Graphical Models

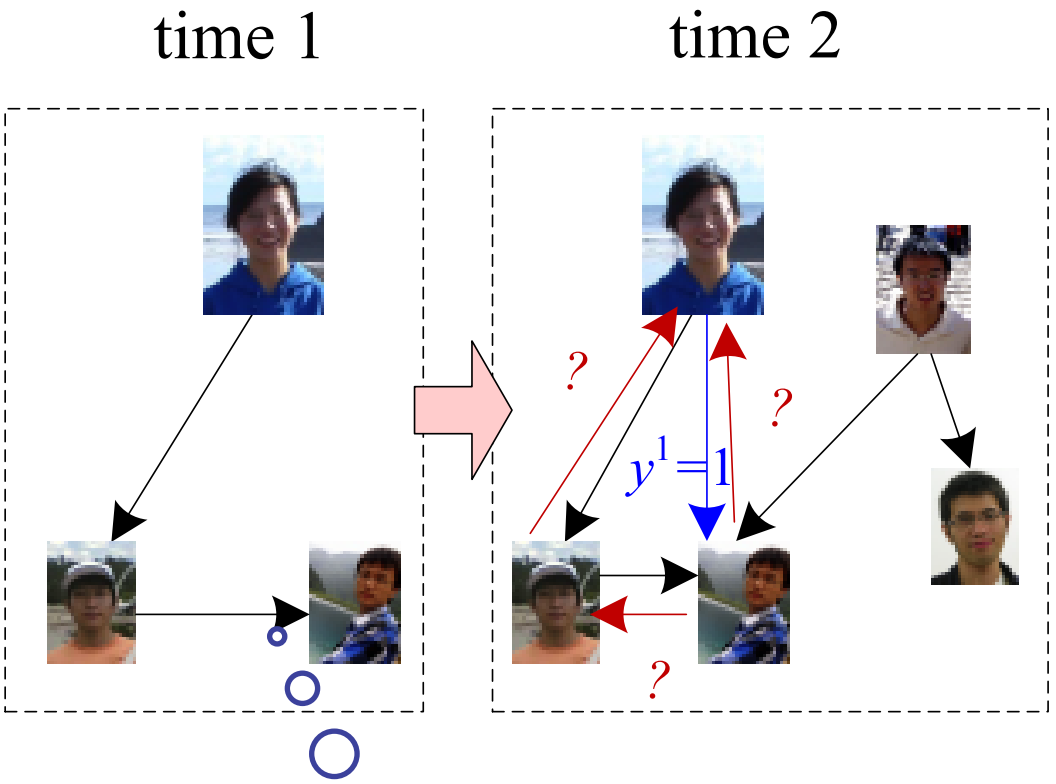
Jie Tang

Department of Computer Science and Technology
Tsinghua University

Outline

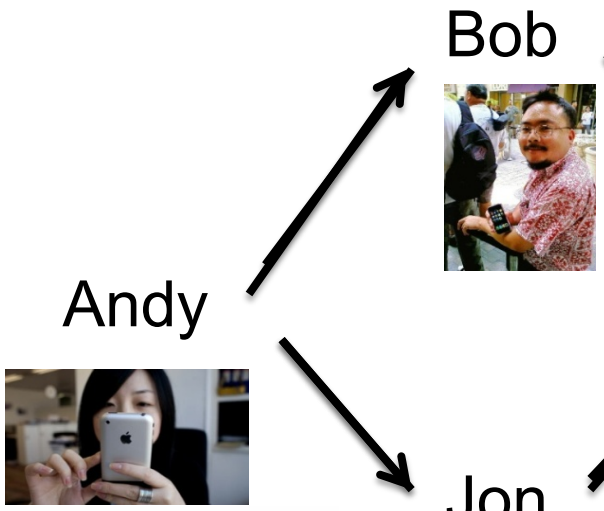
- Introduction
- Directed Graphical Models
- Undirected Graphical Models
- Factor Graph

Follow back Prediction



When you follow a friend in Twitter, how likely he will follow back?

Retweet Predicting



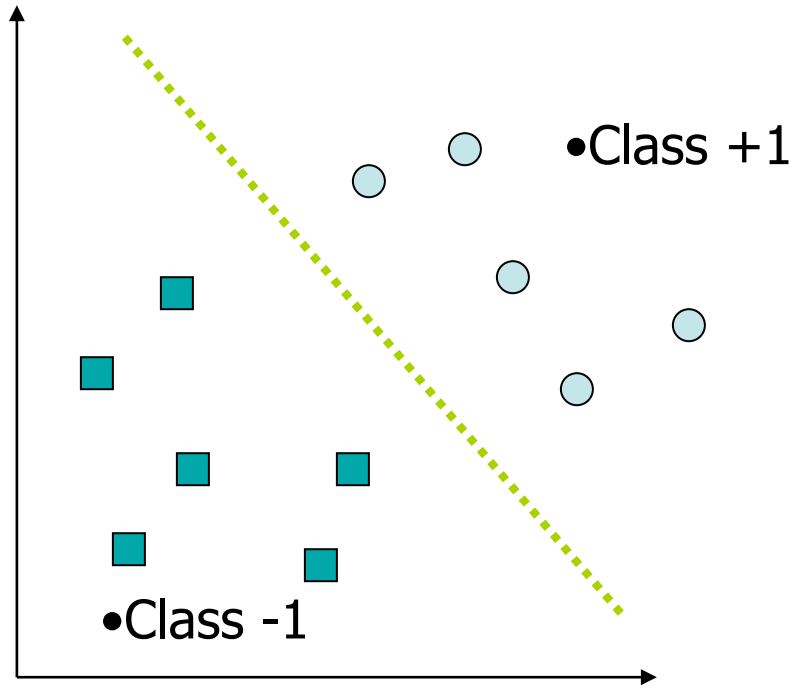
Who will retweet it?

When you post a tweet…





Binary Classifier



Sequence Labeling

- Pos Tagging
  - E.g. [He/PRP] [reckons/VBZ] [the/DT] [current/JJ] [account/NN] [deficit/NN] [will/MD] [narrow/VB] [to/TO] [only/RB] [#/#] [1.8/CD] [billion/CD] [in/IN] [September/NNP] [./.]

- Term Extraction
  - Rockwell International Corp.'s Tulsa unit said it signed a tentative agreement extending its contract with Boeing Co. to provide structural parts for Boeing's 747 jetliners.

IE from Web Page

October 14, 2002, 4:00 a.m. PT

For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open -source software with Orwellian fervor, denouncing its communal licensing as a "cancer" that stifled technological innovation.

Today, Microsoft claims to "love" the open-source concept, by which software code is made public to encourage improvement and development by outside programmers. Gates himself says Microsoft will gladly disclose its crown jewels--the coveted code behind the Windows operating system--to select customers.

"We can be open source. We love the concept of shared source," said Bill Veghte, a Microsoft VP. "That's a super-important shift for us in terms of code access."

Richard Stallman, founder of the Free Software Foundation, countered saying…

* Microsoft Corporation
CEO
Bill Gates

* Microsoft
Gates

* Microsoft
Bill Veghte

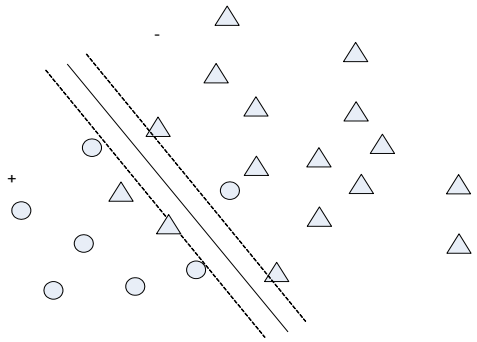
* Microsoft
VP

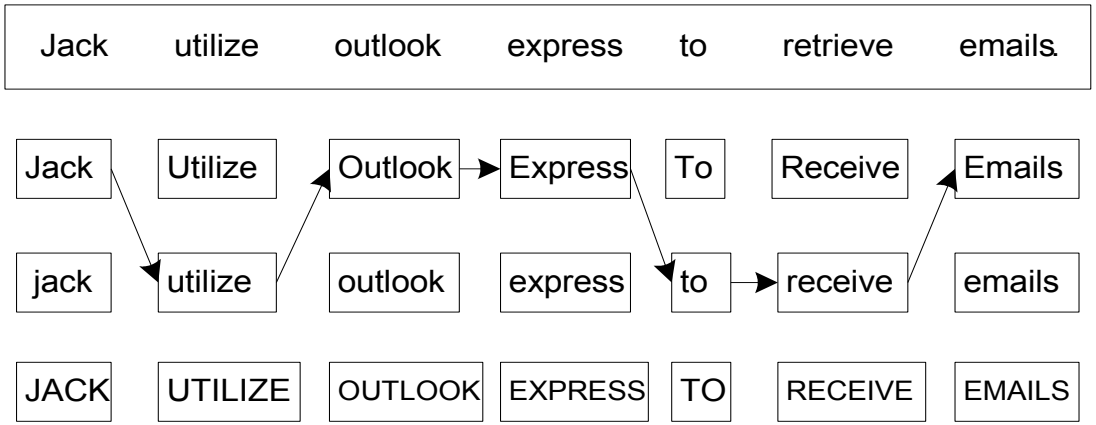
Richard Stallman
founder
Free Software Foundation

NAME	TITLE	ORGANIZATION
Bill Gates	CEO	Microsoft
Bill Veghte	VP	Microsoft
Richard Stallman	founder	Free Soft..

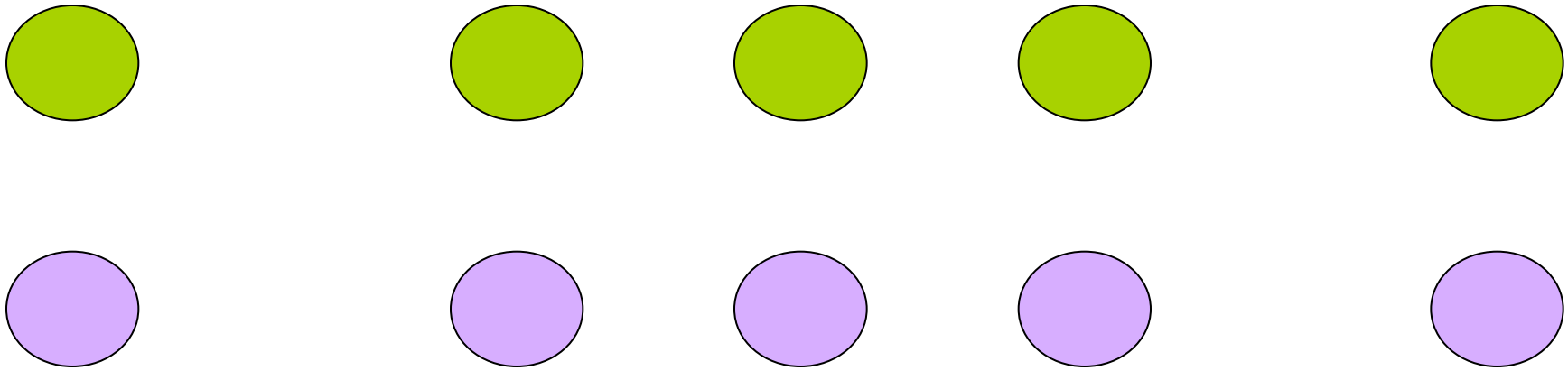
Binary Classifier vs. Sequence Labeling

- Case restoration
  - "jack utilize outlook express to retrieve emails"
  - E.g. SVMs vs. CRFs



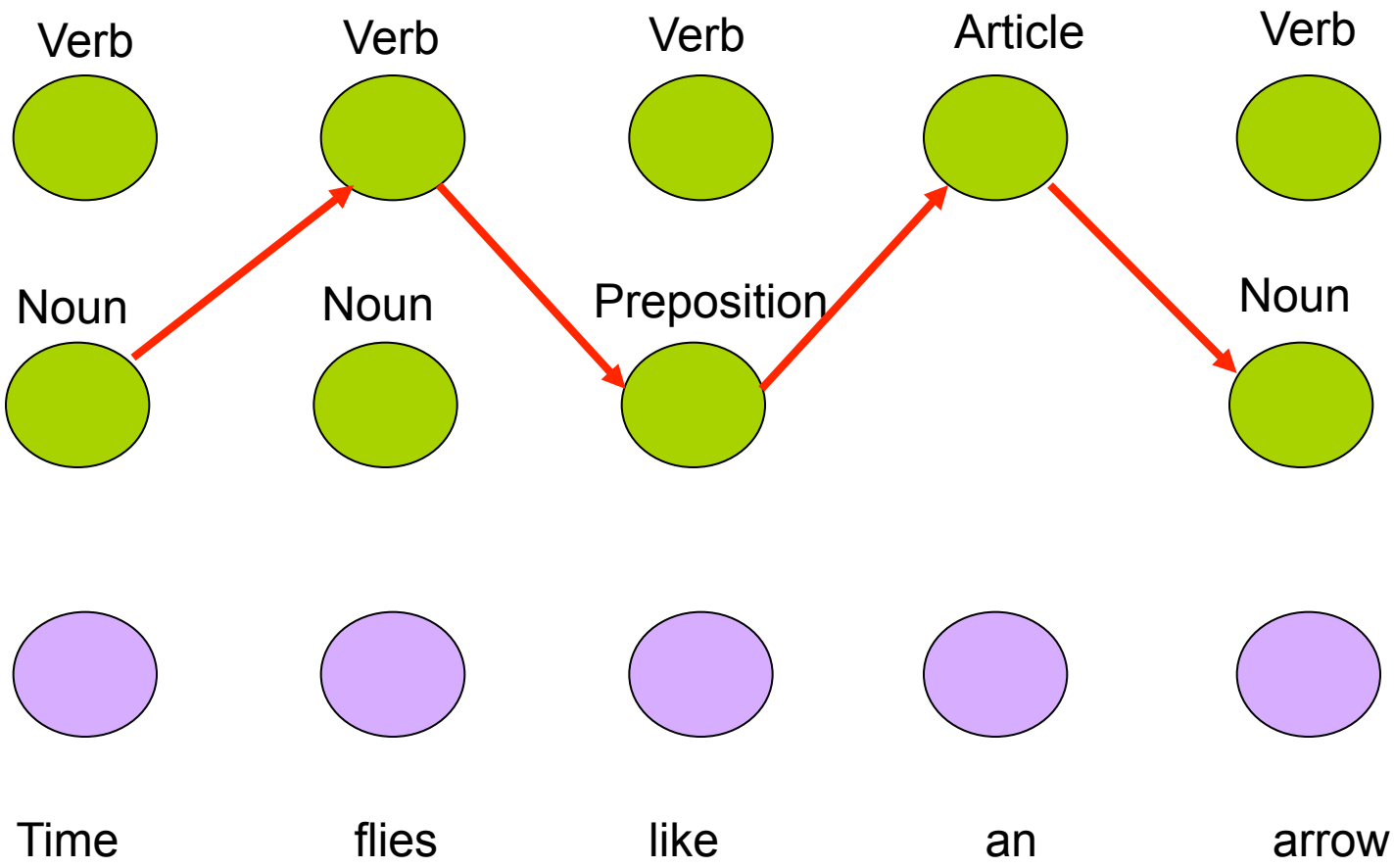


Sequence Labeling Problem

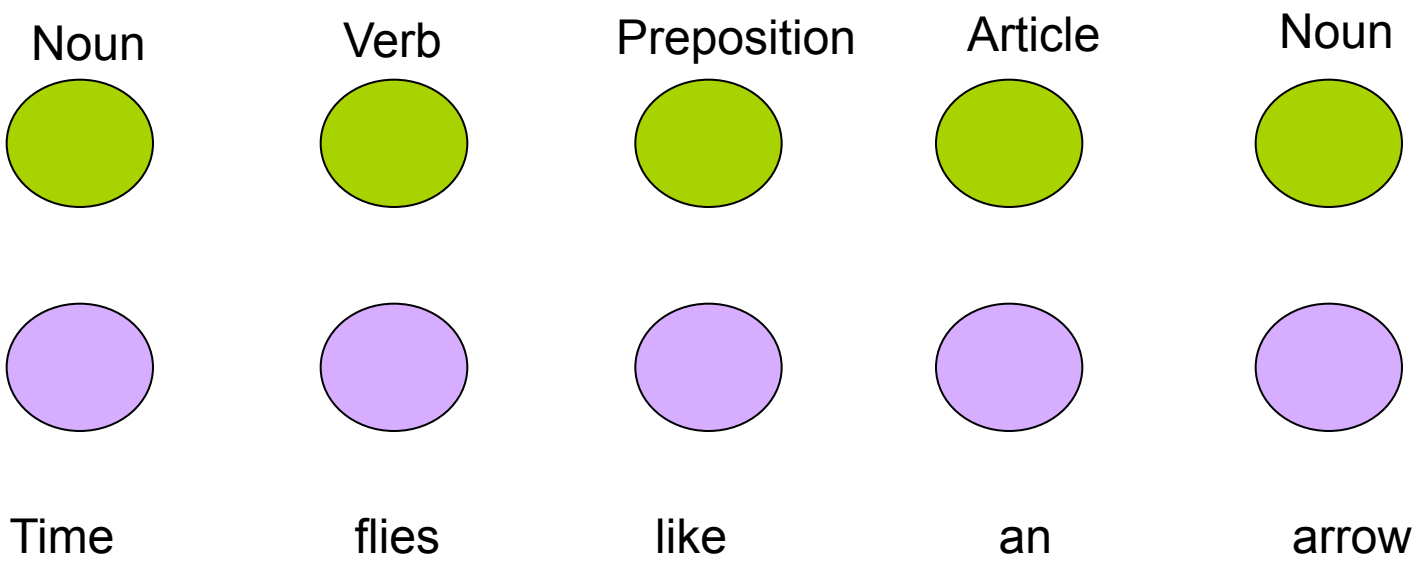


- Green nodes are ***states***
- Purple nodes are ***observations***

Example: POS Tagging Problem



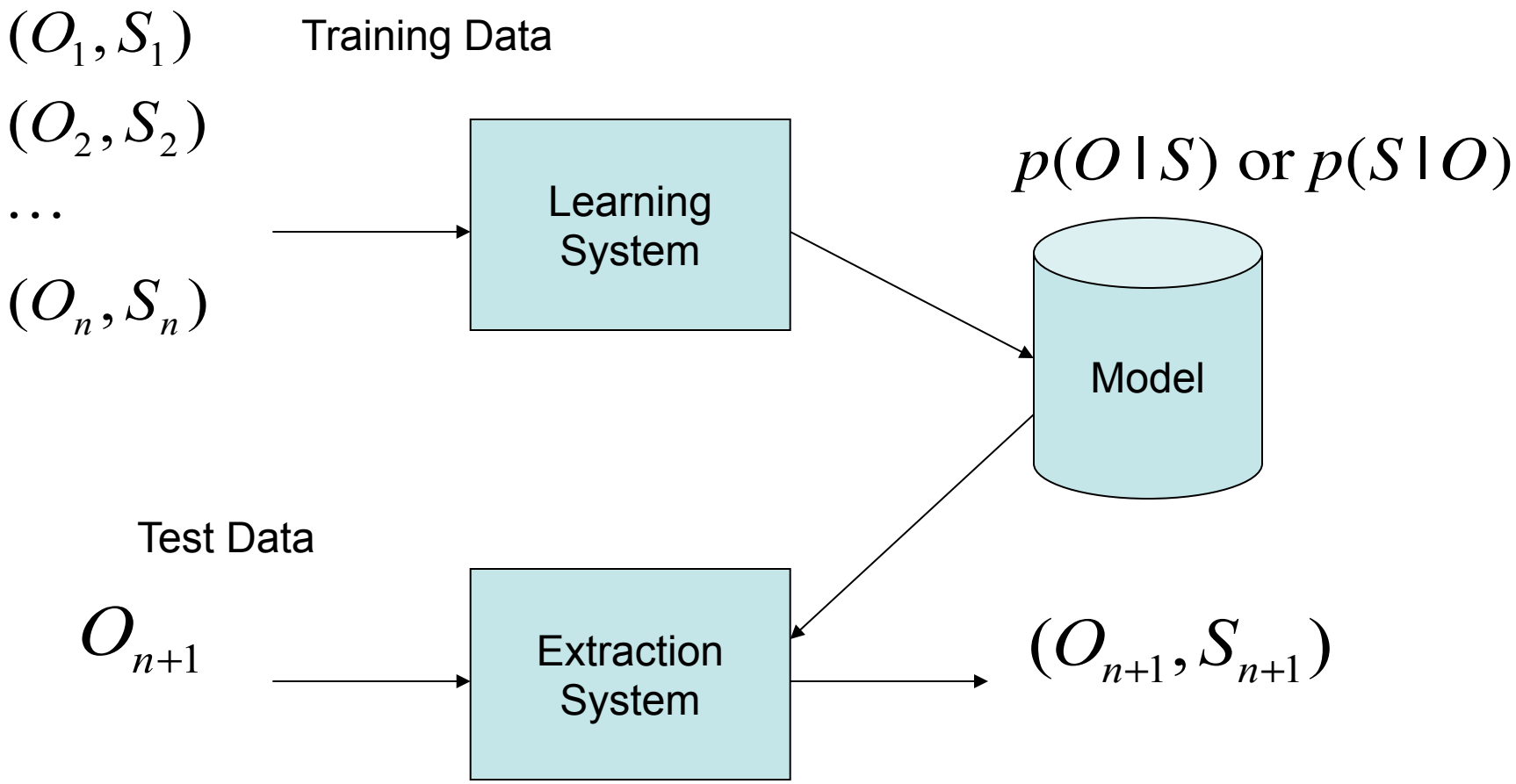
Example: POS Tagging Problem



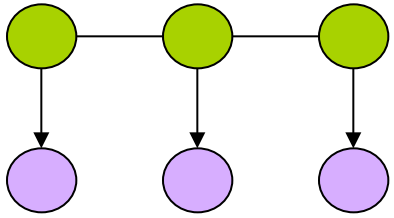
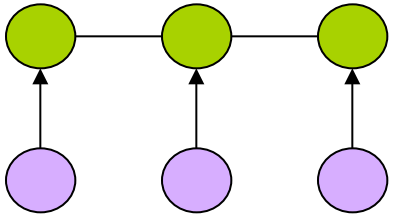
Sequence Labeling Models

- HMM
  - Generative model
  - E.g. Ghahramani (1997), Manning and Schutze (1999)
- MEMM
  - Conditional model
  - E.g. Berger and Pietra (1996), McCallum and Freitag (2000)
- CRFs
  - Conditional model without label bias problem
  - Linear-Chain CRFs
    - E.g. Lafferty and McCallum (2001), Wallach (2004)
  - Non-Linear Chain CRFs
    - Modeling more complex interaction between labels: DCRFs, 2D-CRFs, TCRFs
    - E.g. Sutton and McCallum (2004), Zhu and Nie (2005), Tang et al. (2006)

