Announcements (just FYI):

1) HW3: Due 11/10

2) Project milestones due on 11/17

We expect ~50% project completed

Macroscopic Evolution of Networks

CS224W: Social and Information Network Analysis
Jure Leskovec, Stanford University
http://cs224w.stanford.edu

Macroscopic Evolution

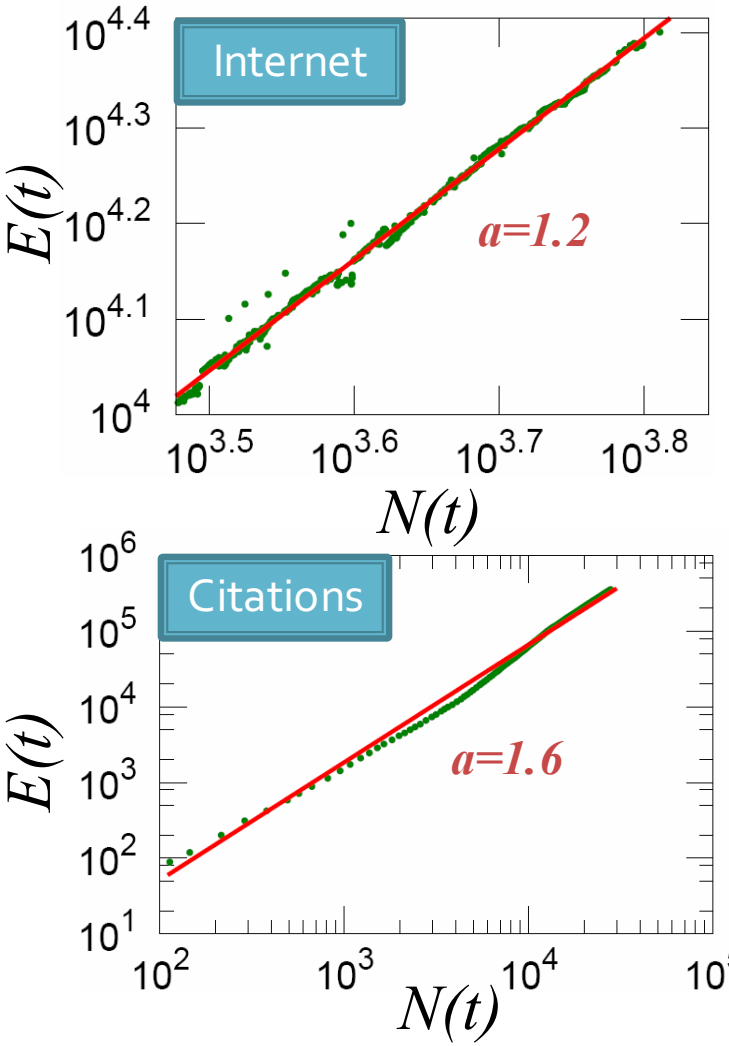
- **How do networks evolve at the macro level?**
  - What are global phenomena of network growth?
- **Questions:**
  - What is the relation between the number of nodes $n(t)$ and number of edges $e(t)$ over time t?
  - How does diameter change as the network grows?
  - How does degree distribution evolve as the network grows?

Q1) Network Evolution

- **What is the relation between the number of nodes and the edges over time?**
- First guess: constant average degree over time
- Networks are **denser** over time
- **Densification Power Law:**

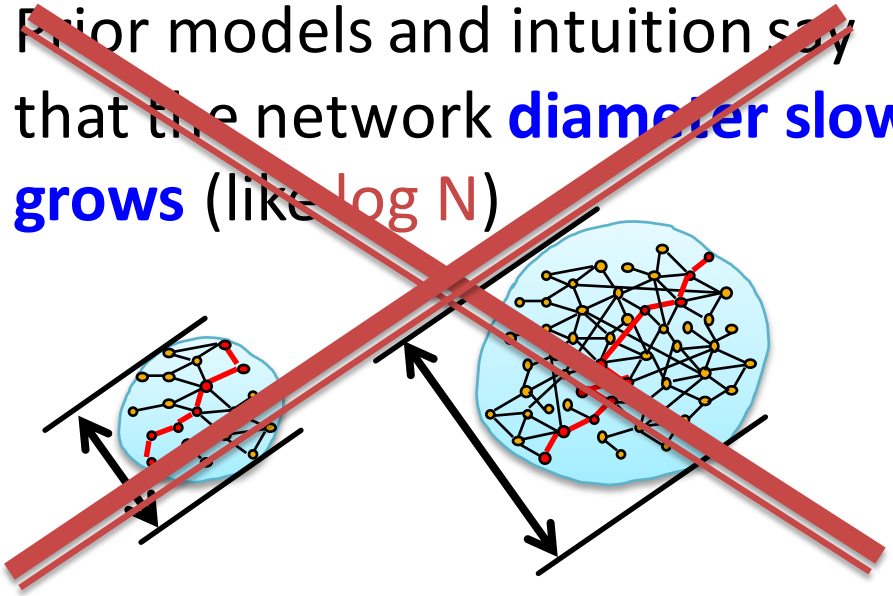
$$E(t) \propto N(t)^a$$

a … densification exponent ($1 \le \mathbf{a} \le 2$)

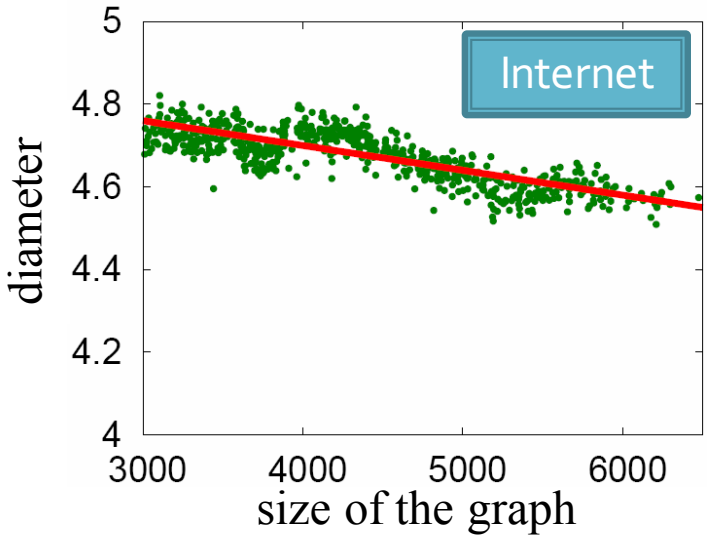


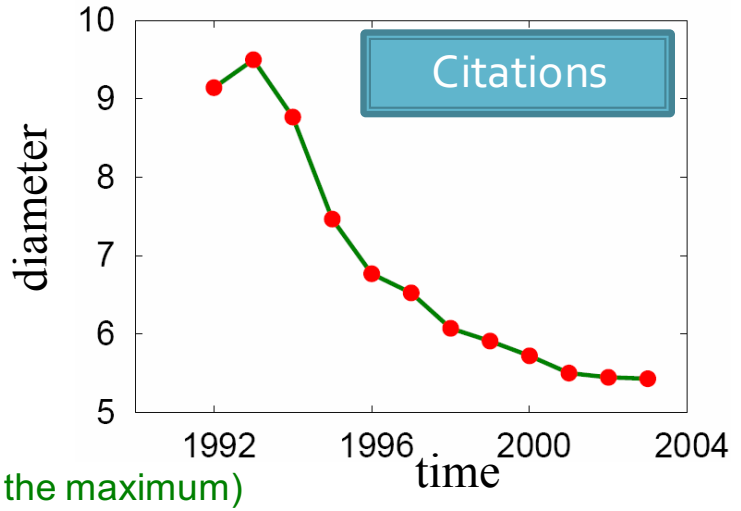
Q1) Network Evolution

- Prior models and intuition say that the network **diameter slowly grows** (like log N)



- **Diameter shrinks over time**
  - As the network grows the distances between the nodes slowly **decrease**





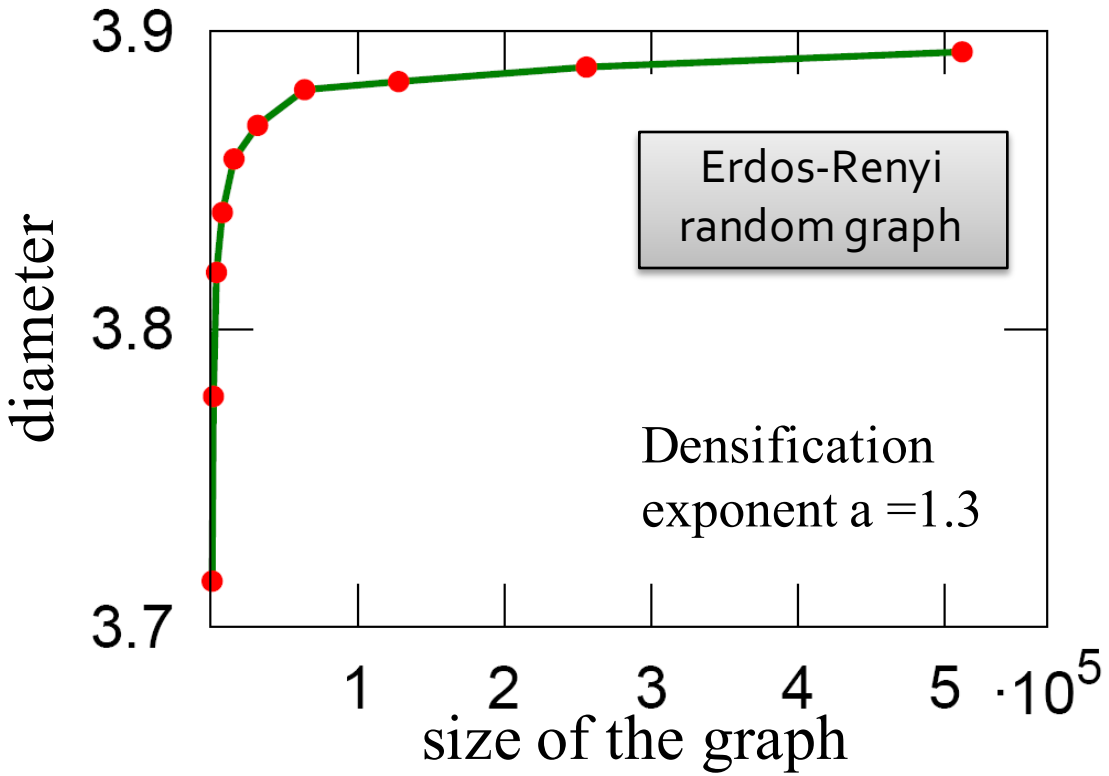
How do we compute diameter in practice?

-- Long paths: Take 90th-percentile or average path length (not the maximum)

-- Disconnected components: Take only largest component or average only over connected pairs of nodes

Diameter of a Densifying G_{np}

Is shrinking diameter just a consequence of densification?



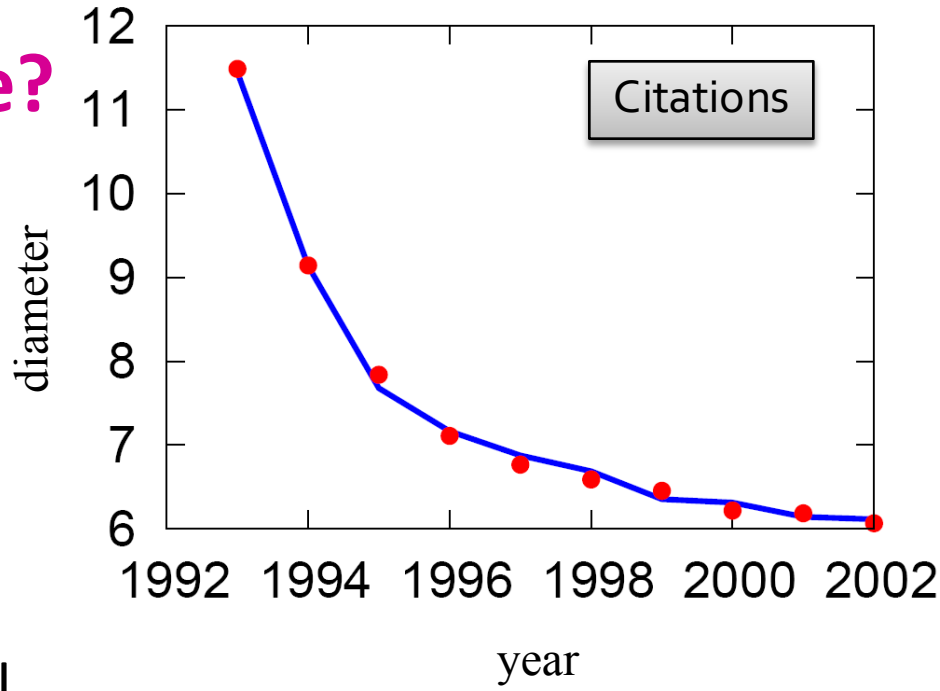
Densifying random graph has increasing diameter
⇒ **There is more to shrinking diameter than just densification!**

Diameter of a Rewired Network

Is it the degree sequence?

