

---

# **Part III: Graph-based spam/fraud detection algorithms and apps**

---

# Part III: Outline

## ➔ Algorithms: **relational learning**

- ❑ Collective classification
- ❑ Relational inference

## ■ Applications: **fraud and spam** detection

- ❑ Online auction fraud
- ❑ Accounting fraud
- ❑ Fake review spam
- ❑ Web spam

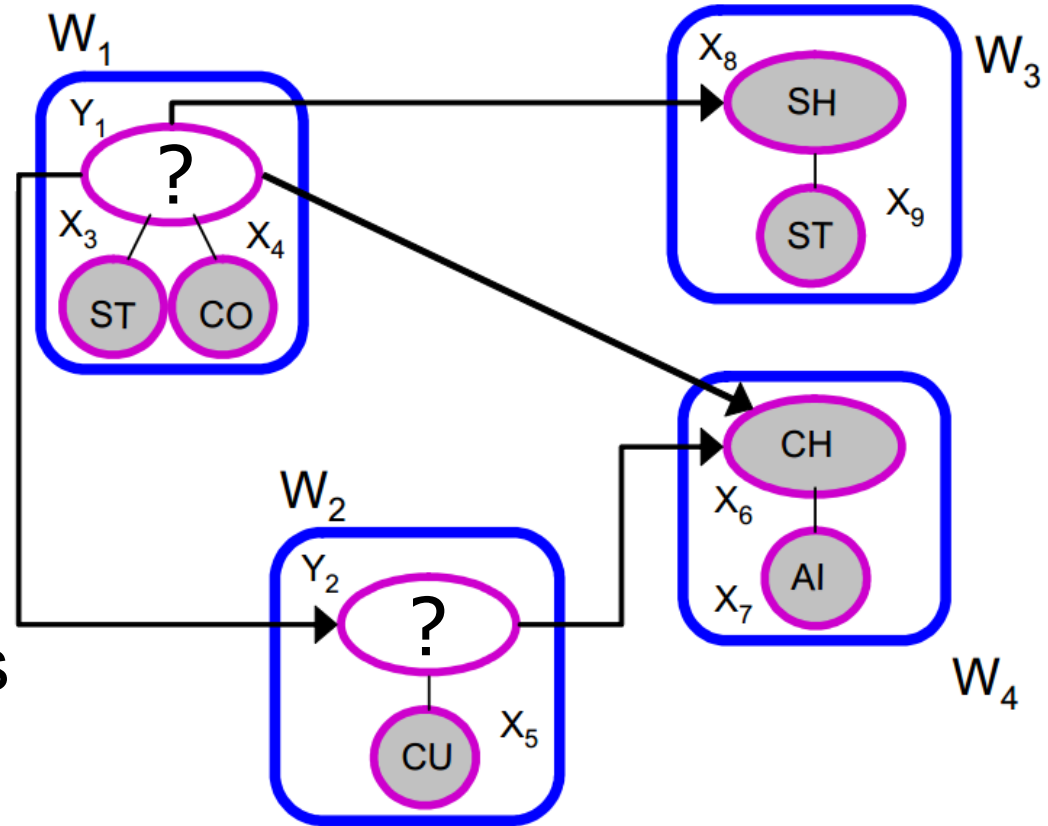
# Collective classification (CC)

异常检测作为分类问题

- Anomaly detection as a **classification problem**
  - ❑ spam/non-spam email, **malicious/benign web page**, **fraud/legitimate transaction**, etc.
- Often **connected** objects → **guilt-by-association**
- Label of object  $o$  in network may depend on:
  - ❑ **Attributes** (features) of  $o$  o的标签取决于：o的属性，o邻居的标签与属性
  - ❑ **Labels of objects** in  $o$ 's neighborhood
  - ❑ **Attributes of objects** in  $o$ 's neighborhood
- **CC: simultaneous** classification of interlinked objects using above correlations

# Problem sketch

- Graph (V, E)
- Nodes as variables
  - X: observed
  - Y: TBD
- Edges
  - **observed** relations
- **Goal:** label Y nodes

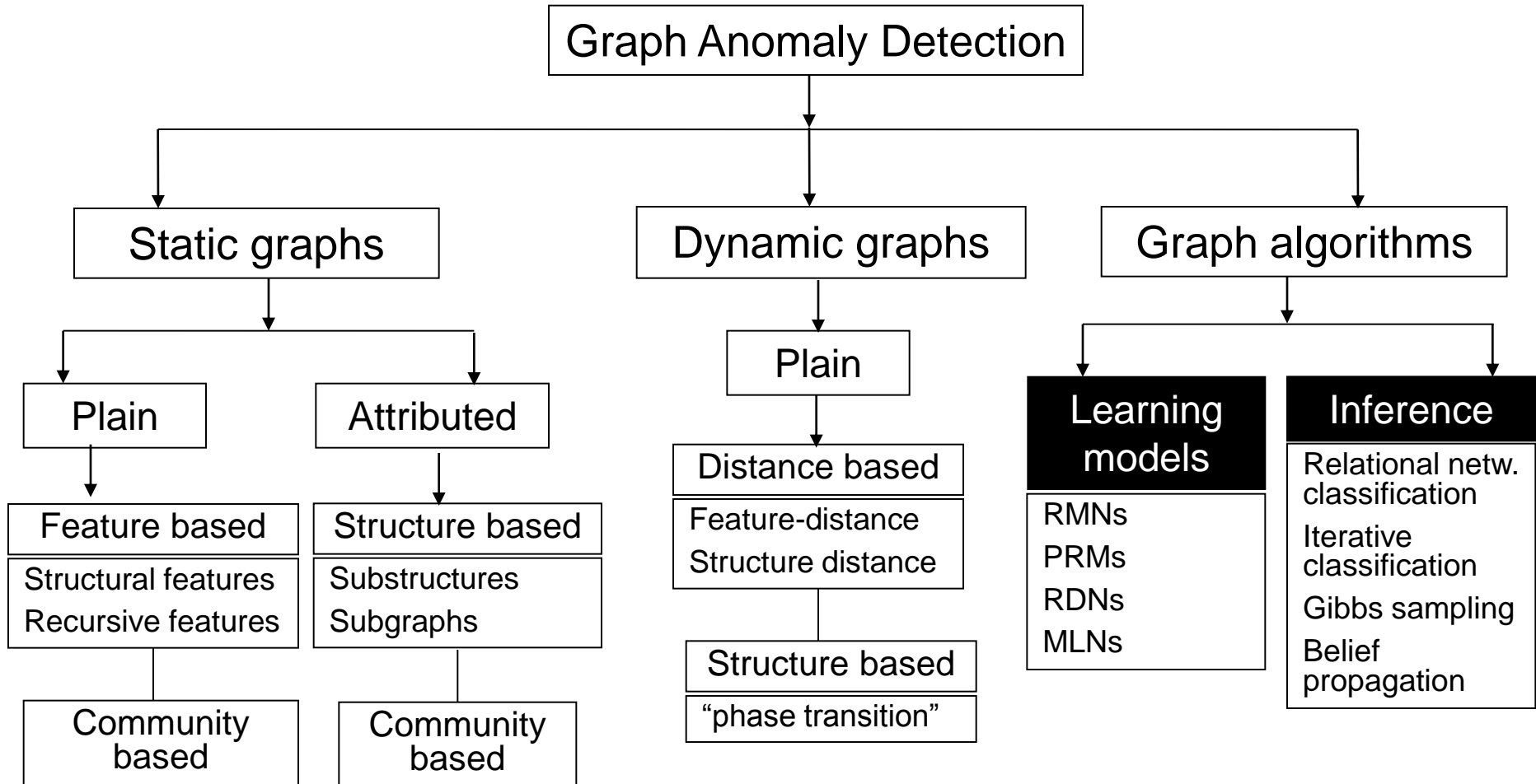


nodes; web pages, edges; hyperlinks, labels; SH or CH: student/course page; features nodes are keywords; ST: student, CO: course, CU: curriculum, AI: artificial intelligence

# Collective classification applications

- Document classification Chakrabarti+'98, Taskar+'02
- Part of speech tagging Lafferty+'01
- Link prediction Taskar+'03
- Optical character recognition Taskar+'03
- Image/3Ddata segmentation Anguelov+'05, Chechotka+'10
- Entity resolution in sensor networks Chen+'03
- Spam and fraud detection Pandit+'07, Kang+'11

# Taxonomy



# Collective classification models

- Relational Markov Networks (RMNs)  
Taskar, Abbeel, Koller'03
- Relational Dependency Networks (RDNs)  
Neville&Jensen'07
- Probabilistic Relational Models (PRMs)  
Friedman, Getoor, Koller, Pfeffer+'99
- Markov Logic Networks (MLNs)  
Richardson&Domingos'o6

# Collective classification inference

- Exact inference is **NP hard** for arbitrary networks
- Approximate inference techniques [in this tutorial]

## ➔ Relational classifier

Macskassy&Provost'03,07

- Iterative classification alg. (ICA)

Neville&Jensen'00, Lu&Getoor'03, McDowell+'07

- Gibbs sampling IC

Gilks et al. '96

- Loopy belief propagation

Yedidia et al. '00

**Note:** All the above are **iterative**



# (prob.) Relational network classifier

- “A simple relational classifier”
- Class probability of  $Y_i$  is a **weighted average** of class probabilities of its **neighbors**
- **Repeat** for each  $Y_i$  and label  $c$

$$P(Y_i = c) = \frac{1}{Z} \sum_{(Y_i, Y_j) \in E} w(Y_i, Y_j) P(Y_j = c)$$

- **pRN challenges:**
  - Convergence not guaranteed
  - Some initial class probabilities should be biased or no propagation
  - Cannot use attribute info

# Iterative classification

- **Main idea:** classify node  $Y_i$  based on its attributes as well as neighbor set  $N_i$ 's labels
- Convert each node  $Y_i$  to a **flat vector**  $a_i$