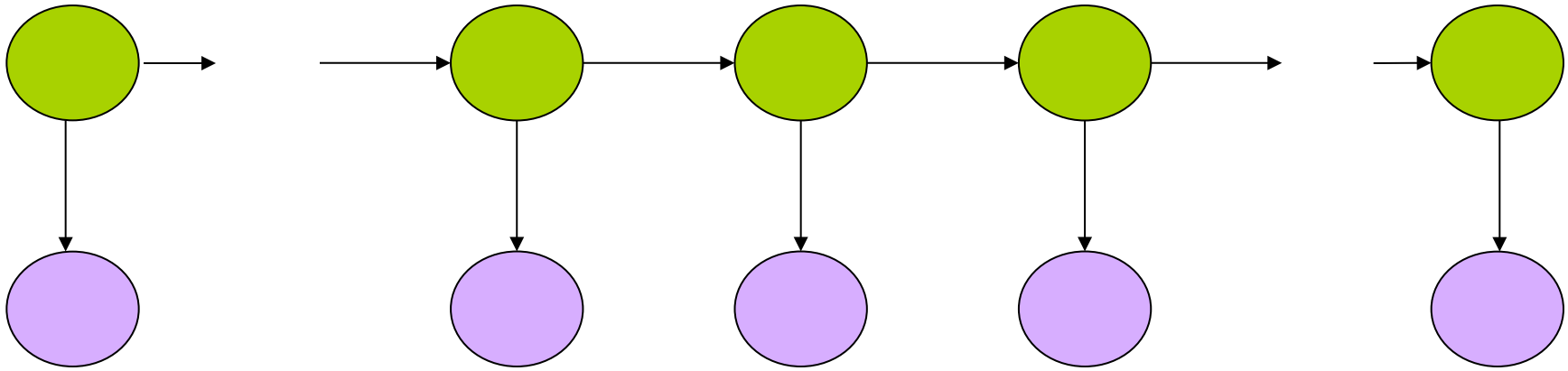
General Framework



Generative vs. Discriminative

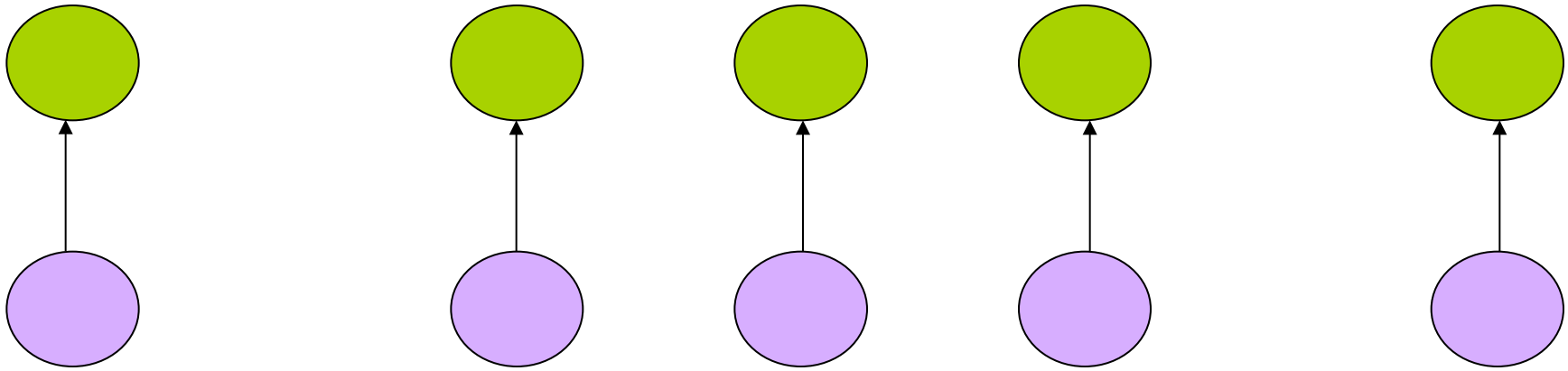
Generative	Discriminative
$p(O \mid S)$	$p(S \mid O)$
Example: HMM	Example: MaxEnt, MEMM, CRF
 States generates observations	 Observations (features) determine states
Learning = finding model generating observation sequence from state sequence	Learning = finding model mapping observation sequence to state sequence
Tagging = finding most likely state sequence having generated given observation sequence	Tagging = finding most likely state sequence mapped from given observation sequence

Assumption 1: Generative Locally Dependent Model



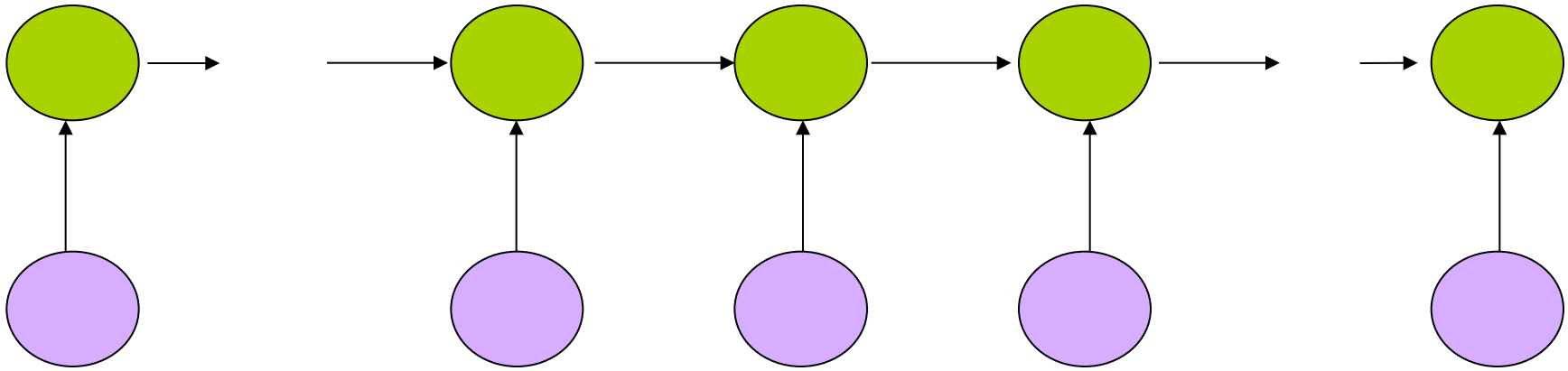
Hidden Markov Model (HMM)

Assumption 2: Discriminative Independent Model



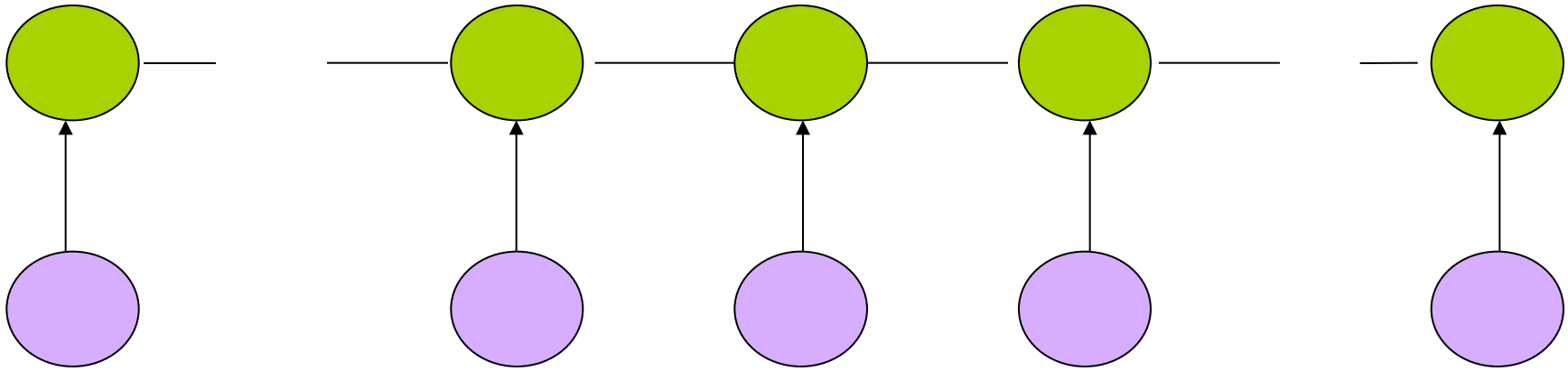
Classifier:
Maximum Entropy Model (ME)
Support Vector Machines (SVM)

Assumption 3: Discriminative Locally Dependent Model



Maximum Entropy Markov Model (MEMM)

Assumption 4: Discriminative Globally Dependent Model



Conditional Random Field (CRF)

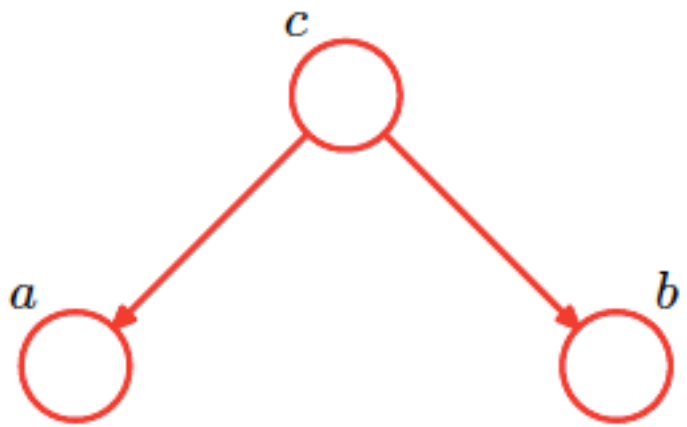
Outline

- Introduction
- Directed Graphical Models
- Undirected Graphical Models
- Factor Graph

Conditional Independence

- Random variables a, b are independent if p(a,b)=p(a)p(b) or p(a|b)=p(a).
- Random variables a, b are conditionally independent given c if p(a,b|c)=p(a|c)p(b|c) or p(a|b,c)=p(a|c).
- Conditional independence in a Bayesian Network is precisely specified by d-separation.

Example 1: Tail-to-Tail



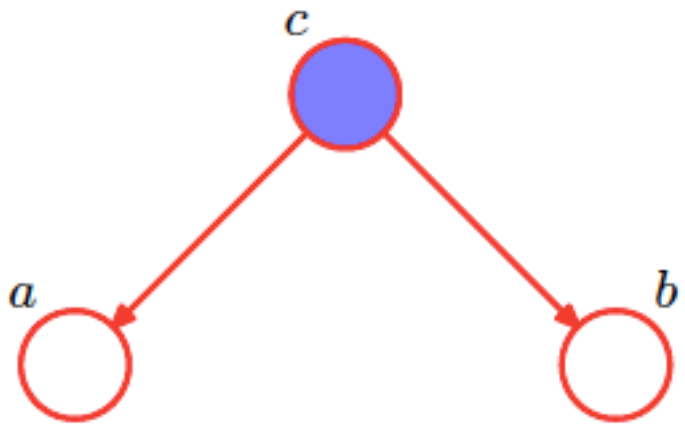
$p(a,b,c)=p(a|c)p(b|c)p(c)$

$p(a,b)=\sum_c p(a|c)p(b|c)p(c)$

$a \not\perp b|\varnothing$

- Random variable a and b are in general dependent.

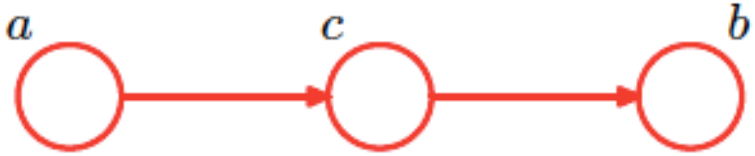
Example 1: Tail-to-Tail



$$p(a,b|c)=p(a,b,c)/p(c)$$
$$=p(a|c)p(b|c)$$
$$a \perp\!\!\!\perp b|c$$

- Random variable a and b are conditionally independent given c.
- An observed c is a tail-to-tail node, blocks the undirected path a-b

Example 2: Head-to-Tail



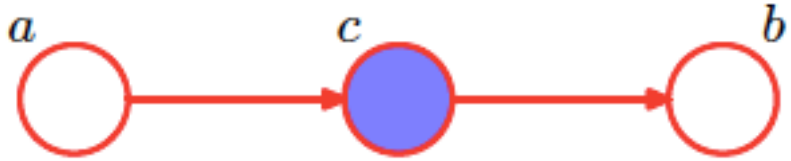
$p(a,b,c)=p(a)p(c|a)p(b|c)$

$p(a,b)=p(a)\sum_c p(c|a)p(b|c)=p(a)p(b|a)$

$a \not\perp b \mid \varnothing$

- Random variables a and b are in general dependent.

Example 2: Head-to-Tail

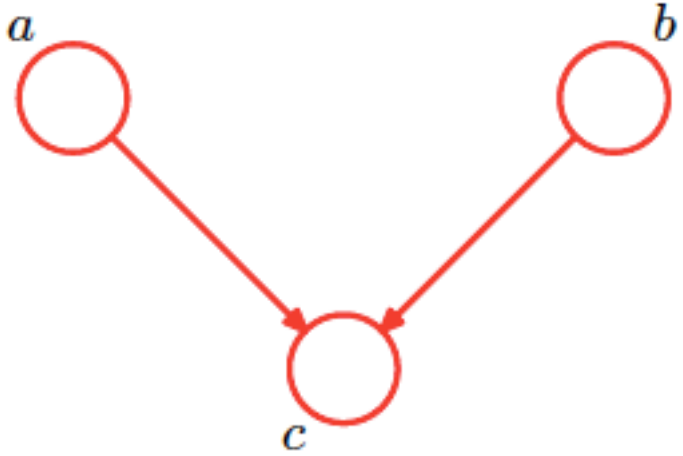


$$p(a,b|c)=p(a,b,c)/p(c)$$
$$=p(a)p(c|a)p(b|c)/p(c)$$
$$=p(a|c)p(b|c)$$

$$a \perp\!\!\!\perp b|c$$

- Random variables a and b are conditionally independent given c.
- An observed c is a head-to-tail node, blocks the path a-b

Example3: Head-to-Head



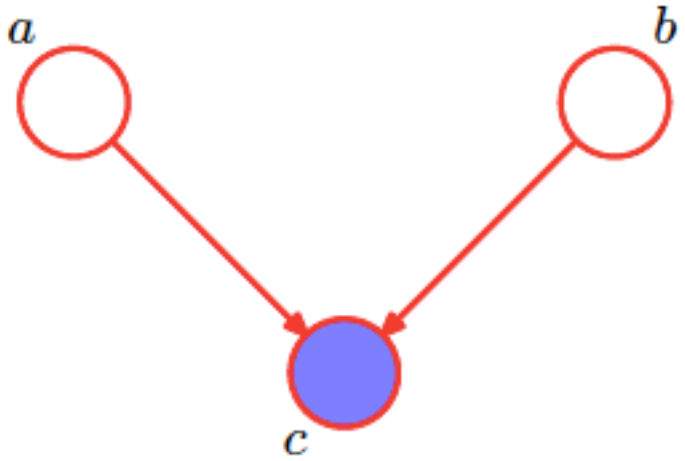
$p(a,b,c)=p(a)p(b)p(c|a,b)$

$p(a,b)=p(a)p(b)$

$a \perp\!\!\!\perp b|\varnothing$

- Random variables a and b are in general independent.

Example3: Head-to-Head



$$p(a,b|c) = p(a,b,c)/p(c)$$
$$= p(a)p(b)p(c|a,b)/p(c)$$
$$a \not\perp b|c$$

- Random variables a and b are dependent given c (observed nodes are shaded)
- An observed c is a head-to-head node, unblocks the undirected path a-b

Summary

- A tail-to-tail node or a head-to-tail node leaves a path unblocked unless it is observed in which case it blocks the path.
- A head-to-head node blocks a path if it is unobserved, but once the node, and/or at least one of its descendants, is observed the path becomes unblocked.

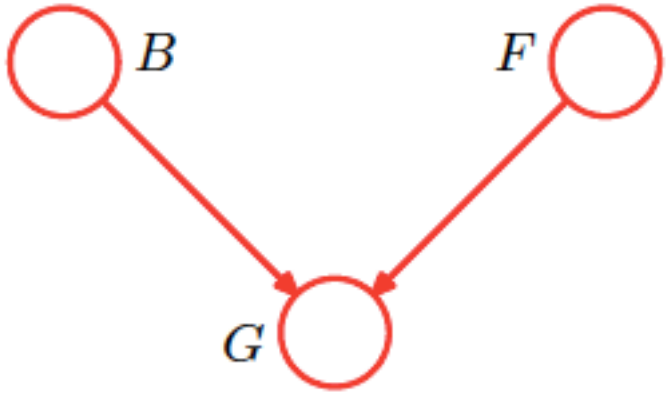
Explaining Away

p(G=1|B=1,F=1) =0.8

p(G=1|B=1,F=0) =0.2

p(G=1|B=0,F=1) =0.2

p(G=1|B=0,F=0) =0.1



p(B=1) =0.9

p(F=1) =0.9

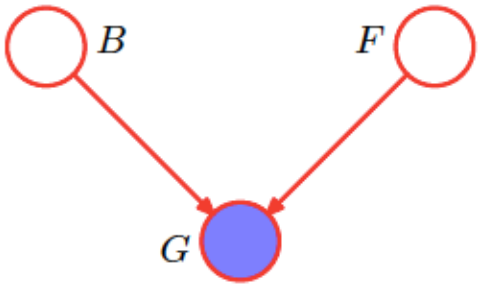
B = Battery (0=flat, 1=fully charged)

F = Fuel Tank (0=empty, 1=full)

G=Fuel Gauge Reading (0=empty, 1=full)

A number of the following pages are from Bishop's slides

Explaining Away



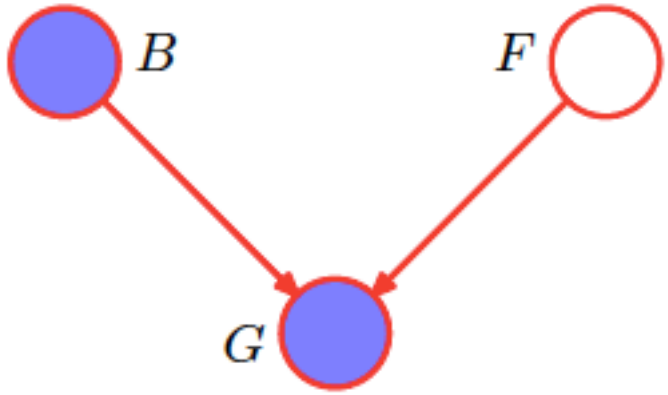
$$p(G=0) = \sum_{B\in\{0,1\}} \sum_{F\in\{0,1\}} p(G=0|B,F)p(B)p(F) = 0.315$$

$$p(G=0|F=0) = \sum_{B\in\{0,1\}} p(G=0|B,F=0)p(B) = 0.81$$

$$p(F=0|G=0) = \frac{p(G=0|F=0)p(F=0)}{p(G=0)} \simeq 0.257$$

- Probability of an empty tank increased by observing G = 0 (p(F=0) = 0.1).

Explaining Away



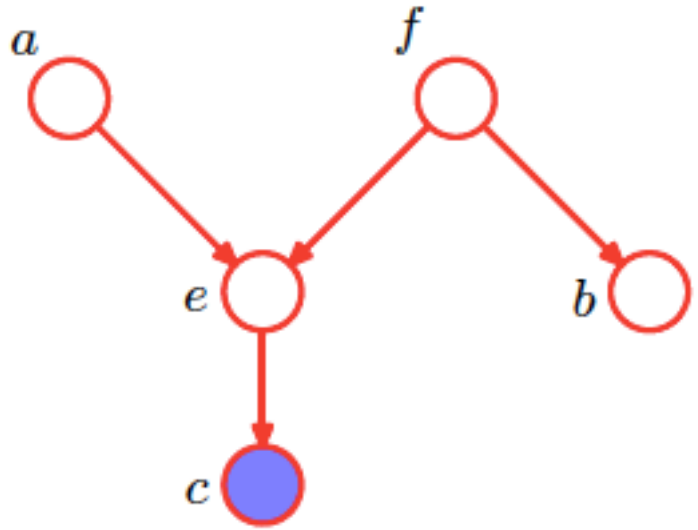
$$p(F=0|G=0,B=0) = \frac{p(G=0|B=0,F=0)p(F=0)}{\sum_{F\in\{0,1\}} p(G=0|B=0,F)p(F)} \simeq 0.111$$

- Probability of an empty tank reduced by observing B = 0.

D-Separation

- A path from A to B is blocked by C if it includes a node x such that either
  - The path is head-to-tail or tail-to-tail at x and $x \in C$, or
  - The path is head-to-head at x, and neither x nor any of its descendants is in C.
- If all paths form A to B are blocked, A is said to be d-separated from B by C, and the joint distribution over all the variables in the graph will satisfy $A \perp\!\!\!\perp B|C$

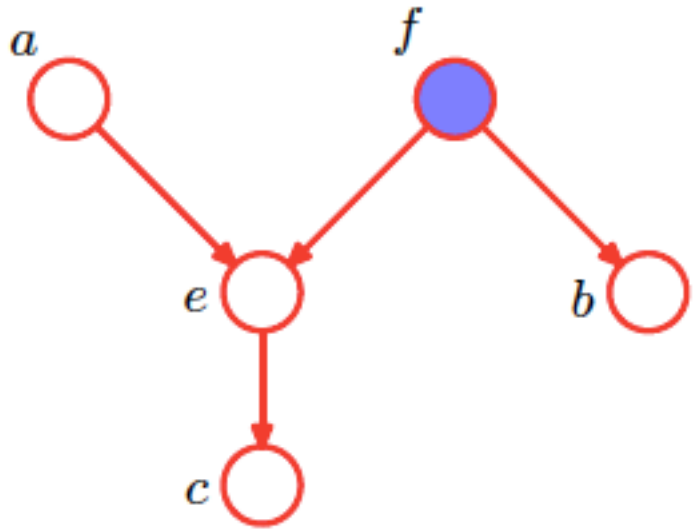
D-Separation Example 1



$a \not\perp b|c$

- The path a-b is not blocked by node f because it is a tail-to-tail node for this path and is not observed.
- The path a-b is not blocked by node e because although it is a head-to-head node, it has a descendant c because is in the conditioning set.

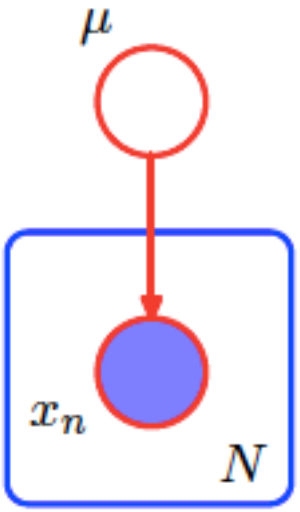
D-Separation Example 2



$a \perp\!\!\!\perp b \mid c$

- The path a-b is blocked by node f because this is a tail-to-tail node that is observed.
- The path a-b is also blocked by node e because e is a head-to-head node and neither it nor its descendent are in the conditioning set.

D-separation: Infer μ from D={x_1,…,x_N}



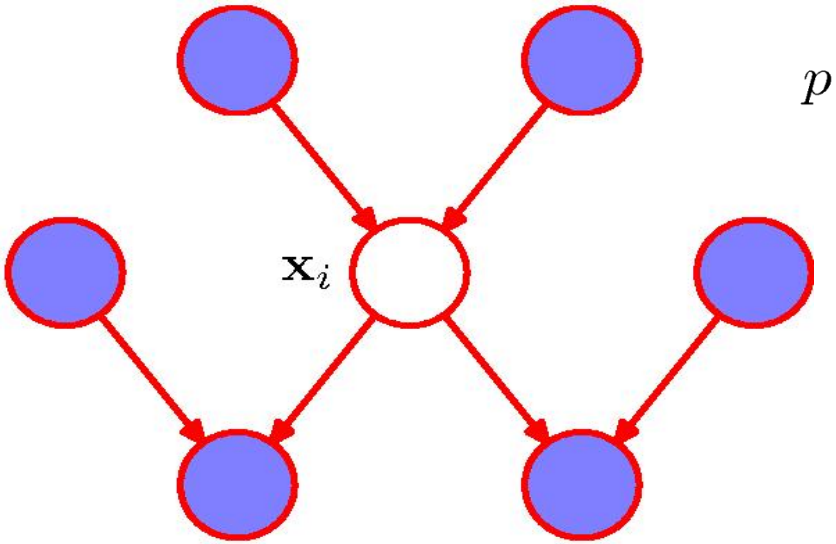
- Suppose we consider the joint distribution of the observations condition on μ.
- Then there is a unique path from any x_i to any other x_j and the path is tail-to-tail with observed node μ.
- Thus every such path is blocked and the observations D are independent given μ, so that

$$p(\mathcal{D}|\mu) = \prod_{n=1}^{N} p(x_n|\mu)$$

However, if we integrate over μ, the observations are in general no longer independent

$$p(\mathcal{D}) = \int_{-\infty}^{\infty} p(\mathcal{D}|\mu)p(\mu)\,\mathrm{d}\mu \neq \prod_{n=1}^{N} p(x_n)$$

The Markov Blanket



$$p(\mathbf{x}_i|\mathbf{x}_{\{j\neq i\}}) = \frac{p(\mathbf{x}_1,\ldots,\mathbf{x}_M)}{\int p(\mathbf{x}_1,\ldots,\mathbf{x}_M)\,\mathrm{d}\mathbf{x}_i}$$

$$= \frac{\prod_k p(\mathbf{x}_k|\mathrm{pa}_k)}{\int \prod_k p(\mathbf{x}_k|\mathrm{pa}_k)\,\mathrm{d}\mathbf{x}_i}$$

- Factors independent of x_i cancel between numerator and denominator.