Compare diameter of a:

- Real network (**red**)
- Random network with the same degree distribution (**blue**)
  - Apply configuration model to a network at time *t*

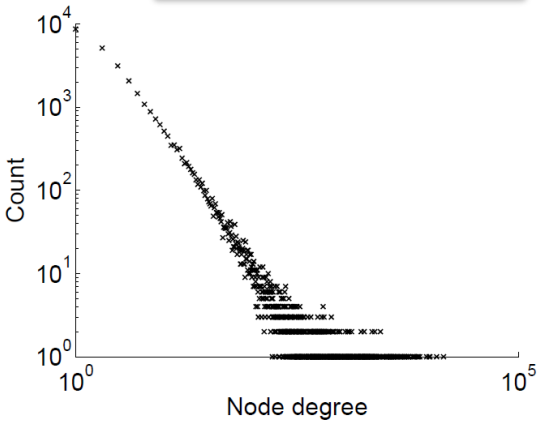


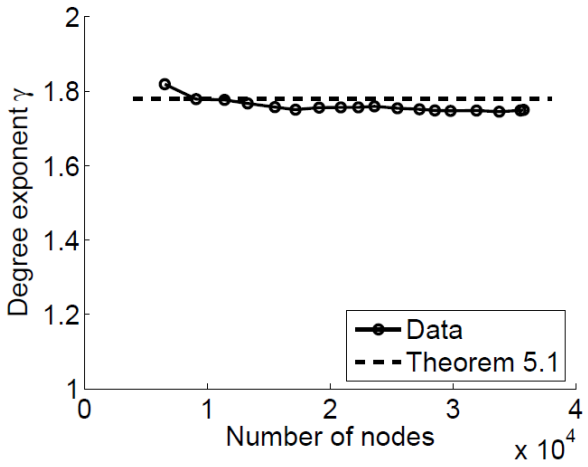
Densification + degree sequence gives shrinking diameter

Connecting Degrees & Densification

- **How does degree distribution evolve to allow for densification?**
- **Option 1) Degree exponent γ_t is constant:**
  - **Fact 1:** If $\alpha_t = \alpha \in [1, 2]$, then: $\alpha = 2/\alpha$

Email network





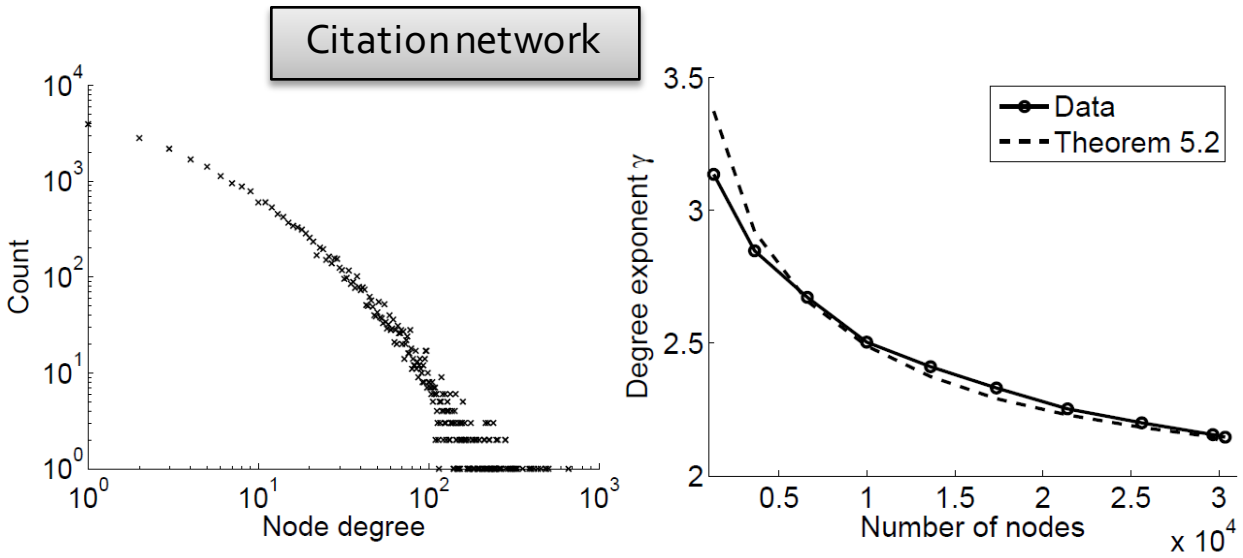
A consequence of what we learned in the Power law lecture:

- Power-laws with exponents <2 have infinite expectations.
- So, by maintaining constant degree exponent α the average degree grows.

Connecting Degrees & Densification

- **How does degree distribution evolve to allow for densification?**
- **Option 2) α_t evolves with graph size n:**
  - **Fact 2:** If $\alpha_t = \frac{4n_t^{x-1}-1}{2n_t^{x-1}-1}$, then: $\alpha = x$

Notice: $\alpha_t \to 2$ as $n_t \to \infty$



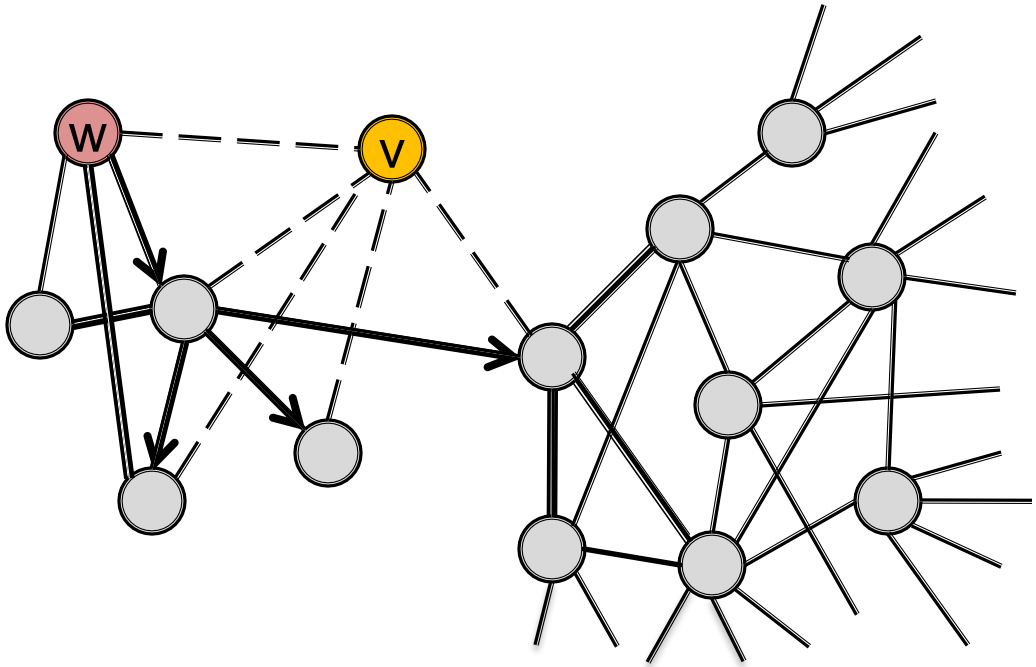
Remember, the expected degree in a power law is:

$$E[X] = \frac{\alpha_t - 1}{\alpha_t - 2} x_m$$

So α_t has to decay as a function of graph size n_t for the avg. degree to go up.

Forest Fire Model

- **Want to model graphs that densify and have shrinking diameters**
- **Intuition:**
  - How do we meet friends at a party?
  - How do we identify references when writing papers?



Forest Fire Model

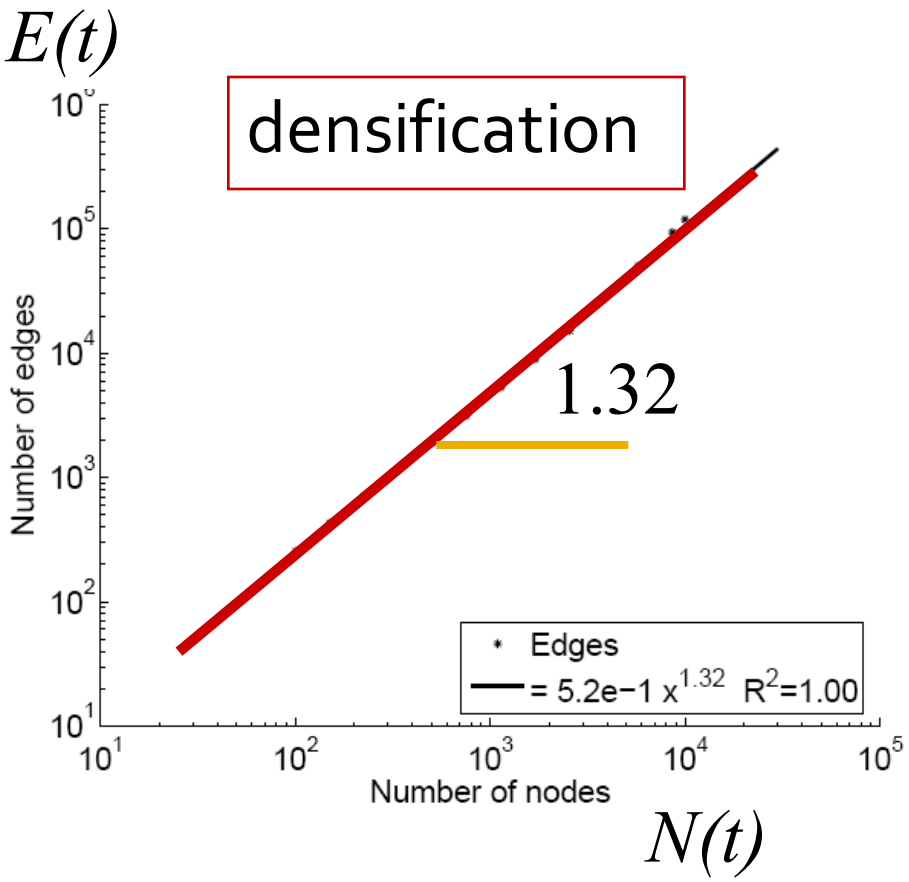
- **The Forest Fire model has 2 parameters:**
  - ***p*** … forward burning probability
  - ***r*** … backward burning probability
- **The model: Directed Graph**
  - Each turn a new node *v* arrives
  - Uniformly at random chooses an "ambassador" *w*
  - Flip 2 geometric coins (based on *p* and *r*) to determine the number of **in-** and **out-links** of *w* to follow
  - "Fire" spreads recursively until it dies
  - New node *v* links to all burned nodes