Various #neighbors → **aggregation**

- count

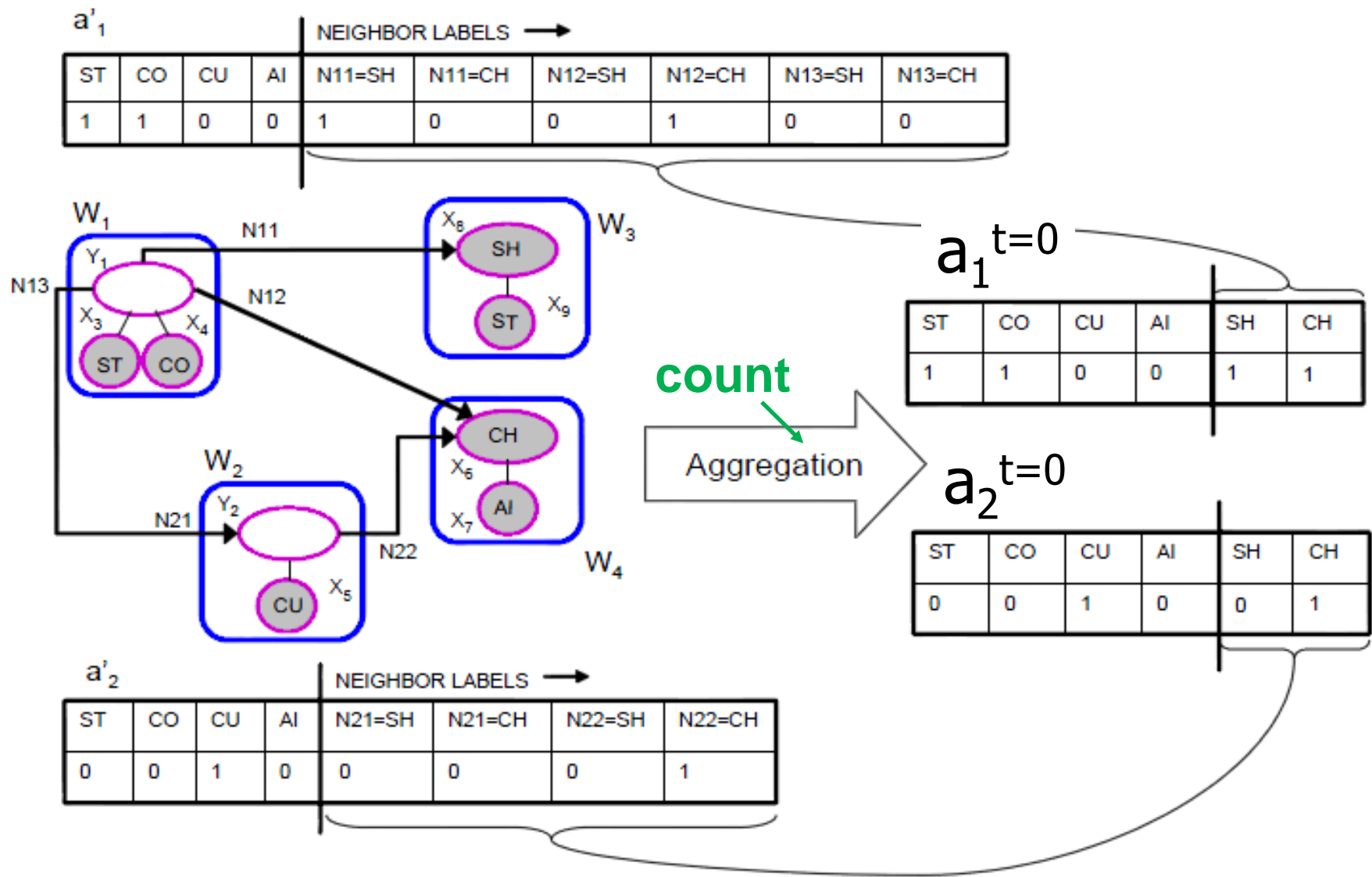
- mode

- proportion

- mean

- exists

# Iterative classification

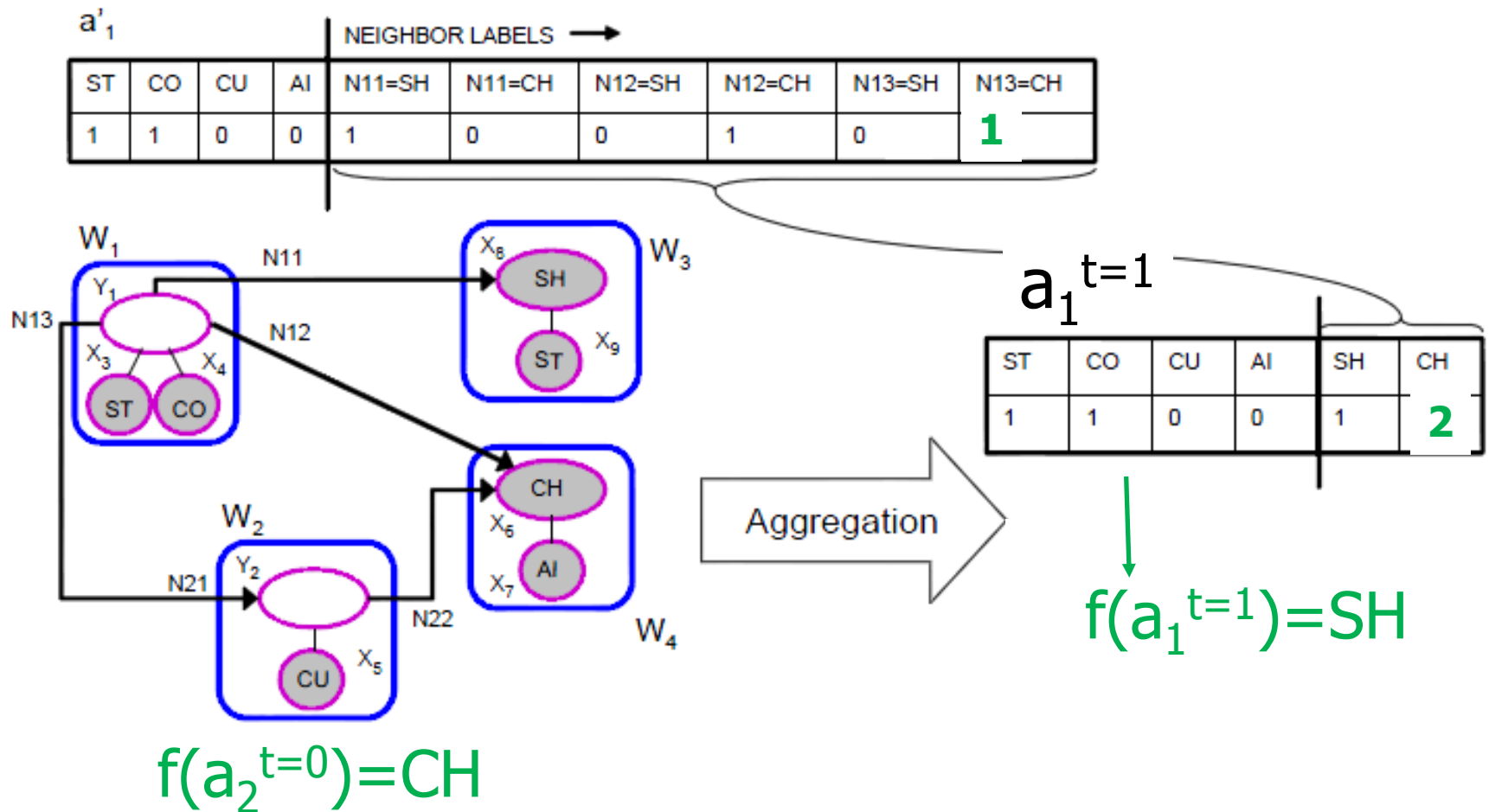


# Iterative classification

- Main idea: classify  $Y_i$  based on  $N_i$
- Convert each node  $Y_i$  to a **flat vector**  $a_i$ 
  - Various #neighbors  $\rightarrow$  **aggregation**
- **Use local classifier**  $f(a_i)$  (e.g., SVM, kNN, ...) to compute best value for  $y_i$
- **Repeat** for each node  $Y_i$ 
  - **Reconstruct** feature vector  $a_i$
  - **Update** label to  $f(a_i)$  (hard assignment)  
 $\text{argmax}_{l \in \mathcal{L}} f$
- Until class labels stabilize or max # iterations

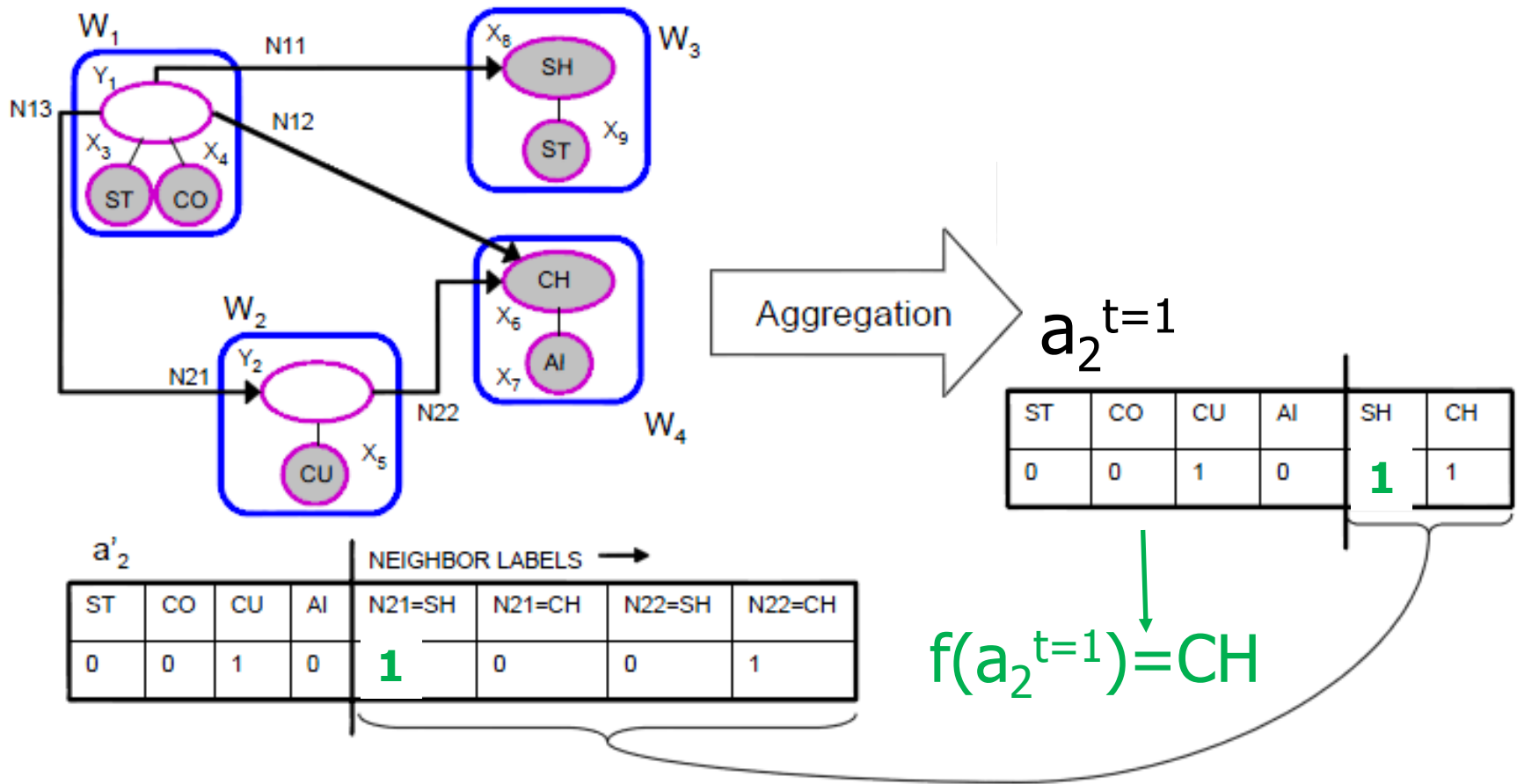
**Note:** convergence not guaranteed

# Iterative classification



# Iterative classification

$$f(a_1^{t=1}) = \text{SH}$$



# Gibbs sampling

Bootstrap

Burn-in

Sample

- Main idea:
  - ❑ Convert each node  $Y_i$  to a flat vector  $a_i$
  - ❑ Use local classifier  $f(a_i)$  to compute best value for  $y_i$
  - ❑ **Repeat B times** for each node  $Y_i$ 
    - Reconstruct feature vector  $a_i$
    - Update label to  $f(a_i)$  (hard assignment)
  - ❑ **Repeat S times** for each node  $Y_i$ 
    - Sample  $y_i$  from  $f(a_i)$
    - Increase count  $c(i, y_i)$  by 1
  - ❑ Assign to each  $Y_i$  label  $y_i \leftarrow \operatorname{argmax}_{l \in \mathcal{L}} c[i, l]$

# IC and GS challenges

- **Feature construction** for local classifier  $f$ 
  - $f$  often needs **fixed-length** vector
  - choice of **aggregation** (avg, mode, count, ...)
  - choice of relations (in-, out-links, both)
  - choice of neighbor attributes (all?, top-k confident?)
- **Local classifier  $f$** 
  - requires **training**
  - choice of classifier (LR, NB, kNN, SVM, ...)
- **Node ordering** for updates (random, diversity based)
- **Convergence**
- **Run time** (many iterations for GS)



# Collective classification inference

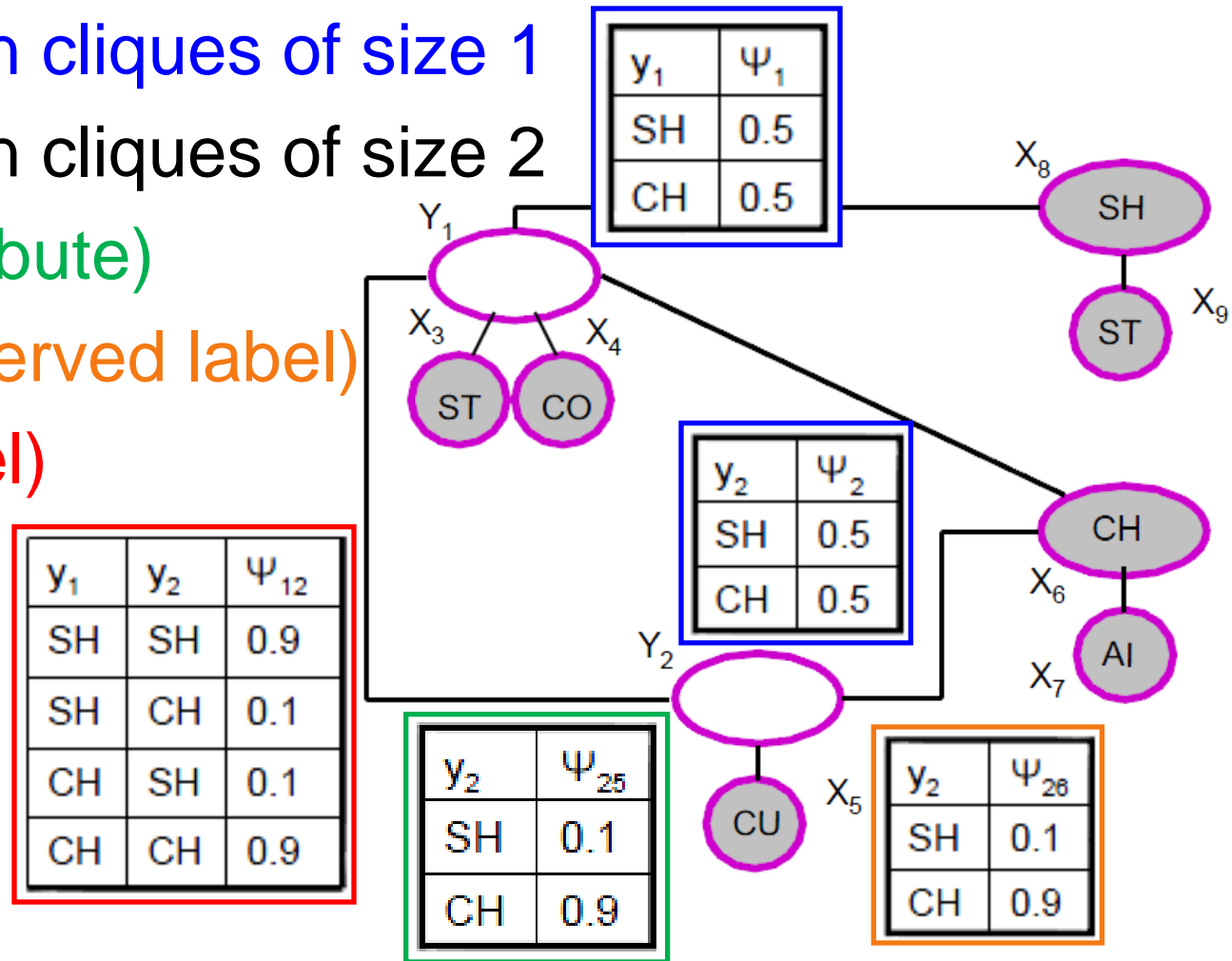
- Exact inference is **NP hard** for arbitrary networks
- Approximate inference techniques [in this tutorial]
  - **Relational classifier**  
Macskassy&Provost'03,07
  - **Iterative classification alg. (ICA)**  
Neville&Jensen'00, Lu&Getoor'03, McDowell+'07
  - **Gibbs sampling IC**  
Gilks et al. '96
  - ➔ **Loopy belief propagation**  
Yedidia et al. '00

**Note:** All the above are **iterative**

# Relational Markov Nets

- Undirected dependencies
- Potentials on cliques of size 1
- Potentials on cliques of size 2
  - (label-attribute)
  - (label-observed label)
  - (label-label)

For pairwise RMNs max clique size is 2



# pairwise Markov Random Field

- For an assignment  $\mathbf{y}$  to all unobserved  $\mathbf{Y}$ , pMRF is associated with probability distr:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{Y_i \in \mathcal{Y}} \phi_i(y_i) \prod_{(Y_i, Y_j) \in E} \psi_{ij}(y_i, y_j)$$

Node labels as  
random variables

compatibility  
potentials  
(label-label)

“known”  
potential

$$\phi_i(y_i) = \psi_i(y_i) \prod_{(Y_i, X_j) \in E} \psi_{ij}(y_i)$$

prior belief  
(1-clique potentials)

observed potentials  
(label-observed label)  
(label-attribute)

