MACHINE LEARNING
DEPARTMENT

Graphical Models

10-715 Fall 2015

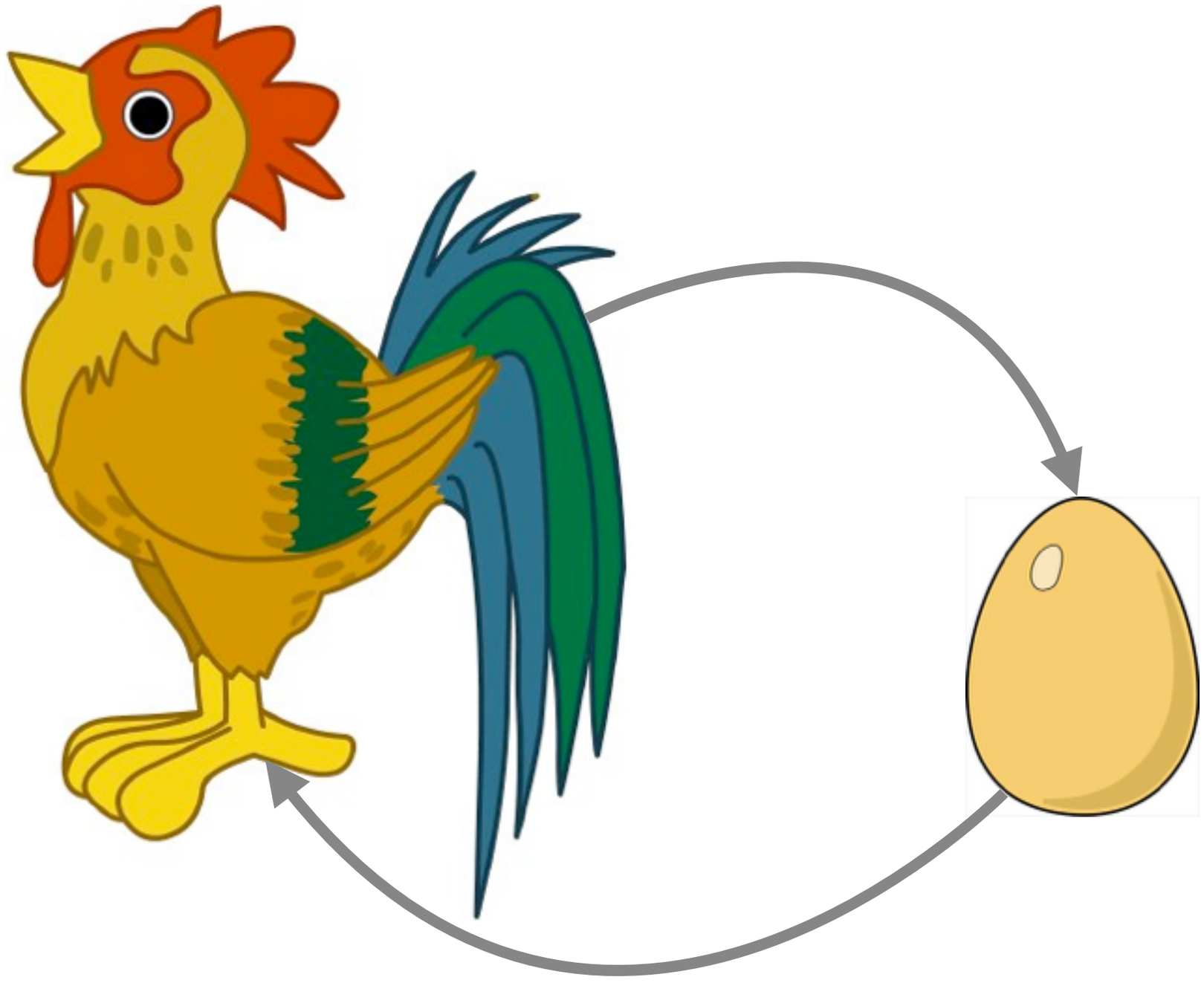
Alexander Smola
alex@smola.org

Office hours - after class in my office

Marianas Labs

Carnegie Mellon University

Directed Graphical Models



Brain & Brawn

$$p(\text{brain}) = 0.1$$

$$p(\text{sports}) = 0.2$$

	0	1
0	0.1	0.8
1	0.8	0.9

$$p(g, s, b) = p(g|s, b)p(s)p(b)$$

Brain & Brawn

$$p(\text{brain}) = 0.1$$

$$p(\text{sports}) = 0.2$$

?	0	1
0	0.72	0.08
1	0.18	0.02

$$p(s, b) = p(s)p(b)$$

	0	1
0	0.1	0.8
1	0.8	0.9

$$p(g, s, b) = p(g|s, b)p(s)p(b)$$

Brain & Brawn

element-wise multiply

$$p(\text{brain}) = 0.1$$
$$p(\text{sports}) = 0.2$$

g=1	0	1
0	0.072	0.064
1	0.144	0.018

	0	1
0	0.1	0.8
1	0.8	0.9

$$p(s, b|g) = \frac{p(s)p(b)p(g|s,b)}{\sum_{s',b'} p(s')p(b')p(g|s',b')} \quad p(g,s,b) = p(g|s,b)p(s)p(b)$$

Brain & Brawn

renormalize to 1

$$p(\text{brain}) = 0.1$$
$$p(\text{sports}) = 0.2$$

g=1	0	1
0	0.242	0.215
1	0.483	0.06

	0	1
0	0.1	0.8
1	0.8	0.9

$$p(s,b|g) = \frac{p(s)p(b)p(g|s,b)}{\sum_{s',b'} p(s')p(b')p(g|s',b')} \quad p(g,s,b) = p(g|s,b)p(s)p(b)$$

Brain & Brawn

$$p(\text{brain}) = 0.1$$
$$p(\text{sports}) = 0.2$$

| g=1	0	1
0	0.242	0.215
1	0.483	0.06

$$p(\text{brain}|\text{graduate}) = 0.275$$
$$p(\text{sports}|\text{graduate}) = 0.544$$

$$p(\text{brain}|\text{graduate}, \text{sports}) = 0.111$$
$$p(\text{brain}|\text{graduate}, \text{nosports}) = 0.471$$
$$p(\text{sports}|\text{graduate}, \text{brain}) = 0.220$$
$$p(\text{sports}|\text{graduate}, \text{nobrain}) = 0.333$$

$$p(s, b|g) = \frac{p(s)p(b)p(g|s, b)}{\sum_{s',b'} p(s')p(b')p(g|s', b')} \quad p(g, s, b) = p(g|s, b)p(s)p(b)$$

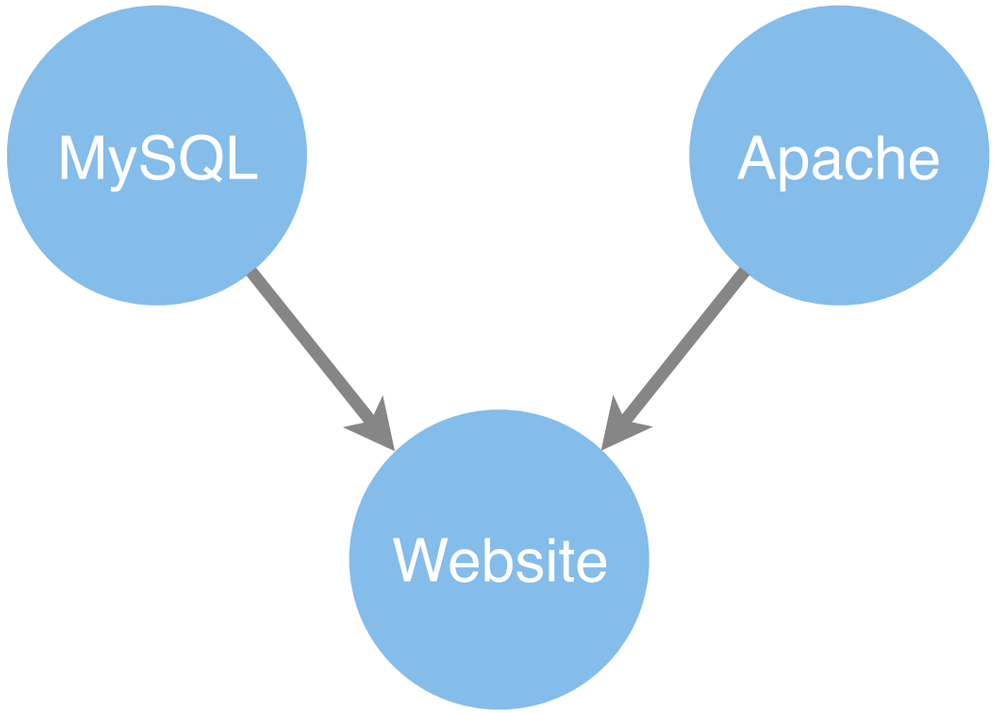
Brain & Brawn

$$p(g, s, b) = p(g)p(s|g)p(b|g)$$

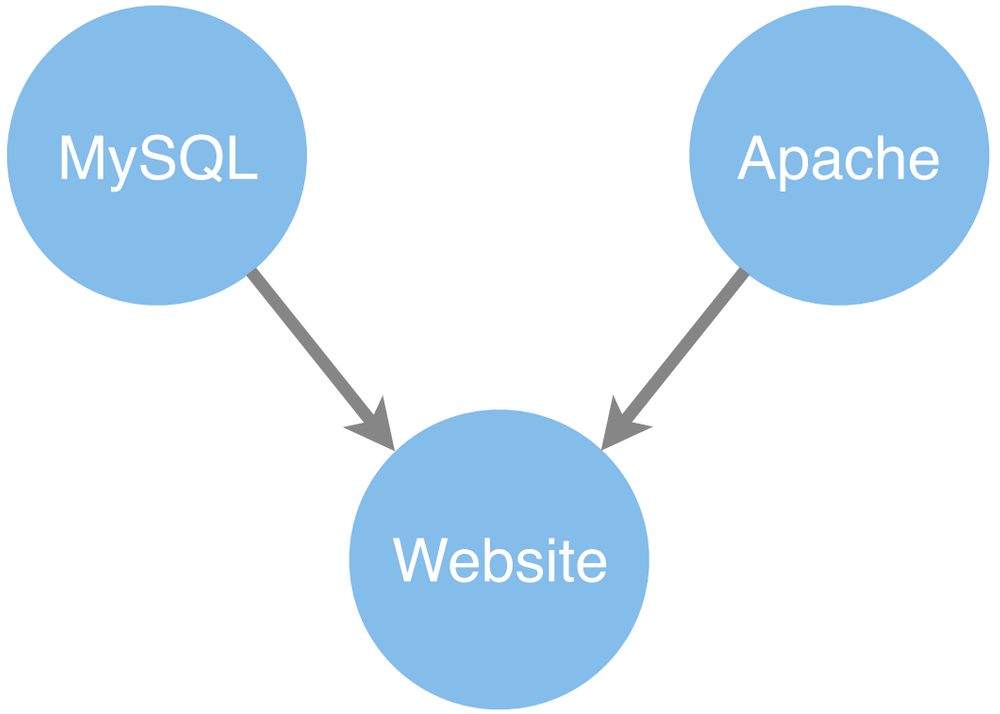
$$p(s, b) = \sum_g p(s|g)p(b|g)p(g)$$

$$p(s, b|g) = p(s|g)p(b|g)$$

… some Web 2.0 service

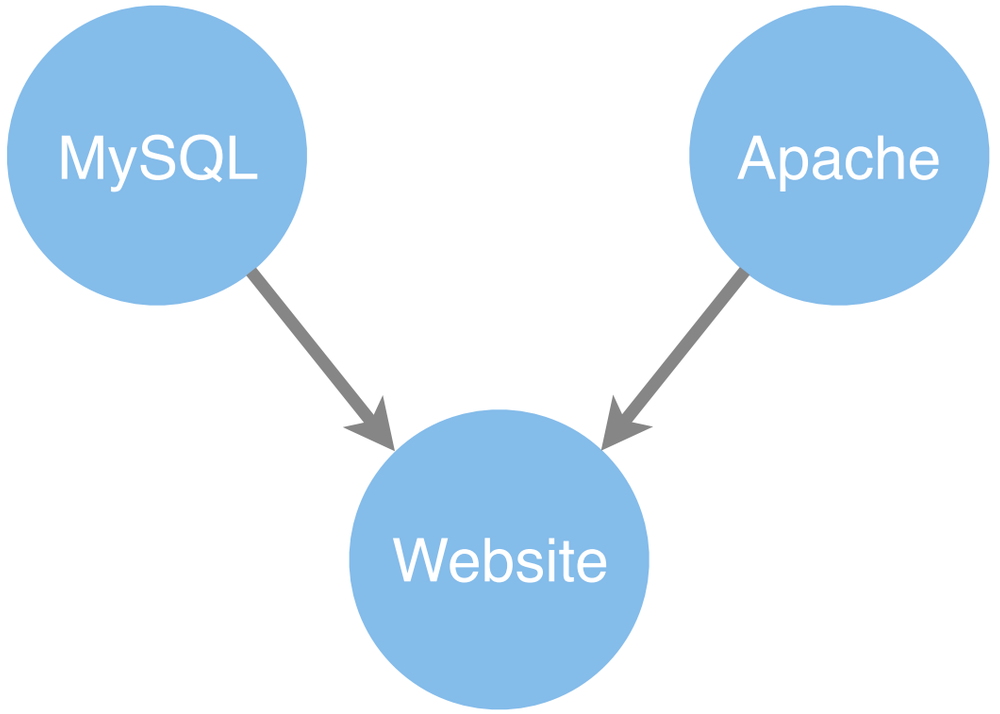


… some Web 2.0 service



- Joint distribution (assume a and m are independent)

… some Web 2.0 service



- Joint distribution (assume a and m are independent)

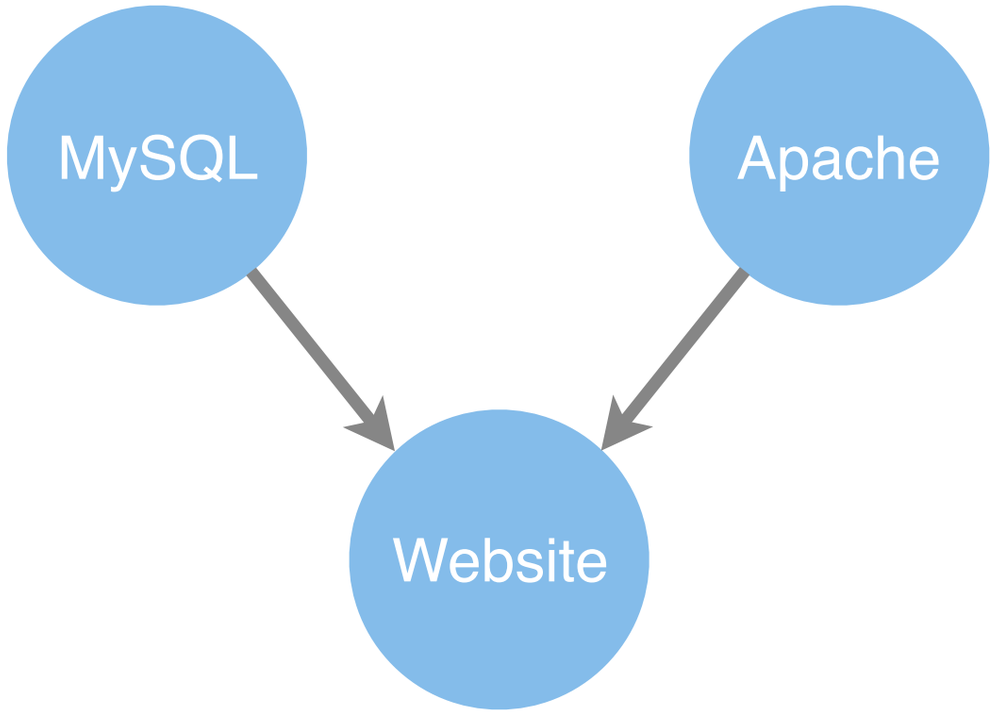
$$p(m, a, w) = p(w|m, a)p(m)p(a)$$

- Explaining away

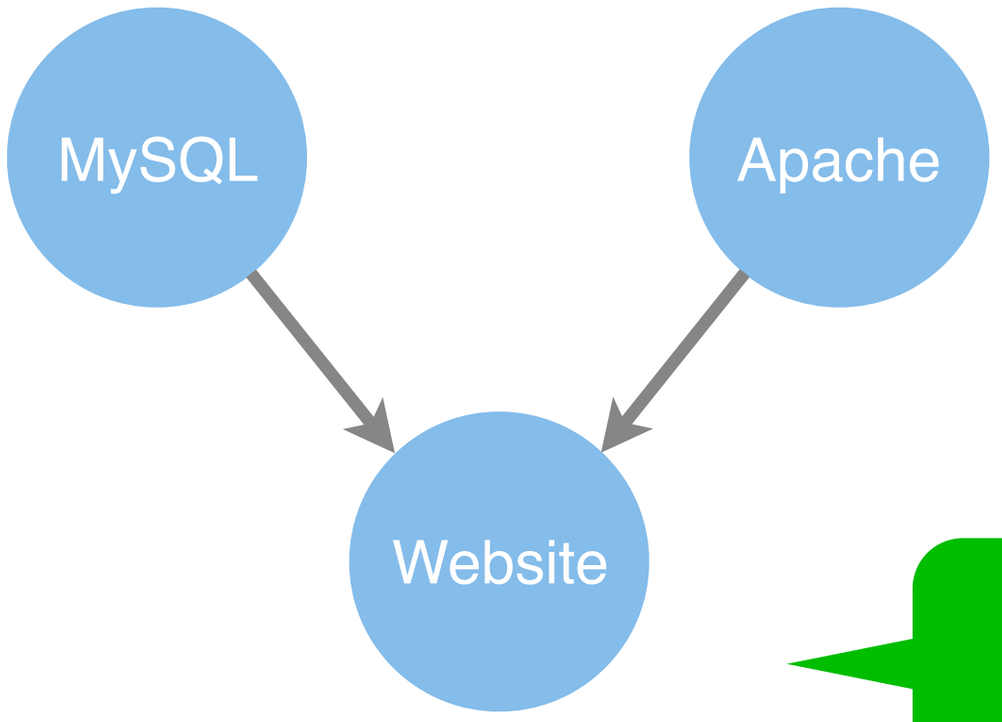
$$p(m, a|w) = \frac{p(w|m, a)p(m)p(a)}{\sum_{m',a'} p(w|m', a')p(m')p(a')}$$

a and m are dependent conditioned on w

… some Web 2.0 service

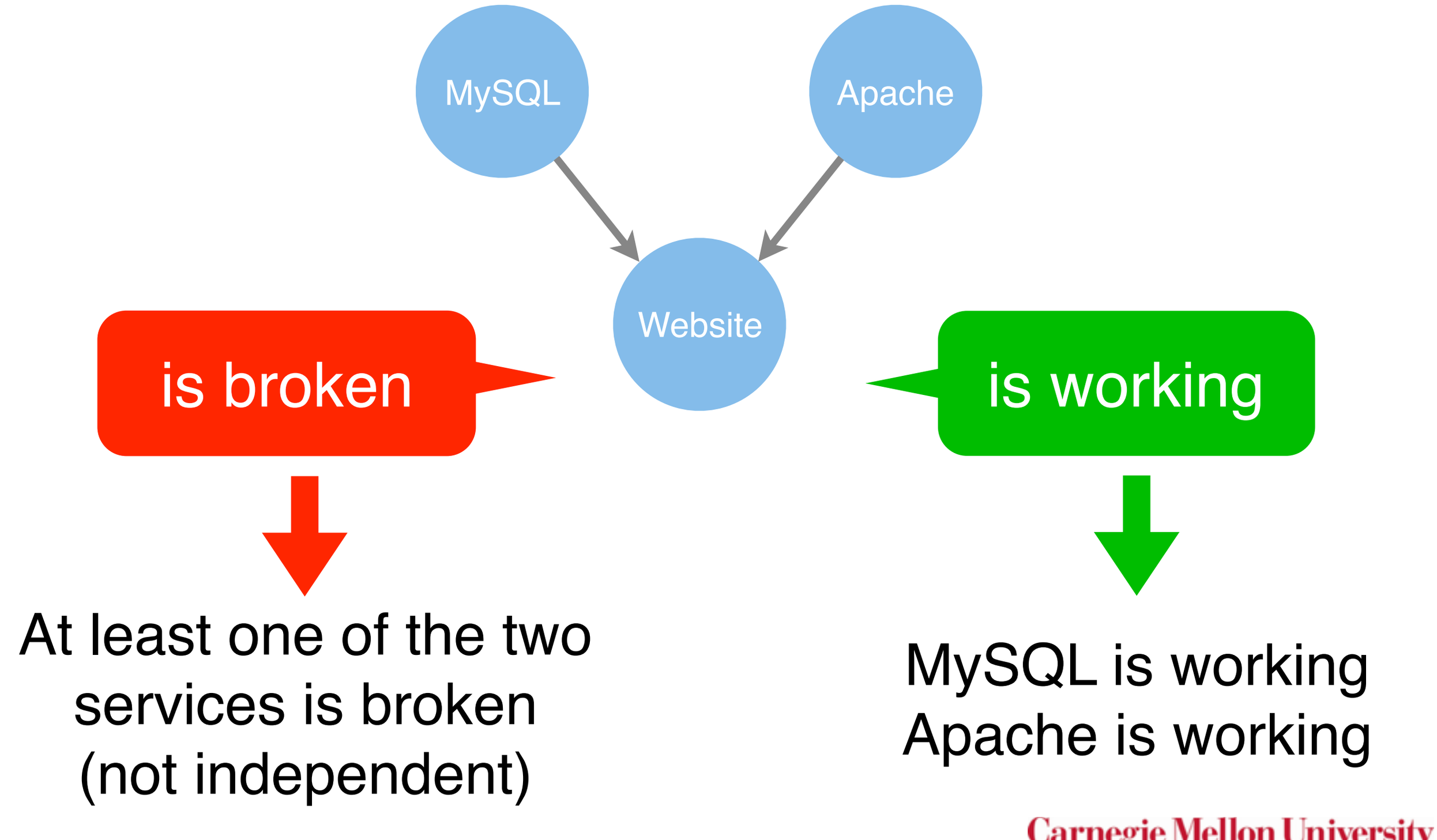


… some Web 2.0 service

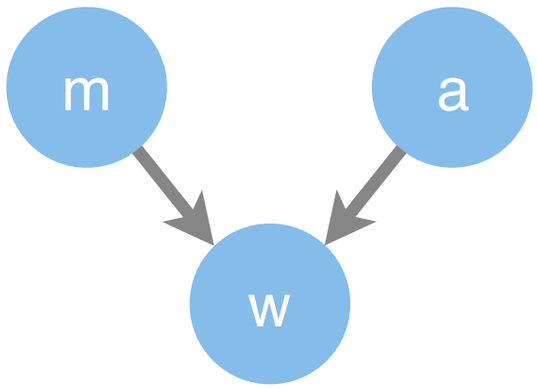


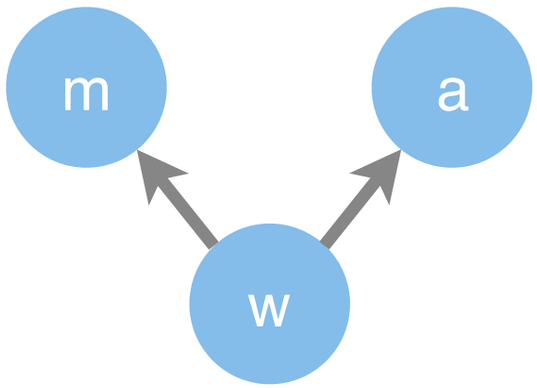
MySQL is working
Apache is working

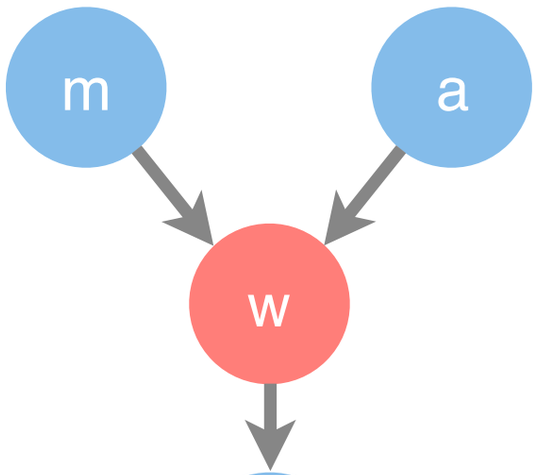
… some Web 2.0 service



Directed graphical model

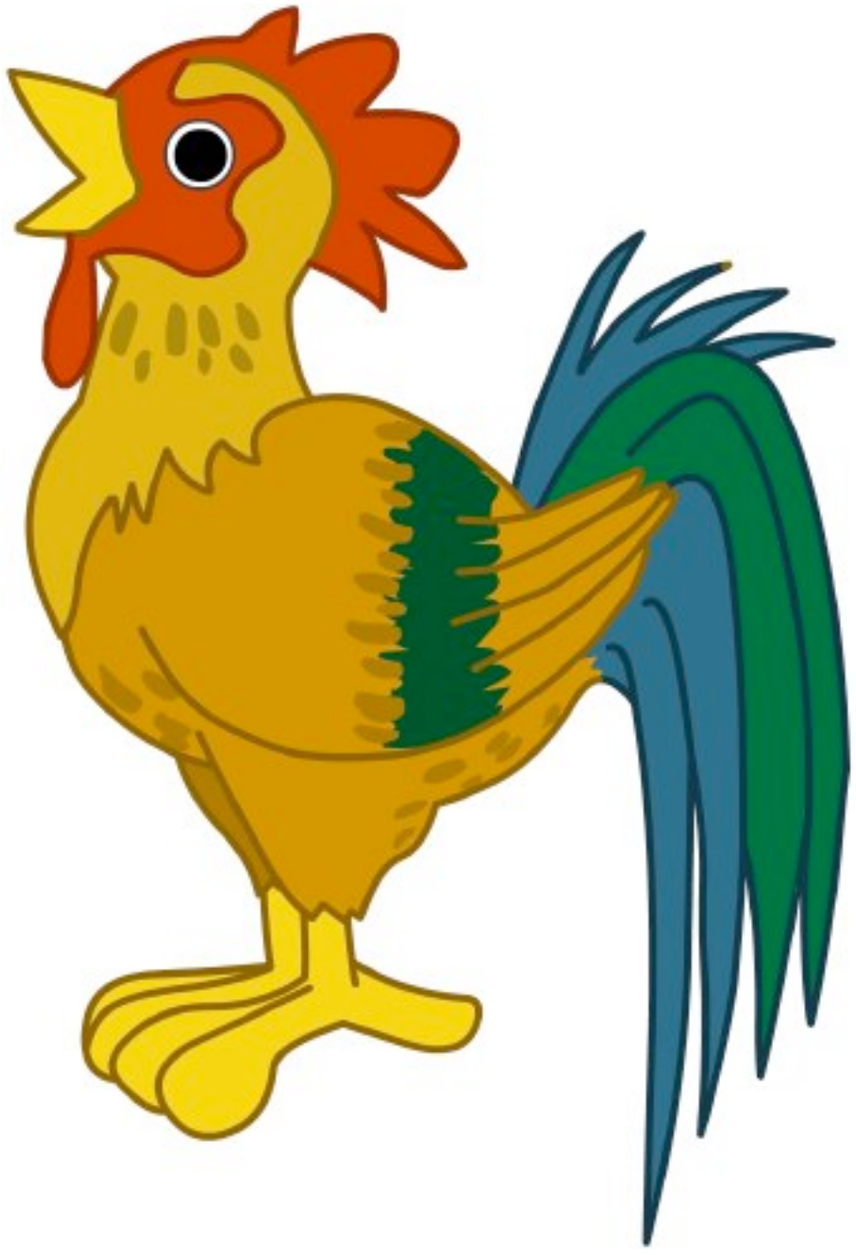




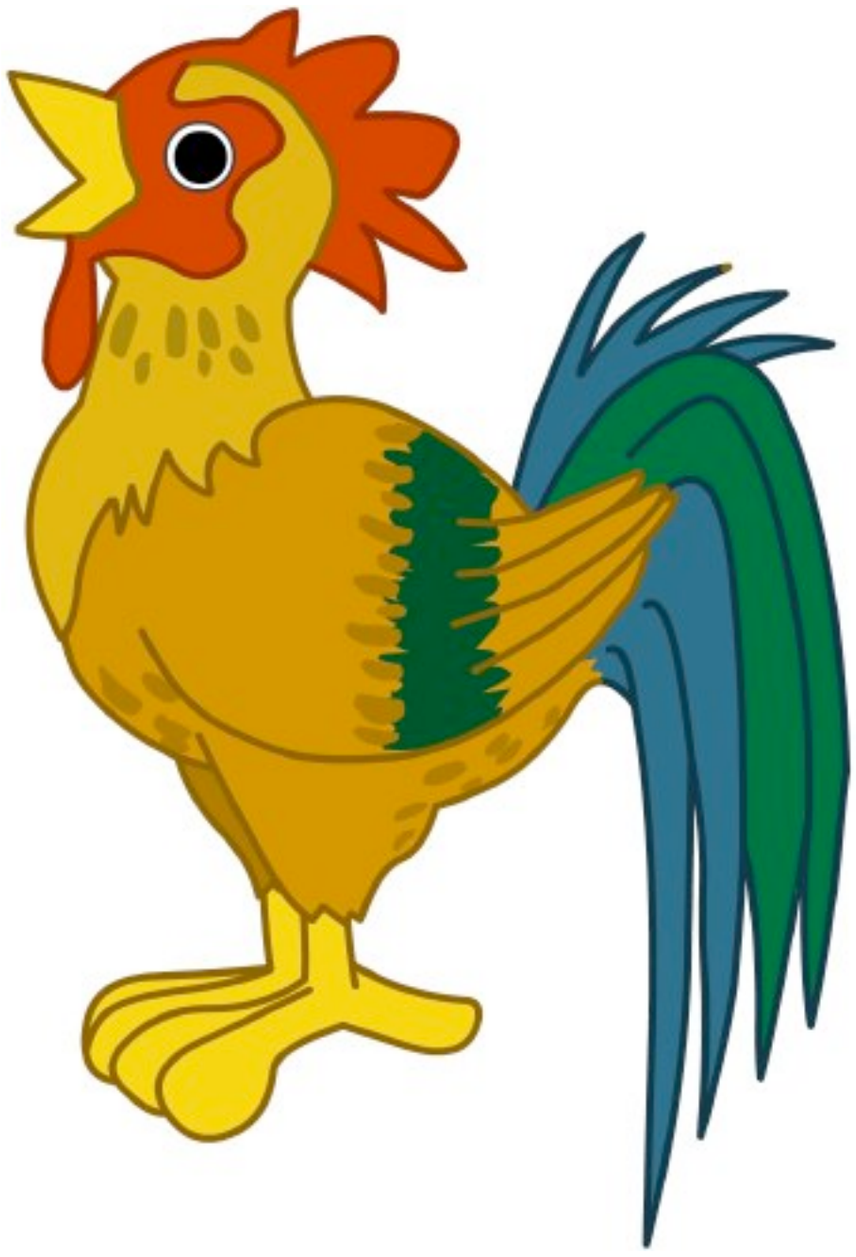


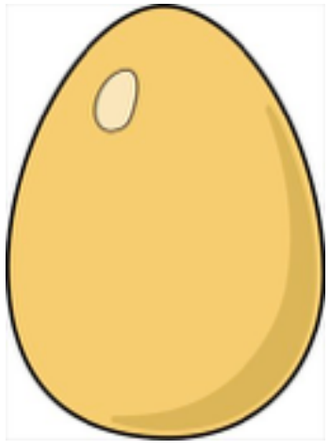
- Easier estimation
  - 15 parameters for full joint distribution
  - 1+1+4+1 for factorizing distribution
- Causal relations
- Inference for unobserved variables