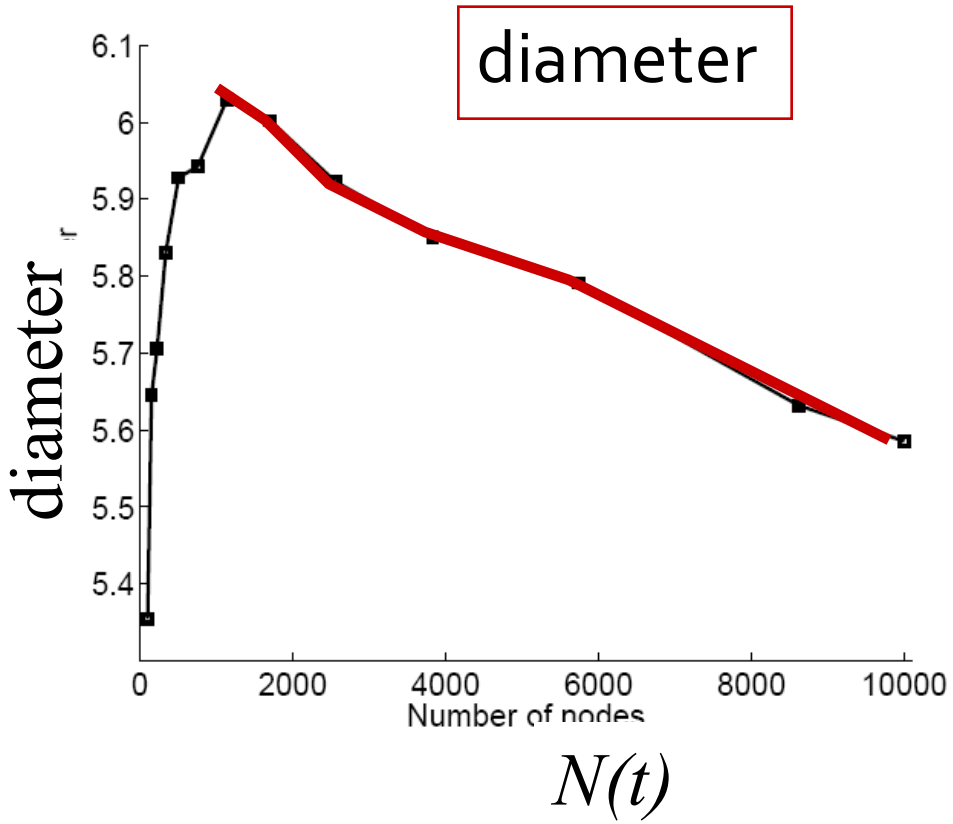
Geometric distribution:

$$\Pr(X = k) = (1 - p)^{k-1} p$$

Forest Fire Model

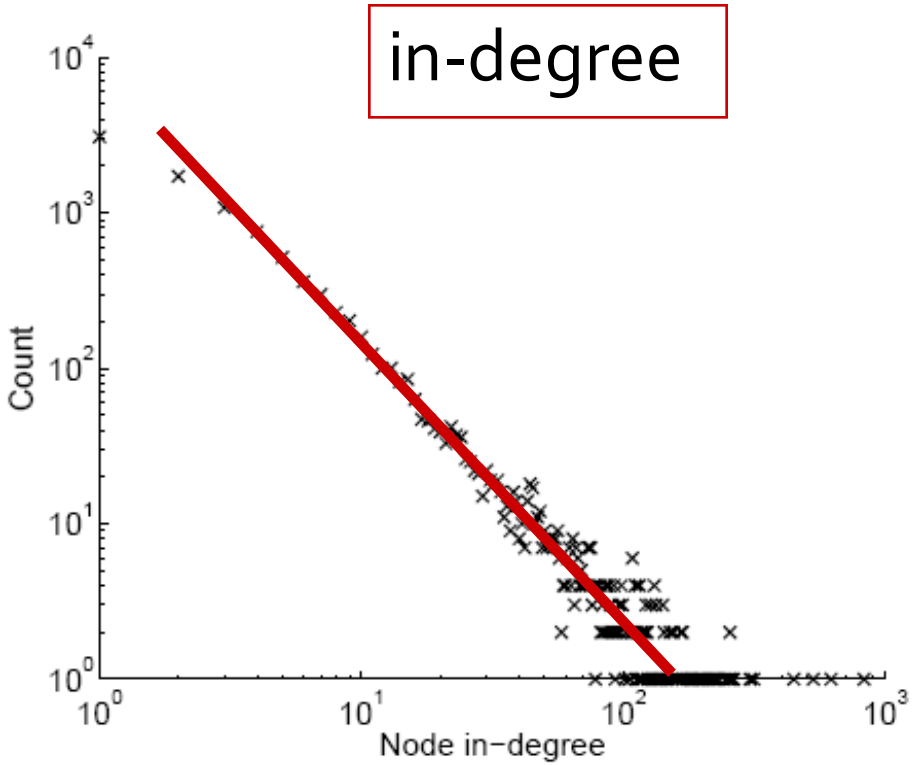
- **Forest Fire generates graphs that densify and have shrinking diameter**



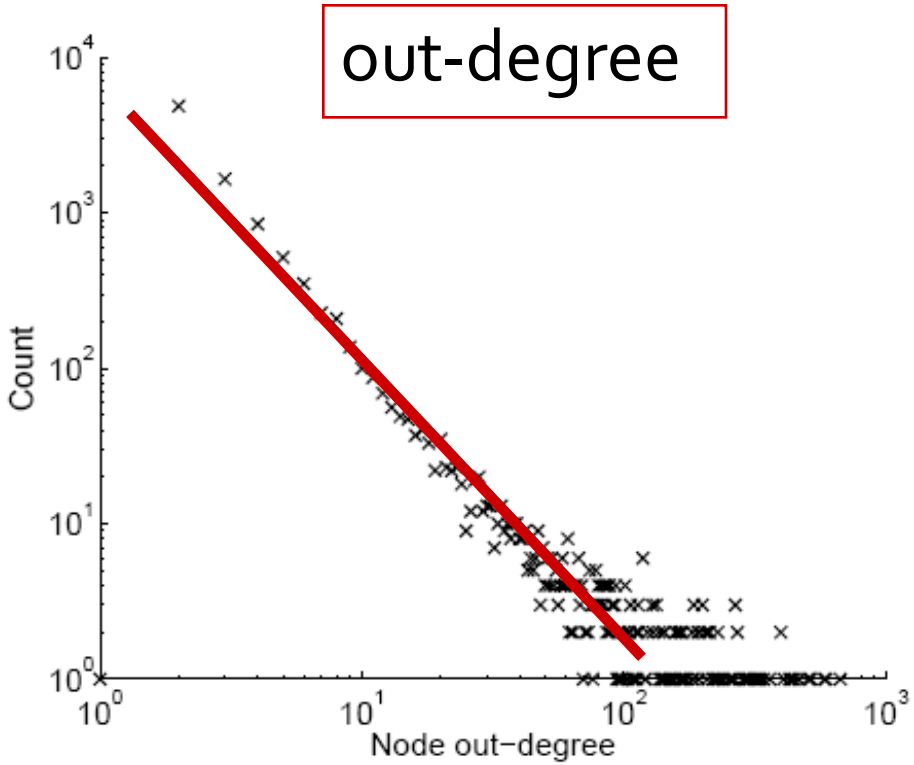


Forest Fire Model

- **Forest Fire also generates graphs with power-law degree distribution**



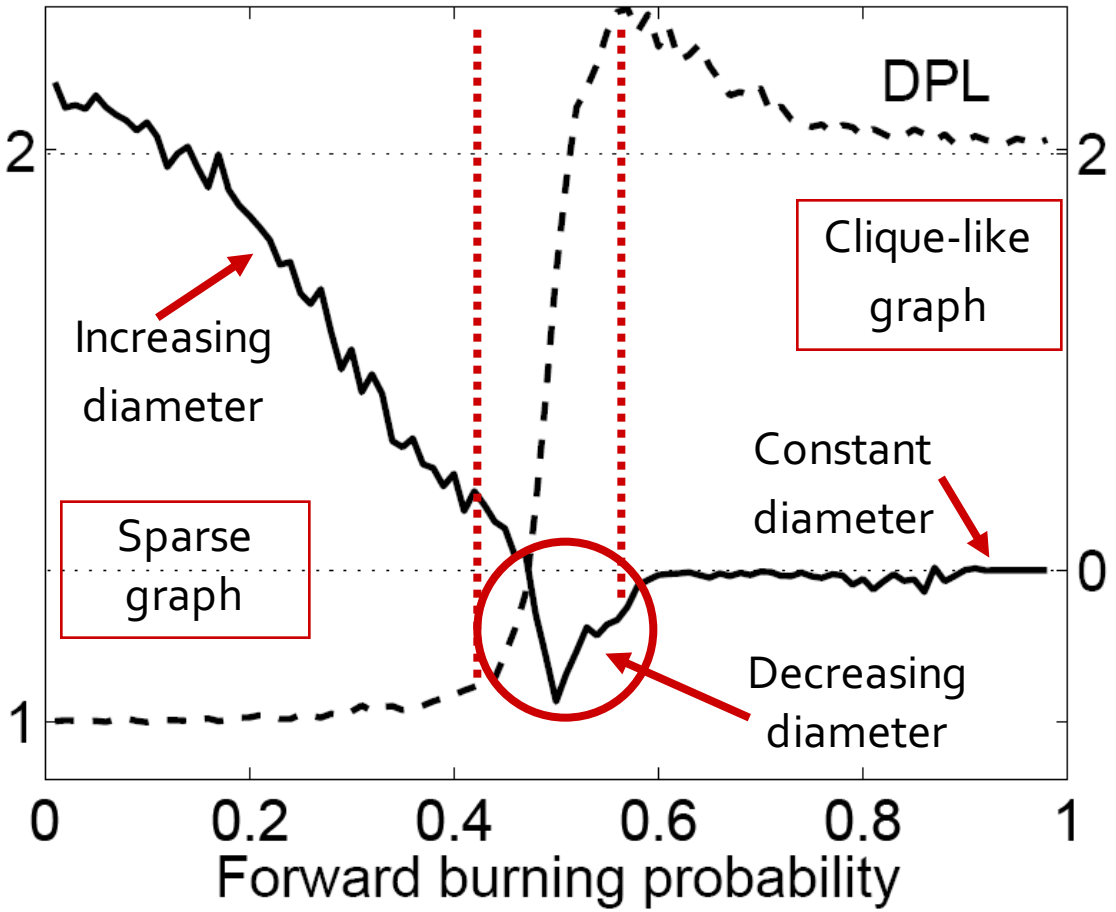
log count vs. log in-degree



log count vs. log out-degree

Forest Fire: Phase Transition

- Fix backward probability *r* and vary forward burning prob. *p*
- Notice a sharp transition between sparse and clique-like graphs
- **The "sweet spot" is very narrow**



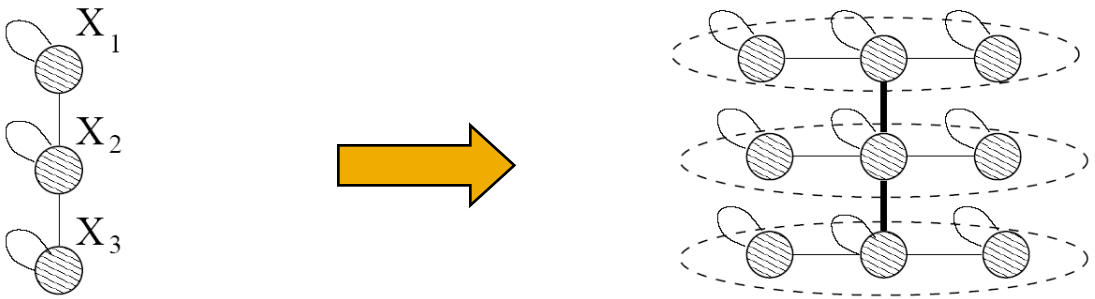
Kronecker Graphs Model

Models of Networks

- **What is the goal of modeling networks?**
  - **Discover structural properties of networks**
    - Small-world, Edge clustering, Heavy-tailed degrees
  - **Find a model that gives graphs with such properties**
    - Erdos-Renyi, Watts-Strogatz, Barabasi-Albert model
- **Today's lecture:**
  - **Can we have a model that attempts to reproduce all of these properties?**
  - **Can we fit the model to a network and accurately reproduce the network?**

Idea: Recursive Graph Generation

- **How can we think of network structure recursively? Intuition:** Self-similarity
  - **Object is similar to a part of itself**: the whole has the same shape as one or more of the parts
- **Mimic recursive graph/community growth:**



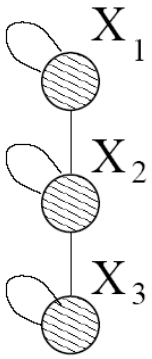
Initial graph

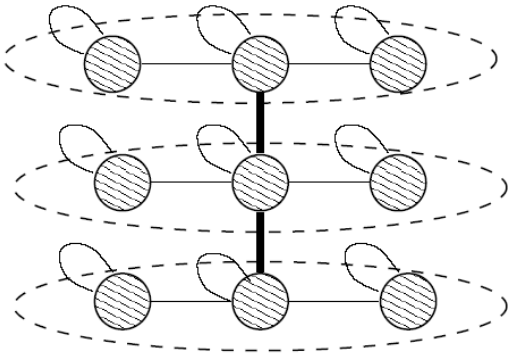
Recursive expansion

- **Kronecker graph** is a way of generating self-similar matrices

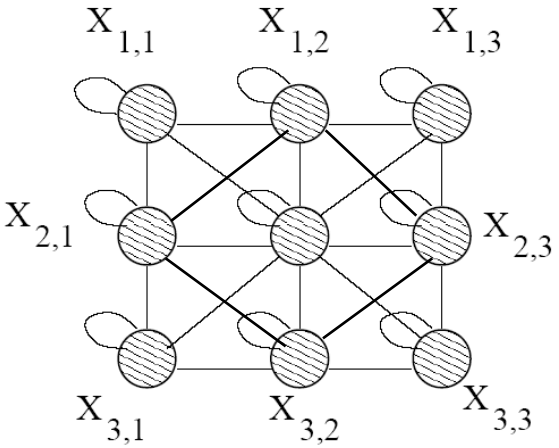
[PKDD '05]

Kronecker: Graph Growth





Intermediate stage



1	1	0
1	1	1
0	1	1

(3x3)

K_1

Initiator graph

K_1	K_1	0
K_1	K_1	K_1
0	K_1	K_1

(9x9)