$y_1$	$\Psi_{13}$
SH	0.6
CH	0.4

$y_1$	$\Psi_{16}$
SH	0.1
CH	0.9

$y_1$	$\Psi_{18}$
SH	0.8
CH	0.2

$y_1$	$\Psi_1$
SH	0.5
CH	0.5

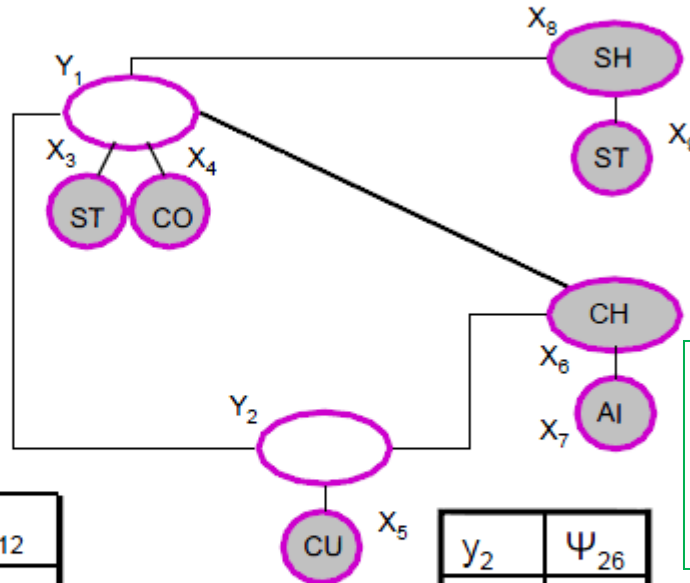
$y_1$	$\Psi_{14}$
SH	0.4
CH	0.6

$y_1$	$y_2$	$\Psi_{12}$
SH	SH	0.9
SH	CH	0.1
CH	SH	0.1
CH	CH	0.9

$y_2$	$\Psi_2$
SH	0.5
CH	0.5

$y_2$	$\Psi_{26}$
SH	0.1
CH	0.9

$y_2$	$\Psi_{25}$
SH	0.1
CH	0.9



$$\Phi_1 = \Psi_1 * \Psi_{13} * \Psi_{14} * \Psi_{16} * \Psi_{18} =$$

$y_1$	$\Phi_1$
SH	0.0096
CH	0.0216

$$\Phi_2 = \Psi_2 * \Psi_{25} * \Psi_{26} =$$

$y_2$	$\Phi_2$
SH	0.005
CH	0.405

# pMRF interpretation

- Defines a joint pdf of all **unknown** labels
- $P(y \mid x)$  is the probability of a given world  $y$
- Best label  $y_i$  for  $Y_i$  is the one with highest marginal probability
- Computing one marginal probability  $P(Y_i = y_i)$  requires summing over **exponential # terms**
- **#P problem**  $\rightarrow$  approximate inference  $\rightarrow$  loopy belief propagation

# Loopy belief propagation

- Invented in 1982 [Pearl] to calculate marginals in Bayes nets.
- Also used to **estimate marginals (=beliefs)**, or most likely states (e.g. MAP) in MRFs
- **Iterative** process in which neighbor variables “talk” to each other, passing **messages**

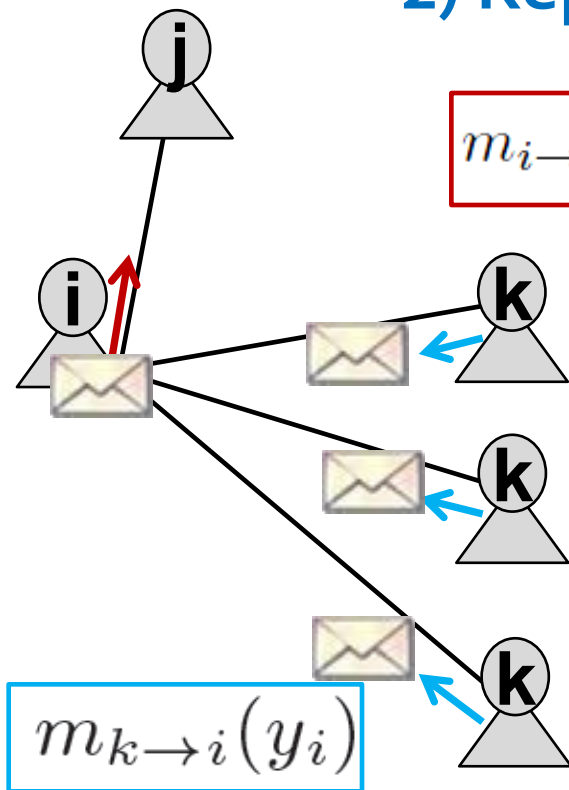
“I (variable  $x_1$ ) **believe** you (variable  $x_2$ ) belong in these states with various likelihoods...”



- When consensus reached, calculate **belief**

# Loopy belief propagation

- 1) Initialize all messages to 1
- 2) Repeat for each node:

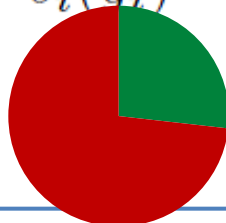


$$m_{i \rightarrow j}(y_j) = \alpha \sum_{y_i \in \mathcal{L}} \psi_{ij}(y_i, y_j) \phi_i(y_i)$$

$$\prod_{Y_k \in \mathcal{N}_i \cap \mathcal{Y} \setminus Y_j} m_{k \rightarrow i}(y_i), \quad \forall y_j \in \mathcal{L}$$

- 3) When messages “stabilize”:

$$b_i(y_i) = \alpha \phi_i(y_i) \prod_{Y_j \in \mathcal{N}_i \cap \mathcal{Y}} m_{j \rightarrow i}(y_i), \quad \forall y_i \in \mathcal{L}$$



$y_1$	$\psi_{13}$
SH	0.6
CH	0.4

$y_1$	$\psi_{16}$
SH	0.1
CH	0.9

$y_1$	$\psi_{18}$
SH	0.8
CH	0.2

$$\Phi_1 = \Psi_1 * \Psi_{13} * \Psi_{14} * \Psi_{16} * \Psi_{18} =$$

$y_1$	$\Phi_1$
SH	0.0096
CH	0.0216

$y_1$	$\psi_1$
SH	0.5
CH	0.5

$y_1$	$\psi_{14}$
SH	0.4
CH	0.6

$y_1$	$y_2$	$\psi_{12}$
SH	SH	0.9
SH	CH	0.1
CH	SH	0.1
CH	CH	0.9

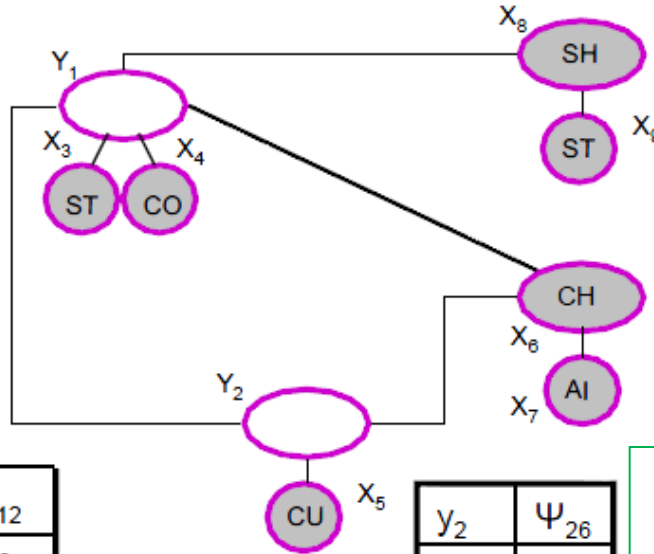
$y_2$	$\psi_2$
SH	0.5
CH	0.5

$y_2$	$\psi_{26}$
SH	0.1
CH	0.9

$y_2$	$\psi_{25}$
SH	0.1
CH	0.9

$$\Phi_2 = \Psi_2 * \Psi_{25} * \Psi_{26} =$$

$y_2$	$\Phi_2$
SH	0.005
CH	0.405



$$m_{1 \rightarrow 2}(y_2) = \sum_{y_1} \Phi_1(y_1) \psi_{12}(y_1, y_2)$$

$$m_{2 \rightarrow 1}(y_1) = \sum_{y_2} \Phi_2(y_2) \psi_{12}(y_1, y_2)$$

$$m_{1 \rightarrow 2}(\text{SH}) = (0.0096 * 0.9 + 0.0216 * 0.1) / (m_{1 \rightarrow 2}(\text{SH}) + m_{1 \rightarrow 2}(\text{CH})) \sim 0.35$$

$$m_{1 \rightarrow 2}(\text{CH}) = (0.0096 * 0.1 + 0.0216 * 0.9) / (m_{1 \rightarrow 2}(\text{SH}) + m_{1 \rightarrow 2}(\text{CH})) \sim 0.65$$



# Loopy belief propagation

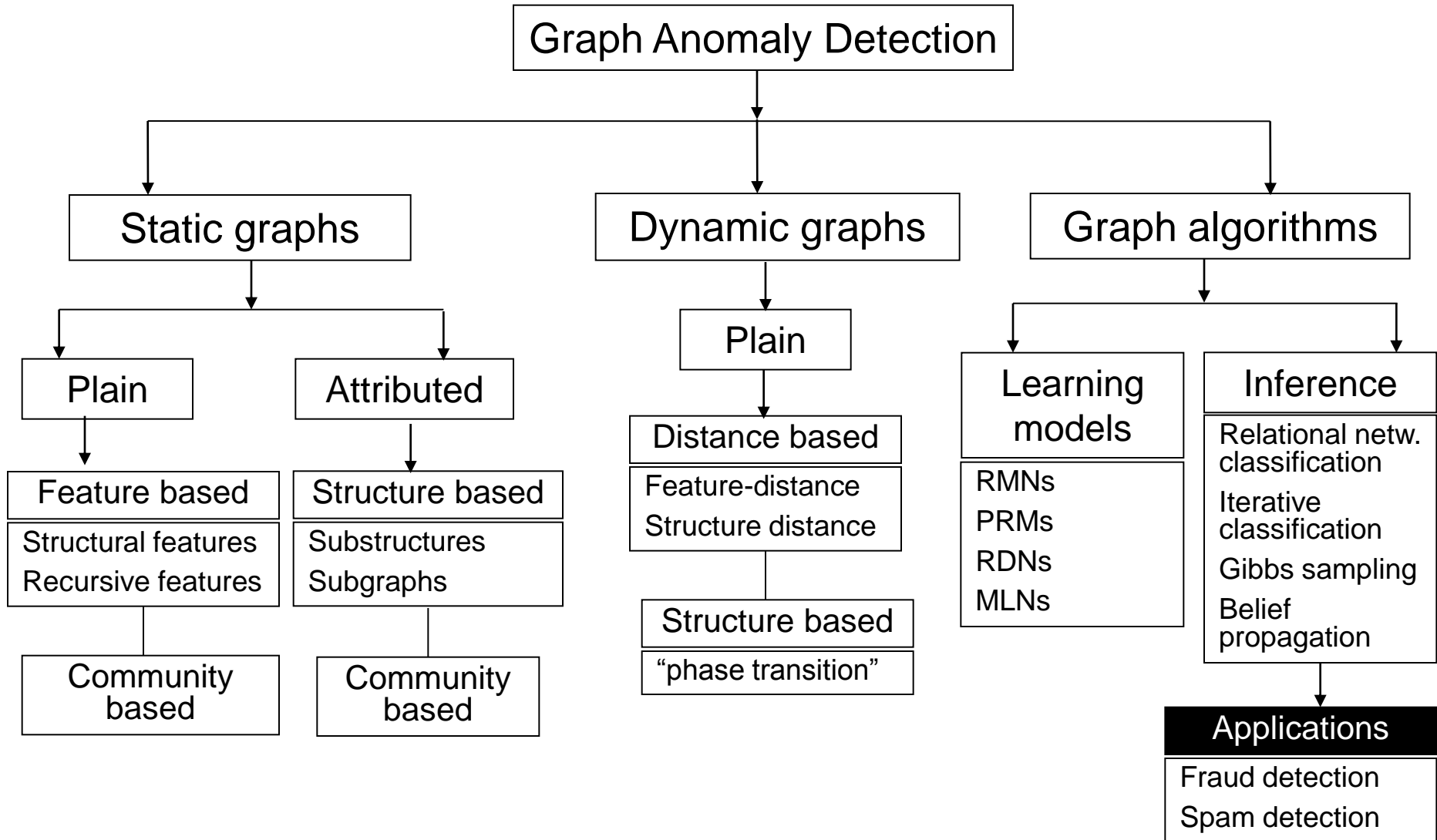
## Advantages:

- Easy to program & parallelize
- **General**: can apply to any graphical model w/ any form of potentials (higher order than pairwise)

## Challenges:

- **Convergence** is not guaranteed (when to stop)
  - esp. if many closed loops
- **Potential functions (parameters)**
  - require **training** to estimate
  - learning by gradient-based optimization: convergence issues during training

# Taxonomy



# Part III: Outline

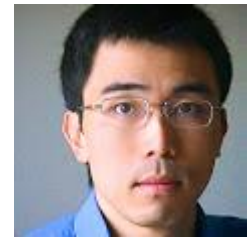
## ■ Algorithms: **relational learning**

- ❑ Collective classification
- ❑ Relational inference

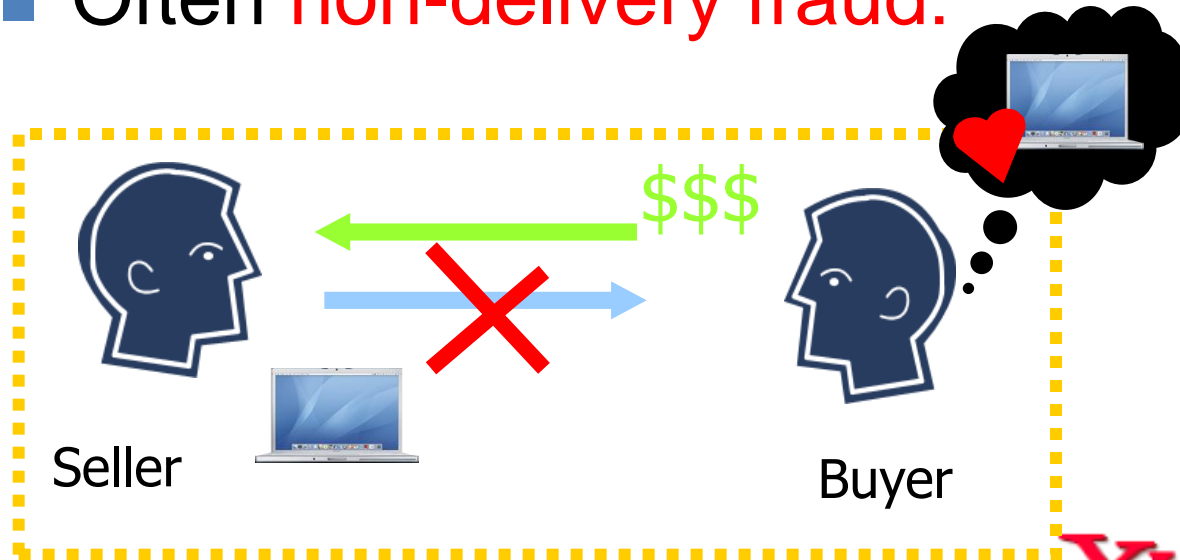
## ➔ Applications: **fraud and spam** detection

- ❑ (1) Online auction fraud
- ❑ (2) Accounting fraud
- ❑ (3) Fake review spam
- ❑ (4) Web spam

# (1) Online auction fraud



- Auction sites: attractive target for fraud
- 63% complaints to Federal Internet Crime Complaint Center in U.S. in 2006
- Average loss per incident: = \$385
- Often non-delivery fraud:



# Online auction fraud detection

- Insufficient solution:

- Look at individual features, e.g., IP addresses, login times, session history, etc.