No loops allowed

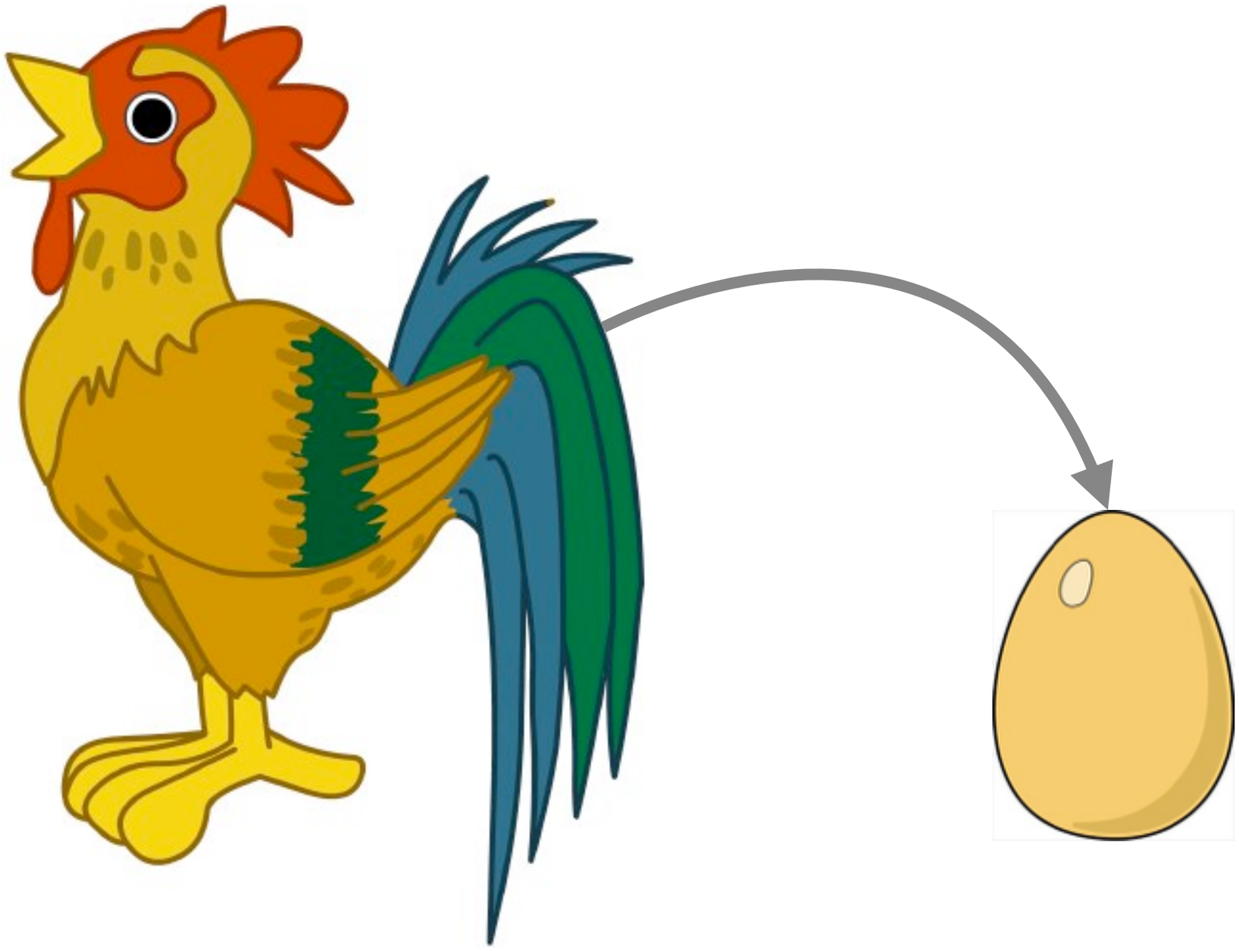


No loops allowed

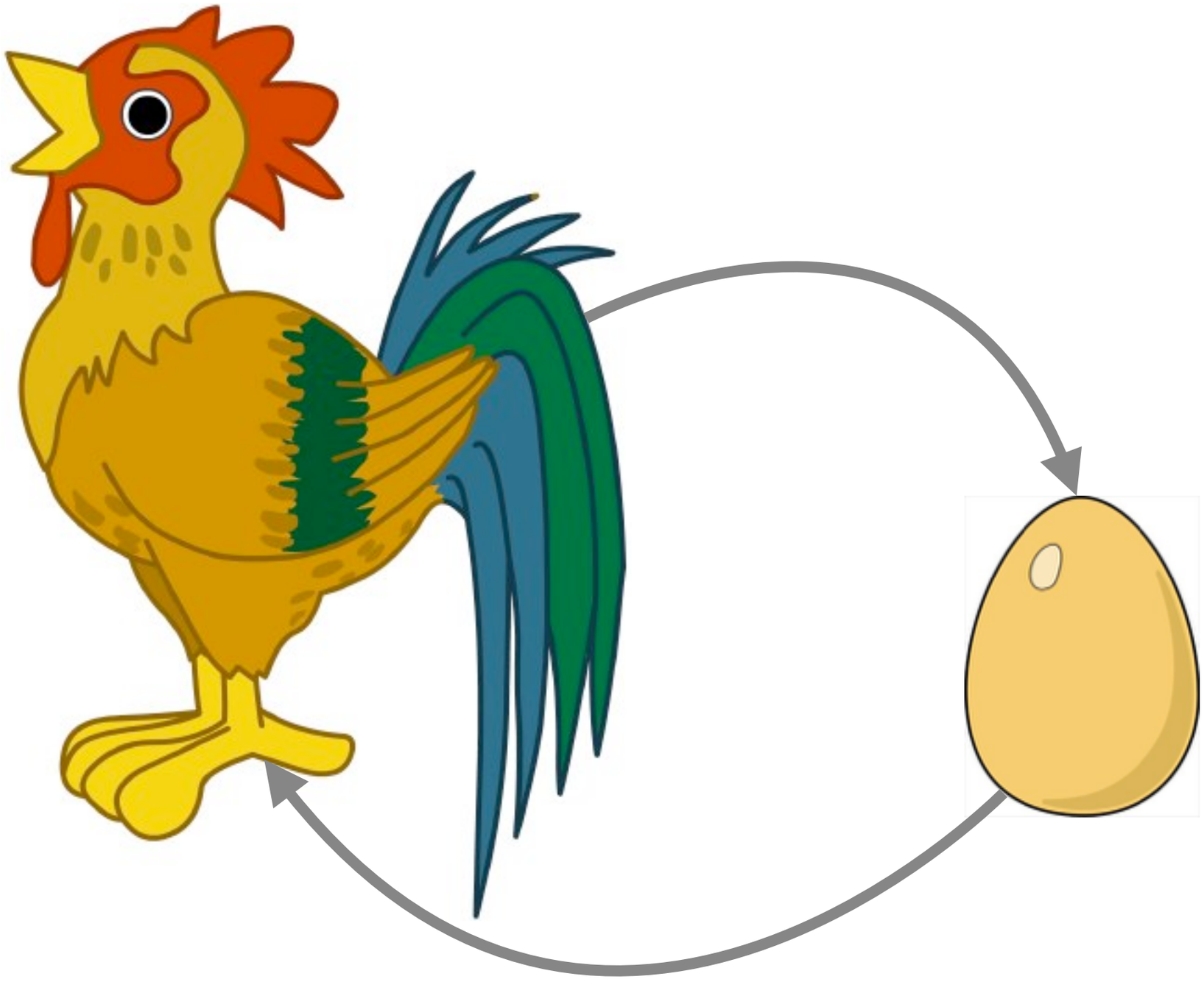




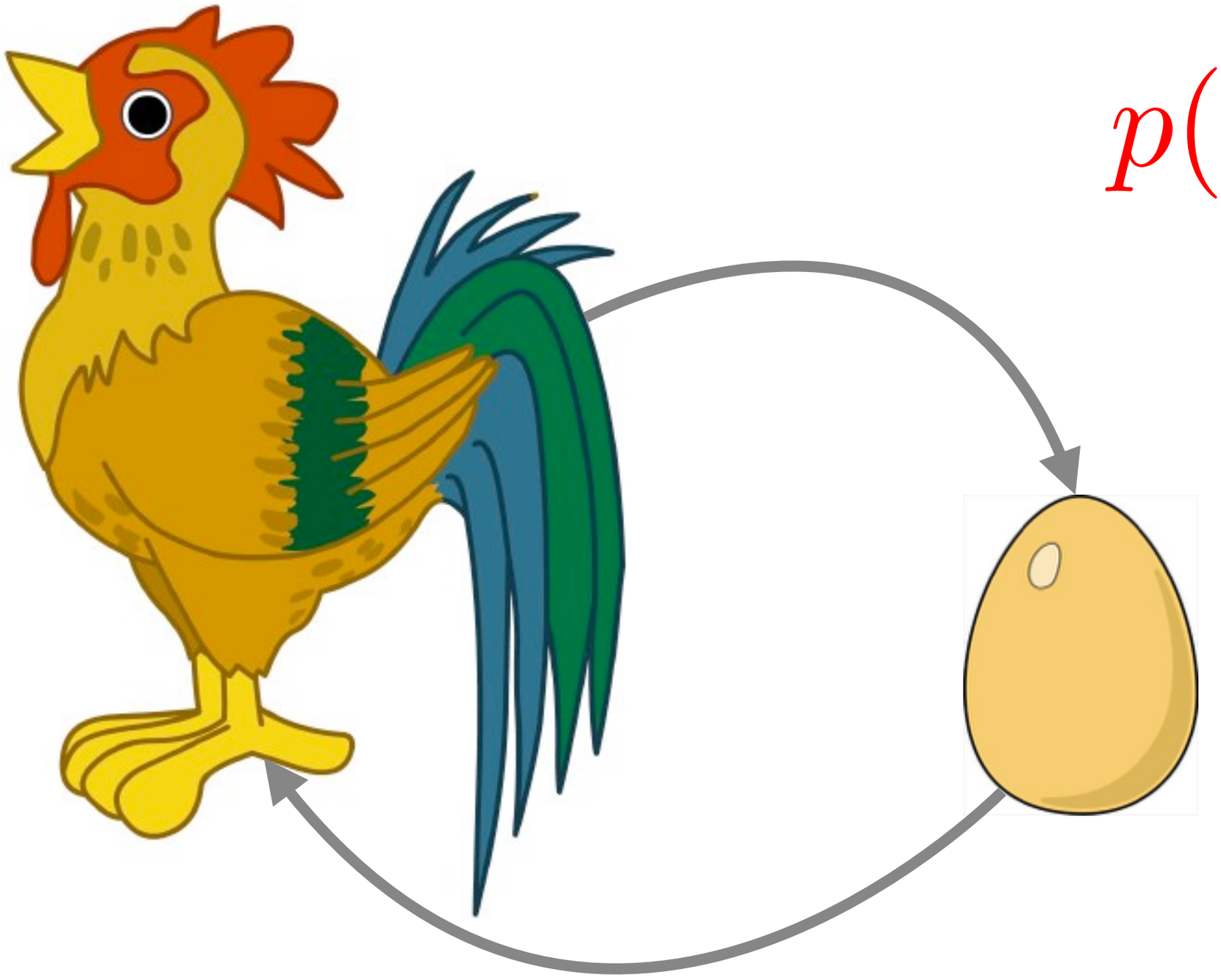
No loops allowed



No loops allowed

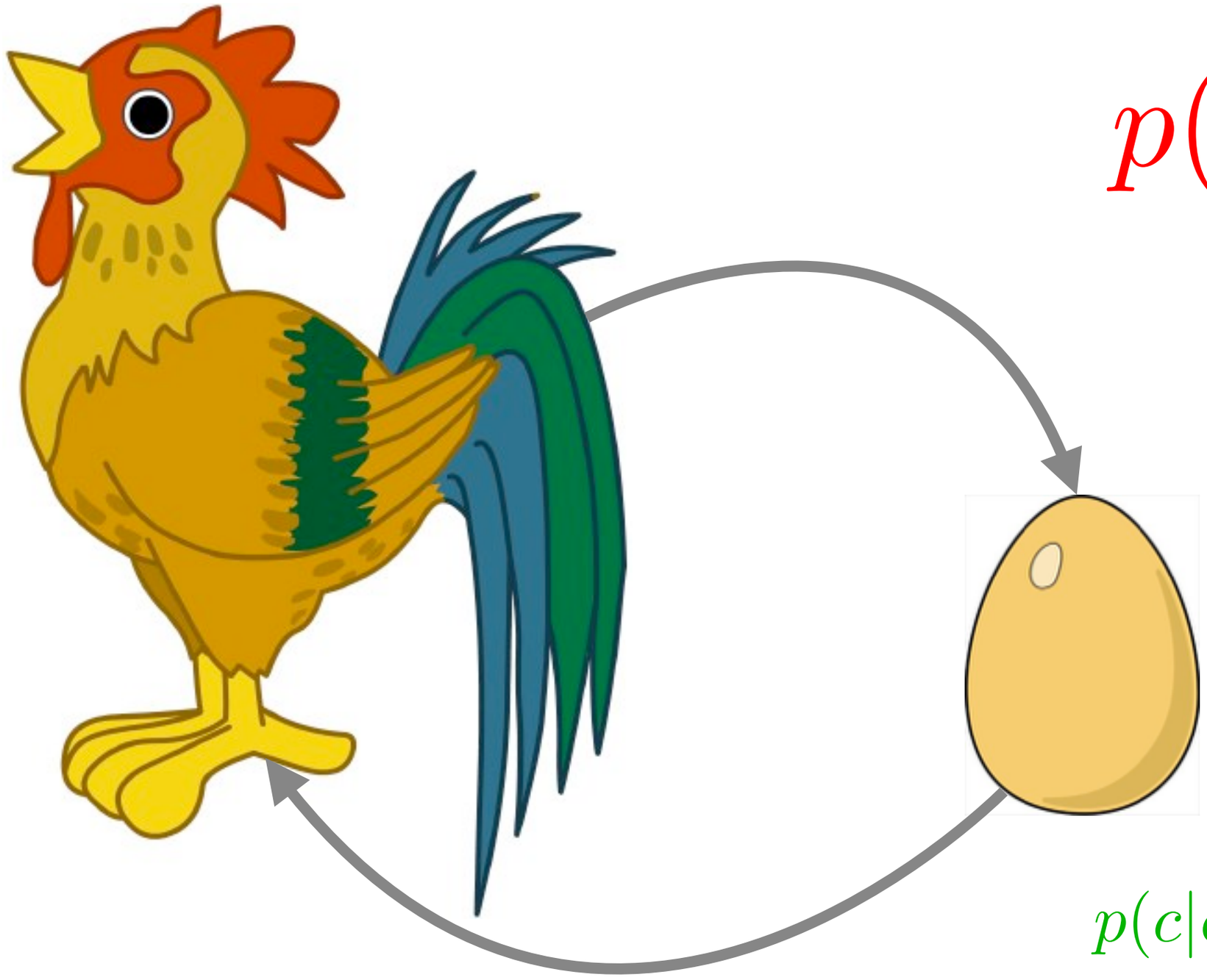


No loops allowed



$$p(c|e)p(e|c)$$

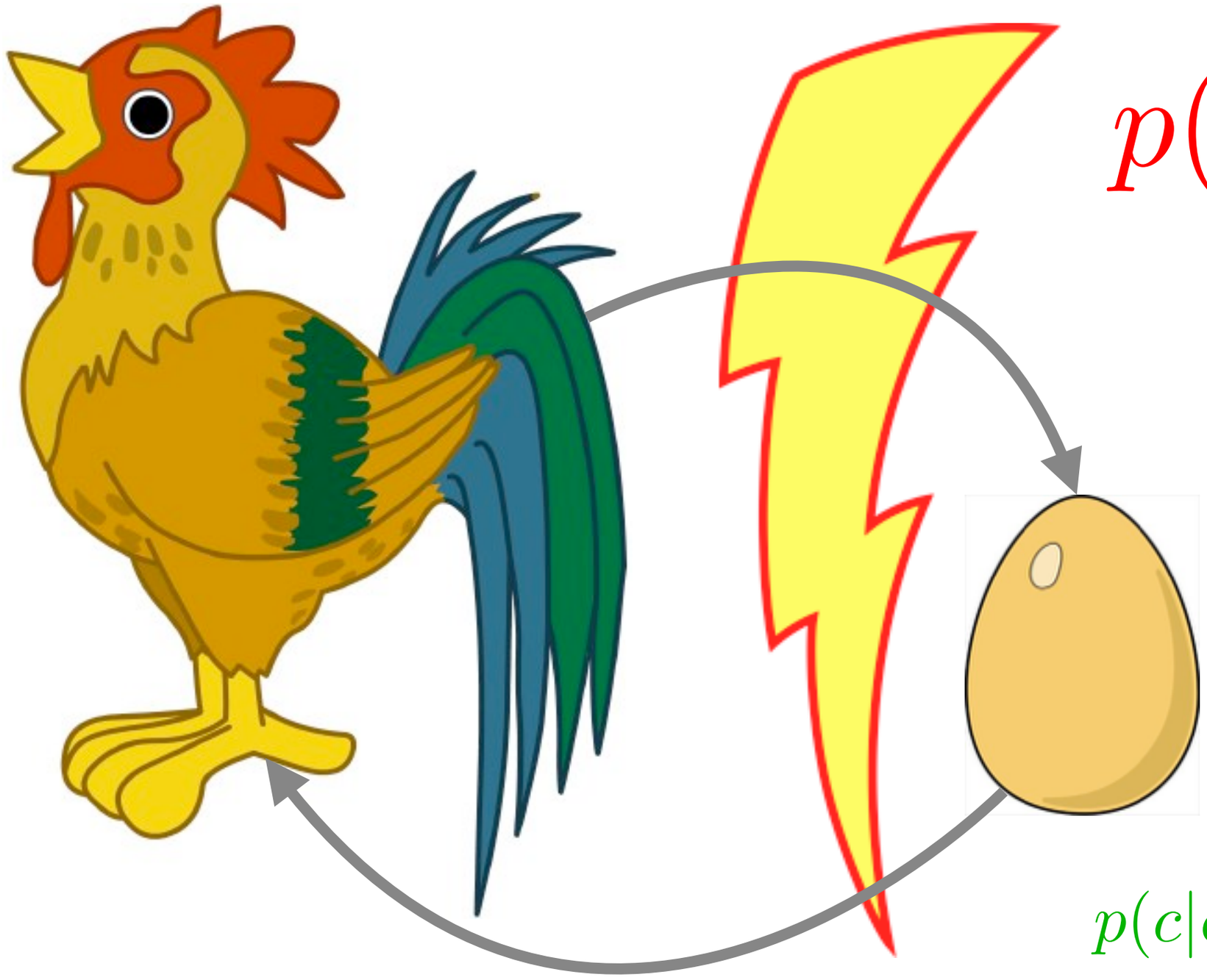
No loops allowed



$$p(c|e)p(e|c)$$

$$p(c|e)p(e) \text{ or } p(e|c)p(c)$$

No loops allowed

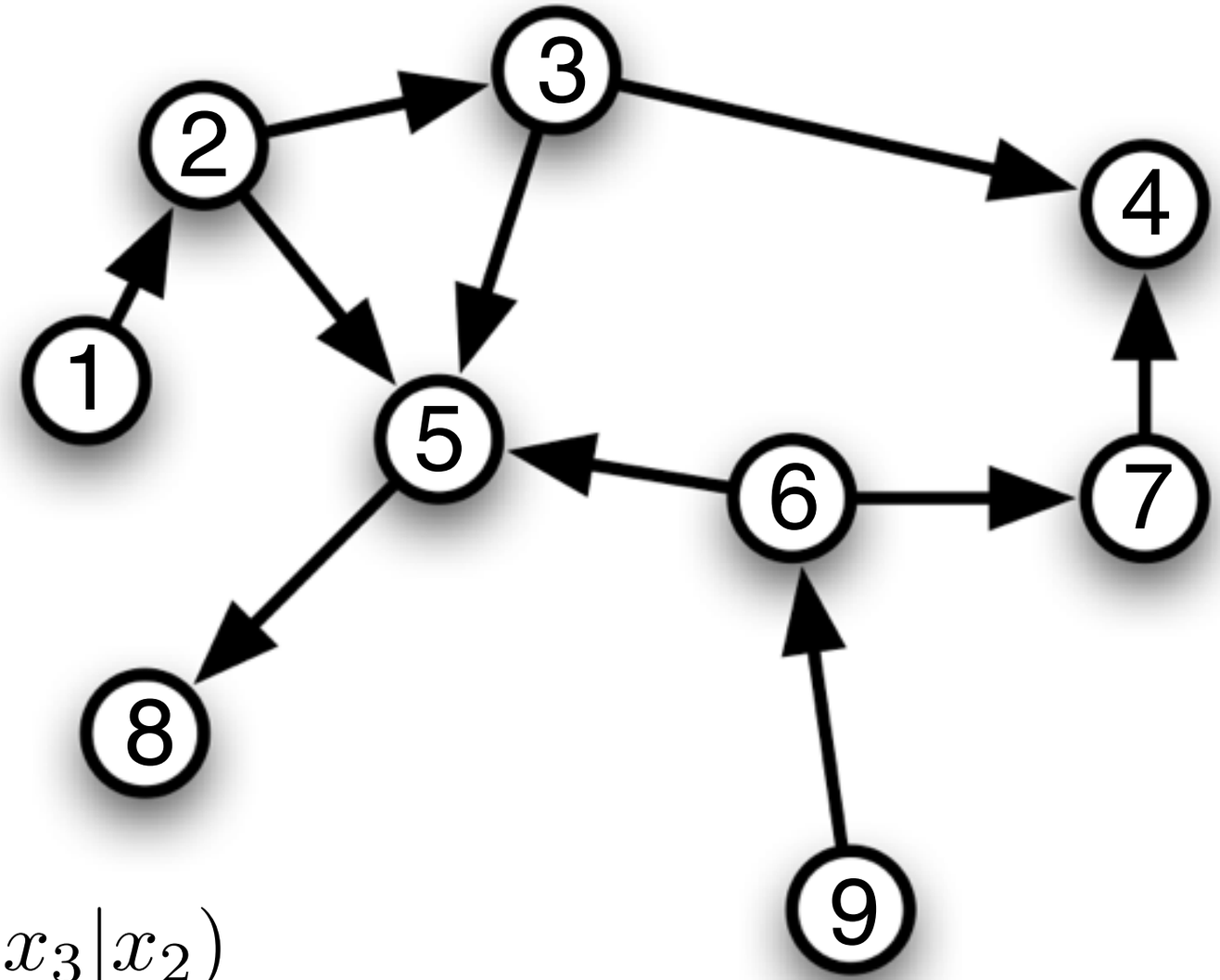


$$p(c|e)p(e|c)$$

$$p(c|e)p(e) \text{ or } p(e|c)p(c)$$

Directed Graphical Model

- Probability distribution
- Iterate over children|parents

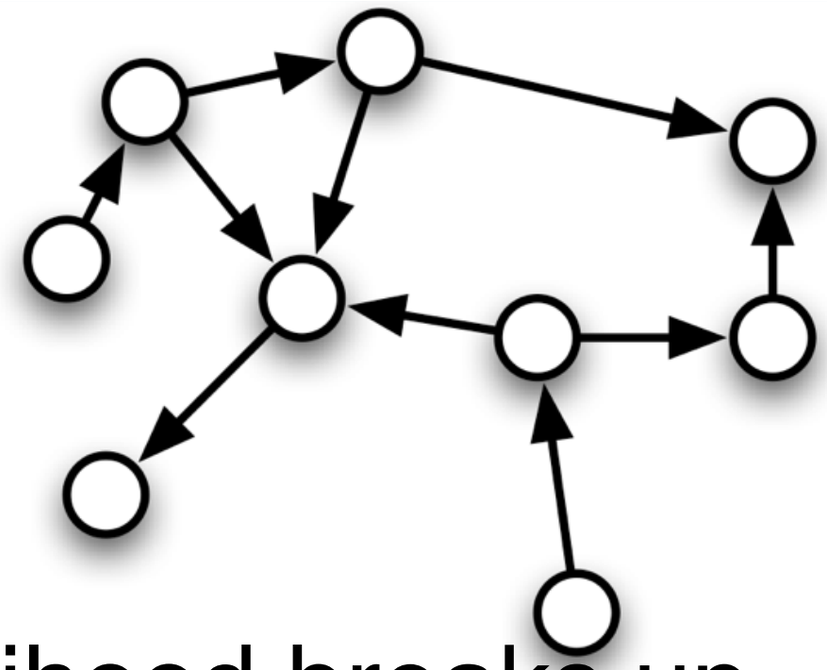


$$p(x) = p(x_1)p(x_2|x_1)p(x_3|x_2)$$
$$p(x_4|x_3, x_7)p(x_5|x_2, x_3, x_6)$$
$$p(x_6|x_9)p(x_7|x_6)p(x_8|x_5)p(x_9)$$

Directed Graphical Model

- Joint probability distribution

$$p(x) = \prod_i p(x_i | x_{\text{parents(i)}})$$



- Parameter estimation
- If x is fully observed the likelihood breaks up

$$\log p(x|\theta) = \sum_i \log p(x_i | x_{\text{parents(i)}}, \theta)$$

- Estimation is trivial. All terms decompose

$$\underset{\theta_i}{\text{minimize}} - \log p(x_i | x_{\text{parents}(i)}, \theta_i) - \log p(\theta_i)$$

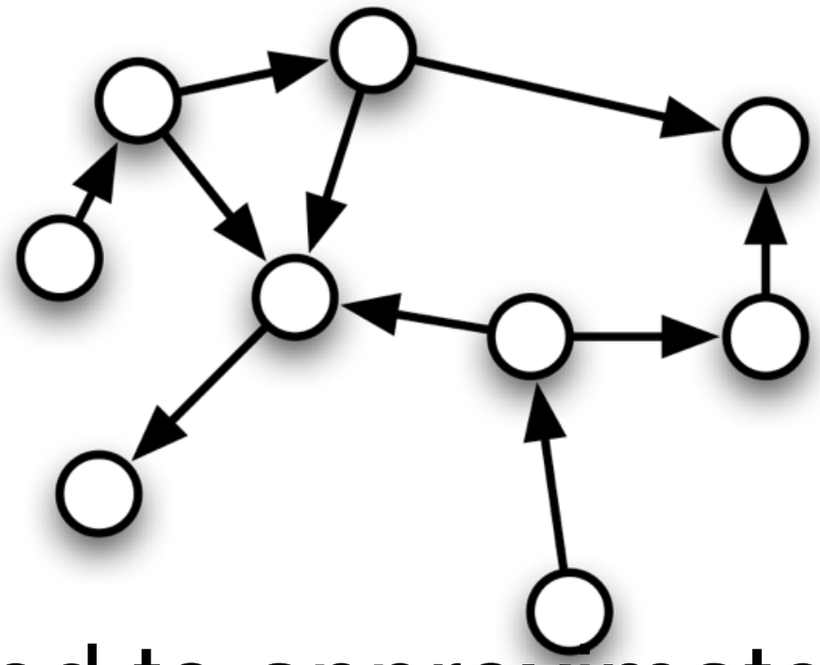
Approximate Inference

… don't worry, there's math why this works …

- Joint probability distribution

$$p(x) = \prod_i p(x_i | x_{\text{parents(i)}})$$

- EM Parameter estimation



- If x is partly observed we need to approximate
- Intuition - use best guess of missing variables

$$q(x_{\text{missing}}) = p(x_{\text{missing}} | x_{\text{observed}})$$

- Estimation is possible (M step)

$$\underset{\theta_i}{\text{minimize}}\, \mathbf{E}_{x_{\text{missing}} \sim q} \left[-\log p(x_i | x_{\text{parents}(i)}, \theta_i) \right] - \log p(\theta_i)$$

Summary

- Directed graphical models

$$p(x) = \prod_i p(x_i | x_{\text{parents(i)}})$$

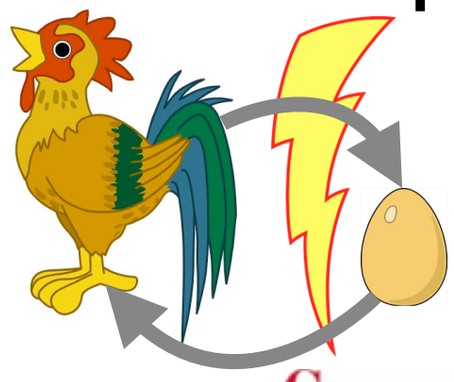
- Explaining away

  Independent variables become dependent conditioned on a joint child.

- Observing yields independence

  Observed parent makes children independent

- No loops in graph allowed



Dependence

1 Chain

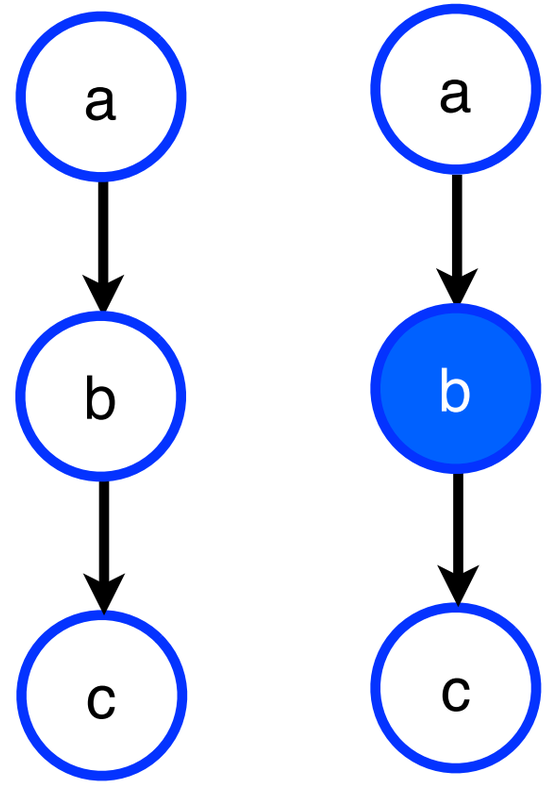
- Joint distribution

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

- Conditioning on b

$$p(a, c|b) = \frac{p(a)p(b|a)p(c|b)}{\sum_{a',c'} p(a')p(b|a')p(c'|b)}$$

$$= \frac{p(a)p(b|a)}{\sum_{a'} p(a')p(b|a')} \frac{p(c|b)}{\sum_{c'} p(c'|b)}$$



- Conditional independence

$$a \perp c|b$$

2 Common Cause

- Joint distribution

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

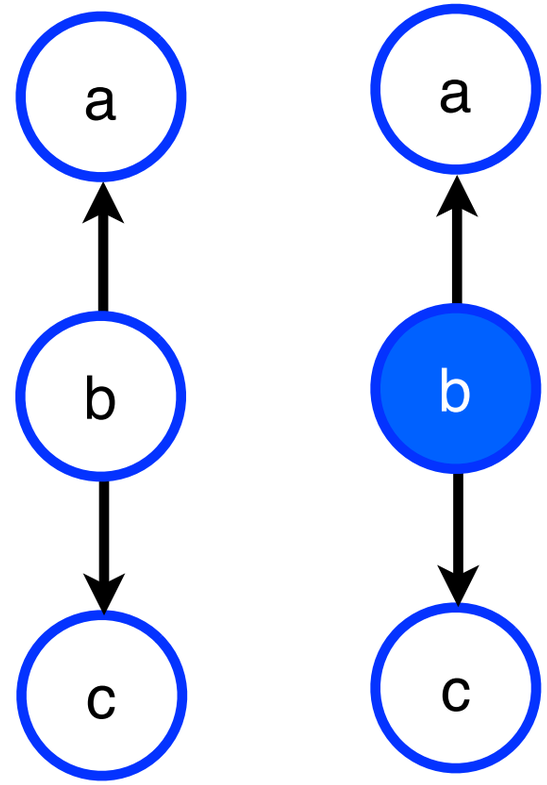
- a and c are dependent

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

- Conditioning on b creates independence

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

$$a \perp c|b$$



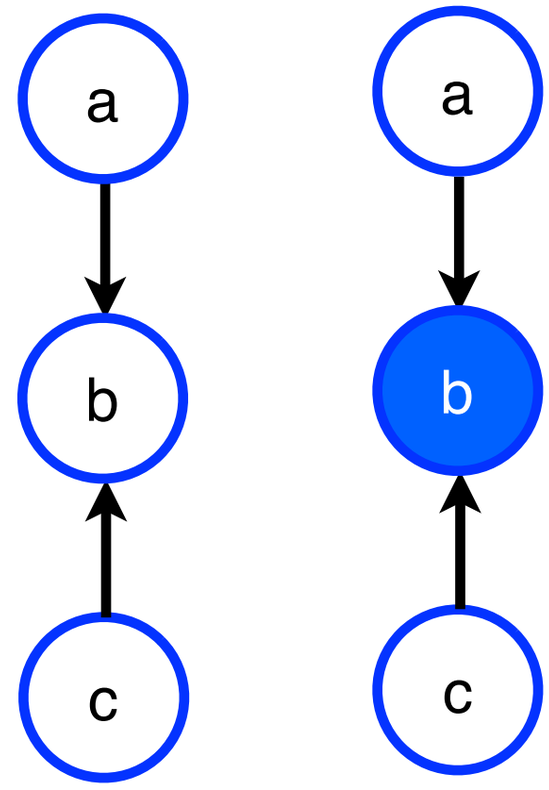
3 Explaining Away

- Joint distribution

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

- a and c are independent
- Conditioning on b creates dependence

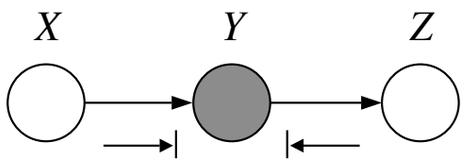
$$p(a, c|b) = \frac{p(a)p(b|a, c)p(c)}{\sum_{a',c'} p(a')p(b|a', c')p(c')}$$



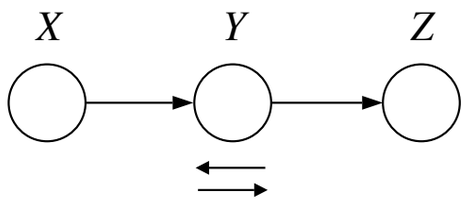
d-Separation

- Given general directed acyclic graph (DAG)
- Determine whether sets A, B of random variables are conditionally independent given C
- Simple algorithm - reachability
  - Start in in vertex of A
  - Check whether any vertex in B can be reached
  - If separated, we have conditional independence

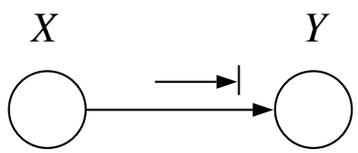
Transition rules



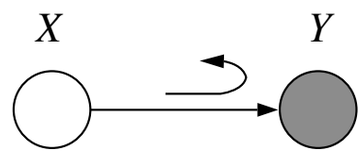
(a)



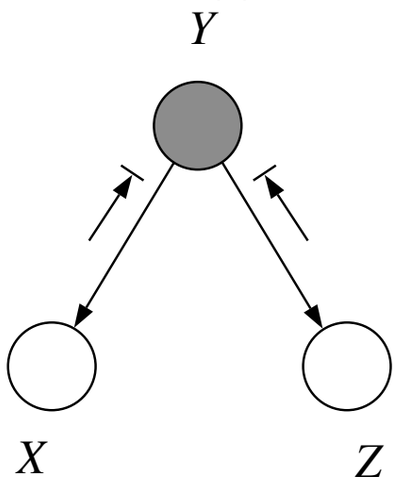
(b)



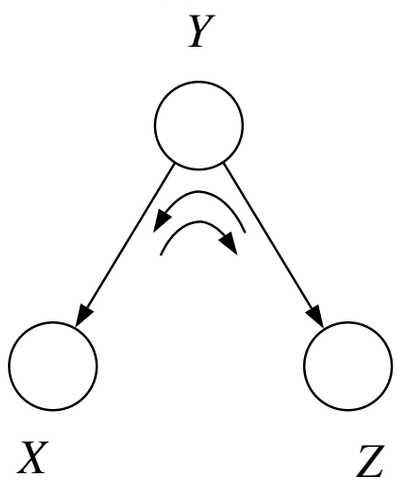
(a)



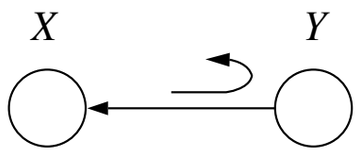
(b)



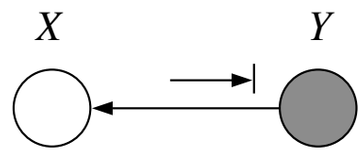
(a)



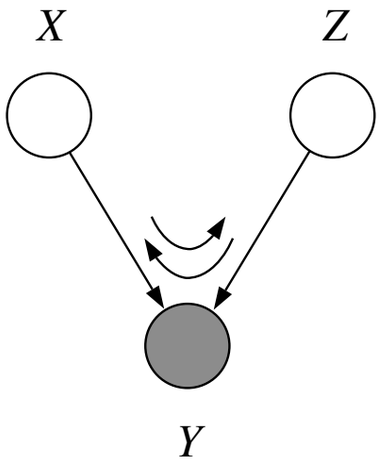
(b)



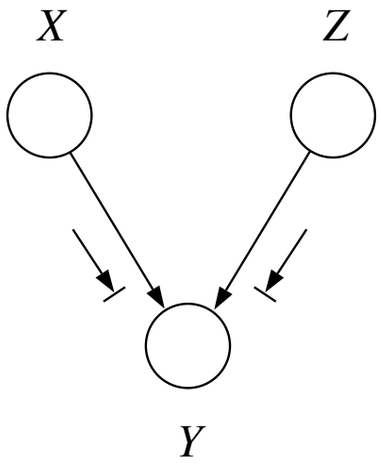
(a)



(b)



(a)

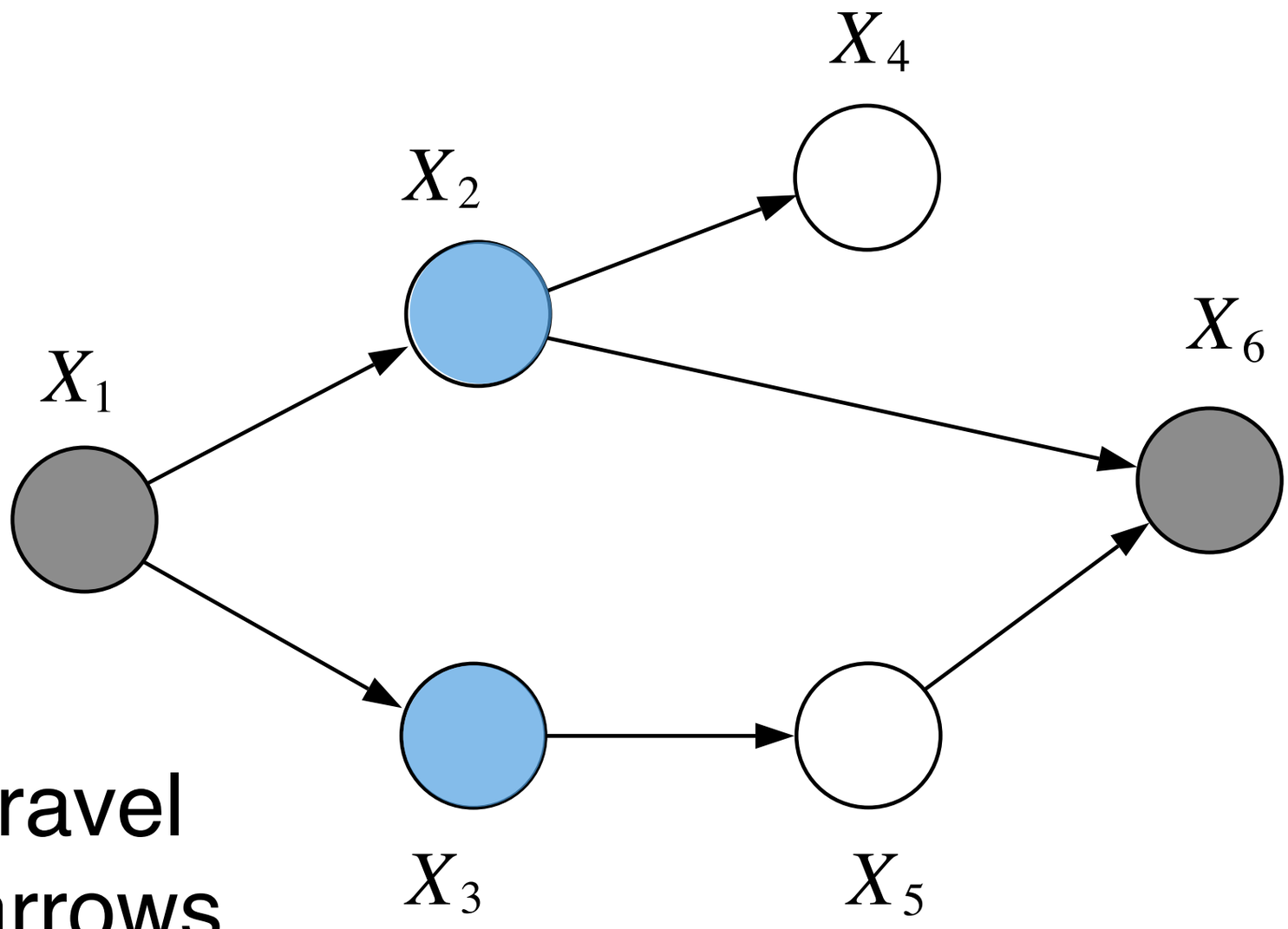


(b)

Courtesy of Sam Roweis

Transition rules

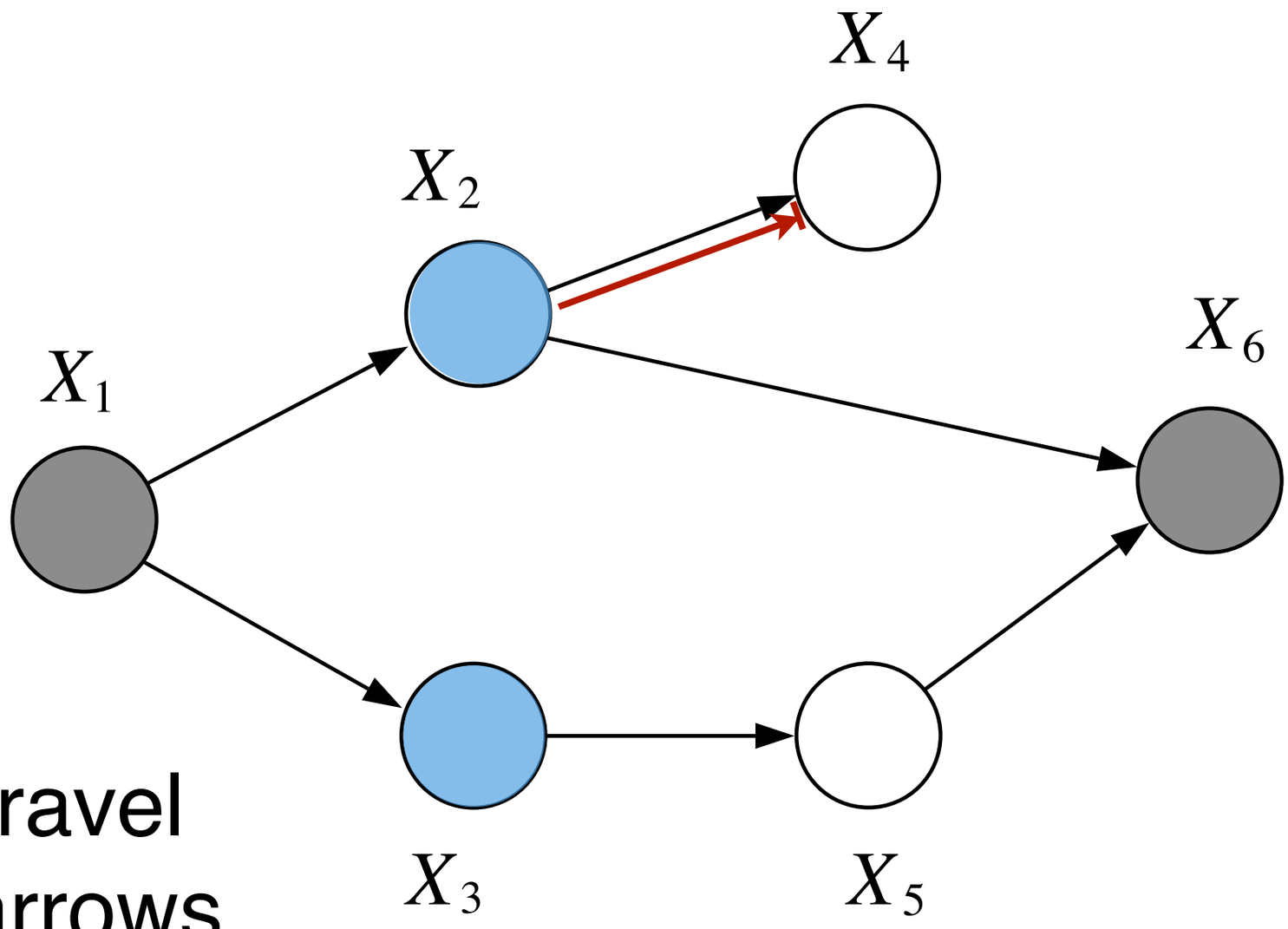
$$\mathbf{x}_2 \perp \mathbf{x}_3 | \{\mathbf{x}_1, \mathbf{x}_6\} \quad ?$$



ball can travel opposite arrows

Transition rules

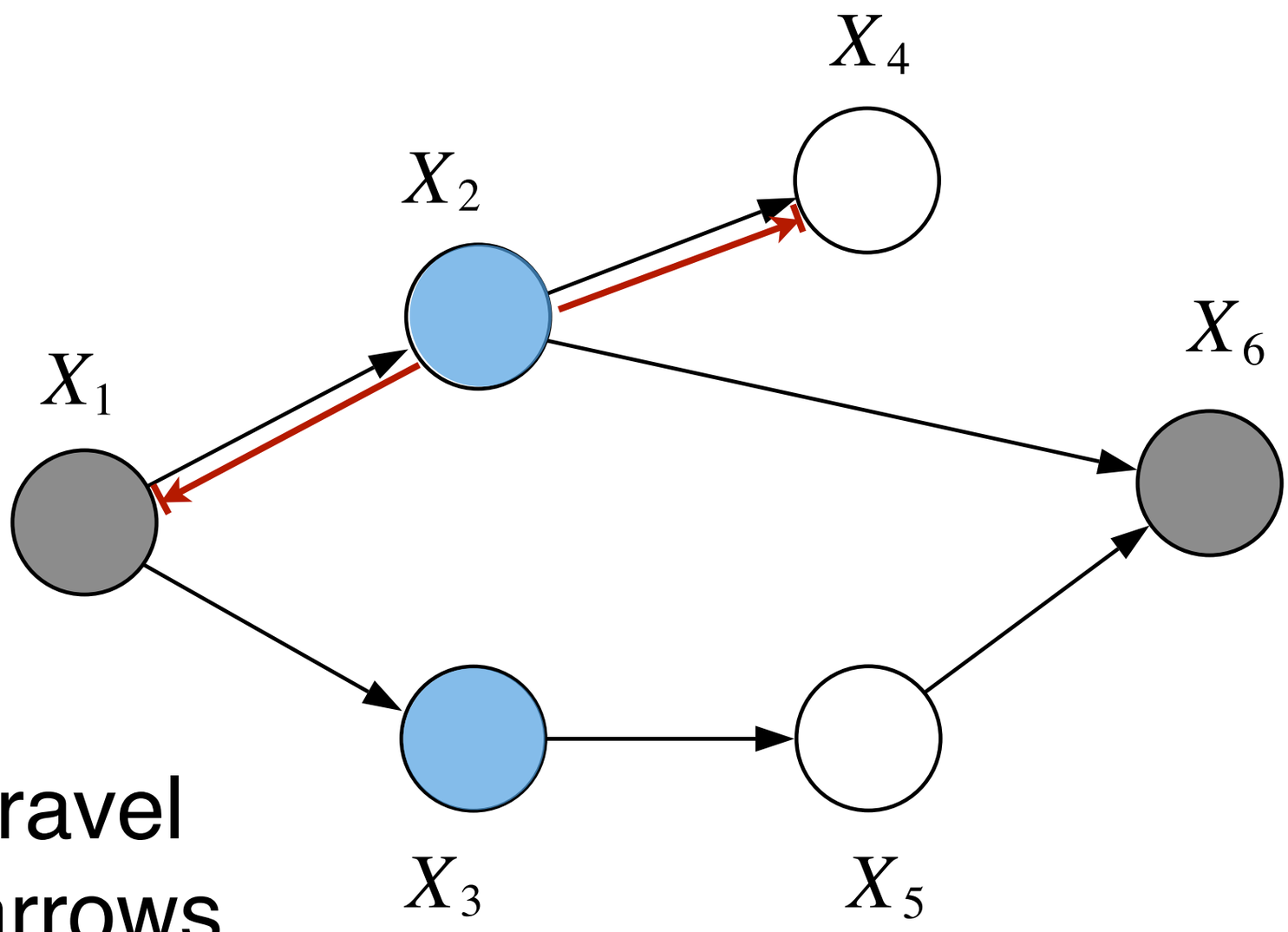
$$\mathbf{x}_2 \perp \mathbf{x}_3 | \{\mathbf{x}_1, \mathbf{x}_6\} \quad ?$$



ball can travel opposite arrows

Transition rules

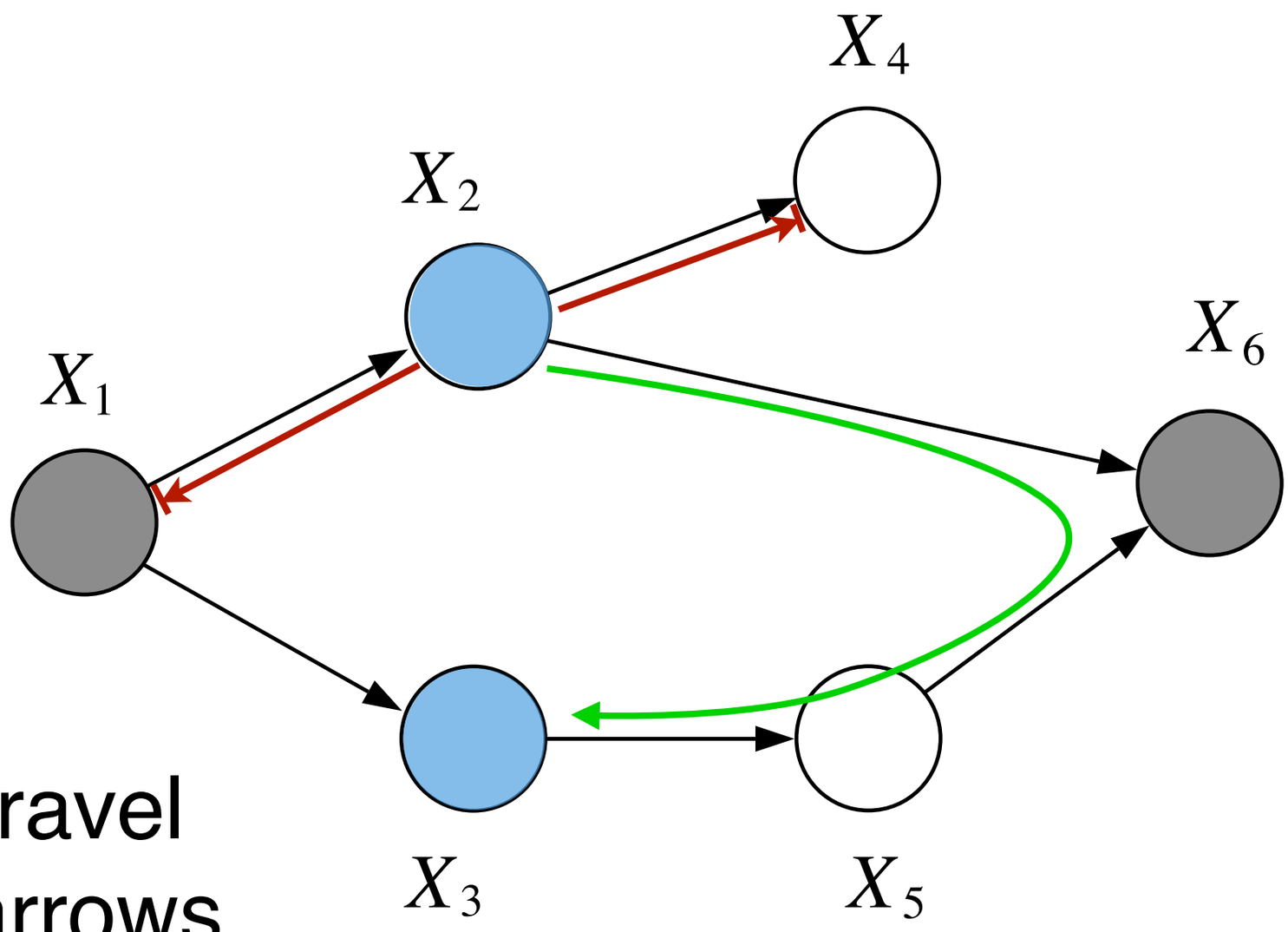
$$\mathbf{x}_2 \perp \mathbf{x}_3 | \{\mathbf{x}_1, \mathbf{x}_6\} \quad ?$$



ball can travel opposite arrows

Transition rules

$$\mathbf{x}_2 \perp \mathbf{x}_3 | \{\mathbf{x}_1, \mathbf{x}_6\} \quad ?$$

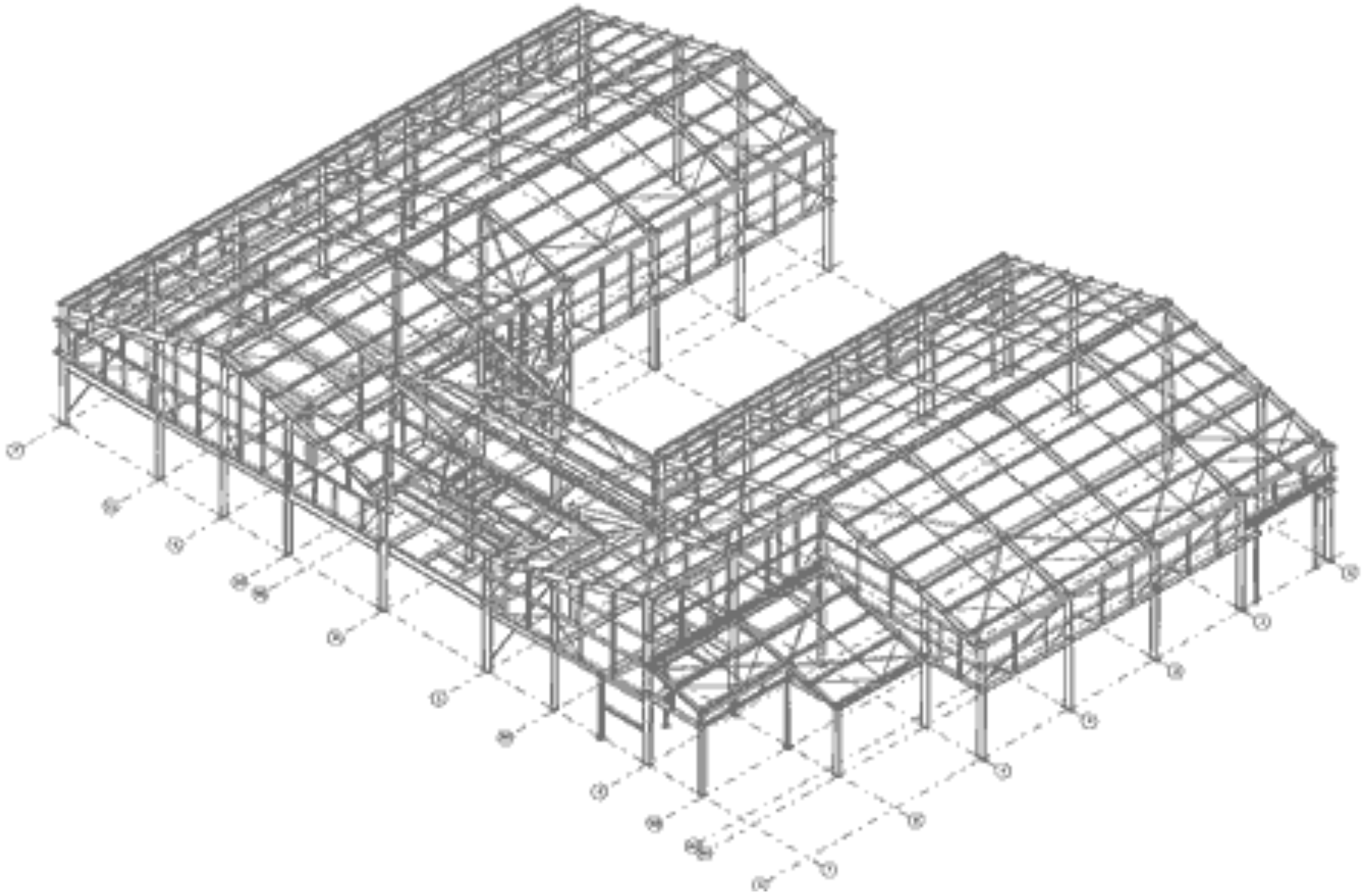


ball can travel opposite arrows

Summary

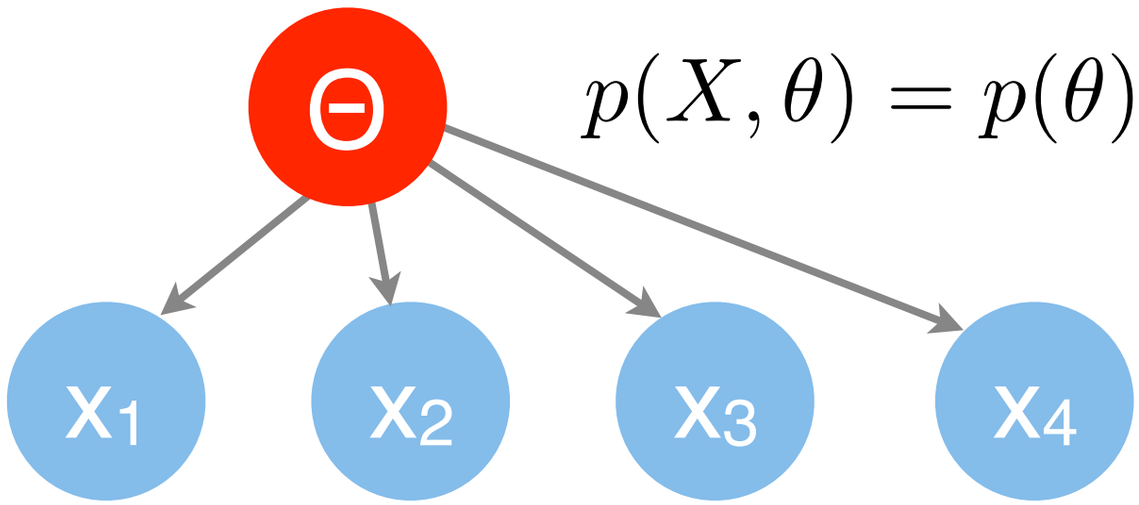
- Dependent random variables
- Observing can make things dependent or independent
- Conditional independence simplifies model
- Bayes ball to check properties
  - Chains (observing stops dependence)
  - Common causes (observing stops dependence)
  - Common children (observing creates dependence)

Structures

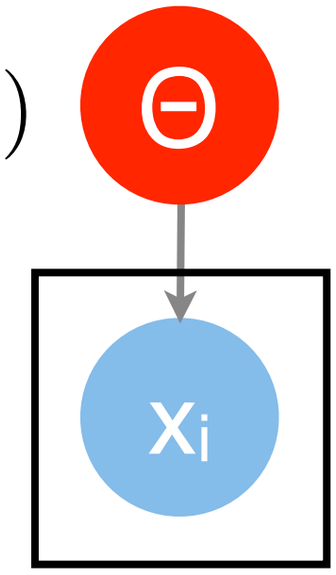


Plates: FOR loops for statisticians

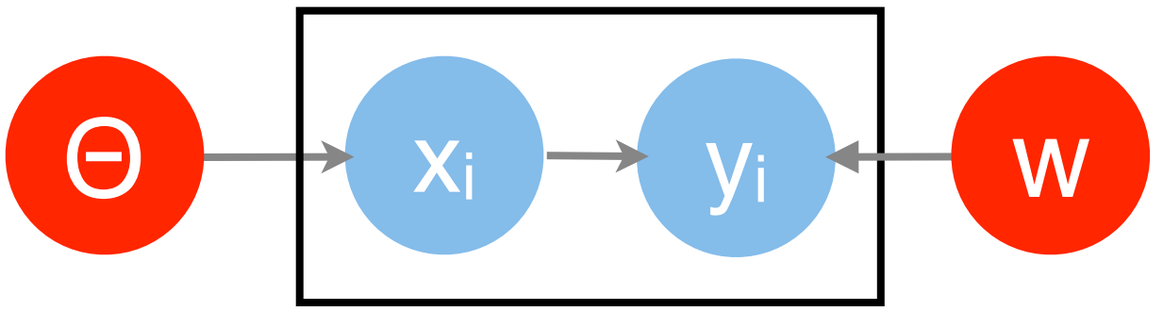
- Repeated dependency structure
  - Modeling iid observations



$$p(X,\theta) = p(\theta)\prod_i p(x_i|\theta)$$



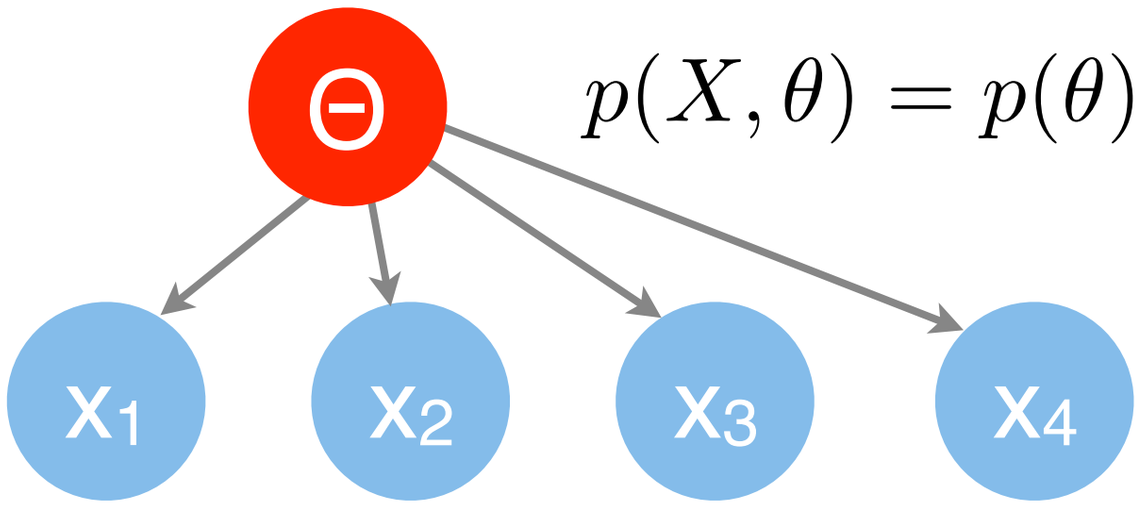
- Supervised learning



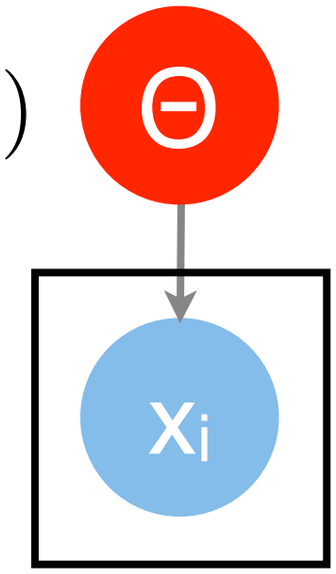
$$p(X,Y,\theta,w)$$
$$=p(\theta)p(w)\prod_i p(x_i|\theta)p(y_i|x_i,w)$$

