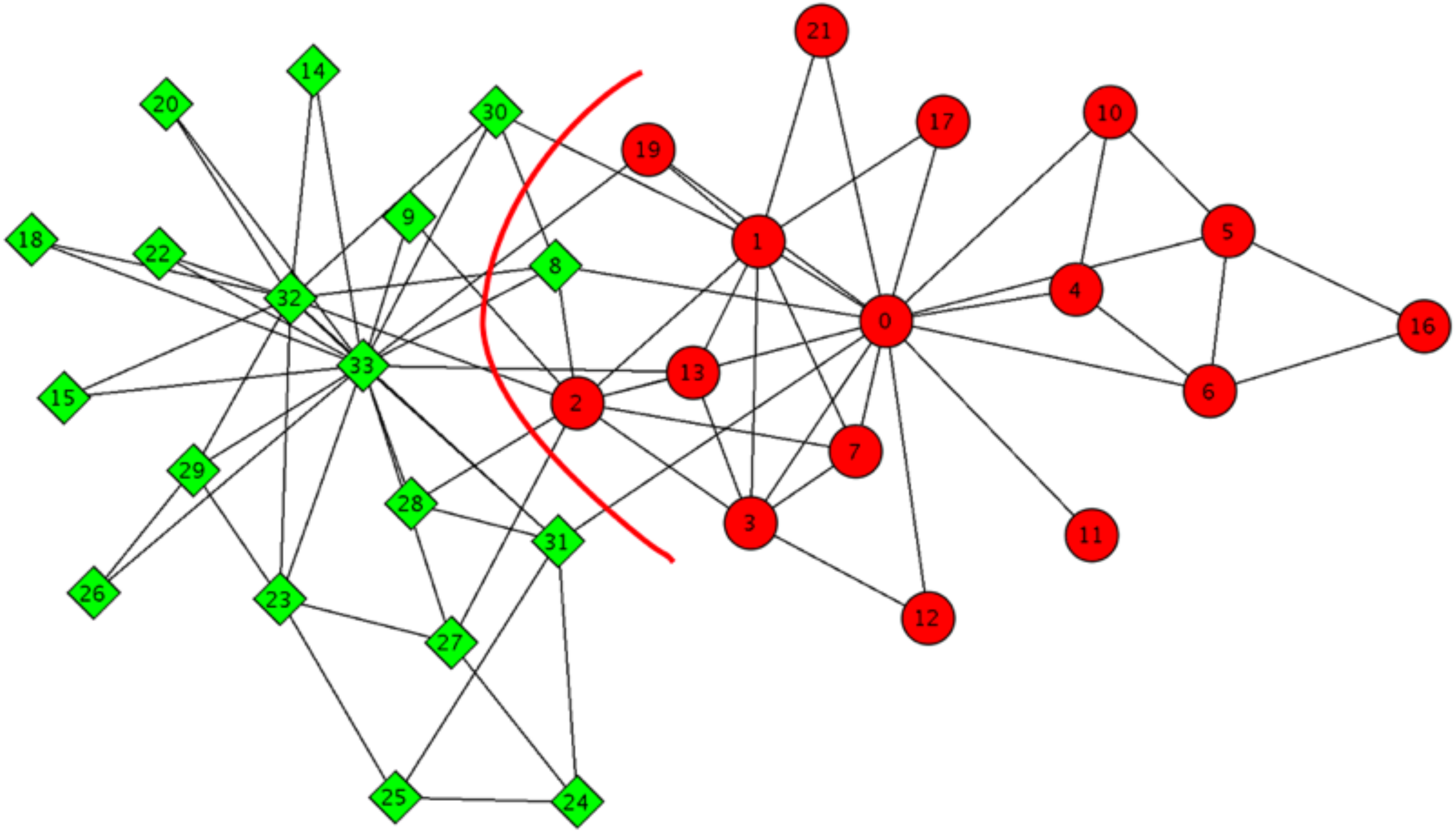


suppose we are initially given the correct types for a fraction $\alpha$ of the nodes. can we use this information to label the rest of the nodes?

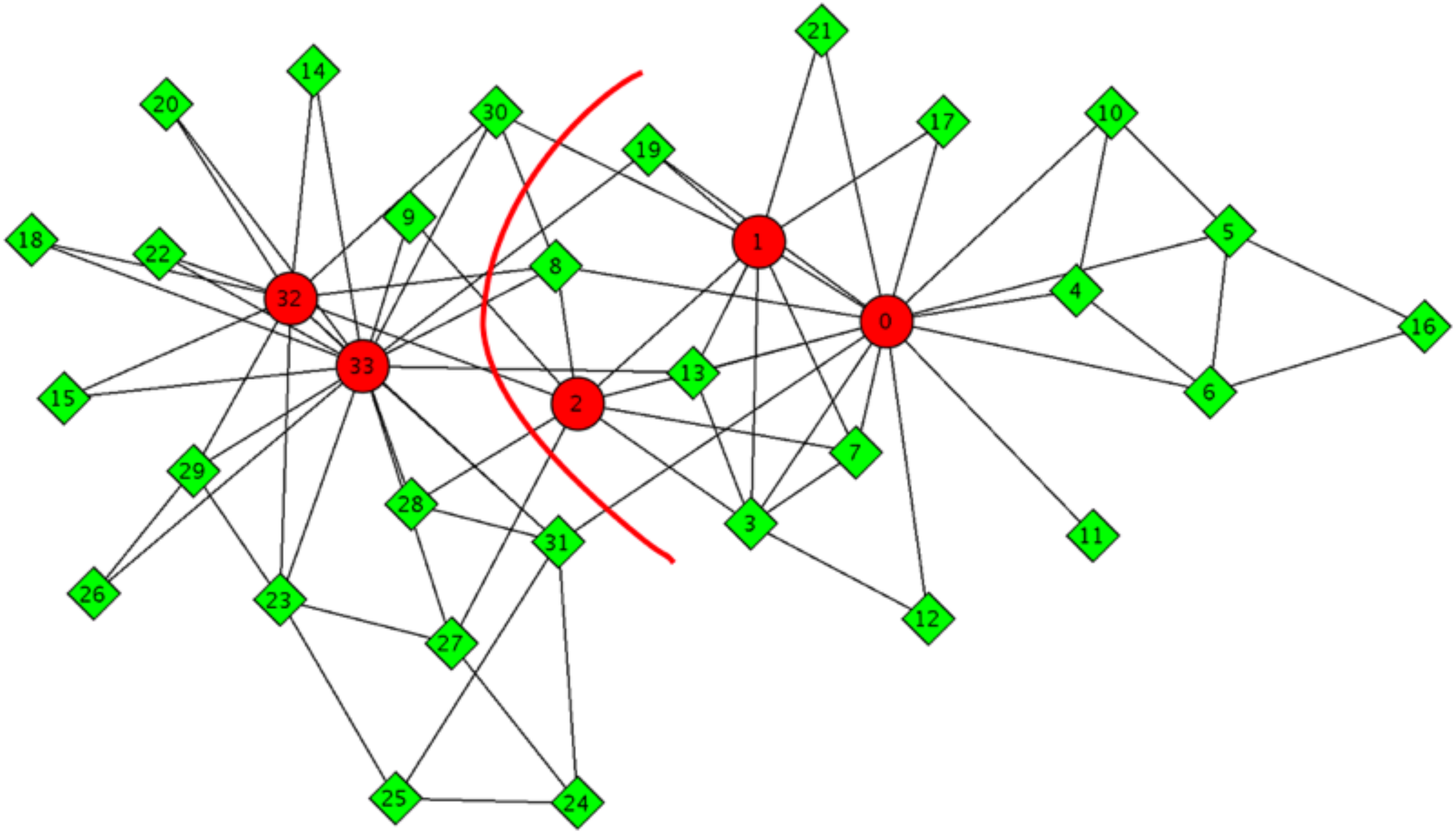
when $\alpha$ crosses a threshold, knowledge percolates throughout the network, causing a discontinuous jump in the accuracy

[Zhang, Moore, Zdeborová]