*Easy to fake!*

- **Hard to fake:** graph structure

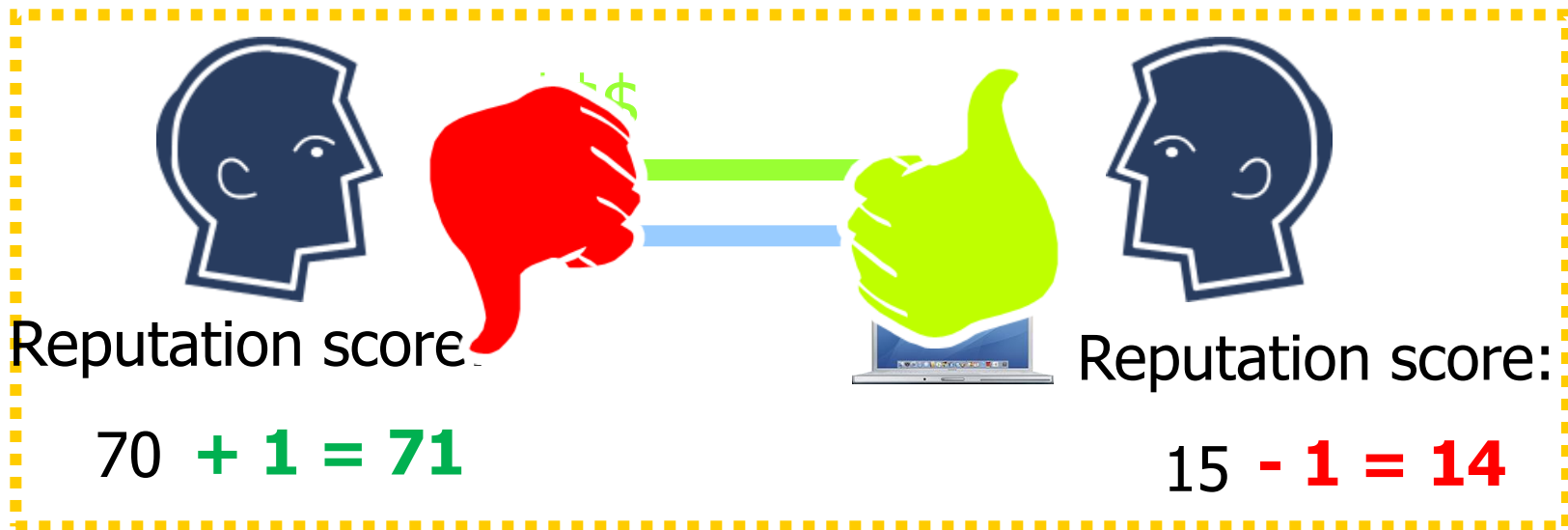
- Capture **relationships** between users

- Q: How do fraudsters **interact** with other users and among each other?

→ in addition to buy/sell relations, there is a feedback mechanism

# Feedback mechanism

- Each user has a **reputation score**
- Users rate each other via feedback

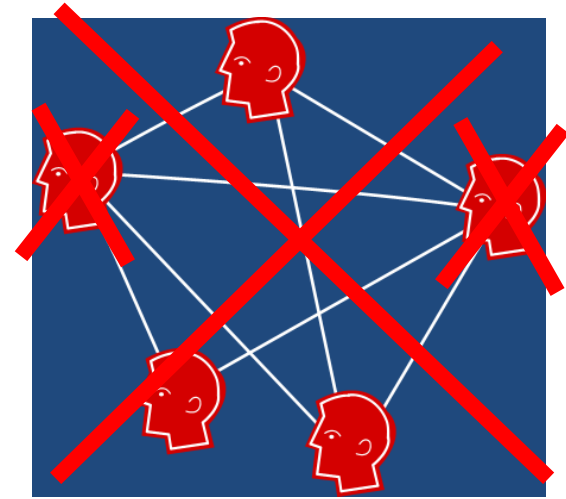


- Q: How do fraudsters **game** the feedback system?

# Auction "roles"

- Do they boost each other's reputation?

**No, if one caught, all caught!**



- They form near-bipartite cores (2 roles)



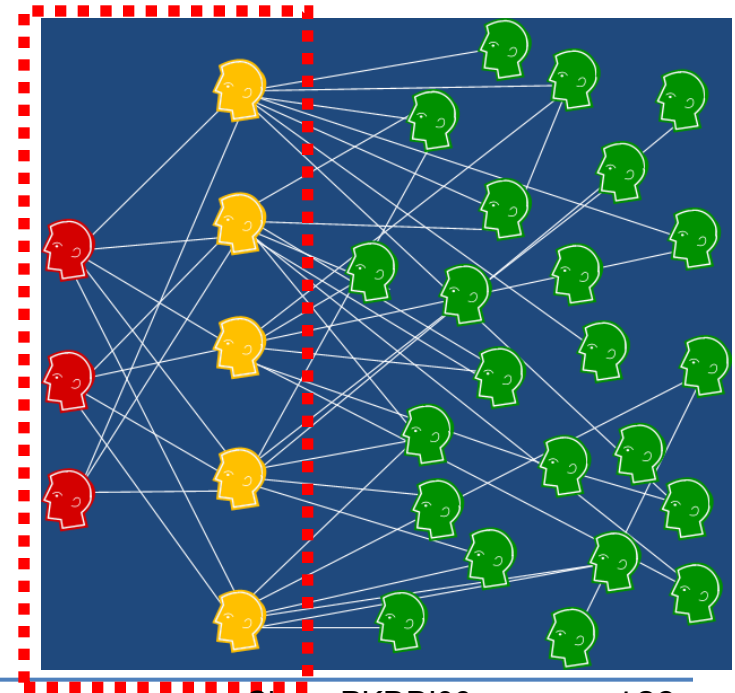
**accomplice**

- trades w/ honest, looks legit



**fraudster**

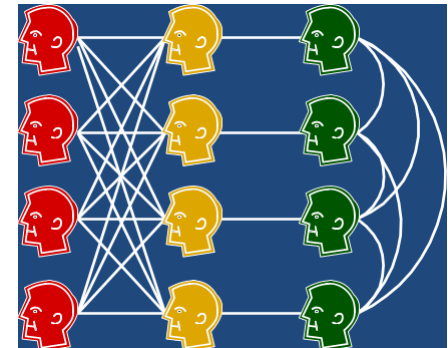
- trades w/ accomplice
- fraud w/ honest



# Detecting online fraud

- How to find near-bipartite cores? How to find roles (**honest**, **accomplice**, **fraudster**)?
  - Use Belief Propagation!
- How to set BP parameters (potentials)?
  - **prior beliefs**: prior knowledge, unbiased if none
  - **compatibility potentials**: by insight

	Fraud	Accomplice	Honest
Fraud	$\varepsilon_p$	$1 - 2\varepsilon_p$	$\varepsilon_p$
Accomplice	0.5	$2\varepsilon_p$	$0.5 - 2\varepsilon_p$
Honest	$\varepsilon_p$	$(1 - 2\varepsilon_p)/2$	$(1 - 2\varepsilon_p)/2$

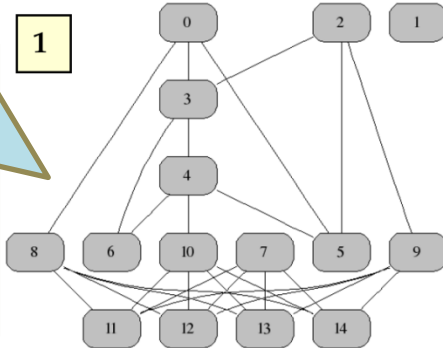




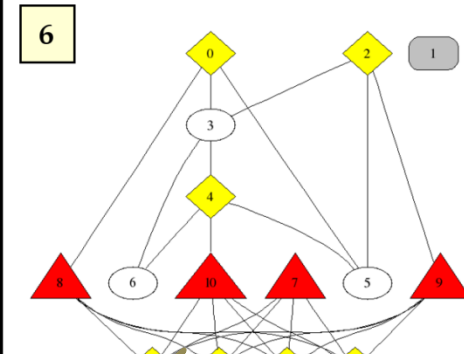
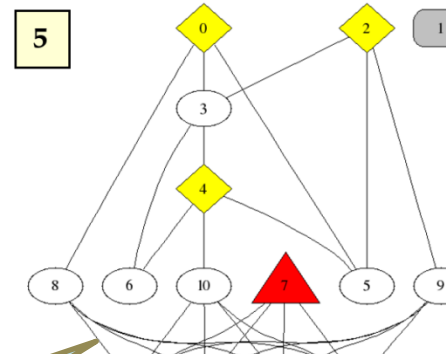
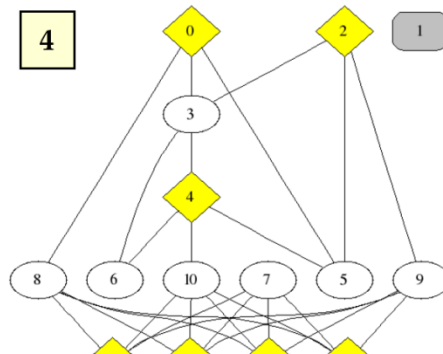
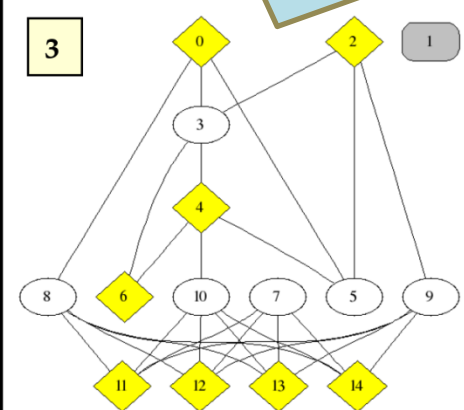
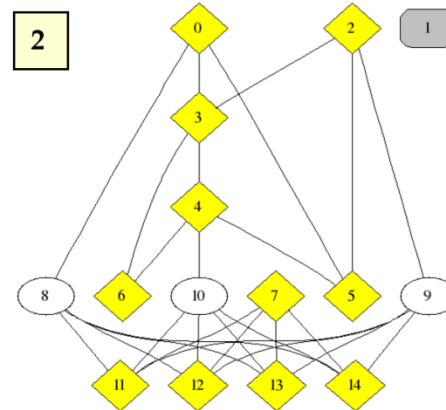
# BP in action

Initialize prior beliefs of fraudsters to  $P(f)=1$

Initialize other nodes as unbiased



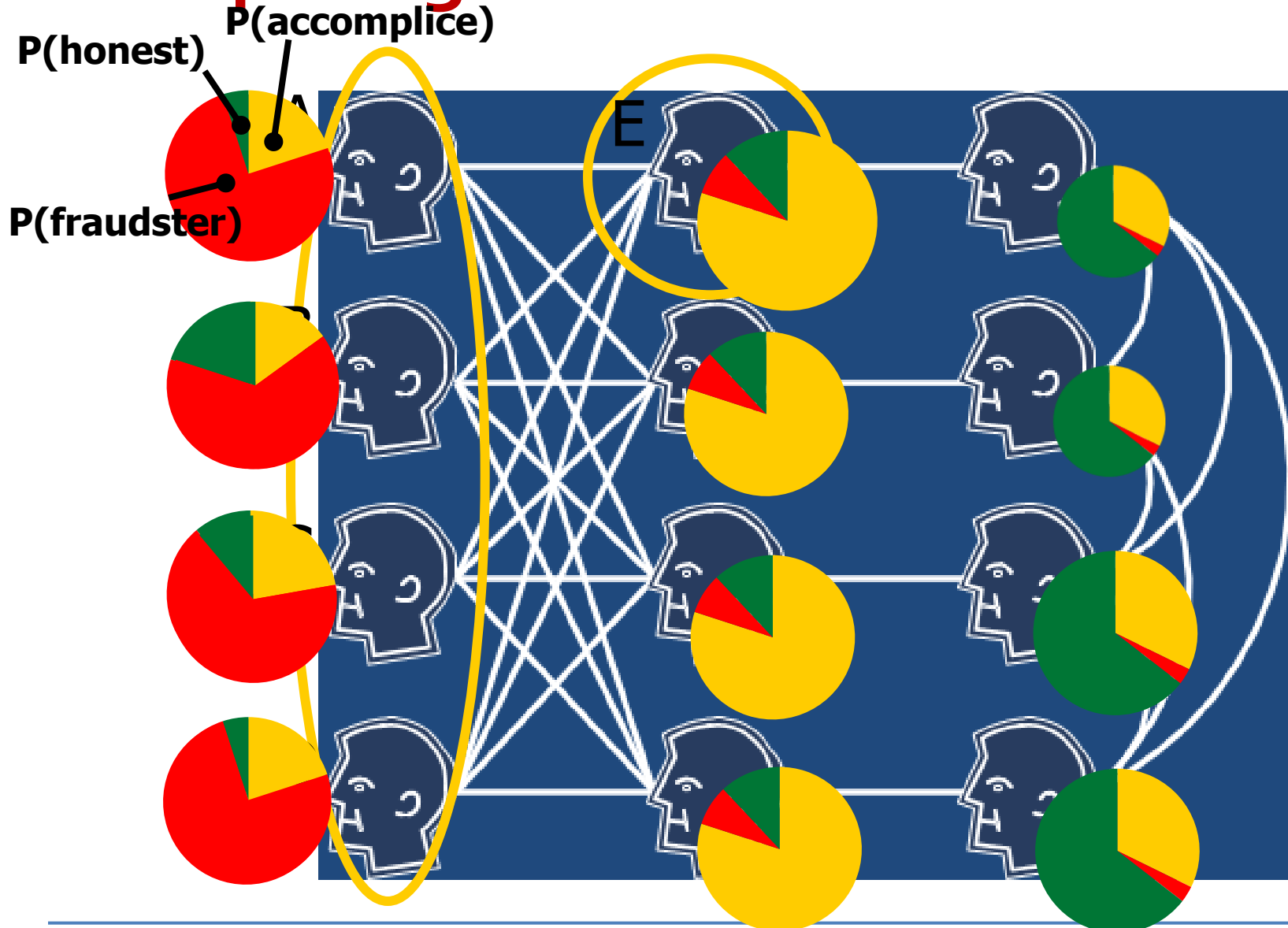
At each iteration, for each node, compute messages to its neighbors