Plates: FOR loops for statisticians

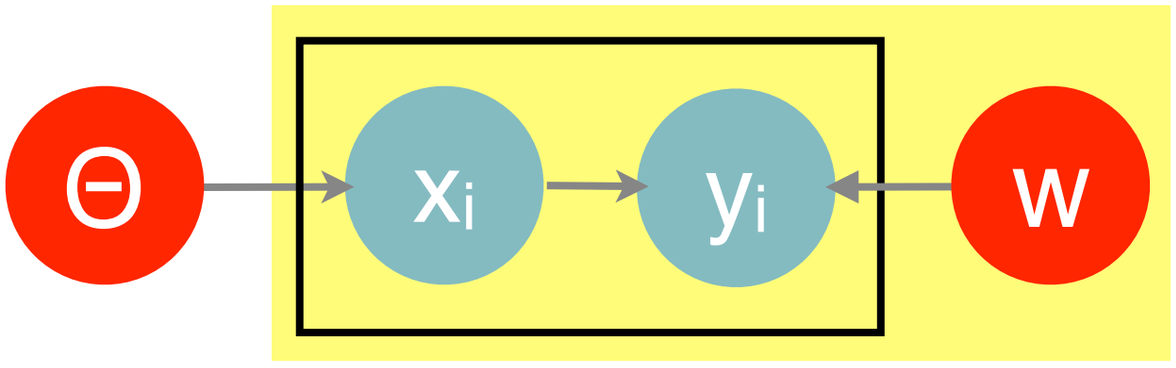
- Repeated dependency structure
  - Modeling iid observations



$$p(X,\theta) = p(\theta)\prod_i p(x_i|\theta)$$



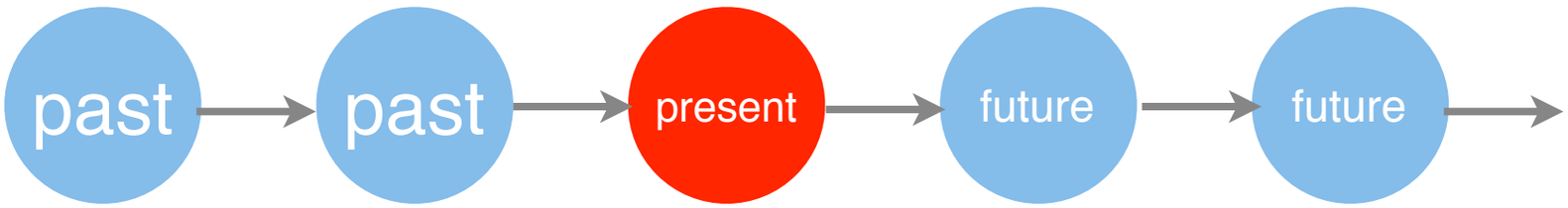
- Supervised learning



$$p(X,Y,\theta,w)$$
$$=p(\theta)p(w)\prod_i p(x_i|\theta)p(y_i|x_i,w)$$

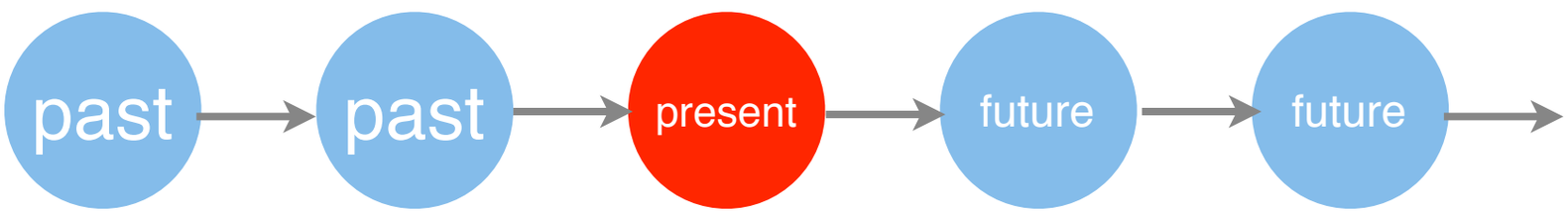
Chains

Markov Chain

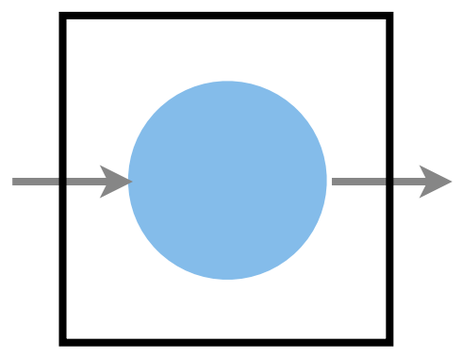


Chains

Markov Chain

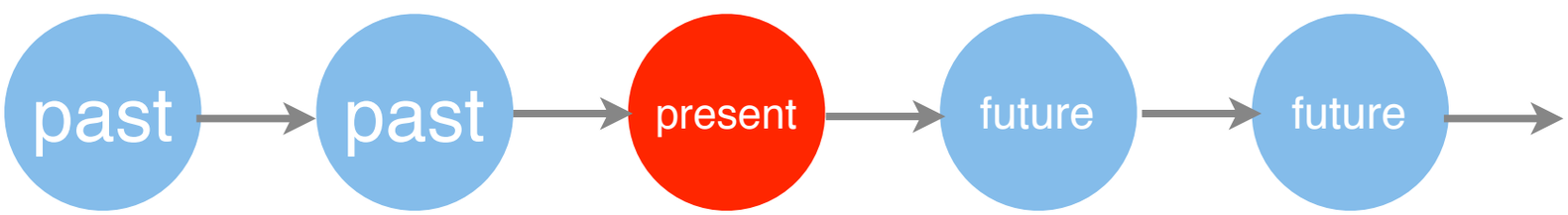


Plate

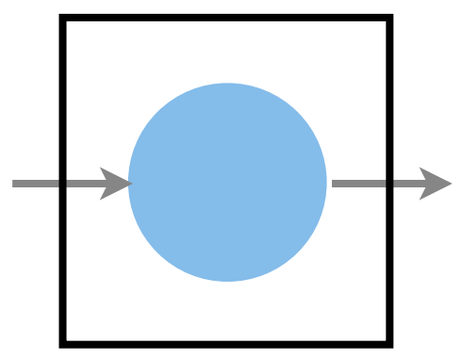


Chains

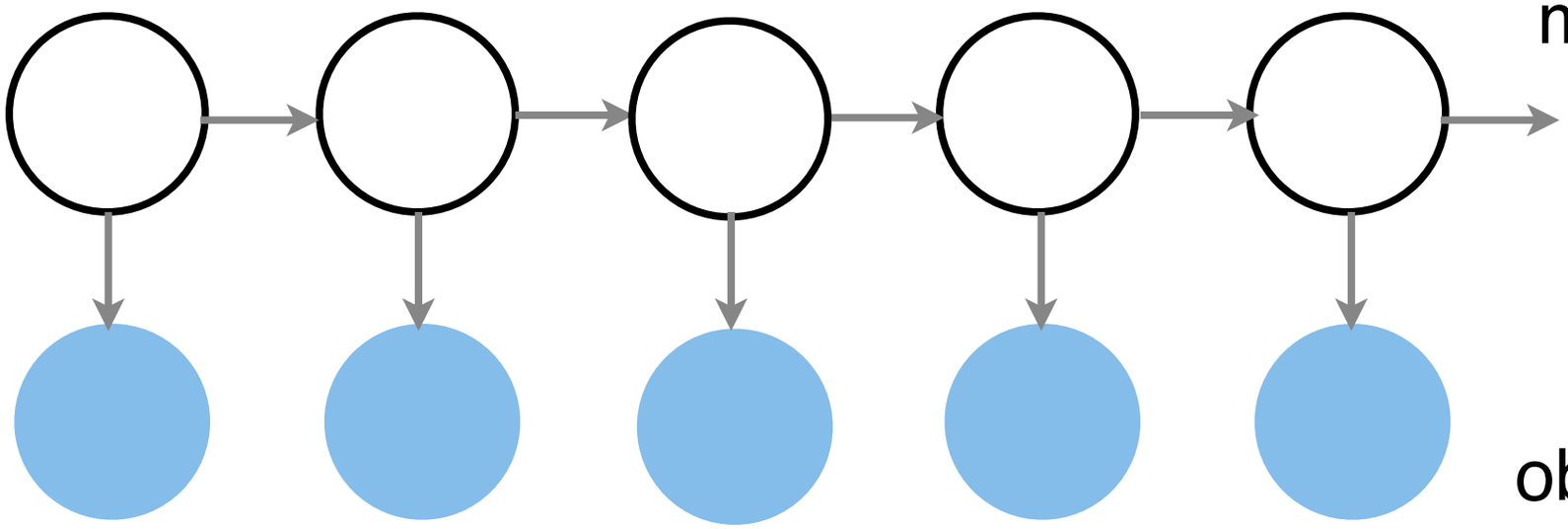
Markov Chain



Plate



Hidden Markov Chain

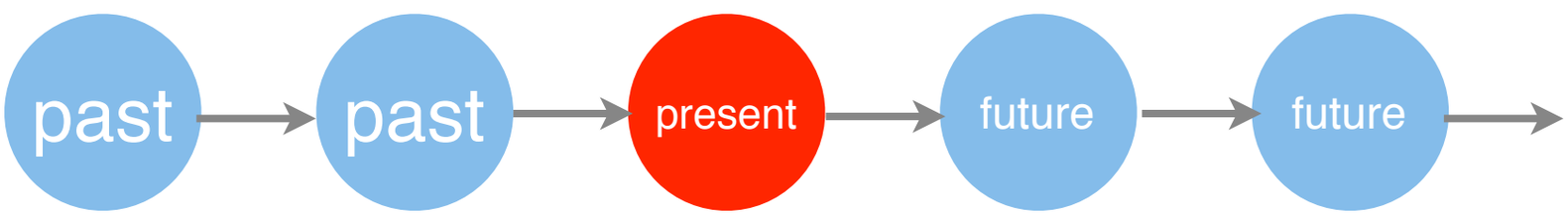


user's mindset

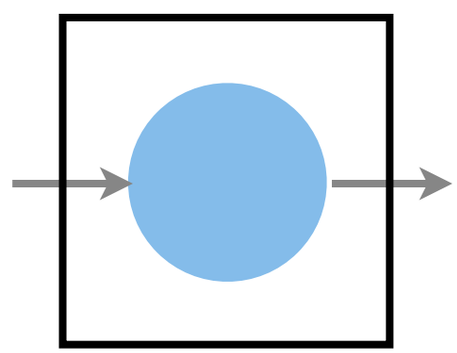
observed user action

Chains

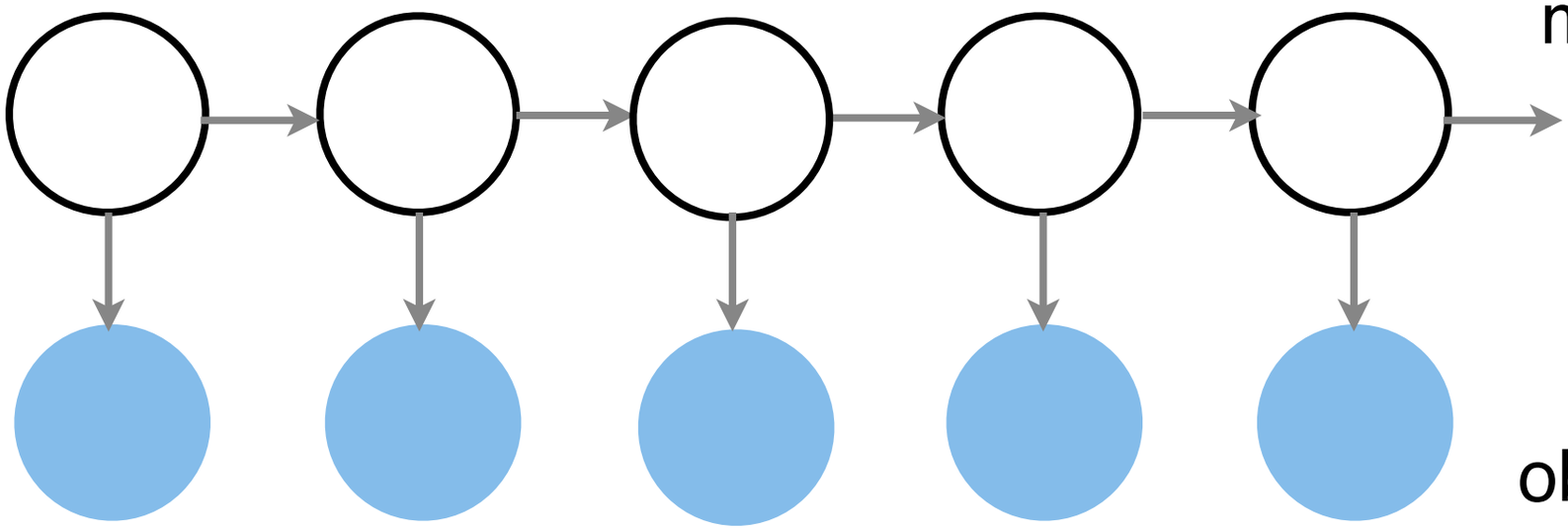
Markov Chain



Plate



Hidden Markov Chain



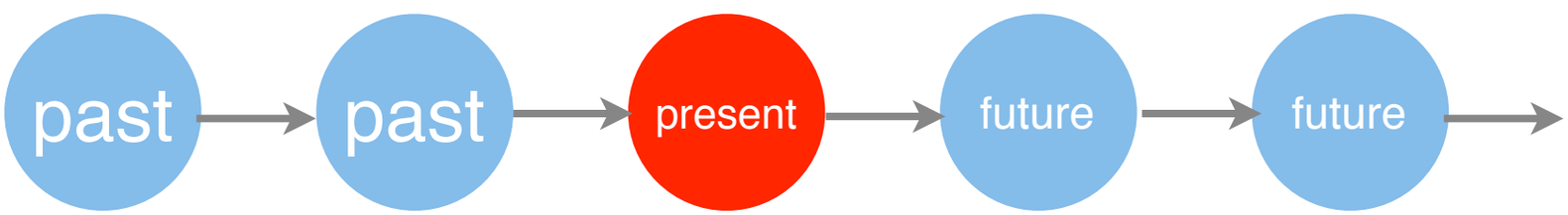
user's mindset

observed user action

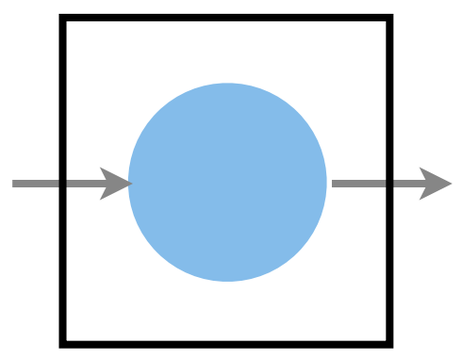
user model for traversal through search results

Chains

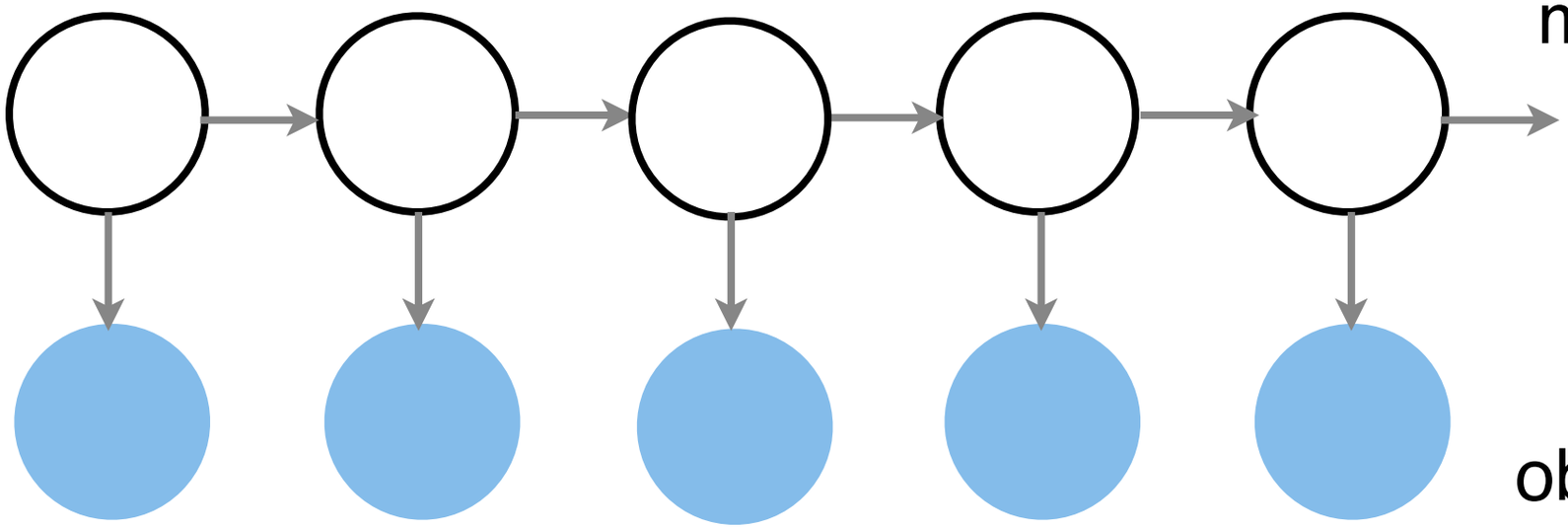
Markov Chain



Plate

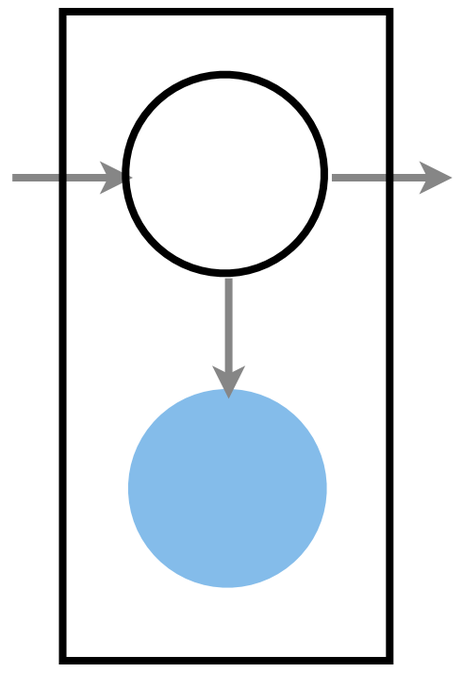


Hidden Markov Chain



user's mindset

observed user action



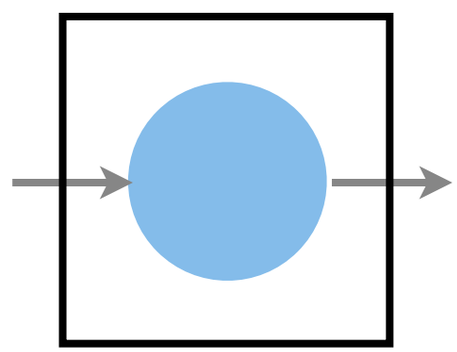
user model for traversal through search results

Chains

Markov Chain

$$p(x;\theta) = p(x_0;\theta) \prod_{i=1}^{n-1} p(x_{i+1}|x_i;\theta)$$

Plate

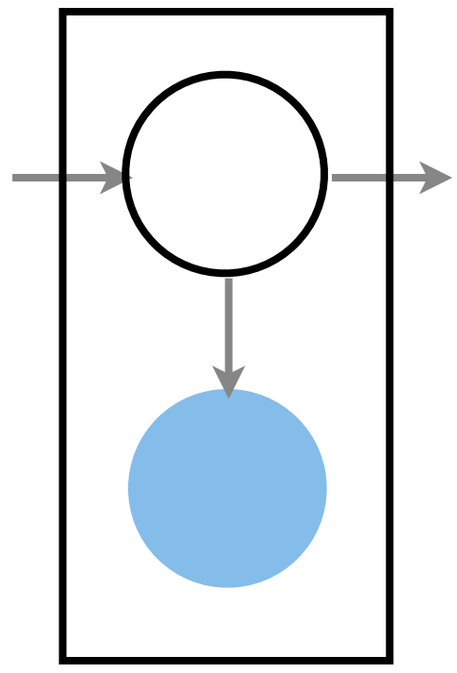


Hidden Markov Chain

$$p(x,y;\theta) = p(x_0;\theta) \prod_{i=1}^{n-1} p(x_{i+1}|x_i;\theta) \prod_{i=1}^{n} p(y_i|x_i)$$

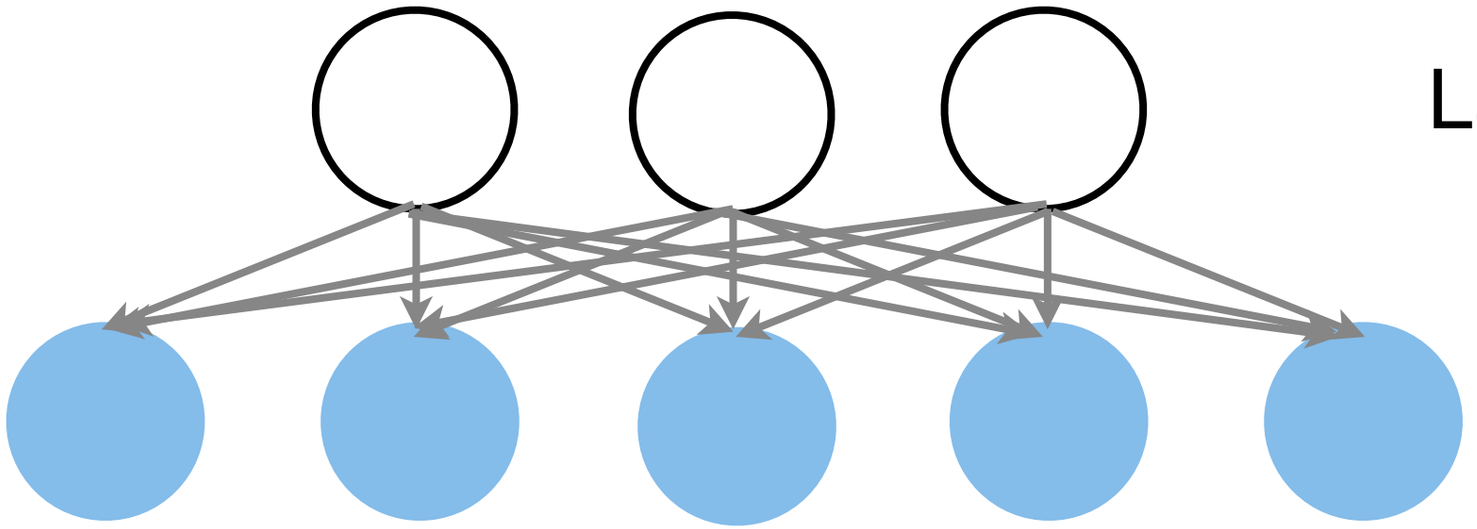
user's mindset

observed user action



user model for traversal through search results

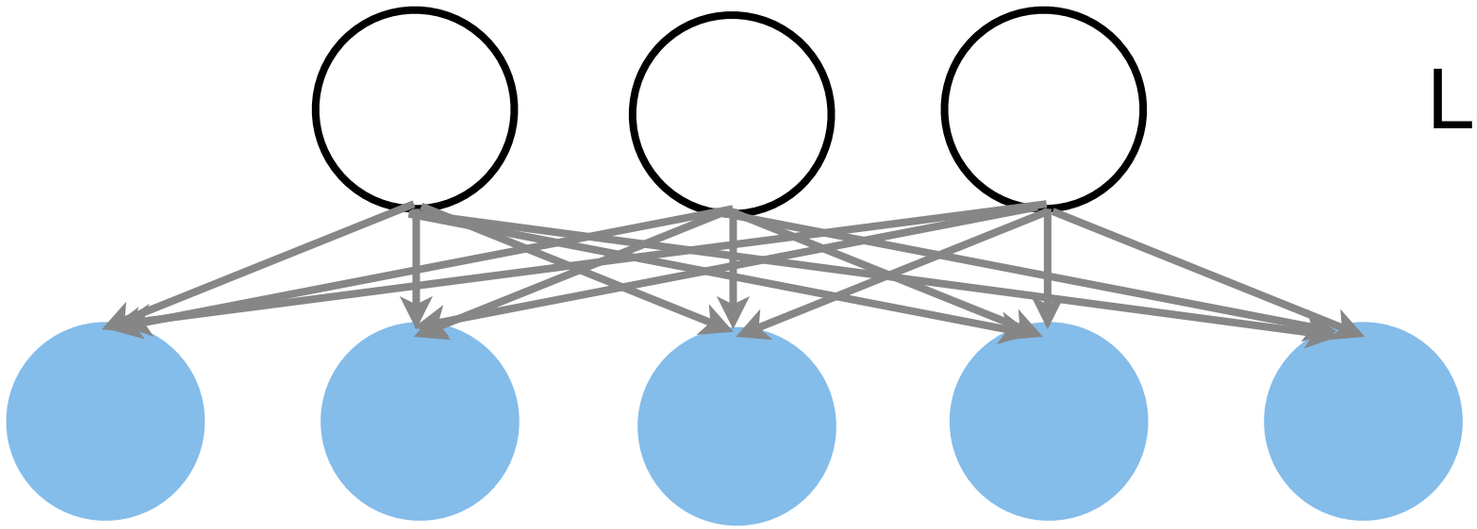
Factor Graphs



Latent Factors

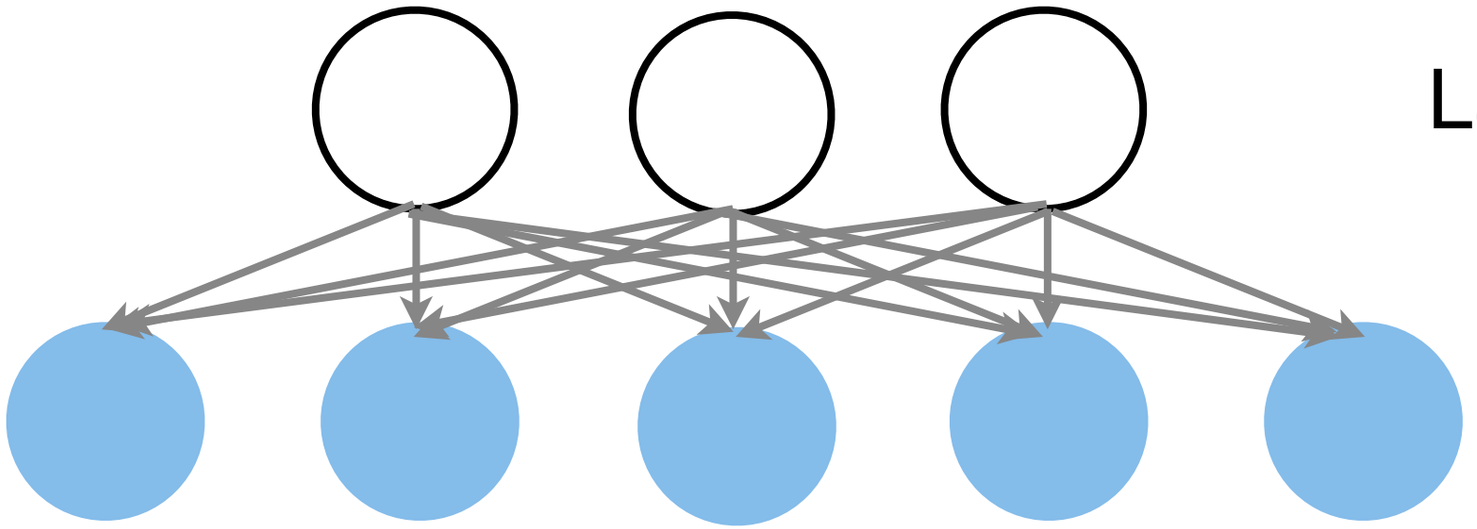
Observed Effects

Factor Graphs



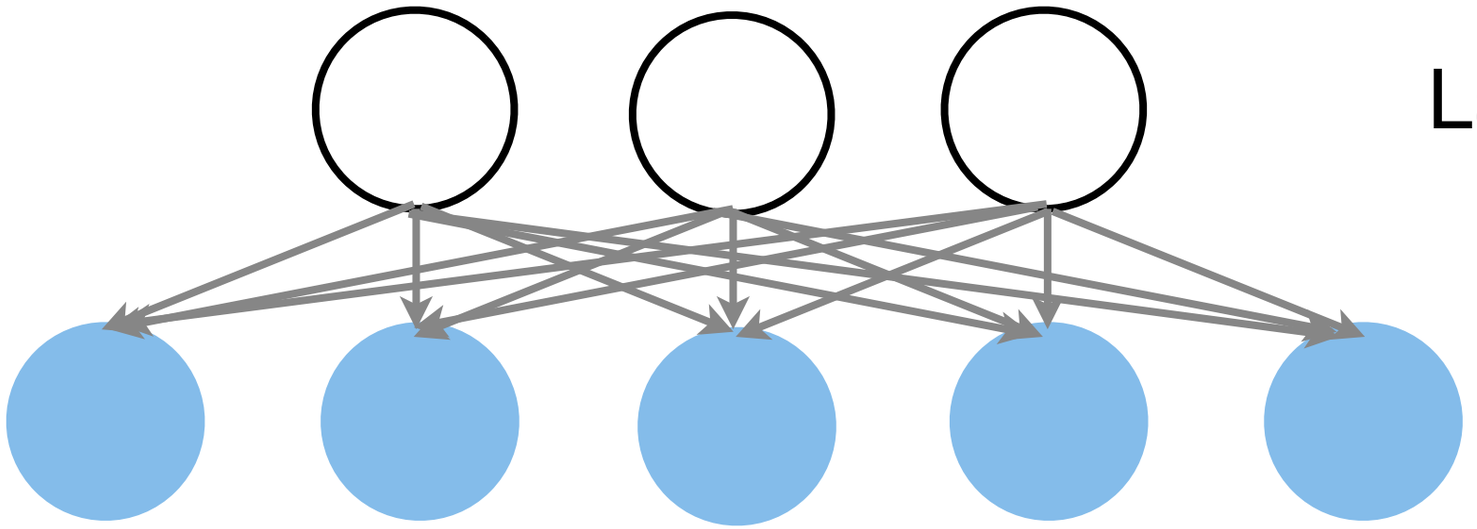
- **Observed effects**
  Click behavior, queries, watched news, emails

Factor Graphs



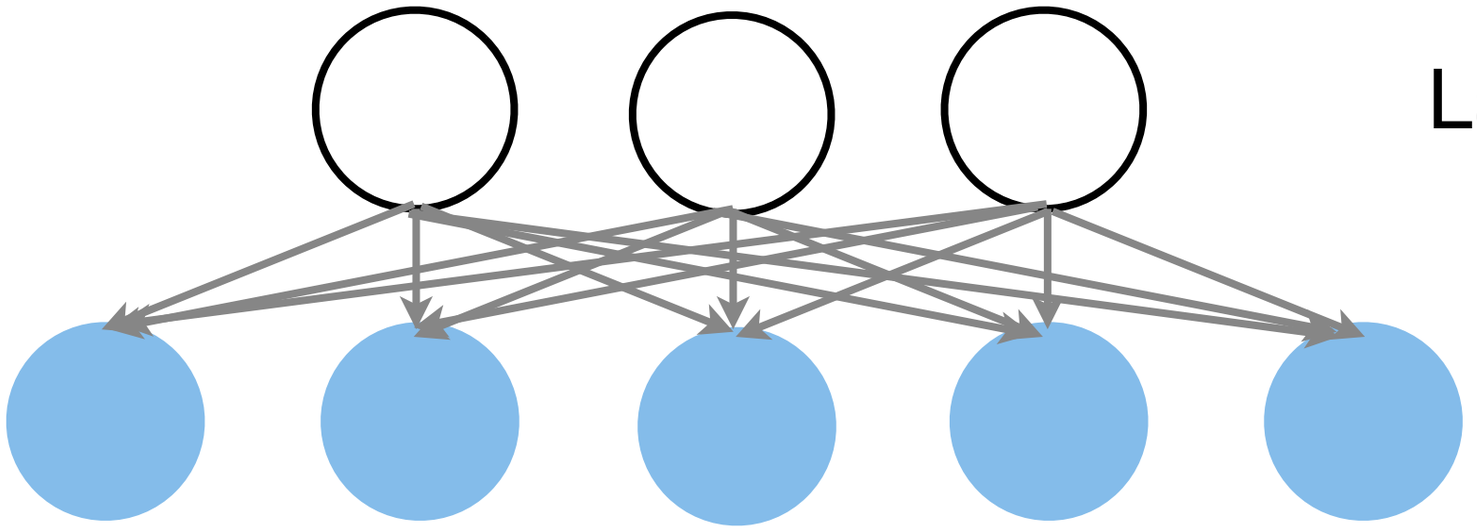
- **Observed effects**
  Click behavior, queries, watched news, emails
- **Latent factors**
  User profile, news content, hot keywords, social connectivity graph, events

Factor Graphs



- **Observed effects**
  Click behavior, queries, watched news, emails
- **Latent factors**
  User profile, news content, hot keywords, social connectivity graph, events
- **Multiple layers**
  Restricted Boltzmann Machine

Example - PCA/ICA

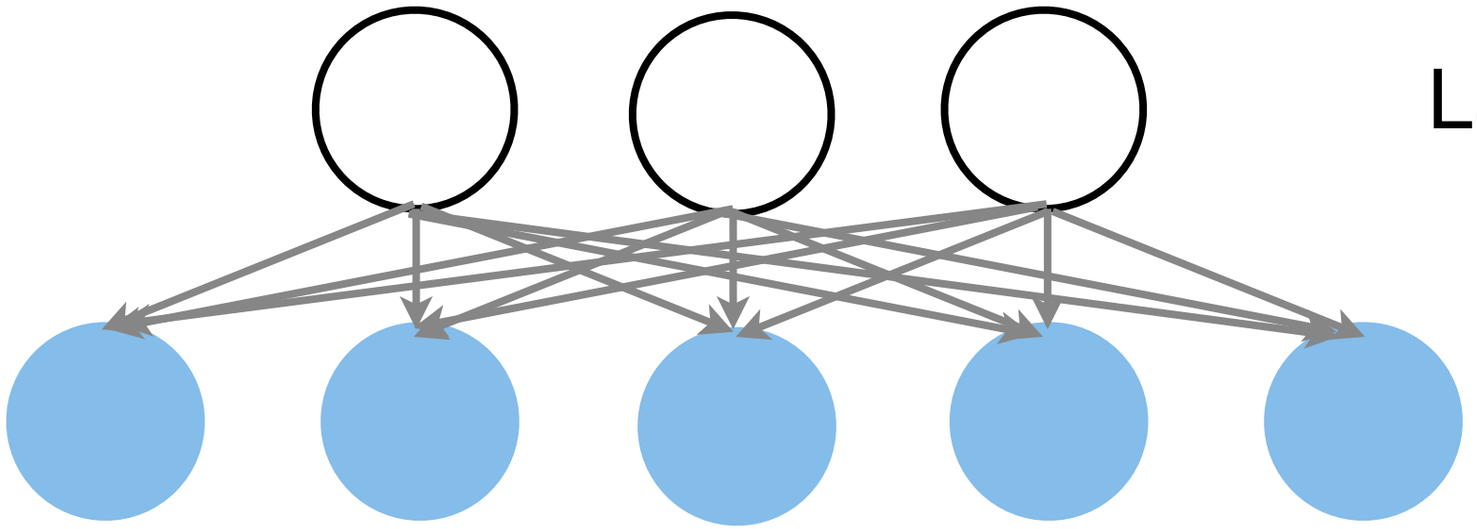


Latent Factors

Observed Effects

$$x \sim \mathcal{N}\left(\sum_{i=1}^{d} y_i v_i, \sigma^2 \mathbf{1}\right) \quad \text{and } p(y) = \prod_{i=1}^{d} p(y_i)$$

Example - PCA/ICA



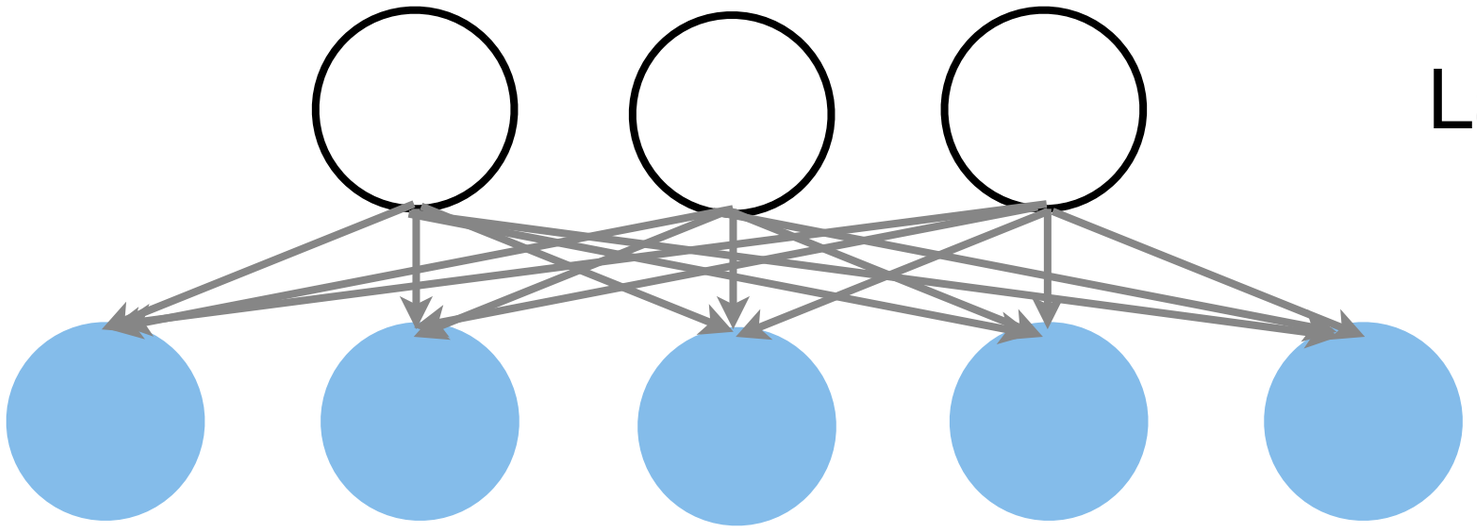
Latent Factors

Observed Effects

- Observed effects
  Click behavior, queries, watched news, emails

$$x \sim \mathcal{N}\left(\sum_{i=1}^{d} y_i v_i, \sigma^2 \mathbf{1}\right) \text{ and } p(y) = \prod_{i=1}^{d} p(y_i)$$

Example - PCA/ICA



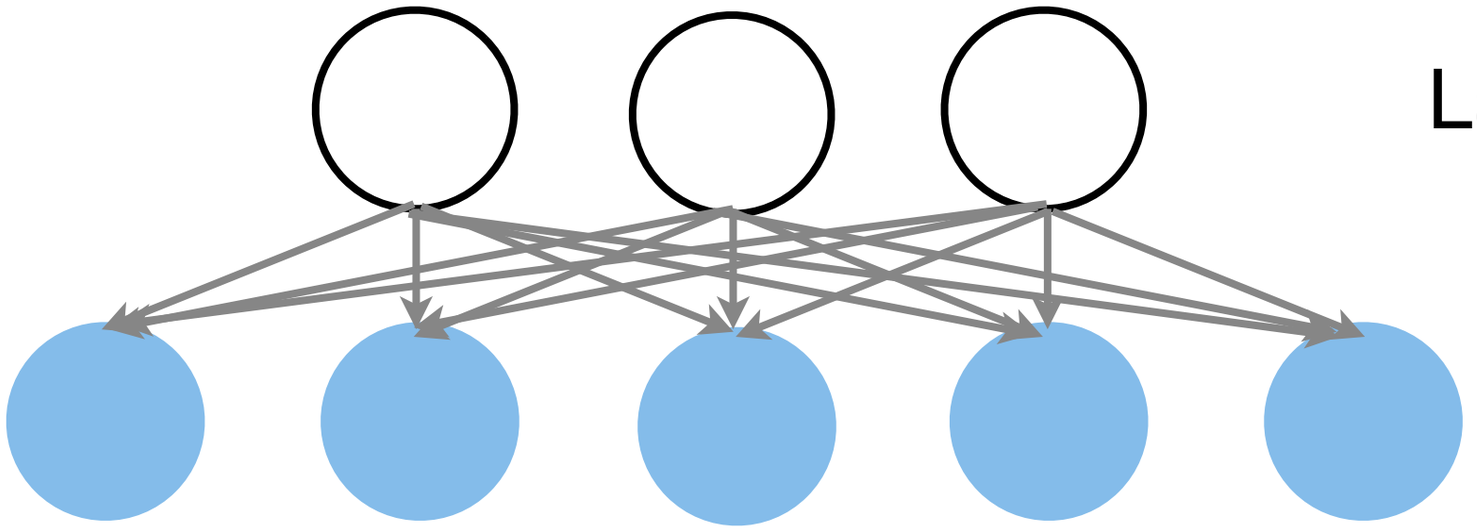
Latent Factors

Observed Effects

- Observed effects
  Click behavior, queries, watched news, emails

$$x \sim \mathcal{N}\left(\sum_{i=1}^{d} y_i v_i, \sigma^2 \mathbf{1}\right) \text{ and } p(y) = \prod_{i=1}^{d} p(y_i)$$

Example - PCA/ICA



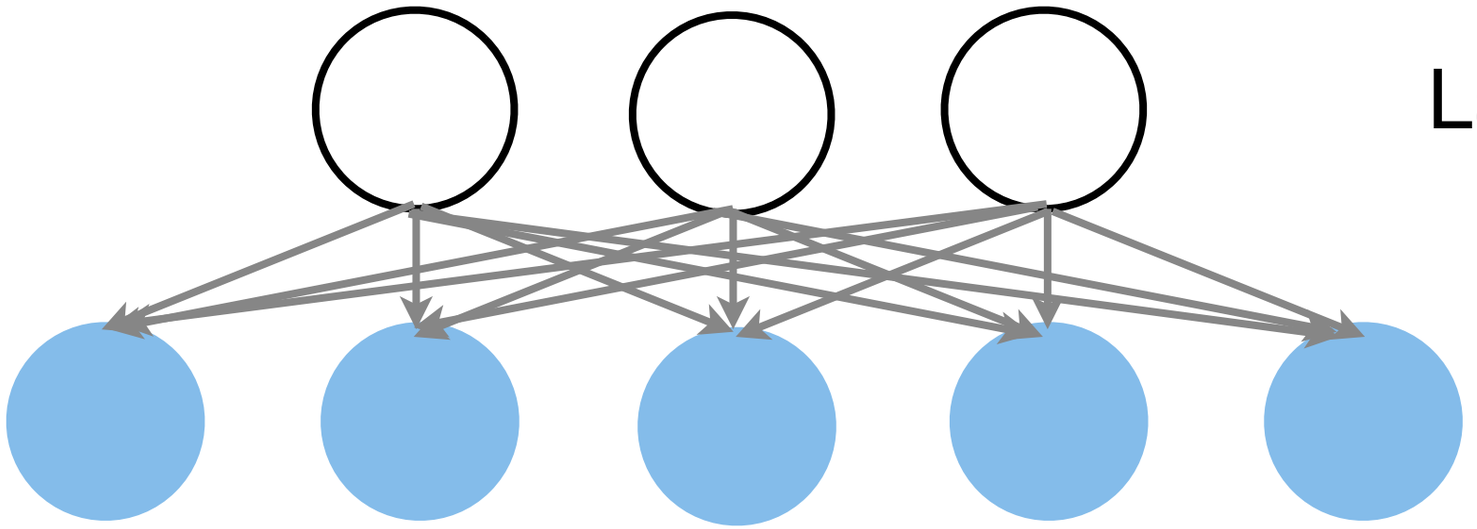
Latent Factors

Observed Effects

- Observed effects
  Click behavior, queries, watched news, emails

$$x \sim \mathcal{N}\left(\sum_{i=1}^{d} y_i v_i, \sigma^2 \mathbf{1}\right) \text{ and } p(y) = \prod_{i=1}^{d} p(y_i)$$

Example - PCA/ICA



Latent Factors

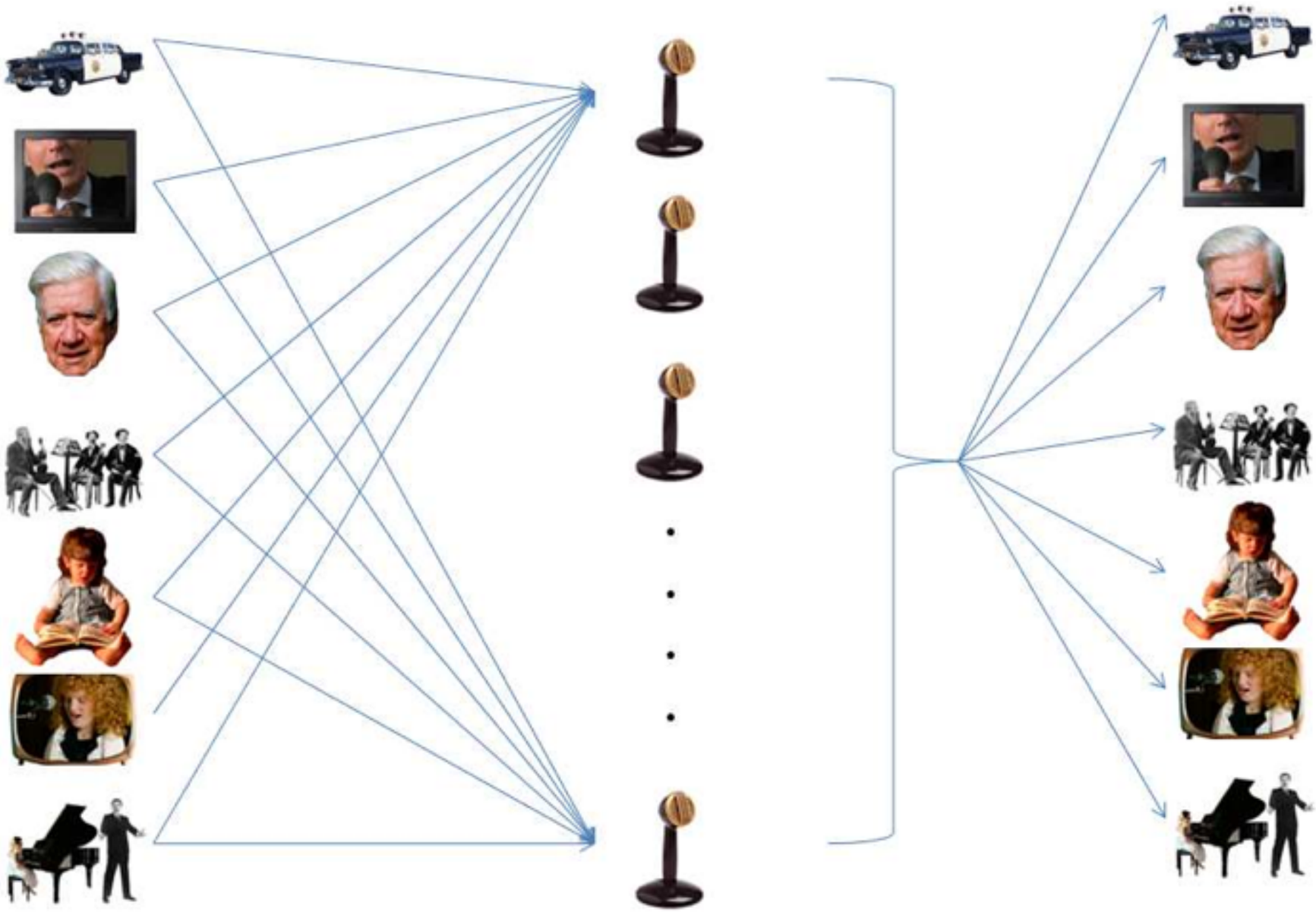
Observed Effects

- Observed effects
  Click behavior, queries, watched news, emails

$$x \sim \mathcal{N}\left(\sum_{i=1}^{d} y_i v_i, \sigma^2 \mathbf{1}\right) \text{ and } p(y) = \prod_{i=1}^{d} p(y_i)$$