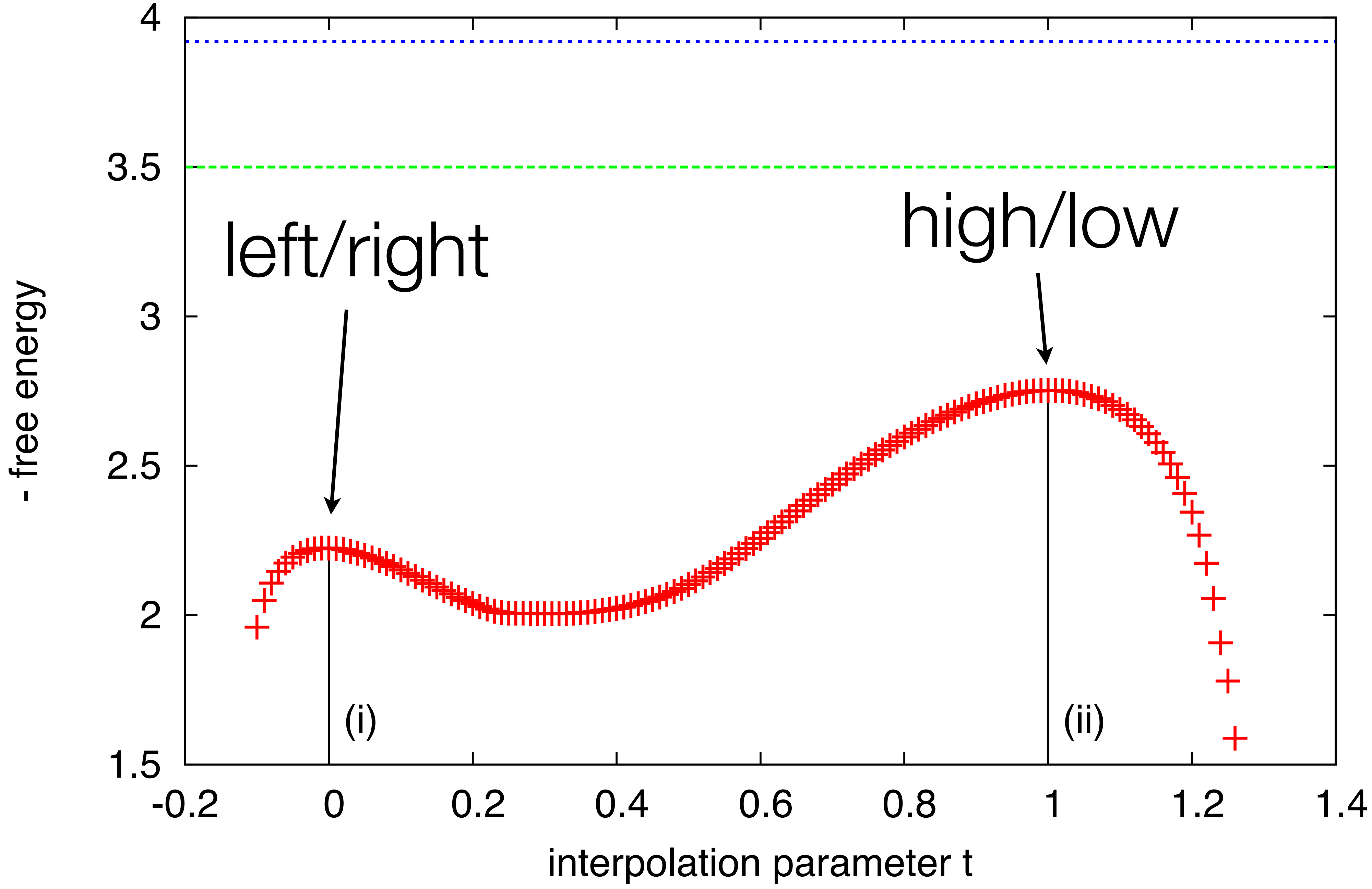
# The Karate Club: two factions



# The Karate Club: leaders vs. followers



# Two local optima in free energy



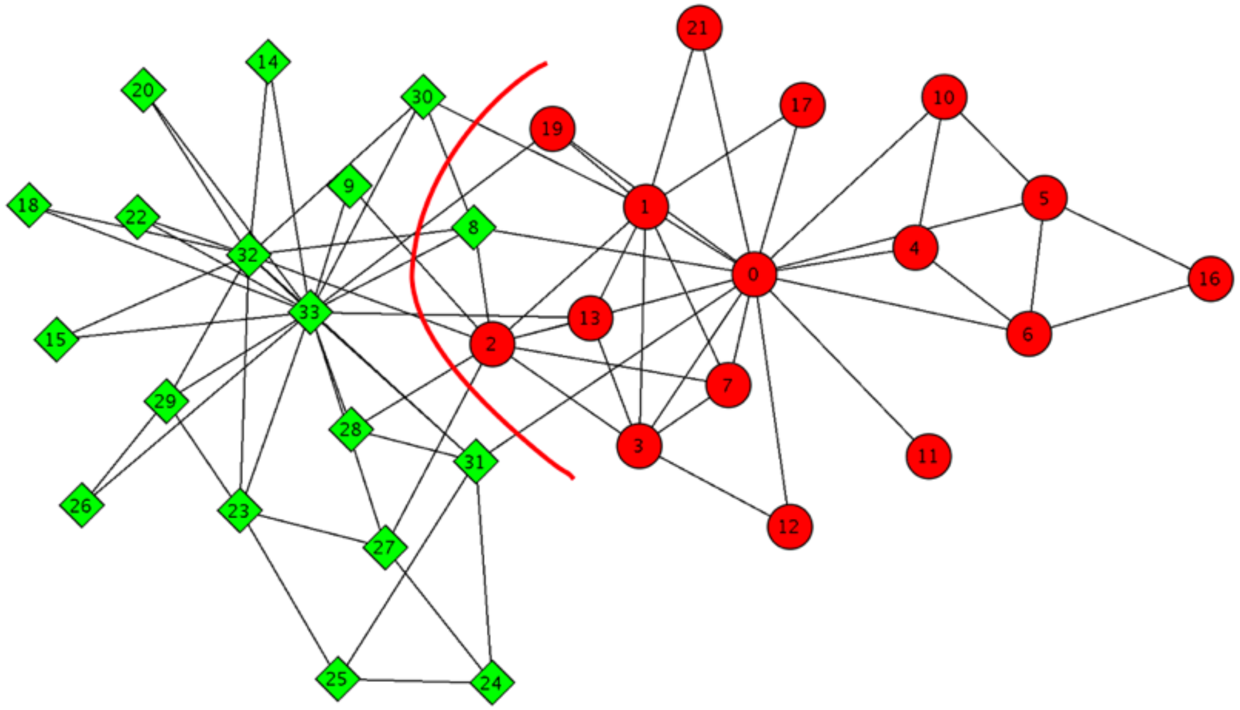
[Decelle, Krzakala, Moore, Zdeborová]

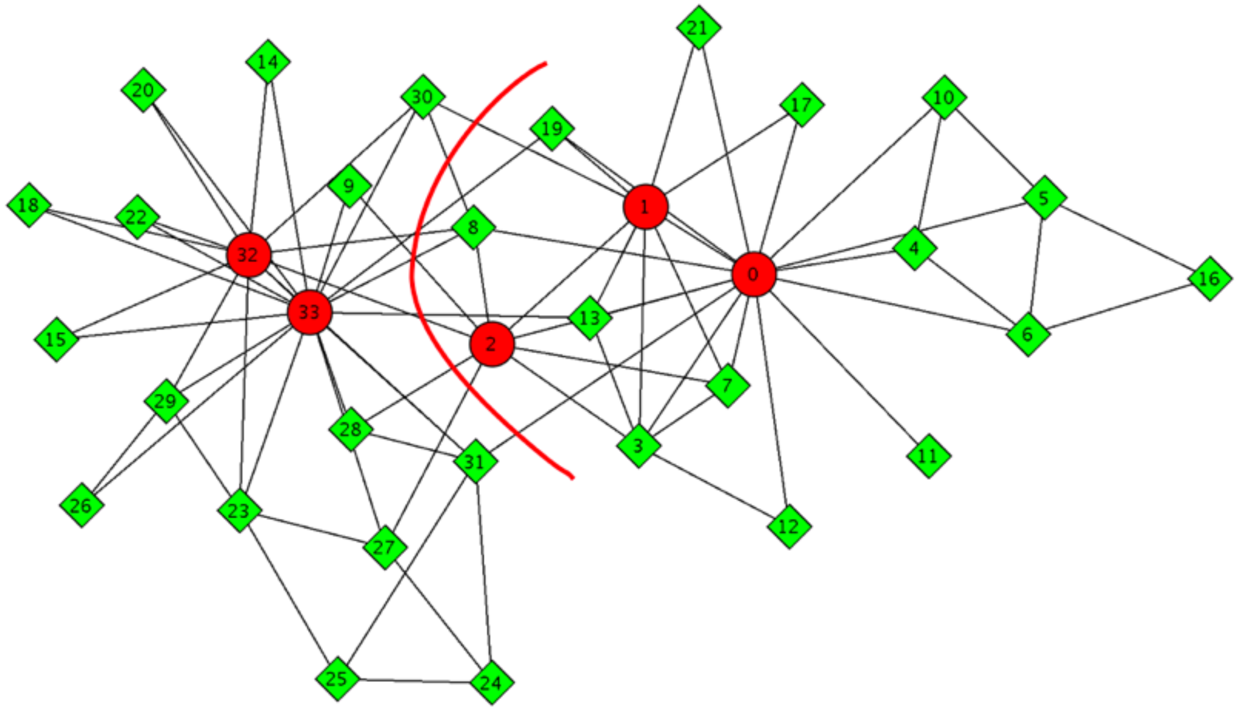
# What kind of structure do you want to find?