Continue till "convergence"

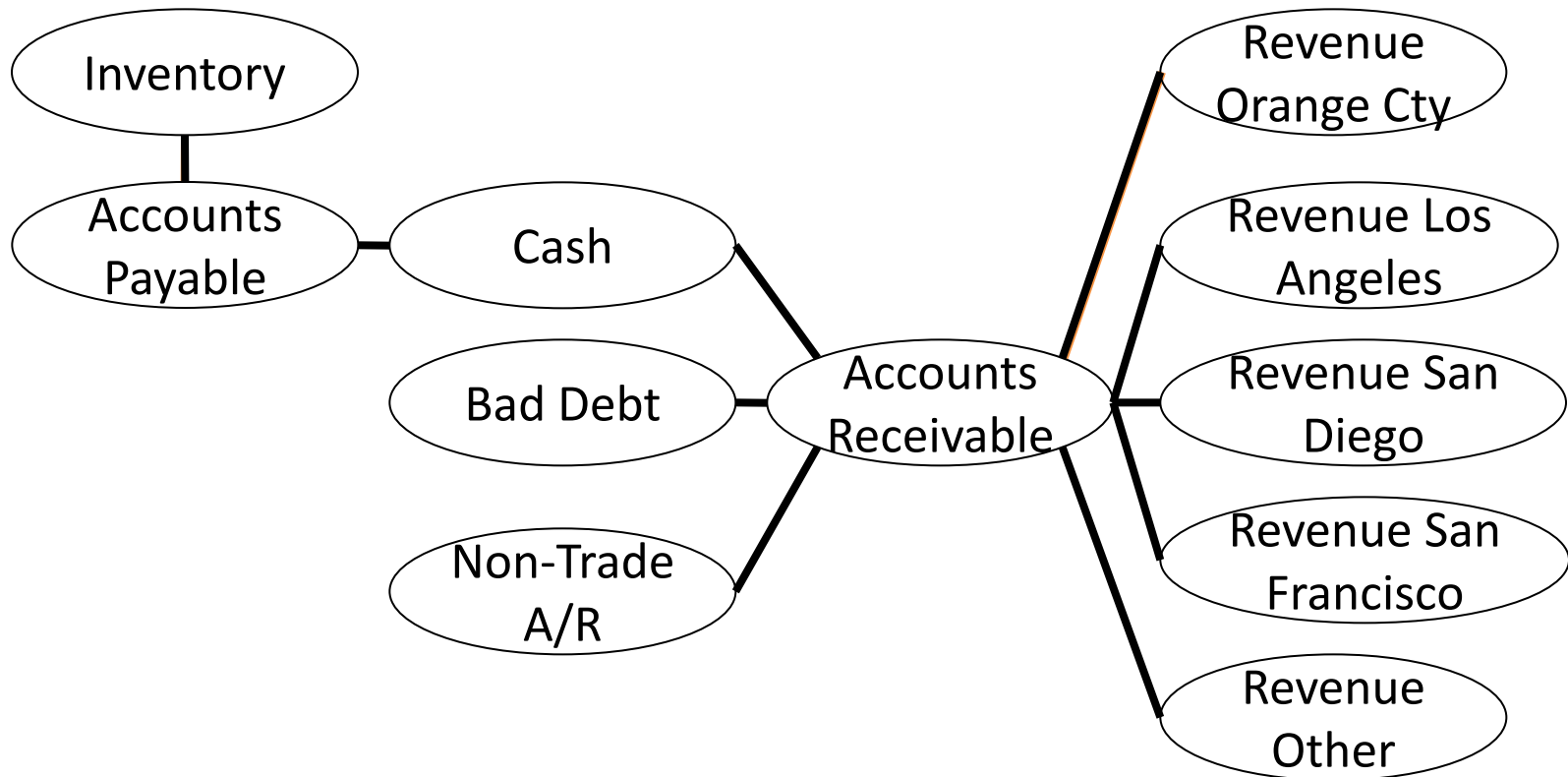
Compute beliefs, use most likely state

# Computing beliefs $\rightarrow$ roles



## (2) Accounting fraud

- **Problem:** Given accounts and their transaction relations, find most **risky** ones

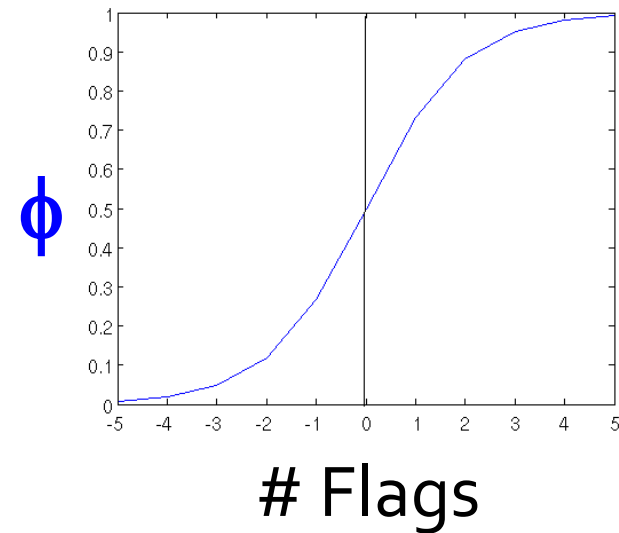
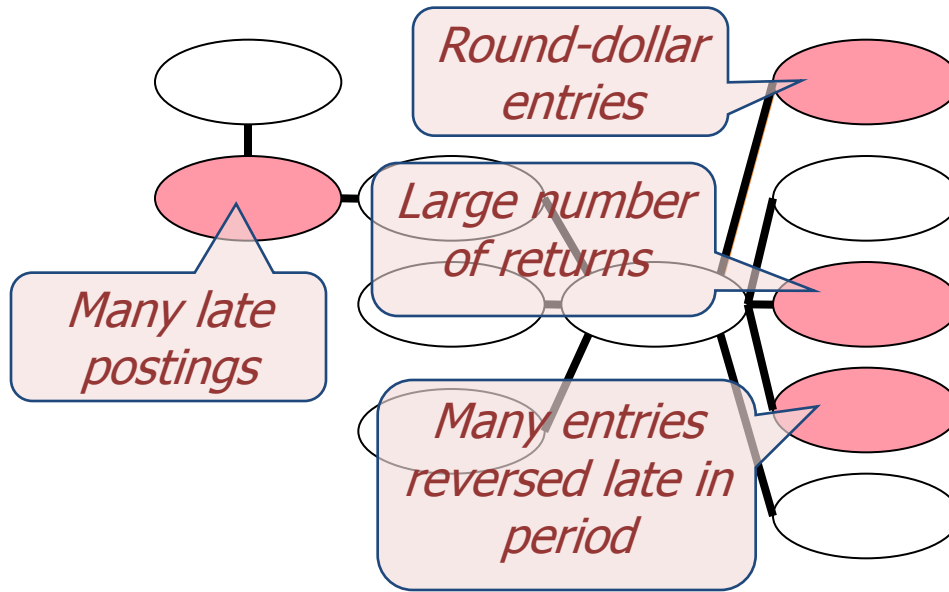


# Accounting fraud detection

- Domain knowledge to flag certain nodes  
prior beliefs
- Assume homophily (“guilt by association”)  
compatibility potentials
- Use belief propagation
  - 2 states (risky R, normal NR)
- final beliefs → end risk scores

# Social Network Analytic Risk Evaluation

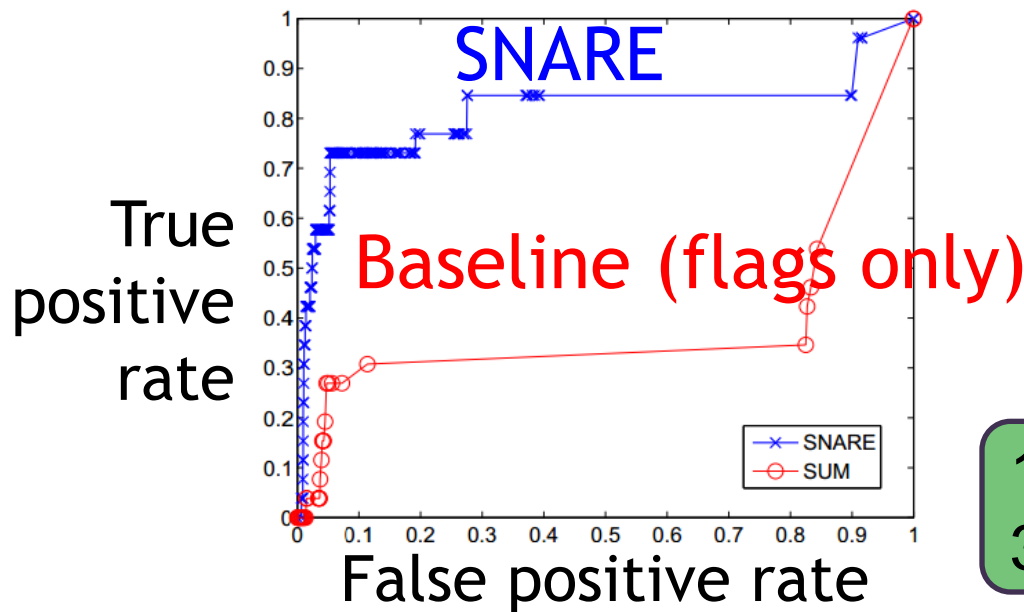
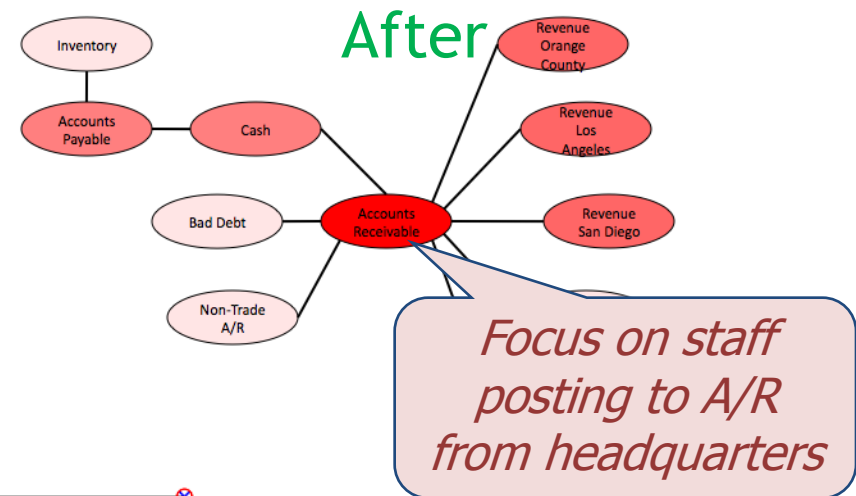
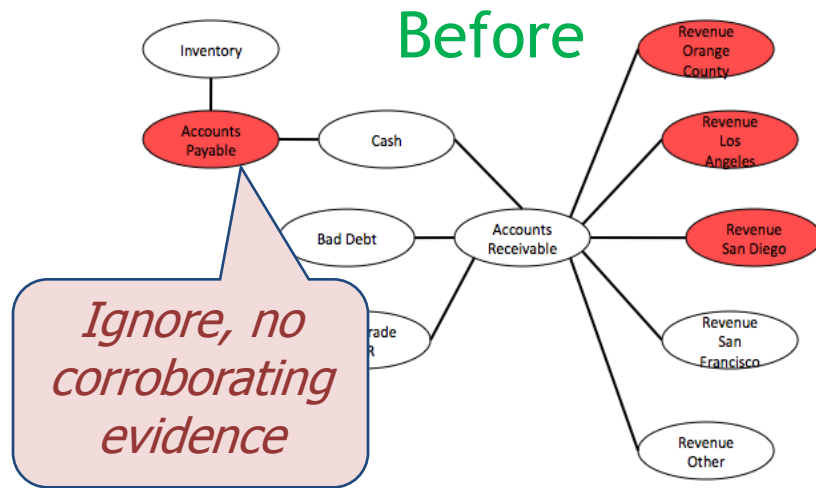
## ■ Prior beliefs (noisy domain knowledge)



## ■ Compatibility potentials (by homophily)

$\psi_{ij}(x_d, x_c)$	$v_i = x_{NR}$	$v_i = x_R$
$v_j = x_{NR}$	$1 - \epsilon$	$\epsilon$
$v_j = x_R$	$\epsilon$	$1 - \epsilon$

