

**Note to other teachers and users of these slides:** We would be delighted if you found this our material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. If you make use of a significant portion of these slides in your own lecture, please include this message, or a link to our web site: <http://www.mmds.org>

# Analysis of Large Graphs: Overlapping Communities

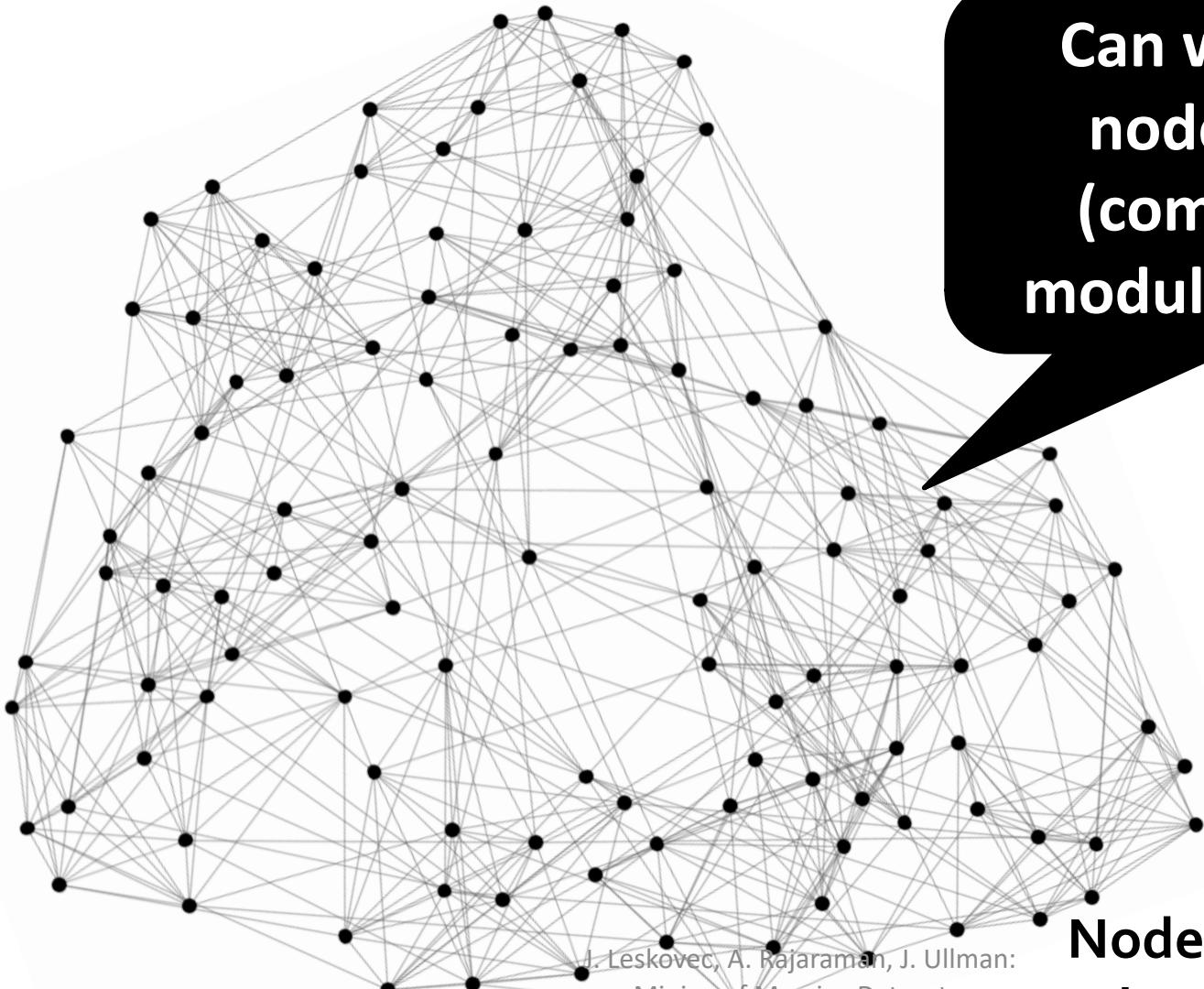
Mining of Massive Datasets

Jure Leskovec, Anand Rajaraman, Jeff Ullman Stanford University

<http://www.mmds.org>



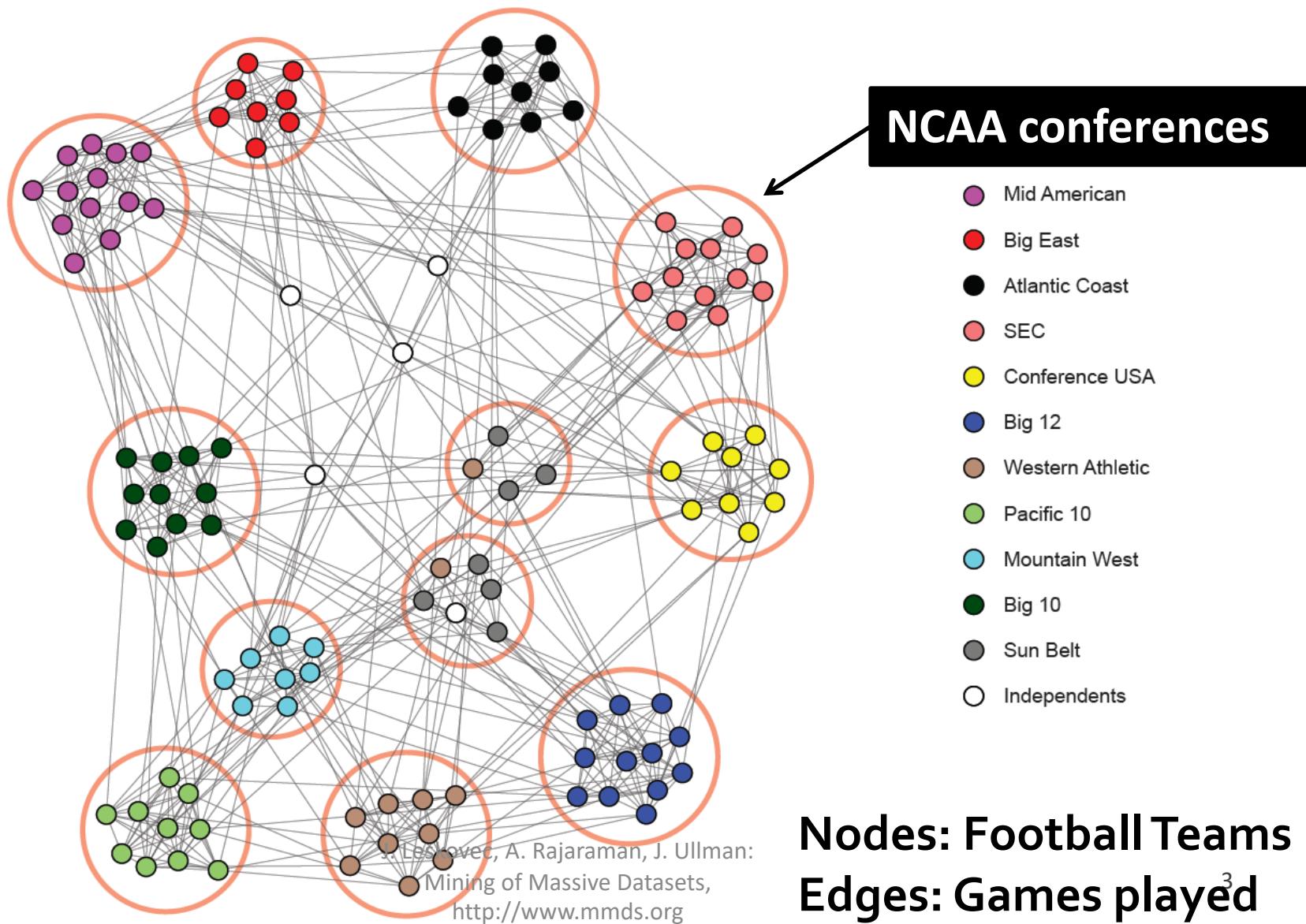
# Identifying Communities



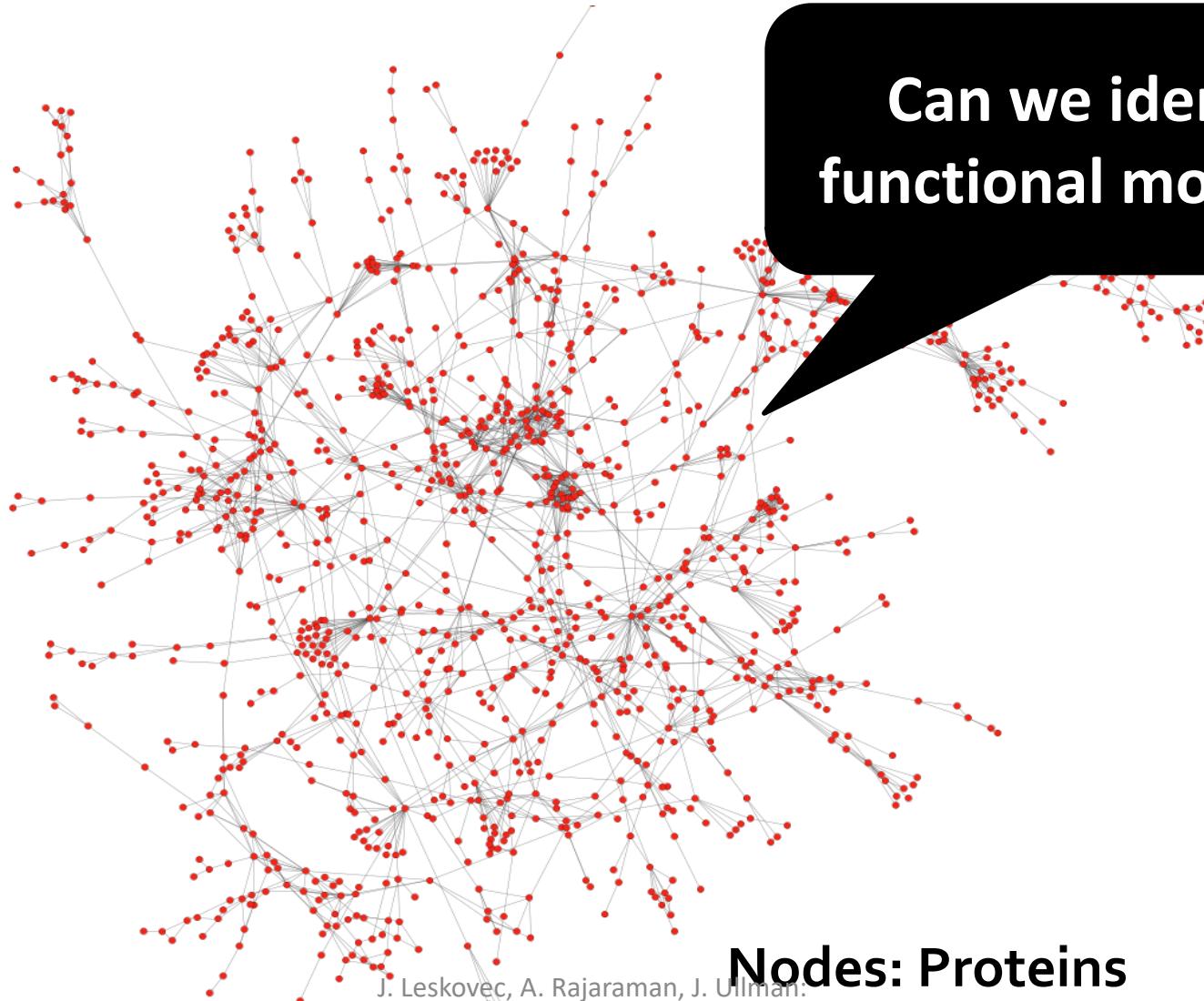
Can we identify  
node groups?  
(communities,  
modules, clusters)