different models give different answers for the communities

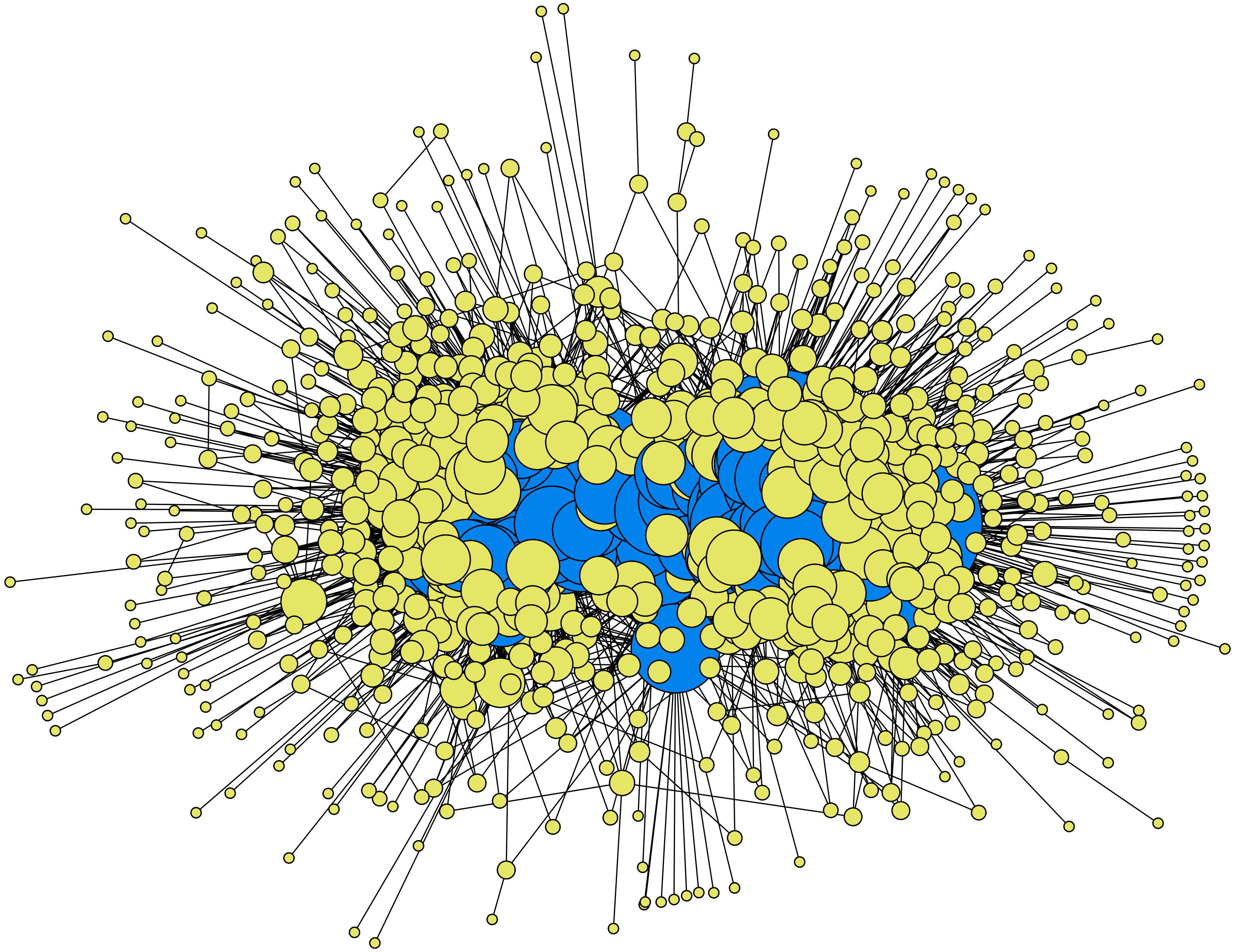
we can compare each one to “ground truth” and judge its accuracy...

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



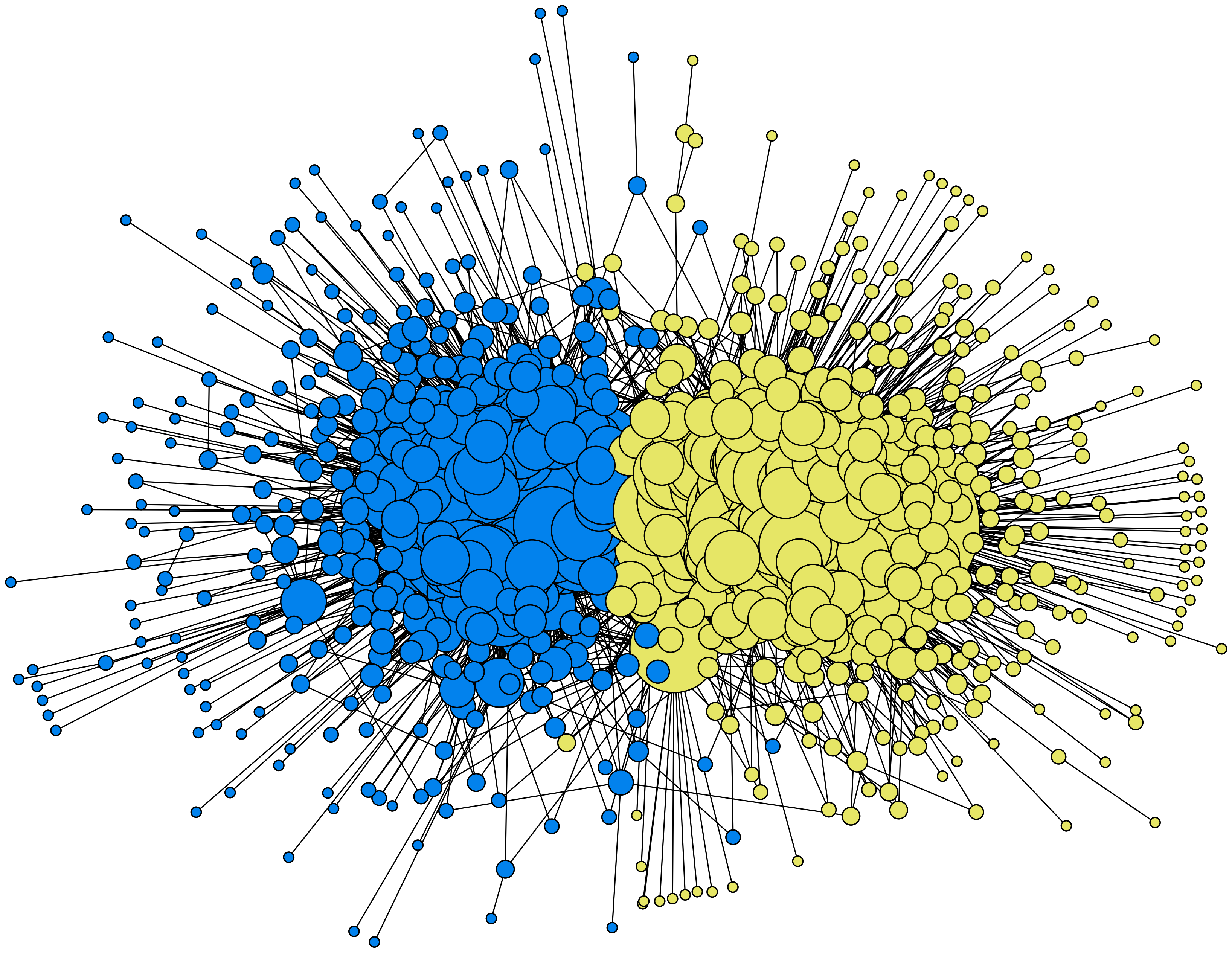


# Blogs: vanilla block model



[Karrer & Newman]