$K_2 = K_1 \otimes K_1$

After the growth phase

Kronecker Graph

- **Kronecker graphs:**
  - **A recursive model of network structure**

1	1	0
1	1	1
0	1	1

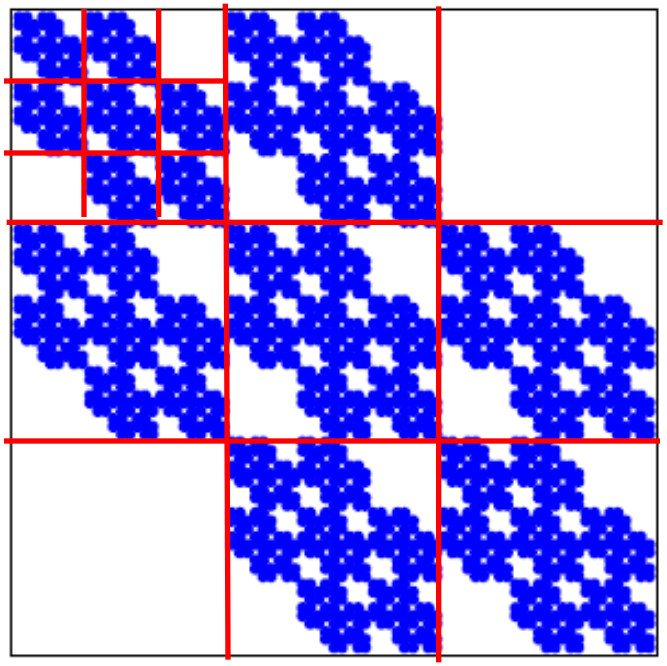
K_1

3 x 3

K_1	K_1	0
K_1	K_1	K_1
0	K_1	K_1

$K_2 = K_1 \otimes K_1$

9 x 9



81 x 81 adjacency matrix

Kronecker Product: Definition

- **Kronecker product** of matrices A and B is given by

$$\mathbf{C} = \mathbf{A} \otimes \mathbf{B} \doteq \begin{pmatrix} a_{1,1}\mathbf{B} & a_{1,2}\mathbf{B} & \dots & a_{1,m}\mathbf{B} \\ a_{2,1}\mathbf{B} & a_{2,2}\mathbf{B} & \dots & a_{2,m}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1}\mathbf{B} & a_{n,2}\mathbf{B} & \dots & a_{n,m}\mathbf{B} \end{pmatrix}$$

N x M *K x L*

*N*K x M*L*

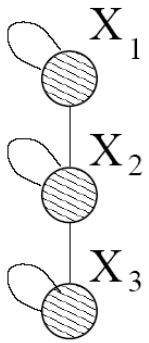
- Define a Kronecker product of two **graphs** as a Kronecker product of their **adjacency matrices**

[PKDD '05]

Kronecker Graphs

- **Kronecker graph:** a growing sequence of graphs by iterating the **Kronecker product:**

1	1	0
1	1	1
0	1	1



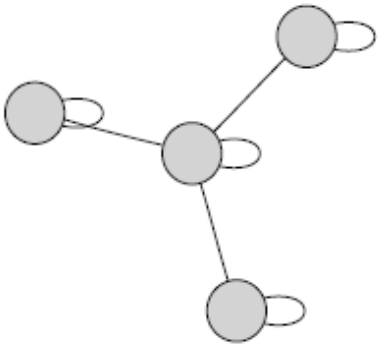
K_1

$$K_1^{[m]} = K_m = \underbrace{K_1 \otimes K_1 \otimes \ldots K_1}_{m \text{ times}} = K_{m-1} \otimes K_1$$

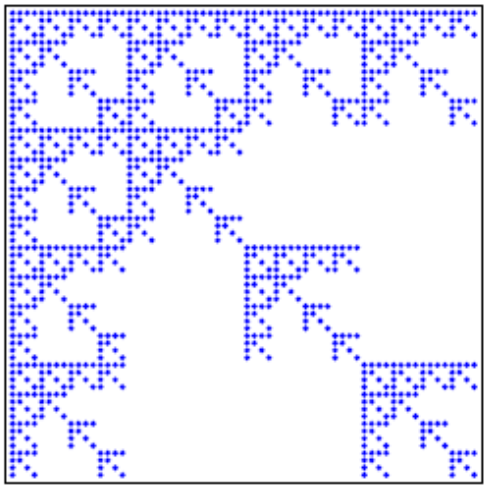
- **Note:** One can easily use multiple initiator matrices (K_1', K_1'', K_1''') (even of different sizes)

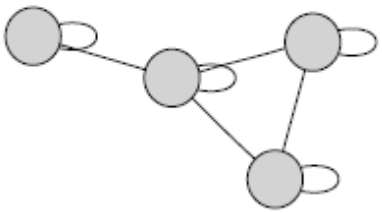
[PKDD '05]

Kronecker Initiator Matrices

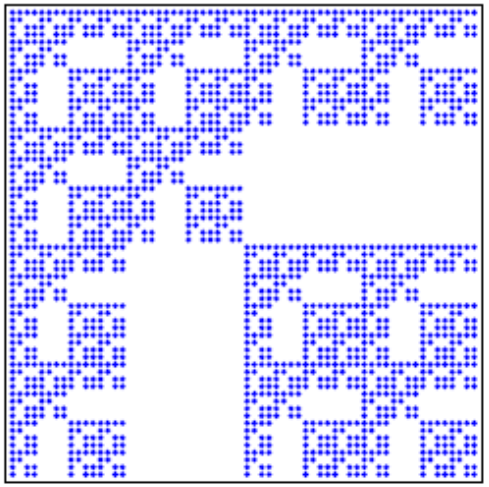


1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1





1	1	1	1
1	1	0	0
1	0	1	1
1	0	1	1



Initiator K_1 | K_1 adjacency matrix | K_3 adjacency matrix

[PKDD '05]

Kronecker Graphs: First Fun Fact

$$K_1^{[m]} = K_m = \underbrace{K_1 \otimes K_1 \otimes \ldots K_1}_{m \text{ times}} = K_{m-1} \otimes K_1$$

First fact about Kronecker Graphs!

- For $\boldsymbol{K_1}$ on $\boldsymbol{N_1}$ nodes and $\boldsymbol{E_1}$ edges $\boldsymbol{K_m}$ (m^{th} Kronecker power of $\boldsymbol{K_1}$) has:
  - $\boldsymbol{N(m) = N_1^m}$ nodes
  - $\boldsymbol{E(m) = E_1^m}$ edges
- So, we get the **densification power-law!**
  - $\boldsymbol{E(t) \propto N(t)^a}$, **so:** $E_1^t = (N_1^t)^a$ What is $\boldsymbol{a}$?
  - $\boldsymbol{a = \frac{\log(E(t))}{\log(N(t))} = \frac{log(E_1^t)}{log(N_1^t)} = \frac{\log(E_1)}{\log(N_1)}}$

1	1	0
1	1	1
0	1	1

K_1

Since E(t) > N(t), then $\boldsymbol{a > 1}$

[PKDD '05]

Properties of Kronecker Graphs

- **Properties of deterministic Kronecker graphs (can be proved!)**
  - **Properties of static networks:**
    - Power-Law like Degree Distribution
    - Power-Law eigenvalue and eigenvector distribution
    - Constant Diameter
  - **Properties of evolving networks:**
    - Densification Power Law **(just proved)**
    - Shrinking/Stabilizing Diameter (for Stochastic Kronecker graphs)

[PKDD '05]

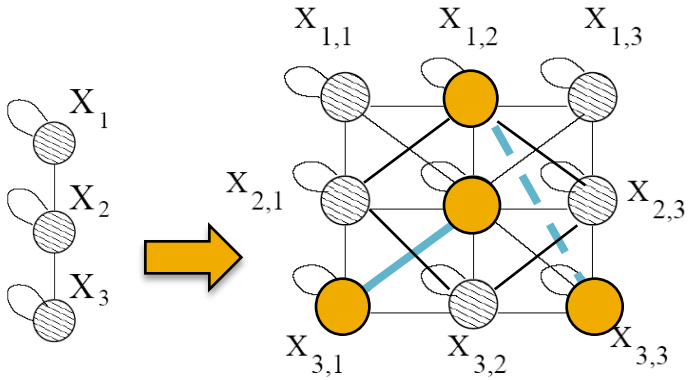
Constant Diameter

- **Observation:** Edges in Kronecker graphs:

$$Edge\ (X_{ij}, X_{kl}) \in G \otimes H$$

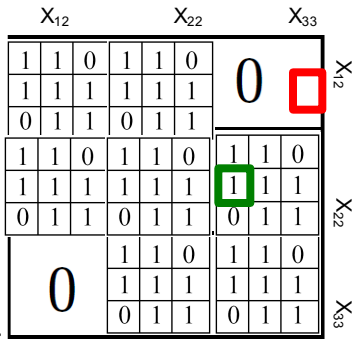
$$iff\ (X_i, X_k) \in G\ and\ (X_j, X_l) \in H$$

where $\boldsymbol{X}$ are appropriate nodes in $\boldsymbol{G}$ and $\boldsymbol{H}$



- **Why?**
  - An entry in matrix **G⊗H** is a multiplication of entries in $\boldsymbol{G}$ and $\boldsymbol{H}$.

1	1	0
1	1	1
0	1	1



Constant Diameter

$$Edge\ (X_{ij}, X_{kl}) \in G \otimes H$$
$$iff\ (X_i, X_k) \in G\ and\ (X_j, X_l) \in H$$

[PKDD '05]

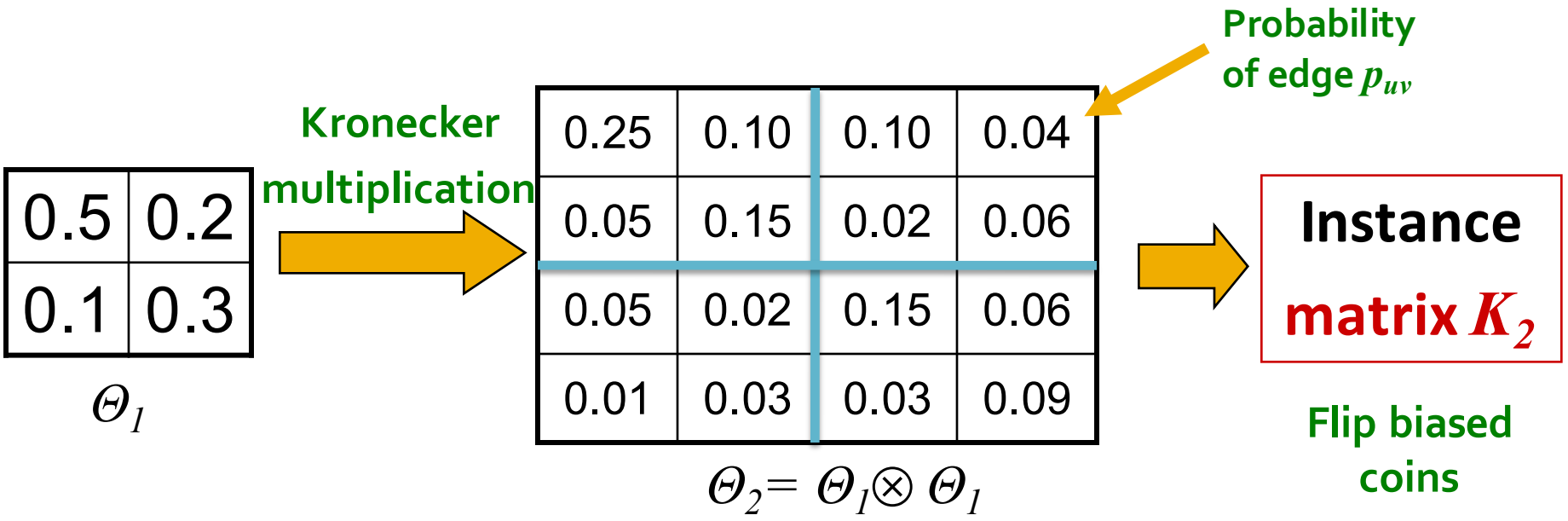
- **Theorem:** **Constant diameter:** If graphs G, H have diameter d then $G \otimes H$ has diameter d
- **What is distance between nodes u, v in $G \otimes H$?**
  - Consider some nodes $u = [a, b], v = [a', b']$ in $G \otimes H$
  - Then, path a to a' in G is less d steps: $a_1, a_2, a_3, \ldots, a_d$
  - And path b to b' in H is less d steps: $b_1, b_2, b_3, \ldots, b_d$
  - **How many steps from u to v?**
    - **We know** edge $([a_1, b_1], [a_2, b_2])$ is in $G \otimes H$
    - So it takes $< d$ steps to get from u to v in $G \otimes H$
- **Consequence:**
  - If K_1 has diameter d then graph K_k also has diameter d

Stochastic Kronecker Graphs

[PKDD '05]

Stochastic Kronecker Graphs

- Create $N_1 \times N_1$ **probability matrix** $\boldsymbol{\Theta_1}$
- Compute the k^{th} Kronecker power $\boldsymbol{\Theta_k}$
- For each entry p_{uv} of $\boldsymbol{\Theta_k}$ include an edge (u, v) in K_k with probability p_{uv}



Generation of Kronecker Graphs

- **How do we generate an instance of a stochastic Kronecker graph?**

Probability of edge p_{uv}

0.25	0.10	0.10	0.04
0.05	0.15	0.02	0.06
0.05	0.02	0.15	0.06
0.01	0.03	0.03	0.09

Flip biased coins

1	1	0	0
0	1	0	1
1	0	1	1
0	1	0	1

Need to flip n^2 coins!! Way too slow!!