Nodes: Football Teams  
Edges: Games played<sup>2</sup>

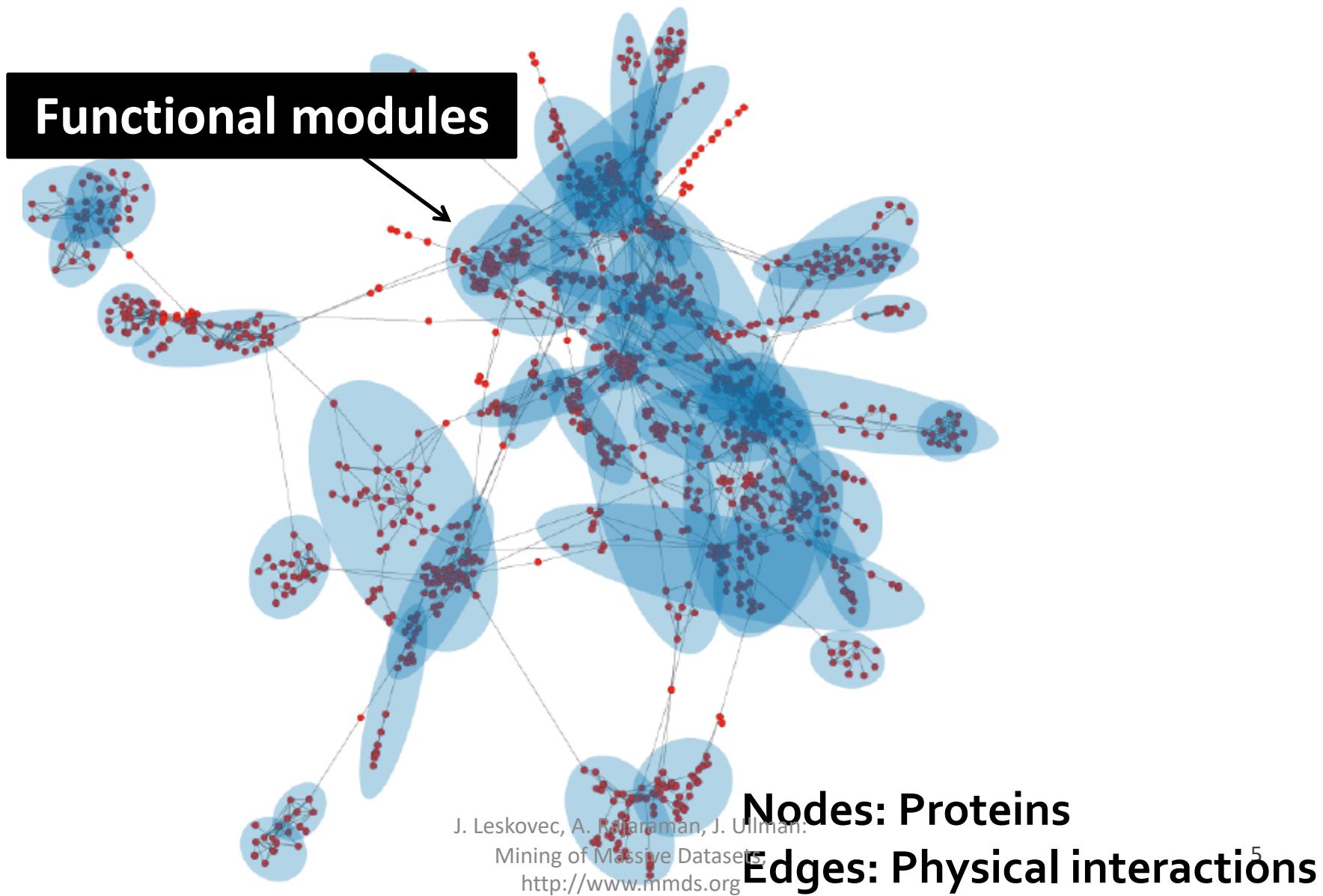
# NCAA Football Network



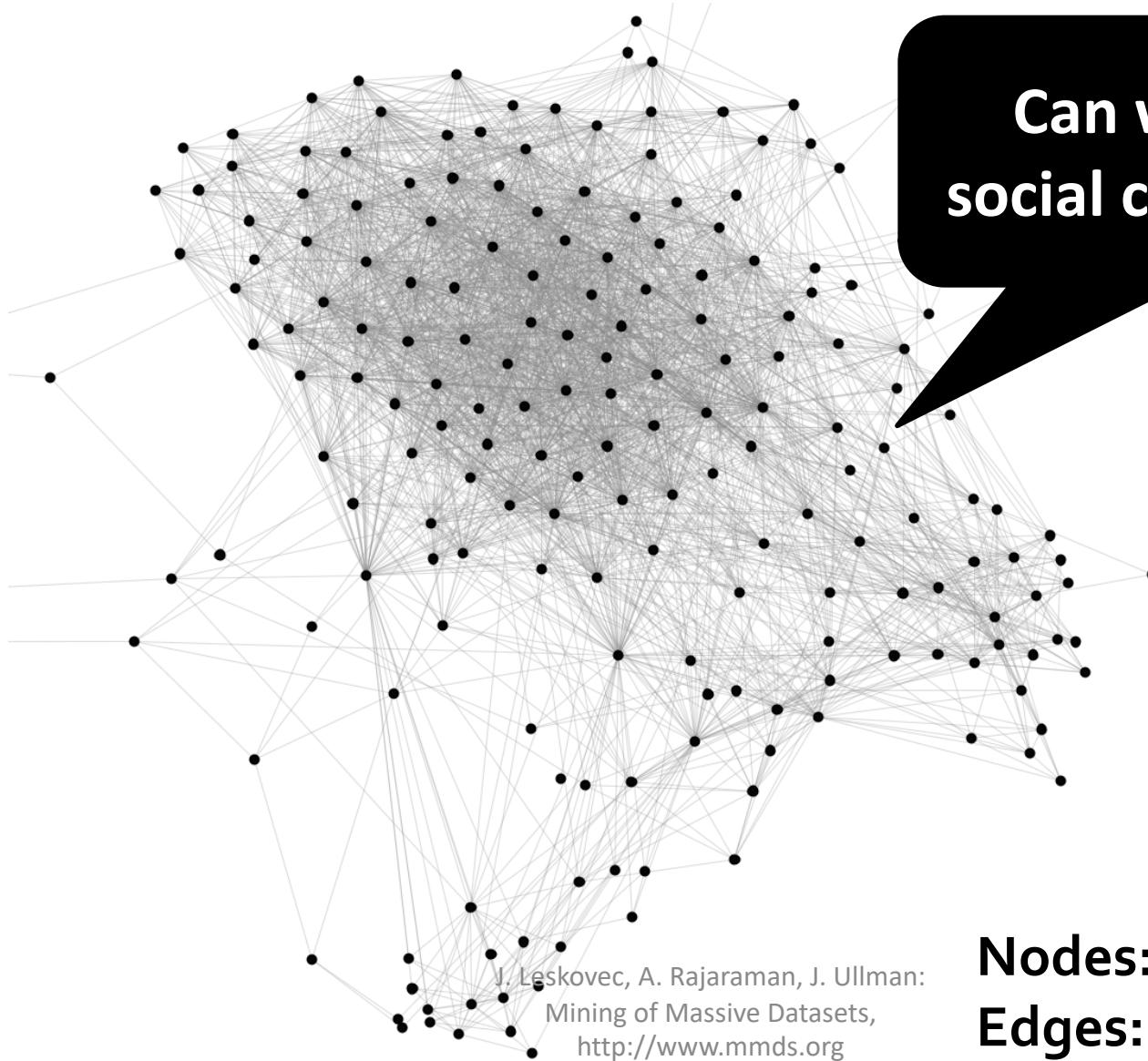
# Protein-Protein Interactions



# Protein-Protein Interactions



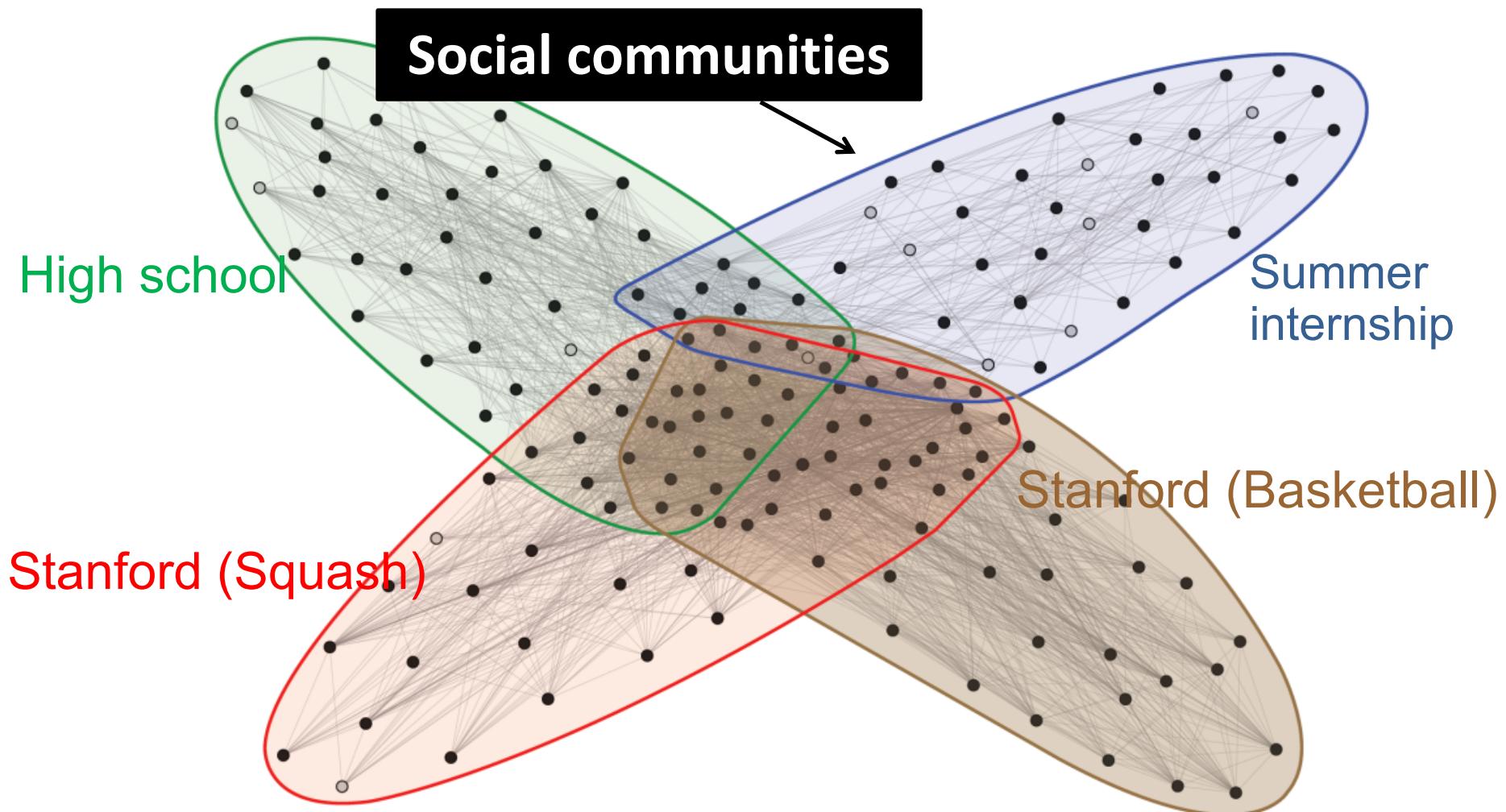
# Facebook Network



J. Leskovec, A. Rajaraman, J. Ullman:  
Mining of Massive Datasets,  
<http://www.mmds.org>

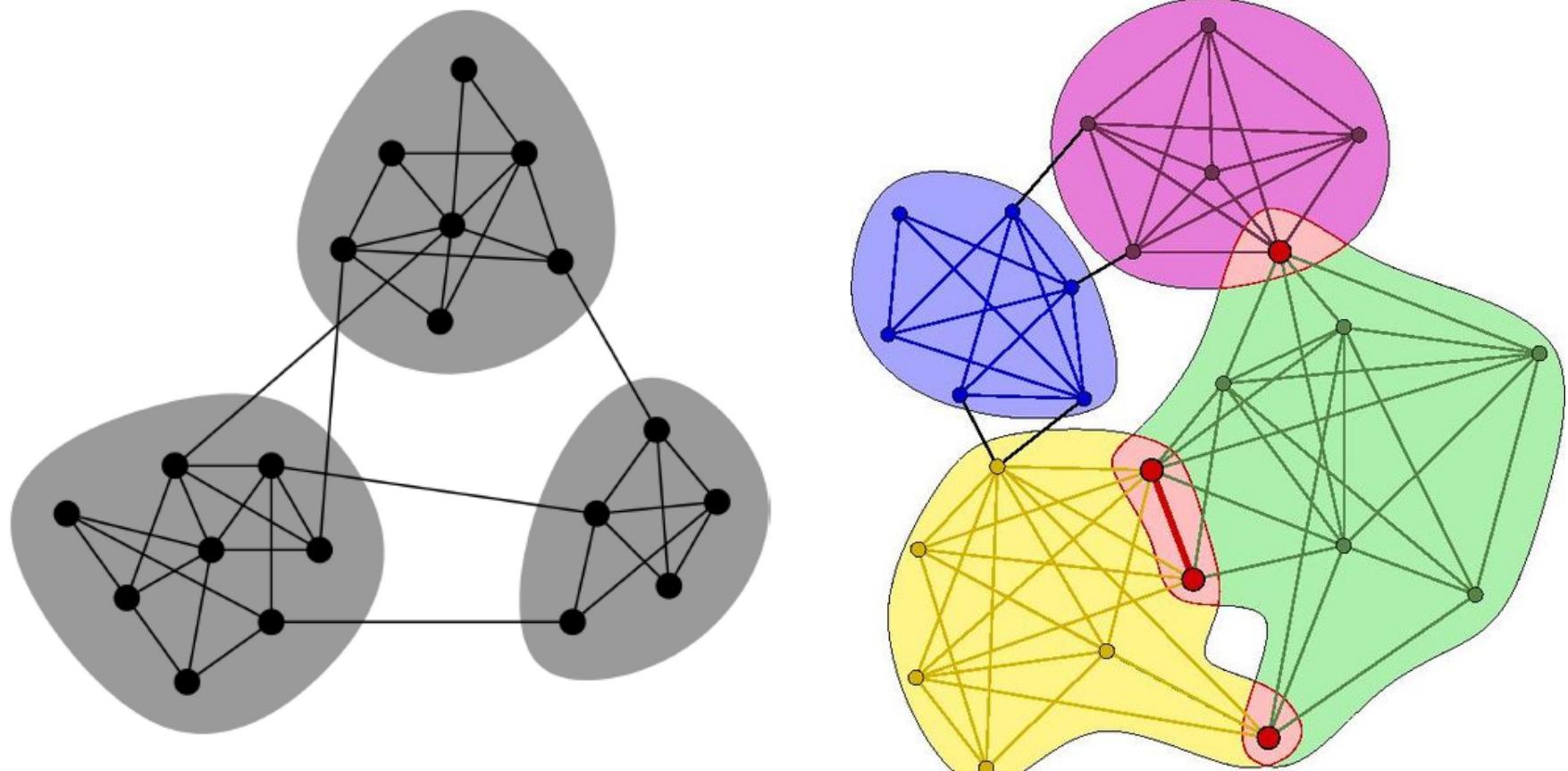
**Nodes: Facebook Users**  
**Edges: Friendships** <sup>6</sup>

# Facebook Network

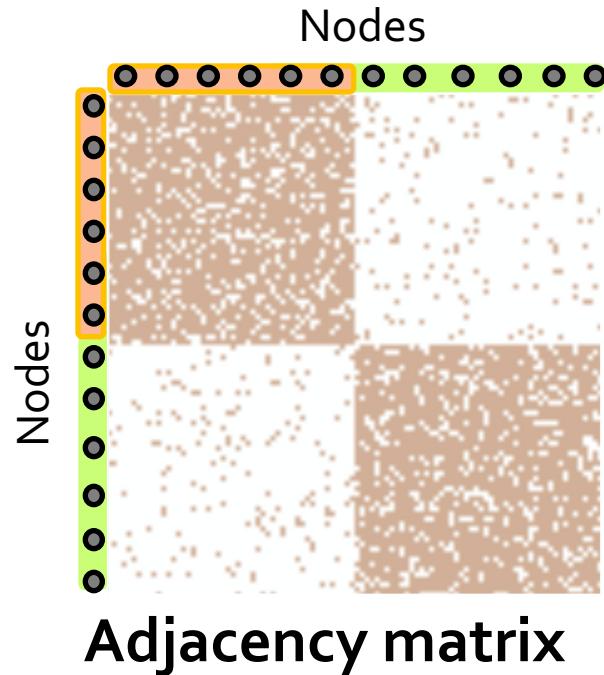