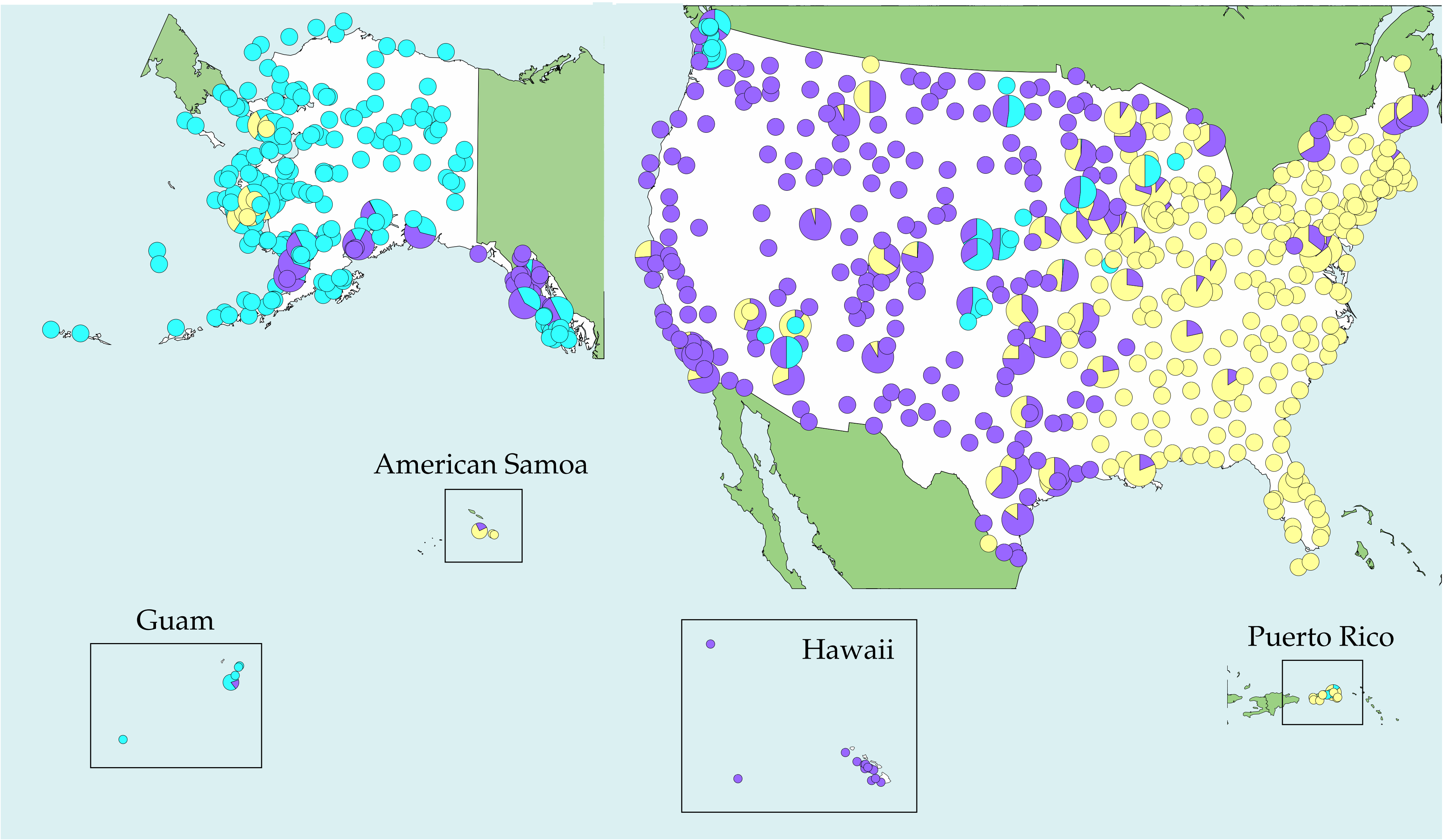
# Blogs: degree-corrected block model



[Karrer & Newman]

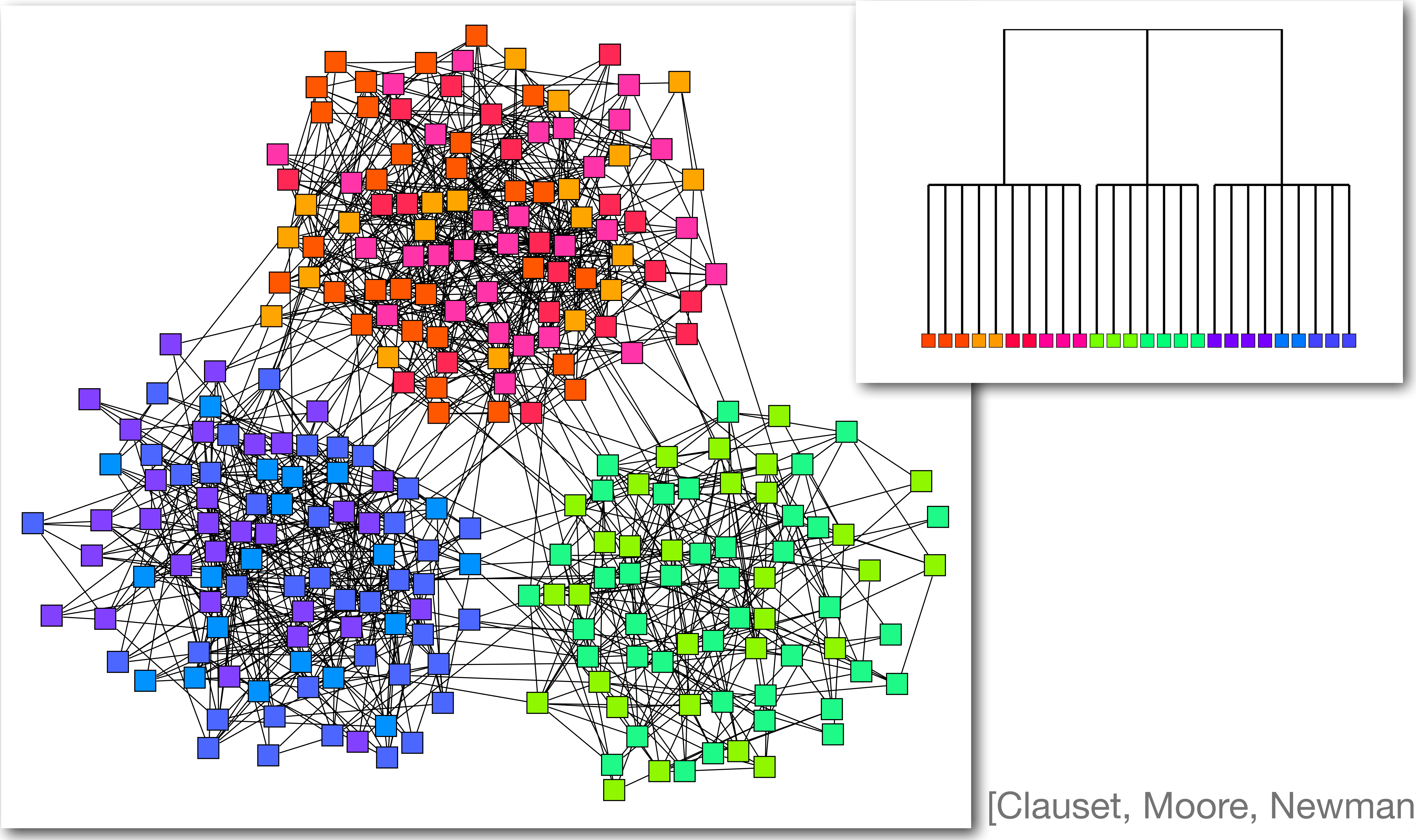
# Overlapping communities

mixed-membership block model: each node has a mix of types, and can act like different types on different edges



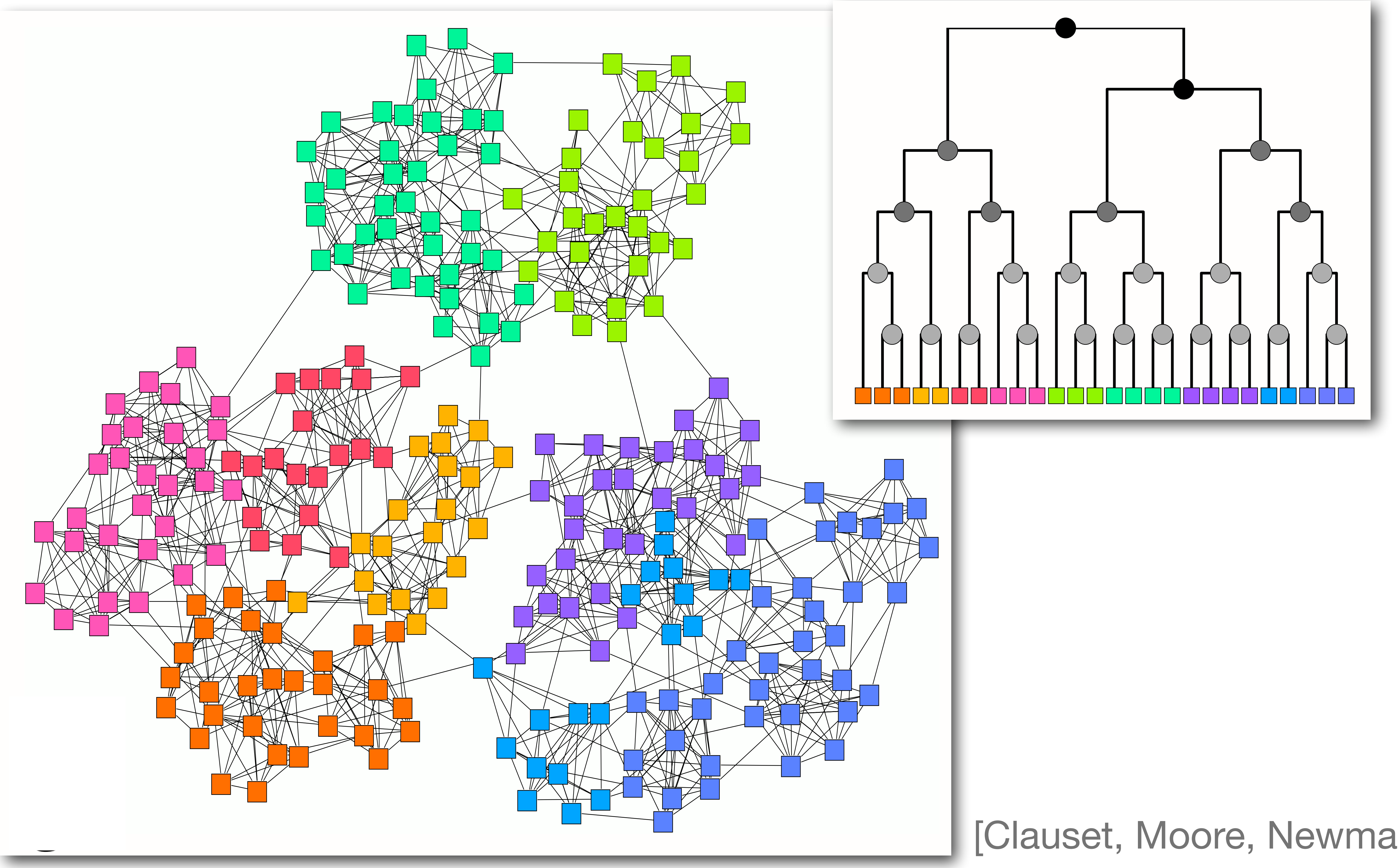
[Ball, Karrer, Newman]