- p(y) is Gaussian for PCA. General for ICA

Cocktail party problem

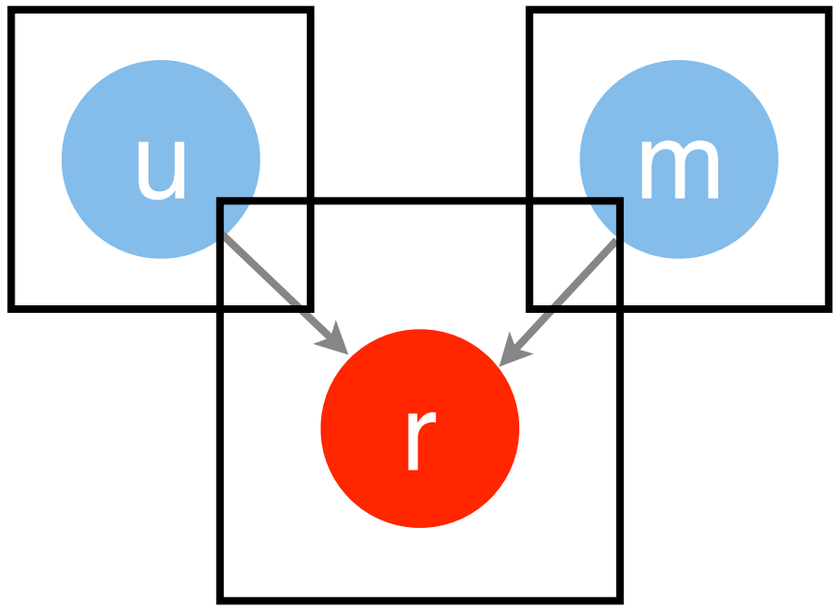


Sources

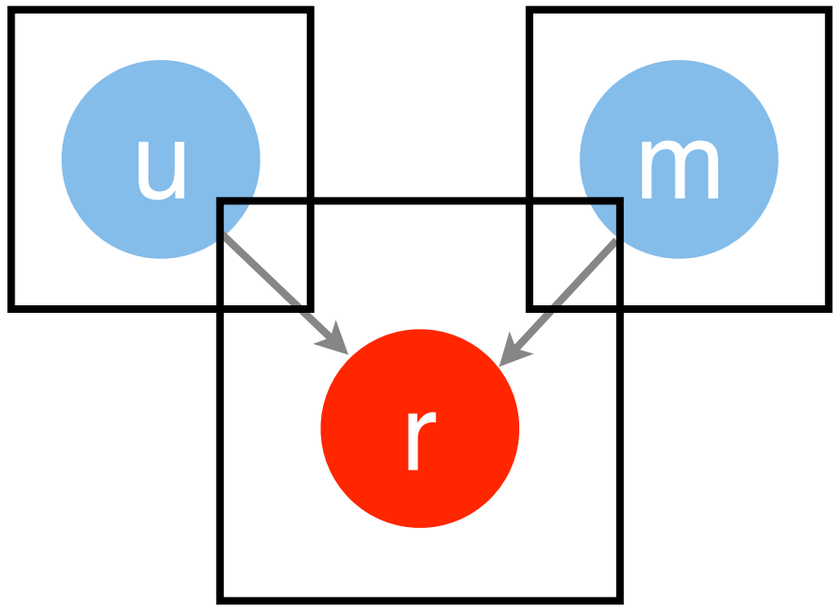
Mixtures

Separated Sources

Recommender Systems

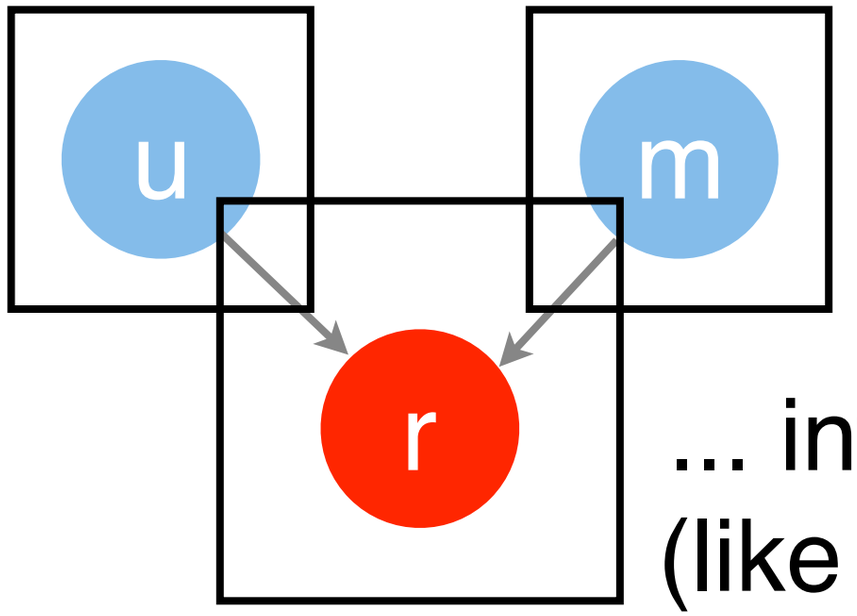


Recommender Systems



- Users u
- Movies m
- Ratings r (but only for a subset of users)

Recommender Systems

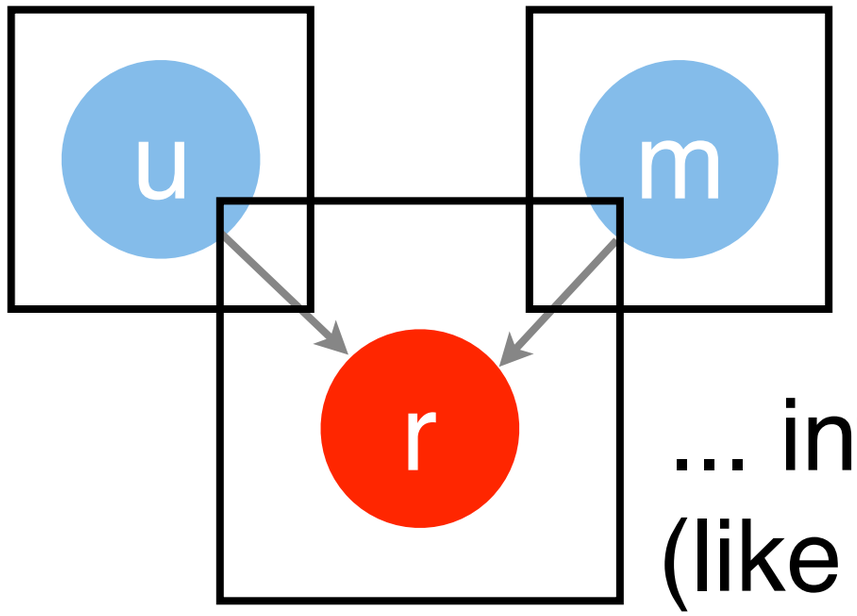


... intersecting plates ...
(like nested FOR loops)

- Users u
- Movies m
- Ratings r (but only for a subset of users)

Recommender Systems

news,
SearchMonkey
answers
social
ranking
OMG
personals



... intersecting plates ...
(like nested FOR loops)

- Users u
- Movies m
- Ratings r (but only for a subset of users)

Challenges

engineering

machine learning

Challenges

- How to design models
  - Common (engineering) sense
  - Computational tractability

engineering

machine learning

Challenges

- How to design models
  - Common (engineering) sense
  - Computational tractability
- Dependency analysis

engineering

machine learning

Challenges

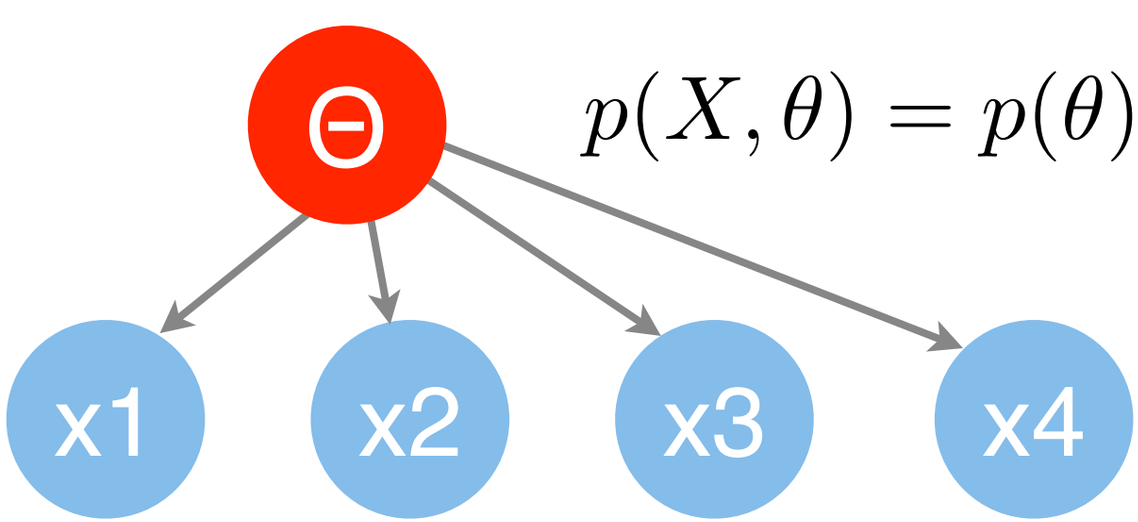
- How to design models
  - Common (engineering) sense
  - Computational tractability
- Dependency analysis
- Inference
  - Easy for fully observed situations
  - Many algorithms if not fully observed
  - Dynamic programming / message passing

engineering

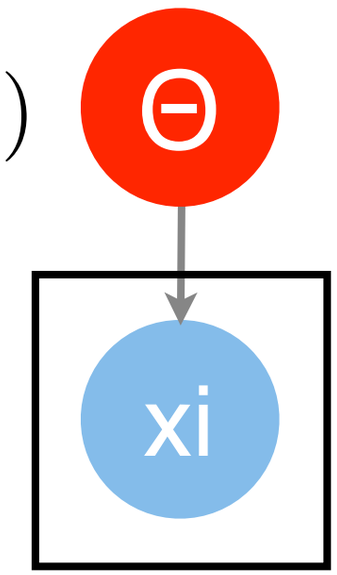
machine learning

Summary

- Repeated structure - encode with plate
- Chains, bipartite graphs, etc (more later)
- Plates can intersect
- Not all variables are observed



$$p(X, \theta) = p(\theta) \prod_i p(x_i|\theta)$$



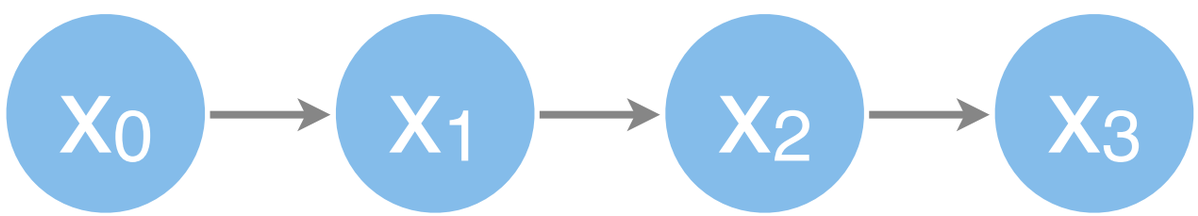
CHAIN
TREE

WAYAN
The Land of Spicy Hills

DISTRICT TOURISM PROMOTION COUNCIL

Markov Chains

$$p(x;\theta) = p(x_0;\theta)\prod_{i=1}^{n-1} p(x_{i+1}|x_i;\theta)$$



Transition Matrices

x_0

0	0.4
1	0.6

x1

	0	1
0	0.2	0.1
1	0.8	0.9

x_2

	0	1
0	0.8	0.5
1	0.2	0.5

x_3

	0	1
0	0	1
1	1	0

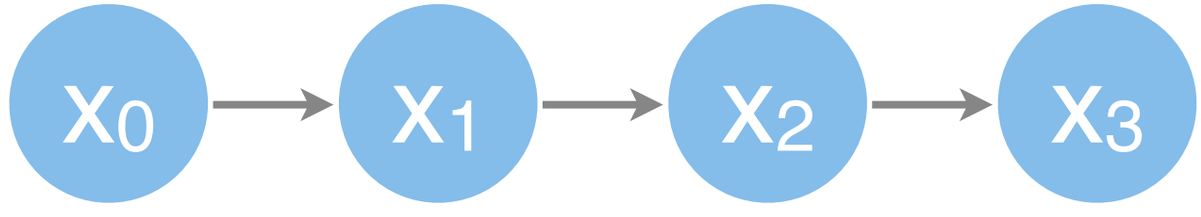
Unraveling the chain

$$p(x_1) = \sum_{x_0} p(x_1|x_0)p(x_0) \Longleftrightarrow \pi_1 = \Pi_{0\to 1}\pi_0$$

$$p(x_2) = \sum_{x_1} p(x_2|x_1)p(x_1) \Longleftrightarrow \pi_2 = \Pi_{1\to 2}\pi_1 = \Pi_{1\to 2}\Pi_{0\to 1}\pi_0$$

Markov Chains

$$p(x;\theta) = p(x_0;\theta)\prod_{i=1}^{n-1} p(x_{i+1}|x_i;\theta)$$



- From the start - sum sequentially

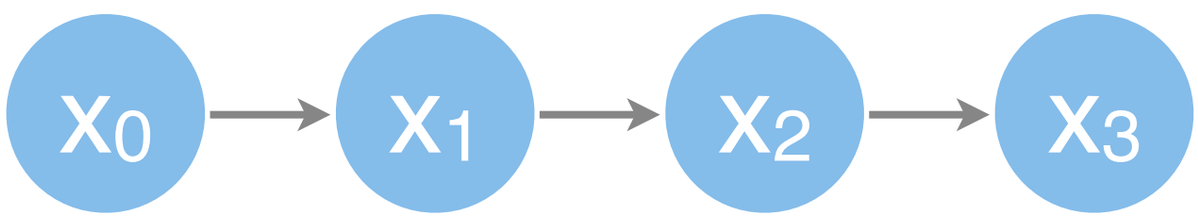
$$p(x_i|x_1) = \sum_{x_j:1<j<i}\prod_{l=2}^{i-1} p(x_{l+1}|x_l)\cdot \underbrace{p(x_2|x_1)}_{=:l_2(x_2)}$$

$$= \sum_{x_j:2<j<i}\prod_{l=3}^{i-1} p(x_{l+1}|x_l)\cdot \underbrace{\sum_{x_2} p(x_3|x_2)l_2(x_2)}_{=:l_3(x_3)}$$

$$= \sum_{x_j:3<j<i}\prod_{l=4}^{i-1} p(x_{l+1}|x_l)\cdot \underbrace{\sum_{x_3} p(x_4|x_3)l_3(x_3)}_{=:l_4(x_4)}$$

Markov Chains

$$p(x;\theta) = p(x_0;\theta)\prod_{i=1}^{n-1} p(x_{i+1}|x_i;\theta)$$



Transition Matrices

x_0

0	0.4
1	0.6

x1

	0	1
0	0.2	0.1
1	0.8	0.9

x_2

	0	1
0	0.8	0.5
1	0.2	0.5

x_3

	0	1
0	0	1
1	1	0

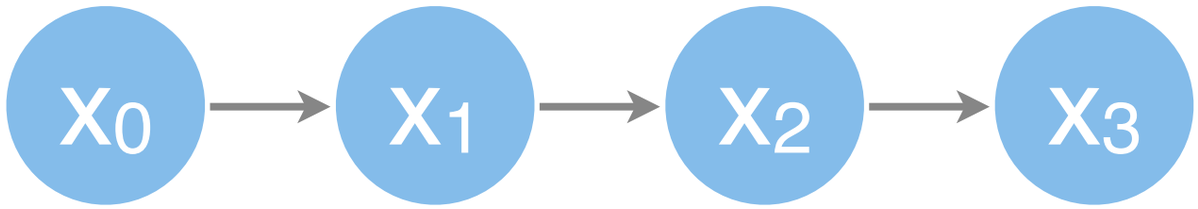
Unraveling the chain

```
x0  = [0.4; 0.6];
Pi1 = [0.2 0.1; 0.8 0.9];
Pi2 = [0.8 0.5; 0.2 0.5];
Pi3 = [0 1; 1 0];
x3  = Pi3 * Pi2 * Pi1 * x0 = [0.45800; 0.54200]
```

only need matrix-vector

Markov Chains

$$p(x;\theta) = p(x_0;\theta)\prod_{i=1}^{n-1} p(x_{i+1}|x_i;\theta)$$



- From the end - sum sequentially

$$p(x_1|x_n) \propto \sum_{x_j:1<j<n} \prod_{l=1}^{n-1} p(x_{l+1}|x_l) \cdot \underbrace{1}_{=:r_n(x_n)}$$

normalize in the end

$$= \sum_{x_j:1<j<n-1} \prod_{l=1}^{n-2} p(x_{l+1}|x_l) \cdot \underbrace{\sum_{x_n} p(x_n|x_{n-1}) r_n(x_n)}_{=:r_{n-1}(x_{n-1})}$$

$$= \sum_{x_j:1<j<n-2} \prod_{l=1}^{n-3} p(x_{l+1}|x_l) \cdot \underbrace{\sum_{x_{n-1}} p(x_{n-1}|x_{n-2}) r_{n-1}(x_{n-1})}_{=:r_{n-2}(x_{n-2})}$$

Example - inferring lunch

current

		
	0.9	0.2
	0.1	0.8

- Initial probability $p(x_0=t)=p(x_0=b) = 0.5$
- Stationary transition matrix
- On fifth day observed at Tazza d'oro $p(x_5=t)=1$
- Distribution on day 3
  - Left messages to 3
  - Right messages to 3
  - Renormalize

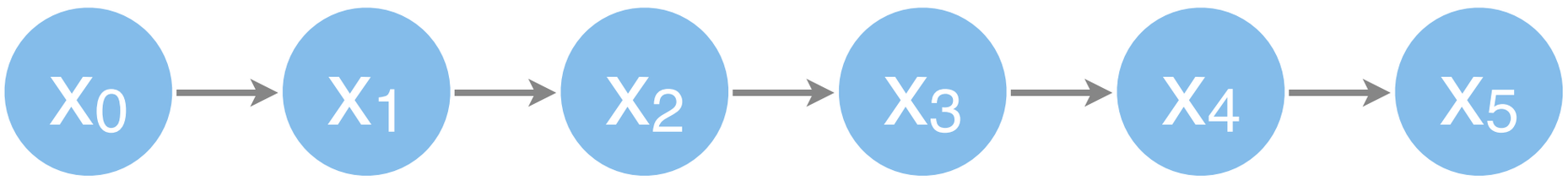
Example - inferring lunch

current

		
	0.9	0.2
	0.1	0.8

```
> Pi = [0.9, 0.2; 0.1 0.8]
Pi =
    0.90000   0.20000
    0.10000   0.80000
> l1 = [0.5; 0.5];
> l3 = Pi * Pi * l1
l3 =
    0.58500
    0.41500
> r5 = [1; 0];
> r3 = Pi' * Pi' * r5
r3 =
    0.83000
    0.34000
> (l3 .* r3) / sum(l3 .* r3)
ans =
    0.77483
    0.22517
```

Message Passing



$$l_i = \Pi_i l_{i-1}$$

$$r_i = \Pi_i^\top r_{i+1}$$

- Send forward messages starting from left node

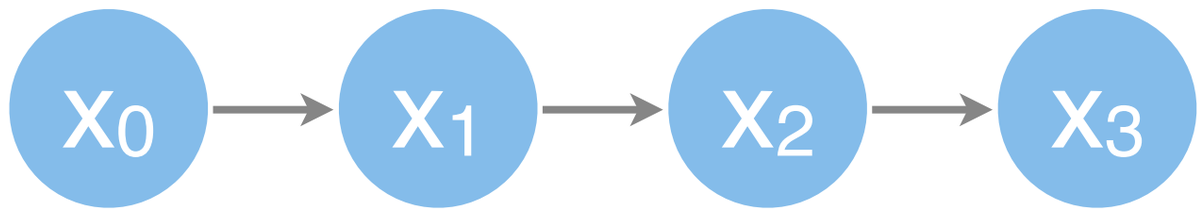
$$m_{i-1\to i}(x_i) = \sum_{x_{i-1}} m_{i-2\to i-1}(x_{i-1}) f(x_{i-1}, x_i)$$

- Send backward messages starting from right node

$$m_{i+1\to i}(x_i) = \sum_{x_{i+1}} m_{i+2\to i+1}(x_{i+1}) f(x_i, x_{i+1})$$

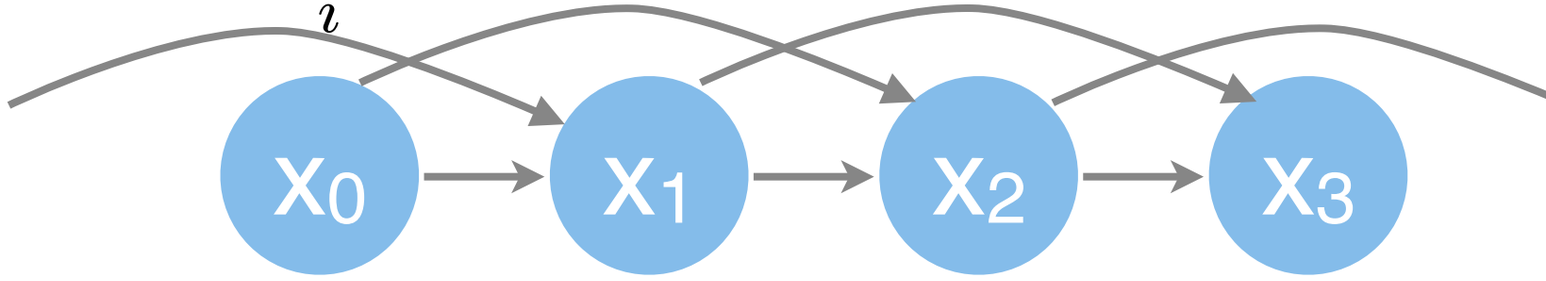
Higher Order Markov Chains

- First order chain



$$p(X) = p(x_0) \prod_i p(x_{i+1}|x_i)$$

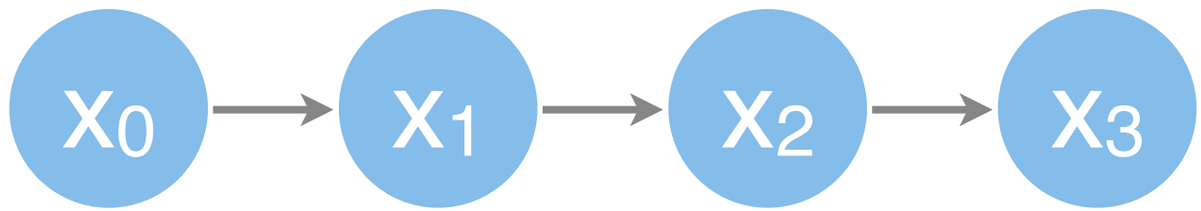
- Second order



$$p(X) = p(x_0, x_1) \prod_i p(x_{i+1}|x_i, x_{i-1})$$

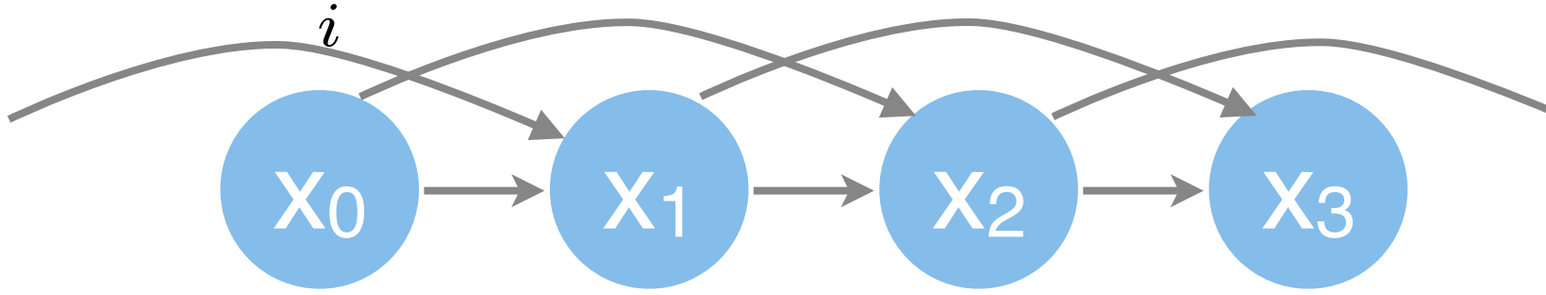
Higher Order Markov Chains

- First order chain



$$p(X) = p(x_0) \prod_i p(x_{i+1}|x_i)$$

- Second order



$$p(X) = p(x_0, x_1) \prod_i p(x_{i+1}|x_i, x_{i-1})$$

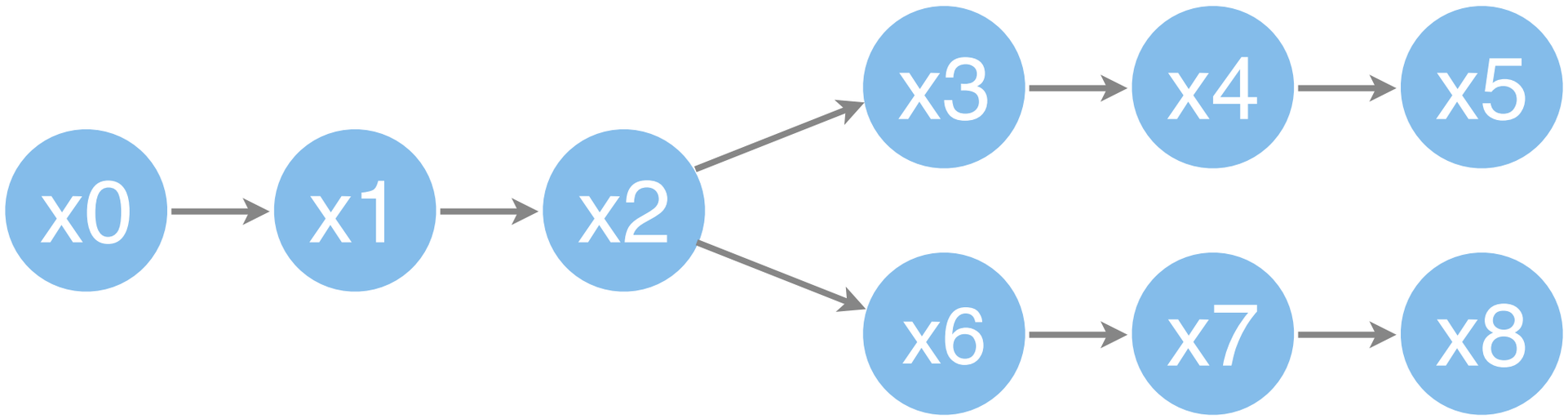
Mark Reid @mdreid 22 Mar

Markov In Chains #MLBandNames

Collapse Reply Retweet Favorite More

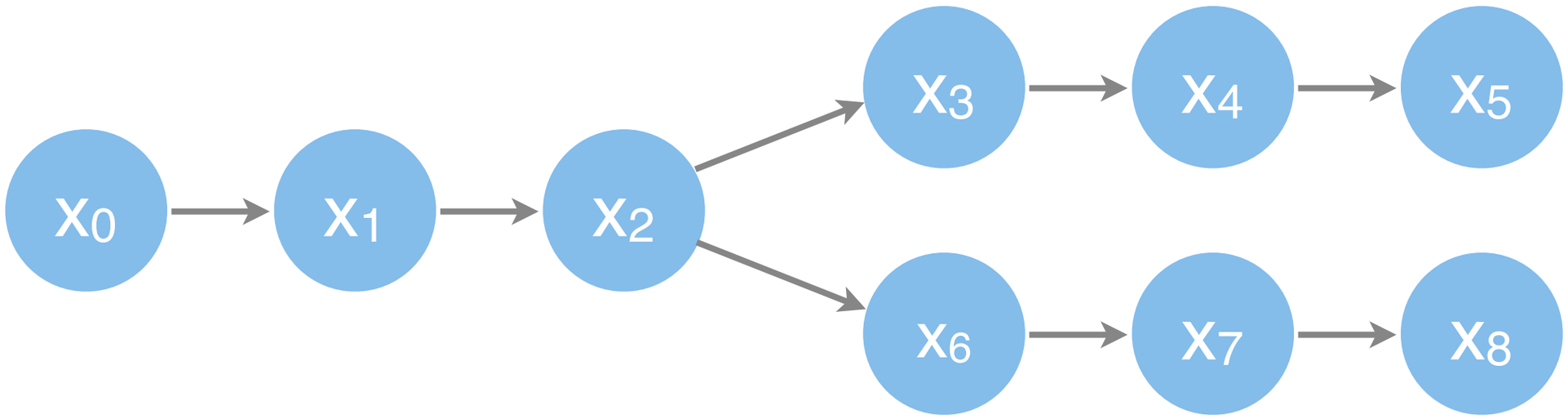
9 RETWEETS 4 FAVORITES

Trees

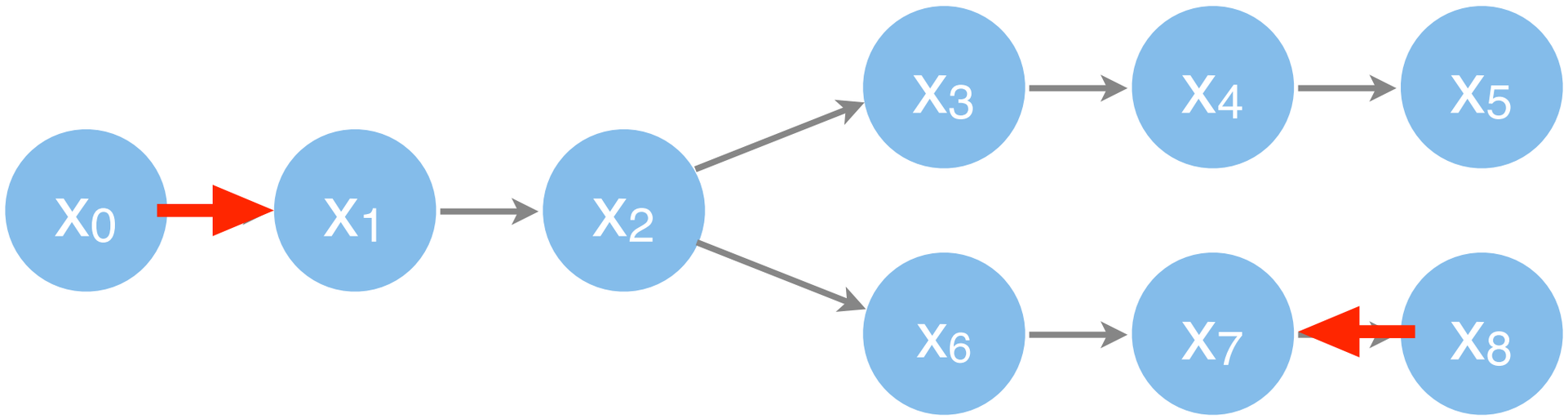


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

Trees



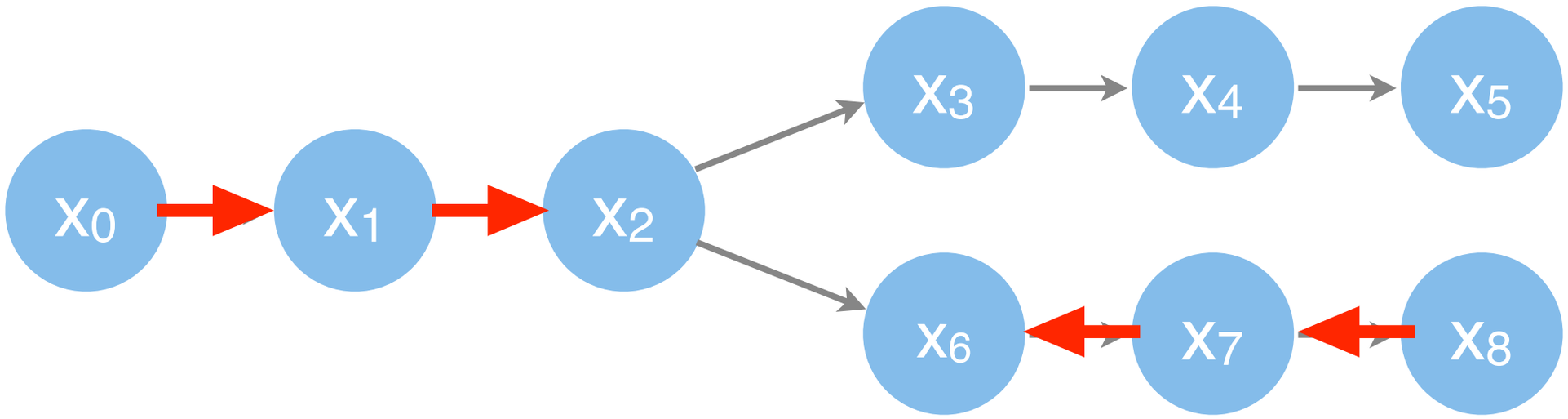
Trees



$$l_1(x_1) = \sum_{x_0} p(x_0)p(x_1|x_0)$$

$$r_7(x_7) = \sum_{x_8} p(x_8|x_7)$$

Trees



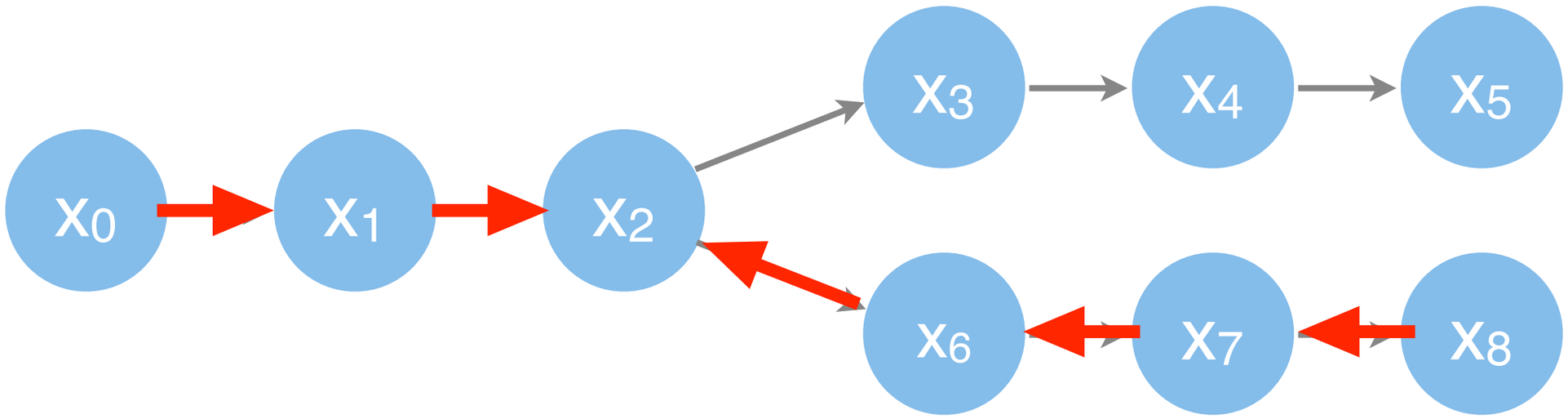
$$l_1(x_1) = \sum_{x_0} p(x_0)p(x_1|x_0)$$

$$l_2(x_2) = \sum_{x_1} l_1(x_1)p(x_2|x_1)$$

$$r_7(x_7) = \sum_{x_8} p(x_8|x_7)$$

$$r_6(x_6) = \sum_{x_7} r_7(x_7)p(x_7|x_6)$$

Trees



$$l_1(x_1) = \sum_{x_0} p(x_0)p(x_1|x_0)$$

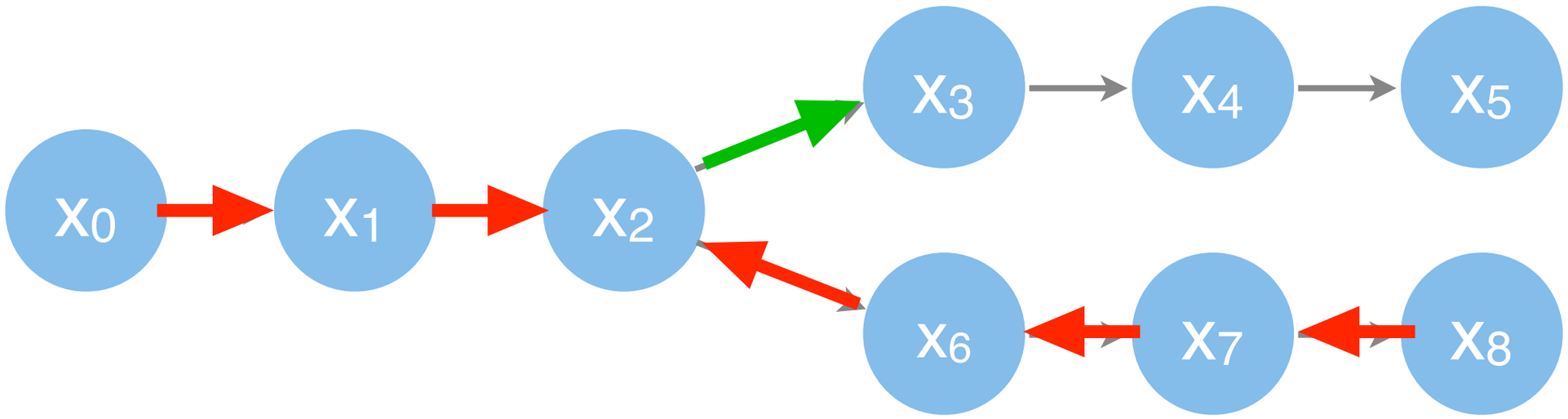
$$l_2(x_2) = \sum_{x_1} l_1(x_1)p(x_2|x_1)$$

$$r_7(x_7) = \sum_{x_8} p(x_8|x_7)$$

$$r_6(x_6) = \sum_{x_7} r_7(x_7)p(x_7|x_6)$$

$$r_2(x_2) = \sum_{x_6} r_6(x_6)p(x_6|x_2)$$

Trees



$$l_1(x_1) = \sum_{x_0} p(x_0)p(x_1|x_0)$$

$$l_2(x_2) = \sum_{x_1} l_1(x_1)p(x_2|x_1)$$

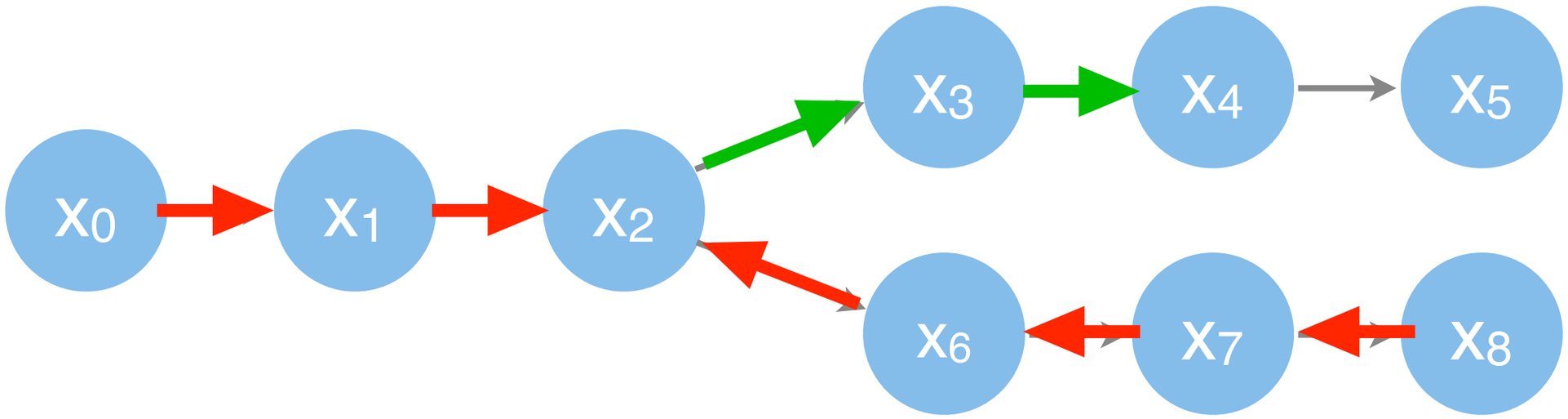
$$l_3(x_3) = \sum_{x_2} l_2(x_2)p(x_3|x_2)r_2(x_2)$$

$$r_7(x_7) = \sum_{x_8} p(x_8|x_7)$$

$$r_6(x_6) = \sum_{x_7} r_7(x_7)p(x_7|x_6)$$

$$r_2(x_2) = \sum_{x_6} r_6(x_6)p(x_6|x_2)$$

Trees



$$l_1(x_1) = \sum_{x_0} p(x_0)p(x_1|x_0)$$

$$l_2(x_2) = \sum_{x_1} l_1(x_1)p(x_2|x_1)$$

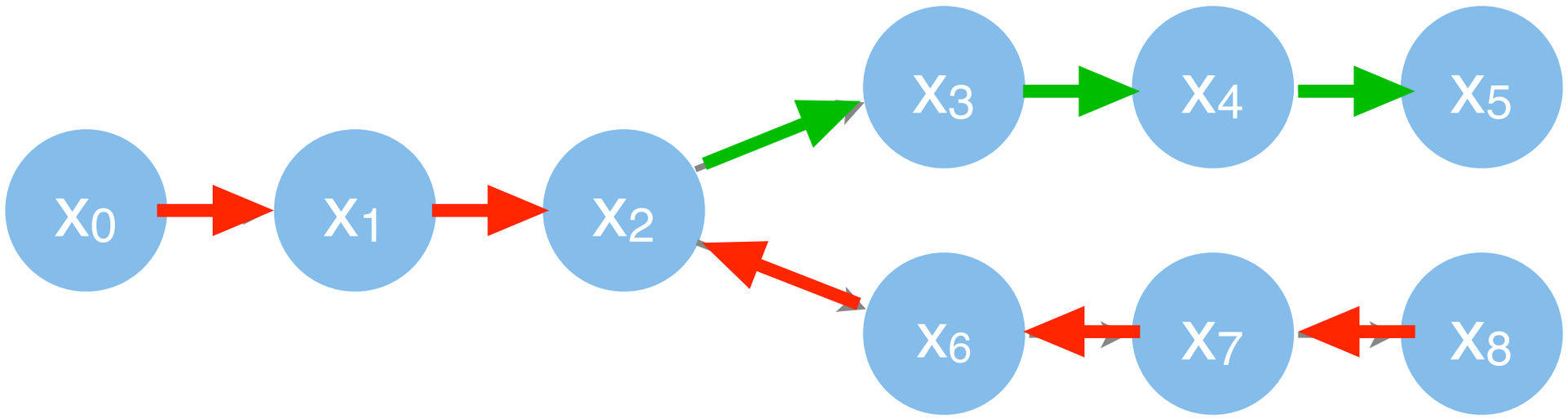
$$l_3(x_3) = \sum_{x_2} l_2(x_2)p(x_3|x_2)r_2(x_2)$$

$$r_7(x_7) = \sum_{x_8} p(x_8|x_7)$$

$$r_6(x_6) = \sum_{x_7} r_7(x_7)p(x_7|x_6)$$

$$r_2(x_2) = \sum_{x_6} r_6(x_6)p(x_6|x_2)$$

Trees



$$l_1(x_1) = \sum_{x_0} p(x_0)p(x_1|x_0)$$

$$l_2(x_2) = \sum_{x_1} l_1(x_1)p(x_2|x_1)$$

$$l_3(x_3) = \sum_{x_2} l_2(x_2)p(x_3|x_2)r_2(x_2)$$

$$r_7(x_7) = \sum_{x_8} p(x_8|x_7)$$

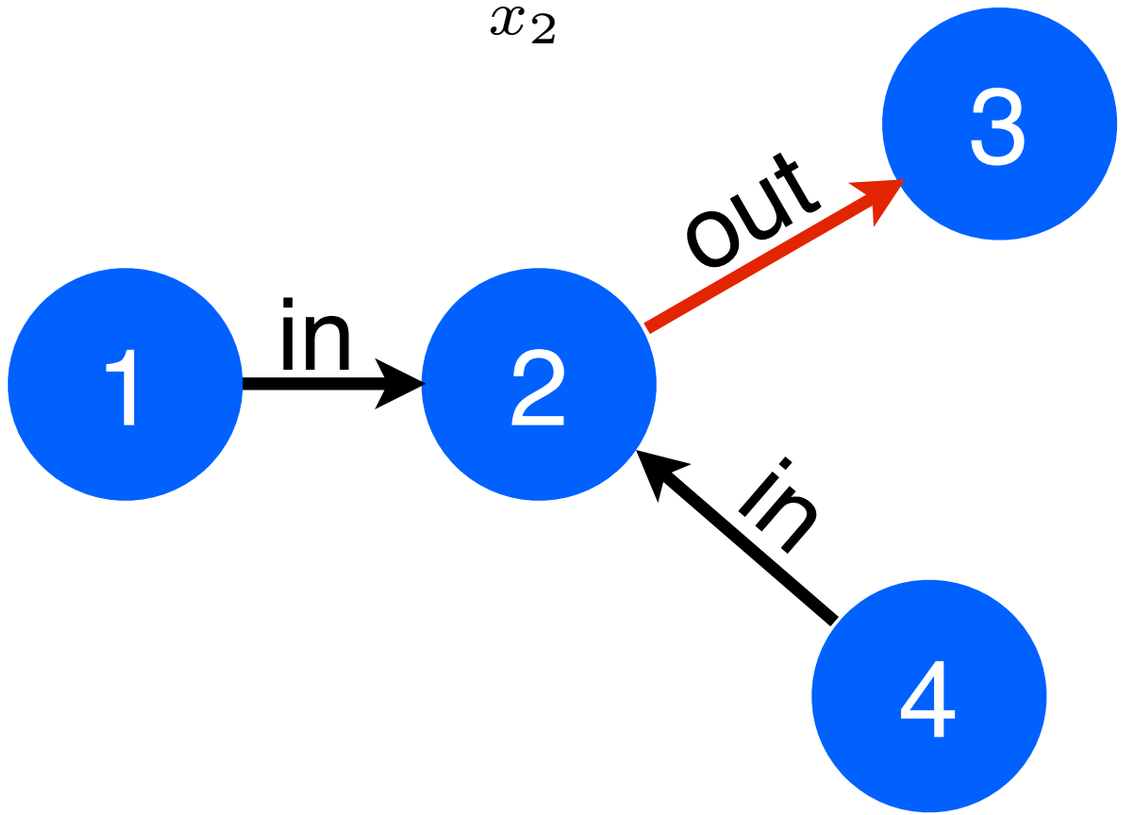
$$r_6(x_6) = \sum_{x_7} r_7(x_7)p(x_7|x_6)$$

$$r_2(x_2) = \sum_{x_6} r_6(x_6)p(x_6|x_2)$$

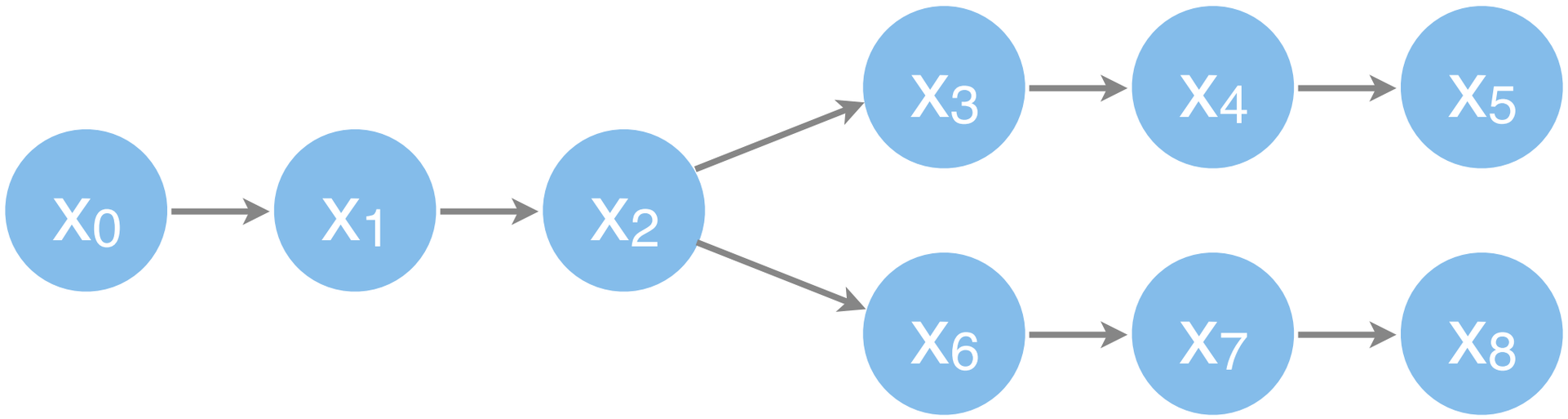
Junction Template

- Order of computation
- Dependence does not matter
  (only matters for parametrization)