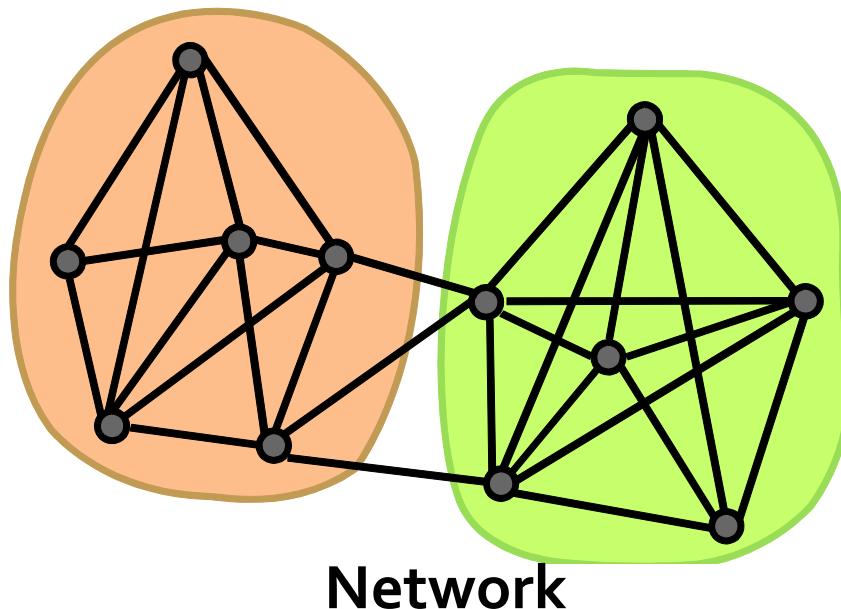


# Overlapping Communities

- Non-overlapping vs. overlapping communities

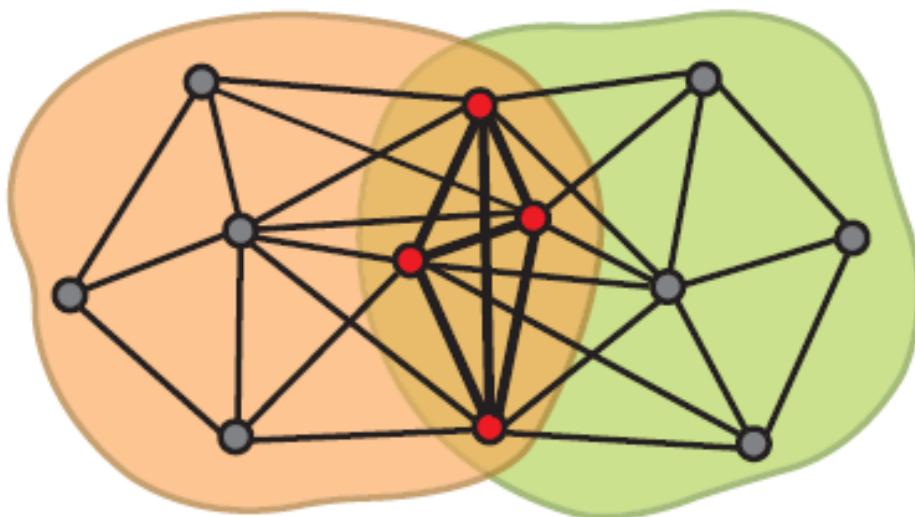


# Non-overlapping Communities



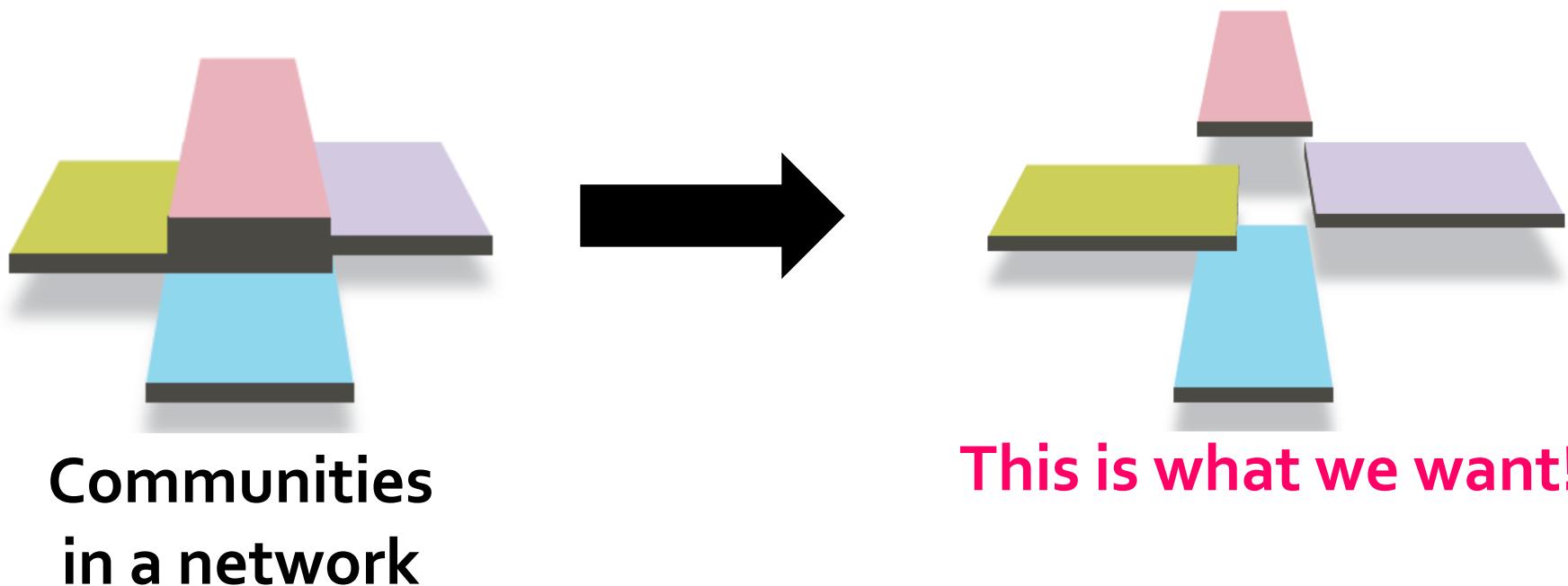
# Communities as Tiles!

- **What is the structure of community overlaps:  
Edge density in the overlaps is higher!**



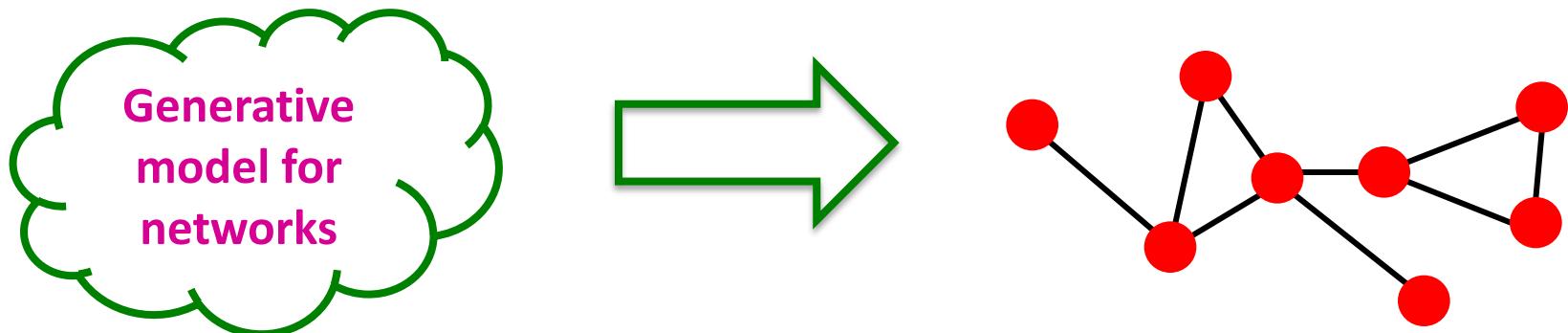
## Communities as “tiles”

# Recap so far...



# Plan of attack

- 1) Given a model, we generate the network:

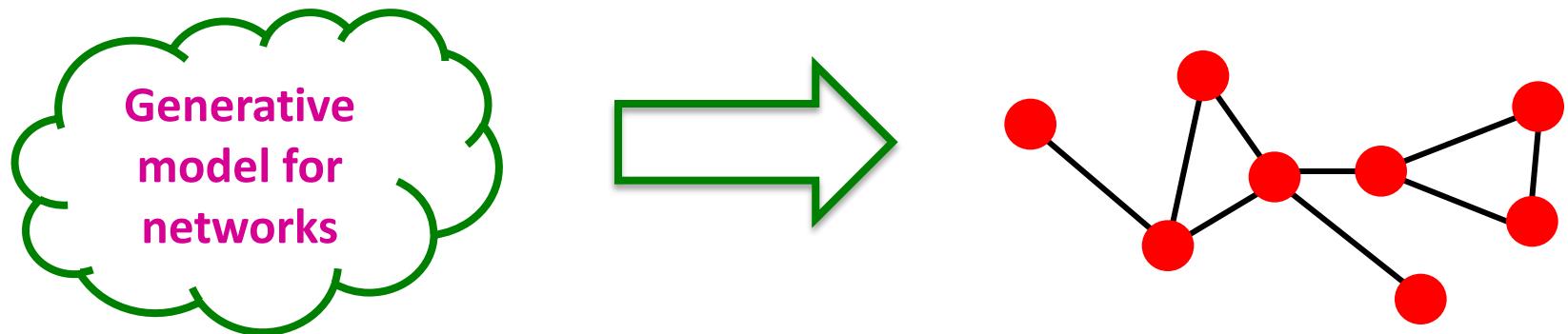


- 2) Given a network, find the “best” model



# Model of networks

- **Goal: Define a model that can generate networks**
  - The model will have a set of “parameters” that we will later want to estimate (and detect communities)



- **Q: Given a set of nodes, how do communities “generate” edges of the network?**

J. Leskovec, A. Rajaraman, J. Ullman:

Mining of Massive Datasets,  
<http://www.mmds.org>

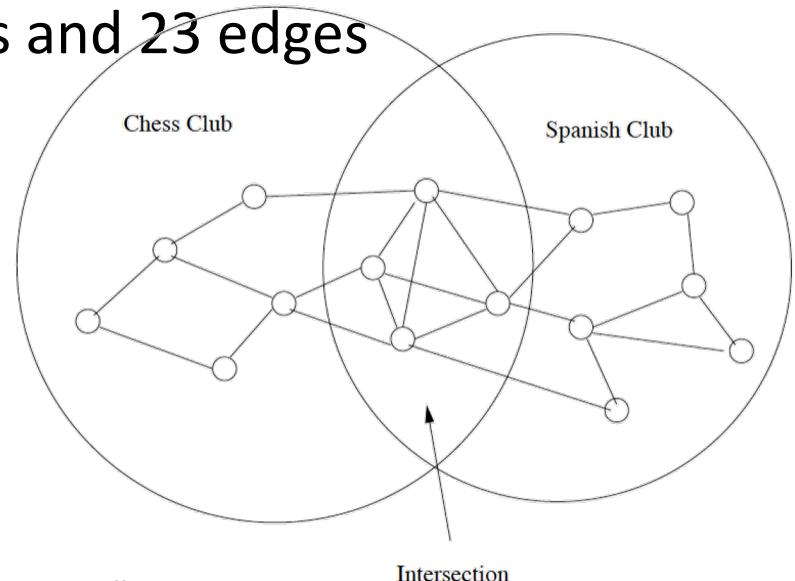
# Maximum-Likelihood Estimation

# Maximum-Likelihood Estimation

- Assume a generative process (the model )
  - for creating instances of some artifact, for example, “friends graphs”
- The model has parameters
  - for determining **the probability of generating any particular instance of the artifact**
  - the “likelihood” of the parameter values
- The value of the parameters that gives the largest value of the likelihood is the correct model for the observed artifact

# Simple MLE Example (1)

- Model: Each edge is present with probability  $p$  with the presence or absence of each edge chosen independently
- Parameter:  $p$
- Example: a graph with 15 nodes and 23 edges
  - How many pairs of nodes do we have? 105
  - If each edge is connected by a probability  $p$ , *the probability of having a graph on the right is*
    - ❖  $p^{23}(1-p)^{82}$



# Simple MLE Example (2)

- Given this graph, what is the best value for  $p$  ?
- Take the first derivative of the probability function and set it to 0 for finding the maximum/minimum of  $p$ , as follows

$$23p^{22}(1-p)^{82} - 82p^{23}(1-p)^{81} = 0$$
$$p^{22}(1-p)^{81}(23(1-p) - 82p) = 0$$

- Maximum or minimum of  $p^{23}(1-p)^{82}$  happens when
  - Either  $p=0$  or  $1$  or  $23(1-p)-82p = 0$
  - The likelihood of generating the graph in the previous slide is maximized when  $23(1-p)-82p = 0$
- That is:*  $p = 23/105$

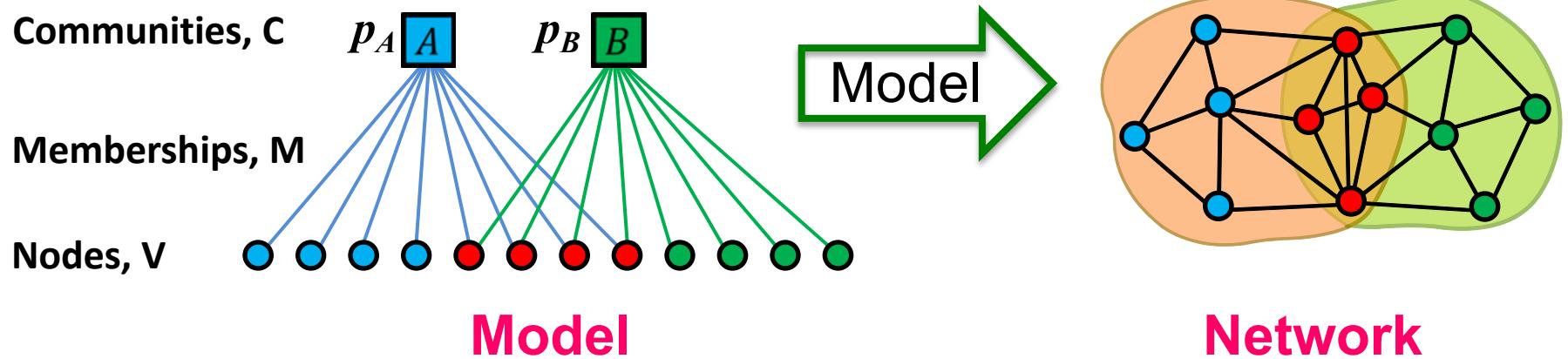
$$p^{23}(1-p)^{82} = (23/105)^{23} * (1-23/105)^{82} = 1.0656135e-24$$

# MLE Approach

- Given a graph  $G$ 
  - Write up a probability function  $f(p)$  of  $G$
  - To find the best  $p$  for  $f(p)$ :
    - Set the derivative of  $f(p)$  to be 0:  $f'(p) = 0$
    - Solve the equation  $f'(p)=0$  to get  $p$
- *The result  $p$  will be the best model for  $G$*

# Affiliation-Graph Model

# Community-Affiliation Graph



- **Generative model  $B(V, C, M, \{p_c\})$  for graphs:**
  - Nodes  $V$ , Communities  $C$ , Memberships  $M$
  - Each community  $c$  has a single probability  $p_c$
  - Later we fit the model to networks to detect communities

# Affiliation-Graph Model

- The Community-Affiliation Graph:

1. Given:

- A number of communities and
- A number of individuals (nodes)

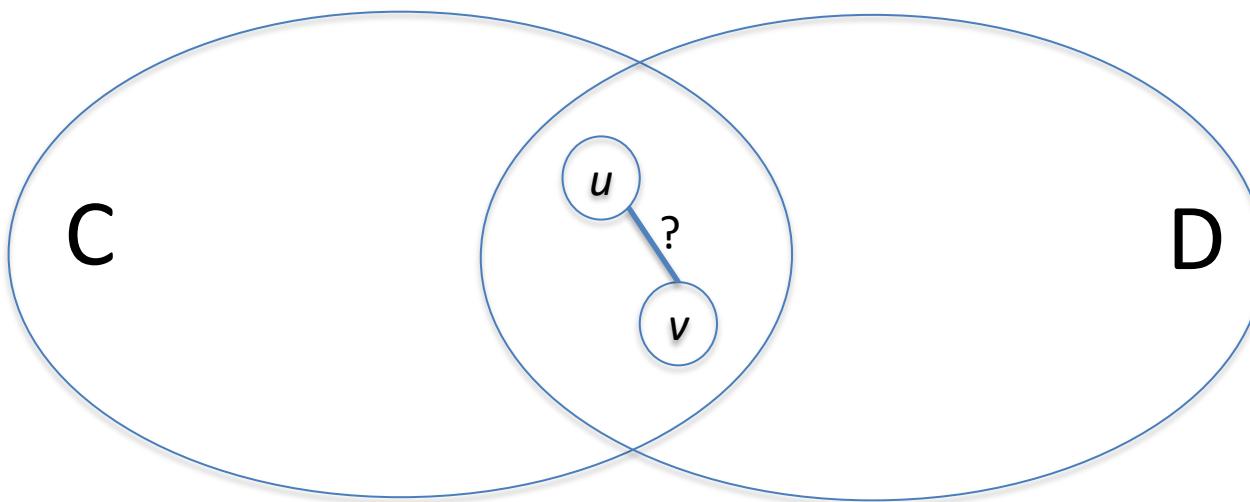
2. Two sets of parameters:

- The membership parameters
  - Each community can have any set of individuals as members (binary memberships)
- A probability  $P_c$  for each community  $C$ 
  - The probability that two members of a community  $C$  are connected because they are both members of  $C$

**Node:** If a pair of nodes is in two or more communities, then there is an edge between them if any of the communities of which both are members justifies that edge according to rule 3 (see the next slide for an example)

# Should $u$ and $v$ have an edge?

- Two communities C and D
  - Node  $u$  and  $v$  are in both communities



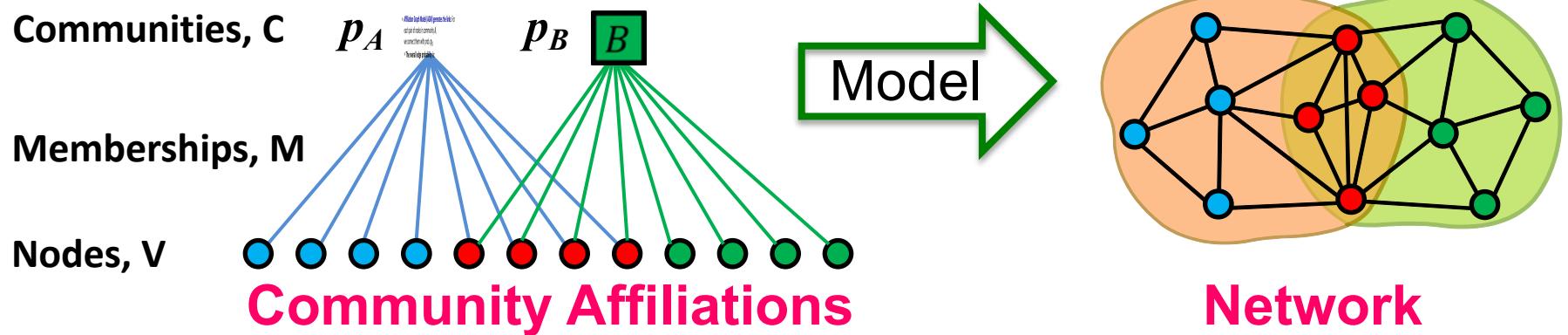
- What is the probability of **no edge between  $u, v$ :**  $(1 - P_C)(1 - P_D)$
- What is the probability of  **$(u, v)$  exists:**  $1 - (1 - P_C)(1 - P_D)$