# Hierarchy



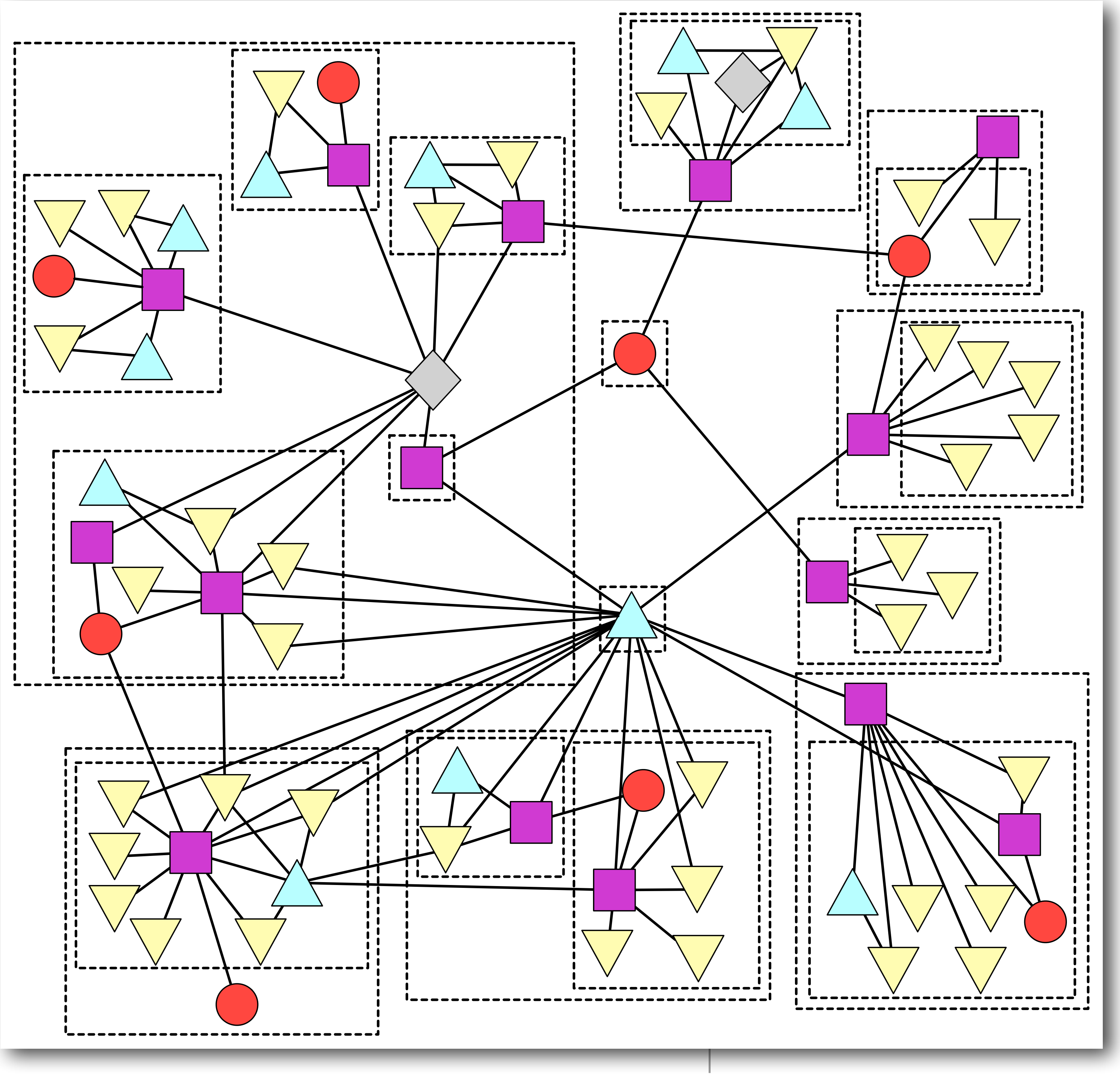
[Clauset, Moore, Newman]

# Hierarchy



[Clauset, Moore, Newman]

# Functional roles in a food web



# Dealing with uncertainty #1: Identifying groups of technologies

patents are documents, and have links (citations) between them

how can we identify groups of technologies, and understand how they depend on each other?

test case: 1,000 microprocessor patents

- arithmetic
- multiplexer
- buses
- microinstructions
- microprograms

- testing
- debugging
- emulator
- error
- traces
- embedding
- jumps
- halting

- power
- reset
- frequencies
- pulses
- voltages
- sensing
- driving
- oscillators

- protection
- transparent
- security
- multi-tasking
- encryption
- restricting

- branching
- prediction
- concurrence
- speculation
- reordering

using both text and links does better than using either one alone

[Zhu, Moore, Valverde]

# Dealing with uncertainty #2: Predicting missing links

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

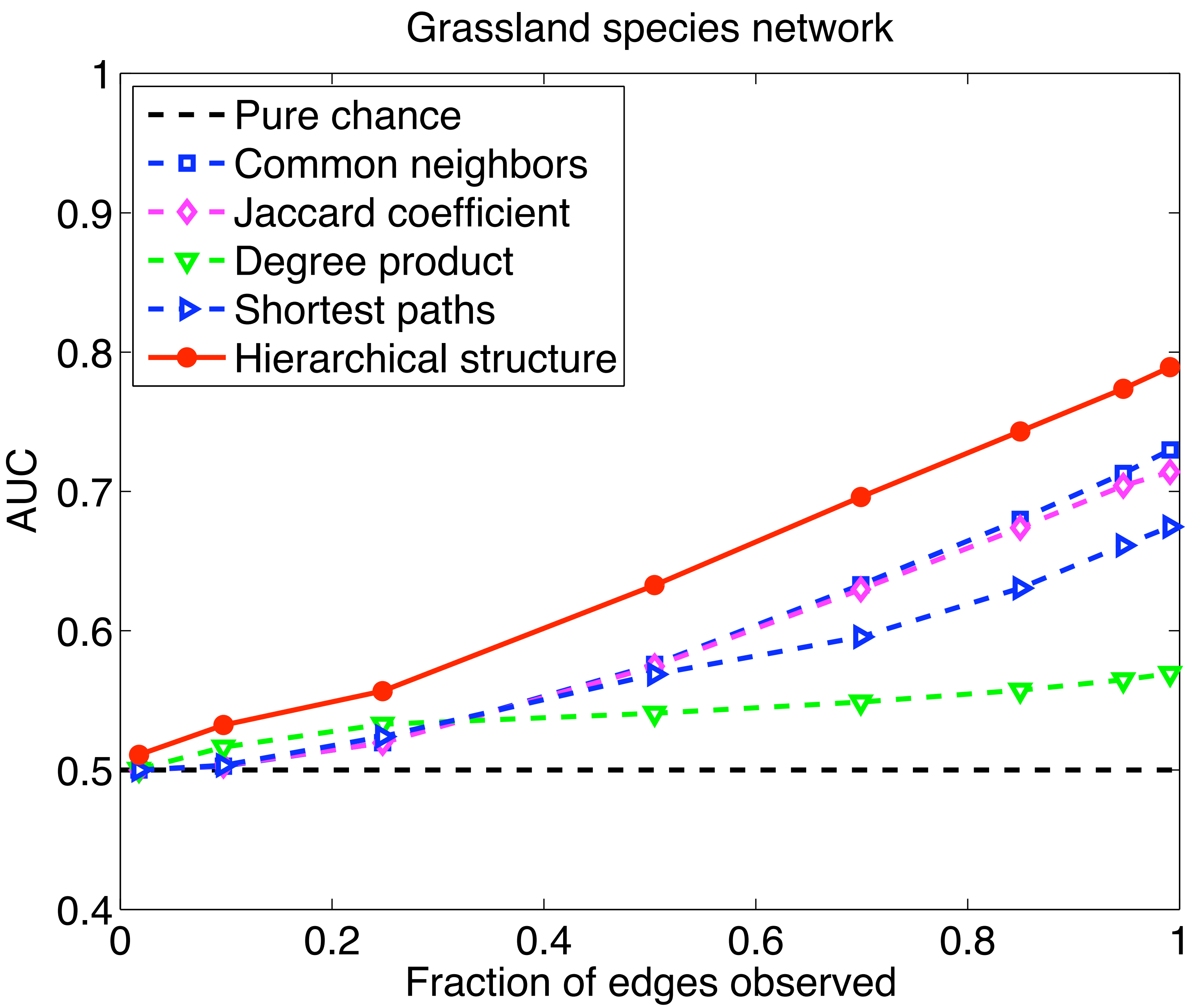
given the links observed so far, can we predict missing links?