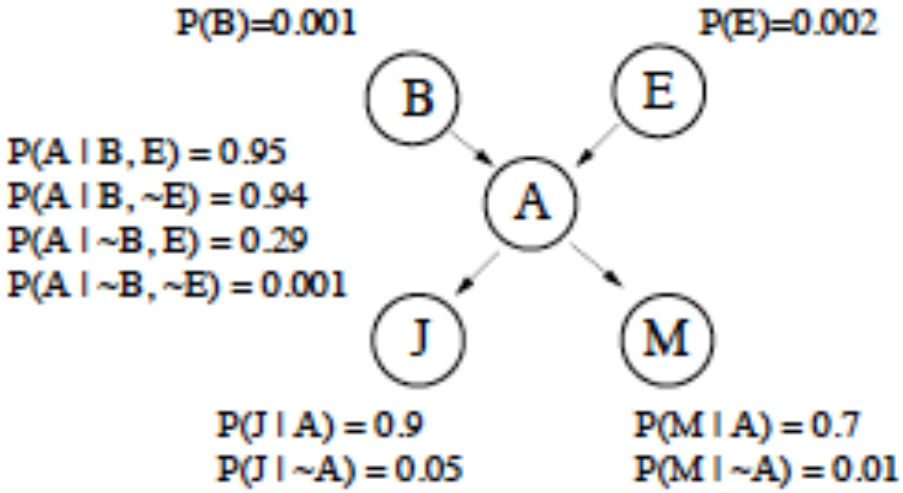
Directed Graphic Models

- Also called Bayesian networks.
- A Bayesian network on the DAG is a family of distributions satisfying

$$\{p \mid p(X) = \prod_i p(x_i \mid Pa(x_i))\}$$

  - Where $Pa(x_i)$ is the set of parents of x_i
  - $p(x_i|Pa(x_i))$ is the conditional probability distribution (CPD) at x_i.

Example 1: Alarm



p(B, ~E, A, J, ~M)

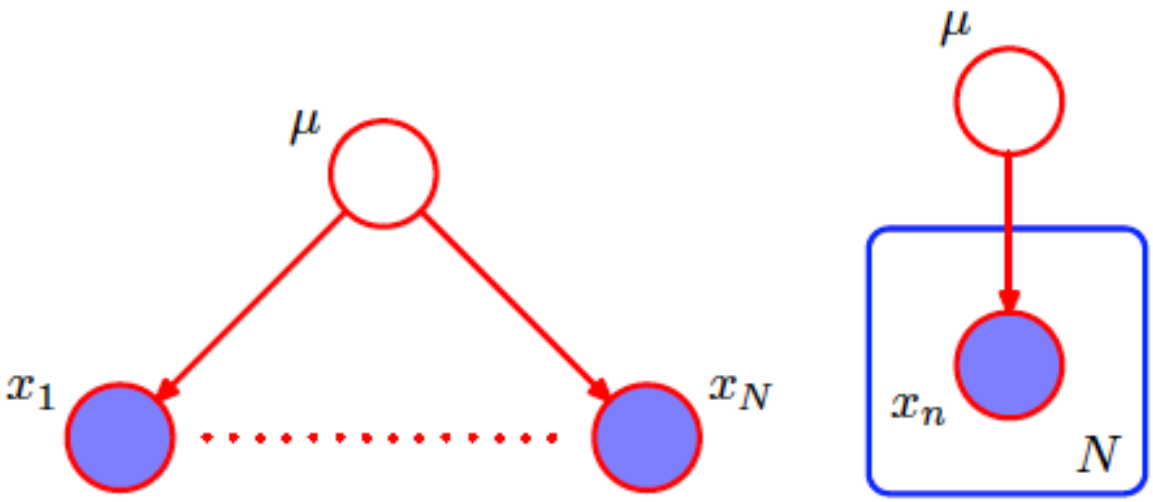
= p(B) p(~E) p(A|B, ~E) p(J|A) p(~M|A)

= 0.001×(1-0.002) ×0.94×0.9×(1-0.7)

≃ 0.000253

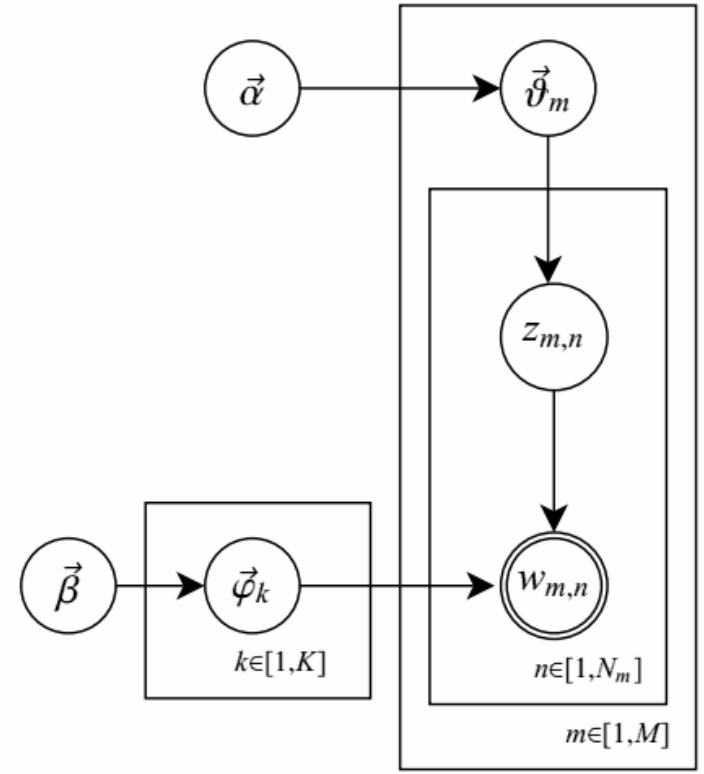
From Xiaojin Zhu, KDD'12 Tutorial

Example 2: Naïve Bayes



$$p(\mathcal{D}|\mu) = \prod_{n=1}^{N} p(x_n|\mu)$$

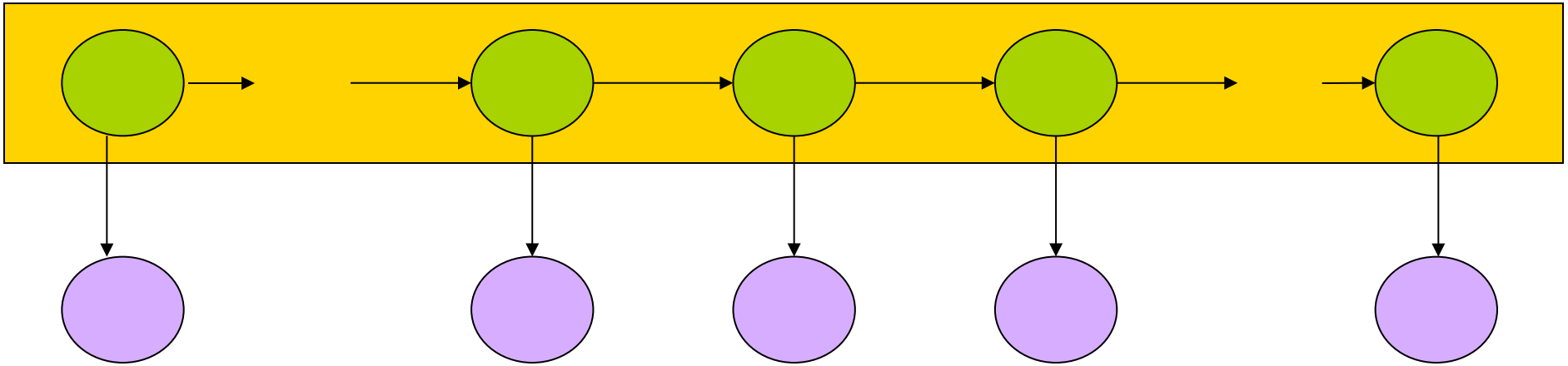
Example 3: Latent Dirichlet Allocation



$$P(\mathcal{W}, \mathcal{Z}, \Theta, \Phi | \alpha, \beta) = \prod_{k=1}^{K} P(\vec{\phi}_k | \vec{\beta}) \prod_{m=1}^{M} \left(P(\vec{\theta}_m | \vec{\alpha}) \prod_{n=1}^{N_m} P(w_{m,n} | \vec{\phi}_{z_{m,n}}) P(z_{m,n} | \vec{\theta}_m) \right)$$

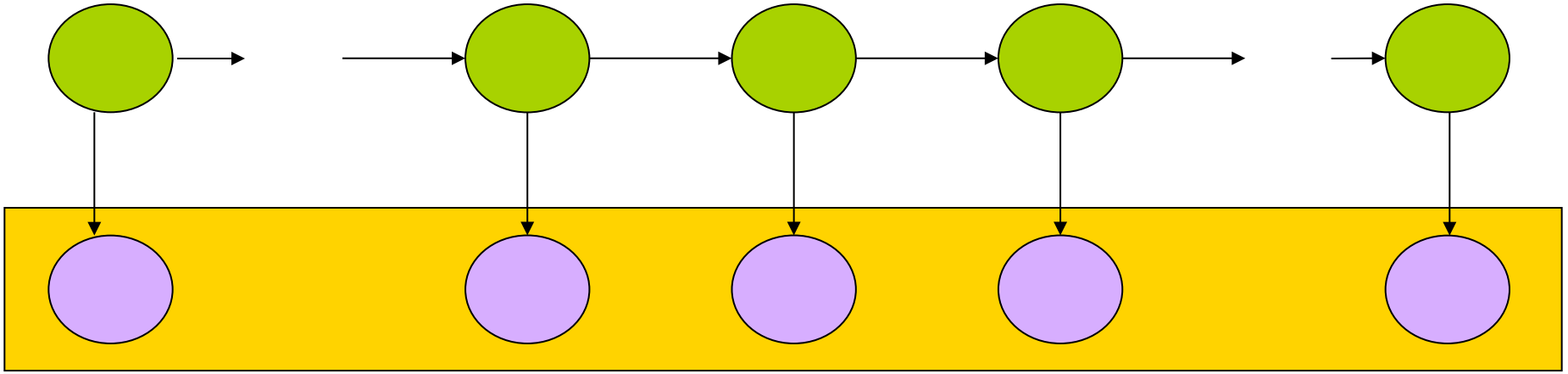
Example 4: HMM

What is HMM?



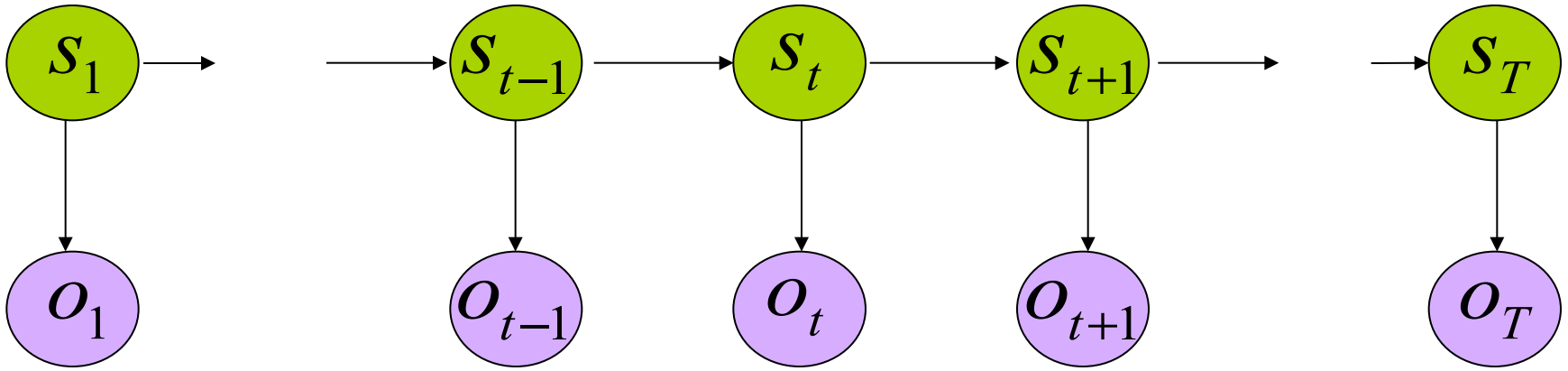
- Green nodes are '***hidden' states***
- State depends only on previous state

What is HMM?



- Purple nodes are ***observations***
- Each state generates an observation

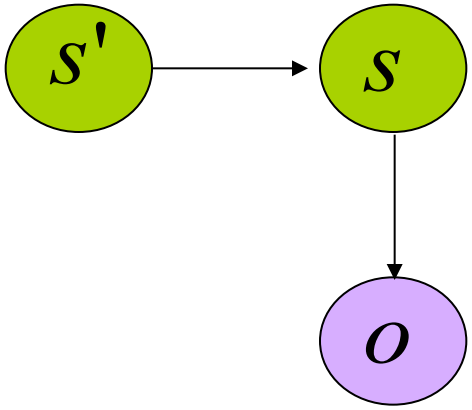
HMM Formalism



- s : {1,2,…,N} are values of hidden states
- o : {1,2,…,M} are values of observations

$$P(O,S) = P(s_1)P(o_1 \mid s_1)\prod_{t=2}^{T} P(s_t \mid s_{t-1})P(o_t \mid s_t)$$

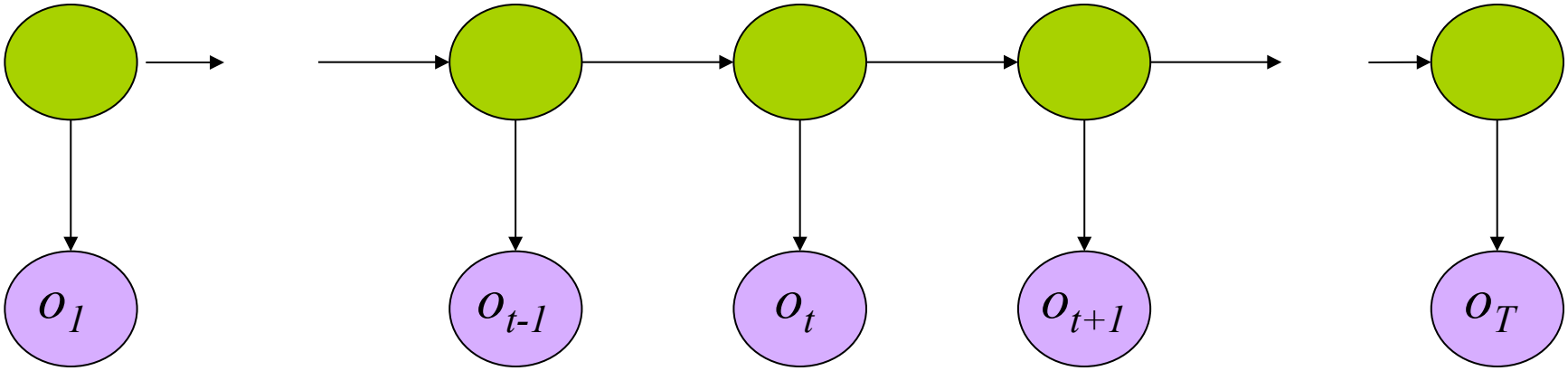
HMM Formalism



$$P(s \mid s')$$

$$P(o \mid s)$$

Tagging



- Viterbi algorithm
  - given observation sequence, compute most likely having generated state sequence

$$\arg\max_S P(S \mid O) = \arg\max_S P(S, O)$$

Summary of HMM

Model

•Baum,1966; Manning, 1999

Applications

•POS tagging (Kupiec, 1992)
•Shallow parsing (Molina, 2002; Ferran Pla, 2000; Zhou, 2000)
•Speech recognition (Rabiner, 1989; Rabiner 1993)
•Gene sequence analysis (Durbin, 1998)
•…

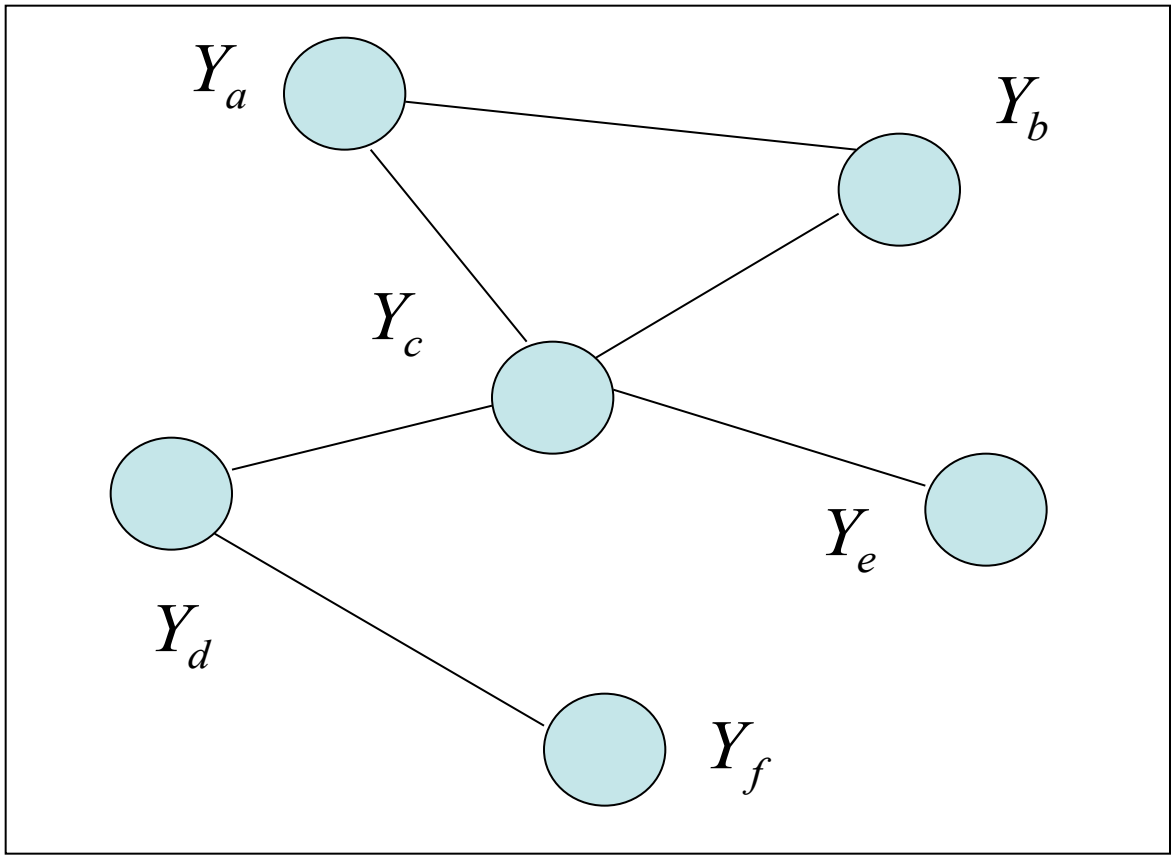
Limitation

•Joint probability distribution $p(x, s)$.
•Cannot represent overlapping features and long range dependences long range dependences between observed elements.

Outline

- Introduction
- Directed Graphical Models
- Undirected Graphical Models
- Factor Graph

Markov Random Field



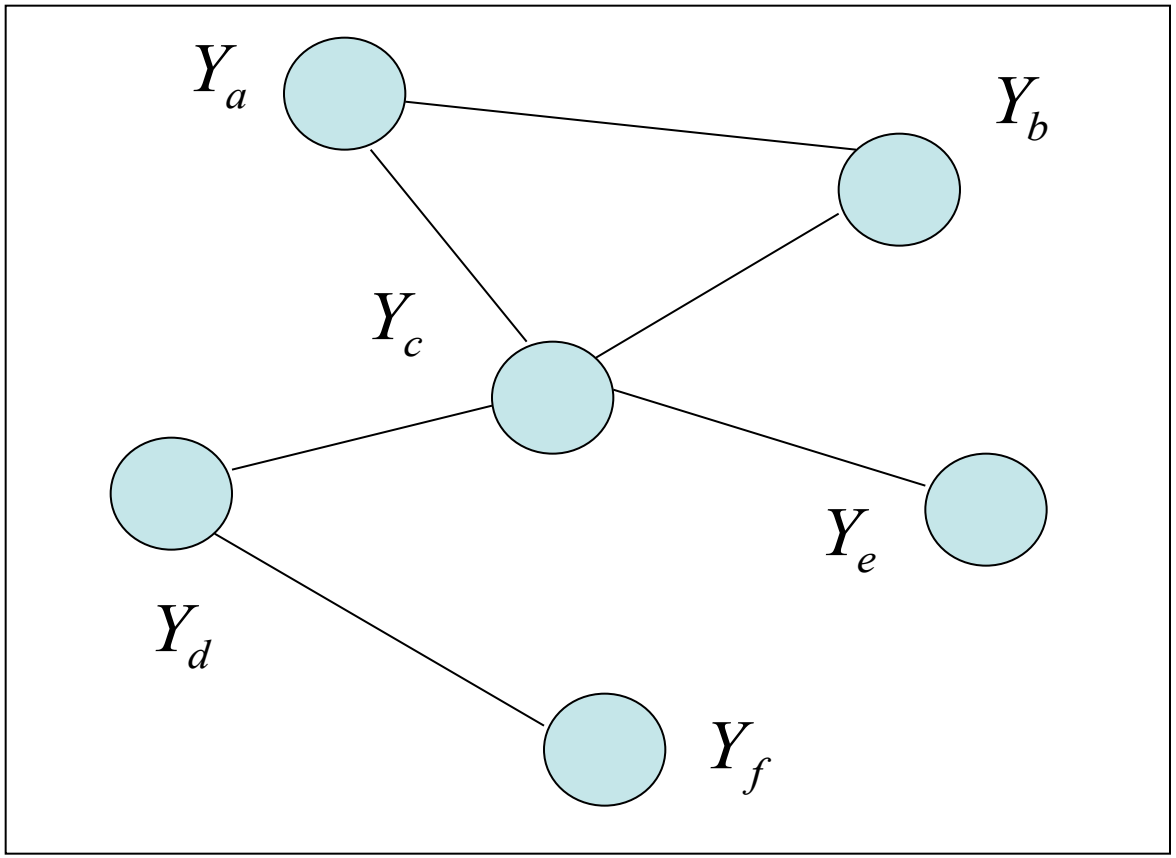
Given an undirected graph $G=(V, E)$ such that $Y=\{Y_v | v \in V\}$, if

The probability of Y_v given X and those random variables corresponding to nodes neighboring v in G. Then (X, Y) is a conditional random field.