# AGM: Generative Process (1)

- Nodes:  $u, v$ ; Edge:  $(u, v)$ 
  - $u$  and  $v$  are in both communities  $C$  and  $D$  but not in other communities
  - Given the probability that two nodes in  $C$  (or  $D$ ) are connected is  $P_C$  (or  $P_D$ )
  - The probability of **no edge between  $u, v$**  is  $(1 - P_C)(1 - P_D)$
  - The probability of  **$(u, v)$  exists** is  $1 - (1 - P_C)(1 - P_D)$
- General case: If  $u$  and  $v$  are members of a set  $M$  of communities, the probability of an edge between  $u$  and  $v$  is:

$$p_{uv} = 1 - \prod_{C \text{ in } M} (1 - p_C)$$

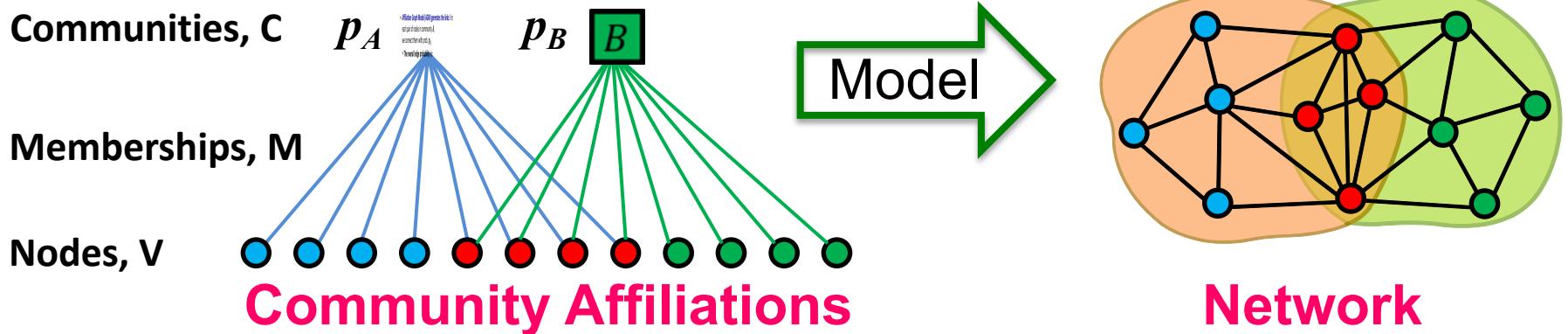
# AGM: Generative Process (2)



- Affiliation Graph Model (AGM) Generates the links:
  - For each pair of nodes  $u$  and  $v$  in community  $C$ , we connect them with probability  $p_c$ , then the overall edge probability is:
$$P(u, v) = 1 - \prod_{c \in M_u \cap M_v} (1 - p_c)$$
    - where  $M_u$  is the set of community  $u$  belongs to
  - If  $u$  and  $v$  share no community, then we let  $P(u, v) = \varepsilon$ , where  $\varepsilon$  is very tiny number

Think of this as an “OR” function: If at least 1 community says “YES” we create an edge

# AGM: Generative Process (2)



- **Affiliation Graph Model (AGM) generates the links:**  
For each pair of nodes in community  $A$ , we connect them with prob.  $p_A$

- **The overall edge probability between any  $u$  and  $v$ :**

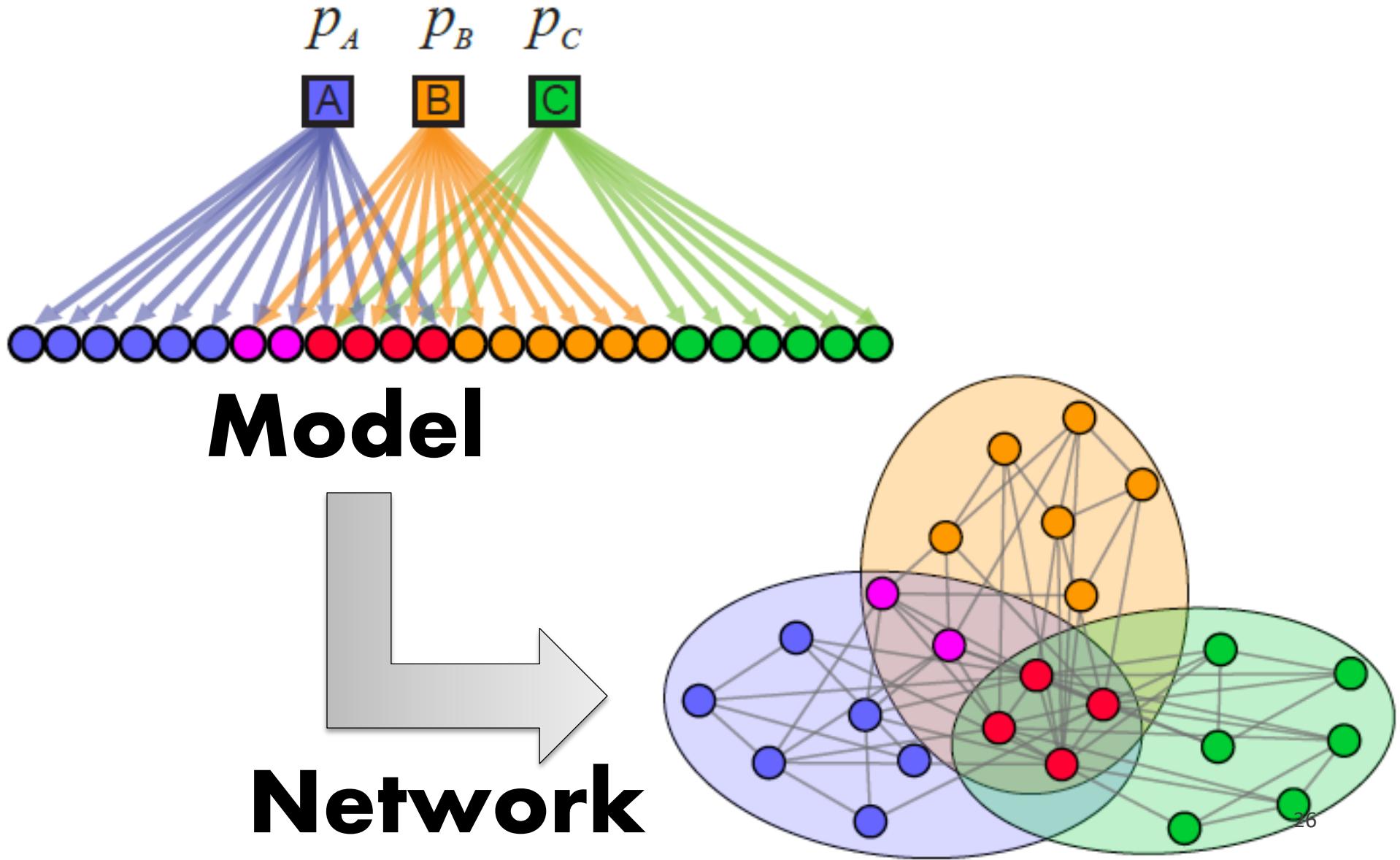
$$P(u, v) = 1 - \prod_{c \in M_u \cap M_v} (1 - p_c) \quad \text{epsilon } \varepsilon \text{ is a very tiny number}$$

If  $u, v$  share no communities:  $P(u, v) = \varepsilon$

$M_u$  ... set of communities  
node  $u$  belongs to

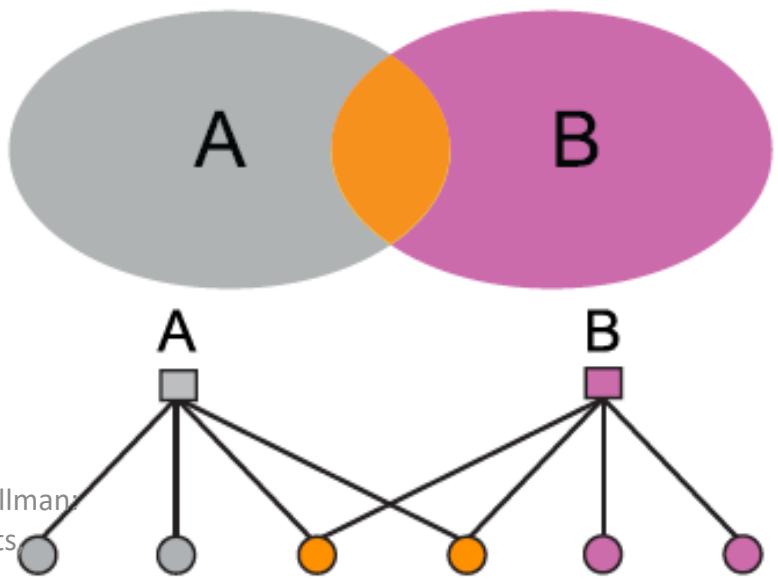
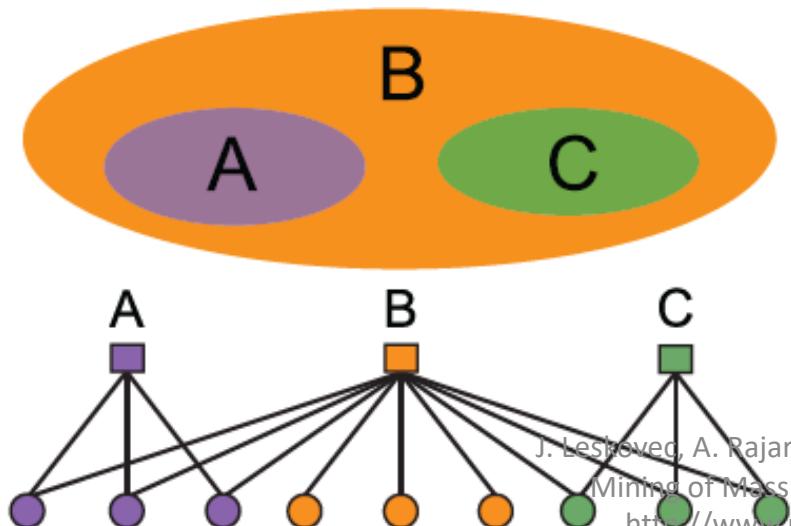
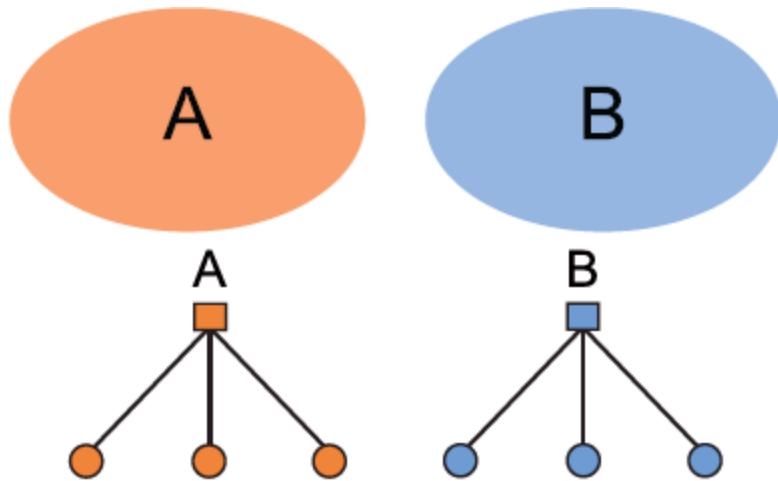
Think of this as an “OR” function: If at least 1 community says “YES” we create an<sup>25</sup>edge

# Recap: AGM networks



# AGM: Flexibility

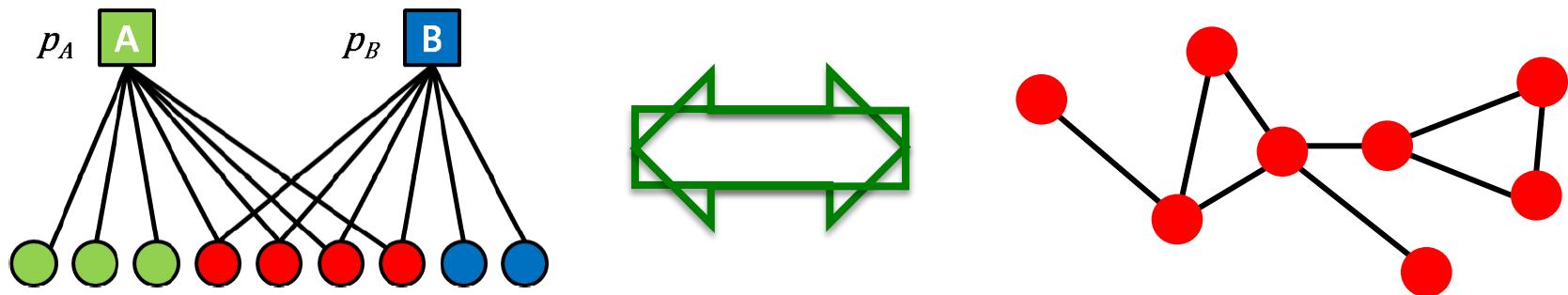
- AGM can express a variety of community structures:  
Non-overlapping,  
Overlapping, Nested