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 their probability

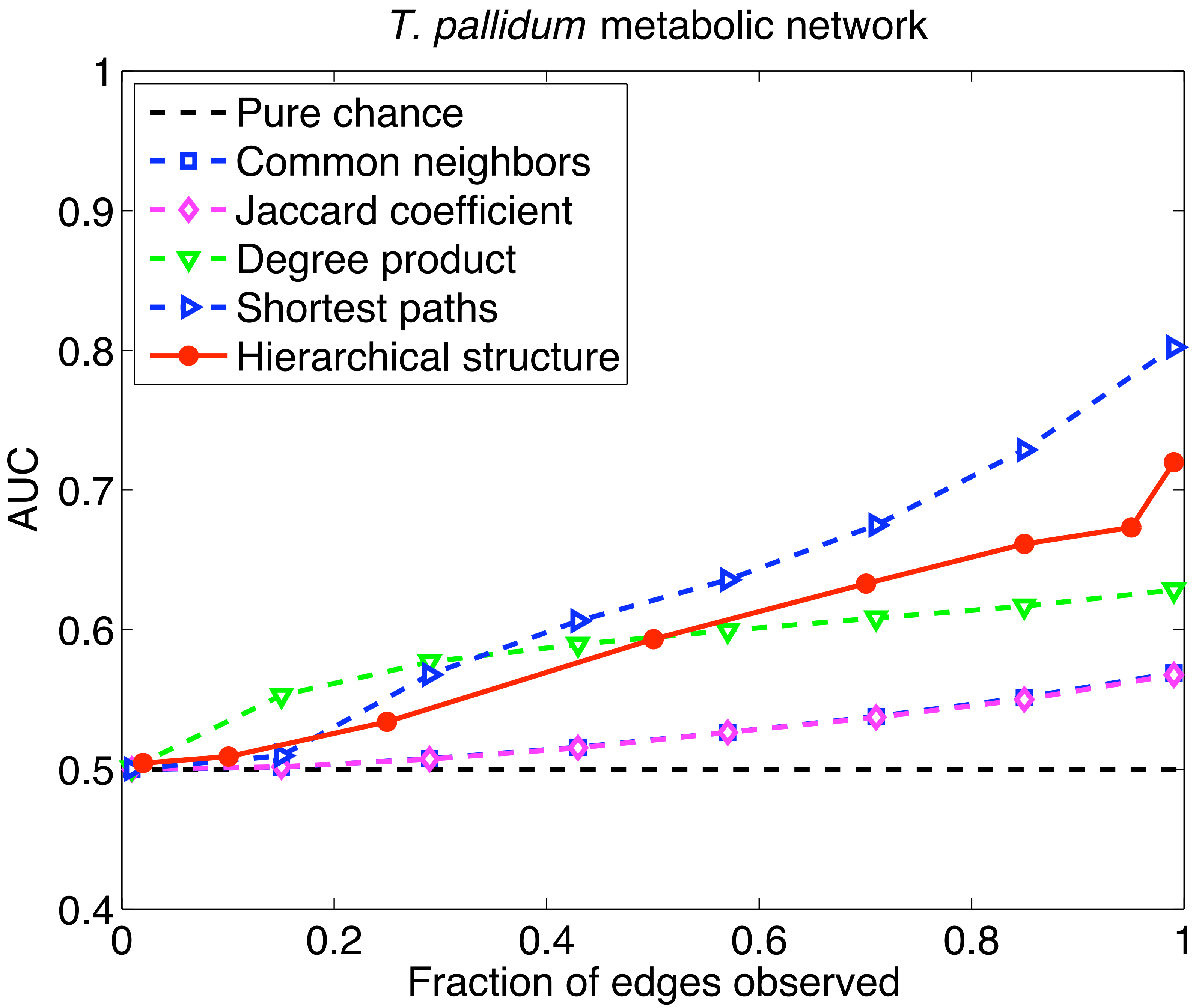
# Predicting missing links: comparison with simple heuristics

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



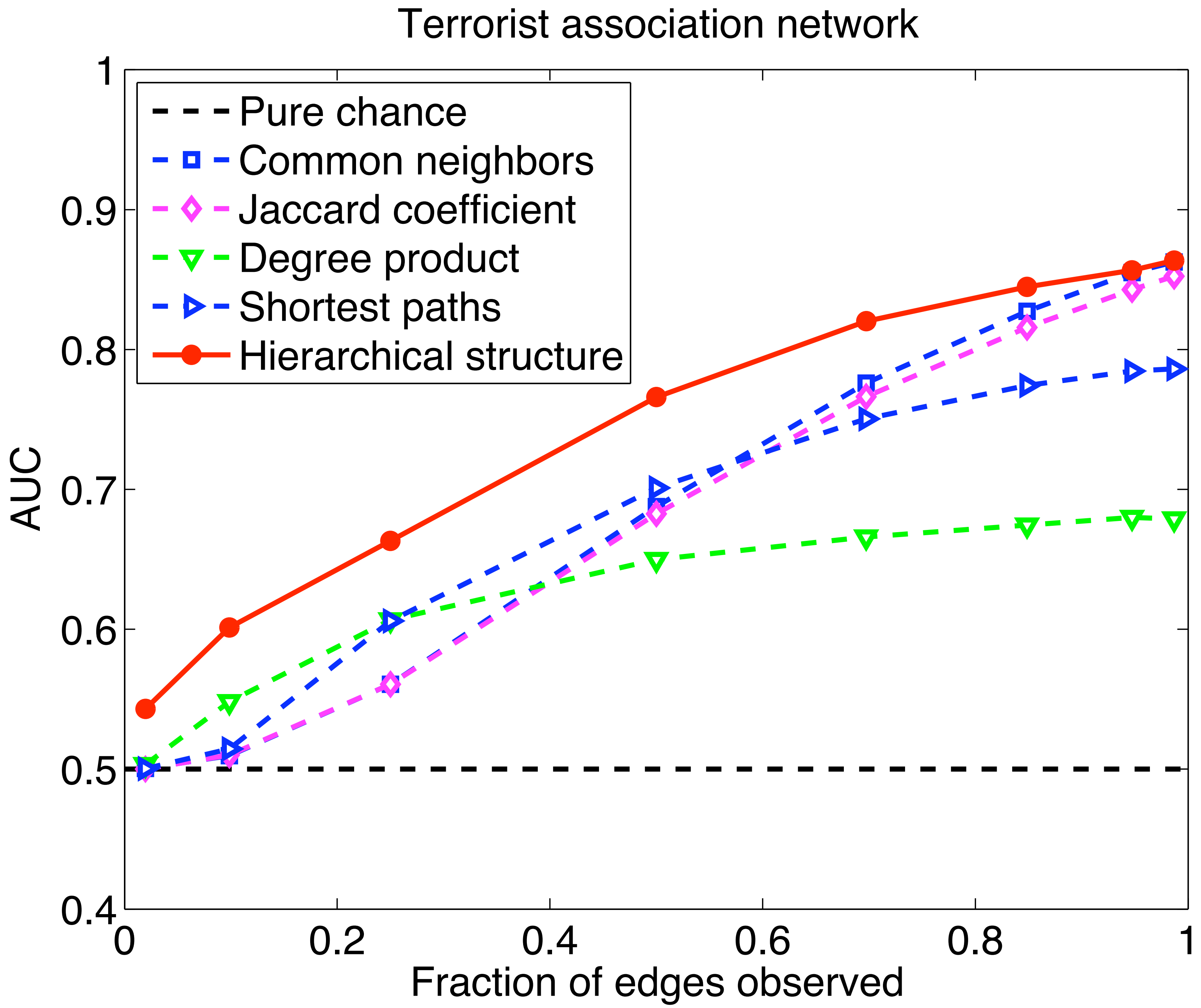
# Predicting missing links: comparison with simple heuristics

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



# Predicting missing links: comparison with simple heuristics

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



# Dealing with uncertainty #3:
# Active exploration of networks

suppose we can learn a node's attributes, but at a cost: interviews, surveys, incentives, warrants

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:

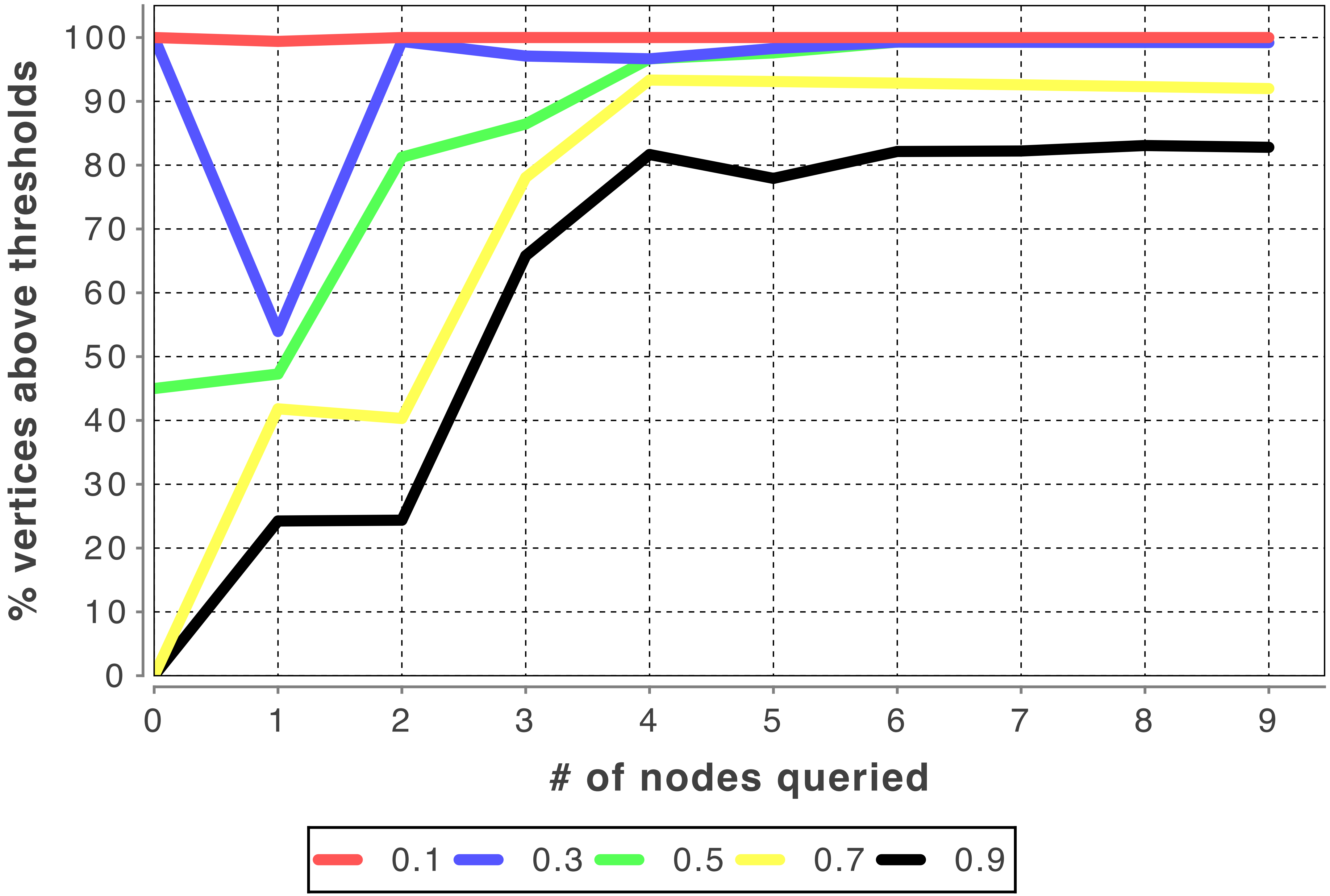
$$\begin{aligned} I(v, G-v) &= H(v) - H(v \mid G-v) \\ &= H(G-v) - H(G-v \mid v) \end{aligned}$$

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

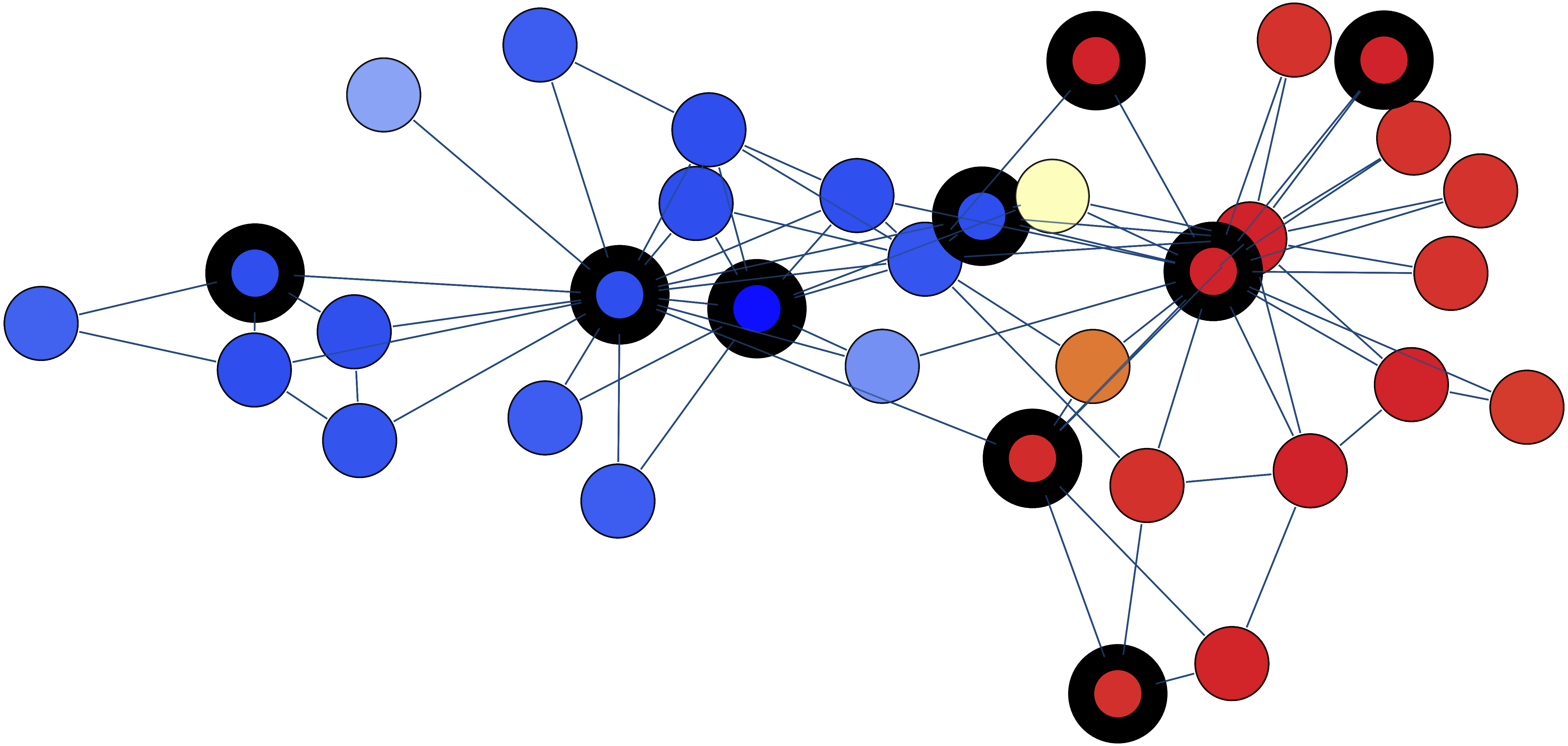
high when we're uncertain about *v*, and when *v* is highly correlated with others

[Moore, Yan, Zhu, Rouquier, Lane]

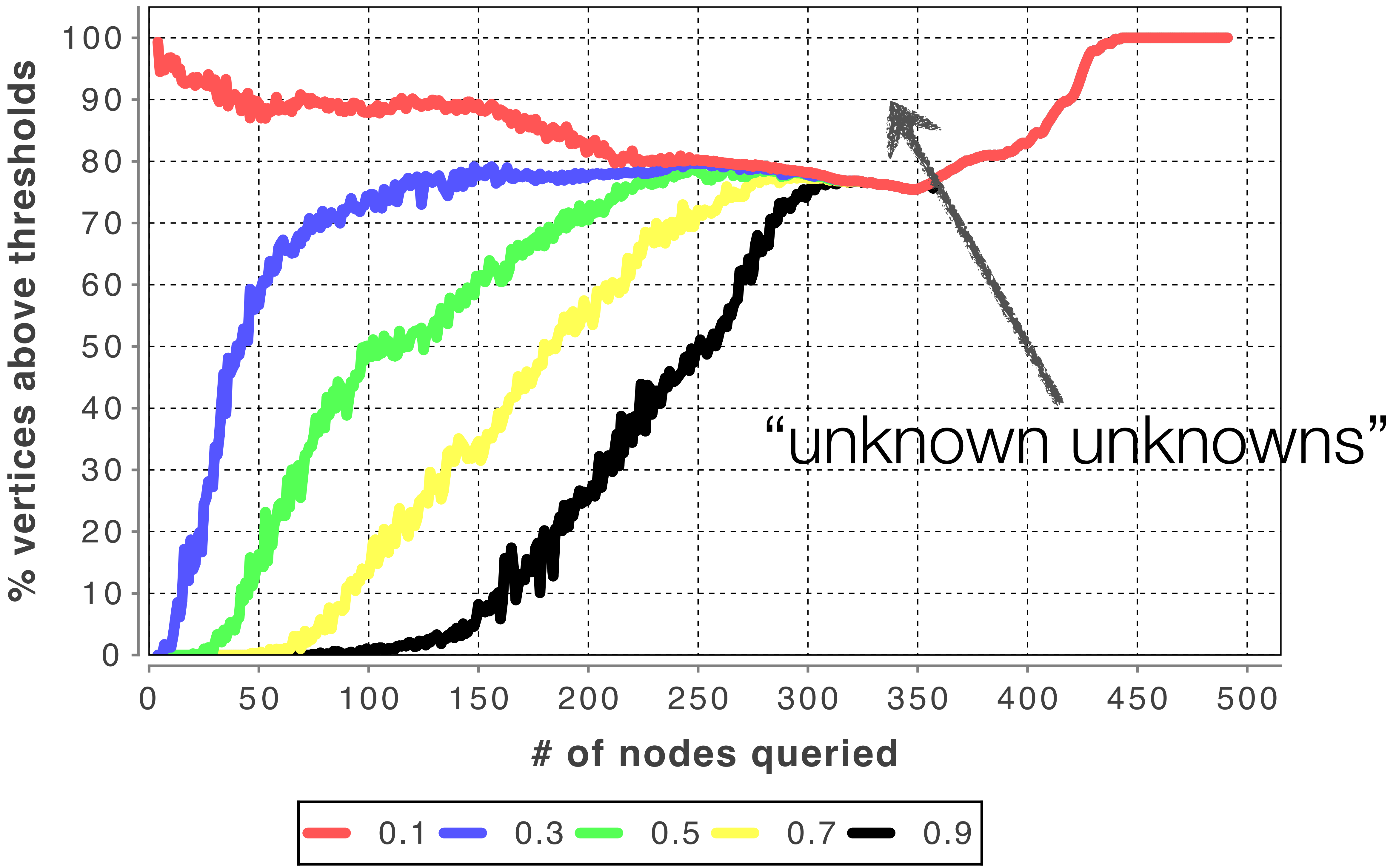
# Learning factions in the Karate Club



# How does the algorithm explore the network?



# An antarctic food web



"unknown unknowns"