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{4\to 2}(x_2) f(x_2, x_3)$$



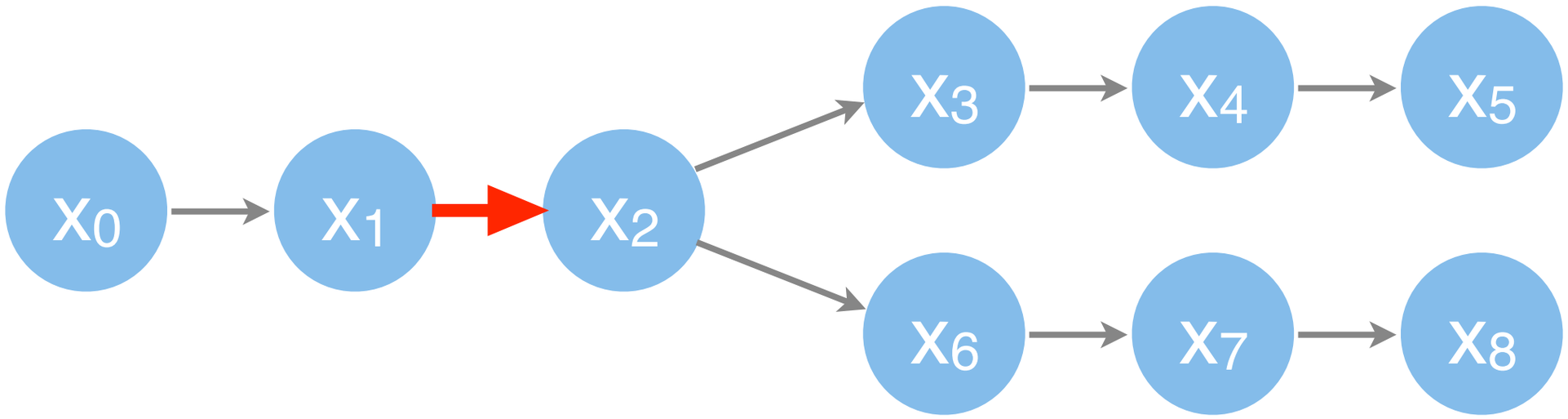
Trees



- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

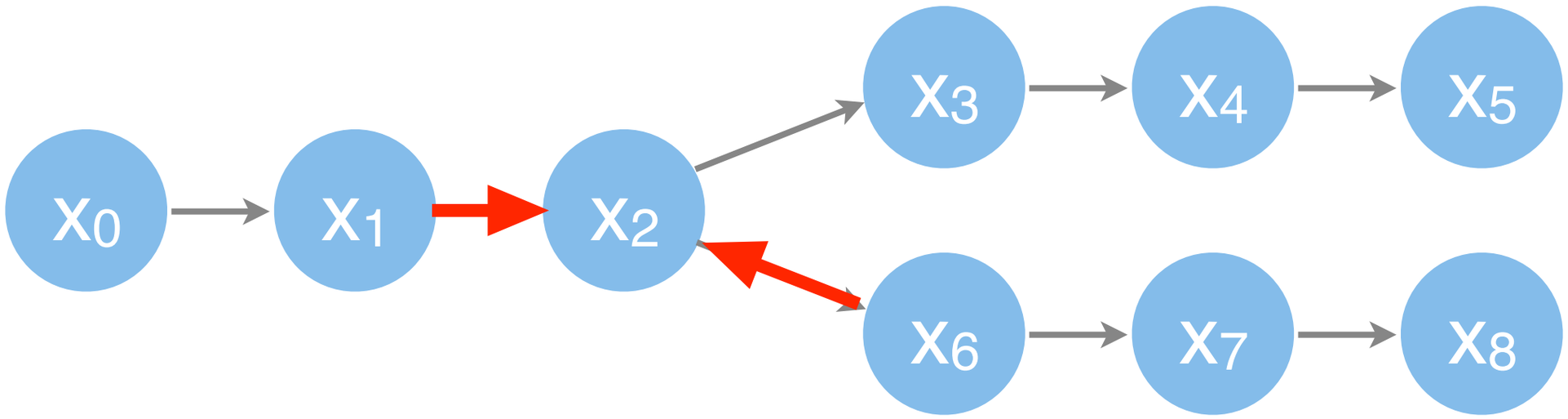
Trees



- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

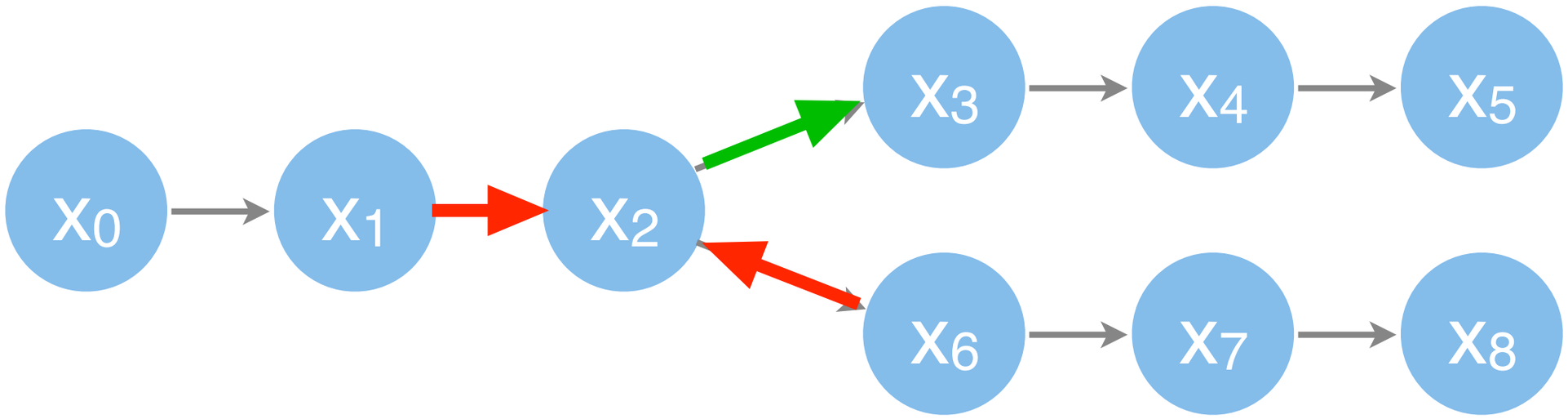
Trees



- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

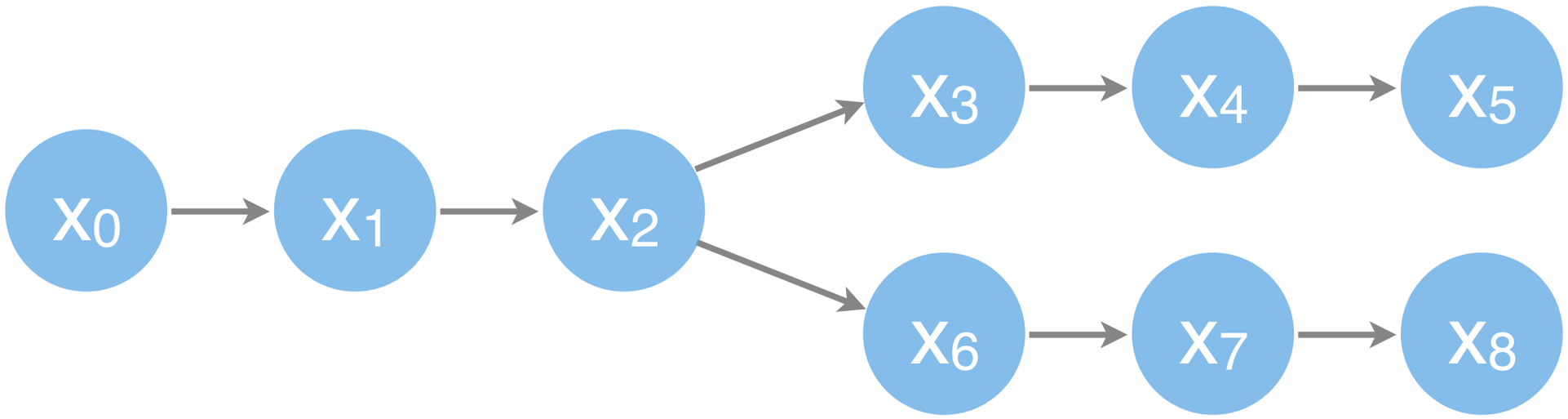
Trees



- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

Trees

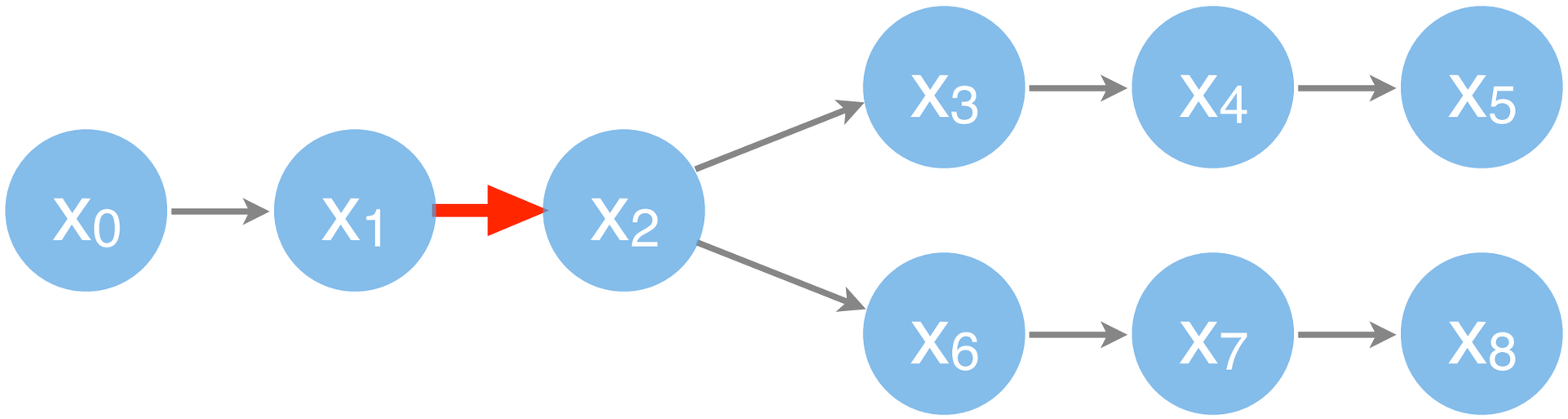


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

Trees

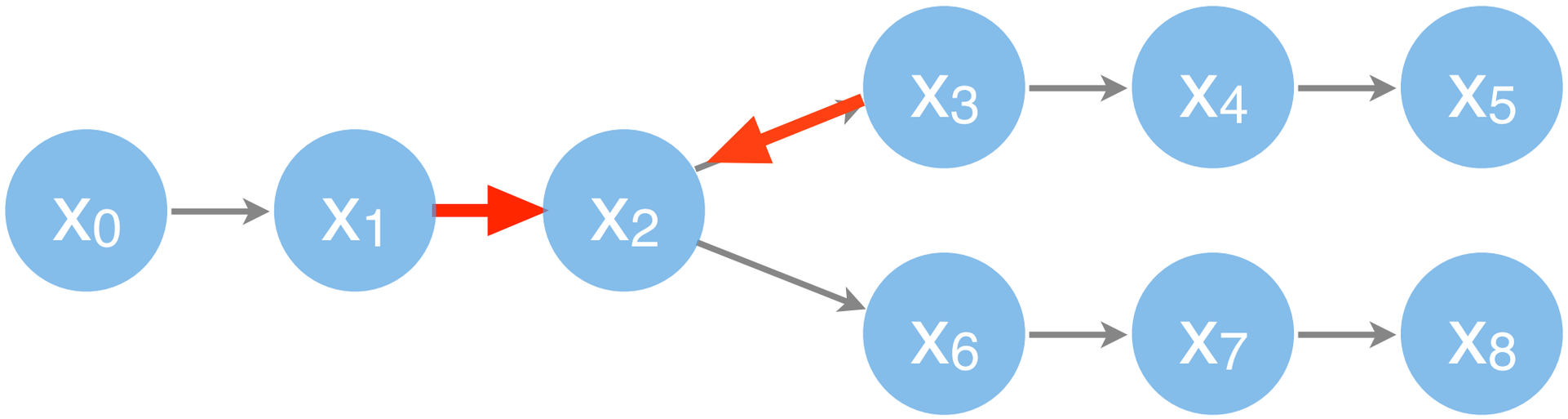


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

Trees

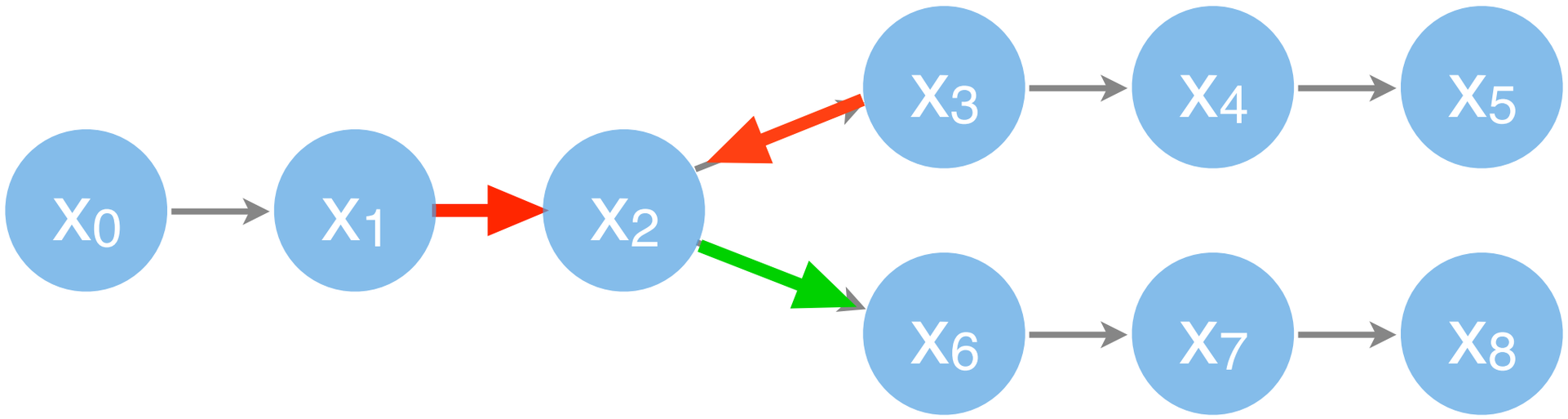


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

Trees

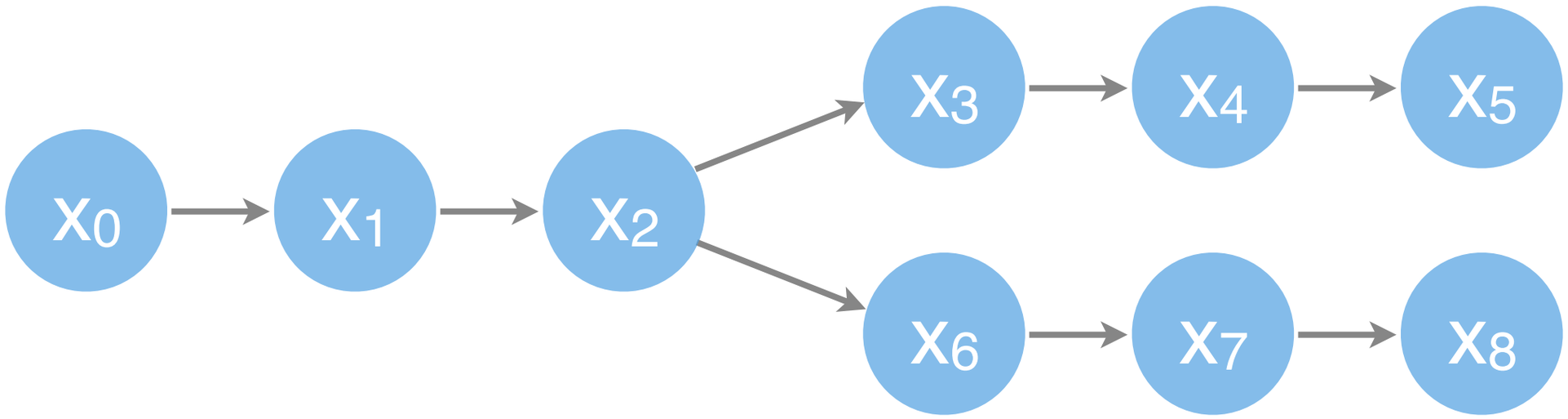


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

Trees



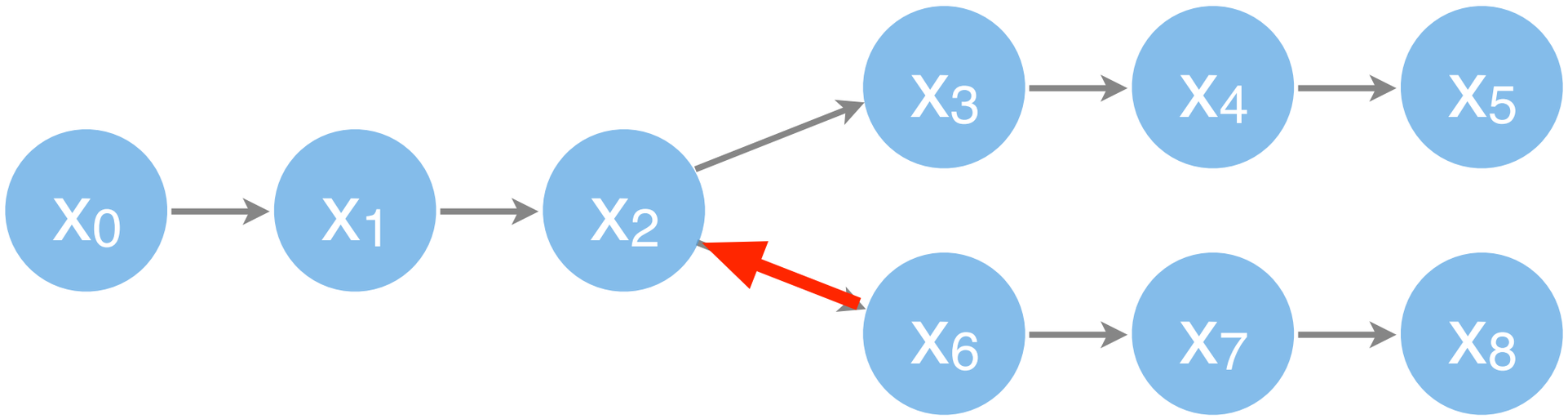
- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

$$m_{2\to 1}(x_1) = \sum_{x_2} m_{3\to 2}(x_2) m_{6\to 2}(x_2) f(x_1, x_2)$$

Trees



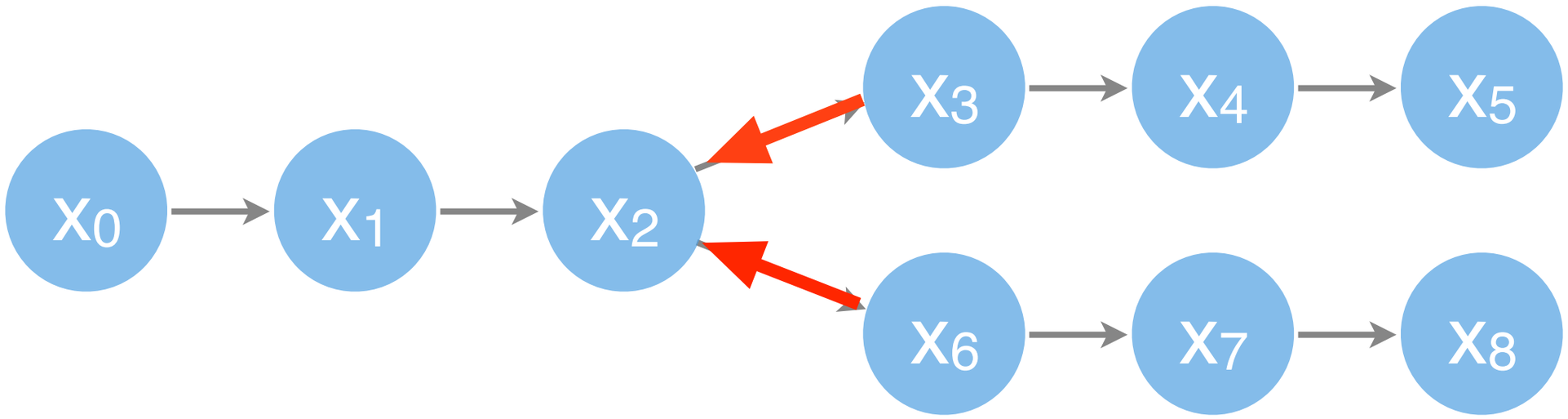
- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

$$m_{2\to 1}(x_1) = \sum_{x_2} m_{3\to 2}(x_2) m_{6\to 2}(x_2) f(x_1, x_2)$$

Trees



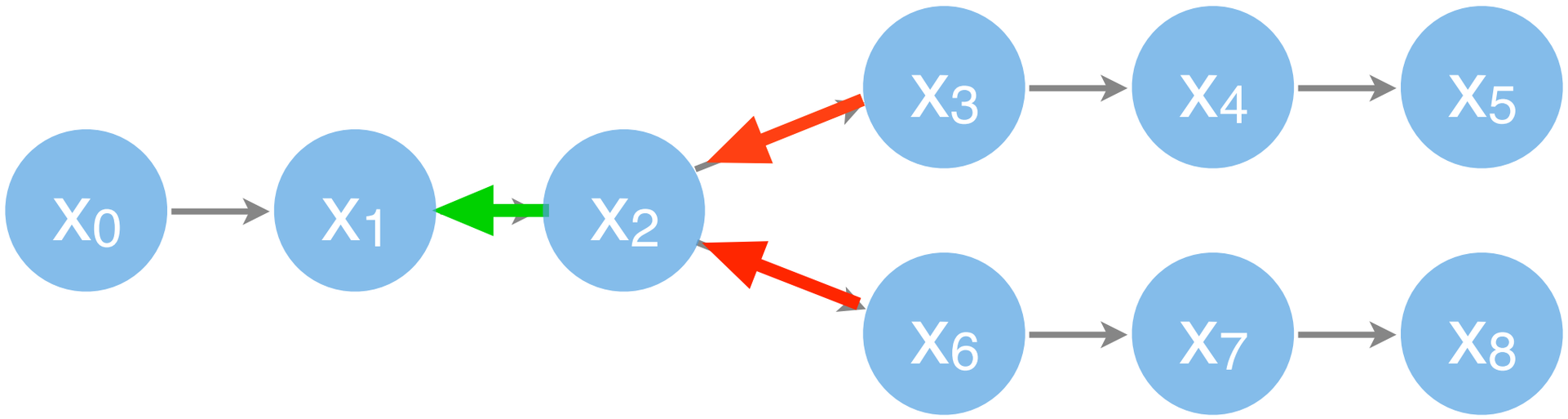
- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

$$m_{2\to 1}(x_1) = \sum_{x_2} m_{3\to 2}(x_2) m_{6\to 2}(x_2) f(x_1, x_2)$$

Trees



- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

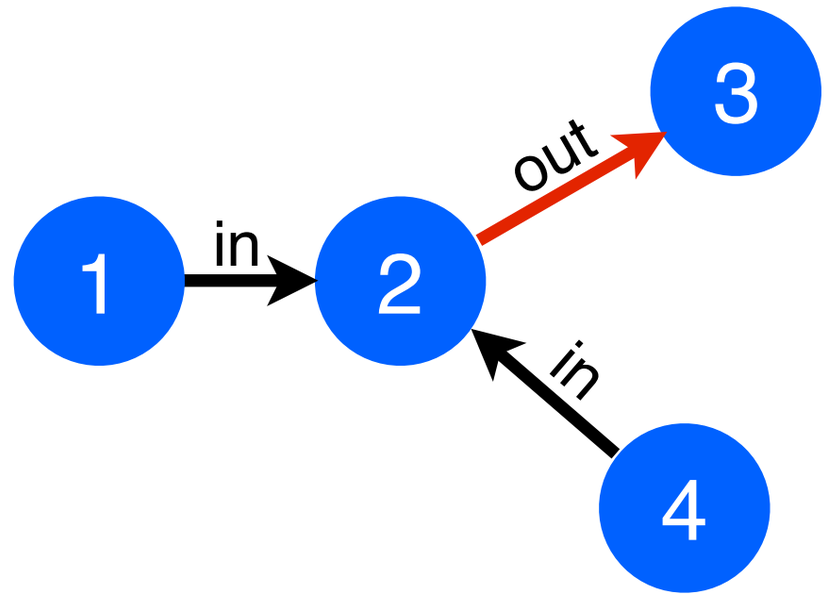
$$m_{2\to 3}(x_3) = \sum_{x_2} m_{1\to 2}(x_2) m_{6\to 2}(x_2) f(x_2, x_3)$$

$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

$$m_{2\to 1}(x_1) = \sum_{x_2} m_{3\to 2}(x_2) m_{6\to 2}(x_2) f(x_1, x_2)$$

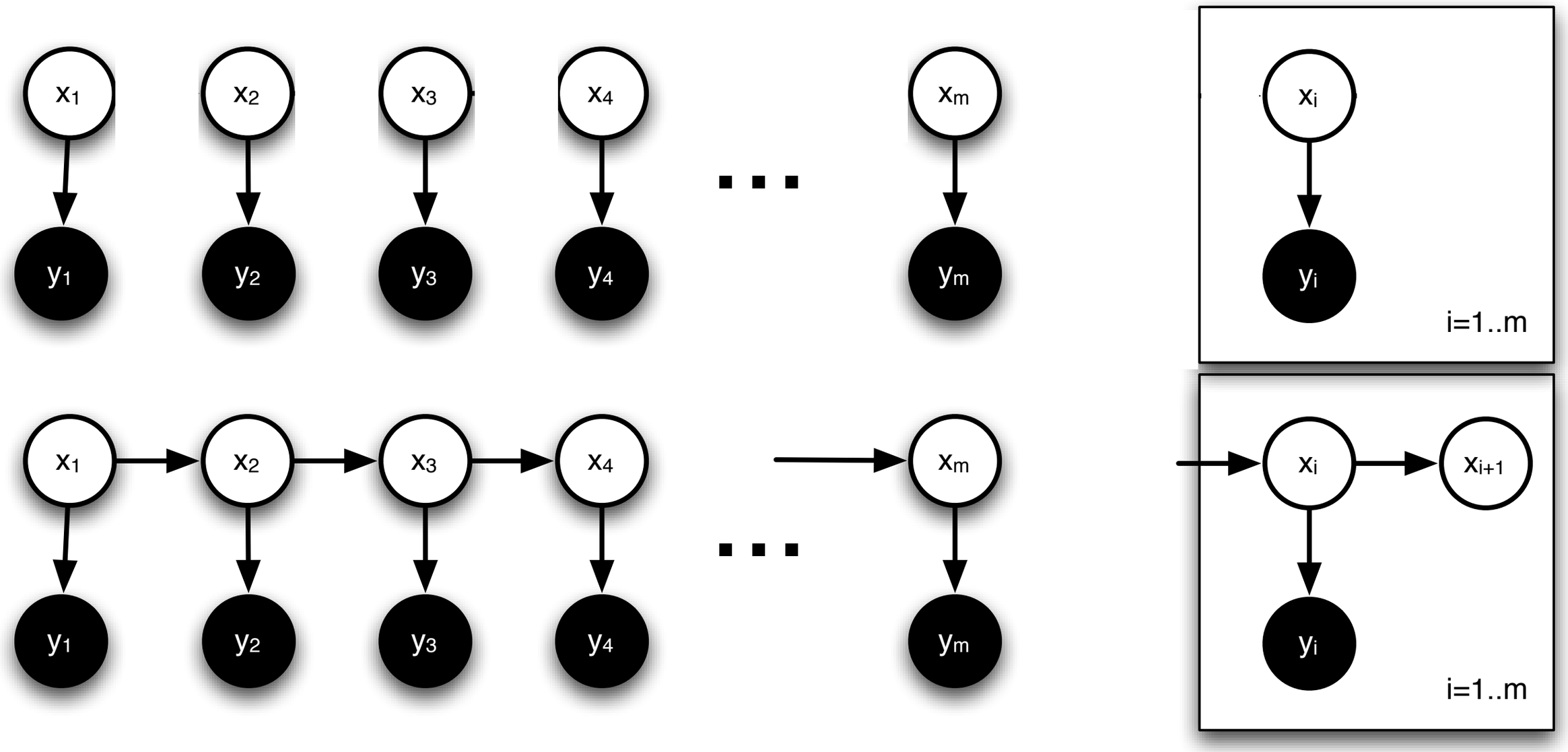
Summary

- Markov chains
  - Present only depends on recent past
  - Higher order - longer history.
- Dynamic programming
  - Exponential if brute force.
  - Linear in chain if we iterate.
  - For junctions treat like chains but integrate signals from all sources.
  - Exponential in the history size.



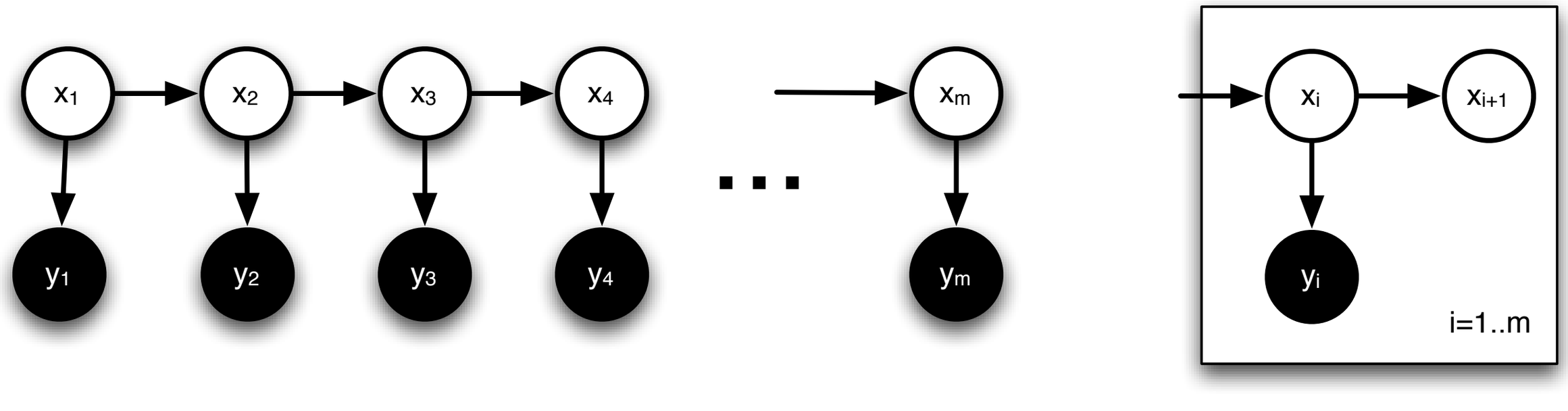
Hidden Markov Models

Clustering and Hidden Markov Models



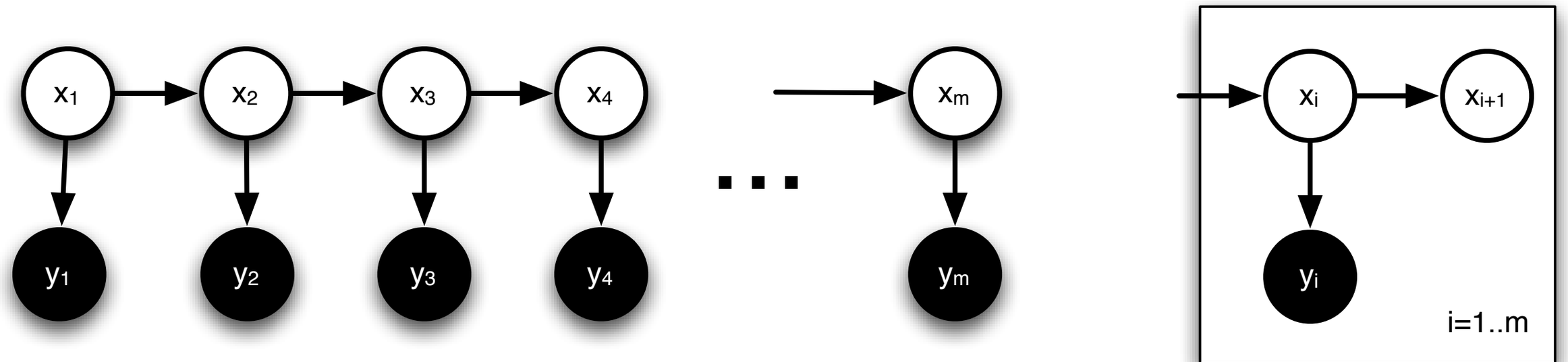
- Clustering - no dependence between observations
- Hidden Markov Model - dependence between states

Applications



- Speech recognition (soundltext)
- Optical character recognition (writingltext)
- Gene finding (DNA sequencelgenes)
- Activity recognition (accelerometerlactivity)

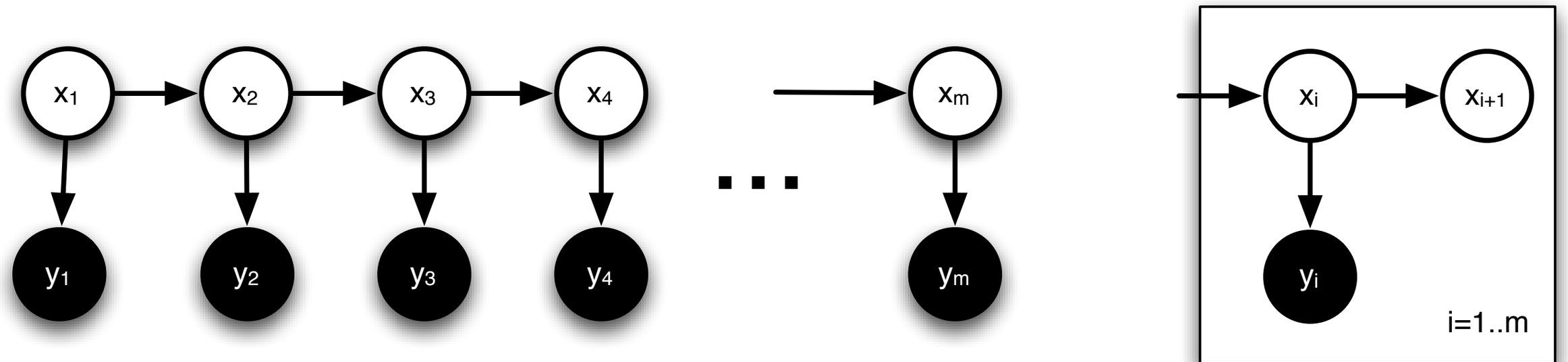
Inference Tasks



$$p(x,y) = p(y_1)\left[\prod_{i=1}^{m-1} p(y_{i+1}|y_i)p(x_i|y_i)\right] p(x_m|y_m)$$

- Infer latent variables p(xly), extend sequence
- Estimate distributions $p(y_i|x_i)$ and $p(x_{i+1}|x_i)$

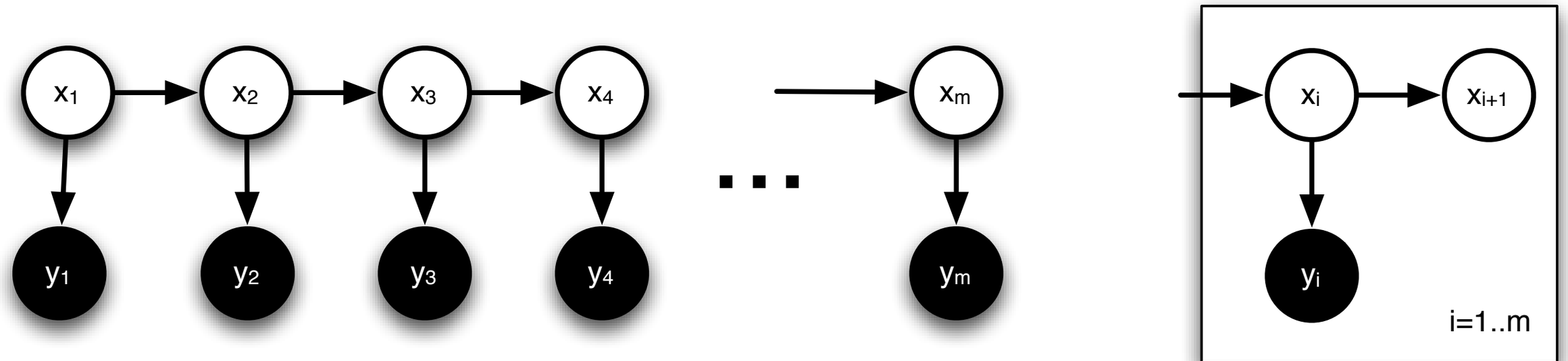
Recall Dynamic Programming



$$p(x,y) = p(y_1)\left[\prod_{i=1}^{m-1} p(y_{i+1}|y_i)p(x_i|y_i)\right] p(x_m|y_m)$$

- Observe y, maybe also part of x
- Likely value for any x_i in the chain means summing over all other variables x_j

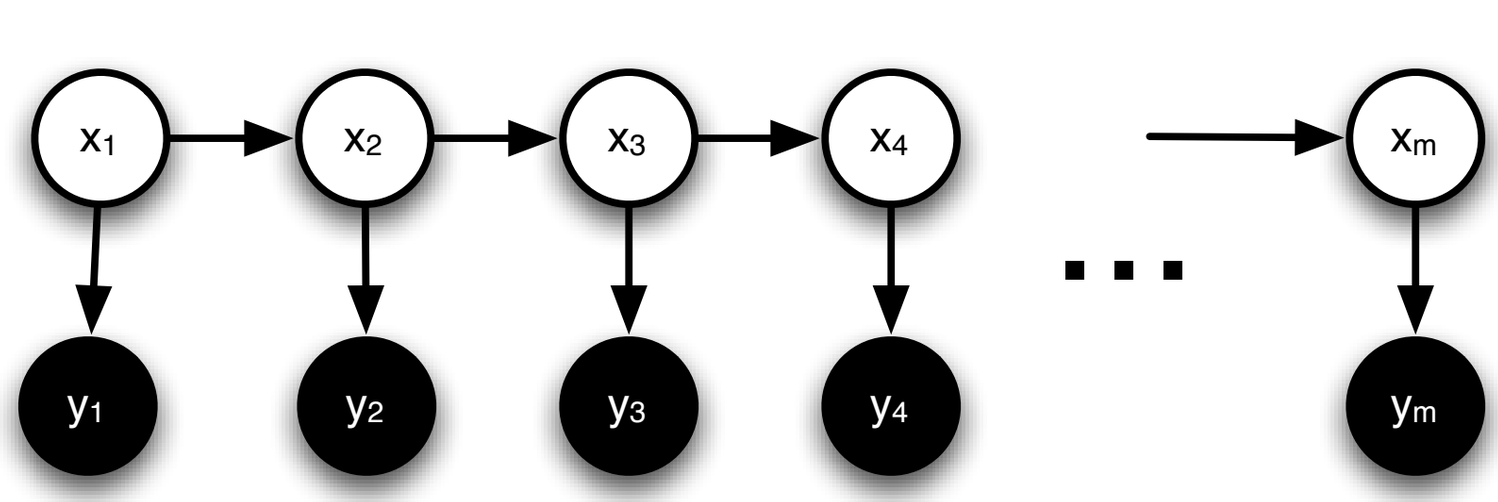
Dynamic Programming

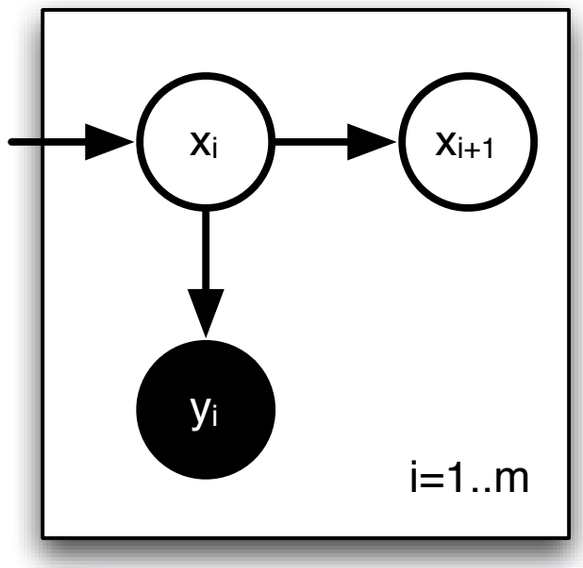


$$p(x|y) = \frac{p(x,y)}{\sum_{x'} p(x',y)} \text{ and } p(x_i|y) \propto \sum_{x_j : j<i} \sum_{x_j : j>i} p(x,y)$$

- Observe y, maybe also part of x
- Likely value for any x_i in the chain means summing over all other variables x_j

Dynamic Programming





$$l_1(x_1) = p(x_1)p(y_1|x_1)$$

$$l_{j+1}(x_{j+1}) = \sum_{x_j} l_j(x_j)p(x_{j+1}|x_j)p(y_j|x_j)$$

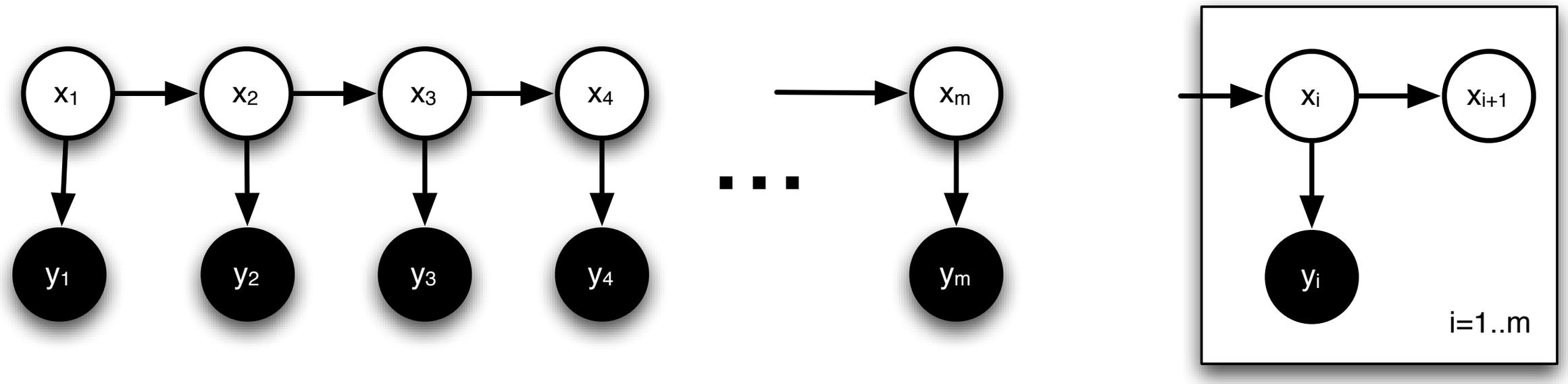
$$r_n(x_n) = 1$$

$$r_{j-1}(x_{j-1}) = \sum_{x_j} r_j(x_j)p(y_j|x_j)p(x_j|x_{j-1})$$

$$p(x_i|\text{rest}) \propto l_i(x_i)r_i(x_i)p(y_i|x_i)$$

Same algorithm for finding most likely values (+,*) (max,+)

Dynamic Programming



$$l_1(x_1) = \log p(x_1) + \log p(y_1|x_1)$$

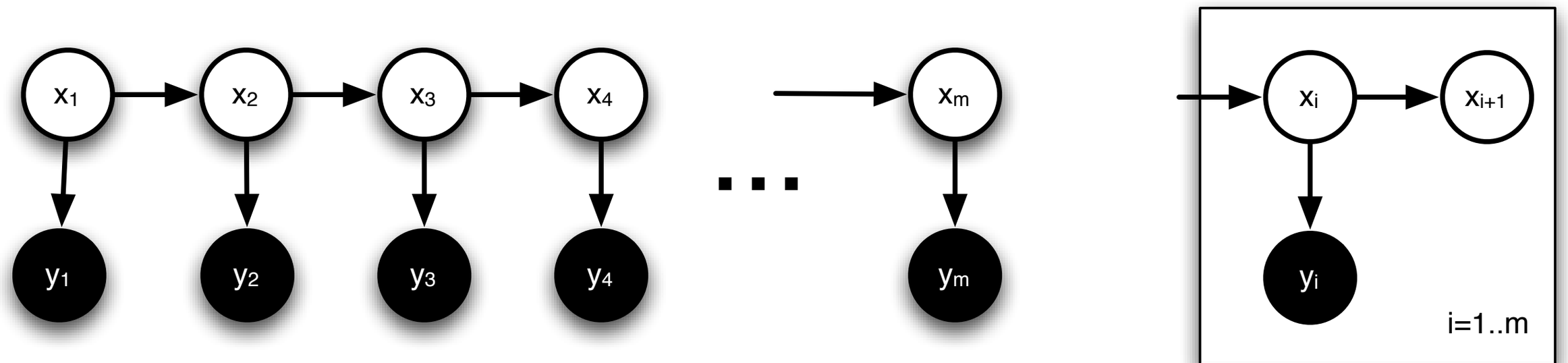
$$l_{j+1}(x_{j+1}) = \max_{x_j} l_j(x_j) + \log p(x_{j+1}|x_j) + \log p(y_j|x_j)$$

$$r_n(x_n) = 1$$

$$r_{j-1}(x_{j-1}) = \max_{x_j} r_j(x_j) + \log p(y_j|x_j) + \log p(x_j|x_{j-1})$$

$$\hat{x_i} = \operatorname{argmax} l_i(x_i) + r_i(x_i) + \log p(y_i|x_i)$$

Inference



$$p(x, y) = p(x_1) \left[\prod_{i=1}^{m-1} p(x_{i+1}|x_i) p(y_i|x_i) \right] p(y_m|x_m)$$

- What if we want to estimate the probabilities directly? Log-likelihood is nonconvex!