# **HOW DO WE DETECT COMMUNITIES WITH AGM?**

# Detecting Communities

- Detecting communities with AGM:



Given a Graph  $G(V, E)$ , find the Model

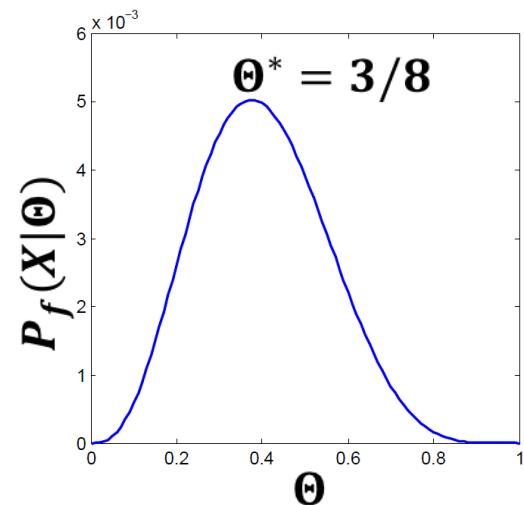
- 1) Affiliation graph  $M$
- 2) Number of communities  $C$
- 3) Parameters  $p_c$

# Maximum Likelihood Estimation

- **Maximum Likelihood Principle (MLE):**
  - **Given:** Data  $X$
  - **Assumption:** Data is generated by some model  $f(\Theta)$ 
    - $f$  ... model
    - $\Theta$  ... model parameters
  - Want to estimate  $P_f(X|\Theta)$ :
    - The probability that our model  $f$  (with parameters  $\Theta$ ) generated the data
  - **Now let's find the most likely model (including the parameters) that could have generated the data:**
$$\arg \max_{\Theta} P_f(X|\Theta)$$

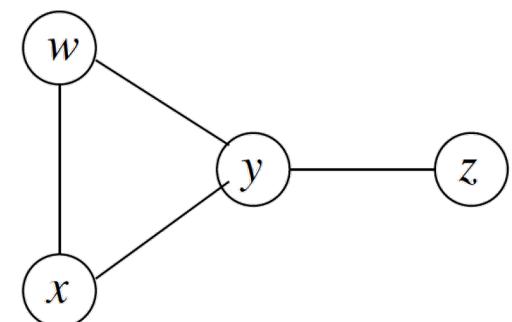
# Example: MLE

- Imagine we are given a set of coin flips
- Task: Figure out the bias of a coin!
  - Data: Sequence of coin flips:  $X = [1, 0, 0, 0, 1, 0, 0, 1]$
  - Model:  $f(\Theta)$  = return 1 with prob.  $\Theta$ , else return 0
  - What is  $P_f(X|\Theta)$ ? Assuming coin flips are independent
    - So,  $P_f(X|\Theta) = P_f(1|\Theta) * P_f(0|\Theta) * P_f(0|\Theta) ... * P_f(1|\Theta)$ 
      - What is  $P_f(1|\Theta)$ ? Simple,  $P_f(1|\Theta) = \Theta$
    - Then,  $P_f(X|\Theta) = \Theta^3(1 - \Theta)^5$
    - For example:
      - $P_f(X|\Theta = 0.5) = 0.003906$
      - $P_f(X|\Theta = \frac{3}{8}) = 0.005029$
  - What did we learn? Our data was most likely generated by coin with bias  $\Theta = 3/8$



# AGM Example (1)

- Using AGM for modeling the graph below
- Assumptions (preset some of the parameters):
  - There are two communities  $C$  and  $D$  with  $P_C$  and  $P_D$
  - We have determined (or are using as a temporary hypothesis) the community memberships:
    - $C = \{w, x, y\}$  and  $D = \{w, y, z\}$



# AGM Example (2)

- Given  $C = \{w, x, y\}$  and  $D = \{w, y, z\}$
- For  $w$  and  $x$ ,  $M_{wx} = \{C\}$ 
  - $P_{wx} = 1 - (1 - P_C) = P_C$
- And
  - $P_{xy} = P_C$  and  $P_{yz} = P_{wz} = P_D$
  - $P_{wy} = 1 - (1 - P_C)(1 - P_D)$
  - $P_{xz} = \text{a tiny value}$

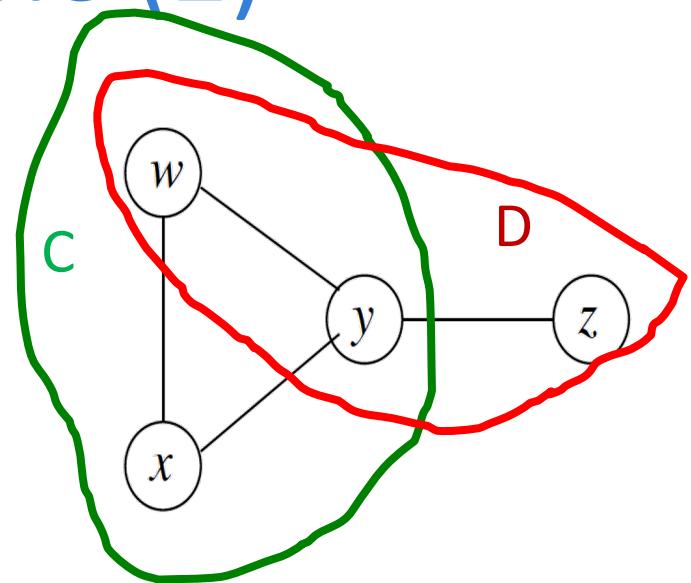


Figure 10.20: A social graph

- The likelihood of this graph given the membership assumption
- $$P_{wx}P_{wy}P_{xy}P_{yz}(1 - P_{wz})(1 - P_{xz})$$
$$= (P_C)^2 P_D (P_C + P_D - P_C P_D) (1 - P_C) (1 - \epsilon)$$

# AGM Example (3)

- Find the values of  $P_C$  and  $P_D$  that maximize:  
$$(p_C)^2 p_D (p_C + p_D - p_C p_D) (1 - p_D) (1 - \epsilon)$$
  - The last factor can be dropped
- You can solve it by setting the derivative = 0
- Or you can do the following reasoning:
  - Let  $P_C$  should be as large as possible
  - Given  $P_C = 1$ , the expression becomes:
  - $P_D (1 - P_D) \rightarrow P_D = 0.5$  when the expression has its maximum
- The maximum likelihood for the graph in Fig. 10.20 occurs when members of C are certain to have an edge between them and
- There is a 50% chance that joint membership in D will cause an edge between the members

# Maximum-Likelihood Estimation for Graphs

- How do we do MLE for graphs?
  - Model generates a **probabilistic adjacency matrix**
  - We then flip all the entries of the probabilistic matrix to obtain the **binary adjacency matrix  $A$**

For every pair  
of nodes  $u, v$   
AGM gives the  
prob.  $p_{uv}$  of  
them being  
linked

0	0.10	0.10	0.04
0.10	0	0.02	0.06
0.10	0.02	0	0.06
0.04	0.06	0.06	0

Flip  
biased  
coins

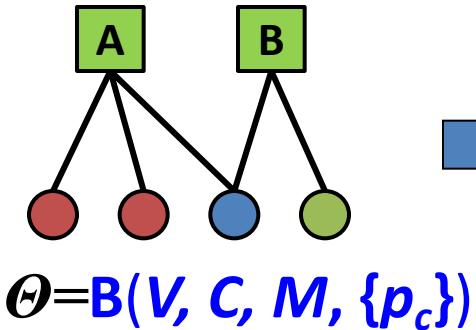
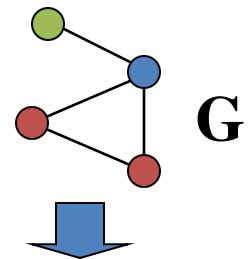
$A$	0	1	0	0
0	1	0	1	1
1	0	1	0	1
0	1	0	1	0
0	1	1	0	0

- The likelihood of AGM generating graph G:

$$P(G | \Theta) = \prod_{(u,v) \in E} P(u,v) \prod_{(u,v) \notin E} (1 - P(u,v))$$

# Graphs: Likelihood $P(G|\Theta)$

- Given graph  $G(V,E)$  and  $\Theta$ , we calculate likelihood that  $\Theta$  generated  $G$ :  $P(G|\Theta)$



0	0.9	0.9	0
0.9	0	0.9	0
0.9	0.9	0	0.9
0	0	0.9	0

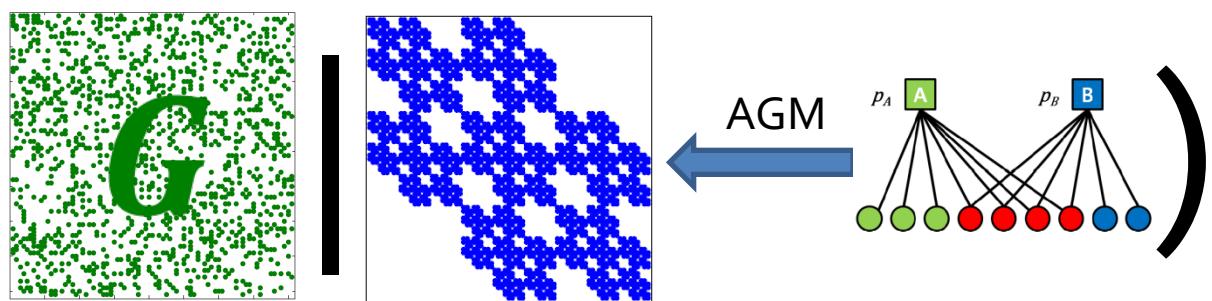
0	1	1	0
1	0	1	0
1	1	0	1
0	0	1	0

$P(G|\Theta)$

$$P(G | \Theta) = \prod_{(u,v) \in E} P(u, v) \prod_{(u,v) \notin E} (1 - P(u, v))$$

# MLE for Graphs

- Our goal: Find  $\Theta = B(V, C, M, \{p_C\})$  such that:

$$\arg \max_{\Theta} P(G | \text{AGM} \rightarrow \text{Graphs})$$


- How do we find  $B(V, C, M, \{p_C\})$  that maximizes the likelihood?

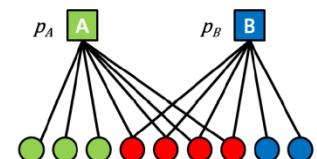
# MLE for AGM

- Our goal is to find  $B(V, C, M, \{p_C\})$  such that:

$$\arg \max_{B(V, C, M, \{p_C\})} \prod_{u, v \in E} P(u, v) \prod_{uv \notin E} (1 - P(u, v))$$

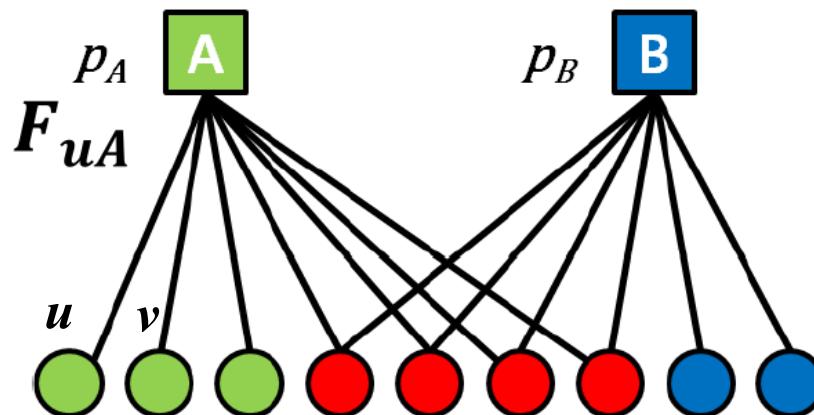
- Problem: Finding  $B$  means finding the bipartite affiliation network.

- There is no nice way to do this.
- Fitting  $B(V, C, M, \{p_C\})$  is too hard, let's change the model (so it is easier to fit)!