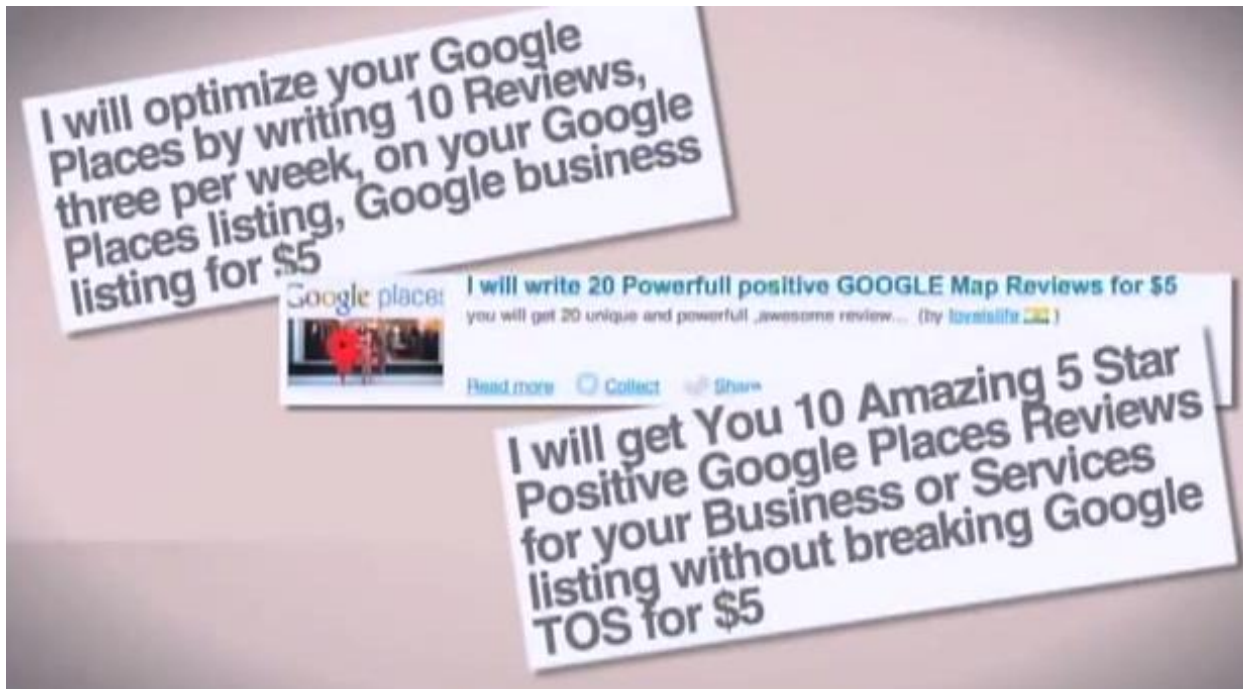
# Social Network Analytic Risk Evaluation



1380 accounts  
3820 transactions

## (3) Fake review spam

- Review sites: attractive target for spam
- Often **hype/defame** spam
- **Paid** spammers

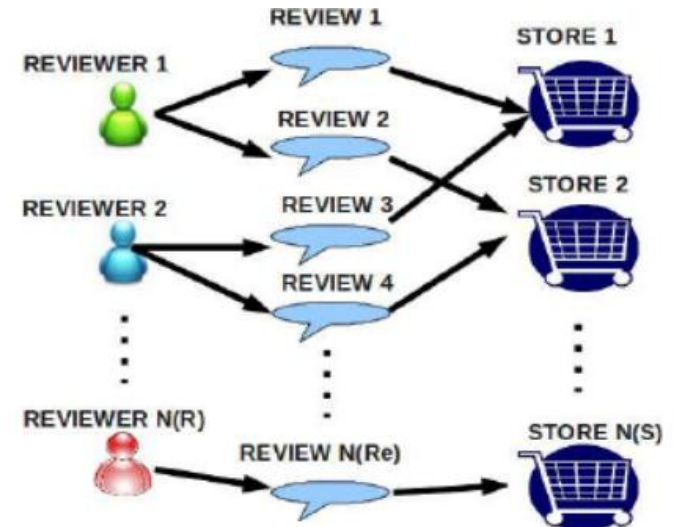


# Fake review spam detection

- Behavioral analysis [Jindal & Liu'o8]
  - individual features, geographic locations, login times, ...
- Language analysis [Liu et al.'11]
  - use of superlatives, many self-referencing, rate of misspell, many agreement words, ...

*Easy to fake!*

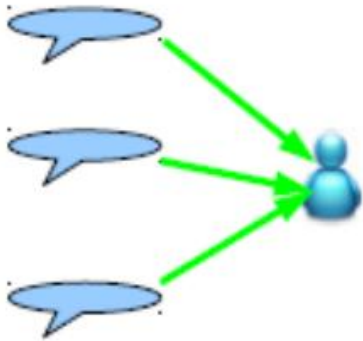
- **Hard to fake:** graph structure
- Capture **relationships** between reviewers, reviews, stores





# Graph-based detection

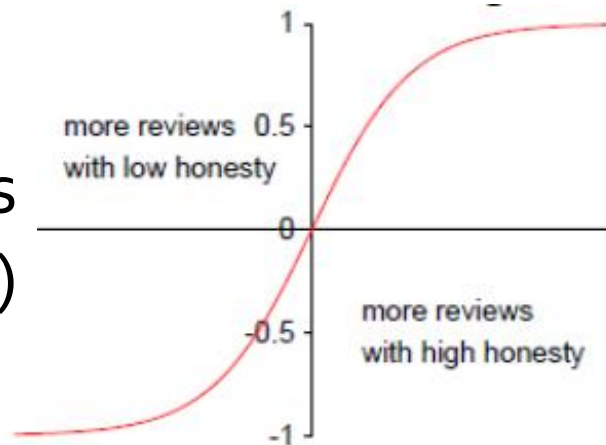
Reviewer  $r$  **trustiness**  $T(r)$



$$H_r = \sum_{i=1}^{n_r} H(\alpha_r^i)$$

$$T(r) = \frac{2}{1 + e^{-H_r}} - 1$$

Trustiness  
 $T(r)$



Honesty  $H_r$

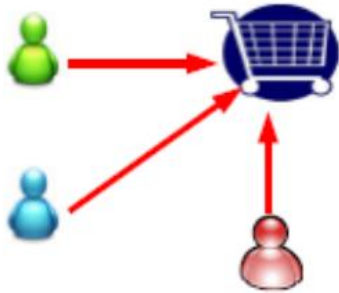


# Graph-based detection

Store  $s$  **reliability**  $R(s)$

$$\theta = \sum_{v \in U_s, T(\kappa_v) > 0} T(\kappa_v)(\Psi_v - \mu)$$

↑  
review  $v$  rating

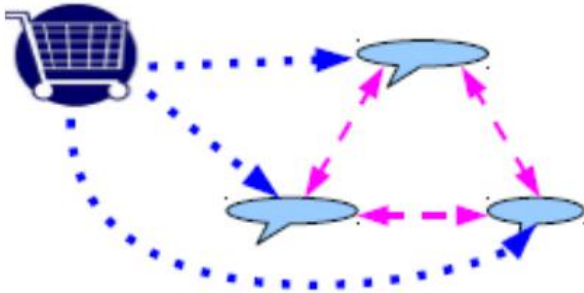


$$R(s) = \frac{2}{1 + e^{-\theta}} - 1$$



# Graph-based detection

Review v **honesty**  $H(v)$



$$A(v, \Delta t) = \sum_{i \in S_{v,a}} T(\kappa_i) - \sum_{j \in S_{v,d}} T(\kappa_j)$$

$$H(v) = |R(\Gamma_v)| A_n(v, \Delta t)$$



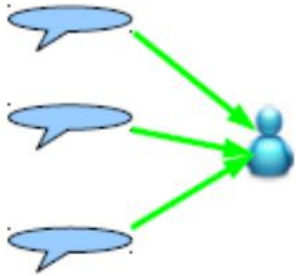
# Graph-based detection



Reviewer  $r$  **trustiness**  $T(r)$

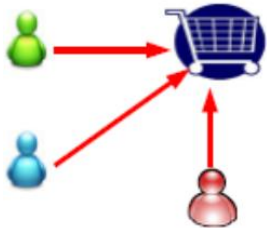
$$\underline{H_r} = \sum_{i=1}^{n_r} H(\alpha_r^i)$$

$$T(r) = \frac{2}{1 + e^{-\underline{H_r}}} - 1$$



Store  $s$  **reliability**  $R(s)$

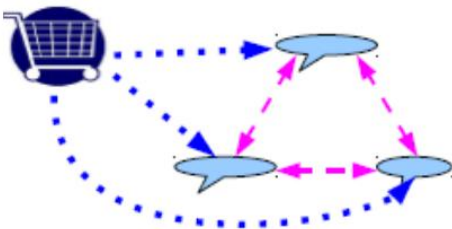
$$\theta = \sum_{v \in U_s, T(\kappa_v) > 0} \underline{T(\kappa_v)} (\Psi_v - \mu) \quad R(s) = \frac{2}{1 + e^{-\theta}} - 1$$



Review  $v$  **honesty**  $H(v)$

$$A(v, \Delta t) = \sum_{i \in S_{v,a}} \underline{T(\kappa_i)} - \sum_{j \in S_{v,d}} \underline{T(\kappa_j)}$$

$$H(v) = \underline{R(\Gamma_v)} | A_n(v, \Delta t)$$



# Graph-based detection

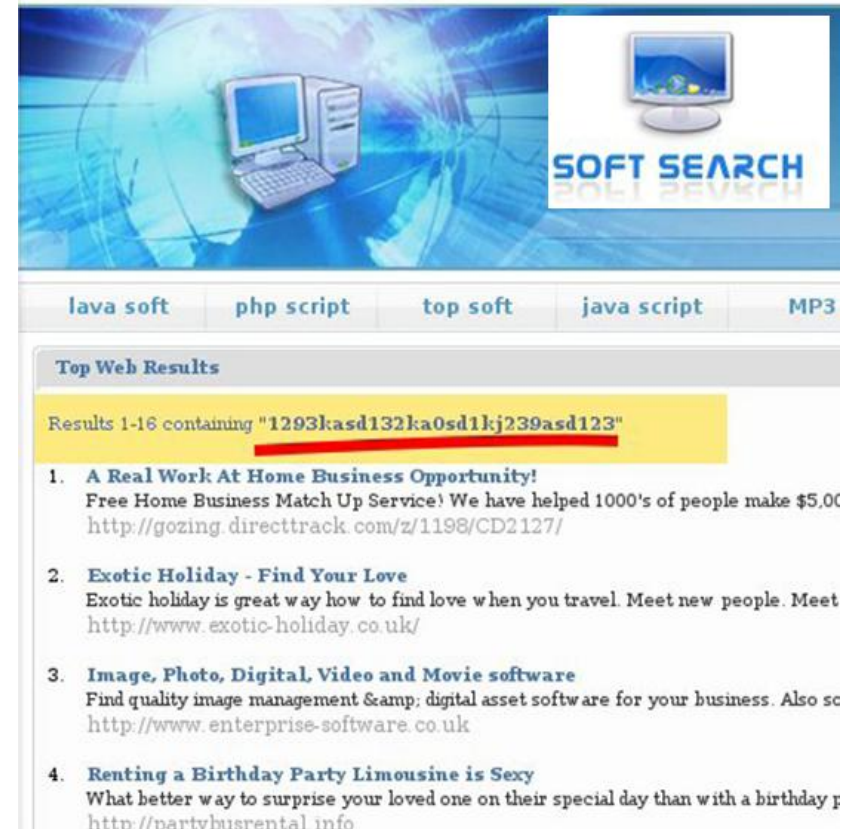
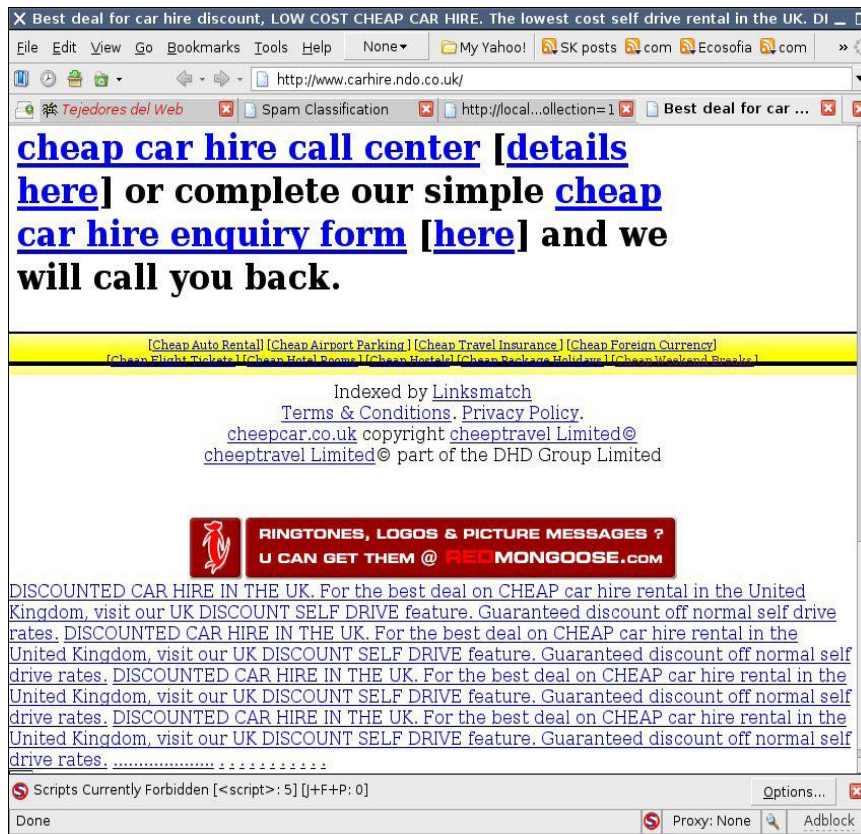
- **Algorithm:** iterate **trustiness, reliability, and honesty scores** in a mutual recursion
  - ❑ similar to Kleinberg's HITS algorithm
  - ❑ non-linear relations
  