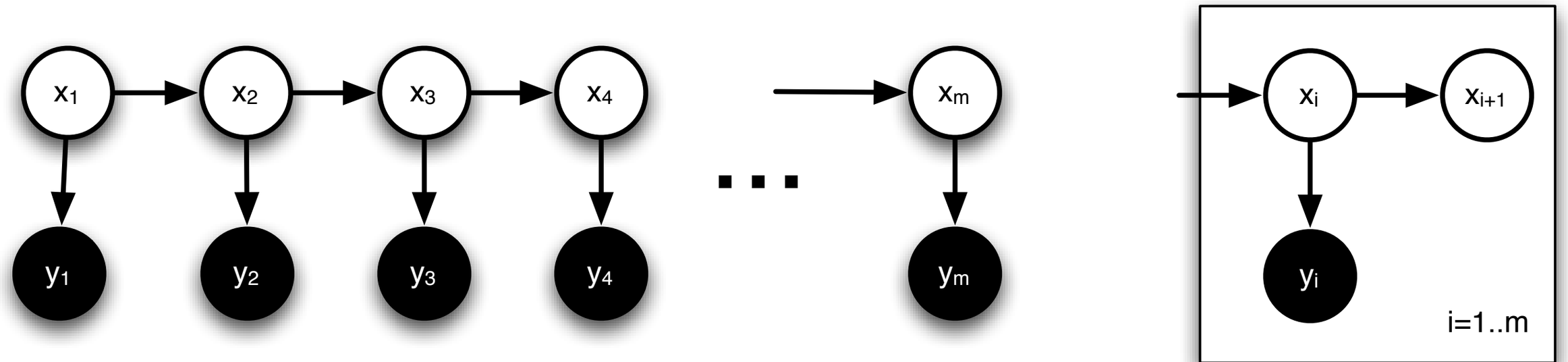
Variational Approximation

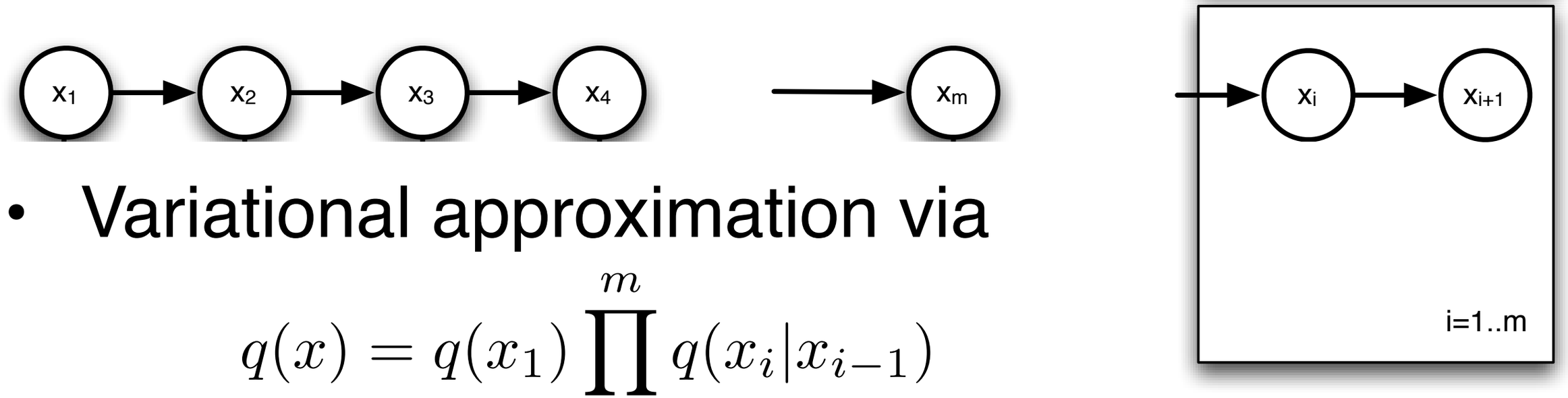
- Lower bound on log-likelihood

$$\log p(y;\theta) \geq \int dq(x) \log p(x,y;\theta) - \int dq(x) \log q(x)$$

  - Inequality holds for any q (equality for p(xly)=q(x))
    - Find q within subset Q to tighten inequality
    - Find parameters to maximize for fixed q
- Inference for graphical models where joint probability computation is infeasible

Variational Approximation





- Variational approximation via

$$q(x) = q(x_1) \prod_{i=2}^{m} q(x_i | x_{i-1})$$

- Compute p(xly) via dynamic programming

Variational Principle in Action

- Initialize parameters somehow
- Set $q(x) = p(x|y)$
  Dynamic programming yields chain
- Maximizing the log-likelihod w.r.t. q

$$\log p(y;\theta) \geq \int dq(x) \log p(x,y;\theta) - \int dq(x) \log q(x)$$

$$p(x,y) = p(x_1) \left[\prod_{i=1}^{m-1} p(x_{i+1}|x_i) p(y_i|x_i) \right] p(y_m|x_m)$$

$q(x_1)$

$q(x_{i+1}|x_i)$

$q(x_i)$

Parameter Estimation

$$\mathbf{E}_{x\sim q}\left[\log p(x,y;\theta)\right] = \mathbf{E}_{x_1\sim q}\left[\log p(x_1;\theta)\right] + \sum_{i=1}^{m}\mathbf{E}_{x_i\sim q}\left[\log p(y_i|x_i;\theta)\right]$$

$$+\sum_{i=1}^{m-1}\mathbf{E}_{(x_i,x_{i+1})\sim q}\left[\log p(x_{i+1}|x_i;\theta)\right]$$

- $p(x_1)$
  Since we have $\mathbf{E}_{q(x_1)}\left[\log p(x_1)\right]$ set $p(x_1) = q(x_1)$
- $p(y_i|x_i)$
  Same as clustering
  e.g. for Gaussians

$$\mu_x = \frac{1}{n_x}\sum_{i=1}^{m} q_i(x) y_i$$

$$\Sigma_x = \frac{1}{n_x}\sum_{i=1}^{m} q_i(x) y_i y_i^\top - \mu_x \mu_x^\top$$

Parameter Estimation

$$\mathbf{E}_{x\sim q}\left[\log p(x,y;\theta)\right] = \mathbf{E}_{x_1\sim q}\left[\log p(x_1;\theta)\right] + \sum_{i=1}^{m} \mathbf{E}_{x_i\sim q}\left[\log p(y_i|x_i;\theta)\right]$$

$$+ \sum_{i=1}^{m-1} \mathbf{E}_{(x_i,x_{i+1})\sim q}\left[\log p(x_{i+1}|x_i;\theta)\right]$$

- Maximum Likelihood estimate

$$\sum_{i=1}^{m-1} q(a,b)\log p(a|b)$$

$$\text{hence } p(a|b) = \frac{\sum_{i=1}^{m-1} q(a,b)}{\sum_{i=1}^{m-1} q(b)}$$

effective sample

Smoothed Estimates

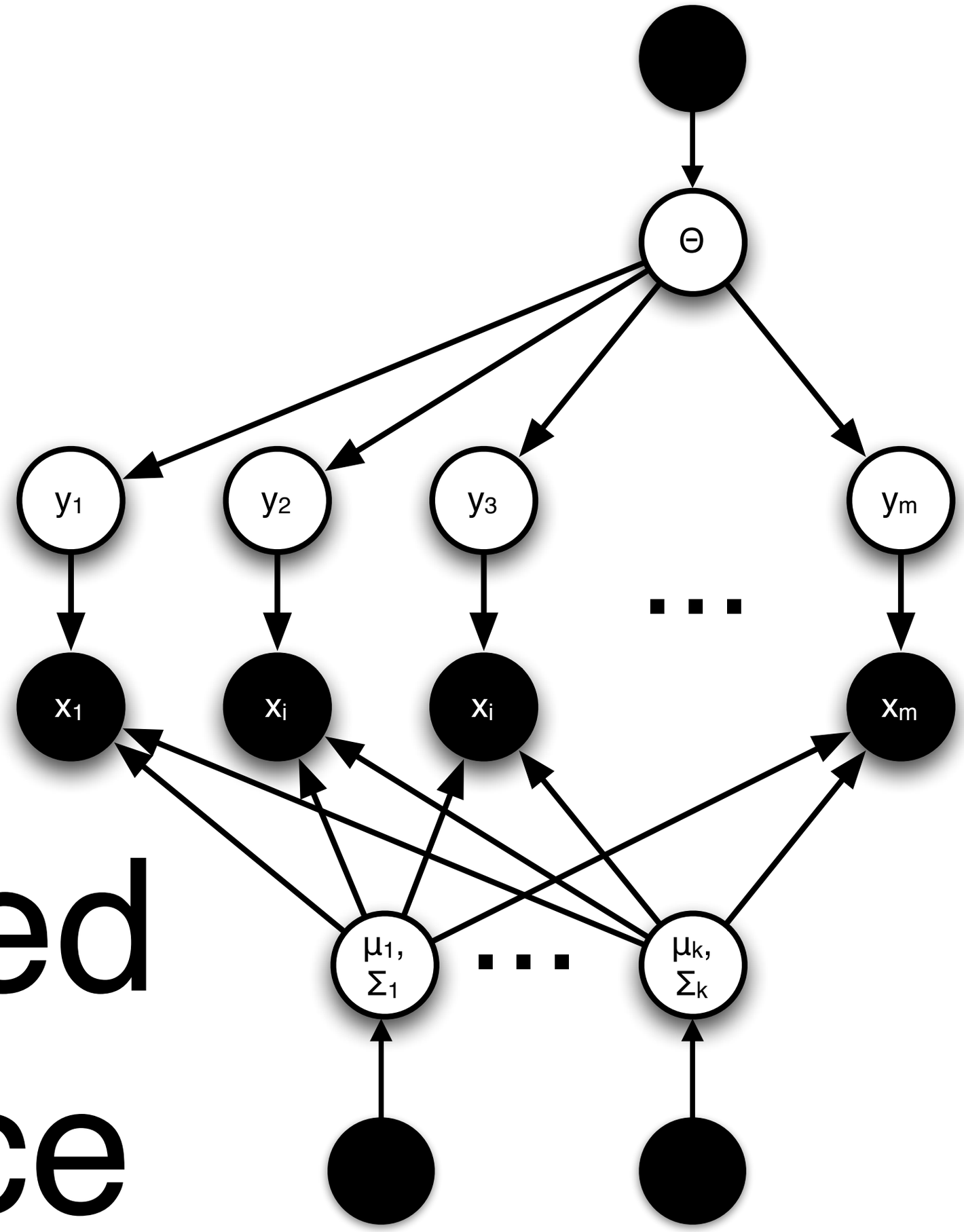
- Laplace prior on latent state distribution
  - Uniform distribution over states
  - Alternatively assume that state remains

$$p(a|b) = \frac{n_{a|b} + \sum_{i=1}^{m-1} q(a,b)}{n_b + \sum_{i=1}^{m-1} q(b)}$$

transition smoother

aggregate mass

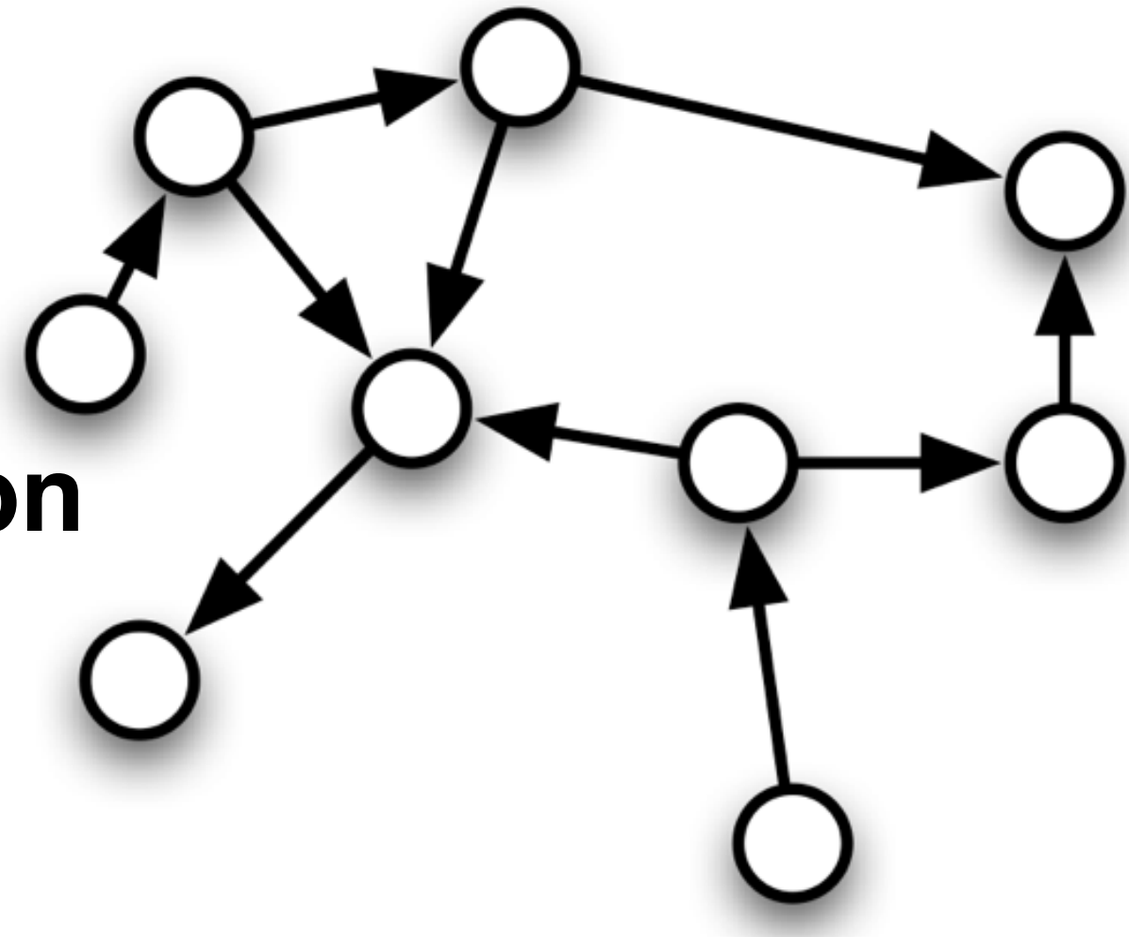
- Same trick for means and variances



Advanced Inference

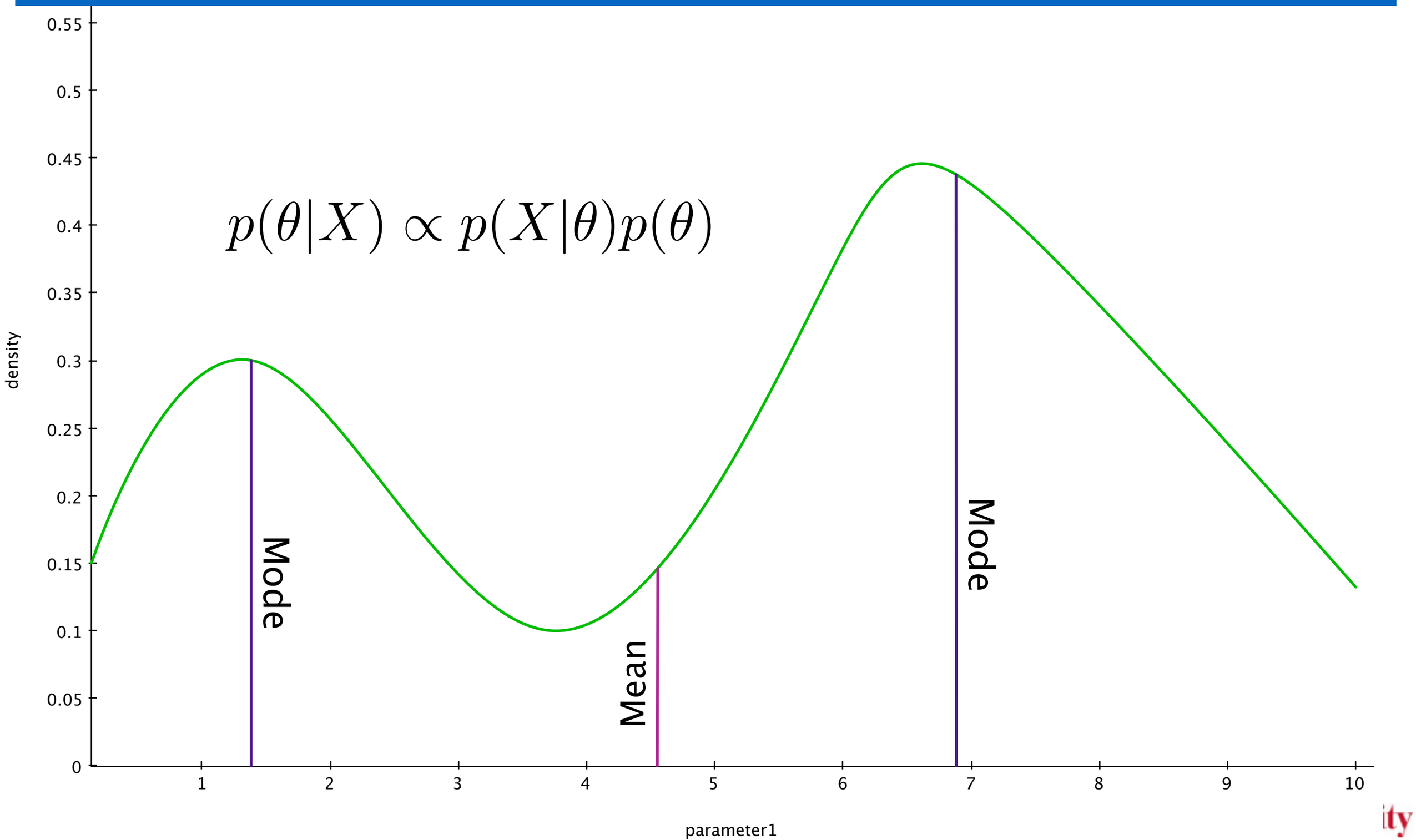
The Problem

- Message passing leads to loops (not arrows)
- **Combine Variables**
  Junction Tree
- **Ignore it**
  Loopy Belief Propagation
- **Variational Approximation**
  Simpler distribution
  without loops
- **Gibbs Sampling**
  Draw from one variable at a time



Is maximization (always) good?

$$p(\theta|X) \propto p(X|\theta)p(\theta)$$



Sampling

- Key idea
  - Want accurate distribution of the posterior
  - Sample from posterior distribution rather than maximizing it
- Problem - direct sampling is usually intractable
- Solutions
  - Markov Chain Monte Carlo (complicated)
  - Gibbs Sampling (somewhat simpler)

$$x \sim p(x|x') \text{ and then } x' \sim p(x'|x)$$

Gibbs sampling

- Gibbs sampling:
  - In most cases direct sampling not possible
  - Draw one set of variables at a time

	0.45	0.05
	0.05	0.45

Gibbs sampling

- Gibbs sampling:
  - In most cases direct sampling not possible
  - Draw one set of variables at a time

	0.45	0.05
	0.05	0.45

(b,g) - draw p(.,g)

Gibbs sampling

- Gibbs sampling:
  - In most cases direct sampling not possible
  - Draw one set of variables at a time

		
	0.45	0.05
	0.05	0.45

(b,g) - draw p(.,g)
(g,g) - draw p(g,.)

Gibbs sampling

- Gibbs sampling:
  - In most cases direct sampling not possible
  - Draw one set of variables at a time

	0.45	0.05
	0.05	0.45

(b,g) - draw p(.,g)
(g,g) - draw p(g,.)
(g,g) - draw p(.,g)

Gibbs sampling

- Gibbs sampling:
  - In most cases direct sampling not possible
  - Draw one set of variables at a time

	0.45	0.05
	0.05	0.45

(b,g) - draw p(.,g)
(g,g) - draw p(g,.)
(g,g) - draw p(.,g)
(b,g) - draw p(b,.)

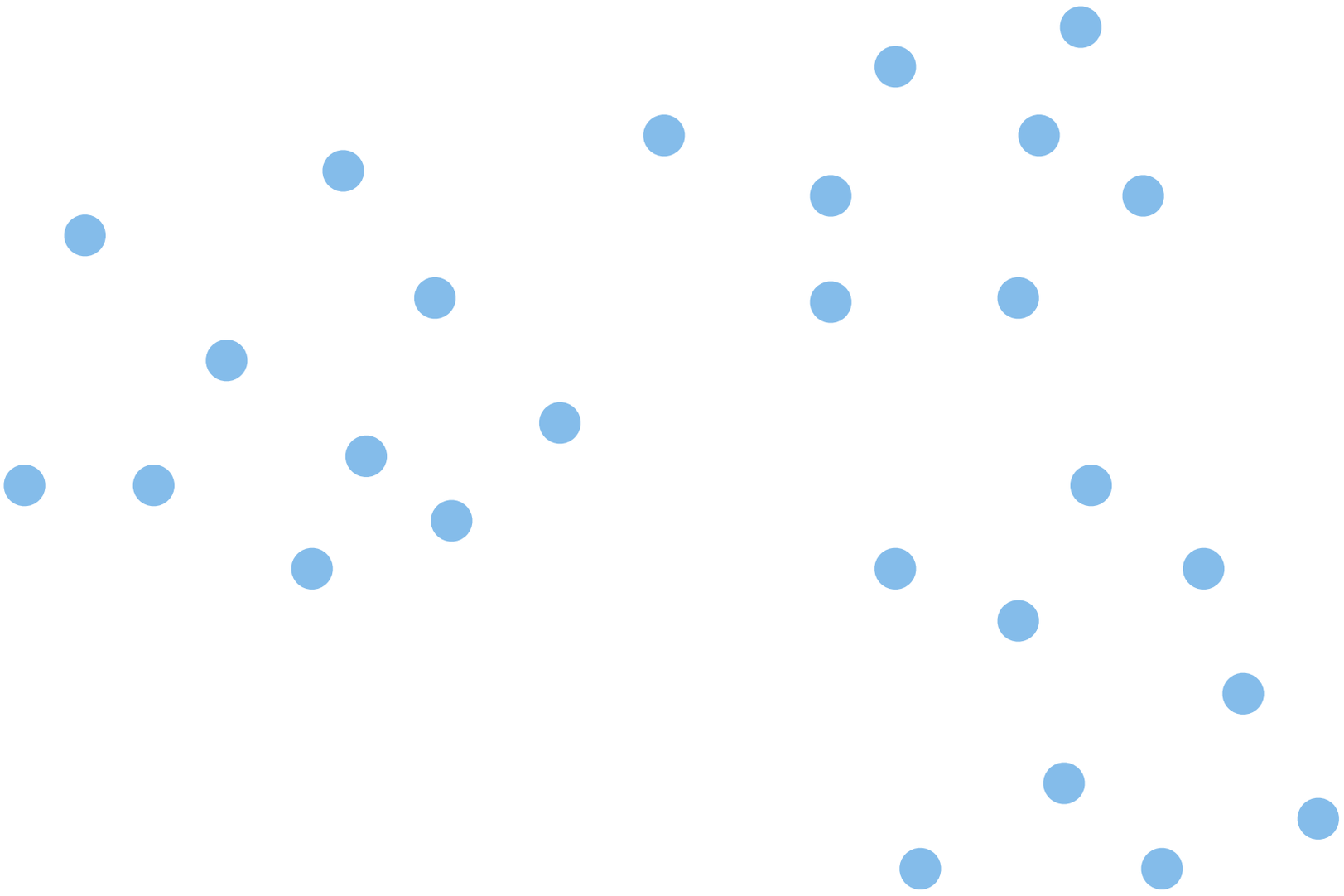
Gibbs sampling

- Gibbs sampling:
  - In most cases direct sampling not possible
  - Draw one set of variables at a time

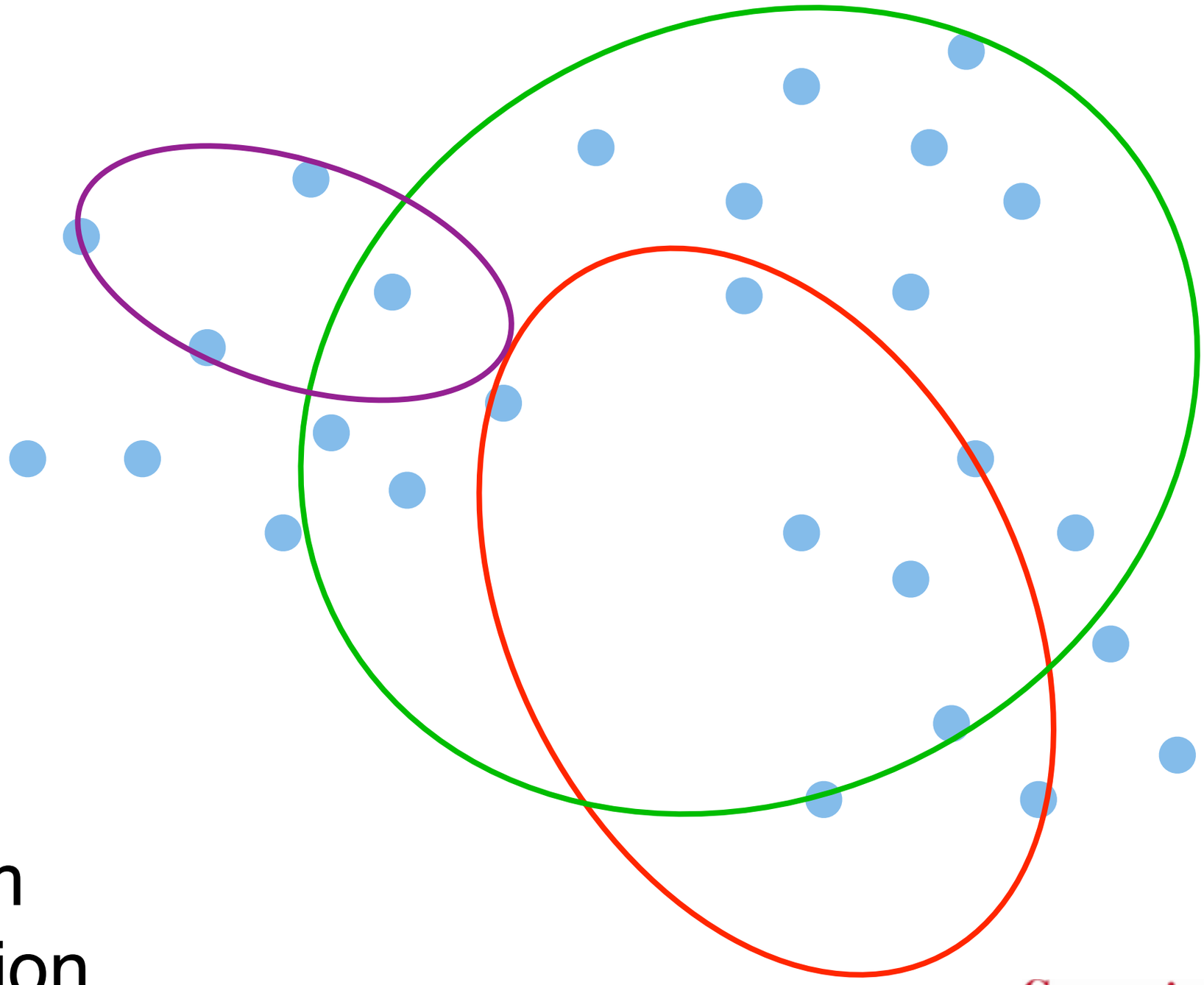
	0.45	0.05
	0.05	0.45

(b,g) - draw p(.,g)
(g,g) - draw p(g,.)
(g,g) - draw p(.,g)
(b,g) - draw p(b,.)
(b,b) ...