Undirected graphical model globally conditioned on X

$$p(Y_v \mid X, Y_u, u \neq v, \{u,v\} \in V) \Leftrightarrow p(Y_v \mid X, Y_u, (u,v) \in E)$$

Markov Random Field



By the Hammersley-Clifford theorem, the probability of a label can be expressed as a Gibbs distribution, so that

$$p(Y) = \frac{1}{Z}\prod_{C}\varphi(Y_C)$$

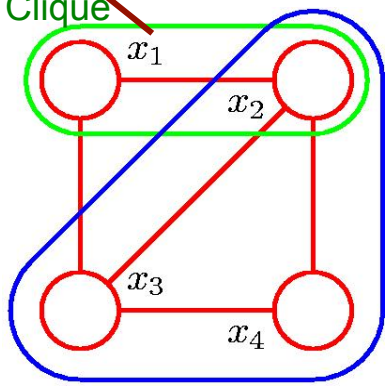
$$Z = \sum_{Y}\prod_{C}\varphi(Y_C)$$

Energy function

$$\varphi(Y_C) = \exp\{-E(Y_C)\}$$

Clique

Maximal Clique



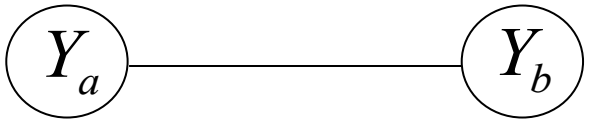
Undirected Graphic Model

- Also called Markov Random Fields
- An undirected graphical model is a family of distributions satisfying

$$\left\{ p \mid p(Y) = \frac{1}{Z} \prod_{C} \varphi(Y_C) \right\}$$

  – where z is the partition function by integrating all the possible values of *X*.

A Toy Example—MRF



$Y_a, Y_b \in \{-1, 1\}$

one clique $\varphi_C(Y_a, Y_b) = e^{aY_aY_b}$

$$p(Y_a, Y_b) = \frac{1}{Z} e^{aY_aY_b}, Z = (e^a + e^a + e^{-a} + e^{-a})$$

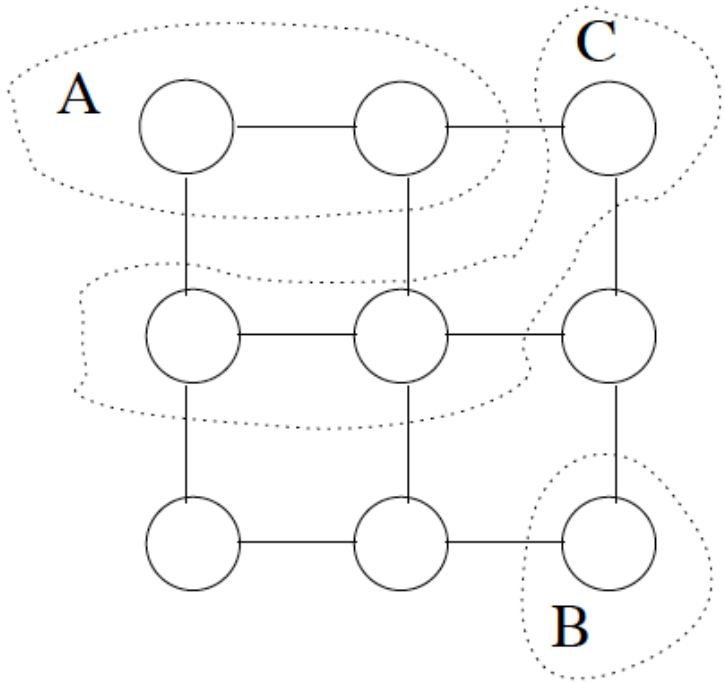
$$p(1,1) = p(-1,-1) = \frac{e^a}{2e^a + 2e^{-a}}$$

Note: when $a>0$, the model favors homogeneous chain

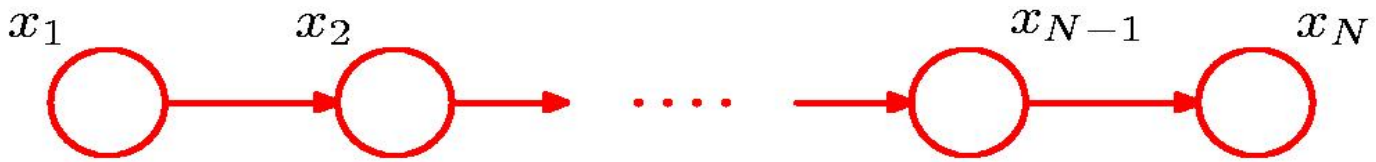
when $a<0$, the model favors inhomogeneous chain

Conditional Independence in Markov Random Fields

- Two group of variables A, B are conditional independent given another group C, if:
- A, B become disconnected by removing C

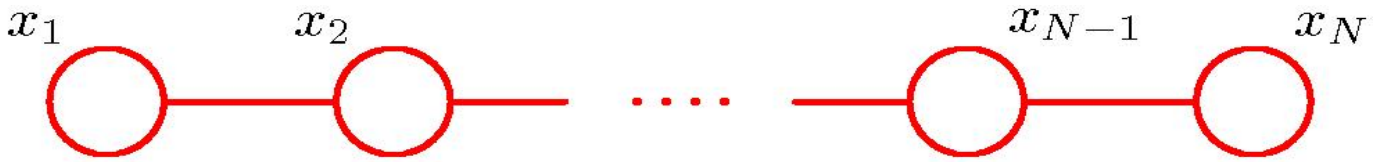


Relationship between the Undirected Graphs and the Directed Graphs



$$p(\mathbf{x}) = p(x_1)p(x_2|x_1)\ p(x_3|x_2) \cdots p(x_N|x_{N-1})$$

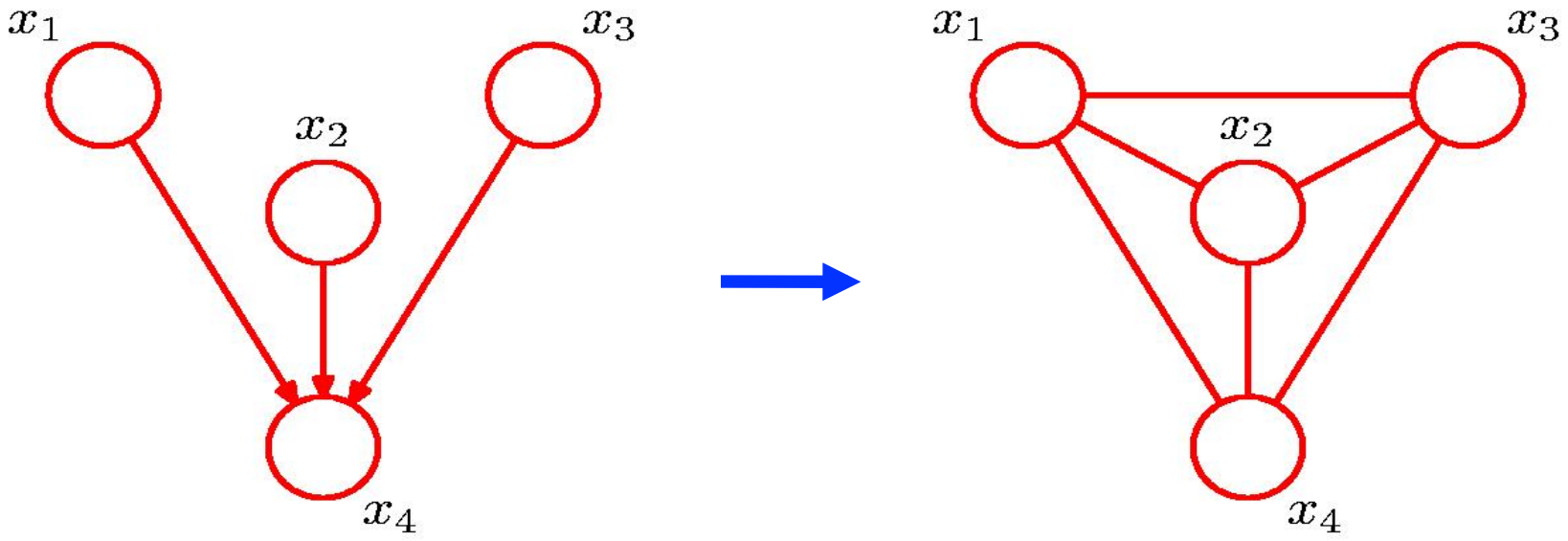
$$p(\mathbf{x}) = \frac{1}{Z}\ \psi_{1,2}(x_1, x_2)\ \psi_{2,3}(x_2, x_3) \cdots \psi_{N-1,N}(x_{N-1}, x_N)$$



A number of the following pages are from Bishop's slides

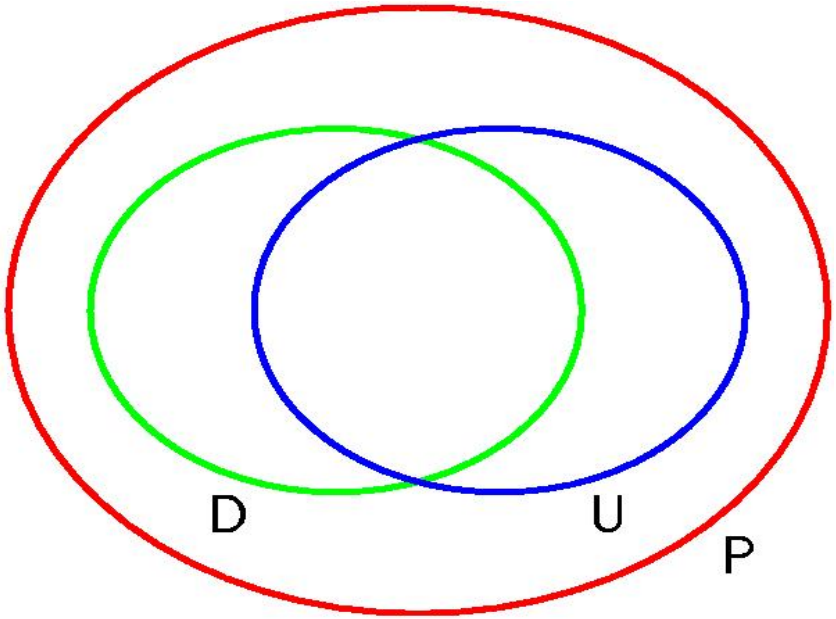
Converting Directed to Undirected Graphs

Additional links

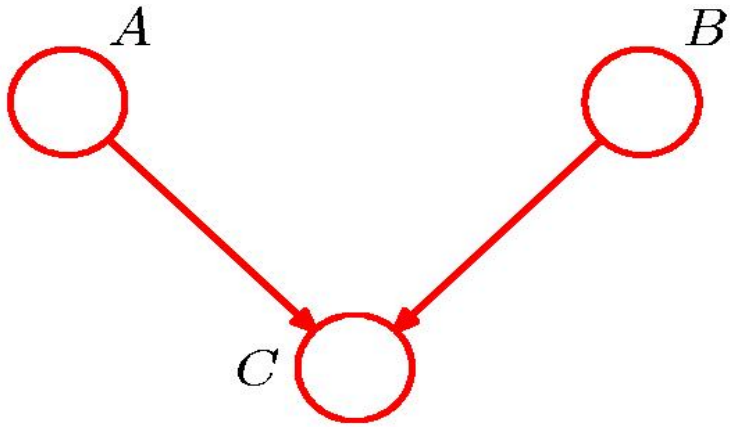


$$
\begin{aligned}
p(\mathbf{x}) &= p(x_1)p(x_2)p(x_3)p(x_4|x_1,x_2,x_3) \\
&= \frac{1}{Z}\psi_A(x_1,x_2,x_3)\psi_B(x_2,x_3,x_4)\psi_C(x_1,x_2,x_4)
\end{aligned}
$$

Directed vs. Undirected Graphs (1)

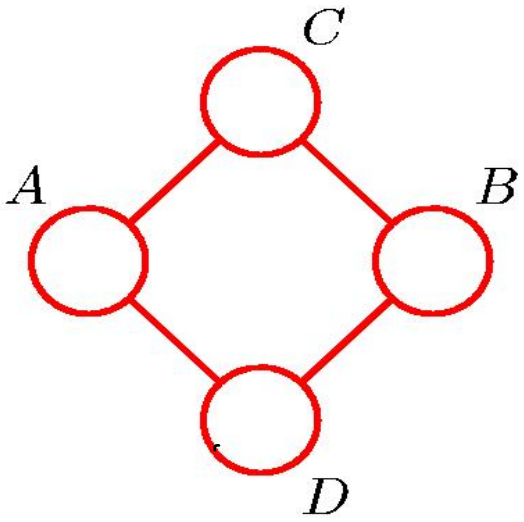


Directed vs. Undirected Graphs (2)



$$A \perp\!\!\!\perp B \mid \emptyset$$

$$A \not\perp\!\!\!\perp B \mid C$$



$$A \not\perp\!\!\!\perp B \mid \emptyset$$

$$A \perp\!\!\!\perp B \mid C \cup D$$

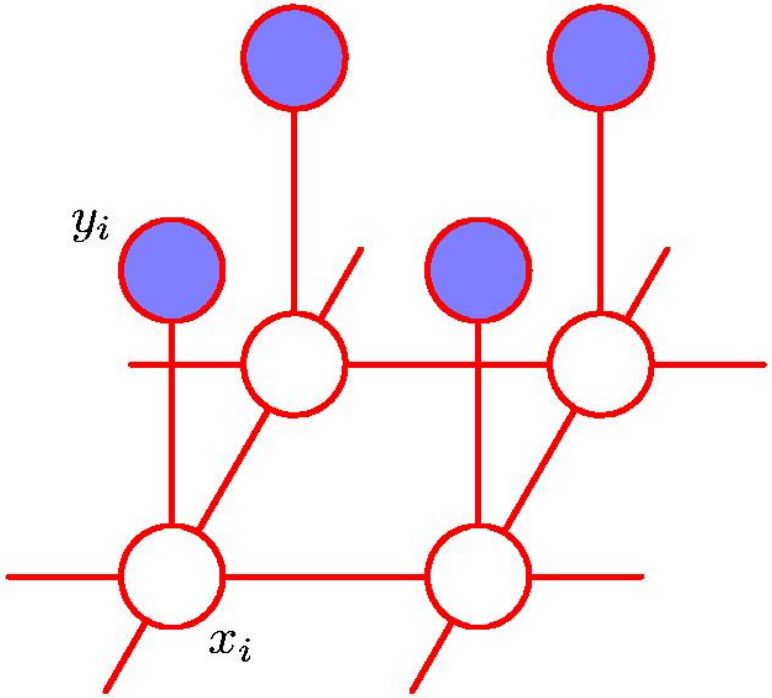
$$C \perp\!\!\!\perp D \mid A \cup B$$

Example 2: Log Linear Models

- Real-valued feature function $f_1(X),\ldots, f_k(X)$
- Real-valued weights $w_1,\ldots, w_k$

$$p(X) = \frac{1}{Z} \exp\left(-\sum_{i=1}^{k} w_i f_i(X)\right)$$

Example 3: The Ising Model



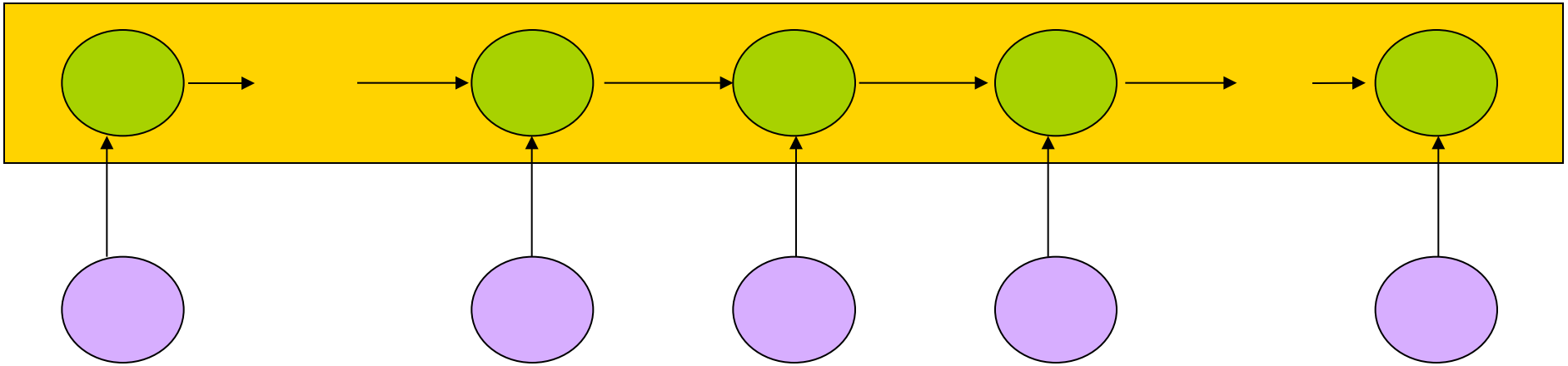
$$E(\mathbf{x}, \mathbf{y}) = h \sum_i x_i - \beta \sum_{\{i,j\}} x_i x_j - \eta \sum_i x_i y_i$$

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \exp\{-E(\mathbf{x}, \mathbf{y})\}$$

$x \in \{0,1\}$

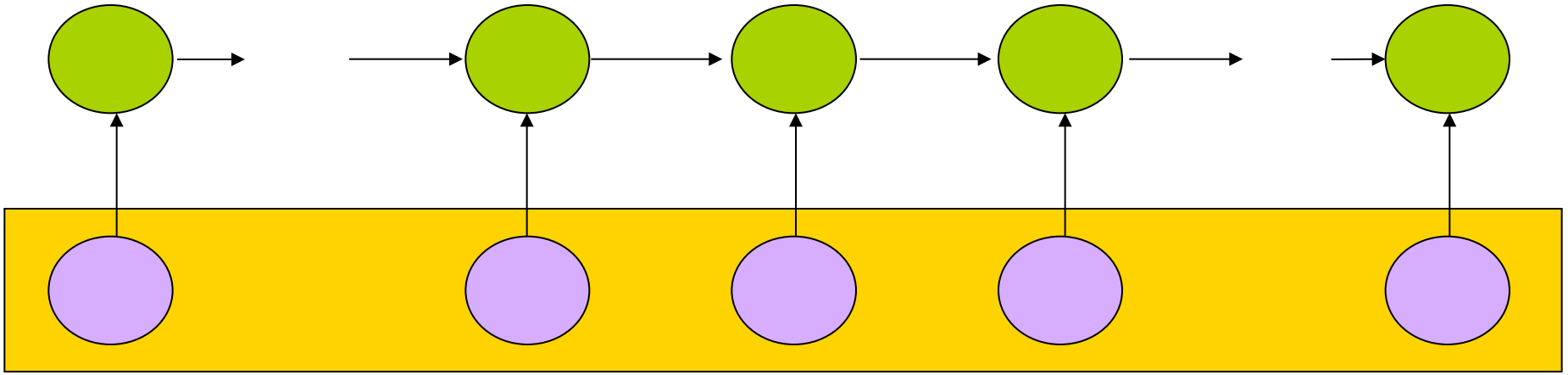
Example 4: MEMM

What is MEMM?



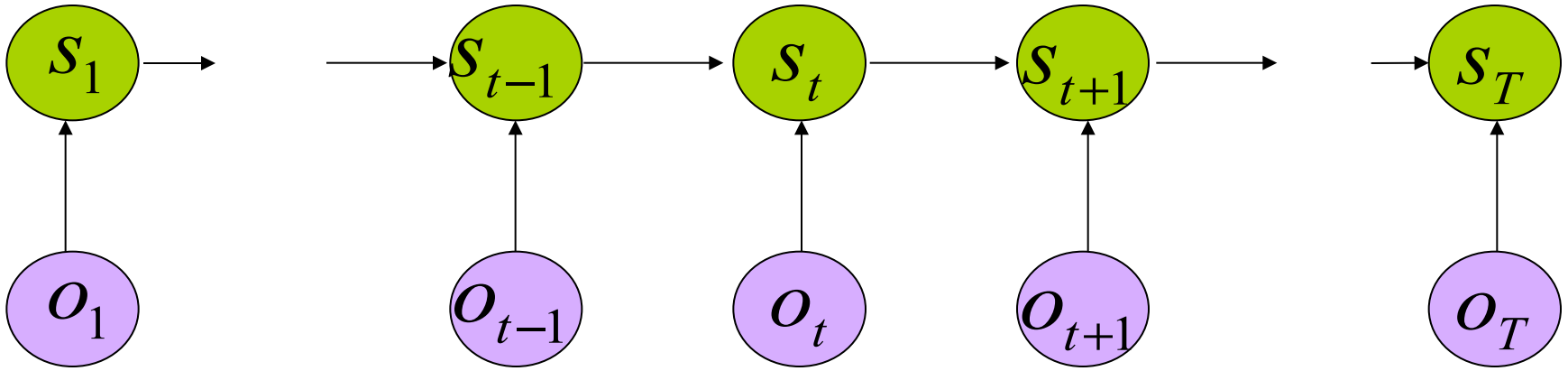
- Green nodes are ***states***
- State depends only on previous state

What is MEMM?