- **Is there a faster way? YES!**
- **Idea:** Exploit the recursive structure of Kronecker graphs
  - "Drop" edges one by one

Generation of Kronecker Graphs

- **A faster way to generate Kronecker graphs**

$\Theta =$

a	b
c	d

	v_1	v_2	v_3	v_4
v_1	a·a	a·b	b·a	b·b
v_2	a·c	a·d	b·c	b·d
v_3	c·a	c·b	d·a	d·b
v_4	c·c	c·d	d·c	d·d

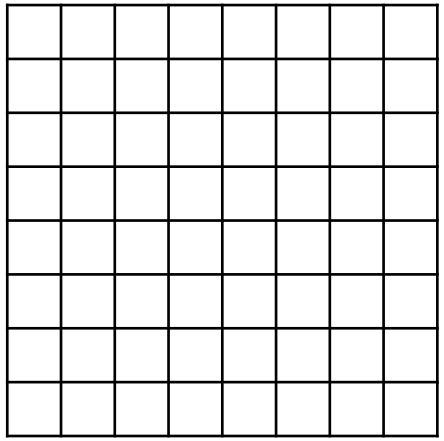
$\Theta \otimes \Theta$

=

	v_1	v_2	v_3	v_4
v_1	a	b	a	b
v_2	c	d	c	d
v_3	a	b	a	b
v_4	c	d	c	d

a **b** **c** **d**

- **How to "drop" an edge into a graph G on $n = 2^m$ nodes**



Adjacency matrix **G**

Generation of Kronecker Graphs

- **A faster way to generate Kronecker graphs**

$\Theta =$

a	b
c	d

	v_1	v_2	v_3	v_4
v_1	a·a	a·b	b·a	b·b
v_2	a·c	a·d	b·c	b·d
v_3	c·a	c·b	d·a	d·b
v_4	c·c	c·d	d·c	d·d

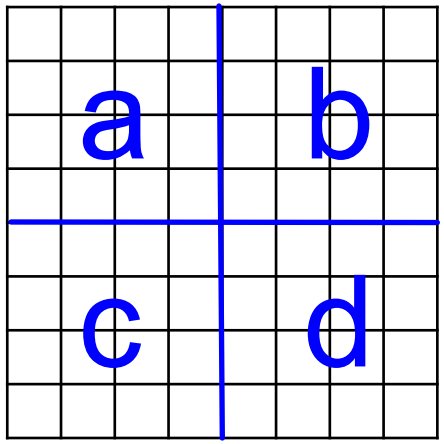
$\Theta \otimes \Theta$

=

	v_1	v_2	v_3	v_4
v_1	a	b	a	b
v_2	c	d	c	d
v_3	a	b	a	b
v_4	c	d	c	d

(**a**, **b**, **c**, **d** mark the corresponding Kronecker blocks)

- **How to "drop" an edge into a graph G on $n = 2^m$ nodes**



Adjacency matrix **G**

Generation of Kronecker Graphs

- **A faster way to generate Kronecker graphs**

$\Theta =$

a	b
c	d

	v_1	v_2	v_3	v_4
v_1	a·a	a·b	b·a	b·b
v_2	a·c	a·d	b·c	b·d
v_3	c·a	c·b	d·a	d·b
v_4	c·c	c·d	d·c	d·d

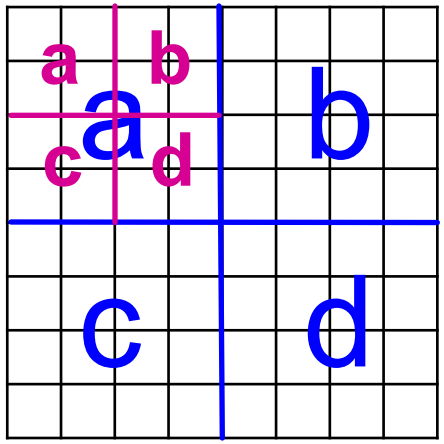
$\Theta \otimes \Theta$

=

	v_1	v_2	v_3	v_4
v_1	a	b	a	b
v_2	c	d	c	d
v_3	a	b	a	b
v_4	c	d	c	d

(a, b, c, d)

- **How to "drop" an edge into a graph G on $n = 2^m$ nodes**



Adjacency matrix **G**

Generation of Kronecker Graphs

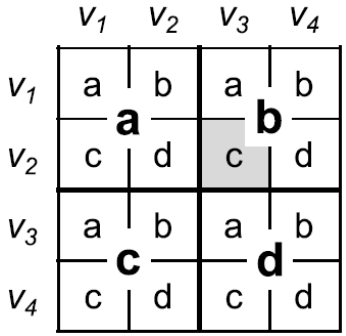
- **A faster way to generate Kronecker graphs**

$\Theta =$ $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

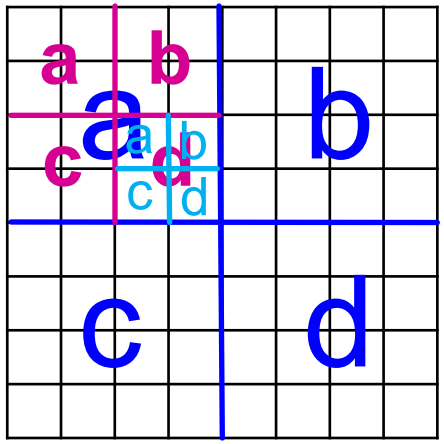
A faster way to generate Kronecker graphs

- How to "drop" an edge into a graph G on $n = 2^m$ nodes:
  - We may get a few edges colliding. We simply reinsert them.

$\Theta \otimes \Theta$ =



- **How to "drop" an edge into a graph G on $n = 2^m$ nodes:**
  - We may get a few edges colliding. We simply reinsert them.

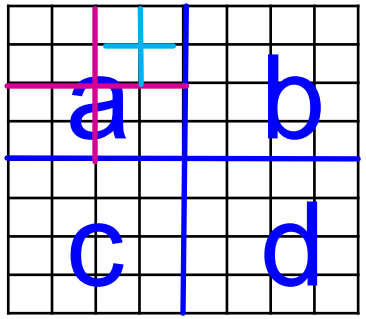


Adjacency matrix **G**

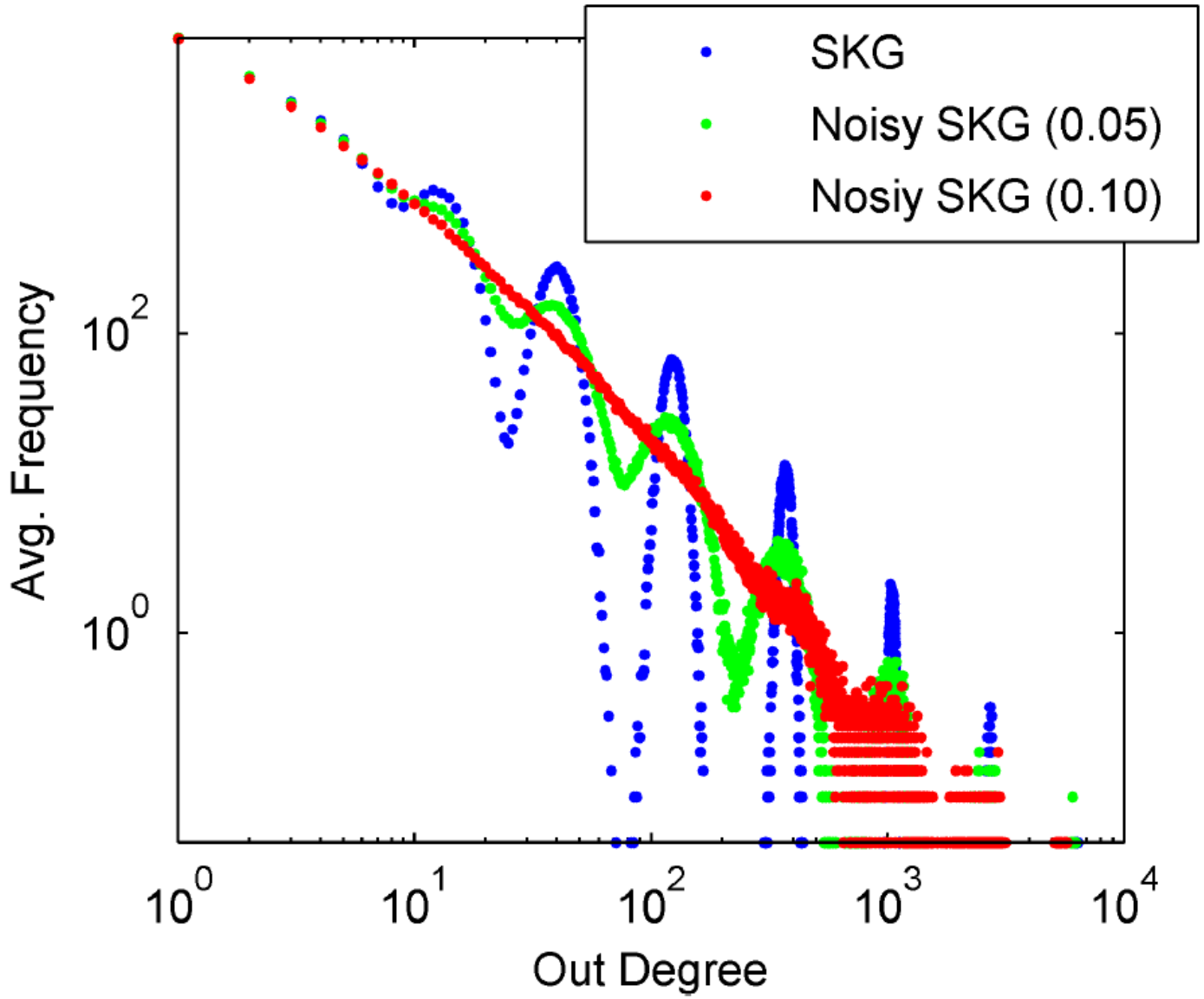
Generation of Kronecker Graphs

Fast Kronecker generator algorithm:

- **Insert 1 edge on graph G on $n = 2^m$ nodes:**
  - Create normalized matrix $L_{uv} = \Theta_{uv} / (\sum_{op} \Theta_{op})$
  - **For $i = 1 \ldots m$**
    - start with $x = 0, y = 0$
    - Pick an row/column (u, v) with prob. L_{uv}
    - Descend into quadrant (u, v) at level i of G
      - **This means:** $x \mathrel{+}= u \cdot 2^{m-i}$, $y \mathrel{+}= v \cdot 2^{m-i}$
  - Add an edge $G[x, y] = 1$



Problem: Spikes in Node Degrees!