Gibbs sampling for clustering

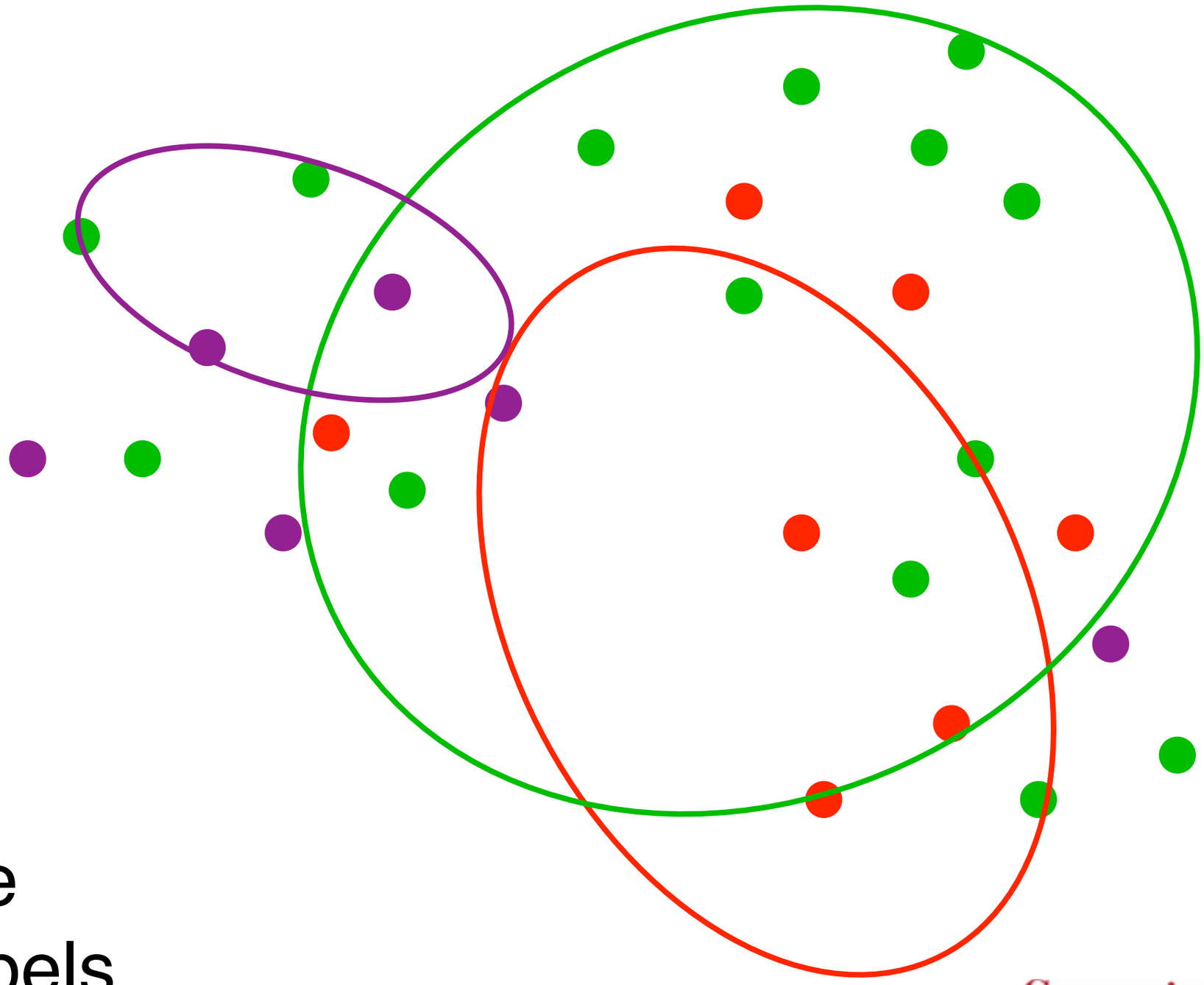


Gibbs sampling for clustering



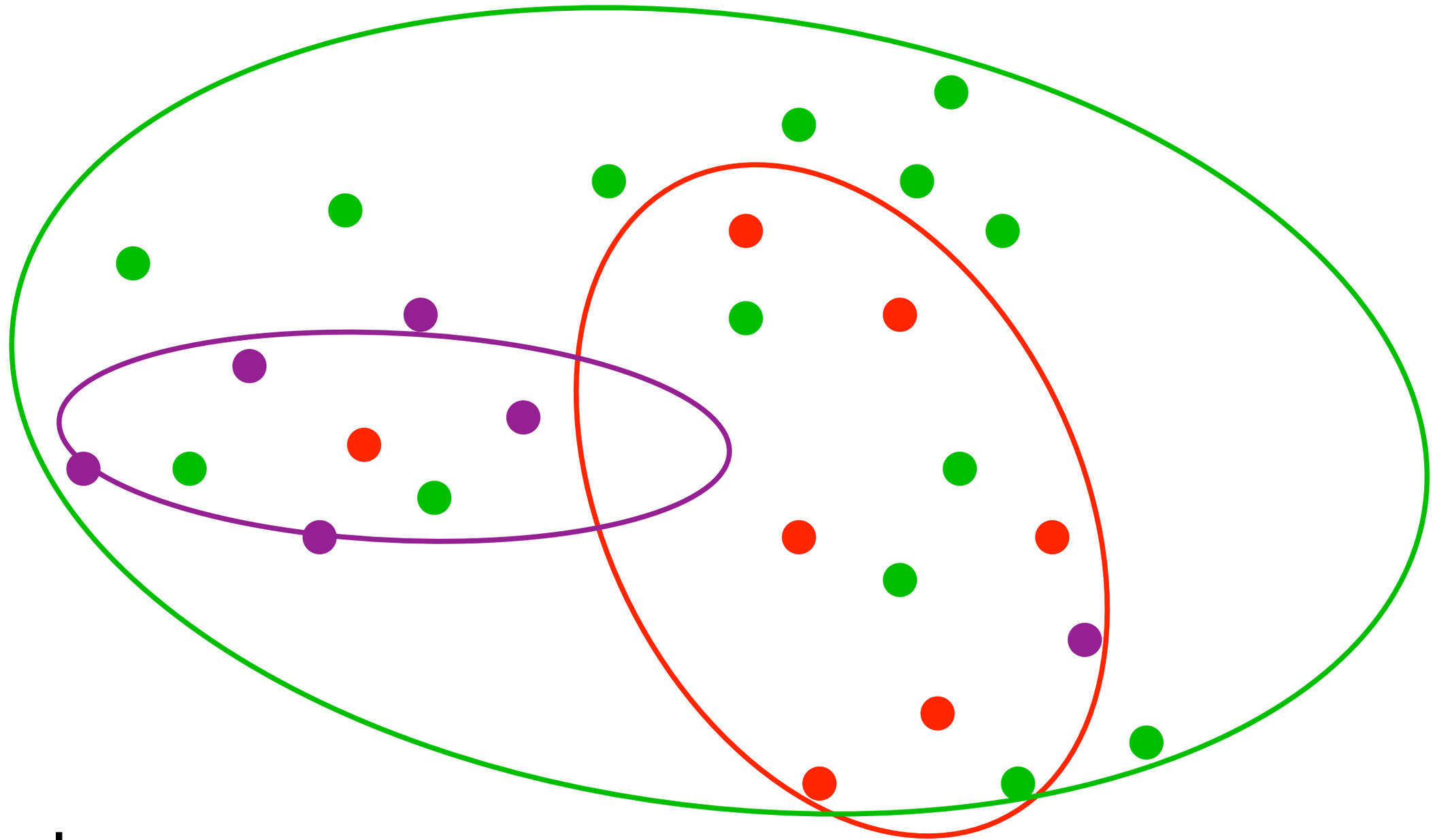
random
initialization

Gibbs sampling for clustering



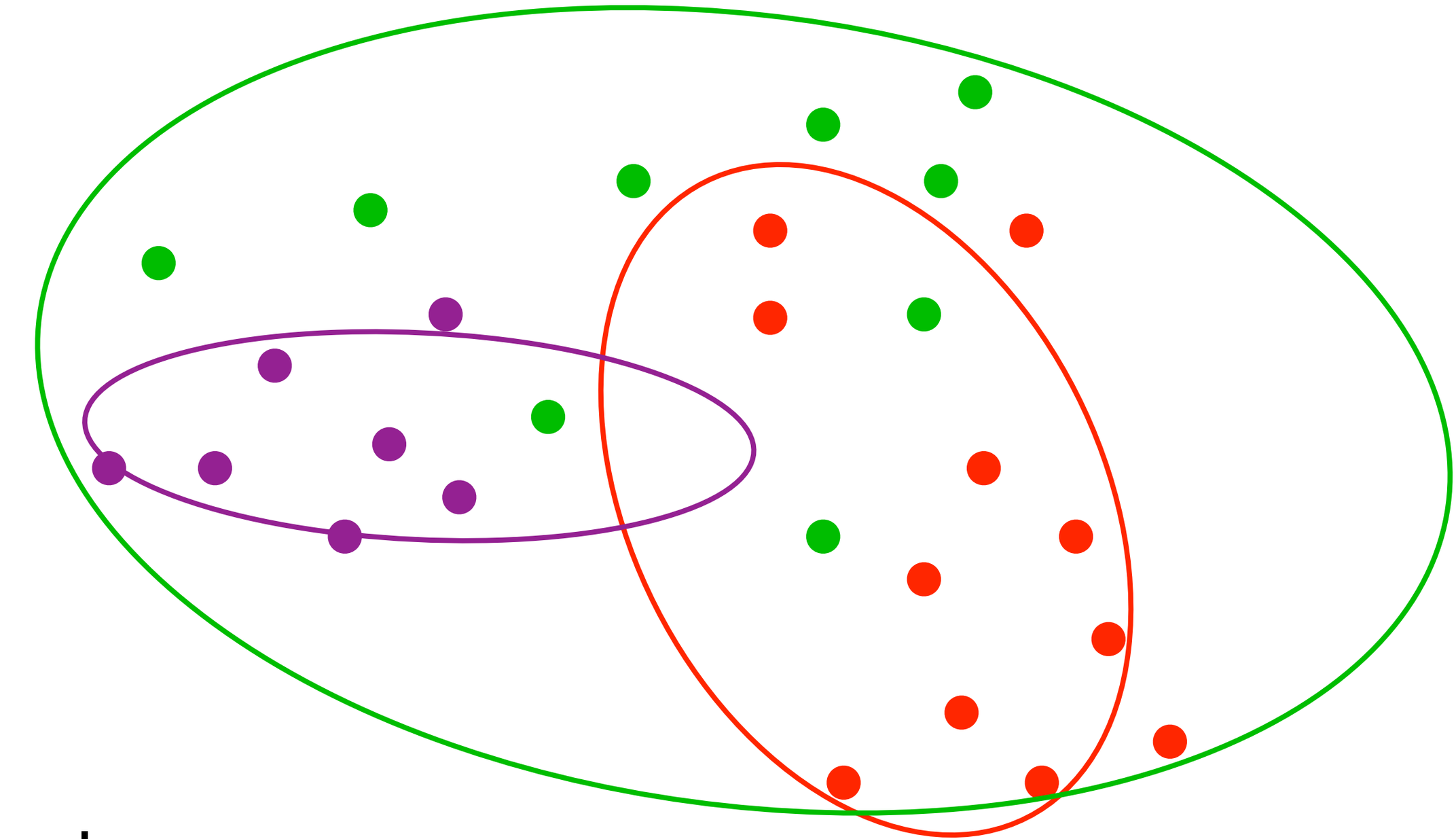
sample
cluster labels

Gibbs sampling for clustering



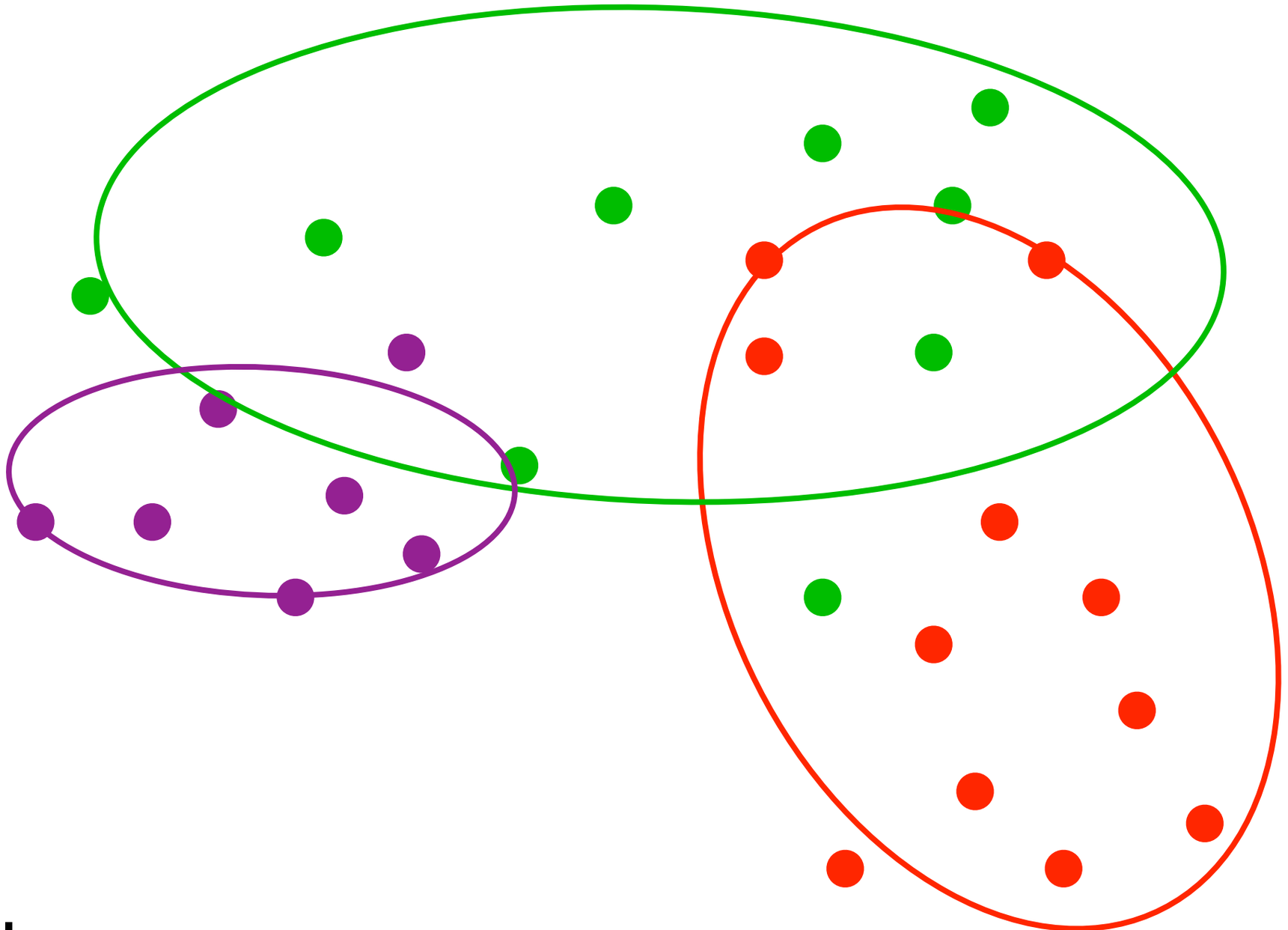
resample
cluster model

Gibbs sampling for clustering



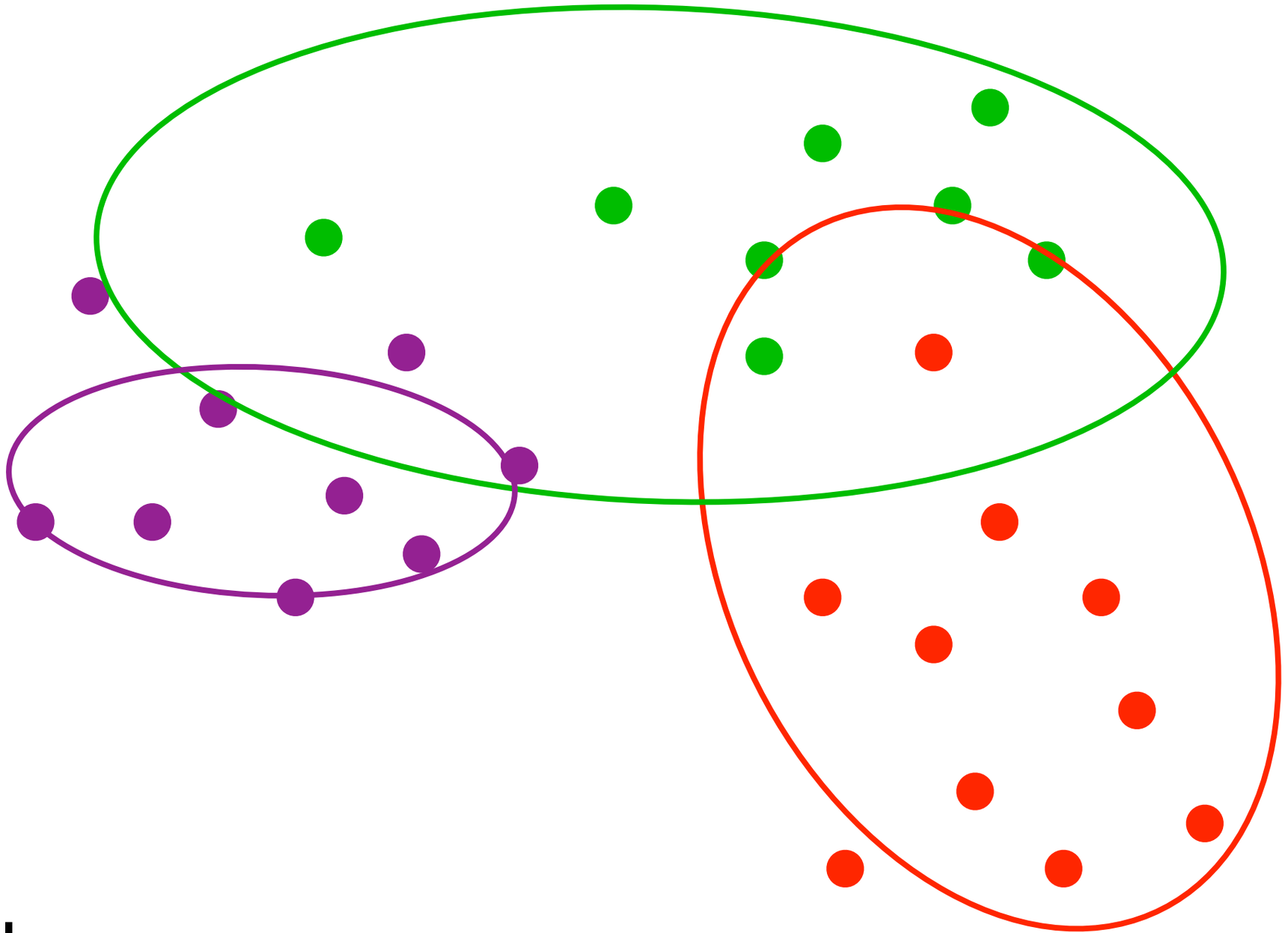
resample
cluster labels

Gibbs sampling for clustering



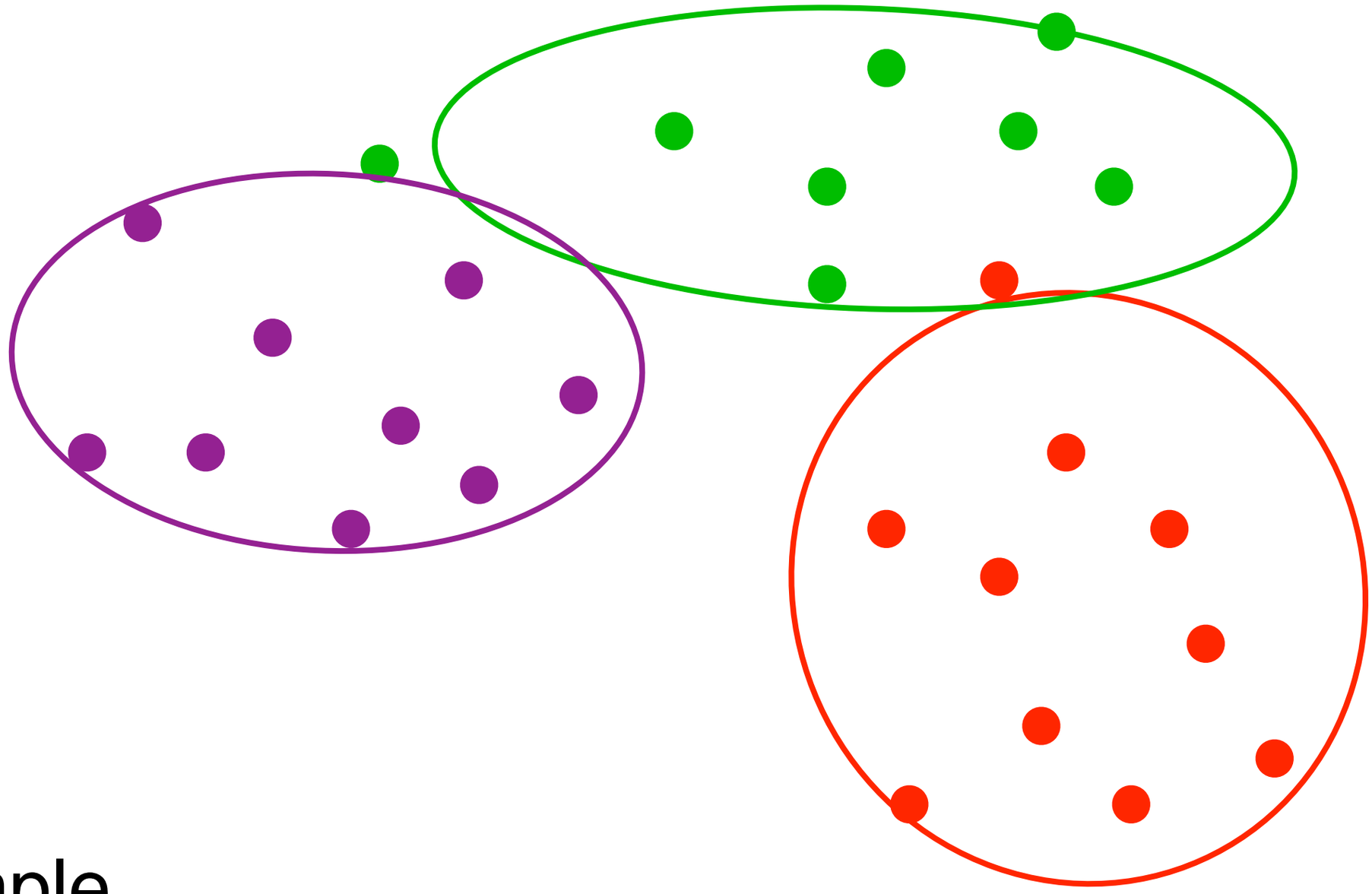
resample
cluster model

Gibbs sampling for clustering



resample
cluster labels

Gibbs sampling for clustering



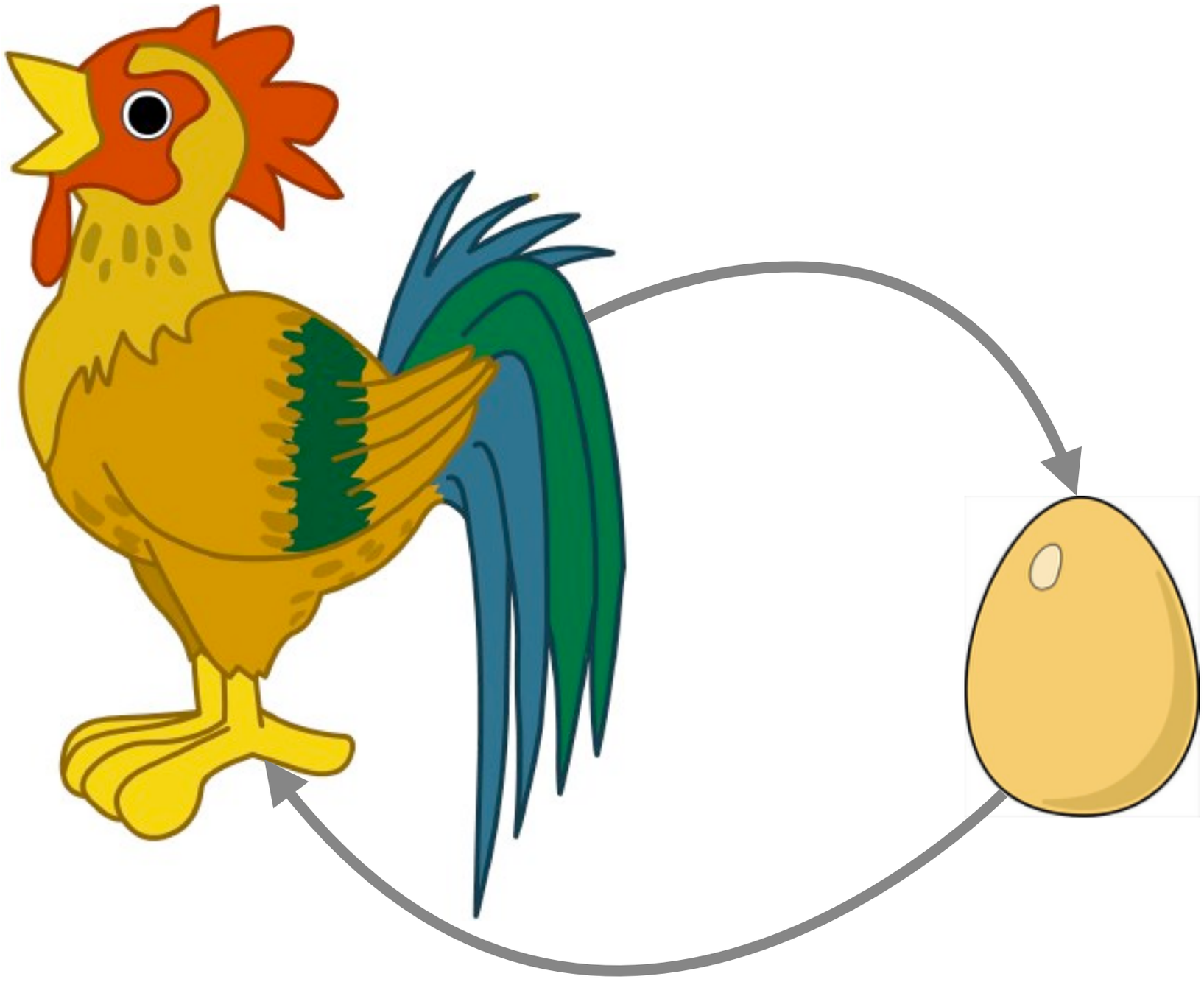
resample cluster model

e.g. Mahout Dirichlet Process Clustering

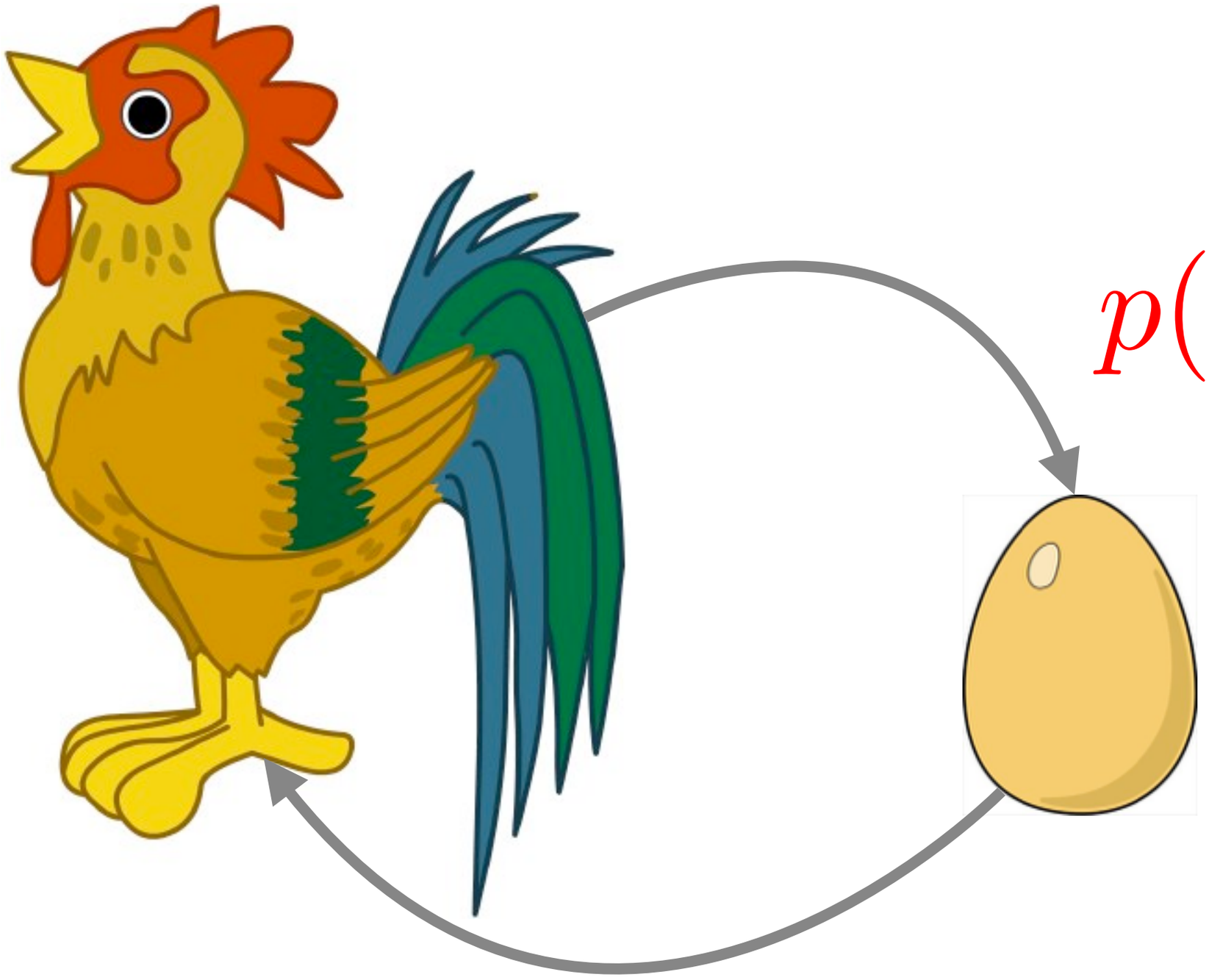
Blunting the arrows
Undirected Graphical Models

Carnegie Mellon University

Chicken and Egg

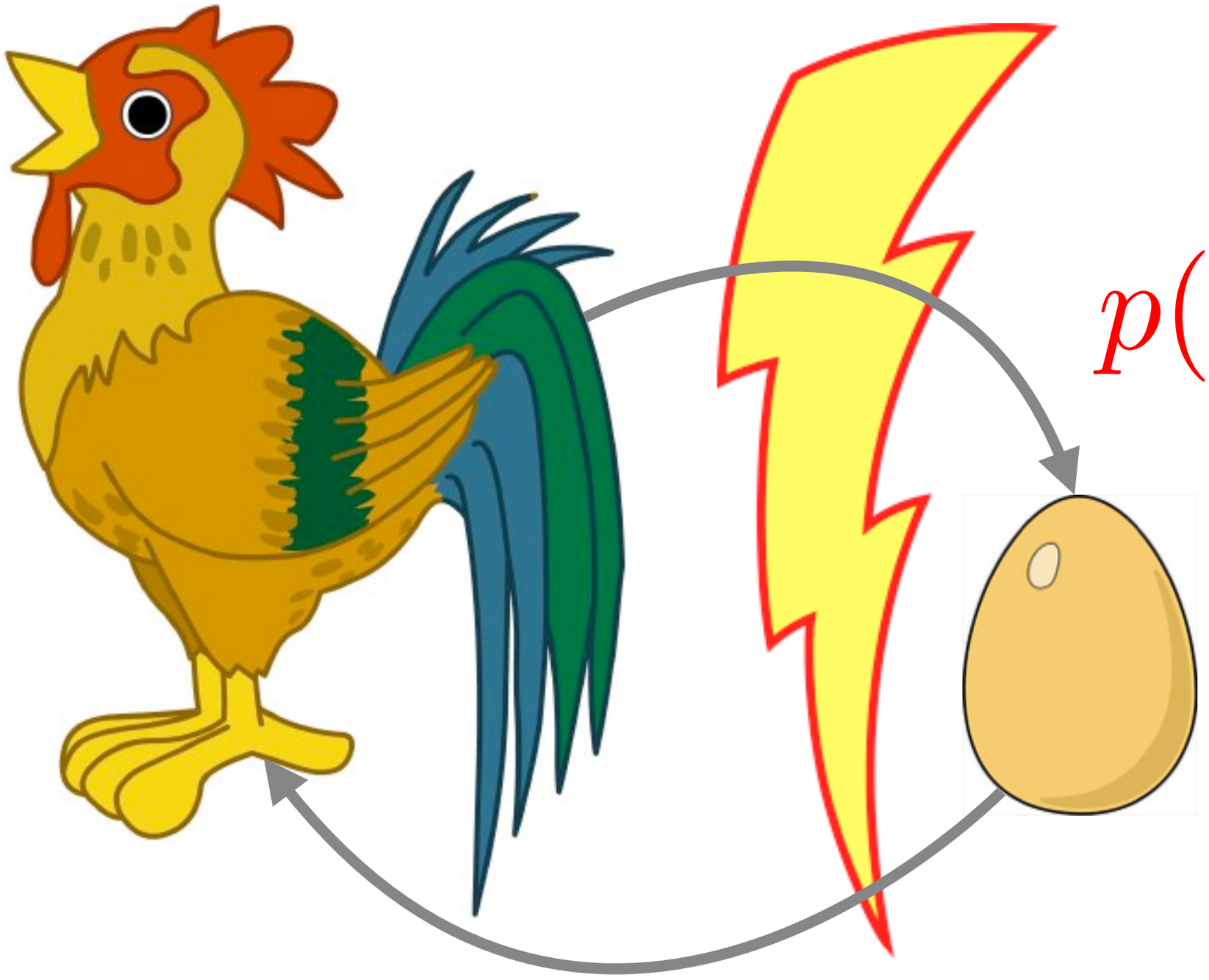


Chicken and Egg



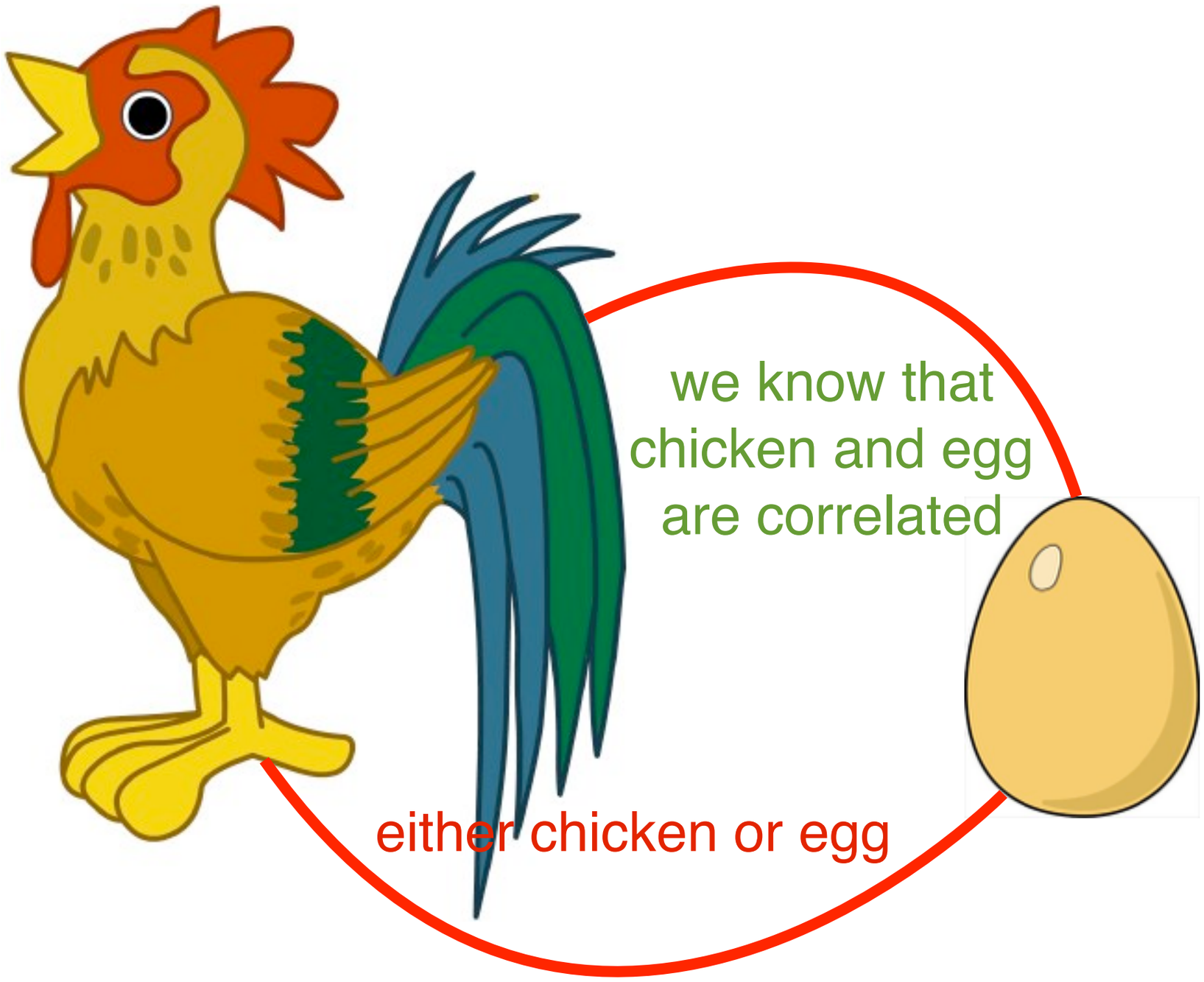
$$p(c|e)p(e|c)$$

Chicken and Egg

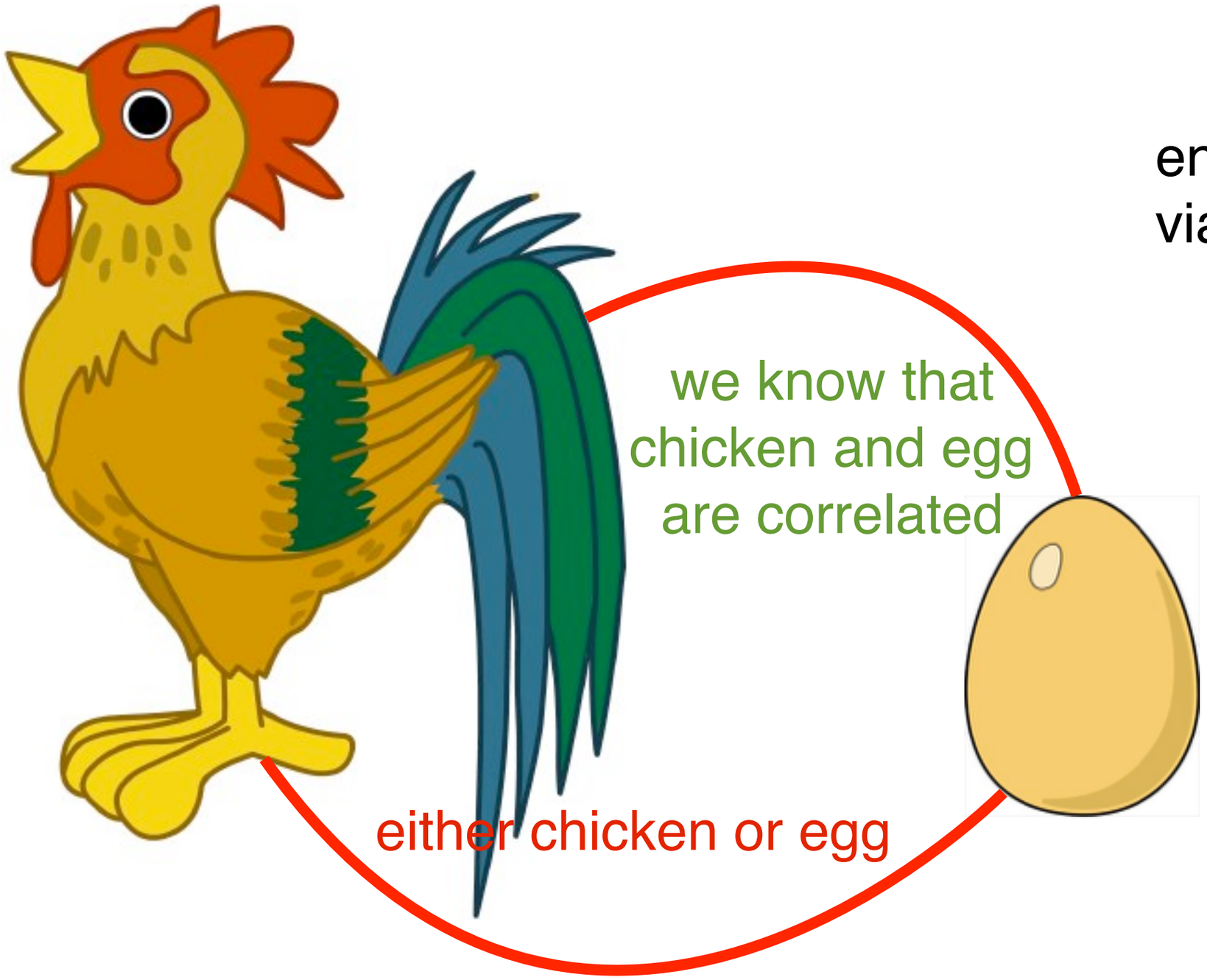


$$p(c|e)p(e|c)$$

Chicken and Egg



Chicken and Egg



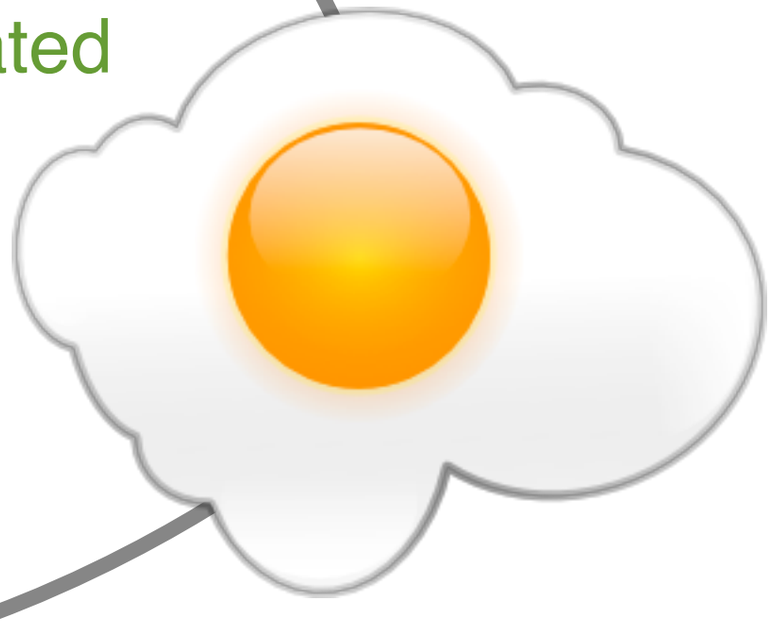
encode the correlation via the clique potential between c and e

$$p(c, e) \propto \exp \psi(c, e)$$

Chicken and Egg

we know that
chicken and egg
are correlated

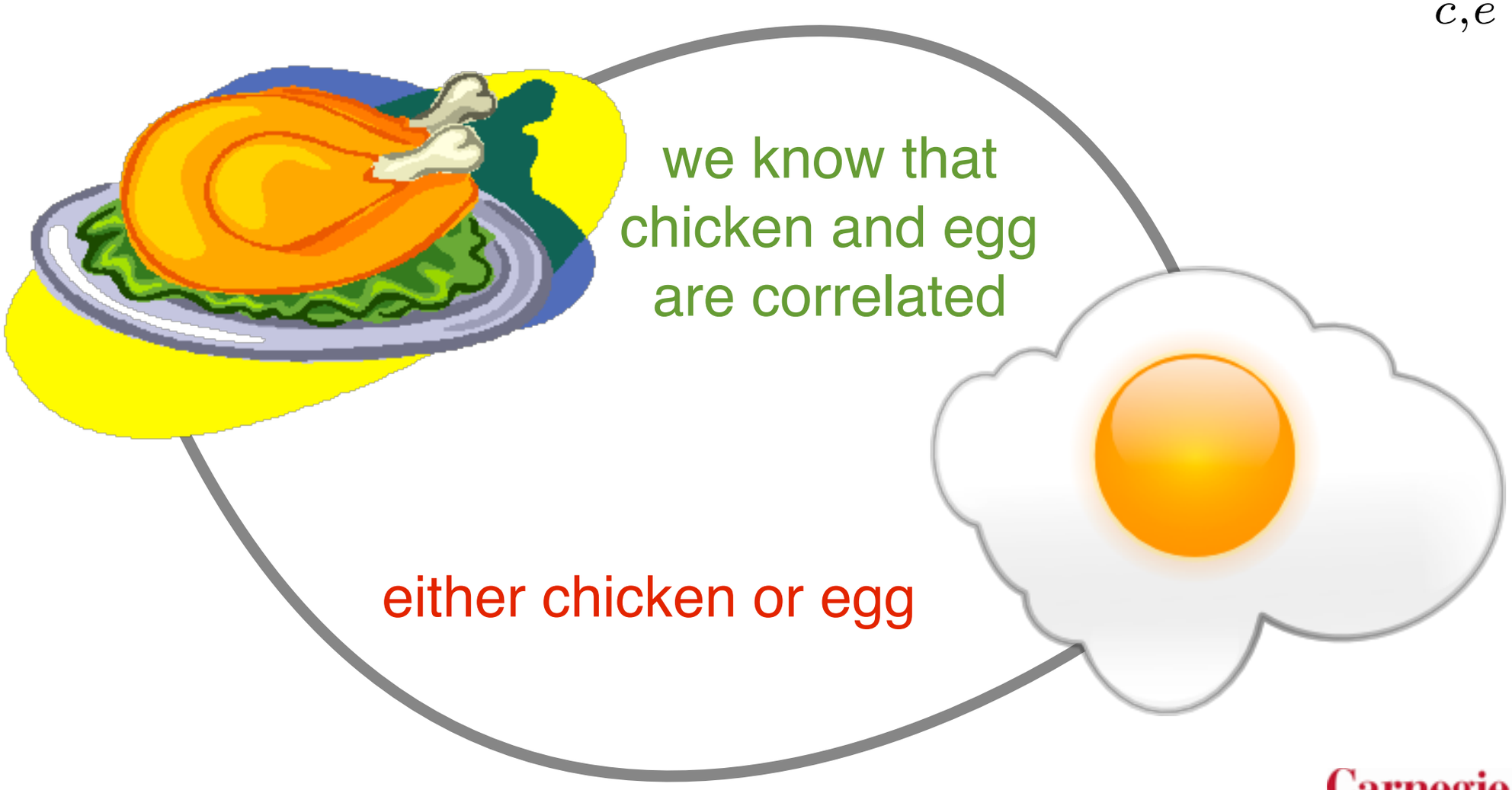
either chicken or egg



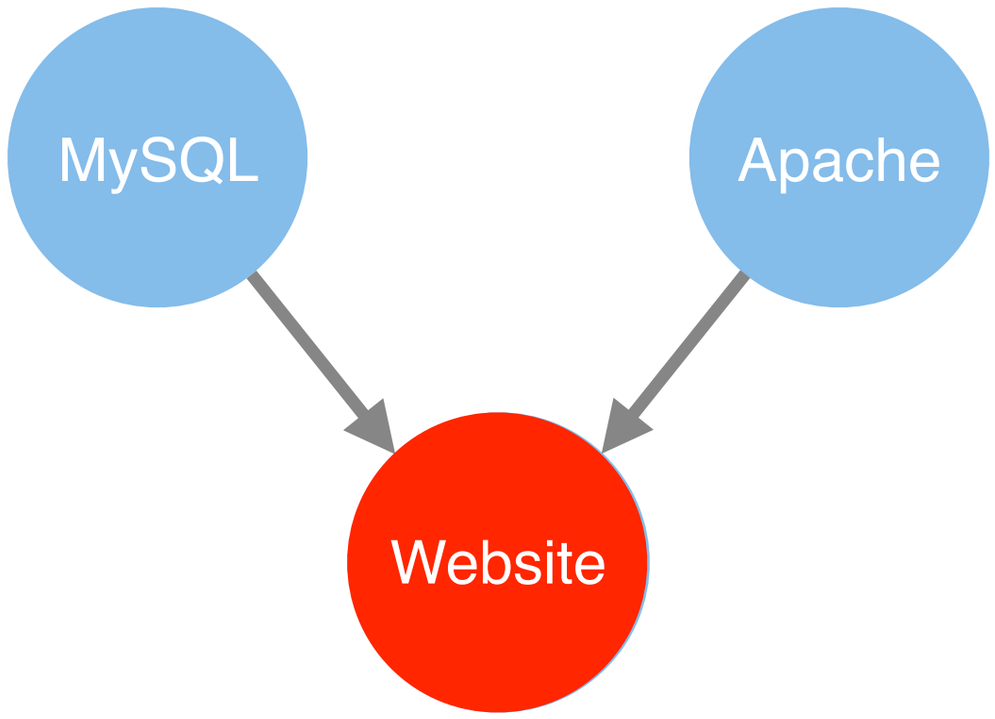
Chicken and Egg

$$p(c,e) = \frac{\exp \psi(c,e)}{\sum_{c',e'} \exp \psi(c',e')}$$

$$= \exp\left[\psi(c,e) - g(\psi)\right] \text{ where } g(\psi) = \log \sum_{c,e} \exp \psi(c,e)$$

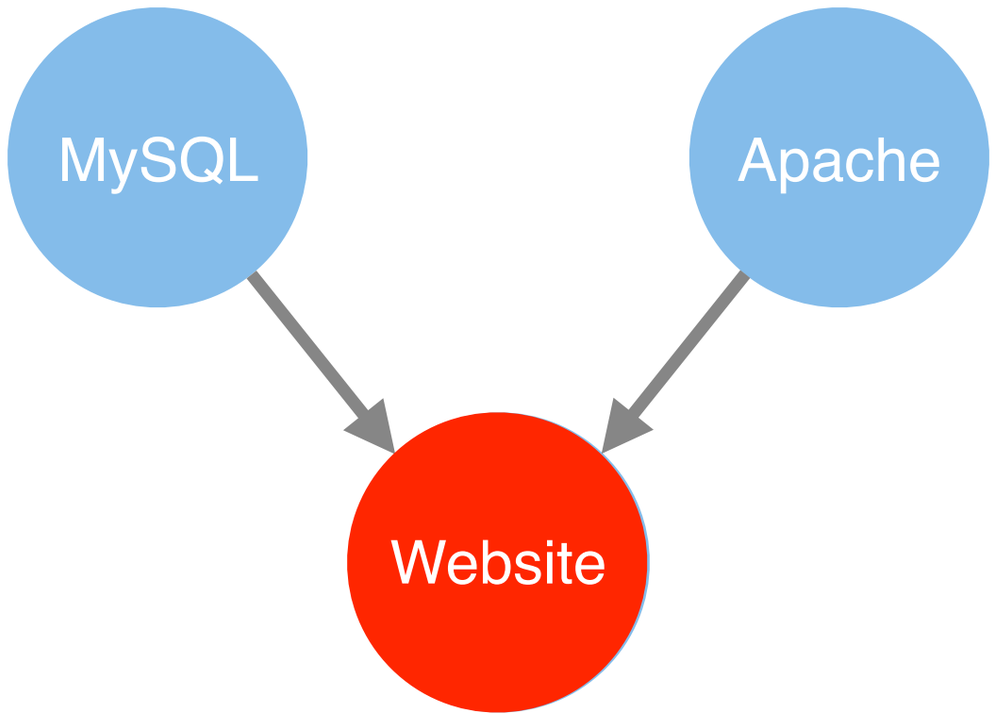


… some web service

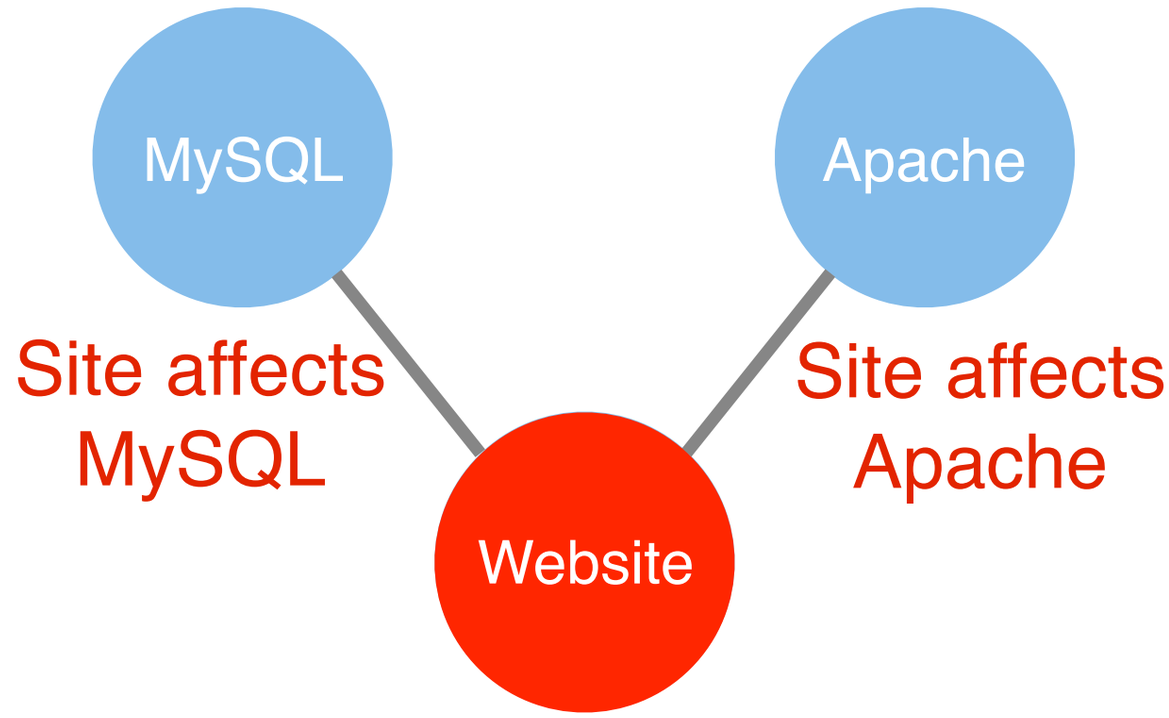


$$p(w|m,a)p(m)p(a)$$
$$m \not\perp\!\!\!\perp a|w$$

… some web service

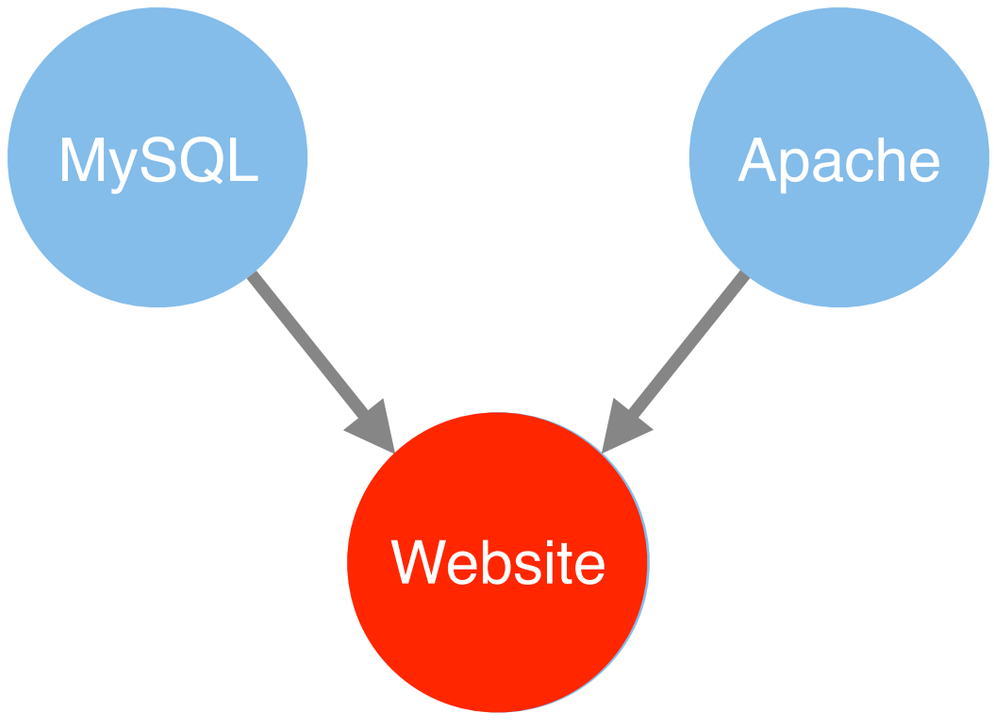


$$p(w|m,a)p(m)p(a)$$
$$m \not\perp\!\!\!\perp a|w$$



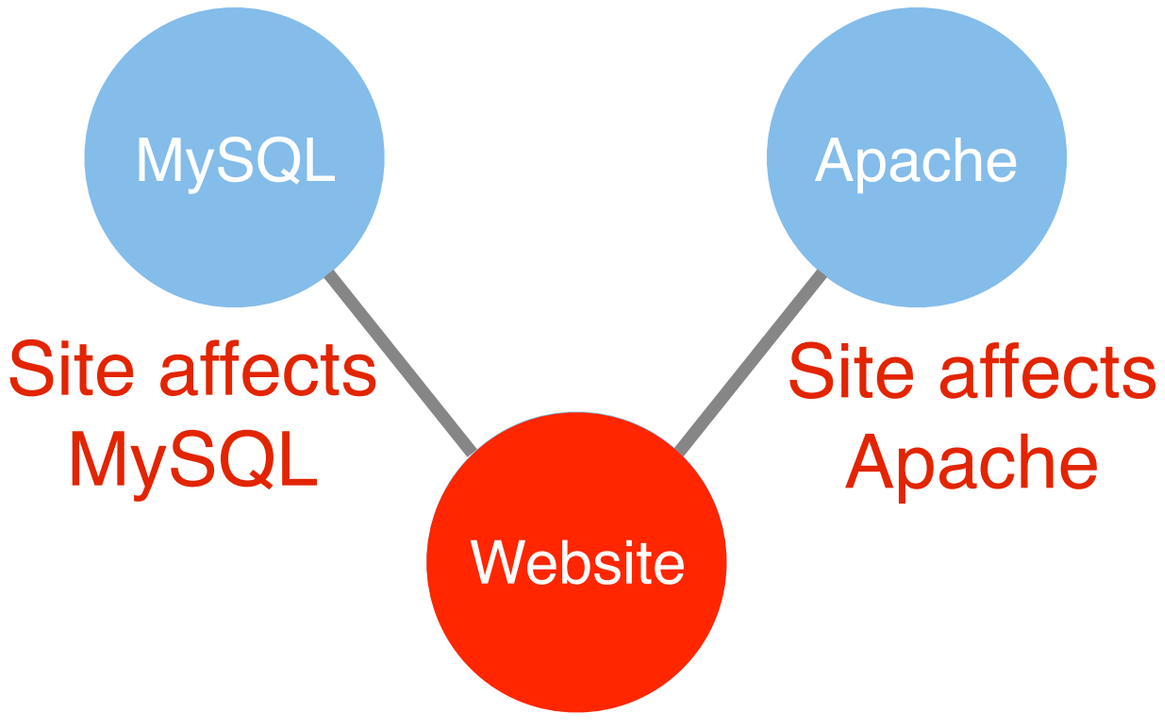
$$p(m,w,a) \propto \phi(m,w)\phi(w,a)$$
$$m \perp\!\!\!\perp a|w$$

… some web service



$$p(w|m,a)p(m)p(a)$$
$$m \not\perp\!\!\!\perp a|w$$

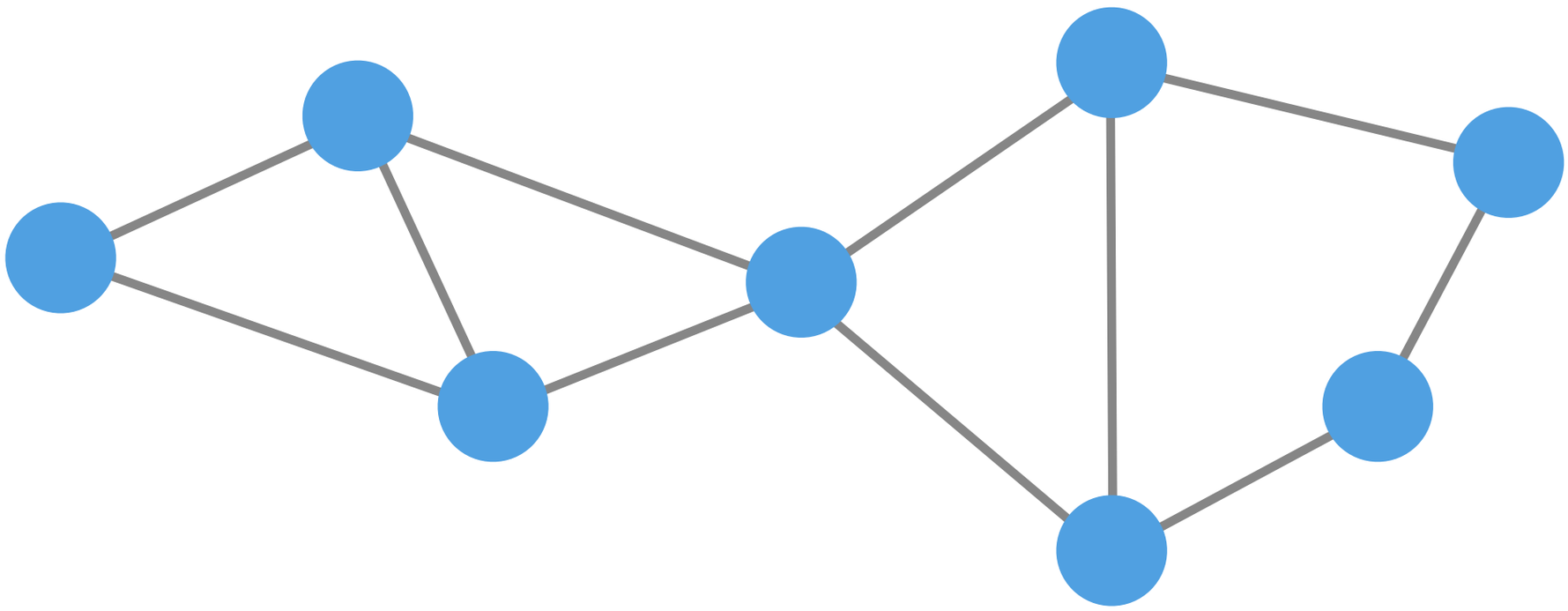
easier
“debugging”



$$p(m,w,a) \propto \phi(m,w)\phi(w,a)$$
$$m \perp\!\!\!\perp a|w$$

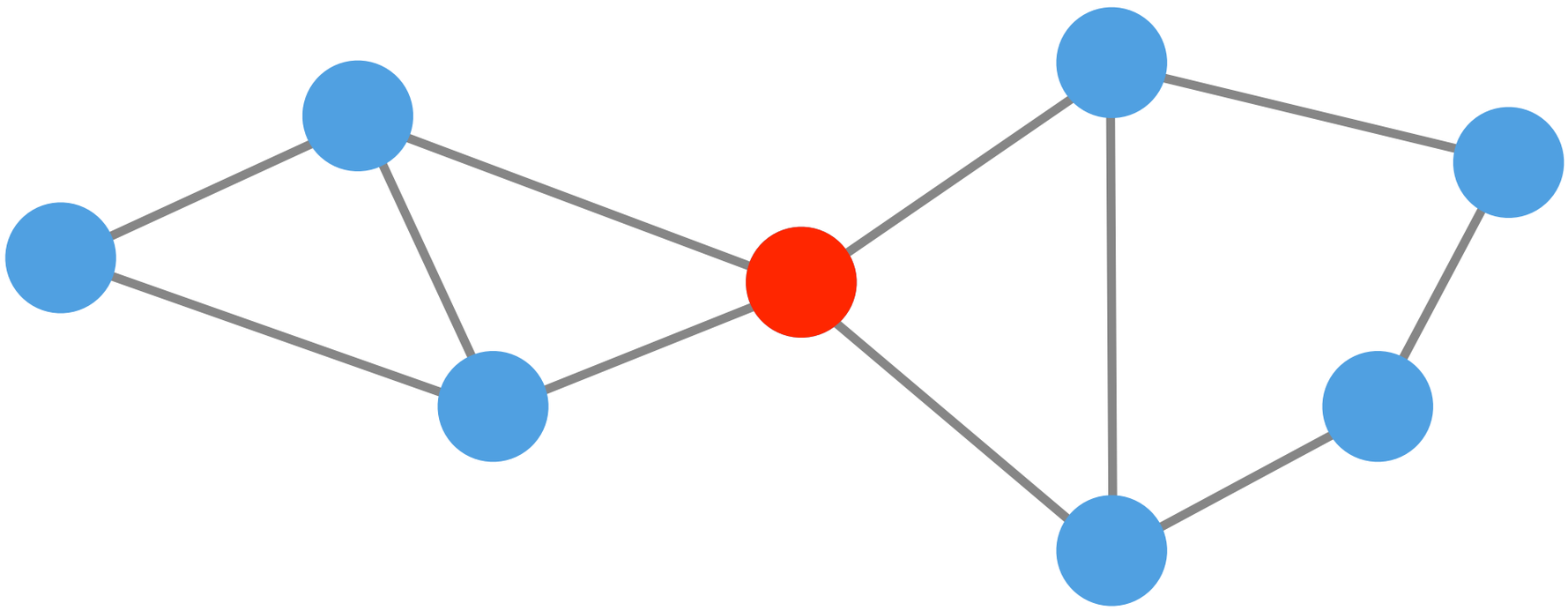
easier
“modeling”

Undirected Graphical Models



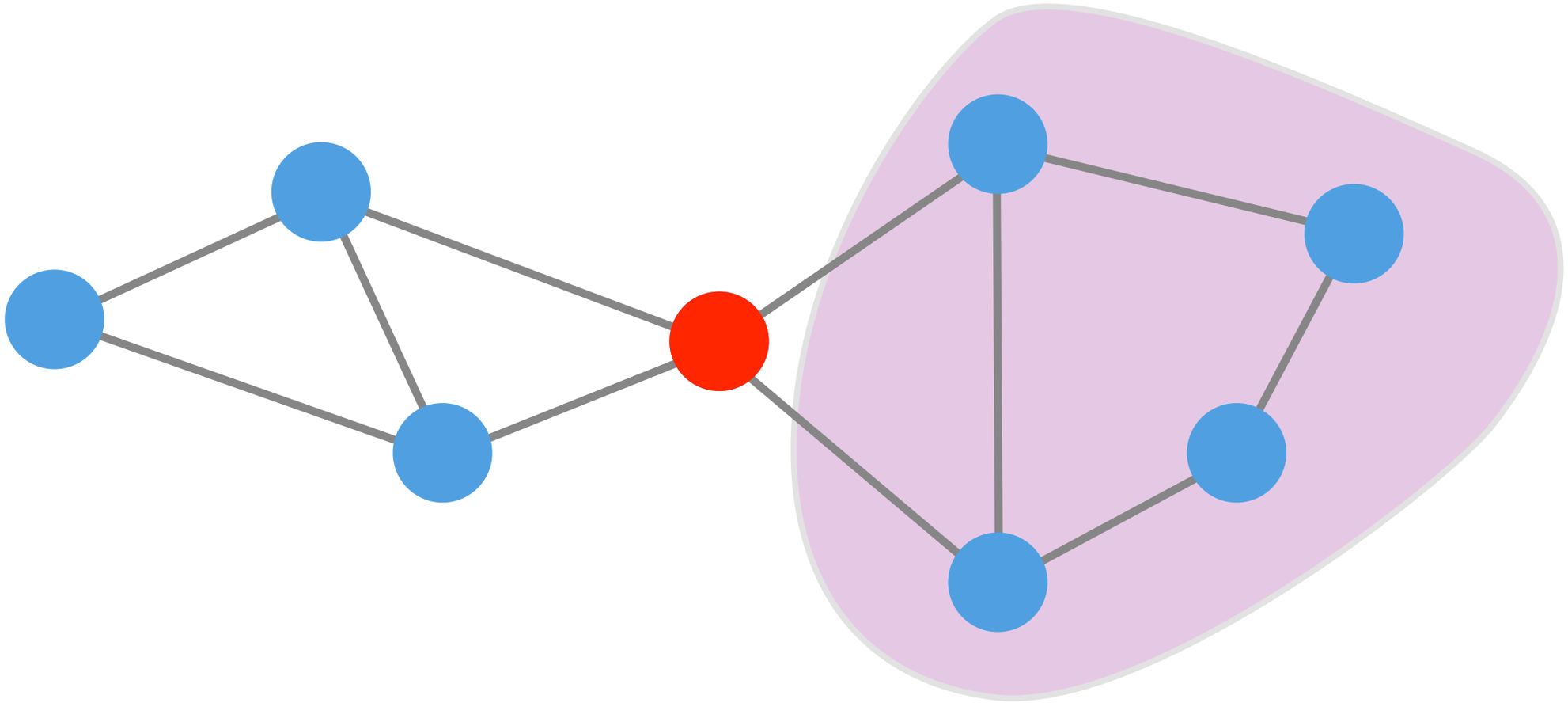
Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