- Purple nodes are ***observations***
- Observations (features of observations) determine states

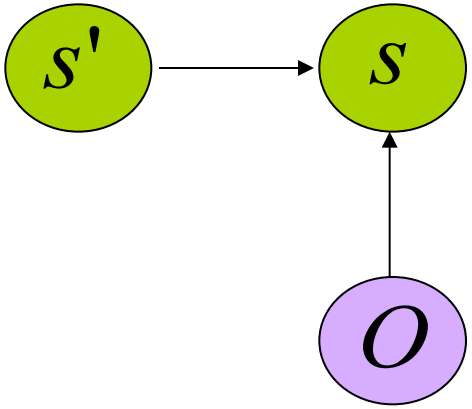
MEMM Formalism



- s : {1,2,…,N } are values for states
- o: {1,2,…,M } are values for observations

$$P(S \mid O) = P(s_1 \mid O) \prod_{t=2}^{T} P(s_t \mid s_{t-1}, O)$$

MEMM Formalism



$$s \leftarrow y, s' \leftarrow y', O \leftarrow x$$

$$P(s \mid s', O) = P(y \mid y', x)$$

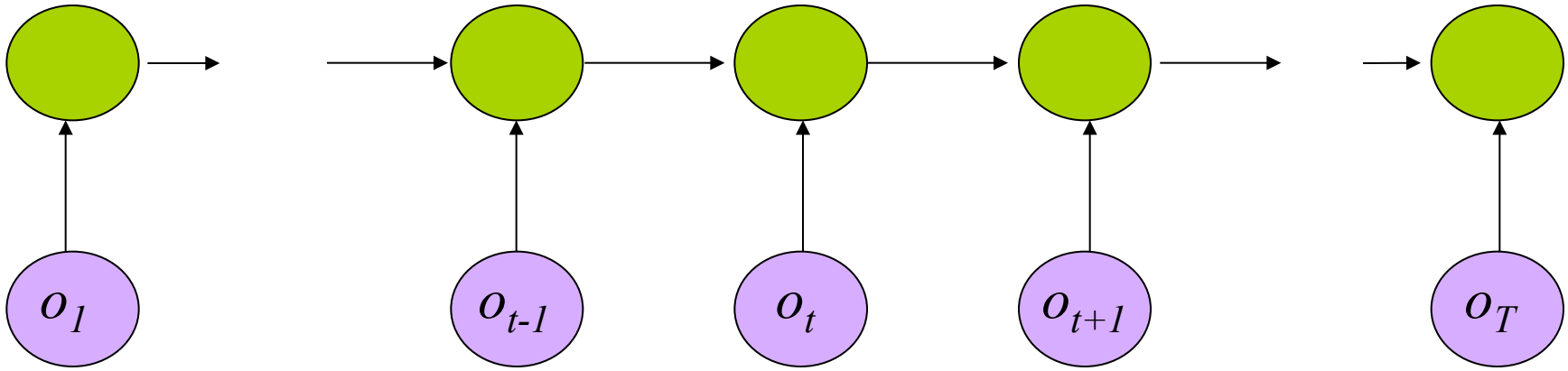
$$P(y \mid y', x) = \frac{\exp(\sum_k \lambda_k f_k(x, y', y))}{Z(y', x)}$$

$$Z(y', x) = \sum_y \exp(\sum_k \lambda_k f_k(x, y', y))$$

Inference in MEMM

- Tagging: given observation sequence, find most likely corresponding state sequence
- Learning: given observation sequence and corresponding state sequence, find model that best explains the matching

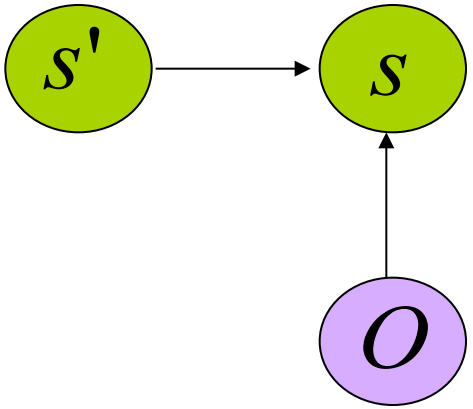
Tagging



- Viterbi algorithm

$$\arg\max_S P(S \mid O) = \arg\max_S P(s_1 \mid O) \prod_{t=1}^{T} P(s_t \mid s_{t-1}, O)$$

Learning



$$(x_1, y_1', y_1), (x_2, y'_2, y_2), \cdots, (x_n, y'_n, y_n)$$

$$P(y \mid y', x) = \frac{\exp(\sum_k \lambda_k f_k(x, y', y))}{Z(x, y')}$$

$$Z(x, y') = \sum_y \exp(\sum_k \lambda_k f_k(x, y', y))$$

$$\arg\max \sum_{i=1}^{n} \log P(y_i \mid y'_i, x_i)$$

Learning Algorithm: IIS

Algorithm 1 : Improved Iterative Scaling

Input: Feature functions $f_1, f_2, \ldots f_n$; empirical distribution $\tilde{p}(x, y)$
Output: Optimal parameter values $\lambda^\star{}_i$; optimal model $p_{\lambda^\star}$

1. Start with $\lambda_i = 0$ for all $i \in \{1, 2, \ldots, n\}$
2. Do for each $i \in \{1, 2, \ldots, n\}$:
   a. Let $\Delta\lambda_i$ be the solution to

$$\sum_{x,y} \tilde{p}(x)p(y|x)f_i(x,y)\exp\Big(\Delta\lambda_i f^{\#}(x,y)\Big) = \tilde{p}(f_i) \tag{16}$$

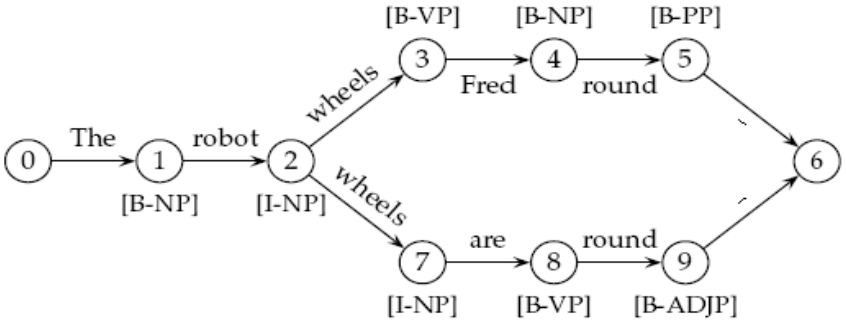
$$\text{where} \quad f^{\#}(x,y) \equiv \sum_{i=1}^{n} f_i(x,y) \tag{17}$$

   b. Update the value of λ_i according to: $\lambda_i \leftarrow \lambda_i + \Delta\lambda_i$
3. Go to step 2 if not all the λ_i have converged

Summary of MEMM

- Discriminative model
- Conditional assumption
- Accuracy is higher than MaxEnt, lower than CRF
- Problem: local model → label bias problem
- MEMM contains MaxEnt as special case

Label Bias Problem



The finite-state acceptor is designed to shallow parse the sentences (chunk/phrase parsing)
1) the robot wheels Fred round
2) the robot wheels are round

Decoding it by:
0123456
0127896

$$p(s \mid o) = p(s_1 \mid o_1) \prod_{t=2}^{T} p(s_t \mid s_{t-1}, o_t)$$

Assuming the probabilities of each of the transitions out of state 2 are approximately equal, the label bias problem means that the probability of each of these chunk sequences given an observation sequence *x* will also be roughly equal irrespective of the observation sequence *x*.

On the other hand, had one of the transitions out of state 2 occurred more frequently in the training data, the probability of that transition would always be greater. This situation would result in the sequence of chunk tags associated with that path being preferred irrespective of the observation sentence.

Summary of MEMM

Model

•Berger, 1996; Ratnaparkhi 1997, 1998

Applications

•Segmentation (McCallum, 2000)

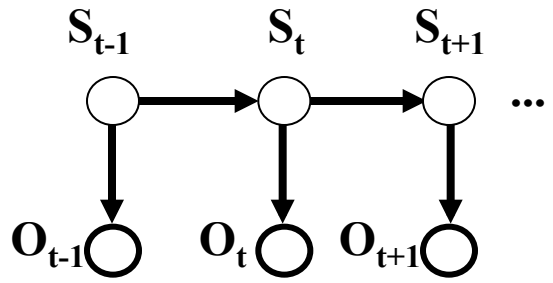
•…

Limitation

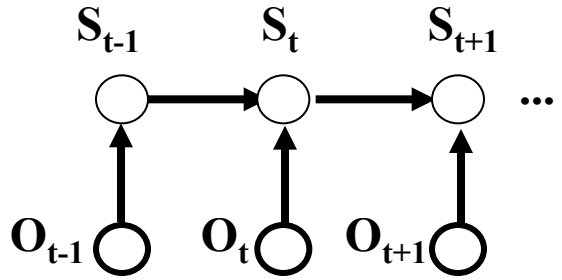
•Label bias problem (HMM do not suffer from the label bias problem)

Conditional Markov Models (CMMs) aka MEMMs aka Maxent Taggers *vs* HMMS

$$p(s,o) = \prod_{t=1}^{T} p(s_t \mid s_{t-1}) p(o_t \mid s_{t-1})$$



$$p(s \mid o) = \prod_{t=1}^{T} p(s_t \mid s_{t-1}, o_{t-1})$$



Example 5: CRFs

MEMM to CRFs

$$p(y_1 \ldots y_N \mid x_1 \ldots x_N) = \prod_{n=1}^{N} p(y_n \mid y_{n-1}, x_n) = \prod_{n=1}^{N} \frac{\exp(\sum_{k=1}^{K} \lambda_k f_k(x_n, y_n, y_{n-1}))}{Z(x_n)}$$

$$= \frac{\exp(\sum_k \lambda_k F_k(\vec{x}, \vec{y}))}{\prod_n Z(x_n)}, \text{ where } F_k(\vec{x}, \vec{y}) = \sum_k f_k(x_n, y_n, y_{n-1})$$

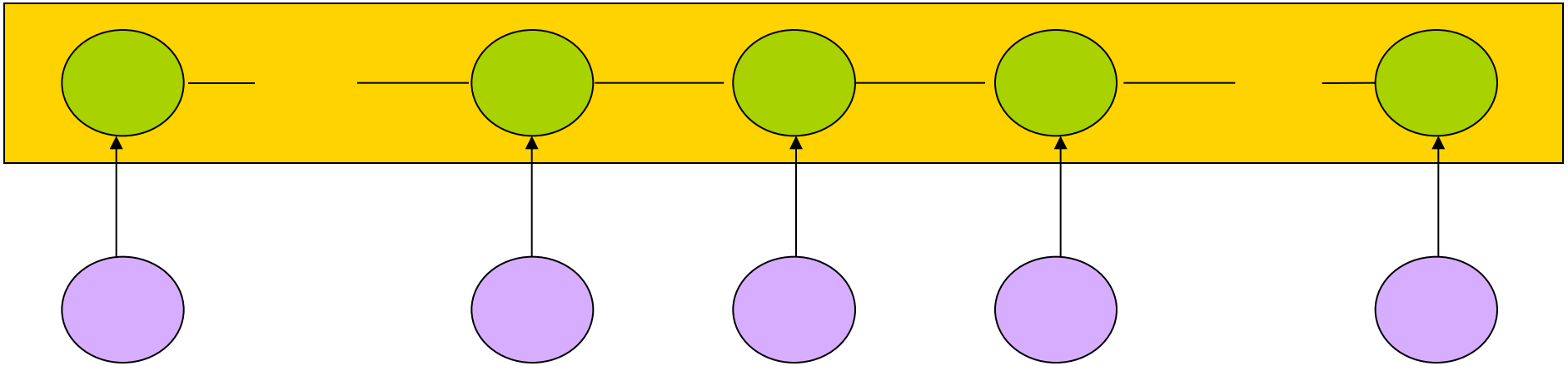
New model

$$\frac{\exp(\sum_k \lambda_k F_k(\vec{x}, \vec{y}))}{Z(\vec{x})}$$

n = 1…N: random variable index

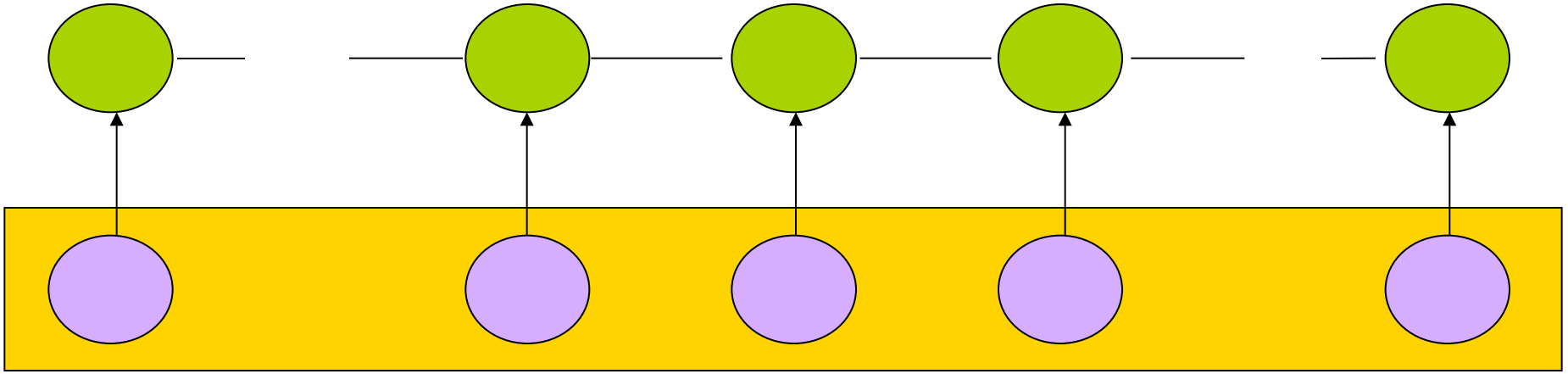
k = 1…K: feature index

What is CRF?



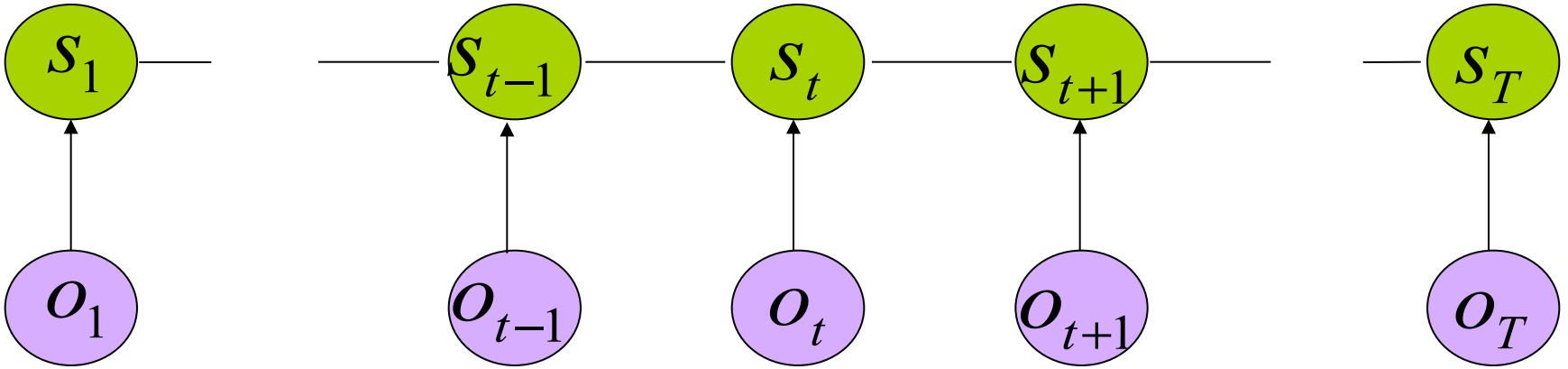
- Green nodes are ***states***
- State depends on neighboring states

What is CRF?



- Purple nodes are ***observations***
- Observations (features of observations) determine states

CRF Formalism



- *s*: {1,2,…,N} are values of states
- *o*: {1,2,…,M } are values of observations

$$P(S \mid O)$$

Back to the CRF

CRF is a Markov Random Fields.

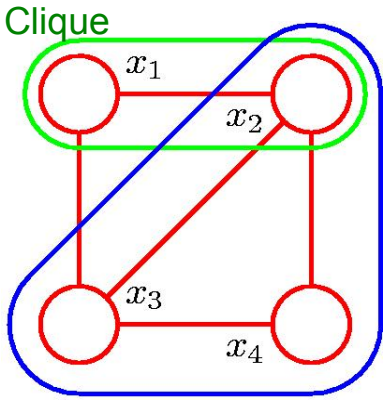
By the Hammersley-Clifford theorem, the probability of a label can be expressed as a Gibbs distribution, so that

$$p(y \mid x, \lambda) = \frac{1}{Z} \exp\left(\sum_k \lambda_k F_k(y, x)\right)$$

$$F_k(y, x) = \sum_{n=1}^{N} f_k(y \mid c_n, x)$$

clique

What is clique?



By only taking consideration of the one node and two nodes cliques, we have

$$p(y \mid x, \lambda, \mu) = \frac{1}{Z} \exp\left(\sum_l \lambda_l \sum_n t_l(y \mid e_n, x) + \sum_k \mu_k \sum_n s_k(y \mid s_n, x)\right)$$

l : index of features associated to one-node clique

k: index of features associated to two-nodes clique

x, y: random vector

Definition (cont.)

Moreover, let us consider the problem in a first-order chain model, we have

$$p(y \mid x, \lambda, \mu) = \frac{1}{Z} \exp\left(\sum_{l} \lambda_l \sum_{n} t_l(y \mid e_n, x) + \sum_{k} \mu_k \sum_{n} s_k(y \mid s_n, x)\right)$$

For simplifying description, let $f_k(y_n, x)$ denote $t_l(y_{n-1}, y_n, x)$ and $s_k(y_n, x)$

$$p(y \mid x, \lambda) = \frac{1}{Z} \exp \sum_{k} \lambda_k F_k(y, x))$$

$$F_k(y, x) = \sum_{n=1}^{N} f_k(y \mid c_n, x)$$

In Labeling

- In labeling, the task is to find the label sequence that has the largest probability

$$\hat{y} = \arg\max_{y} p_{\lambda}(y \mid x) = \arg\max_{y}(\lambda \cdot F(y,x))$$

$$p(y \mid x,\lambda) = \frac{1}{Z}\exp(\sum_{k}\lambda_k F_k(y,x))$$

- Then the key is to estimate the parameter lambda
- Let us first review the optimization formalization

Optimization

- Defining a loss function, that should be convex for avoiding local optimization
- Defining constraints
- Finding a optimization method to solve the loss function
- A formal expression for optimization problem

$$\min_{\theta} f(x)$$

$$s.t. \qquad g_i(x) \geq 0, 0 \leq i \leq k$$

$$h_j(x) = 0, 0 \leq j \leq l$$

Loss Function

Empirical loss vs. structural loss

$$L = \sum_{m=1}^{M} \left| y^{(m)} - f(x^{(m)}, \lambda) \right|$$

$$\min L$$

$$L = \left|\left|\lambda\right|\right| + \sum_{m=1}^{M} \left| y^{(m)} - f(x^{(m)}, \lambda) \right|$$

$$\min L$$

Loss function: Log-likelihood

$$p(y \mid x, \lambda, \mu) = \frac{1}{Z} \exp\left(\sum_k \lambda_k F_k(y, x)\right)$$

$$L(\lambda) = \sum_m \left[-\log Z + \sum_k \lambda_k F_k(y^{(m)}, x^{(m)}) \right]$$

m: index of data sample

$$L = \sum_m \left[\lambda \cdot F(y^{(m)}, x^{(m)}) - \log Z(x^{(m)}) \right] - \frac{\left|\left|\lambda\right|\right|^2}{2\sigma^2} + const$$

IIS Algorithm

First-order numerical optimization

Using Iterative Scaling (GIS, IIS)

- Initialize each λ_k(=0 for example)
- Until convergence
  - Solve $\frac{\delta L}{\delta \lambda_i} = 0$ for each parameter λ_k
  - Update each parameter using λ_k<- $\lambda_k + \Delta\lambda_k$

Parameter estimation

Log-likelihood

$$L(\lambda)=\sum_m\left[-\log Z+\sum_k \lambda_k F_k(y^{(m)},x^{(m)})\right]$$

$$\frac{\delta L_\lambda}{\delta\lambda_k}=\sum_m\left[F_k(y^{(m)},x^{(m)})-\frac{(Z(x^{(m)}))'}{Z(x^{(m)})}\right]$$

$$Z(x^{(m)})=\sum_y \exp\lambda\cdot F(y,x^{(m)})$$

$$\frac{(Z(x^{(m)}))'}{Z(x^{(m)})}=\frac{\sum_y\left(\exp(\lambda\cdot F(y,x^{(m)}))*F_k(y,x^{(m)})\right)}{\sum_y \exp\lambda\cdot F(y,x^{(m)})}$$

Differentiating the log-likelihood with respect to parameter λ_k ⇨

$$=\sum_y\left(\frac{\exp(\lambda\cdot F(y,x^{(m)})}{\sum_y \exp\lambda\cdot F(y,x^{(m)})})*F_k(y,x^{(m)})\right)$$

$$=\sum_y\left(p(y\mid x^{(m)})*F_k(y,x^{(m)})\right)$$

$$=E_{p(Y|x^{(m)})}F_k(Y,x^{(m)})$$

$$\frac{\delta L}{\delta\lambda_k}=E_{p(Y,X)}[F_k(Y,X)]-\sum_m E_{p(Y|x^{(m)},\lambda)}[F_k(Y,x^{(m)})]$$ ⇦

By adding the model penalty, it can be rewritten as

$$\frac{\delta L}{\delta\lambda_k}=E_{p(Y,X)}[F_k(Y,X)]-\sum_m E_{p(Y|x^{(k)},\lambda)}[F_k(Y,x^{(m)})]-\frac{\lambda}{\sigma^2}$$

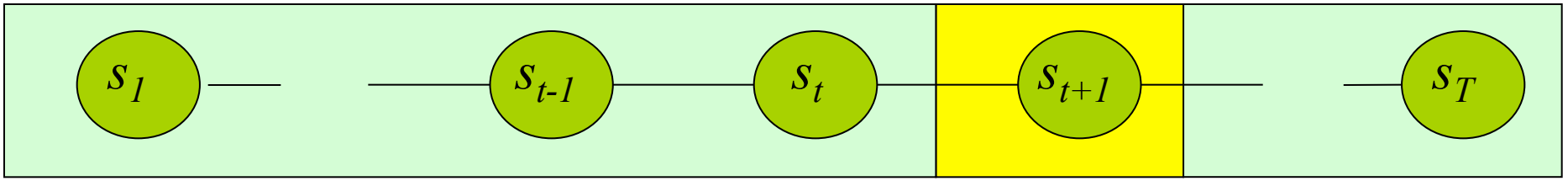
Solve the Optimization

$$L(\lambda) = \sum_m \left[-\log Z + \sum_k \lambda_k F_k(y^{(m)}, x^{(m)}) \right]$$

$$\frac{\delta L}{\delta \lambda_k} = E_{p(Y,X)}[F_k(Y,X)] - \sum_m E_{p(Y|x^{(m)},\lambda)}[F_k(Y,x^{(m)})]$$

- $E_{p(y,x)}F_k(y,x)$ can be calculated easily
- $E_{p(y|x)}F_k(y,x)$ can be calculated by making use of a forward-backward algorithm
- Z can be estimated in the forward-backward algorithm

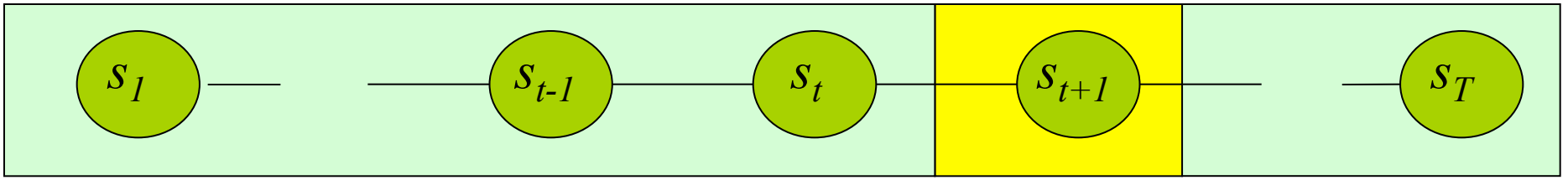
Inference



$$p(\mathbf{s}) = \frac{1}{Z} \prod_{i=1,\ldots,T-1} \varphi(s_i, s_{i+1})$$

$$p(s_{t+1}) = \sum_{s_1} \cdots \sum_{s_t} \sum_{s_{t+2}} \cdots \sum_{s_T} p(\mathbf{s})$$

Inference



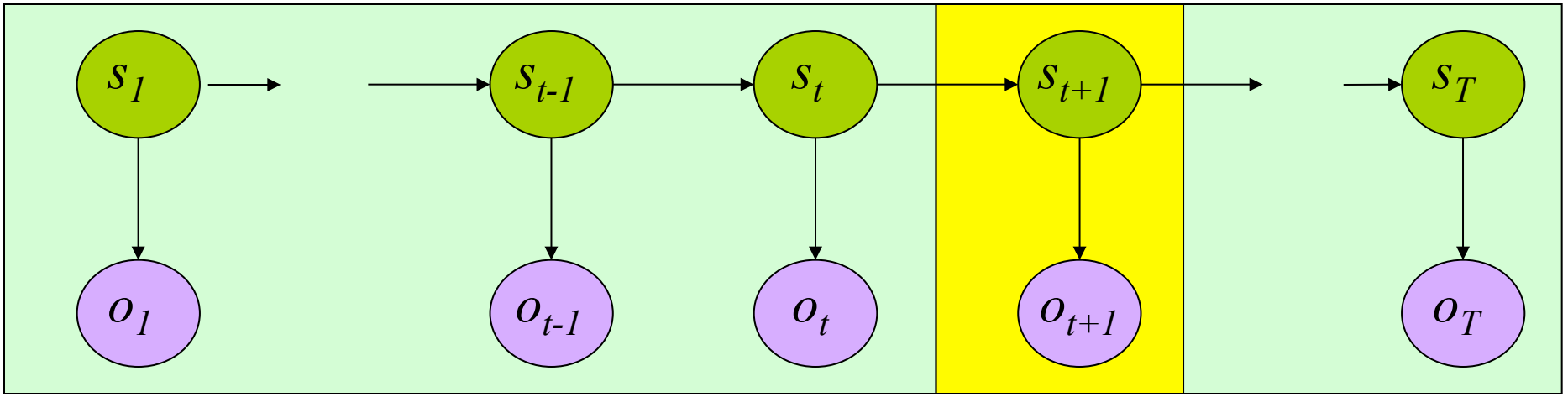
α_t β_t

$$p(s_{t+1}) = \frac{1}{Z}\left[\sum_{s_1}\varphi(s_1,s_2)\cdots\sum_{s_t}\varphi(s_t,s_{t+1})\right]\left[\sum_{s_{t+2}}\varphi(s_{t+1},s_{t+2})\cdots\sum_{s_T}\varphi(s_{T-1},s_T)\right]$$