# The story so far

statistical inference, powered by ideas from physics, and carried out with highly scalable algorithms, lets us

detect communities

label nodes

predict missing links

these models and algorithms reveal phase transitions where communities become detectable, or where knowledge suddenly spreads across the network

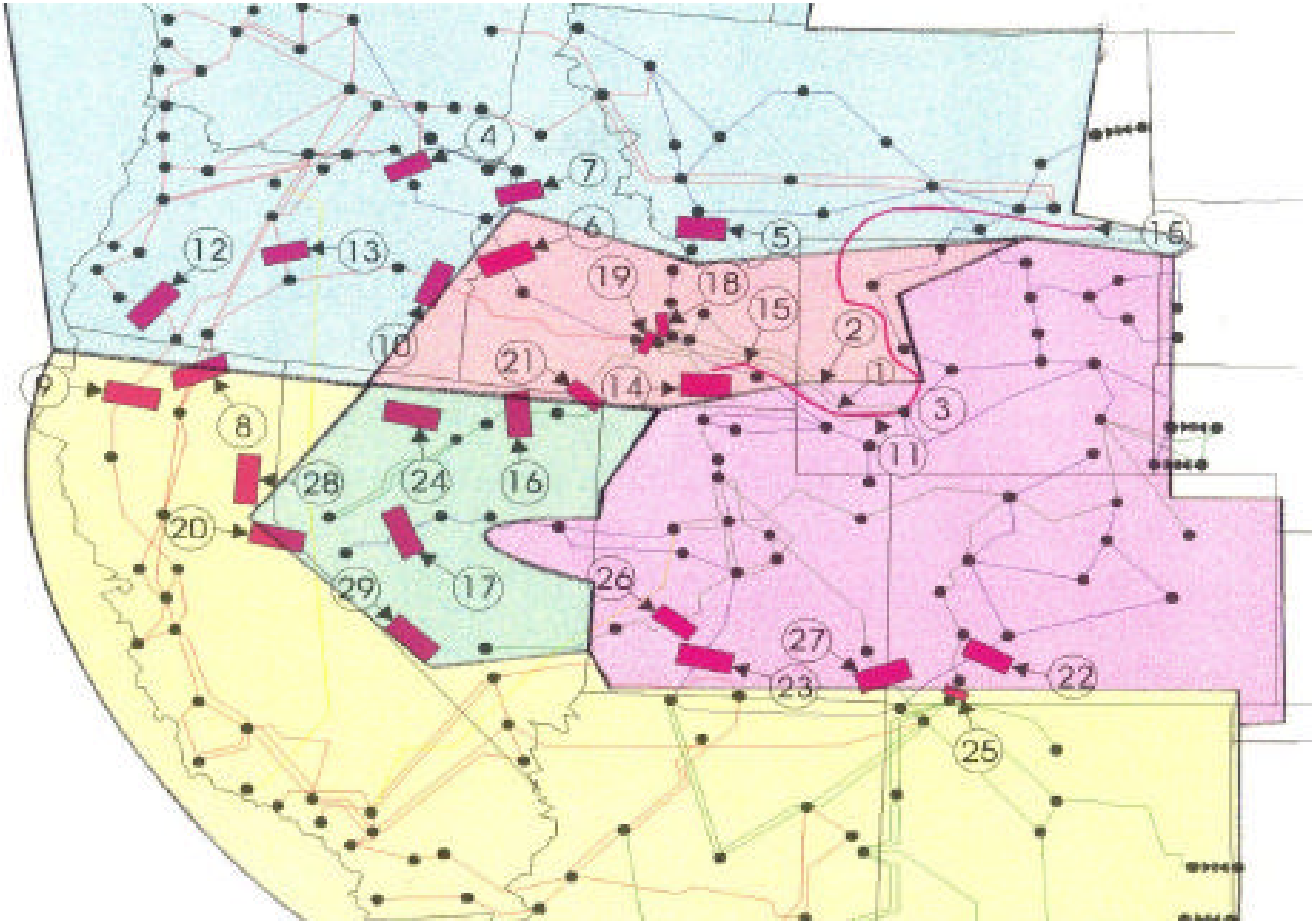
we can elaborate these models by adding discrete or continuous attributes: degree distributions, edge types, social status, overlapping communities, hierarchy, signed edges, document content...

but a cautionary note!

Everything is anstecaphongyine

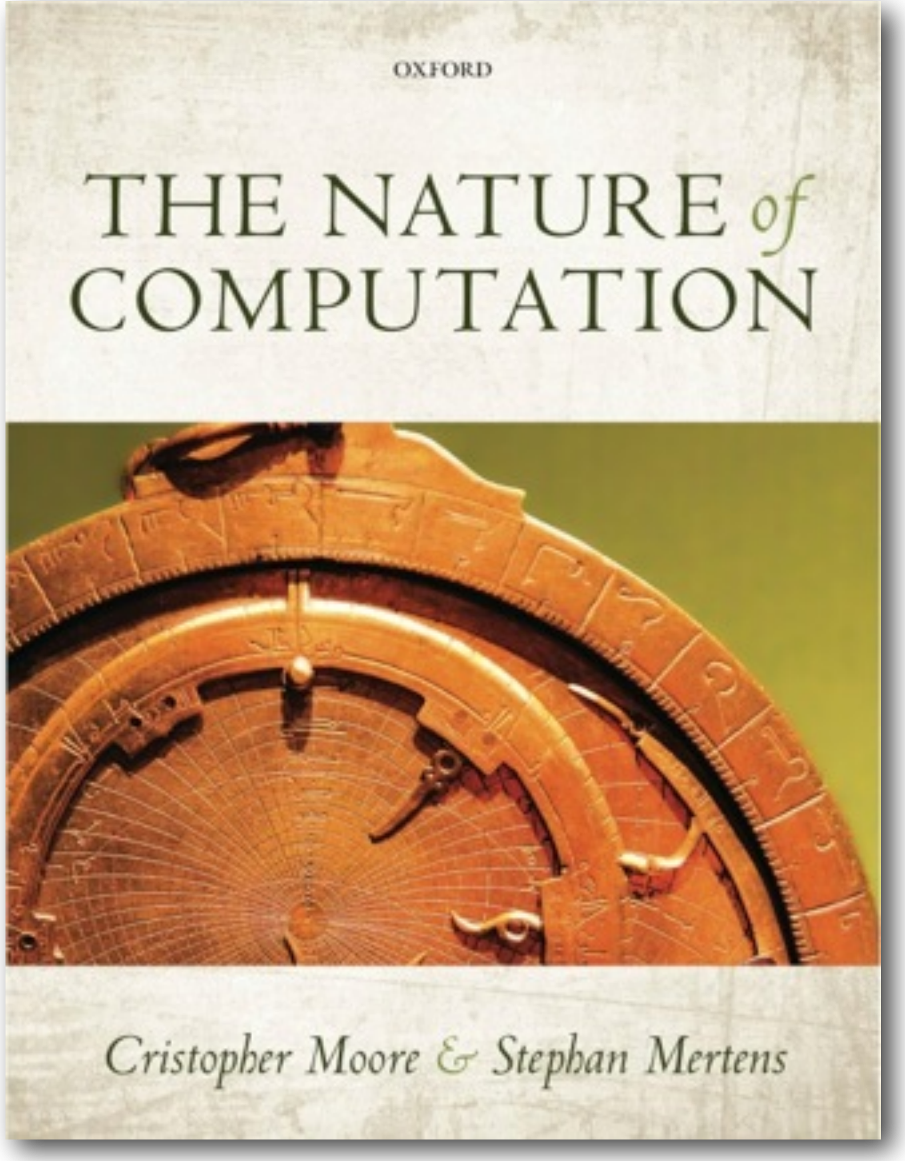
# A real cascade of line and generator failures

Sequence of outages in Western blackout, July 2 1996



from NERC 1996 blackout report

# 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