- **Challenges:**
  - ❑ Convergence not guaranteed
  - ❑ Cannot use attribute info
  - ❑ Parameters: agreement time window  $\Delta t$ , review similarity threshold (for dis/agreement)

# Part III: Outline

- Algorithms: **relational learning**
  - Collective classification
  - Relational inference
  
- Applications: **fraud and spam** detection
  - Online auction fraud
  - Accounting fraud
  - Fake review spam
  - ➔ Web spam

# (4) Web spam

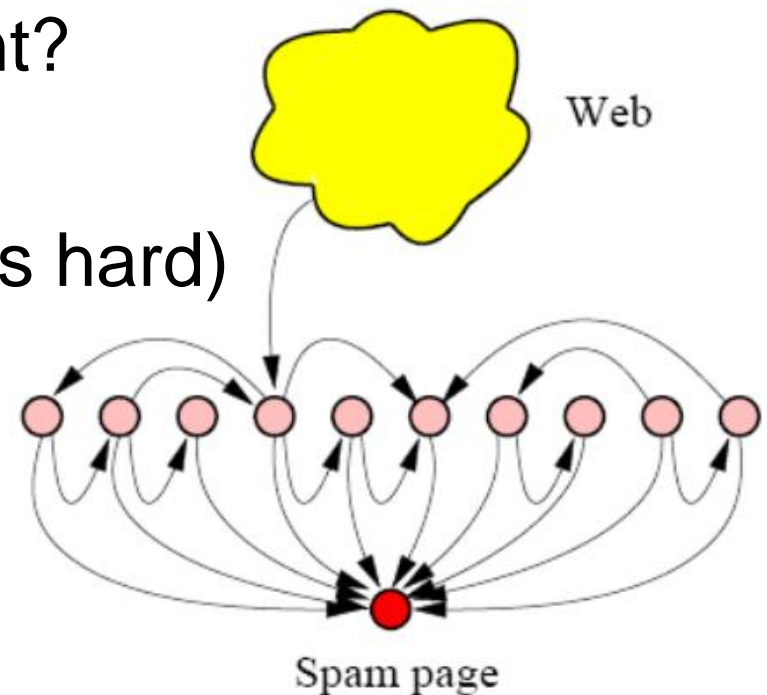
- **Spam pages:** pages designed to trick search engines to direct traffic to their websites



# Web spam

## ■ Challenges:

- ❑ pages are not independent
- ❑ what features are relevant?
- ❑ small training set
- ❑ noisy labels (consensus is hard)
- ❑ content very dynamic



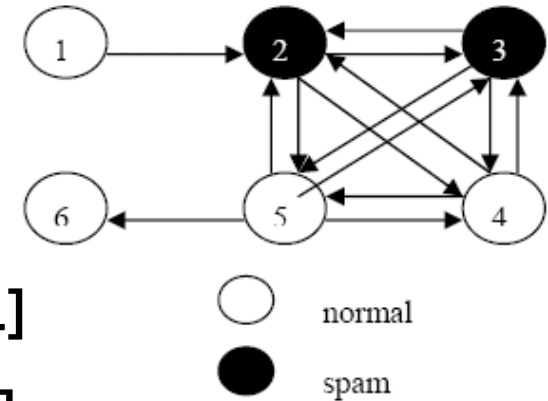




# Web spam

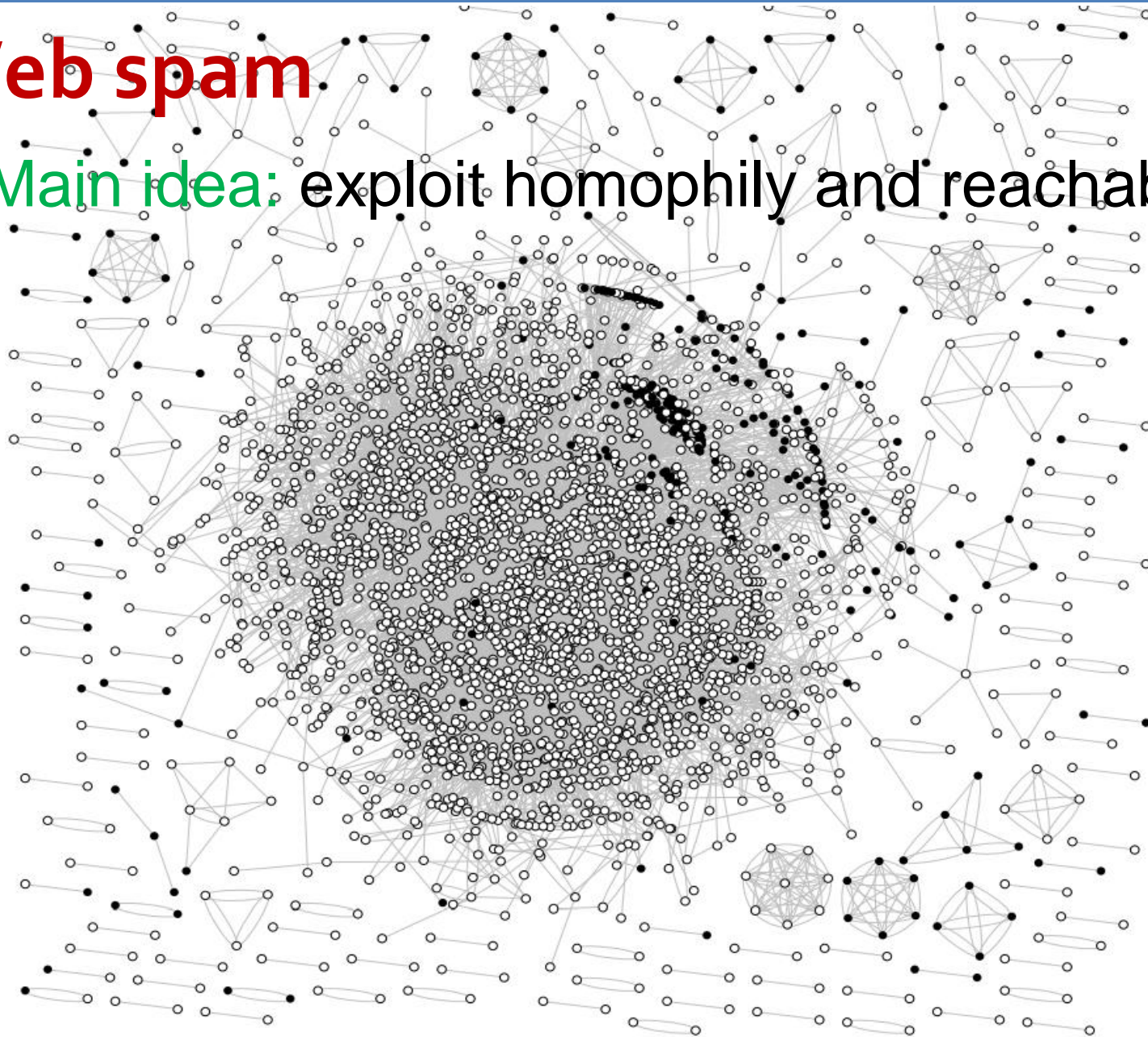
## ■ Many **graph-based** solutions

- ❑ TrustRank [Gyöngyi et al. '04]
- ❑ SpamRank [Benczur et al. '05]
- ❑ Anti-trustRank [Krishnan et al. '06]
- ❑ Propagating trust and distrust [Wu et al. '06]
- ❑ Know your neighbors [Castillo et al. '07]
- ❑ Guilt-by-association [Kang et al. '11]
- ❑ ...



# Web spam

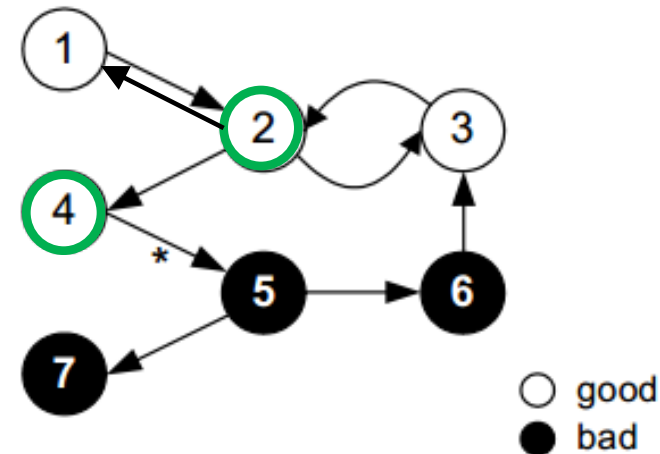
- Main idea: exploit homophily and reachability



# TrustRank: combating web spam

## ■ Main steps:

- ❑ Find **seed set S** of “good” pages (e.g. using oracle)
- ❑ Compute trust scores by **biased (personalized) PageRank** from good pages



## ■ **Intuition:** spam pages are hardly reachable from trustworthy pages

- ❑ Hard to acquire direct inlinks from good pages

# TrustRank mathematically

- Remember PageRank score of a page  $p$ :

$$\mathbf{r}(p) = \alpha \cdot \sum_{q:(q,p) \in \mathcal{E}} \frac{\mathbf{r}(q)}{\omega(q)} + (1 - \alpha) \cdot \frac{1}{N}$$

- In closed form:

$$\mathbf{r} = \alpha \cdot \mathbf{T} \cdot \mathbf{r} + (1 - \alpha) \cdot \frac{1}{N} \cdot \mathbf{1}_N \quad \mathbf{T}(p, q) = \begin{cases} 0 & \text{if } (q, p) \notin \mathcal{E}, \\ 1/\omega(q) & \text{if } (q, p) \in \mathcal{E}. \end{cases}$$

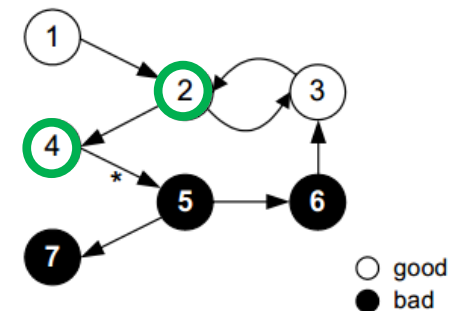
damping factor

Transition matrix

- Personalized PageRank:

$$\mathbf{r} = \alpha \cdot \mathbf{T} \cdot \mathbf{r} + (1 - \alpha) \cdot \mathbf{d}$$

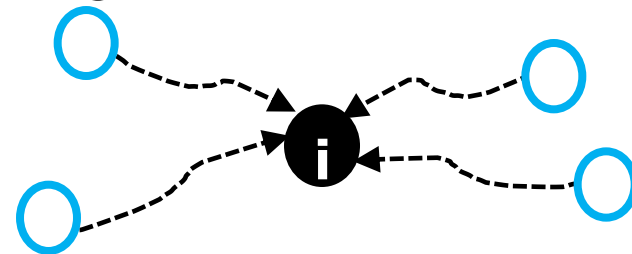
$\mathbf{1}/|S|$  for  $S$  nodes of interest (seeds)



$$\mathbf{d} = [0, \frac{1}{2}, 0, \frac{1}{2}, 0, 0, 0]$$

# SpamRank: link spam detection

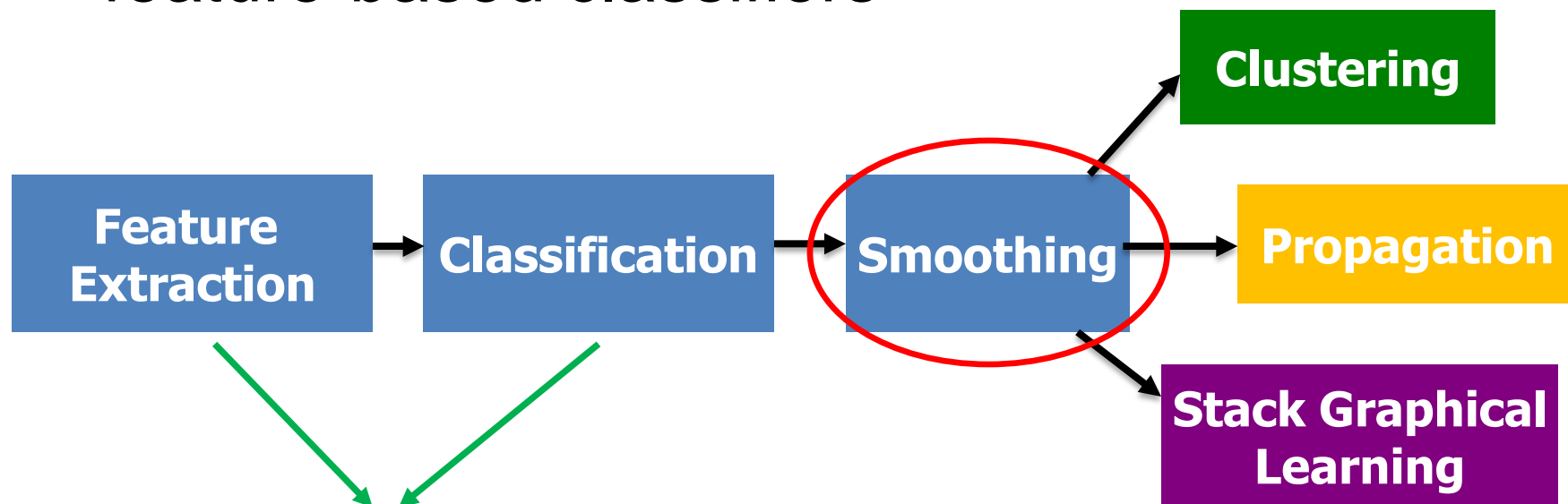
- **Intuition:** PageRank distribution of “good” set of **supporters** should be power law (as in entire Web)
  - Page  $v$  is a **supporter** of page  $i$  if:  $PPR_i(v) > 0$
- For each page  $i$ 
  - get PageRank scores of all supporters of  $i$
  - test PageRank histogram for power law
  - calculate irregularity score  $s(i)$
- $\text{SpamRank} \leftarrow PPR(\vec{s})$