α_t

β_t

Forward Backward Algorithm

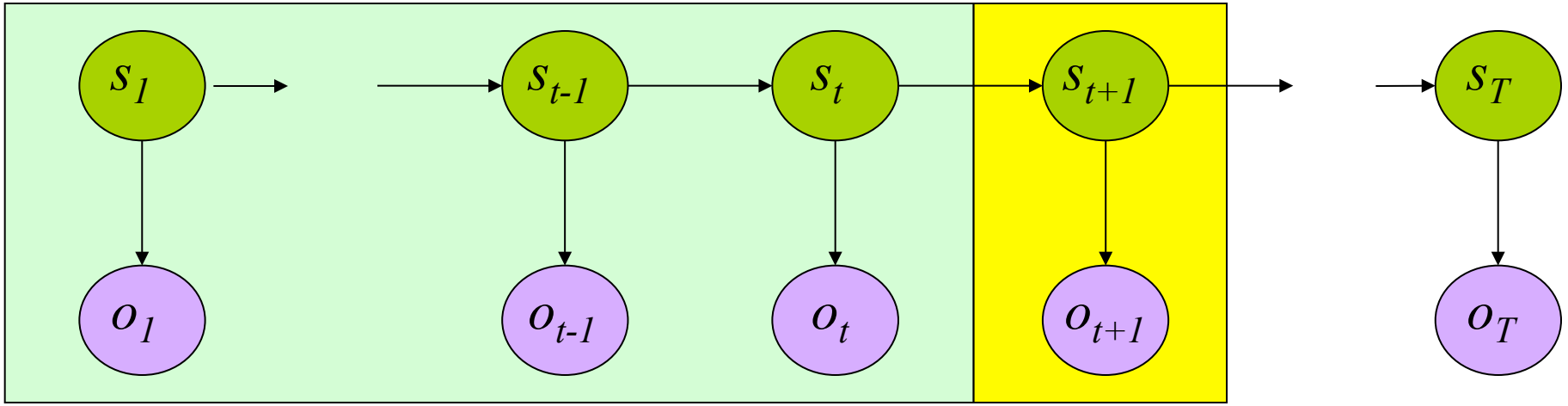


- An efficient algorithm using *dynamic programming*.

$$\alpha_t(i) = P(o_1 \ldots o_t, s_t = i \mid \lambda)$$

$$\beta_t(i) = P(o_t \ldots o_T \mid s_t = i, \lambda)$$

Forward Probability



$$\alpha_t(i) = P(o_1 \ldots o_t, s_t = i \mid \mu)$$

$$\alpha_{t+1}(j) = \sum_{i=1 \ldots N} \alpha_t(i) a_{ij} b_{jo_{t+1}}$$

$$\alpha_1(i) = \pi_i b_{io_1}$$

π are initial state probabilities

Forward Probability

$$\alpha_{t+1}(j)$$

$$= P(o_1 \ldots o_{t+1}, s_{t+1} = j)$$

$$= P(o_1 \ldots o_{t+1} \mid s_{t+1} = j) P(s_{t+1} = j)$$

$$= P(o_1 \ldots o_t \mid s_{t+1} = j) P(o_{t+1} \mid s_{t+1} = j) P(s_{t+1} = j)$$

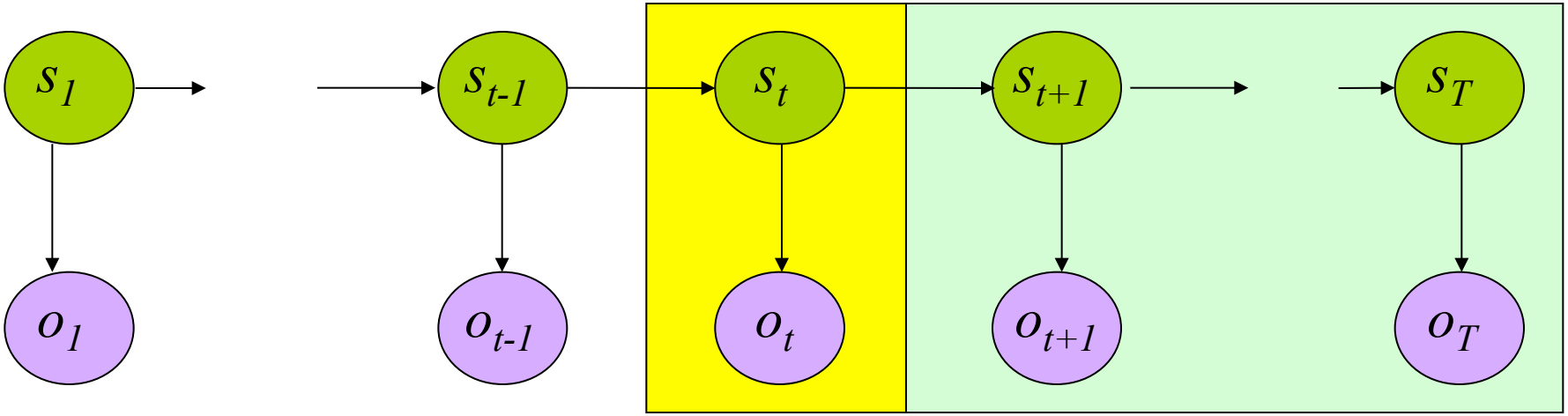
$$= P(o_1 \ldots o_t, s_{t+1} = j) P(o_{t+1} \mid s_{t+1} = j)$$

$$= \sum_{i=1 \ldots N} P(o_1 \ldots o_t, s_t = i, s_{t+1} = j) P(o_{t+1} \mid s_{t+1} = j)$$

$$= \sum_{i=1 \ldots N} P(o_1 \ldots o_t, s_t = i) P(s_{t+1} = j \mid s_t = i) P(o_{t+1} \mid s_{t+1} = j)$$

$$= \sum_{i=1 \ldots N} \alpha_t(i) a_{ij} b_{jo_{t+1}}$$

Backward Probability



$$\beta_t(i) = P(o_t \ldots o_T \mid s_t = i)$$

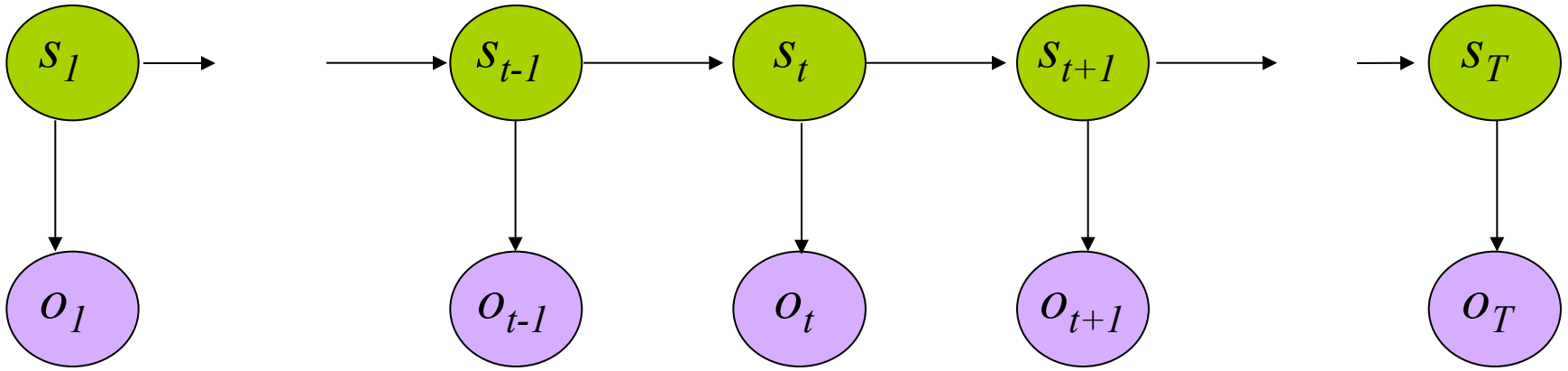
$$\beta_{T+1}(i) = 1$$

$$\beta_t(i) = \sum_{j=1\ldots N} a_{ij} b_{io_t} \beta_{t+1}(j)$$

Inference Algorithm on a Chain

- To compute local marginals:
  - Compute and store all forward messages, $\alpha_t(i)$.
  - Compute and store all backward messages, $\beta_t(i)$.
  - Compute Z at any node s_i
  - Compute marginal probability for all variables required $p(y)$.

Marginal Probability



$$p_t(i,j) = \frac{\alpha_{t-1}(i) a_{ij} b_{jo_t} \beta_t(j)}{\sum_{k=1\ldots N} \alpha_t(k)\beta_t(k)}$$

$$p_t(i) = \frac{\alpha_t(i)\beta_t(j)}{\sum_{k=1\ldots N} \alpha_t(k)\beta_t(k)}$$

Calculating the Expectation

- First we define the transition matrix of y for position x as

$$M_n[y_{n-1}, y_n] = \exp \lambda \cdot f(y_{n-1}, y_n, x)$$

$$E_{p(Y|x^{(m)})}\left[F_k(Y, x^{(m)})\right] = \sum_y p(y \mid x^{(m)}) F_k(y, x)$$

$$= \sum_{n=1}^{N} \sum_{y_{n-1}, y_n} p(y_{n-1}, y_n \mid x^{(m)}) f_k(y_{n-1}, y_n, x^{(k)}) + \sum_{n=1}^{N} \sum_{y_n} p(y_n \mid x^{(m)}) f_k(y_n, x^{(m)})$$

$$p(y_{n-1}, y_n \mid x^{(k)}) = \frac{\alpha_{n-1}(M_n * V_n)\beta_n^T}{Z(x)}$$

$$p(y_n \mid x^{(k)}) = \frac{\alpha_n \beta_n^T}{Z(x)}$$

$$Z(x) = \left[\prod_{n=1}^{N+1} M_n(x)\right] = \alpha_N \cdot 1^T$$

$$\alpha_n = \begin{cases} \alpha_n M_n & 0 < n \le N \\ 1 & n = 0 \end{cases}$$

$$\beta_n^T = \begin{cases} M_{n+1}\beta_{n+1}^T & 1 \le n < N \\ 1 & n = N \end{cases}$$

All state features at position *n*

IIS Algorithm

First-order numerical optimization

Using Iterative Scaling (GIS, IIS)

- Initialize each λ_k(=0 for example)
- Until convergence
  - Solve $\frac{\delta L}{\delta \lambda_k} = 0$ for each parameter λ_k
  - Update each parameter using λ_k<- $\lambda_k + \Delta\lambda_k$

Low efficient!!

Second-order numerical optimization

Using newton optimization technique for the parameter estimation

$$\lambda^{(k+1)} = \lambda^{(k)} + (\frac{\partial^2 L}{\partial \lambda^2})^{-1} \frac{\partial L}{\partial \lambda}$$

Drawbacks: parameter value initialization
And compute the second order (i.e. hesse matrix), that is difficult

Solutions:
- Conjugate-gradient (CG) (Shewchuk, 1994)
- Limited-memory quasi-Newton (L-BFGS) (Nocedal and Wright, 1999)
- Voted Perceptron (Colloins 2002)

Summary of CRFs

Model

•Lafferty, 2001

Applications

•Efficient training (Wallach, 2003)
•Training via. Gradient Tree Boosting (Dietterich, 2004)
•Bayesian Conditional Random Fields (Qi, 2005)
•Name entity (McCallum, 2003)
•Shallow parsing (Sha, 2003)
•Table extraction (Pinto, 2003)
•Signature extraction (Kristjansson, 2004)
•Accurate Information Extraction from Research Papers (Peng, 2004)
•Object Recognition (Quattoni, 2004)
•Identify Biomedical Named Entities (Tsai, 2005)
•…

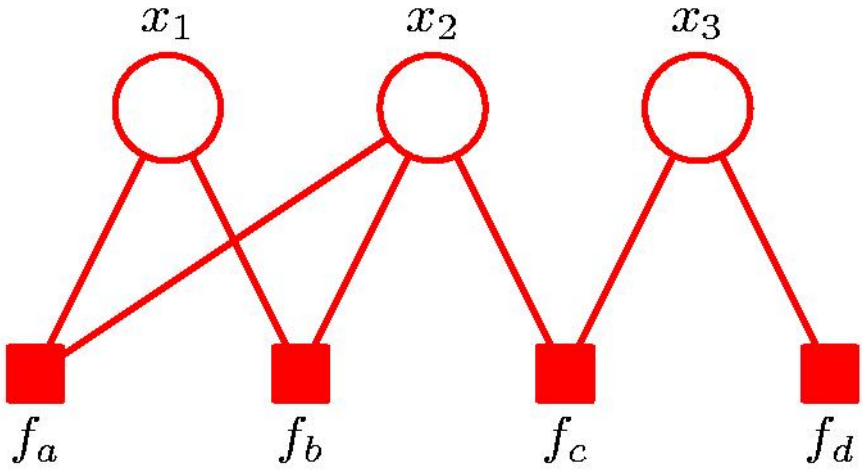
Limitation

•Huge computational cost in parameter estimation

Outline

- Introduction
- Directed Graphical Models
- Undirected Graphical Models
- Factor Graph

Factor Graphs

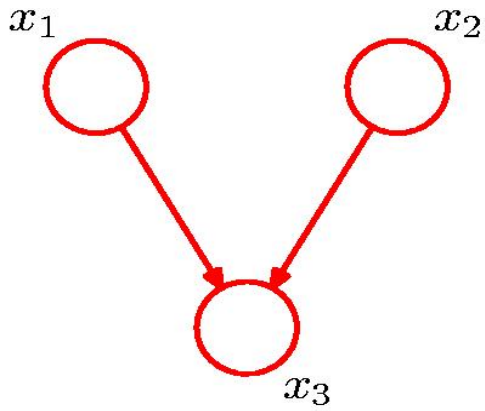


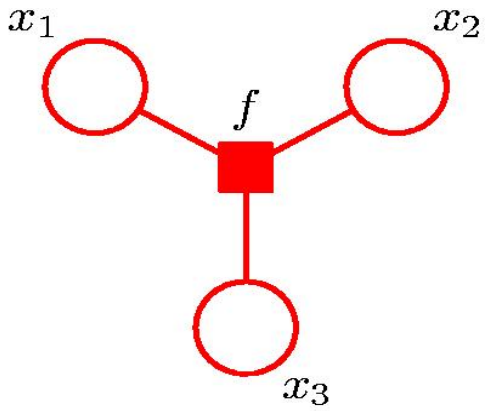
$$p(\mathbf{x}) = f_a(x_1, x_2) f_b(x_1, x_2) f_c(x_2, x_3) f_d(x_3)$$

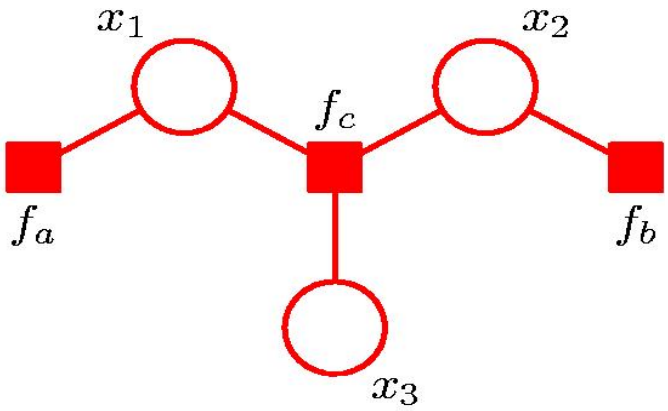
$$p(\mathbf{x}) = \prod_s f_s(\mathbf{x}_s)$$

A number of the following pages are from Bishop's slides

Factor Graphs from Directed Graphs







$$p(\mathbf{x}) = p(x_1)p(x_2)p(x_3|x_1,x_2)$$

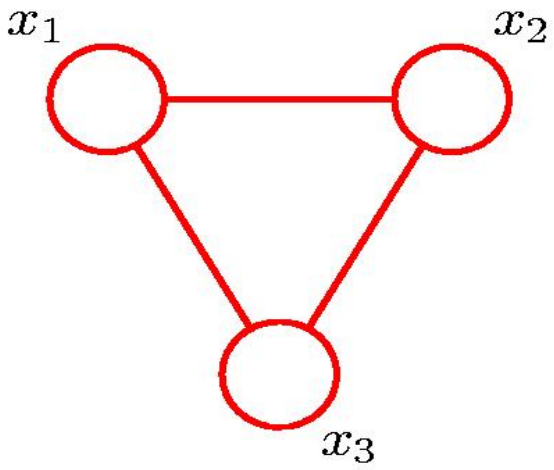
$$f(x_1,x_2,x_3) = p(x_1)p(x_2)p(3|x_1,x_2)$$

$$f_a(x_1) = p(x_1)$$

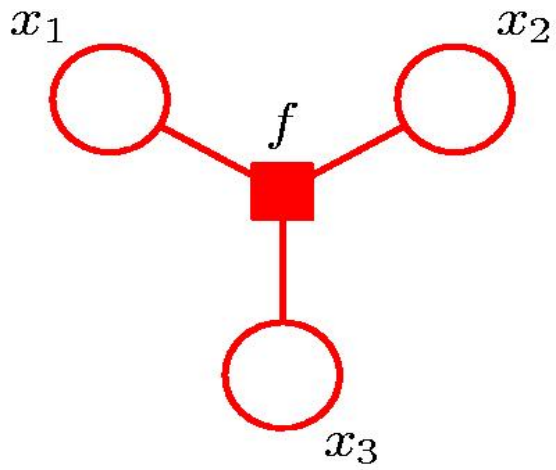
$$f_b(x_2) = p(x_2)$$

$$f_c(x_1,x_2,x_3) = p(x_3|x_1,x_2)$$

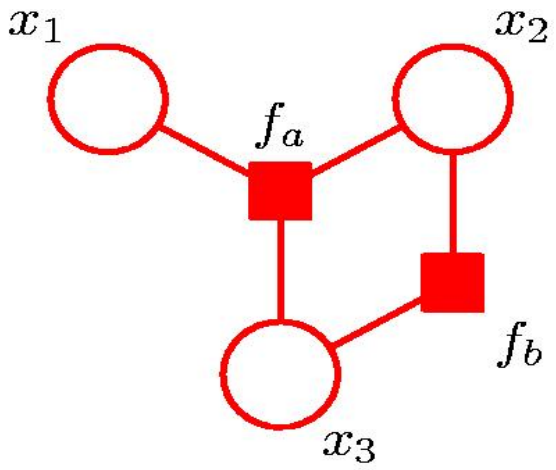
Factor Graphs from Undirected Graphs



$$\psi(x_1, x_2, x_3)$$



$$f(x_1, x_2, x_3) = \psi(x_1, x_2, x_3)$$



$$f_a(x_1, x_2, x_3) f_b(x_2, x_3) = \psi(x_1, x_2, x_3)$$

Example

Inferring Social Ties across Heterogeneous Networks

(WSDM'12, ACM TOIS'15)

Jie Tang*, Tiancheng Lou*, and Jon Kleinberg+

*Tsinghua University

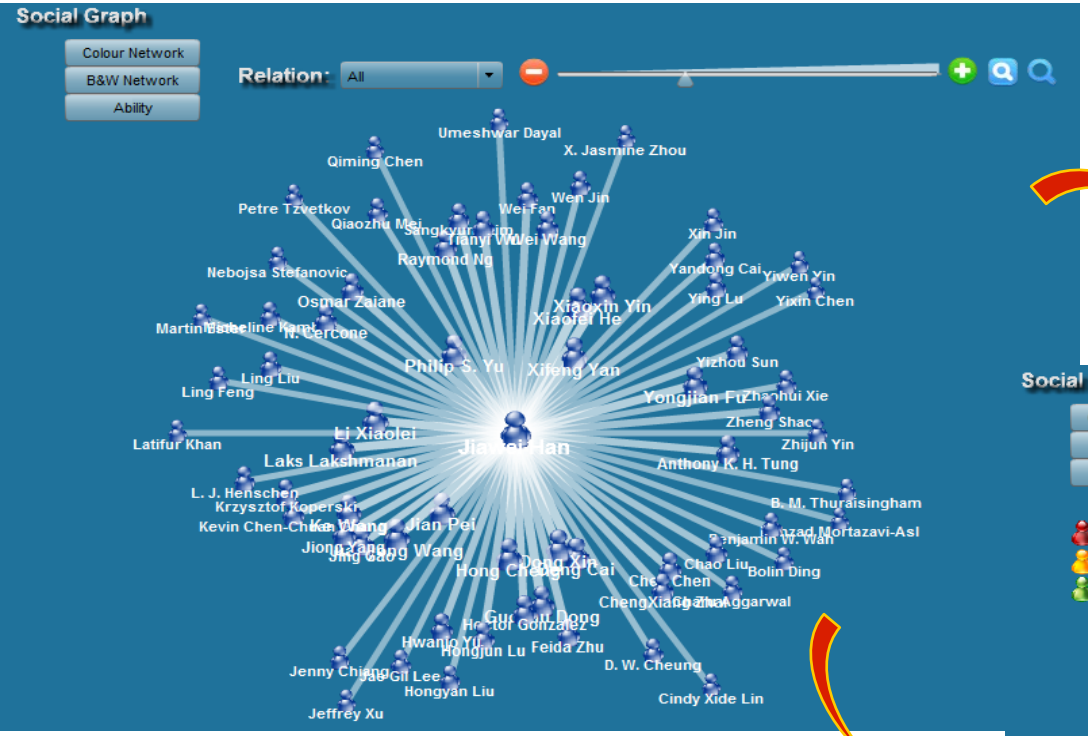
+Cornell University

Real social networks are complex...