[Seshadhri et al. '13]

Solution: Noisy SKG

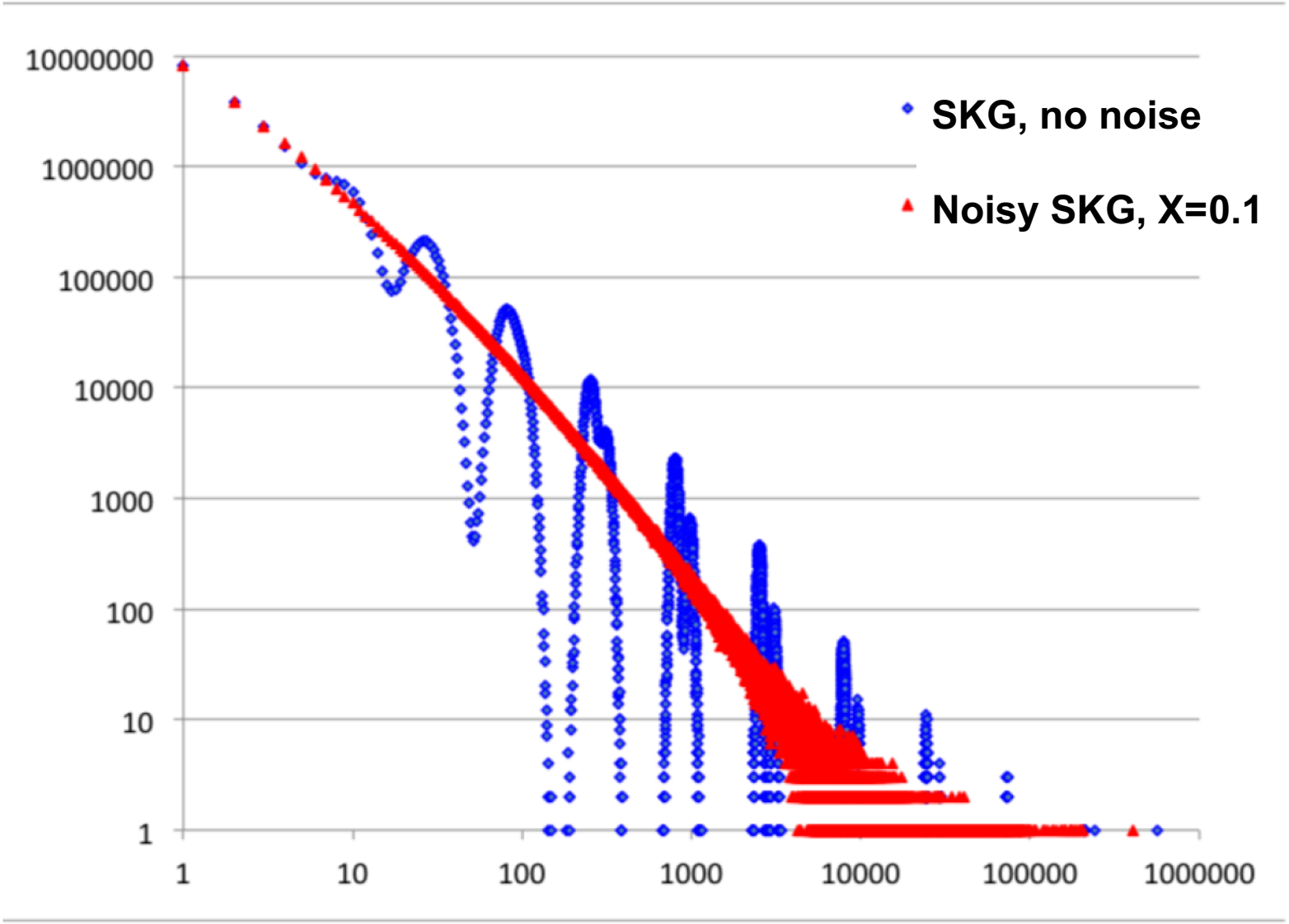
- **Solution: Noisy Stochastic Kronecker Graphs**
  - **Idea:** Add noise to the matrix Θ
    - There are many ways how one could do this, but here is the correct way!
  - Assume $\boldsymbol{\Theta} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ and **G** has $\mathbf{2}^{\boldsymbol{m}}$ nodes
  - **Then create "noisy" matrices** $\Theta_1 \ldots \Theta_m$ where:
  - $\Theta_i = \begin{bmatrix} a - \frac{2x_i a}{a+d} & b + x_i \\ c + x_i & d - \frac{2x_i d}{a+d} \end{bmatrix}$

Where x_i is a random number on interval [-X, +X]
And X is the noise level.

  - Apply Kronecker generator to this set of matrices

[Seshadhri et al. '13]

SKG Degree Distribution



[Mahdian-Xu, WAW '07]

Stochastic Kronecker Graphs

What is known about Stochastic Kronecker?

- **Undirected** Kronecker graph model with:
  - **Connected**, if:
    - $b + c > 1$
  - **Connected component of size $\Theta(n)$**, if:
    - $(a + b)(b + c) > 1$
  - **Constant diameter**, if:
    - $b + c > 1$
  - **Not searchable** by a decentralized algorithm

$$\Theta_1 = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$$

$a > b > c$
("undirected" Kronecker)

[ICML '07]

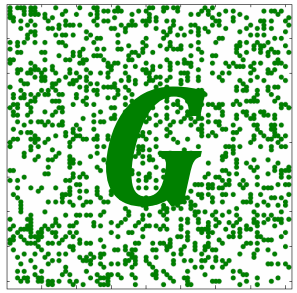
Kronecker Graphs: Estimation

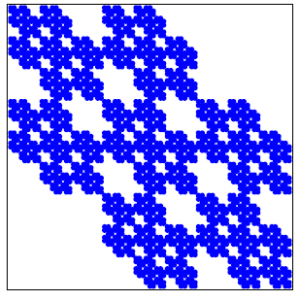
$\Theta = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$

How to estimate Θ given a G?

- **KronFit: Maximum likelihood estimation**
- Given real graph G
- Find Stochastic Kronecker initiator Θ which

$$\arg\max_{\Theta} P(G \mid \Theta)$$



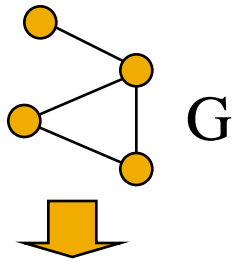


Kronecker

- **To solve this we need to:**
  - Efficiently calculate $\boldsymbol{P(G|\Theta)}$
  - Then maximize over $\boldsymbol{\Theta}$ (e.g., using gradient descent)

KronFit: Likelihood $P(G|\Theta)$

- Given G and Θ we calculate likelihood that Θ generated G: $P(G|\Theta)$



0.5	0.2
0.1	0.3

Θ

0.25	0.10	0.10	0.04
0.05	0.15	0.02	0.06
0.05	0.02	0.15	0.06
0.01	0.03	0.03	0.09

Θ_k

1	0	1	1
0	1	0	1
1	0	1	1
1	1	1	1

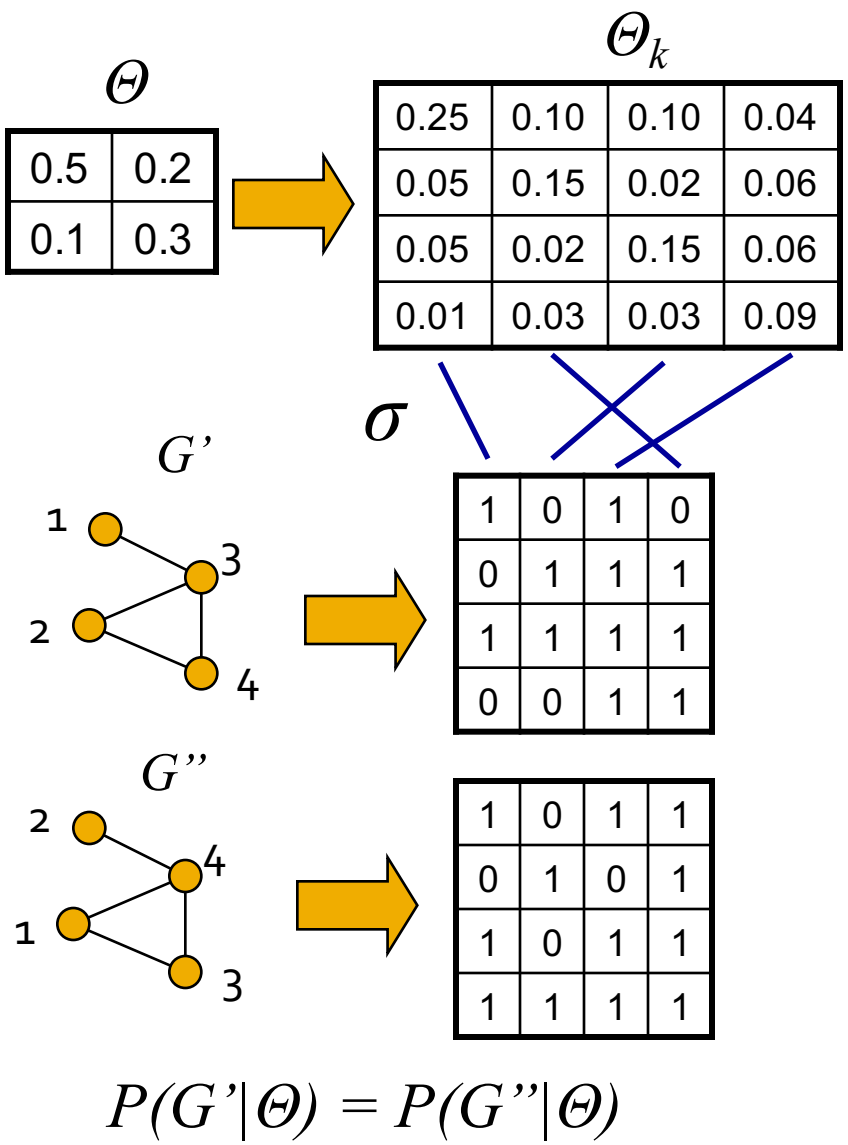
G

$P(G|\Theta)$

$$P(G \mid \Theta) = \prod_{(u,v)\in G} \Theta_k[u,v] \prod_{(u,v)\notin G} (1 - \Theta_k[u,v])$$

Likelihood of edges in the graph Likelihood of edges not in the graph

Challenge 1: Node Correspondence



- Nodes are **unlabeled**
- Graphs ***G'*** and ***G''*** should have the same likelihood
  $\boldsymbol{P(G'|\Theta) = P(G''|\Theta)}$
- One needs to consider all node correspondences $\boldsymbol{\sigma}$

$$P(G|\Theta) = \sum_{\sigma} P(G|\Theta,\sigma)P(\sigma)$$

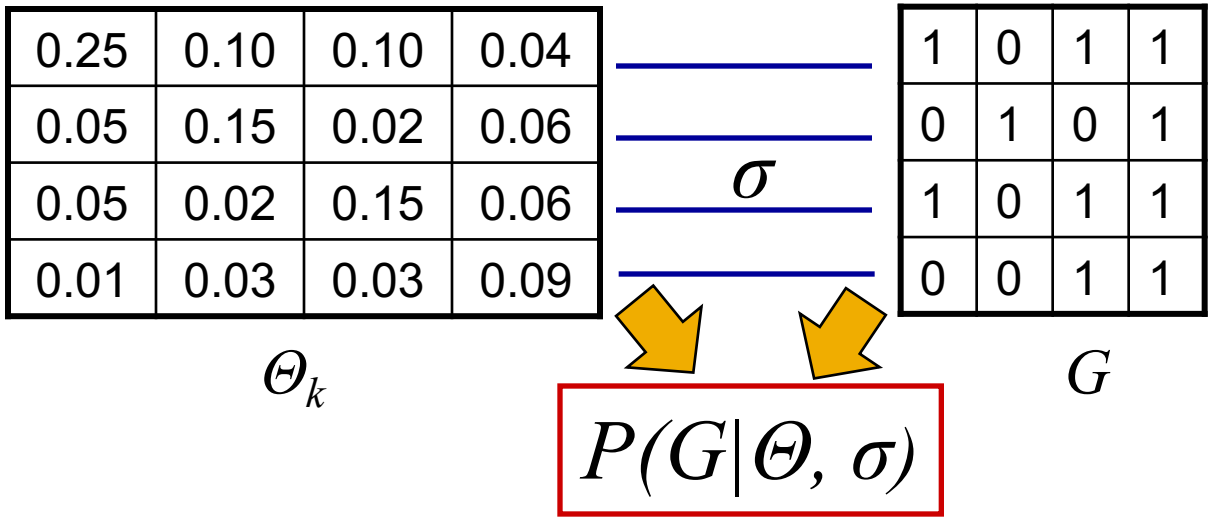
- All correspondences are a priori equally likely
- There are ***O(n!)*** correspondences

Challenge 2: Calculating $P(G|\Theta,\sigma)$

- **Assume that we solved the node correspondence problem**
- **Calculating:**

$$P(G \mid \Theta) = \prod_{(u,v)\in G} \Theta_k[u,v] \prod_{(u,v)\notin G} (1 - \Theta_k[u,v])$$

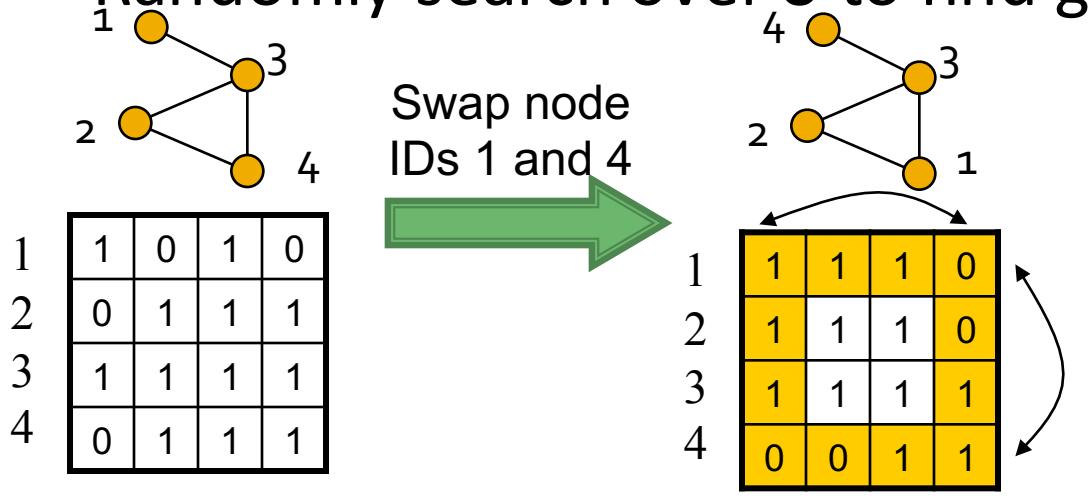
- Takes $O(n^2)$ time!