**Note:** no user labeling (as for TrustRank)

# “Know your neighbors”

- Graph-based techniques can help improve feature-based classifiers

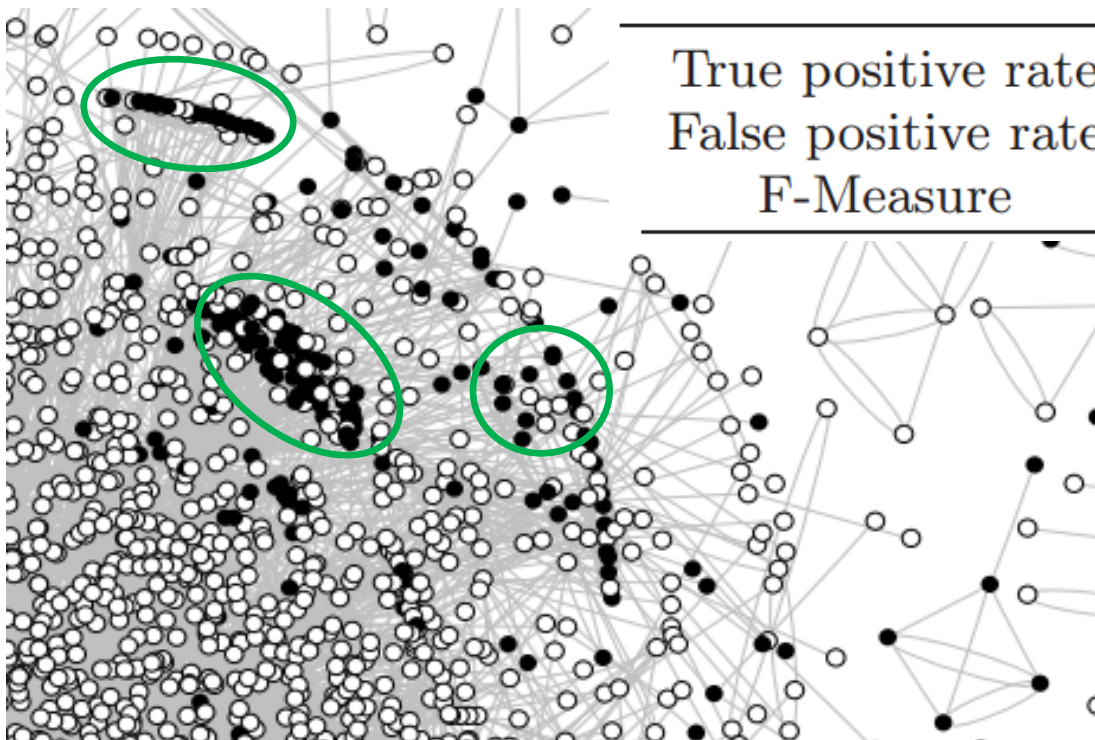


- Graph features: reciprocity, assortativity, TrustRank, PageRank, ...
- Content features: fraction visible text, compression rate, entropy of trigrams, ...



# Smoothing –clustering

- Split graph into many **clusters**. (e.g. by METIS)
- If **majority** of nodes in cluster are **spam**, then **all** pages in cluster are **spam**.

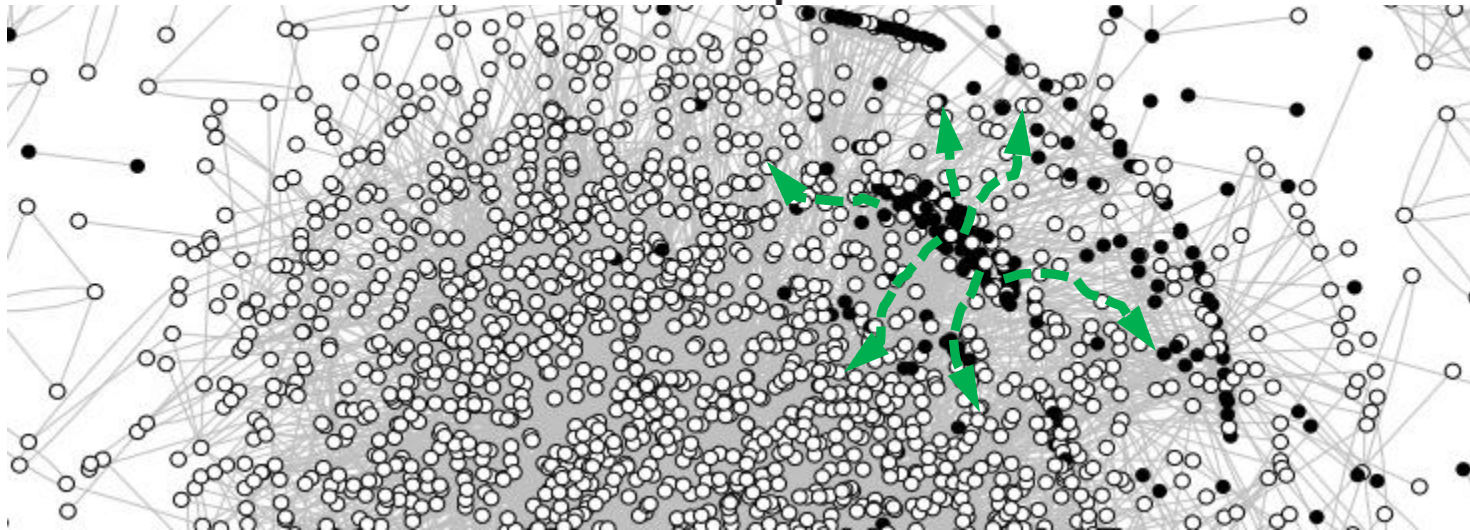


	Baseline	Clustering
True positive rate	78.7%	76.9%
False positive rate	5.7%	5.0%
F-Measure	0.723	<b>0.728</b>

# Smoothing –propagation

- Propagate predictions using random walks.
- $PPR(\xi)$ ;  $s(i)$ : spamicity score by baseline classifier (backward and/or forwards steps)

	Baseline	Fwds.	Backwds.	Both
True positive rate	78.7%	76.5%	75.0%	75.2%
False positive rate	5.7%	5.4%	4.3%	4.7%
F-Measure	0.723	0.716	<b>0.733</b>	0.724

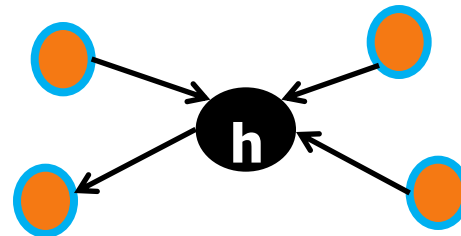




# Smoothing –stacked learning

- Create additional features by combining predictions for related nodes
  - e.g., avg. **spamicity score**  $p$  of **neighbors**  $r(h)$  of  $h$

$$f(h) = \frac{\sum_{g \in r(h)} p(g)}{|r(h)|}$$



- similar to pRN classifier by Macskassy&Provost
- can repeat, although 1-2 steps add most gain

	Baseline	First pass	Second pass
True positive rate	78.7%	85.2%	88.4%
False positive rate	5.7%	6.1%	6.3%
F-Measure	0.723	0.750	<b>0.763</b>

## Part III: References (alg.s and app.s)

- P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. [Collective Classification in Network Data](#). AI Magazine, 29(3):93-106, 2008.
- S. A. Macskassy and F. Provost. [A Simple Relational Classifier](#). KDD Workshops, 2003.
- S. Pandit, D. H. Chau, S. Wang, C. Faloutsos. [NetProbe: A Fast and Scalable System for Fraud Detection in Online Auction Networks](#). WWW, 2007.
- M. McGlohon, S. Bay, M. G. Anderle, D. M. Steier, C. Faloutsos. [SNARE: a link analytic system for graph labeling and risk detection](#). KDD, 2009.
- Zhongmou Li, Hui Xiong, Yanchi Liu, Aoying Zhou. [Detecting Blackhole and Volcano Patterns in Directed Networks](#). ICDM, pp 294-303, 2010.

## Part III: References (alg.s and app.s) (2)

- G. Wang, S. Xie, B. Liu, P. S. Yu. [Review Graph based Online Store Review Spammer Detection](#). ICDM, 2011.
- Zoltán Gyöngyi , Hector Garcia-molina , Jan Pedersen. [Combating web spam with TrustRank](#). VLDB, 2004.
- Andras A. B., Karoly C., Tamas S., Mate U. [SpamRank - Fully Automatic Link Spam Detection](#). AIRWeb, 2005.
- Vijay Krishnan, Rashmi Raj: [Web Spam Detection with Anti-Trust Rank](#). AIRWeb, pp. 37-40, 2006.
- Baoning W., Vinay G., and Brian D. D.. [Propagating Trust and Distrust to Demote Web Spam](#). WWW, 2006.
- Castillo, C. and Donato, D. and Gionis, A. and Murdock, V. and Silvestri, F. [Know your neighbors: web spam detection using the web topology](#). SIGIR, pp. 423-430, 2007.

# Other references

- Matthew J. Rattigan, David Jensen: [The case for anomalous link discovery](#). SIGKDD Explorations 7(2): 41-47 (2005)
- Chao Liu, Xifeng Yan, Hwanjo Yu, Jiawei Han, Philip S. Yu. [Mining Behavior Graphs for "Backtrace" of Noncrashing Bugs](#). SDM, 2005.
- Shetty, J. and Adibi, J. [Discovering Important Nodes through Graph Entropy: The Case of Enron Email Database](#). KDD, workshop, 2005.
- Boden B., Günnemann S., Hoffmann H., Seidl T. [Mining Coherent Subgraphs in Multi-Layer Graphs with Edge Labels](#). KDD 2012.
- K. Henderson, B. Gallagher, T. Eliassi-Rad, H. Tong, S. Basu, L. Akoglu, D. Koutra, L. Li, C. Faloutsos. [RoIX: Structural Role Extraction & Mining in Large Graphs](#). KDD, 2012.
- J. Neville, O. Simsek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg. [Using Relational Knowledge Discovery to Prevent Securities Fraud](#). KDD, pp. 449–458, 2005.
- H. Bunke and K. Shearer. [A graph distance metric based on the maximal common subgraph](#). Pattern Rec. Let., 19(3-4):255–259, 1998.

# Other references

- B. T. Messmer and H. Bunke. [A new algorithm for error-tolerant subgraph isomorphism detection](#). Pattern Analysis and Machine Intelligence, IEEE Transactions on, 20(5):493–504, 1998.
- P. J. Dickinson, H. Bunke, A. Dadej, M. Kraetzl: [Matching graphs with unique node labels](#). Pattern Anal. Appl. 7(3): 243-254 (2004)
- Peter J. Dickinson, Miro Kraetzl, Horst Bunke, Michel Neuhaus, Arek Dadej: [Similarity Measures For Hierarchical Representations Of Graphs With Unique Node Labels](#). IJPRAI 18(3): 425-442 (2004)
- Kraetzl, M. and Wallis, W. D., [Modality Distance between Graphs](#). Utilitas Mathematica, 69, 97–102, 2006.
- Gaston, M. E., Kraetzl, M. and Wallis, W. D., [Graph Diameter as a Pseudo-Metric for Change Detection in Dynamic Networks](#), Australasian Journal of Combinatorics, 35, 299—312, 2006.
- B. Pincombe. [Detecting changes in time series of network graphs using minimum mean squared error and cumulative summation](#). ANZIAM J. 48, pp.C450–C473, 2007.

# Tutorial Outline

- Motivation, applications, challenges
- **Part I: Anomaly detection in static data**
  - Overview: Outliers in **clouds of points**
  - Anomaly detection in **graph data**
- **Part II: Event detection in dynamic data**
  - Overview: Change detection in **time series**
  - Event detection in **graph sequences**
- **Part III: Graph-based algorithms and apps**
  - Algorithms: **relational learning**
  - Applications: **fraud and spam detection**

# Conclusions

- Graphs are powerful tools to detect
  - **Anomalies**
  - **Events**
  - **Fraud/Spam**in complex real-world data (attributes, (noisy) side information, weights, ...)
- Nature of the problem highly dependent on the application domain
- Each problem formulation needs a different approach

# Open challenges

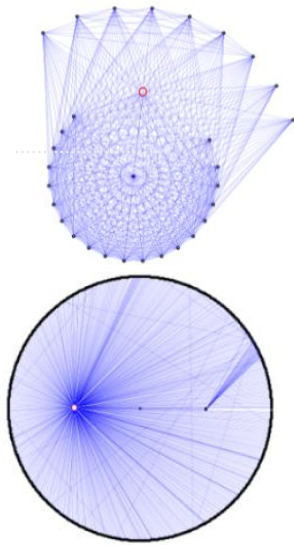
- Anomalies in dynamic graphs
  - dynamic **attributed** graphs (definitions, formulations, real-world scenarios)
  - temporal effects: node/edge **history** (not only updates)
- Fraud/spam detection
  - adversarial **robustness**
  - **cost** (to system in measurement , to adversary to fake, to user in exposure)
  - detection **timeliness** and other system design aspects; e.g. dynamicity, latency



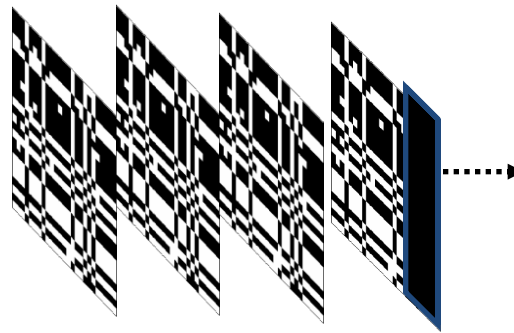
# Q & A

[leman@cs.stonybrook.edu](mailto:leman@cs.stonybrook.edu)

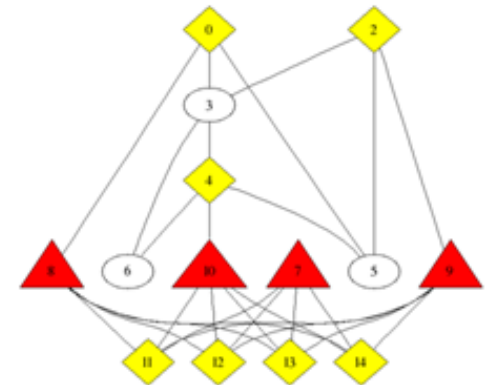
<http://www.cs.stonybrook.edu/~leman/>



**anomalies**



**events**



**fraud/spam**