- Different social ties have different influence on people
  - Close friends vs. Acquaintances
  - Colleagues vs. Family members
- However, existing networks (e.g., Facebook and Twitter) are trying to lump everyone into one big network
  - FB tries to solve this problem via lists/groups
  - However…
- Google+



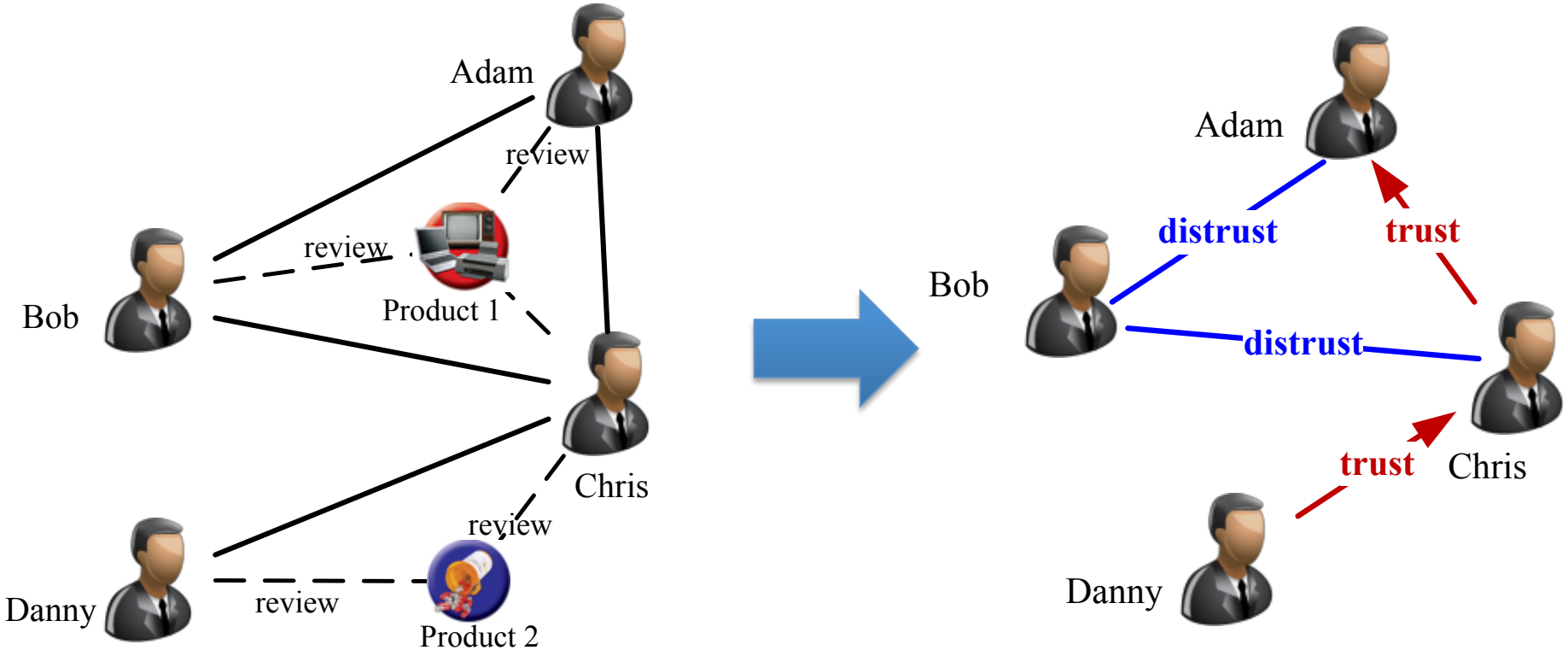
Example 1. Advisor-advisee relationship



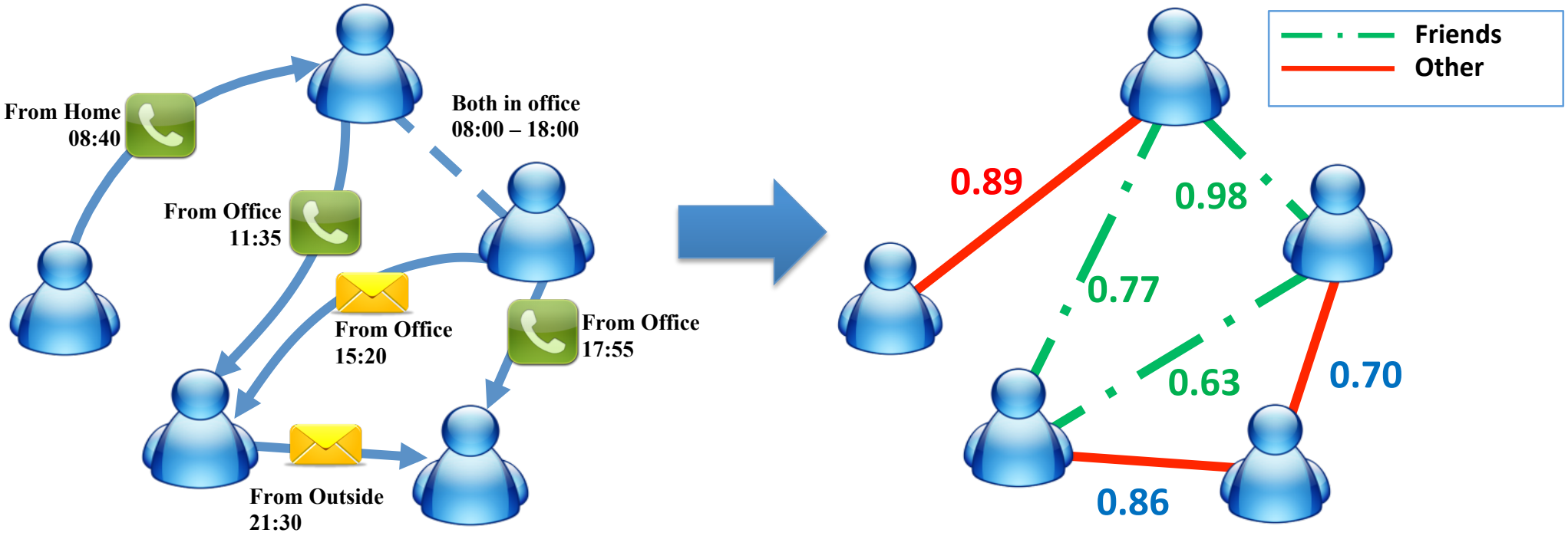


Arnetminer

Example 2. Trustful relationship



Example 3: Friendship in mobile network



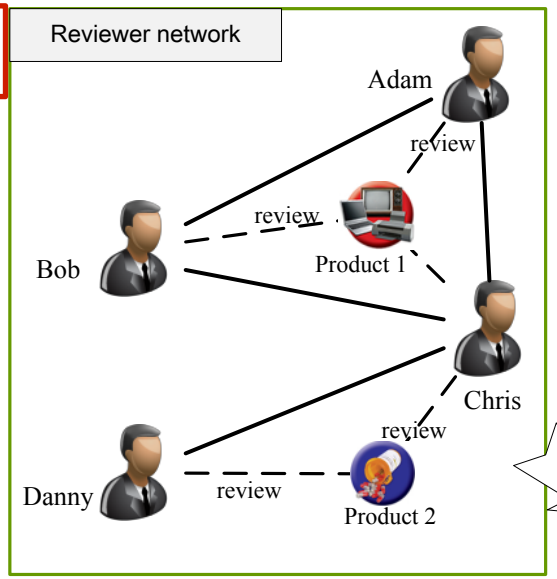
Mobile

Inferring Social Ties Across Networks

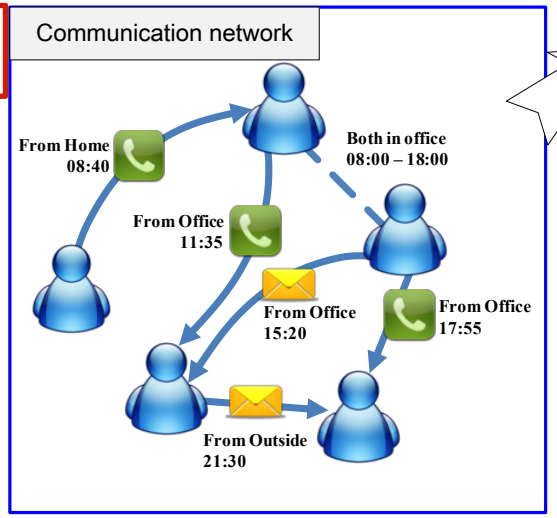
Input: Heterogeneous Networks

Output: Inferred social ties in different networks

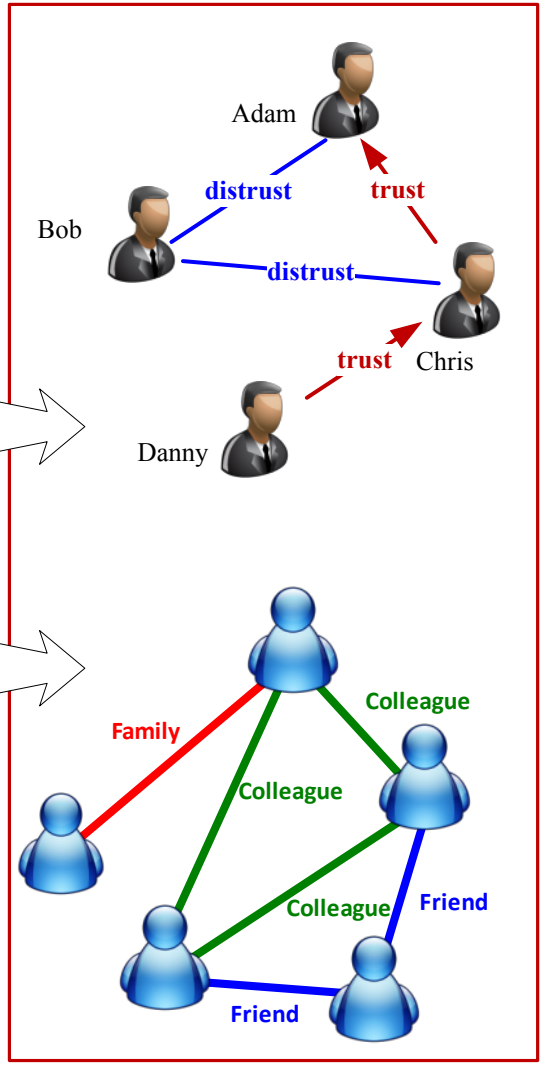
Epinions



Mobile



Knowledge Transfer for Inferring Social Ties

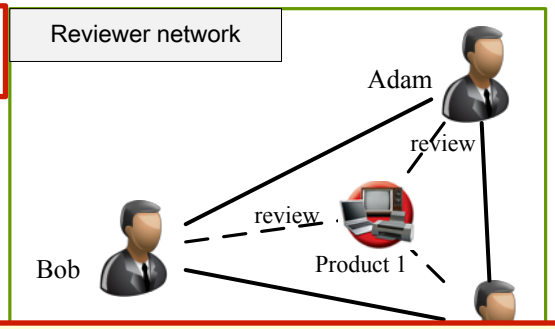


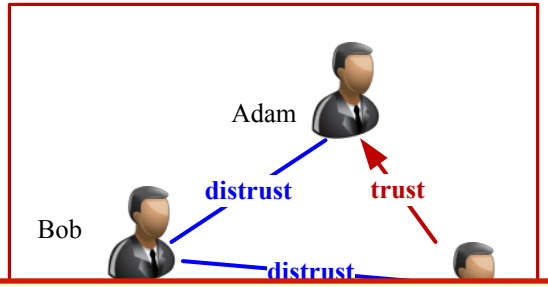
Inferring Social Ties Across Networks

Input: Heterogeneous Networks

Output: Inferred social ties in different networks

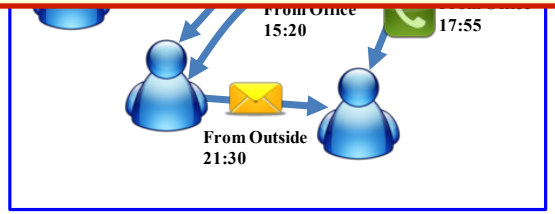
Epinions

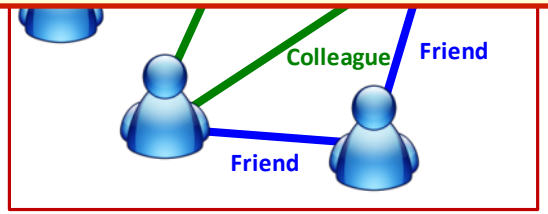




Questions:

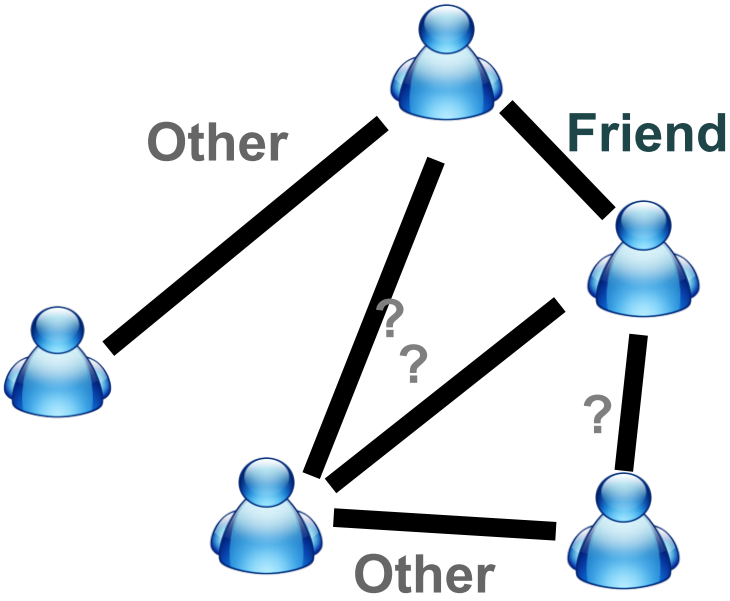
- What are the fundamental forces behind?
- A generalized framework for inferring social ties?
- How to connect the different networks?





Problem Formulation in a Single Network

Input: $G=(V,E^L,E^U,R^L,W)$



V: Set of Users

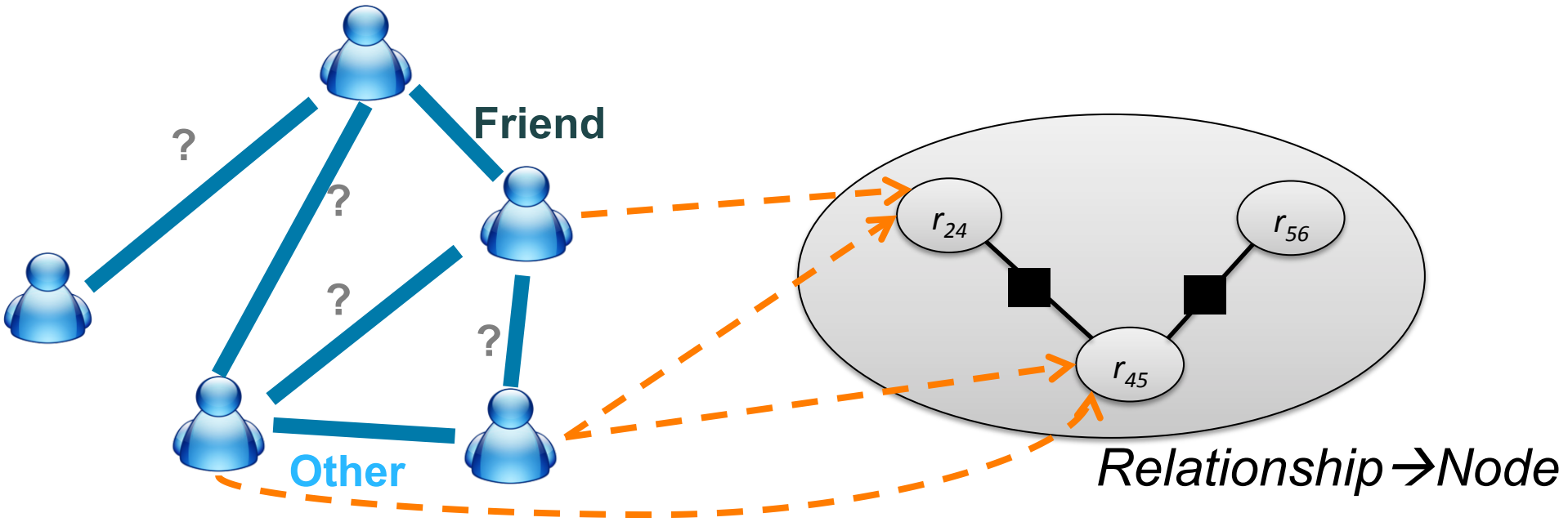
E^L, R^L: Labeled relationships

E^U: Unlabeled relationships

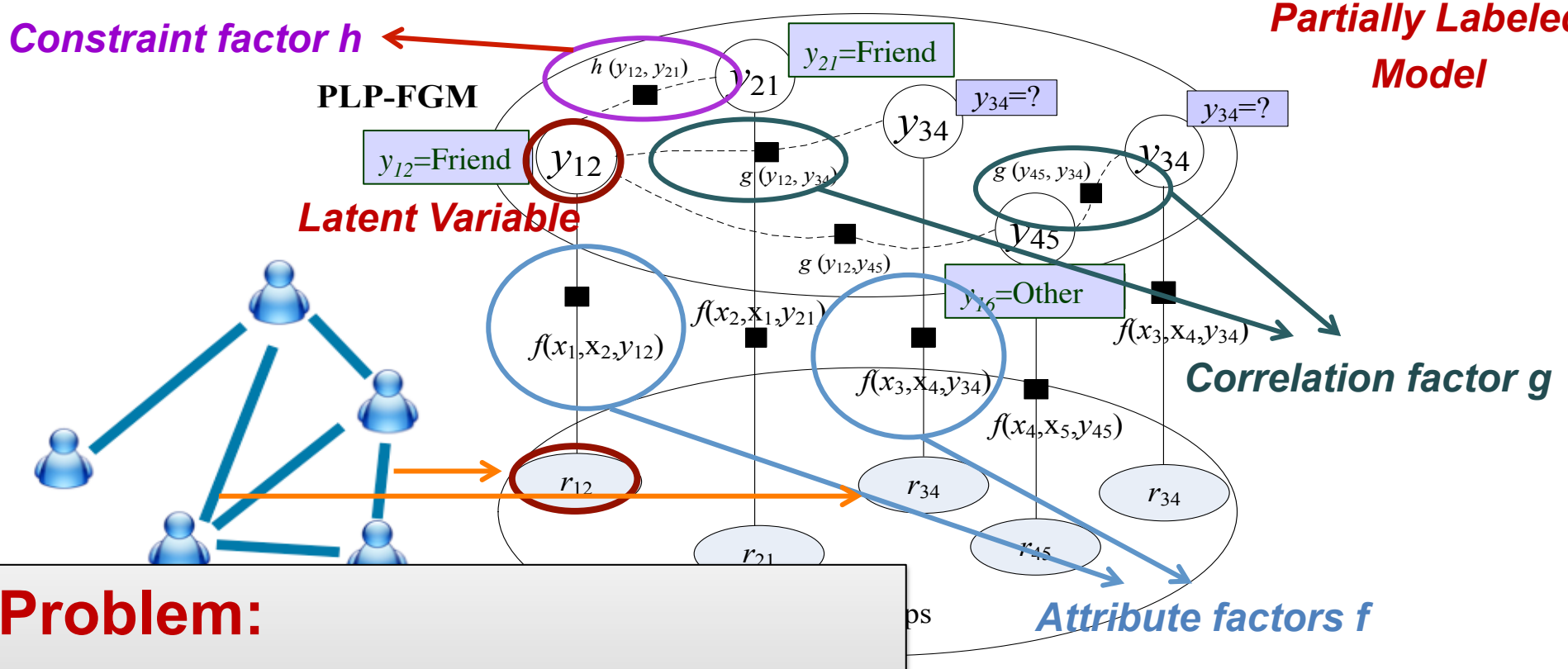
Input: $G=(V,E^L,E^U,R^L,W)$

Output: $f: G \rightarrow R$

Basic Idea



Partially Labeled Pairwise Factor Graph Model (PLP-FGM)



Problem:

For each relationship, identify which type has the highest probability?

A makes call to B immediately after the call to C.

Solutions(con't)

- Different ways to instantiate factors
  - We use exponential-linear functions
    - Attribute Factor:

$$f(y_i, \mathbf{x}_i) = \frac{1}{Z_\lambda} \exp\{\lambda^T \mathbf{\Phi}(y_i, \mathbf{x}_i)\}$$

    - Correlation / Constraint Factor:

$$g(y_i, G(y_i)) = \frac{1}{Z_\alpha} \exp\{\sum_{y_j \in G(y_i)} \alpha^T \mathbf{g}(y_i, y_j)\}$$

$$h(y_i, H(y_i)) = \frac{1}{Z_\beta} \exp\{\sum_{y_j \in H(y_i)} \beta^T \mathbf{h}(y_i, y_j)\}$$

  - Log-Likelihood of labeled Data:

$$\mathcal{O}(\theta) = \log \sum_{Y|Y^L} \exp\{\theta^T \mathbf{S}\} - \log \sum_{Y} \exp\{\theta^T \mathbf{S}\}$$

Parameters to estimate $\theta = [\lambda, \alpha, \beta], s = [\Phi^T, g^T, h^T]^T$

Learning Algorithm

- Maximize the log-likelihood of labeled relationships

Input: learning rate η
Output: learned parameters θ

Initialize θ;
repeat
  Calculate $\mathbb{E}_{p_\theta(Y|Y^L,G)}\mathbf{S}$ using LBP ;
  Calculate $\mathbb{E}_{p_\theta(Y|G)}\mathbf{S}$ using LBP ;
  Calculate the gradient of θ according to Eq. 7:

$$\nabla_\theta = \mathbb{E}_{p_\theta(Y|Y^L,G)}\mathbf{S} - \mathbb{E}_{p_\theta(Y|G)}\mathbf{S}$$

  Update parameter θ with the learning rate η:

$$\theta_{\text{new}} = \theta_{\text{old}} - \eta \cdot \nabla_\theta$$

until *Convergence*;

Algorithm 1: Learning PLP-FGM.

Expectation Computing

Loopy Belief Propagation

Gradient Ascent Method

Still Challenges?

Questions:

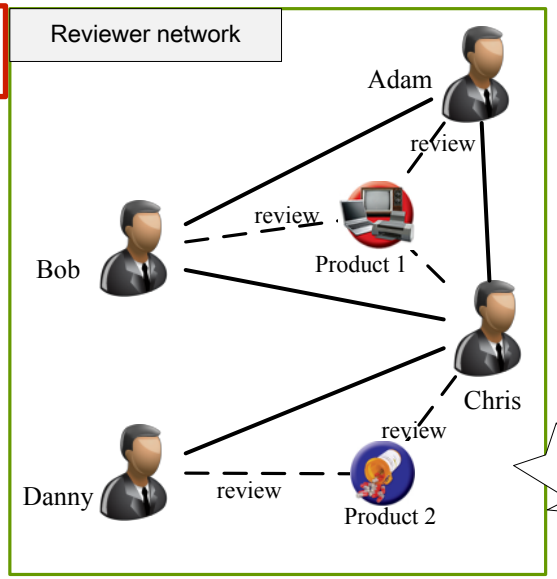
- How to obtain sufficiently training data?
- Can we leverage knowledge from other network?

Inferring Social Ties Across Networks

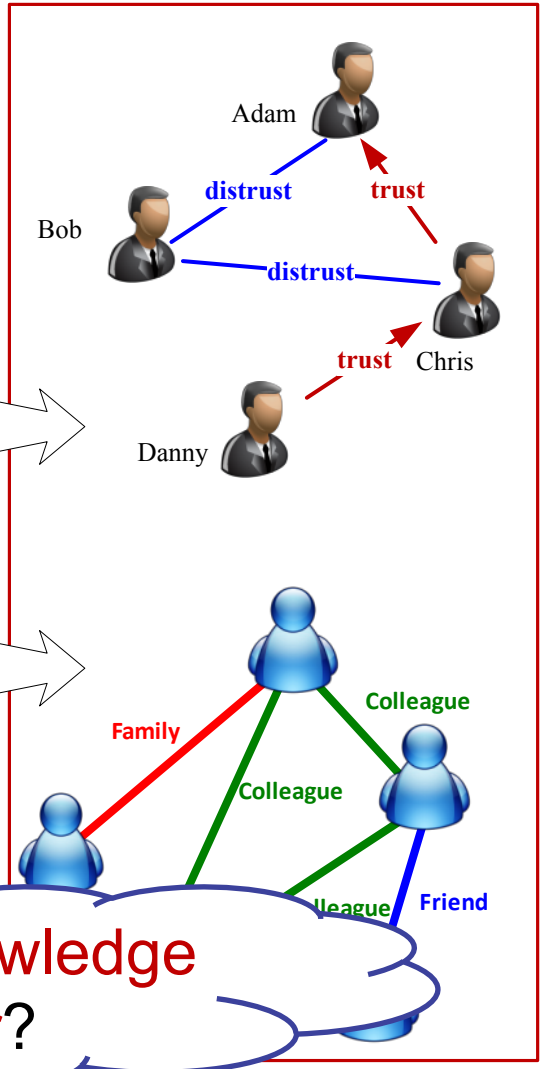
Input: Heterogeneous Networks

Output: Inferred social ties in different networks

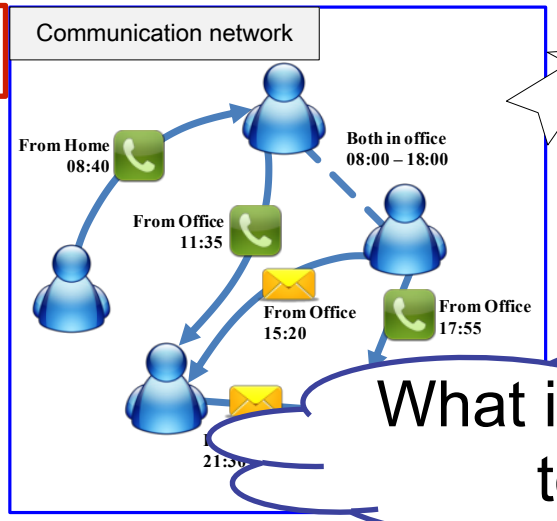
Epinions



Knowledge Transfer for Inferring Social Ties



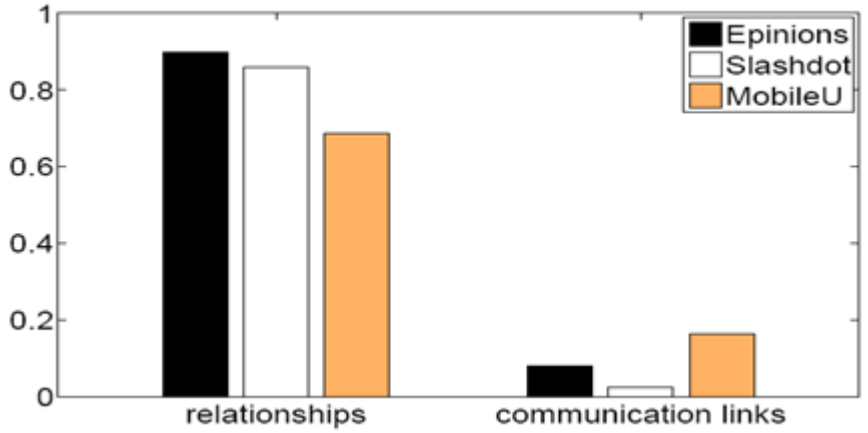
Mobile



What is the knowledge to transfer?