# From AGM to BigCLAM (1)

- **Relaxation:** Memberships have strengths (avoid discrete membership changes)

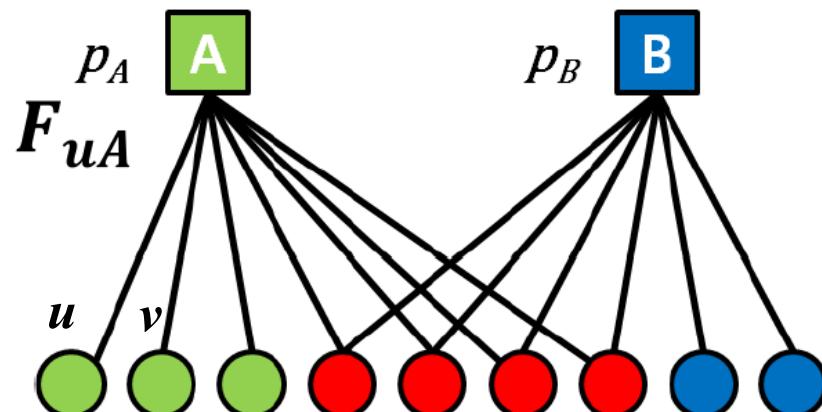


- $F_{uA}$ : The membership strength of node  $u$  to community  $A$  ( $> 0$ ) ( $F_{uA} = 0$ : no membership)
- Each community  $A$  links nodes independently.  
Define:

$$P_A(u, v) = 1 - \exp(-F_{uA} \cdot F_{vA})$$

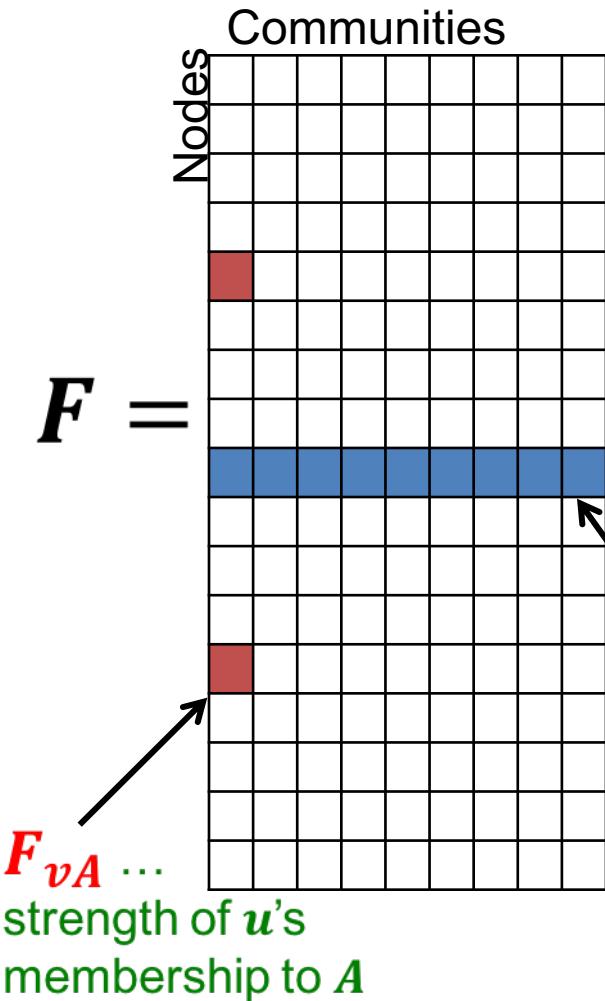
# From AGM to BigCLAM (2)

- $P_A(u, v) = 1 - \exp(-F_{uA} \cdot F_{vA})$
- If  $u$  and  $v$  are not in the same community
  - $P_A(u, v) = 1 - \exp(0) = 0$
- If  $F_{uA}$  and  $F_{vA}$  are both large numbers:
  - $P_A(u, v) = 1 - e^{-ALargeNumber} = 1 - ASmallNumber$
- If  $F_{uA}$  and  $F_{vA}$  are both small numbers:
  - $P_A(u, v) = 1 - e^{-ASmallNumber} = 1 - ALargeNumber$



# Factor Matrix $F$

## ■ Community membership strength matrix $F$



- $P_A(u, v) = 1 - \exp(-F_{uA} \cdot F_{vA})$ 
  - Probability of connection is proportional to the product of strengths
    - Notice: If one node doesn't belong to the community ( $F_{uC} = 0$ ) then  $P(u, v) = 0$
- Prob. that **at least one** common community  $C$  links the nodes:
  - $P(u, v) = 1 - \prod_C (1 - P_C(u, v))$

# From AGM to BigCLAM

- Community  $A$  links nodes  $u, v$  independently:

$$P_A(u, v) = 1 - \exp(-F_{uA} \cdot F_{vA})$$

- Then prob. at least one common  $C$  links them:

$$\begin{aligned} P(u, v) &= 1 - \prod_C (1 - P_C(u, v)) \\ &= 1 - \exp(-\sum_C F_{uC} \cdot F_{vC}) \\ &= 1 - \exp(-F_u \cdot F_v^T) \end{aligned}$$

- Example  $F$  matrix:

$$F_u : \begin{array}{|c|c|c|c|} \hline 0 & 1.2 & 0 & 0.2 \\ \hline \end{array}$$

$$F_v : \begin{array}{|c|c|c|c|} \hline 0.5 & 0 & 0 & 0.8 \\ \hline \end{array}$$

$$F_w : \begin{array}{|c|c|c|c|} \hline 0 & 1.8 & 1 & 0 \\ \hline \end{array}$$

Node community  
membership strengths

Then:  $F_u \cdot F_v^T = 0.16$

And:  $P(u, v) = 1 - \exp(-0.16) = 0.14$

But:  $P(u, w) = 0.88$  //  $1 - \exp(-(1.2 * 1.8))$

$P(v, w) = 0$

# BigCLAM: How to find $F$

- **Task:** Given a network  $G(V, E)$ , estimate  $F$ 
  - Find  $F$  that maximizes the likelihood:

$$\arg \max_F \prod_{(u,v) \in E} P(u, v) \prod_{(u,v) \notin E} (1 - P(u, v))$$

- where:  $P(u, v) = 1 - \exp(-F_u \cdot F_v^T)$
- Many times we take the logarithm of the likelihood, and call it log-likelihood:  $l(F) = \log P(G|F)$

- **Goal:** Find  $F$  that maximizes  $l(F)$ :

$$l(F) = \sum_{(u,v) \in E} \log(1 - \exp(-F_u F_v^T)) - \sum_{(u,v) \notin E} F_u F_v^T$$

# BigCLAM: V1.0

$$l(F_u) = \sum_{v \in \mathcal{N}(u)} \log(1 - \exp(-F_u F_v^T)) - \sum_{v \notin \mathcal{N}(u)} F_u F_v^T$$

- **Compute gradient of a single row  $F_u$  of  $F$ :**

$$\nabla l(F_u) = \sum_{v \in \mathcal{N}(u)} F_v \frac{\exp(-F_u F_v^T)}{1 - \exp(-F_u F_v^T)} - \sum_{v \notin \mathcal{N}(u)} F_v$$

- **Coordinate gradient ascent:**

$\mathcal{N}(u)$ .. Set out outgoing neighbors

- **Iterate over the rows of  $F$ :**
  - Compute gradient  $\nabla l(F_u)$  of row  $u$  (while keeping others fixed)
  - Update the row  $F_u$ :  $F_u \leftarrow F_u + \eta \nabla l(F_u)$
  - Project  $F_u$  back to a non-negative vector: If  $F_{uC} < 0$ :  $F_{uC} = 0$
- **This is slow! Computing  $\nabla l(F_u)$  takes linear time!**

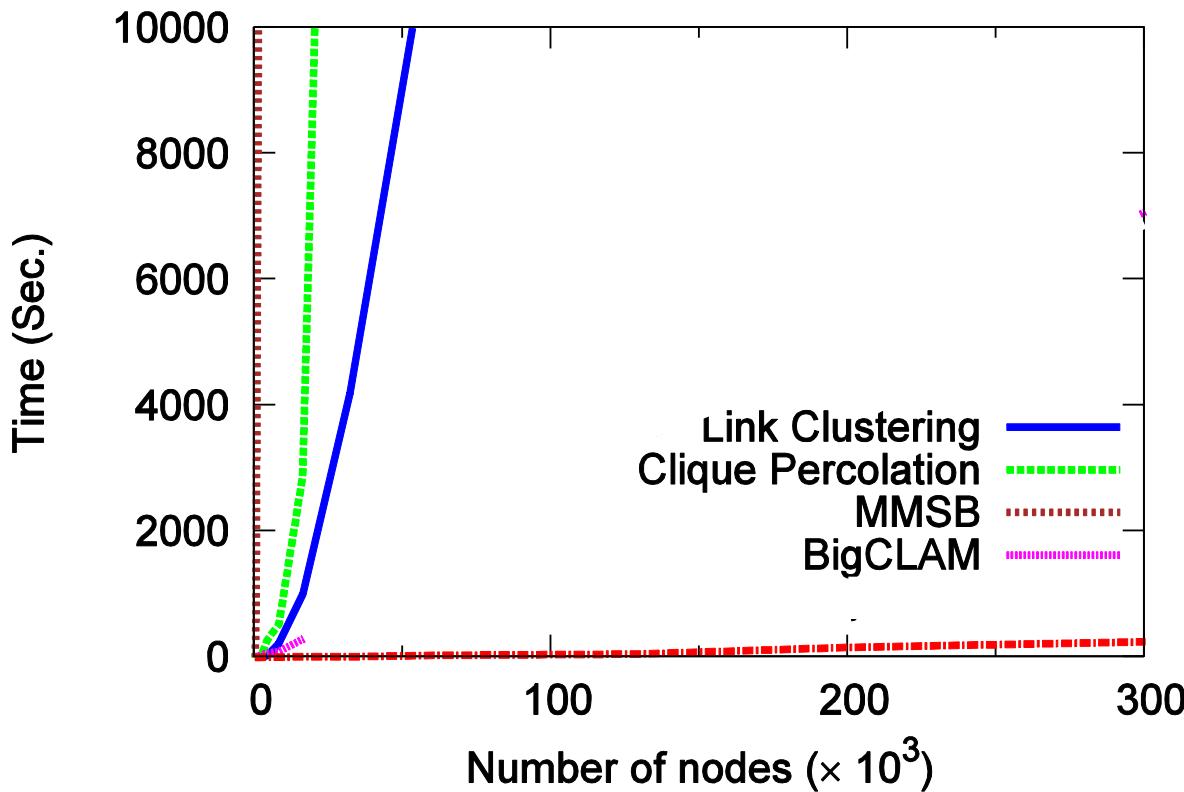
# BigCLAM: V2.0

- **However, we notice:** (By examining the F matrix)

$$\sum_{v \notin \mathcal{N}(u)} F_v = \left( \sum_v F_v - F_u - \sum_{v \in \mathcal{N}(u)} F_v \right)$$

- We cache  $\sum_v F_v$
- So, computing  $\sum_{v \notin \mathcal{N}(u)} F_v$  now takes **linear time** in the degree  $|\mathcal{N}(u)|$  of  $u$ 
  - In networks degree of a node is much smaller to the total number of nodes in the network, so this is a significant speedup!