KronFit: Solution Overview

Details in Leskovec-Faloutsos, ICML '07

- **Node correspondence:**
  - Node permutation **σ** defines the mapping
  - Randomly search over **σ** to find good mappings



The algorithm (called Metropolis sampling):
(1) Pick 2 nodes at random
(2) Swap their IDs
(3) Does it improve the fit $P(G|\Theta, \sigma)$? **If, yes, keep the swap, else undo it**
(4) Go to (1)

- **Calculating the likelihood $P(G|\Theta,\sigma)$**
  - Calculate likelihood of **empty graph** (**G** with **0** edges)
  - Correct it for edges that we observe in the graph

Solution 1: Node correspondence

Details!

- **Log-likelihood**

$$l(\Theta) = \log \sum_{\sigma} P(G|\Theta, \sigma) P(\sigma)$$

- **Gradient of log-likelihood**

$$\frac{\partial}{\partial \Theta} l(\Theta) = \sum_{\sigma} \frac{\partial \log P(G|\sigma, \Theta)}{\partial \Theta} P(\sigma|G, \Theta)$$

- Sample the permutations from $P(\sigma|G,\Theta)$ and average the gradients

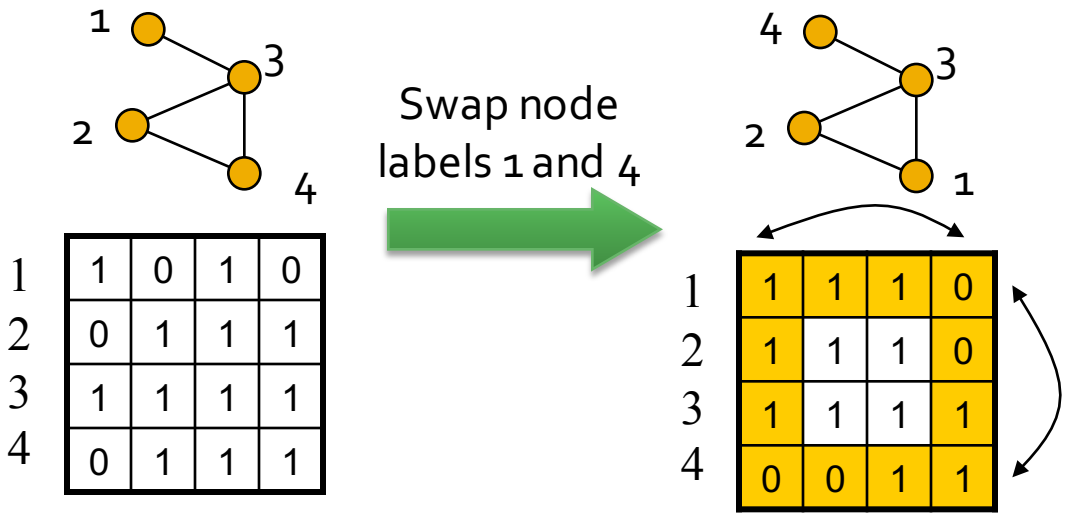
See Leskovec-Faloutsos, ICML '07 for details

Solution 1: Node correspondence

Details!

- **Metropolis sampling:**
  - Start with a random permutation **σ**
  - **σ'** = swap two elements in permutation **σ**
  - Accept the new permutation **σ'**
    - If new permutation is better (gives higher likelihood)
    - Else accept with prob. proportional to the ratio of likelihoods (**no need to calculate the normalizing constant!**)

$$\frac{P(\sigma'|G,\Theta)}{P(\sigma|G,\Theta)}$$



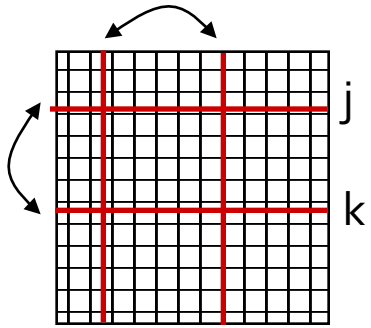
See Leskovec-Faloutsos, ICML '07 for details

Sampling node labelings (2)

Details!

$\sigma^{(0)} := (1, \ldots, N)$
repeat
  Draw j and k uniformly from $(1, \ldots, N)$
  $\sigma^{(i)} := \texttt{SwapElements}(\sigma^{(i-1)}, j, k)$
  Draw u from $U(0,1)$
  if $u > \frac{P(\sigma^{(i)}|G,\Theta)}{P(\sigma^{(i-1)}|G,\Theta)}$ **then**
    $\sigma^{(i)} := \sigma^{(i-1)}$
  end if
  i = i + 1
until $\sigma^{(i)} \sim P(\sigma|G,\Theta)$
return $\sigma^{(i)}$

Metropolis permutation sampling algorithm



- Need to efficiently calculate the likelihood ratios
- But the permutations $\boldsymbol{\sigma^{(i)}}$ and $\boldsymbol{\sigma^{(i+1)}}$ only differ at 2 positions
- So we only traverse to update 2 rows (columns) of $\boldsymbol{\Theta_k}$
- We can evaluate the likelihood ratio efficiently

Solution 2: Calculating $P(G|\Theta,\sigma)$

Details!

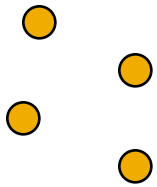
- **Problem:**
- **Calculating naively $P(G|\Theta,\sigma)$ takes $O(N^2)$**
- **Idea:**
  - First calculate likelihood of **empty graph**, a graph with **0** edges
  - Correct the likelihood for edges that we observe in the graph
- By exploiting the structure of Kronecker product we obtain **closed form** for likelihood of an empty graph

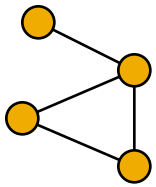
See Leskovec-Faloutsos, ICML '07 for details

Solution 2: Calculating $P(G|\Theta,\sigma)$

Details!

- **We approximate the likelihood:**





$$l(\Theta) \approx \underbrace{l_e(\Theta)}_{\text{Empty graph}} + \sum_{(u,v)\in G} \underbrace{-\log(1-\Theta_k[\sigma_u,\sigma_v])}_{\text{No-edge likelihood}} + \underbrace{\log(\Theta_k[\sigma_u,\sigma_v])}_{\text{Edge likelihood}}$$

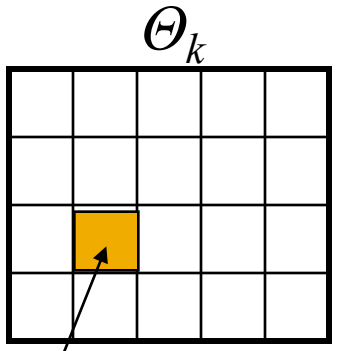
- The sum goes only over the edges
- Evaluating $\boldsymbol{P(G|\Theta,\sigma)}$ takes $\boldsymbol{O(E)}$ time
- Real graphs are sparse, $\mathbf{E} \ll \boldsymbol{N^2}$

See Leskovec-Faloutsos, ICML '07 for details

Calculating $P(G|\Theta,\sigma)$

Details!

- Real graphs are sparse so we first calculate likelihood of empty graph
- Probability of edge (i,j) is in general $p_{ij} = \theta_1^a \theta_2^b \theta_3^c \theta_4^d$
- By using Taylor approximation to p_{ij} and summing the multinomial series we obtain:



$$l_e(\Theta) = \sum_{i,j=1}^{N} \log(1 - p_{ij}) \approx -\left(\sum_{i,j=1}^{N_1} \theta_{i,j}\right)^k - \frac{1}{2}\left(\sum_{i,j=1}^{N_1} \theta_{i,j}^2\right)^k$$

- **We approximate the likelihood:**

Taylor approximation
$\log(1-x) \sim -x - 0.5\,x^2$

$$l(\Theta) \approx \underbrace{l_e(\Theta)}_{\text{Empty graph}} + \sum_{(u,v)\in G} \underbrace{-\log(1 - \Theta_k[\sigma_u, \sigma_v])}_{\text{No-edge likelihood}} + \underbrace{\log(\Theta_k[\sigma_u, \sigma_v])}_{\text{Edge likelihood}}$$

Experiments: real networks

- **Experimental setup**
  - Given real graph G
  - Estimate parameters Θ
  - Generate synthetic graph K using Θ
  - Compare properties of graphs G and K

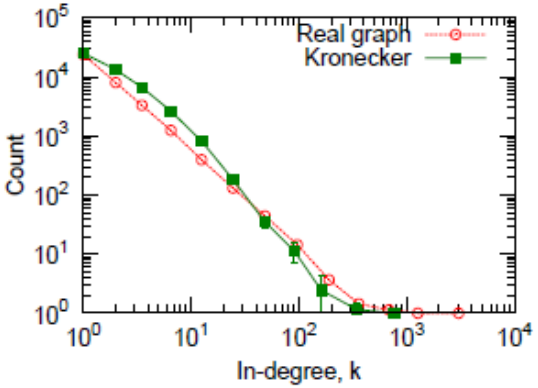
$$\Theta = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

- **Note:**
  - We do not fit the graph properties themselves
  - We fit the likelihood and then compare the properties

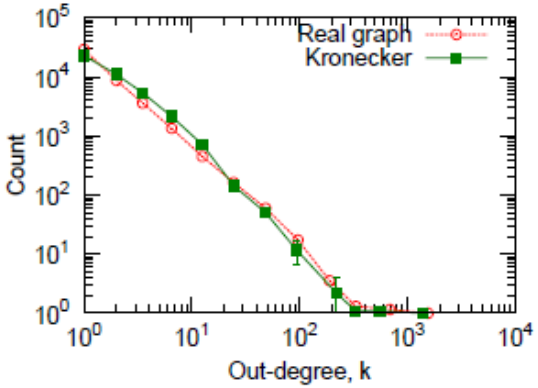
Estimation: Epinions (n=76k, m=510k)

- **Real** and **Kronecker** are very close:

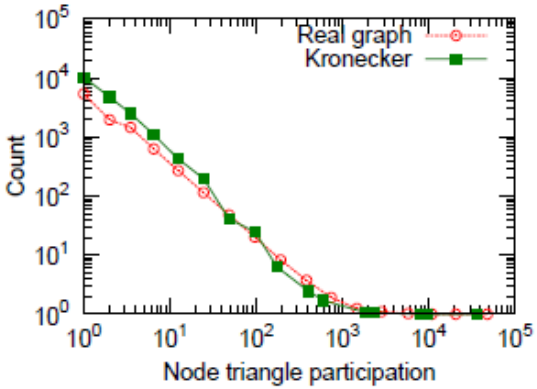
$$\Theta_1 = \begin{bmatrix} 0.99 & 0.54 \\ 0.49 & 0.13 \end{bmatrix}$$