Social Theories

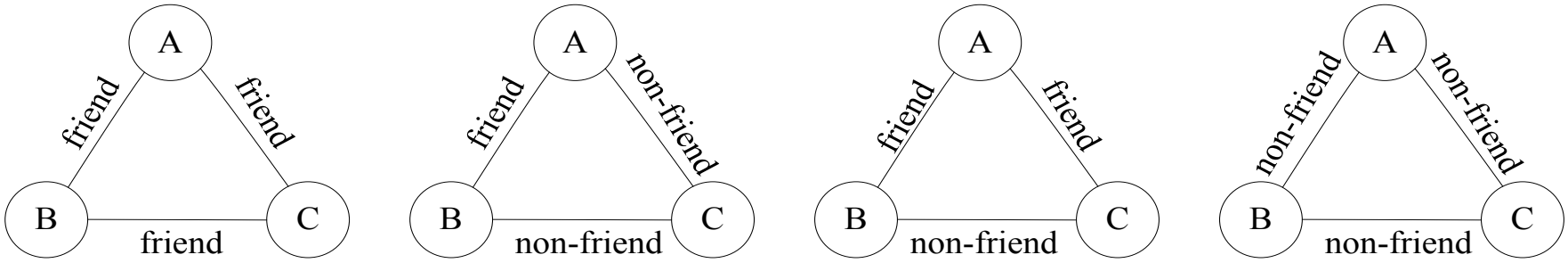
- Social balance theory
- Structural hole theory
- Social status theory
- Two-step-flow theory



Observations:

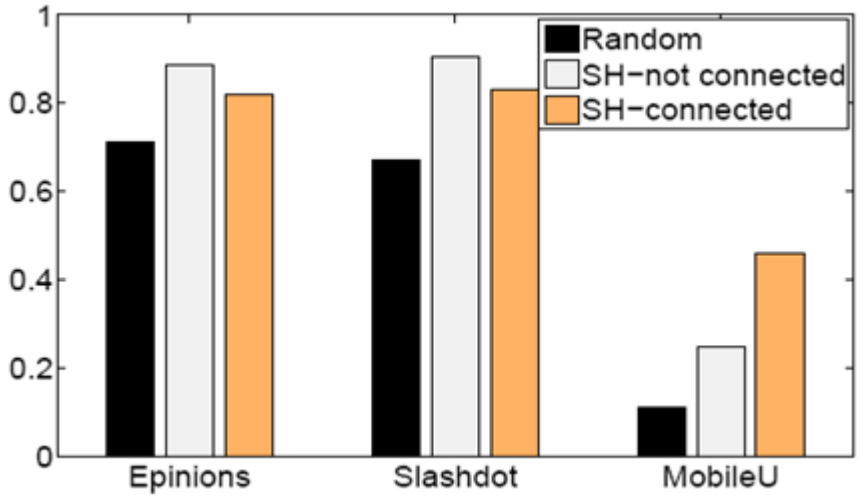
(1) The underlying networks are unbalanced;

(2) While the friendship networks are balanced.

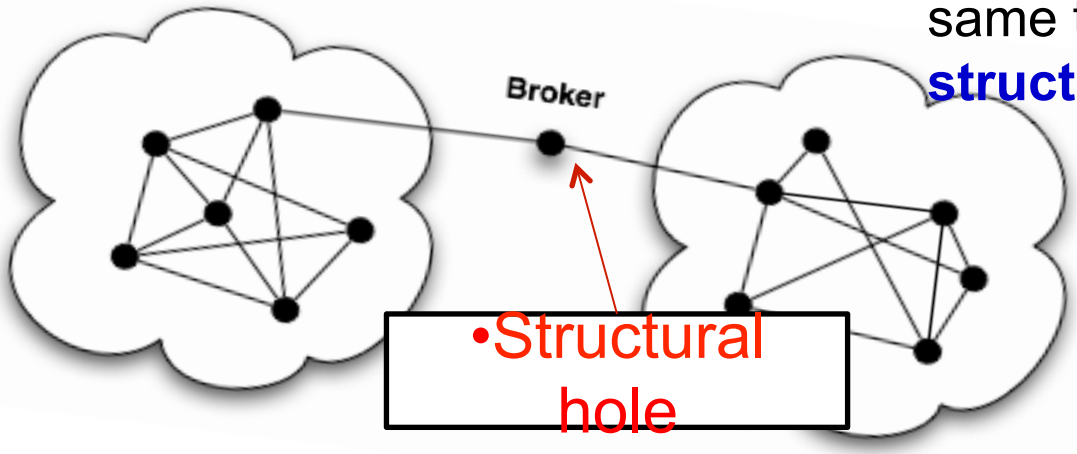


Social Theories—Structural hole

- Social balance theory
- Structural hole theory
- Social status theory
- Two-step-flow theory

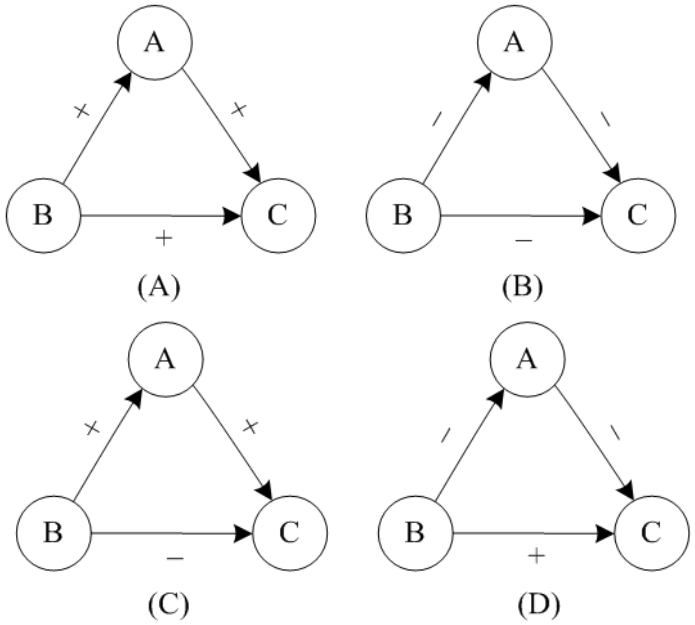


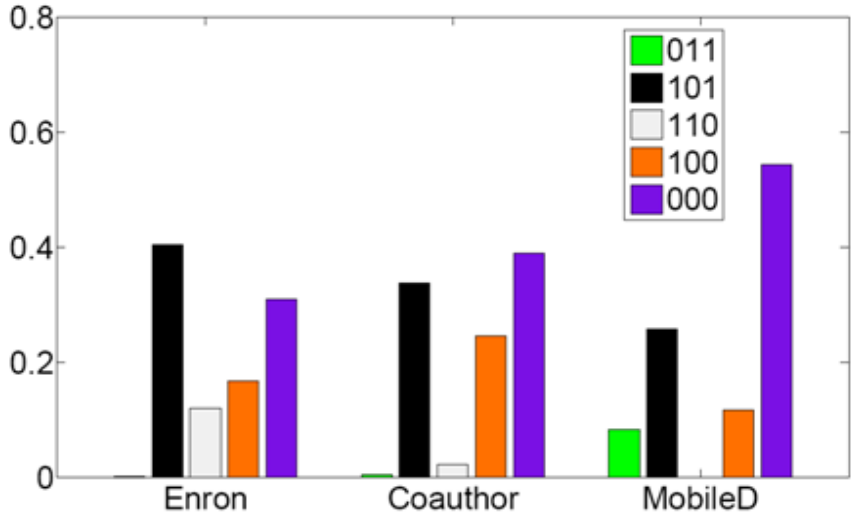
Observations: Users are more likely (+25-150% higher than change) to have the same type of relationship with C if C **spans structural holes**



Social Theories—Social status

- Social balance theory
- Structural hole theory
- Social status theory
- Two-step-flow theory



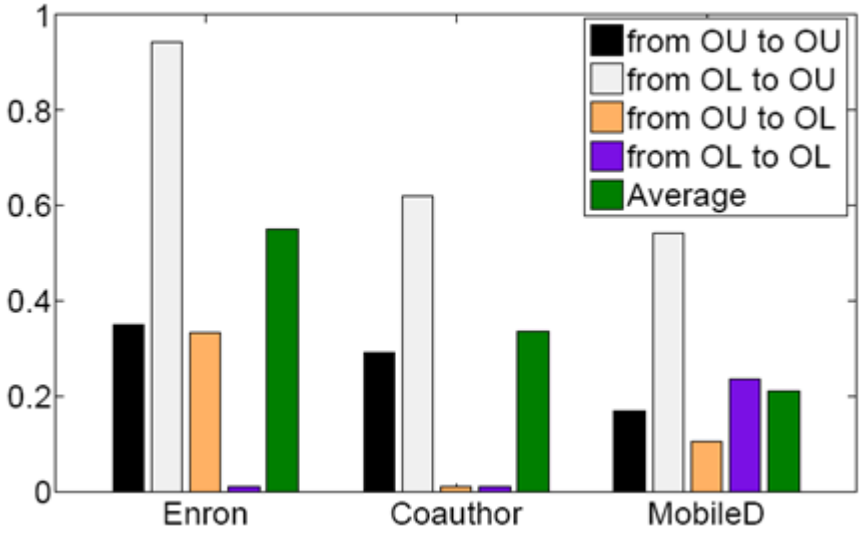


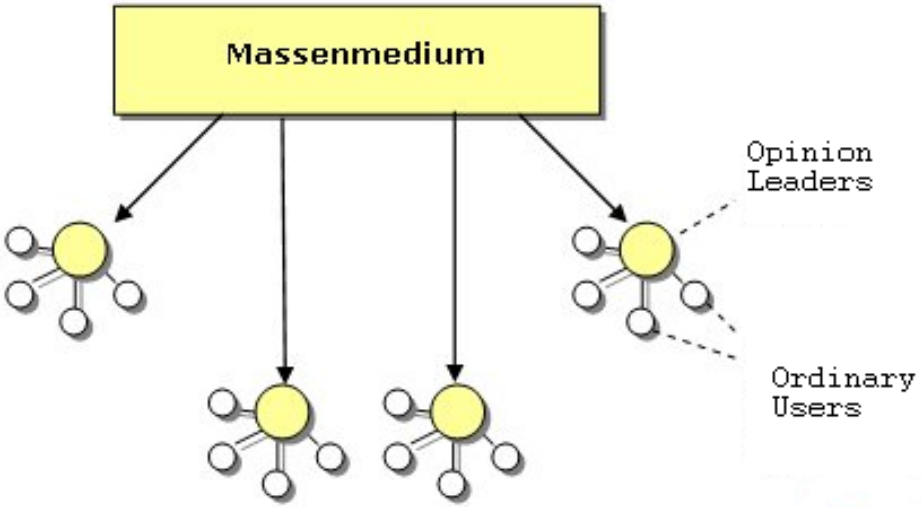
Observations: 99% of triads in the networks satisfy the social status theory

•**Note:** Given a triad (A,B,C), let us use 1 to denote the advisor-advisee relationship and 0 colleague relationship. Thus the number 011 to denote A and B are colleagues, B is C's advisor and A is C's advisor.

Social Theories—Two-step-flow

- Social balance theory
- Structural hole theory
- Social status theory
- Two-step-flow theory





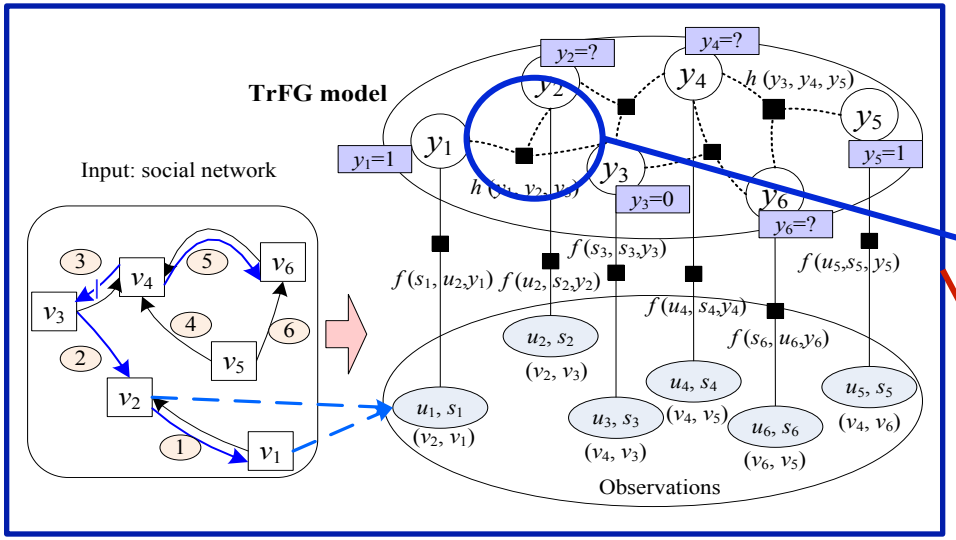
OL : Opinion leader;

OU : Ordinary user.

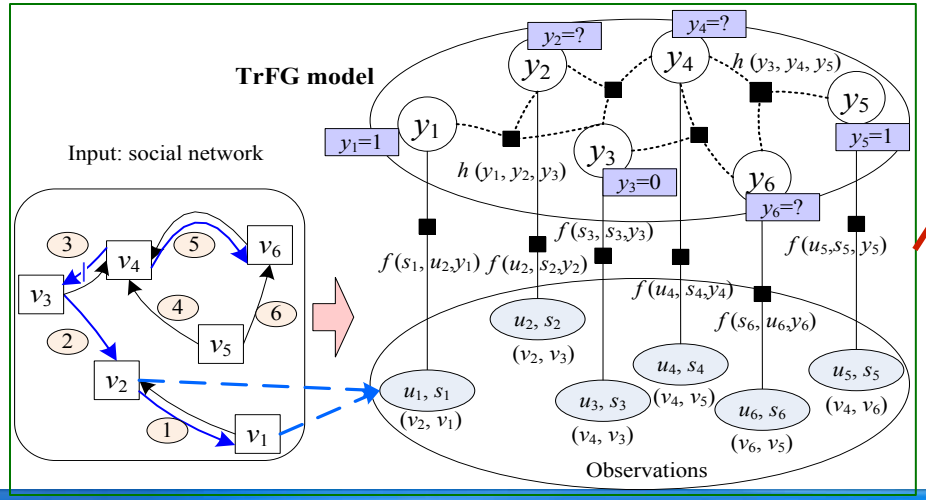
Observations: Opinion leaders are more likely (+71%-84% higher than chance) to have a higher social-status than ordinary users.

Transfer Factor Graph Model

Coauthor network



mobile



Triad-based factor

Bridge via social theories

Mathematical Formulation

Features defined in source network

Features defined in target network

$$\mathcal{O}(\alpha, \beta, \mu) = \mathcal{O}_S(\alpha, \mu) + \mathcal{O}_T(\beta, \mu)$$

$$= \sum_{i=1}^{|V_S|} \sum_{j=1}^{d} \alpha_j g_j(x_{ij}^S, y_i^S) + \sum_{i=1}^{|V_T|} \sum_{j=1}^{d'} \beta_j g'_j(x_{ij}^T, y_i^T)$$

$$+ \sum_k \mu_k \Big(\sum_{c \in G_S} h_k(Y_c^S) + \sum_{c \in G_T} h_k(Y_c^T) \Big)$$

$$- \log Z$$

Triad-based features shared across networks

Data Sets

- **Epinions** a network of product reviewers: 131,828 nodes (users) and 841,372 edges
  - trust relationships between users
- **Slashdot**: 82,144 users and 59,202 edges
  - "friend" relationships between users
- **Mobile**: 107 mobile users and 5,436 edges
  - to infer friendships between users

Undirected network

- **Coauthor**: 815,946 authors and 2,792,833 coauthor relationships
  - to infer advisor-advisee relationships between coauthors
- **Enron**: 151 Enron employees and 3572 edges
  - to infer manager-subordinate relationships between users

Directed network

Results – undirected networks

- **SVM** and **CRF** are two baseline methods
- **PFG** is the proposed partially-labeled factor graph model
- **TranFG** is the proposed transfer–based factor graph model.

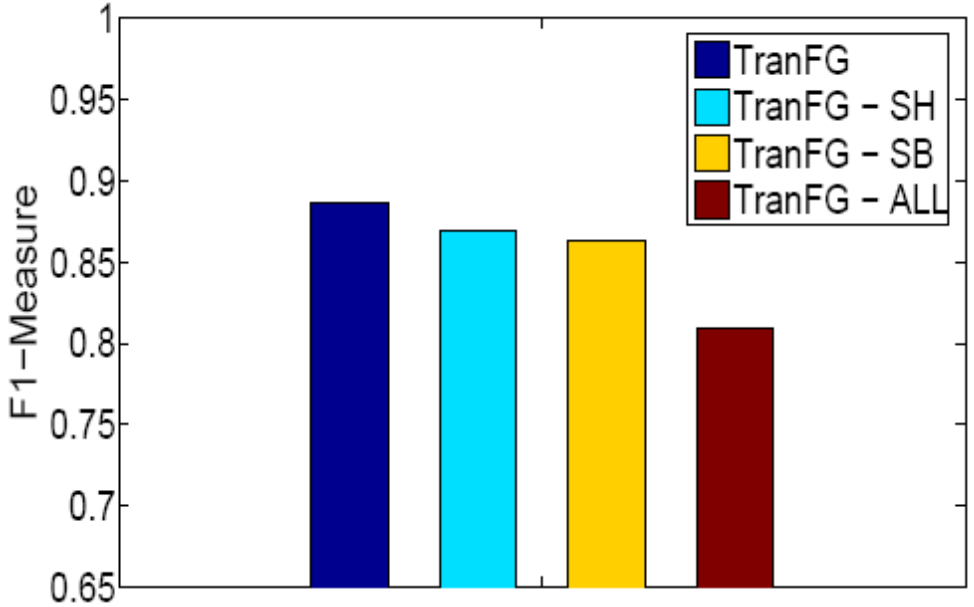
Data Set	Method	Prec.	Rec.	F1-score
Epinions (S) to Slashdot (T) (40%)	SVM	0.7157	**0.9733**	0.8249
	CRF	0.8919	0.6710	0.7658
	PFG	0.9300	0.6436	0.7607
	TranFG	**0.9414**	0.9446	**0.9430**
Slashdot (S) to Epinions (T) (40%)	SVM	0.9132	**0.9925**	0.9512
	CRF	0.8923	0.9911	0.9393
	PFG	**0.9954**	0.9787	**0.9870**
	TranFG	**0.9954**	0.9787	**0.9870**
Epinions (S) to Mobile (T) (40%)	SVM	0.8983	0.5955	0.7162
	CRF	0.9455	0.5417	0.6887
	PFG	**1.0000**	0.5924	0.7440
	TranFG	0.8239	**0.8344**	**0.8291**
Slashdot (S) to Mobile (T) (40%)	SVM	0.8983	0.5955	0.7162
	CRF	0.9455	0.5417	0.6887
	PFG	**1.0000**	0.5924	0.7440
	TranFG	0.7258	**0.8599**	**0.7872**

Results – directed networks

- **SVM** and **CRF** are two baseline methods
- **PFG** is the proposed partially-labeled factor graph model
- **TranFG** is the proposed transfer–based factor graph model.

Data Set	Method	Prec.	Rec.	F1-score
Coauthor (S) to Enron (T) (40%)	SVM	0.9524	0.5556	0.7018
	CRF	0.9565	0.5366	0.6875
	PFG	**0.9730**	0.6545	0.7826
	TranFG	0.9556	**0.7818**	**0.8600**
Enron (S) to Coauthor (T) (40%)	SVM	0.6910	0.3727	0.4842
	CRF	**1.0000**	0.3043	0.4666
	PFG	0.9916	0.4591	0.6277
	TPFG	0.5936	**0.7611**	0.6669
	TranFG	0.9793	0.5525	**0.7065**

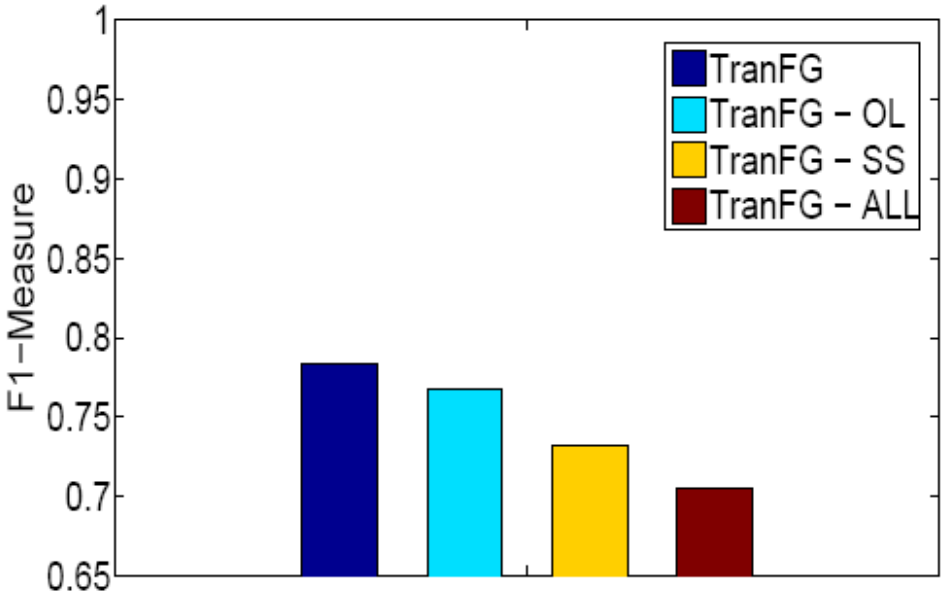
Factor Contribution Analysis



SH-Structural hole;

SB-Social balance.

Undirected Network



OL-Opinion leader;

SS-Social status.

Directed Network

Conclusions and Future Work

- **Conclusions**
  - different types of social ties have essentially different structural patterns in social networks;
  - By incorporating social theories, our proposed model can significantly improve (+4-14%) the inferring accuracy.

- **Future work**
  - Inferring complex relationships between users, e.g., family, colleague, manager-subordinate;
  - Active learning for inferring social ties.

Thanks!

HP: http://keg.cs.tsinghua.edu.cn/~jietang/

Email: jietang@tsinghua.edu.cn