# BigClam: Scalability



- **BigCLAM takes 5 minutes for 300k node nets**
  - Other methods take 10 days
- **Can process networks with 100M edges!**

# Summary of BigCLAM

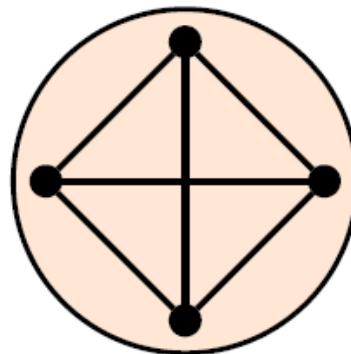
- Given  $G(V, E)$ , compute  $B(C, M, \{p_c\})$  such that  $P(G | B)$  is maximum
  - But it is too hard to find  $C$ ,  $M$ , and  $\{p_c\}$  this way
- V1.0. Change to a model easier to find
  - Replace  $C$  and  $M$  by  $F$
  - Define  $P(u, v) = 1 - \exp(-F_{uA} F_{vA})$
  - Compute  $F$  such that  $P(G | F)$  is maximum
- V2.0. Speed up the computation of  $\sum_{v \in \mathcal{N}(u)} F_v$ 
  - By computing  $\sum_{v \in \mathcal{N}(u)} F_v$

# **EXTENSION: DIRECTED MEMBERSHIPS**

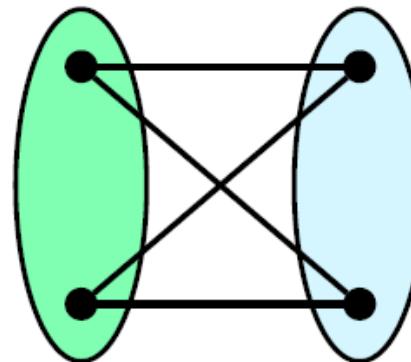
# Extension: Beyond Clusters

Undirected

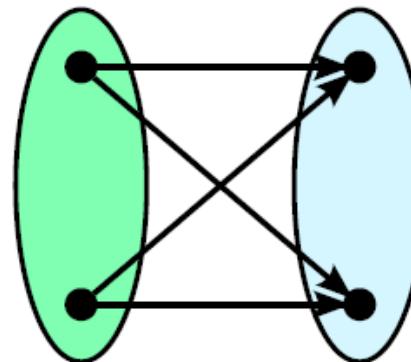
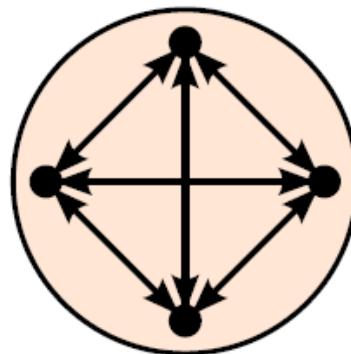
Cohesive



2-mode



Directed

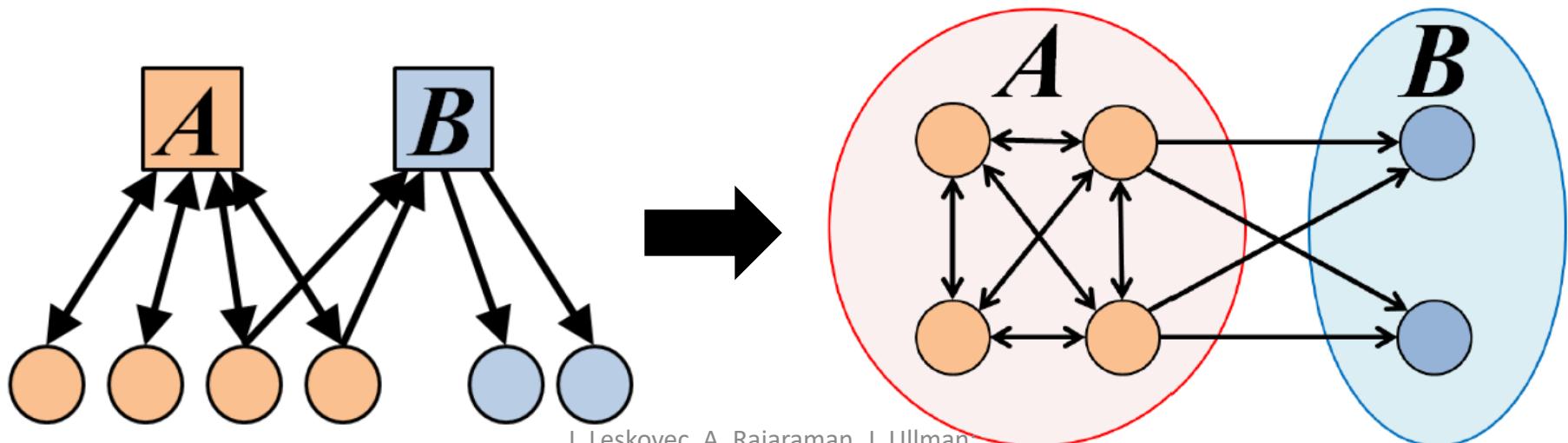


# Extension: Directed AGM

- **Extension:**

**Make community membership edges directed!**

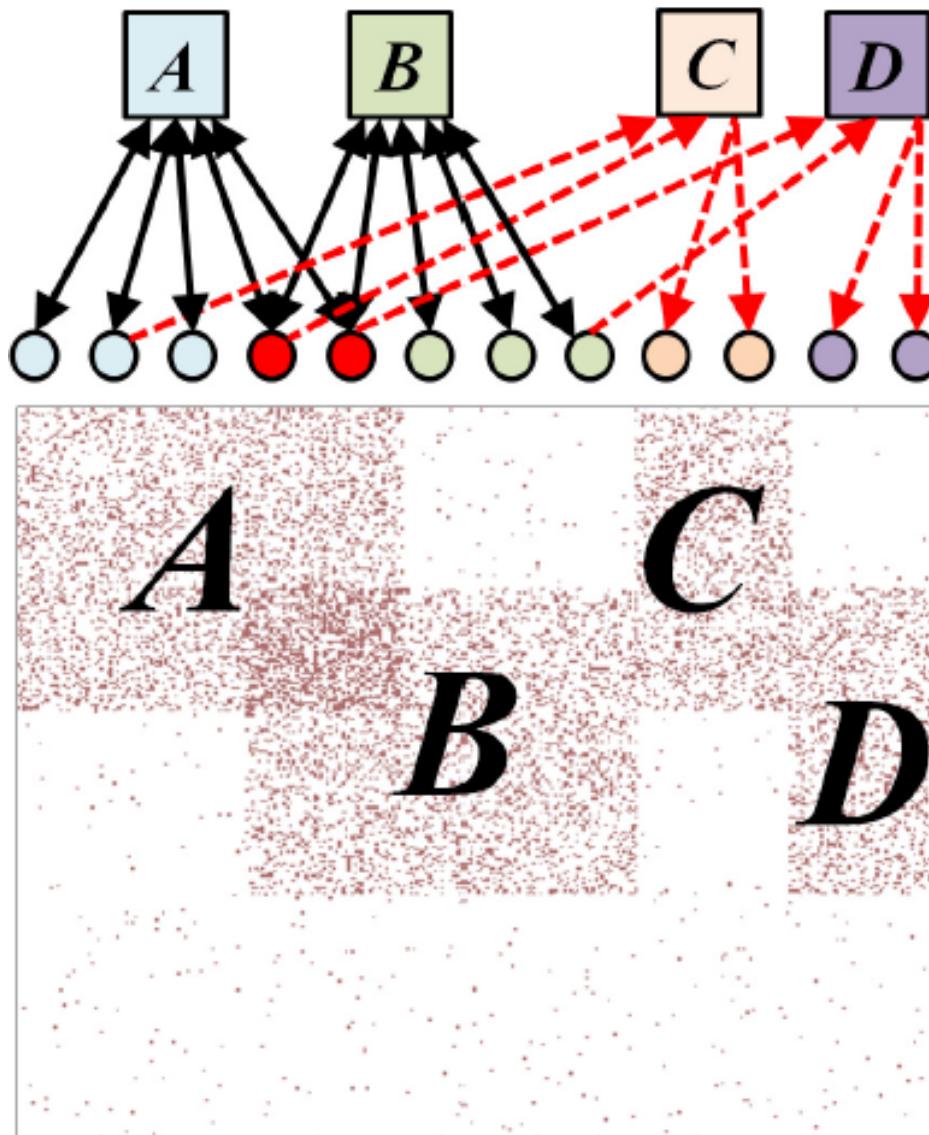
- Outgoing membership: Nodes “**sends**” edges
- Incoming membership: Node “**receives**” edges



J. Leskovec, A. Rajaraman, J. Ullman:

Mining of Massive Datasets,  
<http://www.mmds.org>

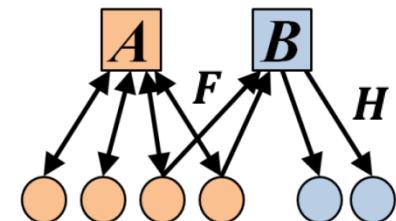
# Example: Model and Network



# Directed AGM

- Everything is almost the same except now we have 2 matrices:  $F$  and  $H$

- $F$ ... out-going community memberships
- $H$ ... in-coming community memberships

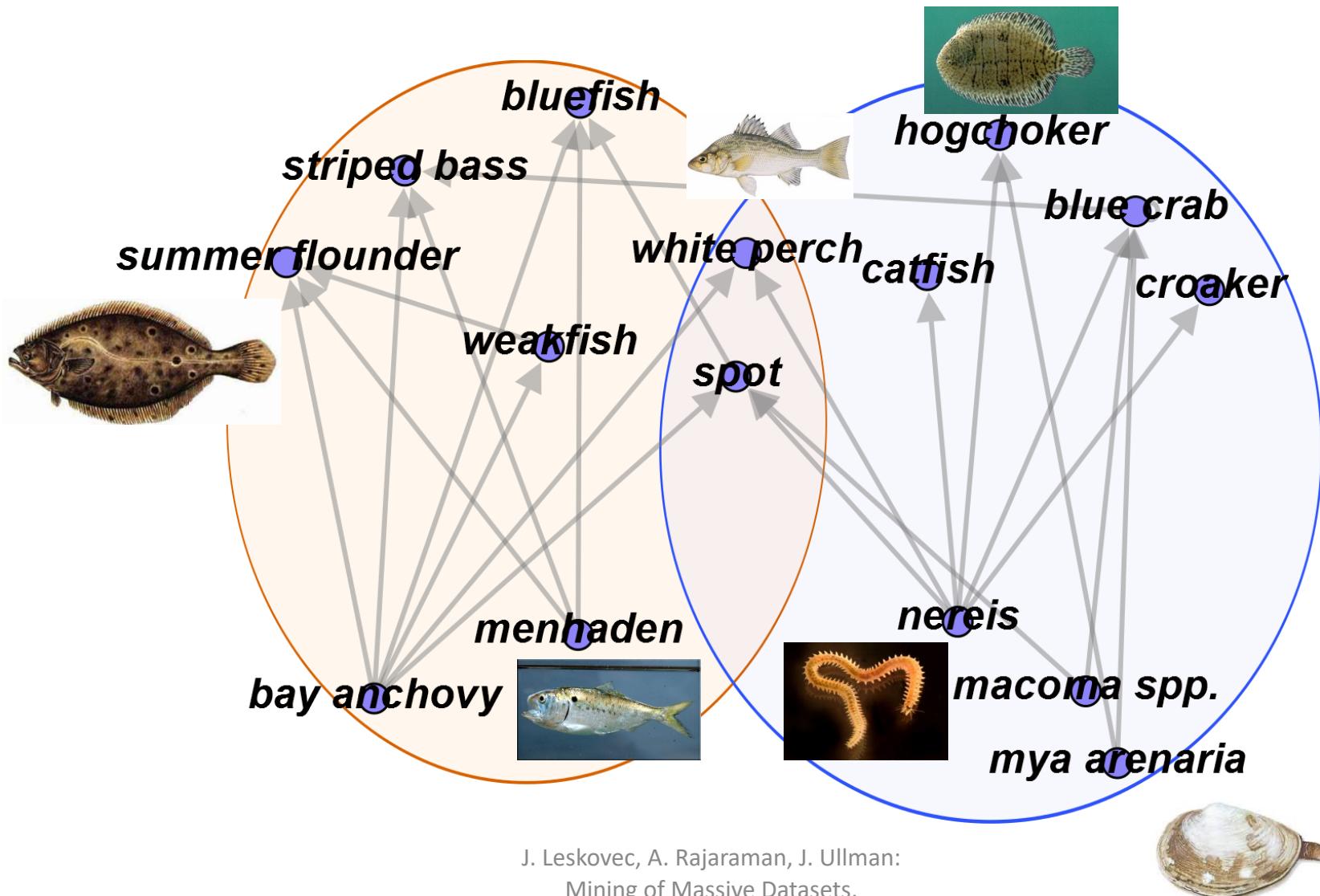


- Edge prob.:  $P(u, v) = \overbrace{1 - \exp(-F_u H_v^T)}$
- Network log-likelihood:

$$l(F, H) = \sum_{(u, v) \in E} \log(1 - \exp(-F_u H_v^T)) - \sum_{(u, v) \notin E} F_u H_v^T$$

which we optimize the same way as before

# Predator-prey Communities



# More Details at...

- Overlapping Community Detection at Scale: A Nonnegative Matrix Factorization Approach by J. Yang, J. Leskovec. *ACM International Conference on Web Search and Data Mining (WSDM)*, 2013.
- Detecting Cohesive and 2-mode Communities in Directed and Undirected Networks by J. Yang, J. McAuley, J. Leskovec. *ACM International Conference on Web Search and Data Mining (WSDM)*, 2014.
- Community Detection in Networks with Node Attributes by J. Yang, J. McAuley, J. Leskovec. *IEEE International Conference On Data Mining (ICDM)*, 2013.