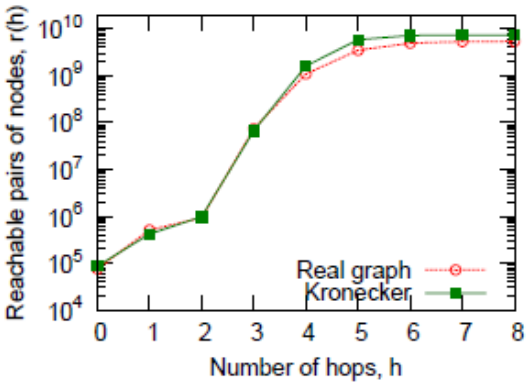
(a) In-Degree



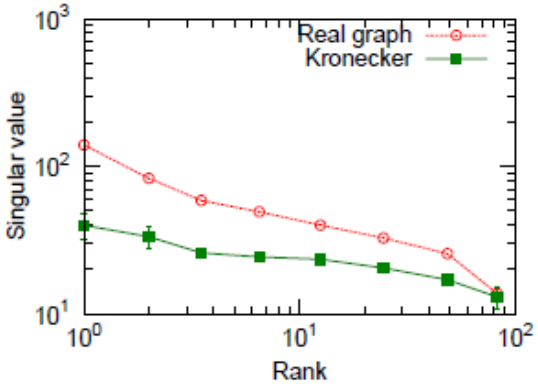
(b) Out-degree



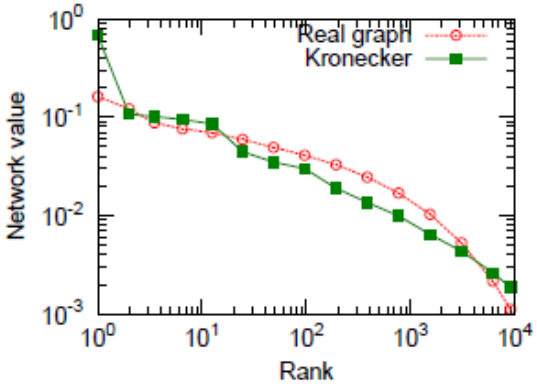
(c) Triangle participation



(d) Hop plot



(e) Scree plot

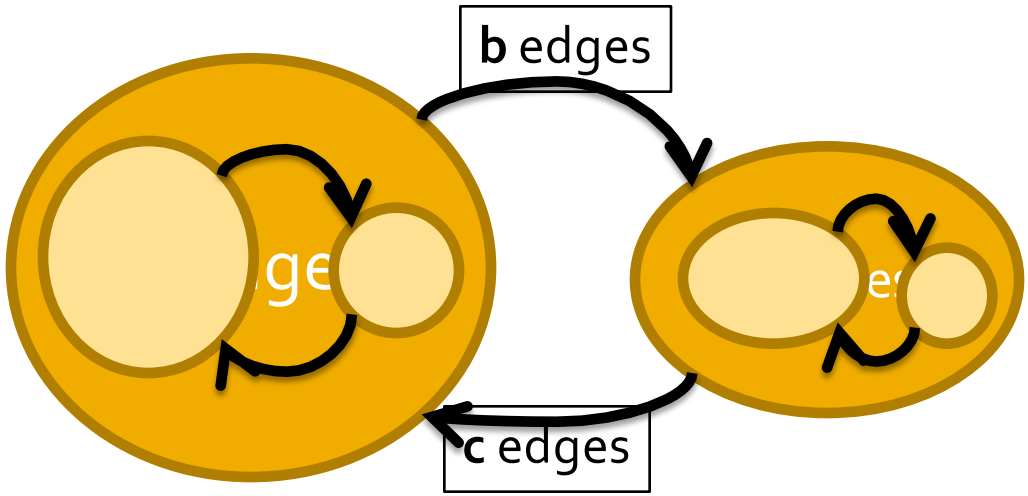


(f) "Network" value

Kronecker & Network Structure

- **What do estimated parameters tell us about the network structure?**

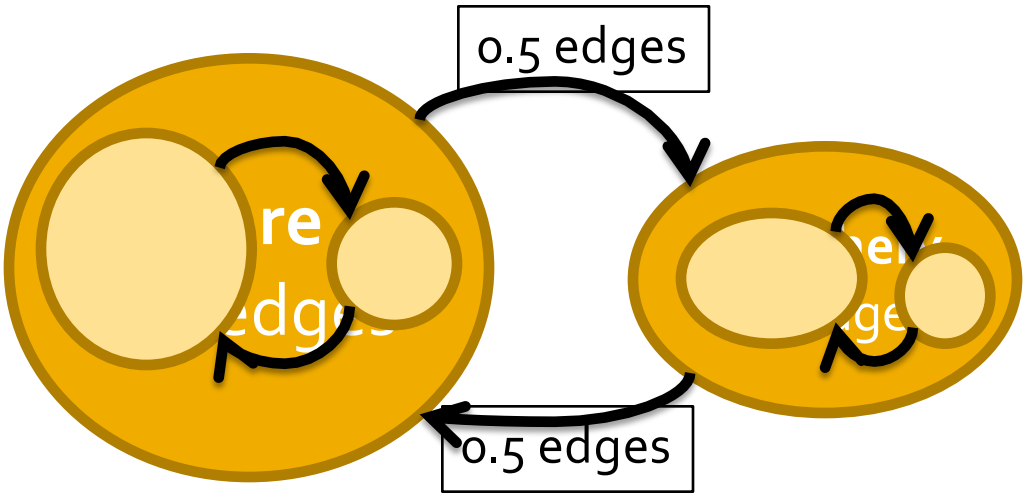
$$\Theta = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

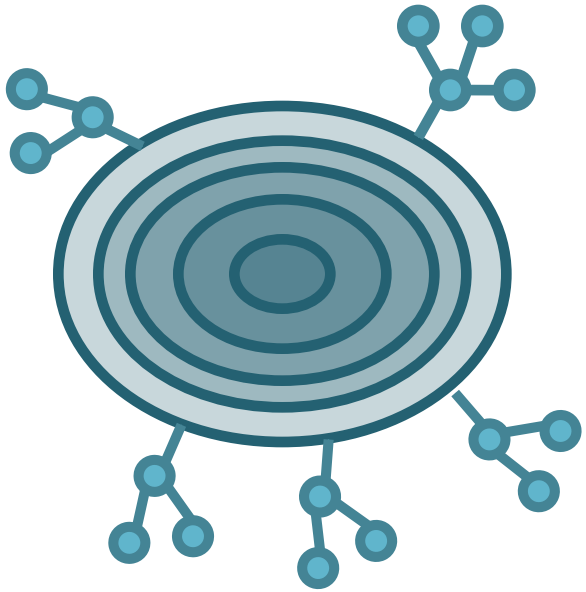


Kronecker & Network structure

- **What do estimated parameters tell us about the network structure?**

$$\Theta = \begin{bmatrix} 0.9 & 0.5 \\ 0.5 & 0.1 \end{bmatrix}$$



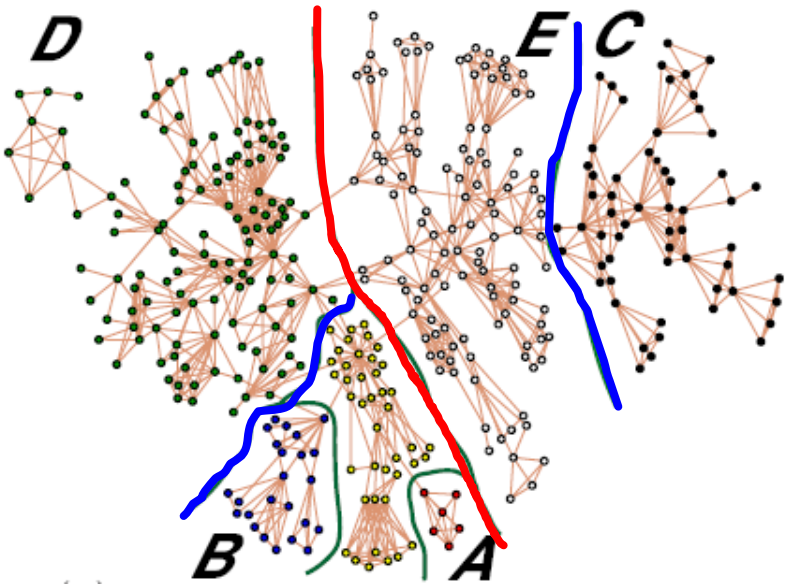


Nested Core-periphery

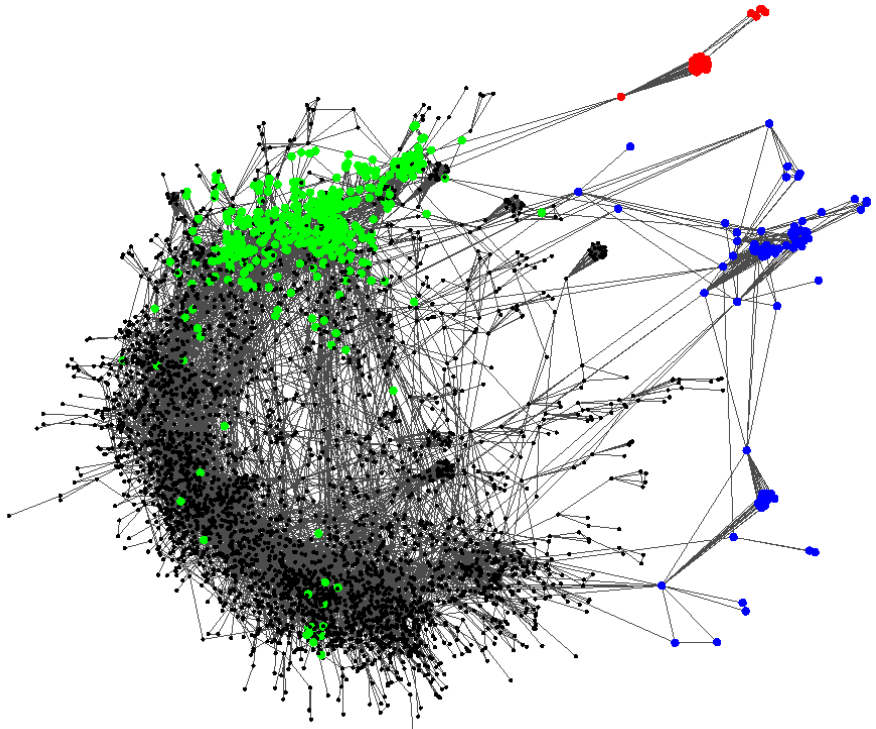
[JMLR '10]

Small vs. Large Networks

- **Small and large networks are very different:**



$$\Theta = \begin{bmatrix} 0.99 & 0.17 \\ 0.17 & 0.82 \end{bmatrix}$$

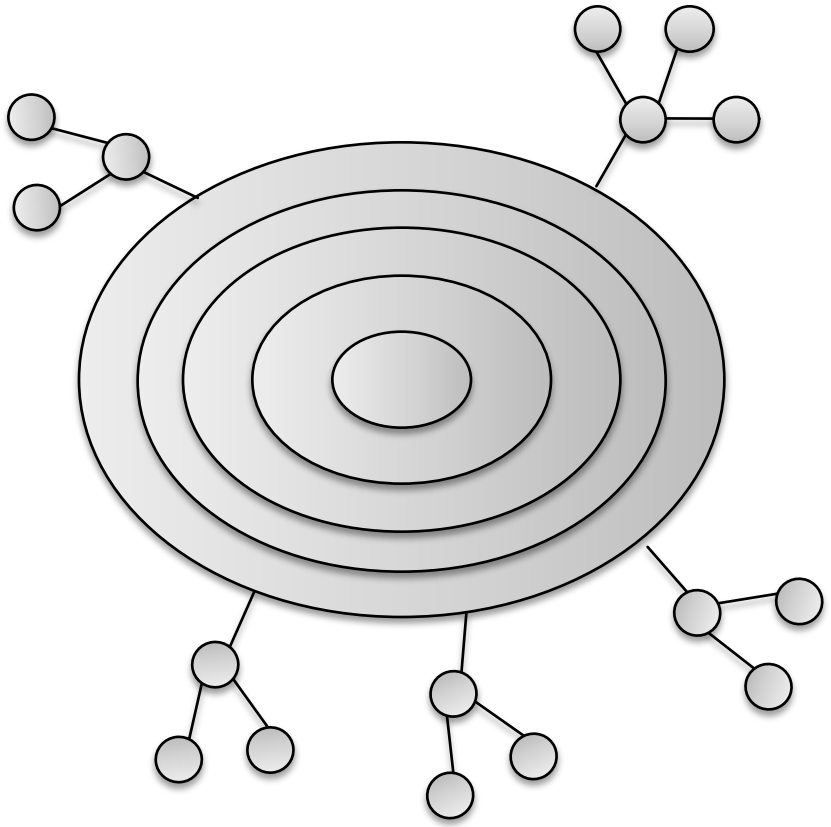


$$\Theta = \begin{bmatrix} 0.99 & 0.54 \\ 0.49 & 0.13 \end{bmatrix}$$

Implications (1)

Large scale network structure:

- **Nested Core-periphery**
  - Recursive onion-like structure of the network where each layer decomposes into a core and periphery



Implications (2)

- **Remember the SKG theorems:**
  - **Connected**, if $b+c>1$:
    - 0.55+0.15 > 1. **No!**
  - **Giant component**, if $(a+b)\cdot(b+c)>1$:
    - (0.99+0.55)·(0.55+0.15) > 1. **Yes!**

$$\Theta = \begin{bmatrix} 0.99 & 0.55 \\ 0.55 & 0.15 \end{bmatrix}$$

- Real graphs are in the in the parameter region analogous to the giant component of an **extremely sparse G_{np}**