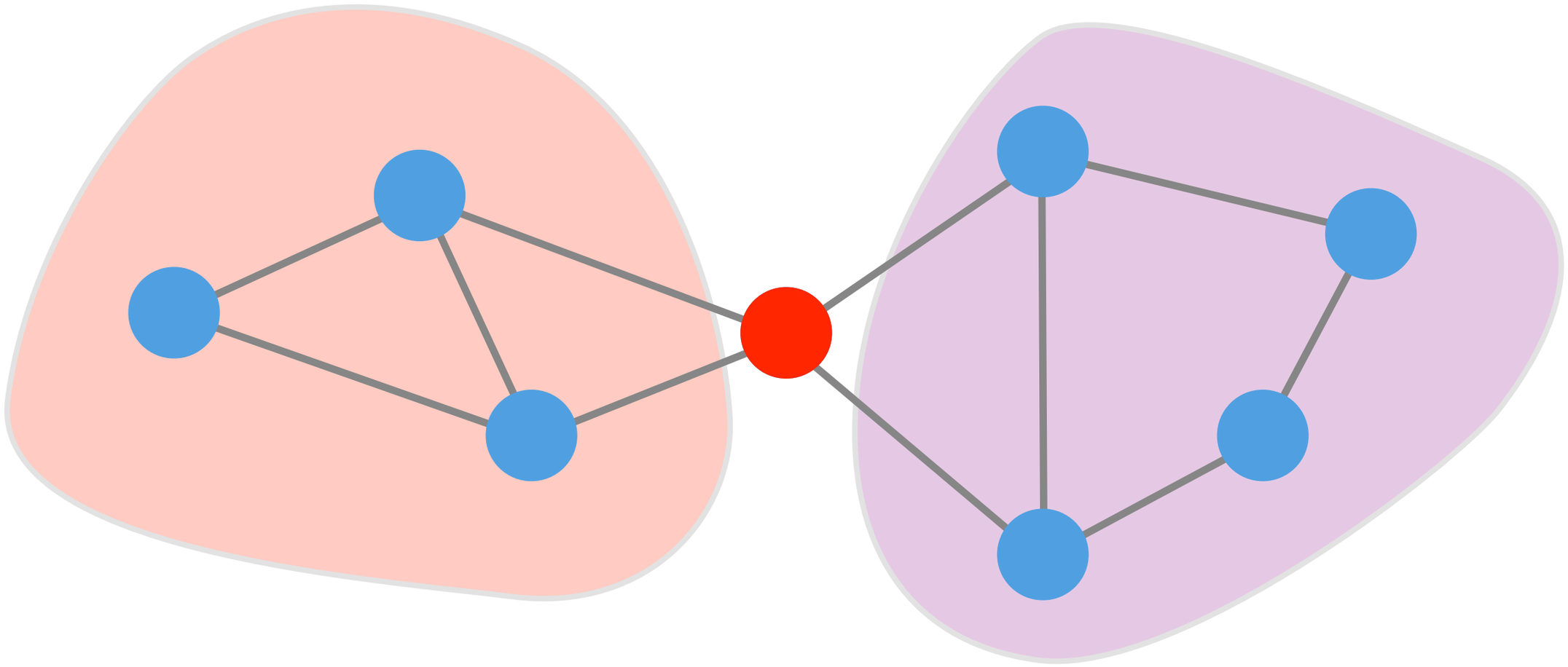
Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



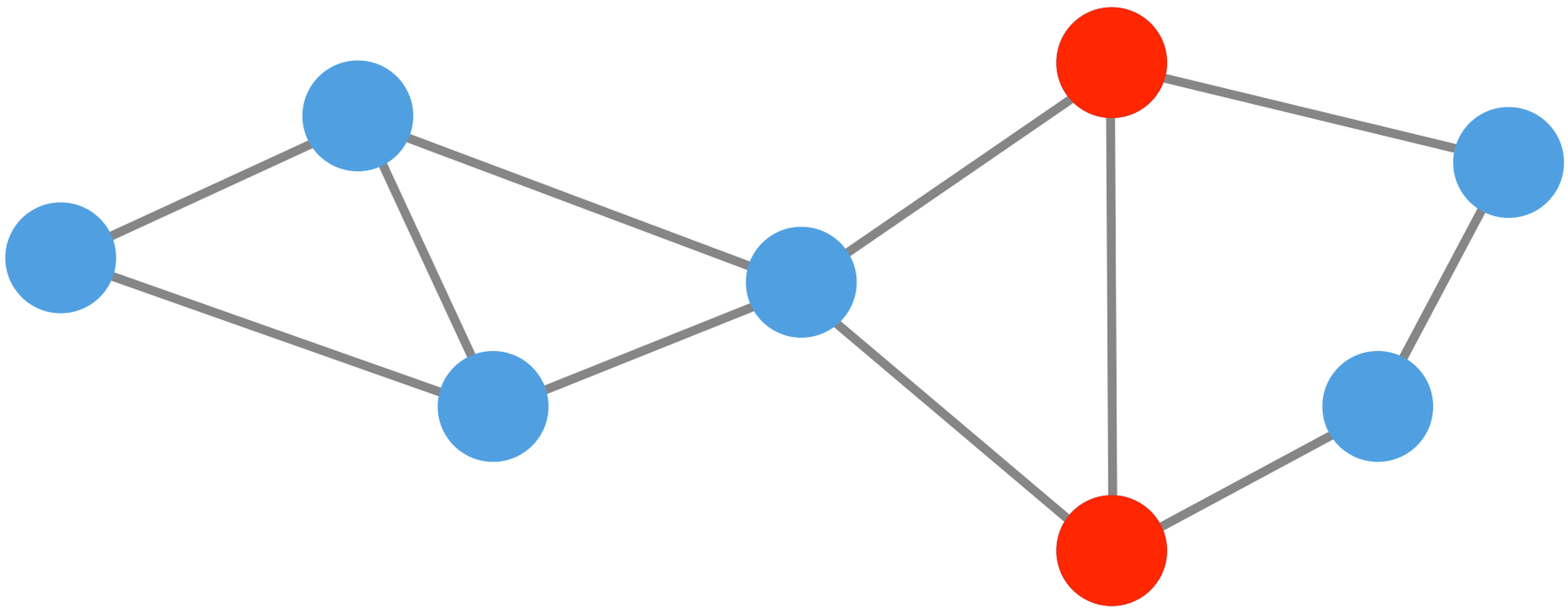
Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



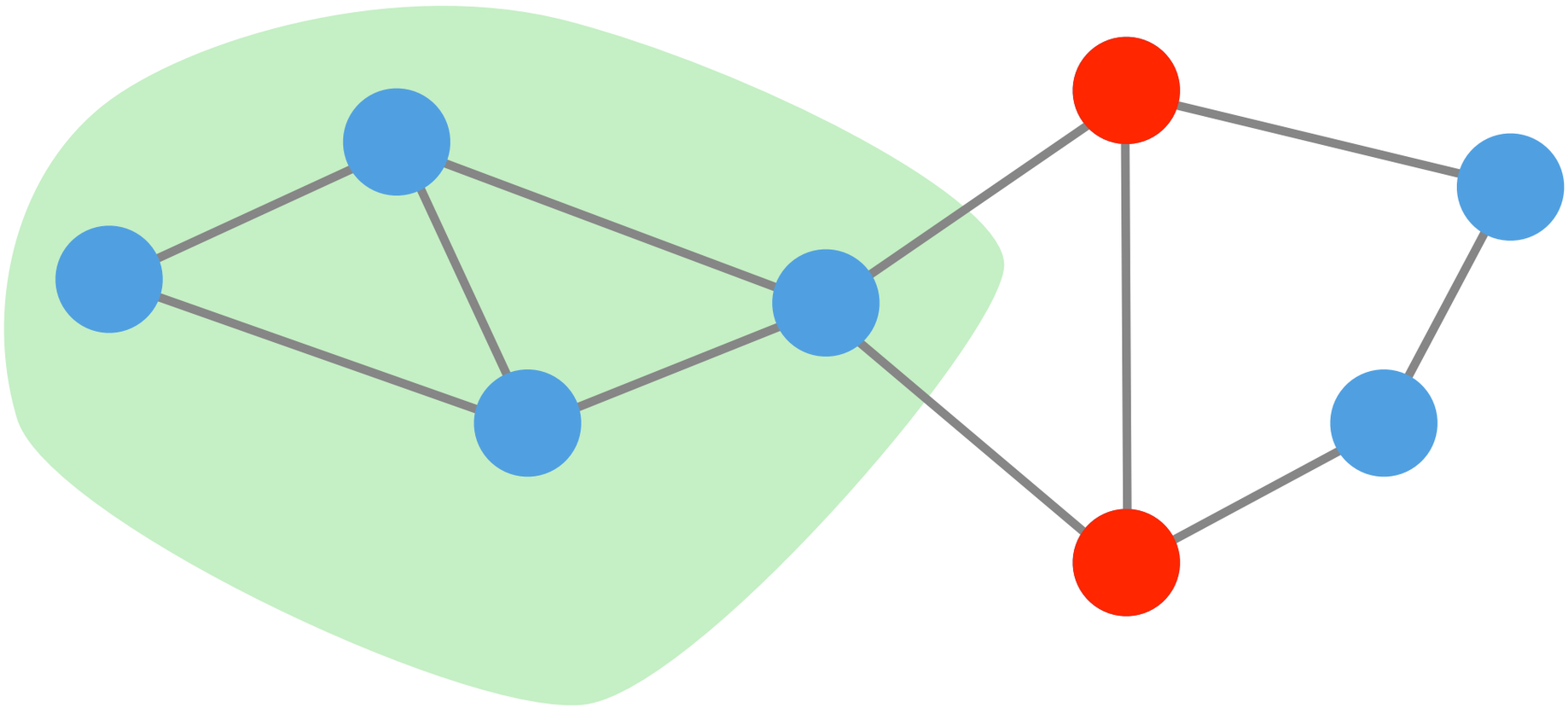
Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



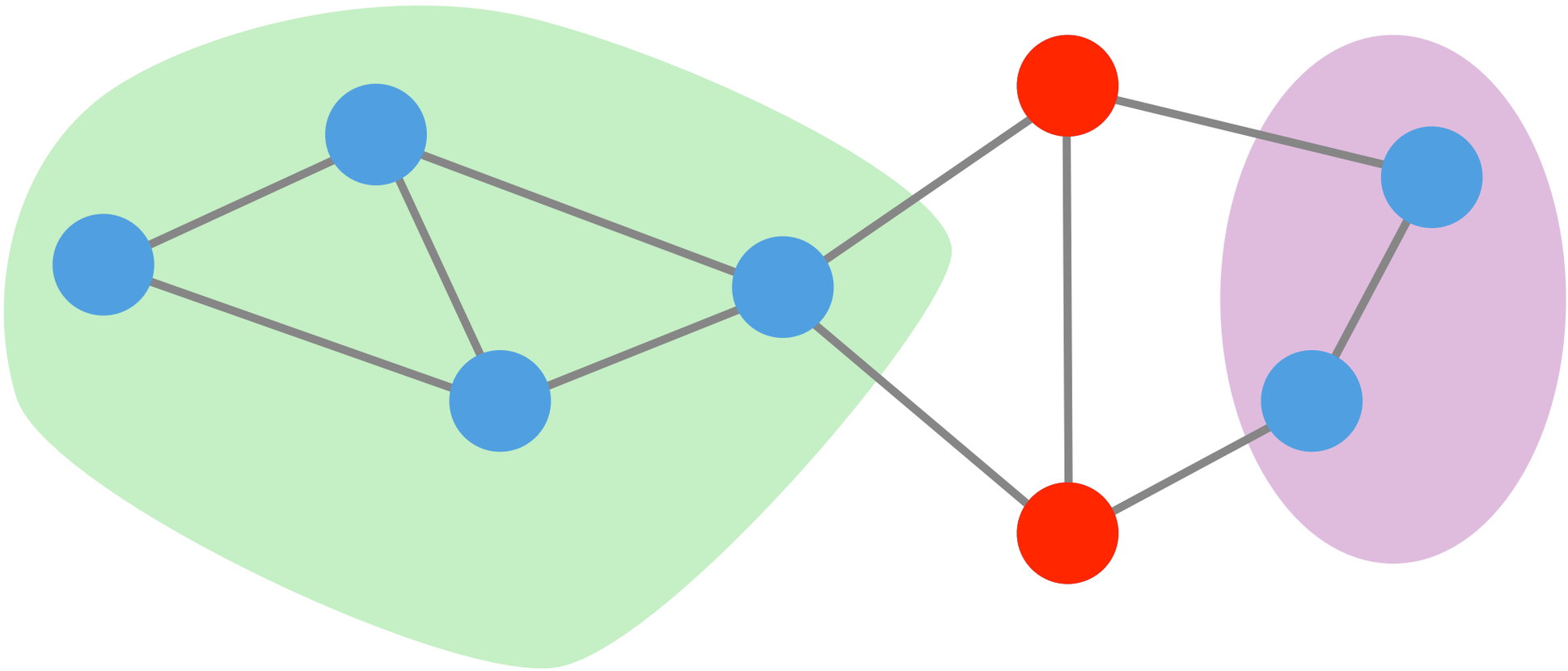
Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



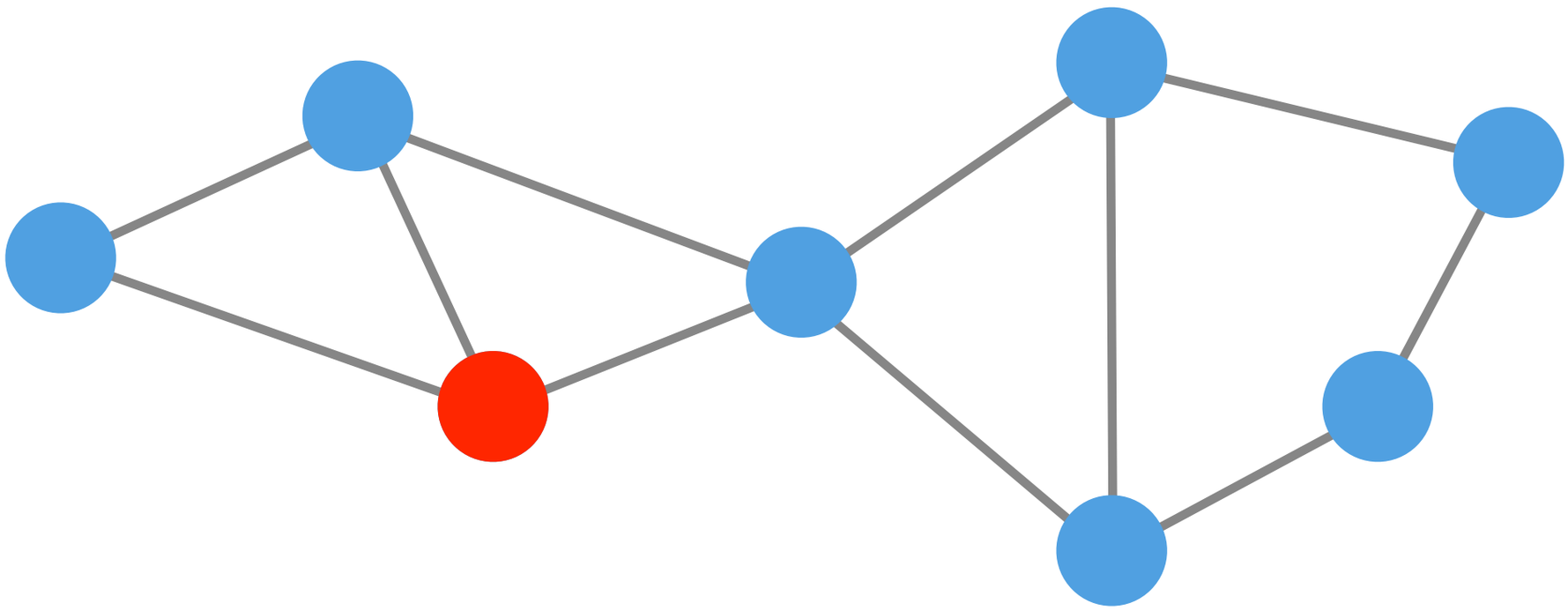
Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



Key Concept
Observing nodes makes remainder conditionally independent

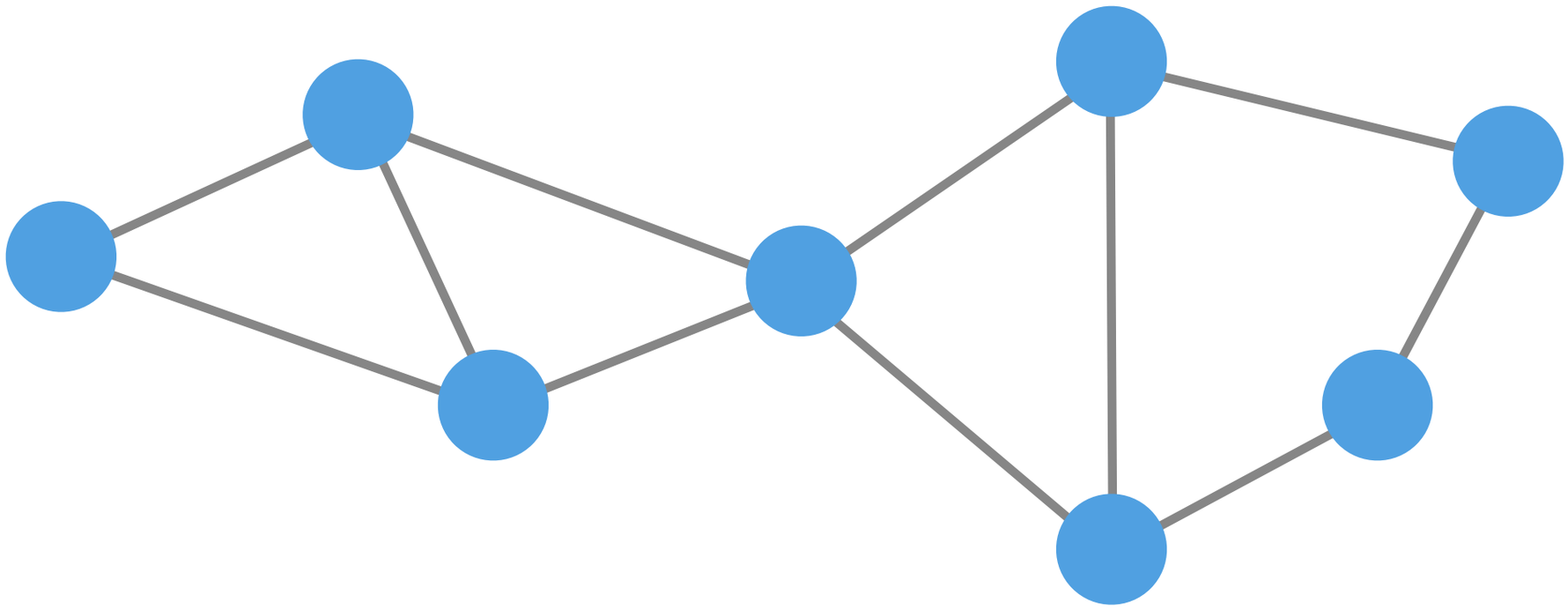
Undirected Graphical Models



Key Concept
Observing nodes makes remainder conditionally independent

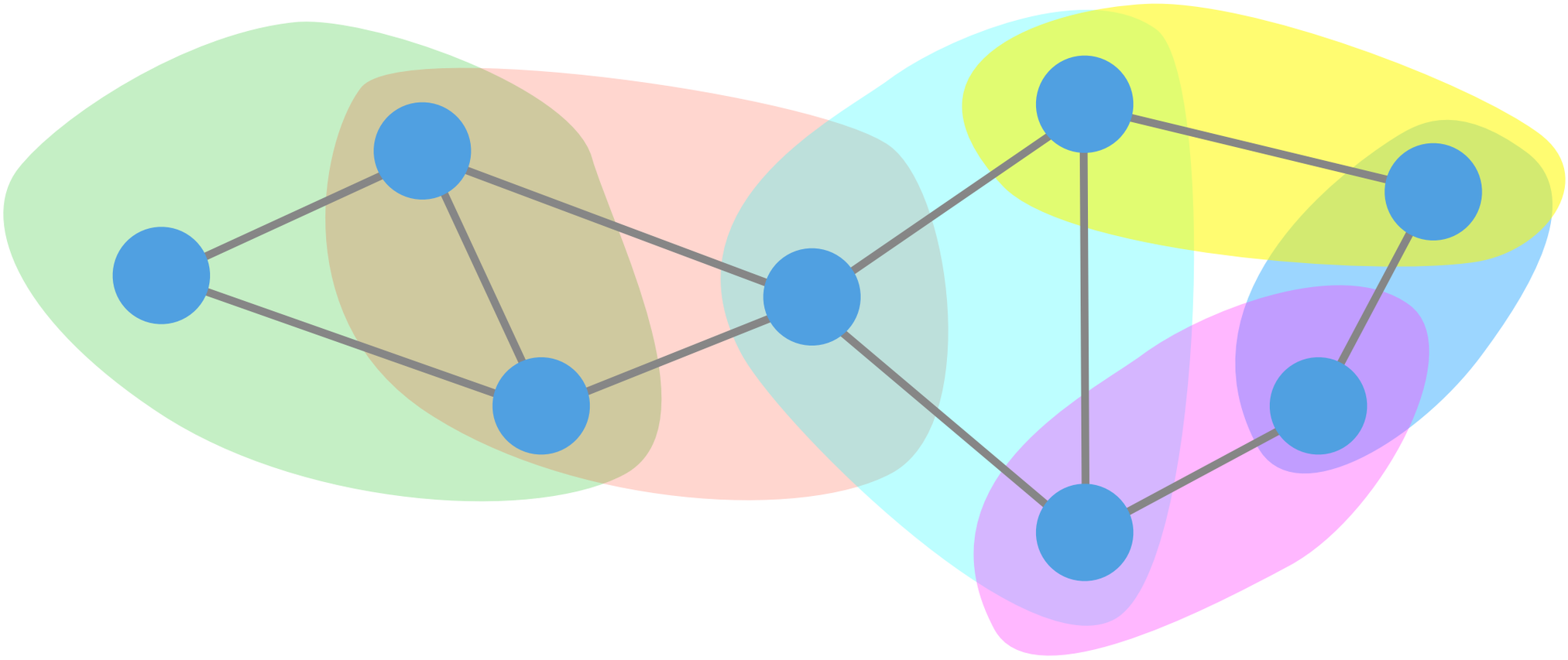
Cliques

Cliques



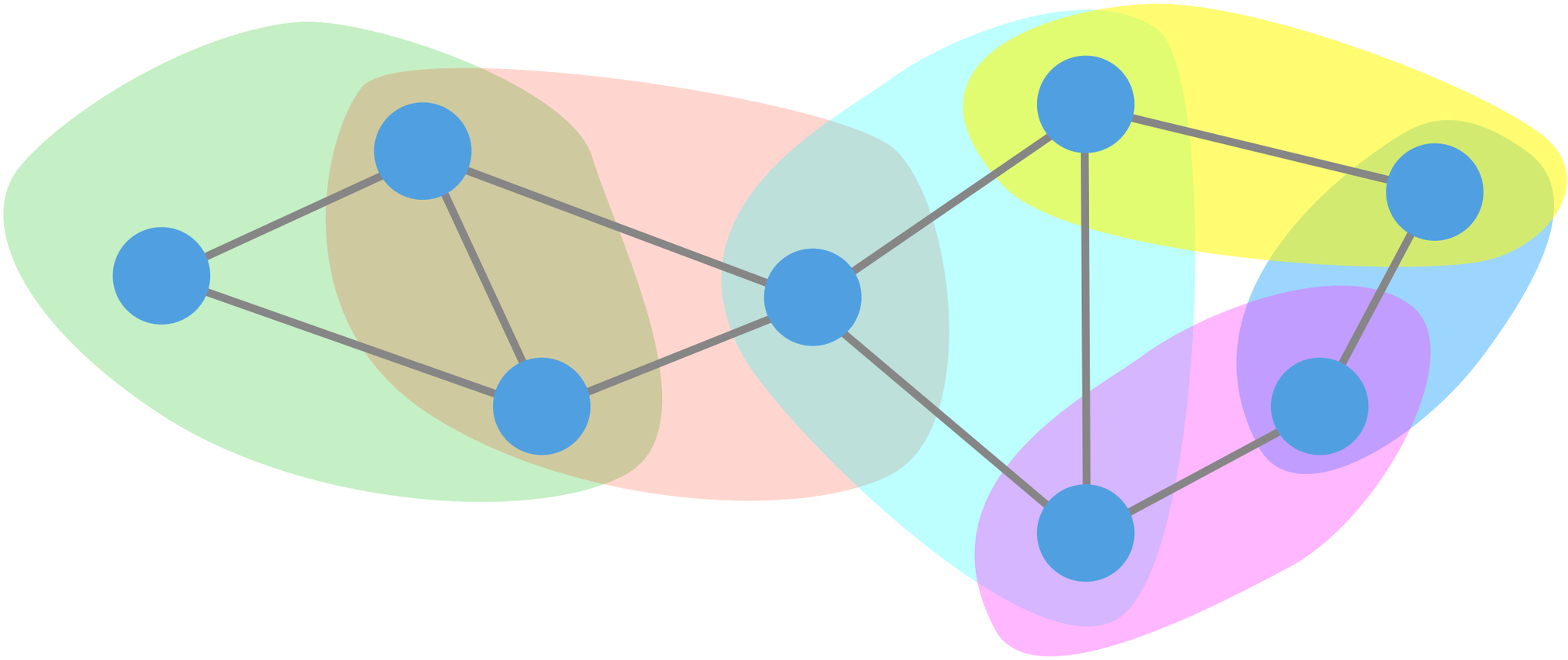
maximal fully connected subgraph

Cliques



maximal fully connected subgraph

Hammersley Clifford Theorem



If density has full support then it decomposes into products of clique potentials

$$p(x) = \prod_c \psi_c(x_c)$$

Directed vs. Undirected

- Causal description
- Normalization automatic
- Intuitive
- Requires knowledge of dependencies
- Conditional independence tricky (Bayes Ball algorithm)

- Noncausal description (correlation only)
- Intuitive
- Easy modeling
- Normalization difficult
- Conditional independence easy to read off (graph connectivity)

VS.

Exponential Families and Graphical Models

Exponential Family Recap

- Density function

$$p(x;\theta) = \exp\left(\langle \phi(x), \theta \rangle - g(\theta)\right)$$

$$\text{where } g(\theta) = \log \sum_{x'} \exp\left(\langle \phi(x'), \theta \rangle\right)$$

- Log partition function generates cumulants

$$\partial_\theta g(\theta) = \mathbf{E}\left[\phi(x)\right]$$

$$\partial_\theta^2 g(\theta) = \mathrm{Var}\left[\phi(x)\right]$$

- g is convex (second derivative is p.s.d.)

Log Partition Function

$$p(x|\theta) = e^{\langle \phi(x), \theta \rangle - g(\theta)}$$

$$g(\theta) = \log \sum_x e^{\langle \phi(x), \theta \rangle}$$

$$\partial_\theta g(\theta) = \frac{\sum_x \phi(x) e^{\langle \phi(x), \theta \rangle}}{\sum_x e^{\langle \phi(x), \theta \rangle}} = \sum_x \phi(x) e^{\langle \phi(x), \theta \rangle - g(\theta)}$$

Unconditional model

$$p(y|\theta, x) = e^{\langle \phi(x,y), \theta \rangle - g(\theta|x)}$$

$$g(\theta|x) = \log \sum_y e^{\langle \phi(x,y), \theta \rangle}$$

$$\partial_\theta g(\theta|x) = \frac{\sum_y \phi(x,y) e^{\langle \phi(x,y), \theta \rangle}}{\sum_y e^{\langle \phi(x,y), \theta \rangle}} = \sum_y \phi(x,y) e^{\langle \phi(x,y), \theta \rangle - g(\theta|x)}$$

Conditional model

Estimation

- Conditional log-likelihood

$$\log p(y|x;\theta) = \langle \phi(x,y), \theta \rangle - g(\theta|x)$$

- Log-posterior (Gaussian Prior)

$$\begin{aligned}\log p(\theta|X,Y) &= \sum_i \log(y_i|x_i;\theta) + \log p(\theta) + \text{const.} \\ &= \left\langle \sum_i \phi(x_i,y_i), \theta \right\rangle - \sum_i g(\theta|x_i) - \frac{1}{2\sigma^2}\|\theta\|^2 + \text{const.}\end{aligned}$$

- First order optimality conditions

maxent model

expensive

prior

$$\sum_i \phi(x_i,y_i) = \sum_i \mathbf{E}_{y|x_i}\left[\phi(x_i,y)\right] + \frac{1}{\sigma^2}\theta$$

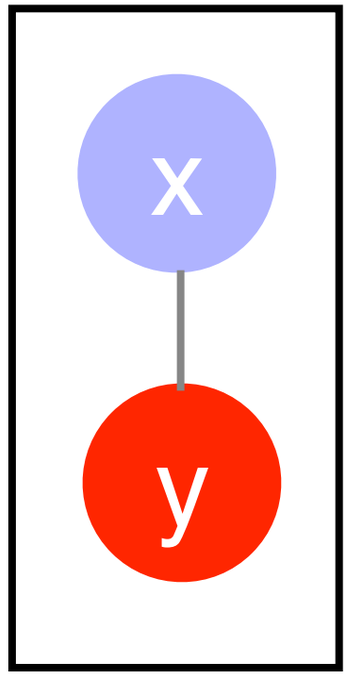
Logistic Regression

- Label space

  $\phi(x, y) = y\phi(x)$ where $y \in \{\pm 1\}$

- Log-partition function

  $$g(\theta|x) = \log\left[e^{1\cdot\langle\phi(x),\theta\rangle} + e^{-1\cdot\langle\phi(x),\theta\rangle}\right] = \log 2\cosh\langle\phi(x),\theta\rangle$$



- Convex minimization problem

  $$\underset{\theta}{\text{minimize}}\ \frac{1}{2\sigma^2}\|\theta\|^2 + \sum_i \log 2\cosh\langle\phi(x_i),\theta\rangle - y_i\langle\phi(x_i,\theta\rangle$$

- Prediction

  $$p(y|x,\theta) = \frac{e^{y\langle\phi(x),\theta\rangle}}{e^{\langle\phi(x),\theta\rangle} + e^{-\langle\phi(x),\theta\rangle}} = \frac{1}{1 + e^{-2y\langle\phi(x),\theta\rangle}}$$

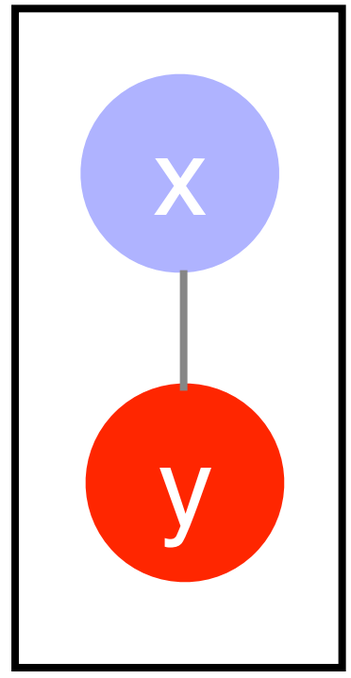
Logistic Regression

- Label space

  $\phi(x, y) = y\phi(x)$ where $y \in \{\pm 1\}$

- Log-partition function

  $$g(\theta|x) = \log \left[e^{1 \cdot \langle \phi(x), \theta \rangle} + e^{-1 \cdot \langle \phi(x), \theta \rangle} \right] = \log 2 \cosh \langle \phi(x), \theta \rangle$$



- Convex minimization problem

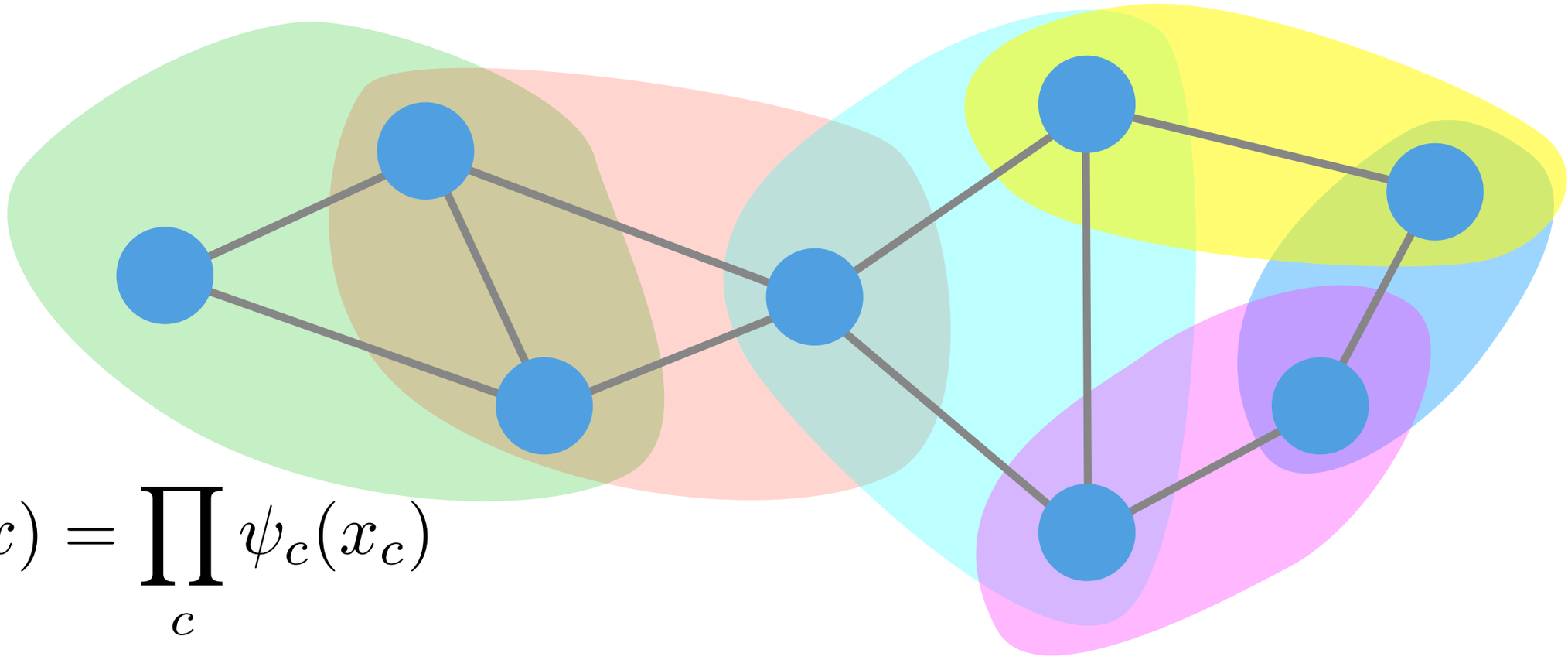
  $$\underset{\theta}{\text{minimize}} \frac{1}{2\sigma^2} \|\theta\|^2 + \sum_i \log 2 \cosh \langle \phi(x_i), \theta \rangle - y_i \langle \phi(x_i, \theta \rangle$$

- Prediction

GP Classification

$$p(y|x, \theta) = \frac{e^{y\langle \phi(x), \theta \rangle}}{e^{\langle \phi(x), \theta \rangle} + e^{-\langle \phi(x), \theta \rangle}} = \frac{1}{1 + e^{-2y\langle \phi(x), \theta \rangle}}$$

Exponential Clique Decomposition



$$p(x) = \prod_c \psi_c(x_c)$$

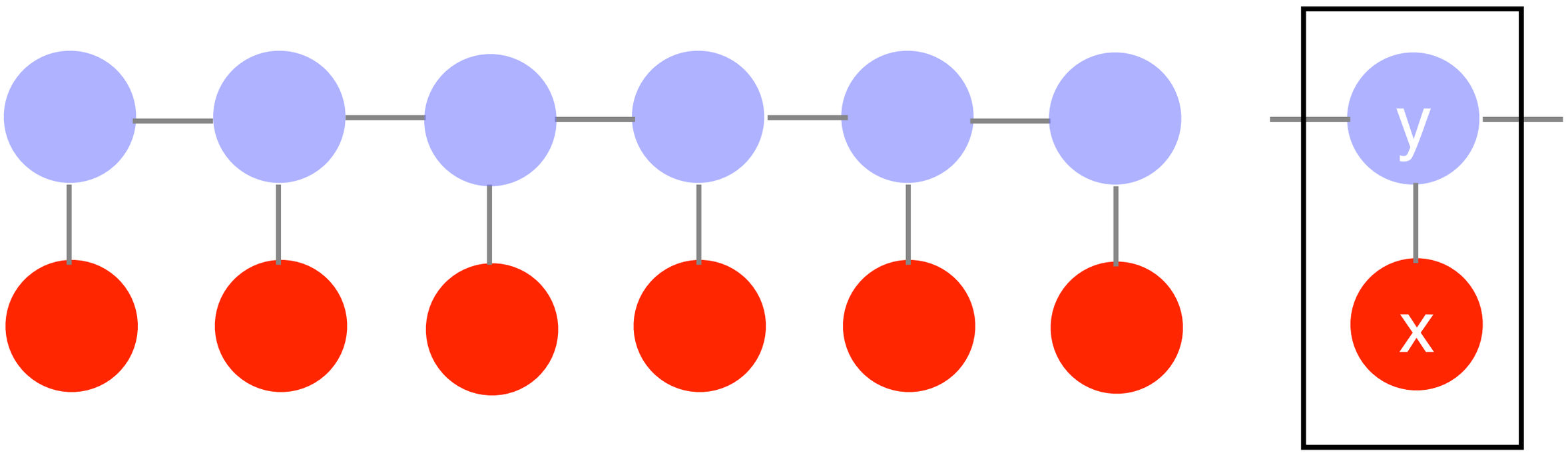
Theorem: Clique decomposition holds in sufficient statistics

$$\phi(x) = (\ldots, \phi_c(x_c), \ldots) \text{ and } \langle \phi(x), \theta \rangle = \sum_c \langle \phi_c(x_c), \theta_c \rangle$$

Corollary: we only need expectations on cliques

$$\mathbf{E}_x[\phi(x)] = (\ldots, \mathbf{E}_{x_c}[\phi_c(x_c)], \ldots)$$

Conditional Random Fields



$$\phi(x) = (y_1\phi_x(x_1), \ldots, y_n\phi_x(x_n), \phi_y(y_1, y_2), \ldots, \phi_y(y_{n-1}, y_n))$$

$$\langle \phi(x), \theta \rangle = \sum_i \langle \phi_x(x_i, y_i), \theta_x \rangle + \sum_i \langle \phi_y(y_i, y_{i+1}), \theta_y \rangle$$

$$g(\theta|x) = \sum_y \prod_i f_i(y_i, y_{i+1}) \text{ where}$$

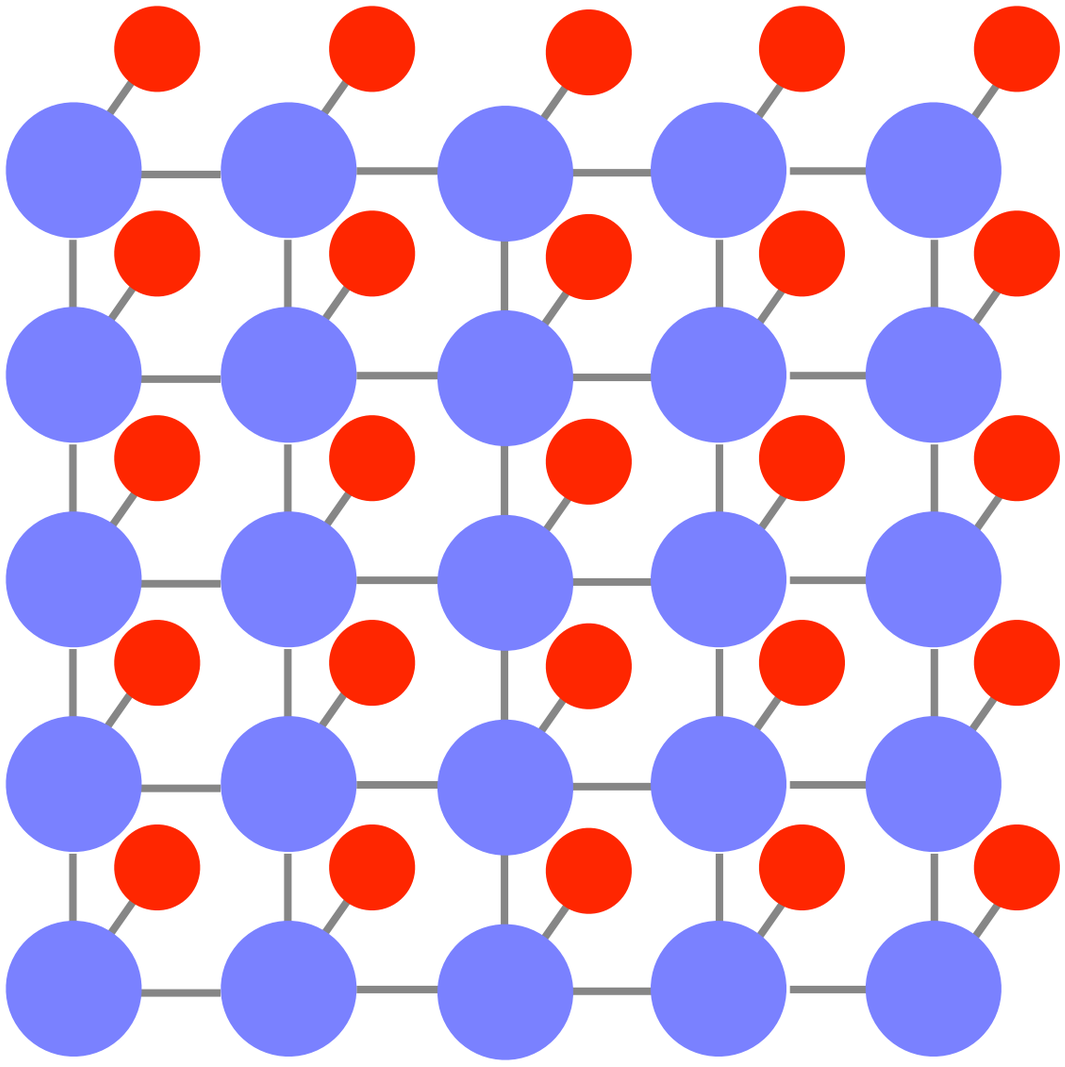
$$f_i(y_i, y_{i+1}) = e^{\langle \phi_x(x_i, y_i), \theta_x \rangle + \langle \phi_y(y_i, y_{i+1}), \theta_y \rangle}$$

dynamic programming

Conditional Random Fields

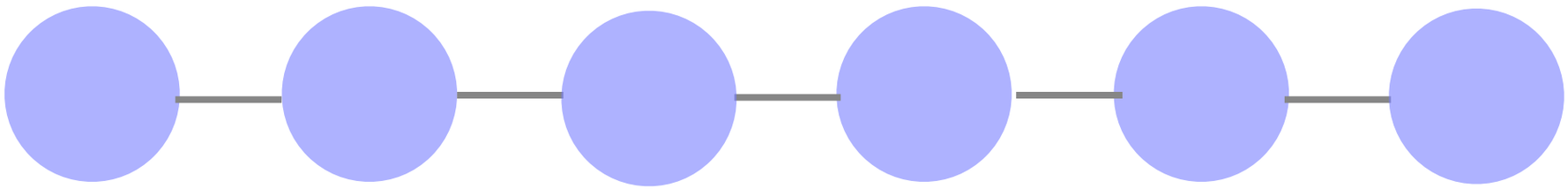
- Compute distribution over marginal and adjacent labels
- Take conditional expectations
- Take update step (batch or online)

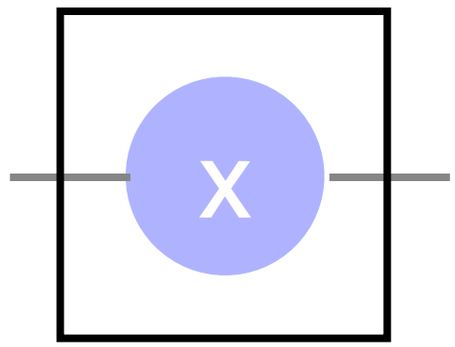
- More general techniques for computing normalization via message passing ...



Examples

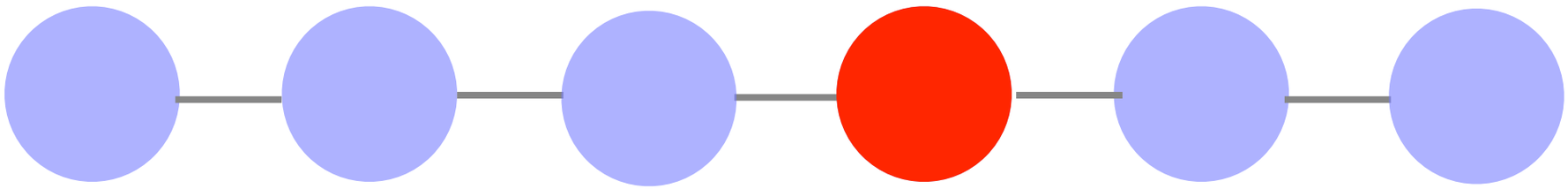
Chains

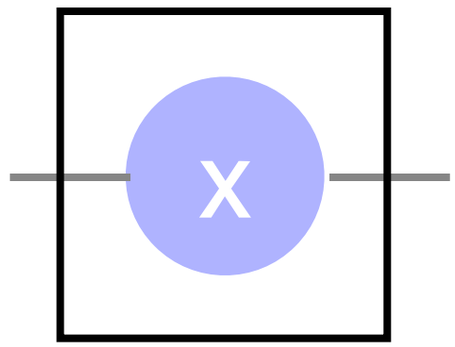




$$p(x) = \prod_i \psi_i(x_i, x_{i+1})$$

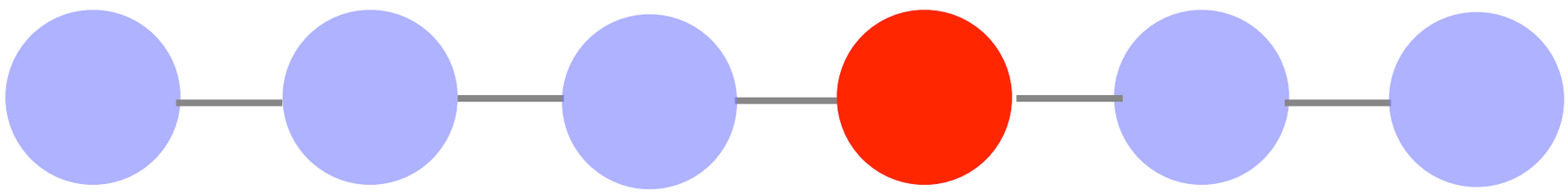
Chains

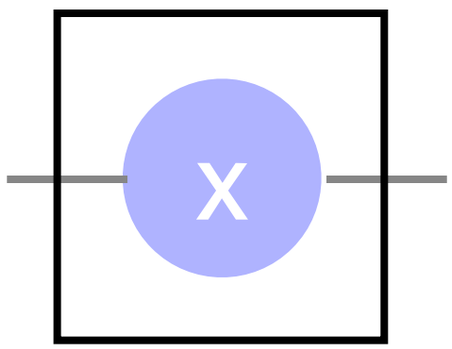




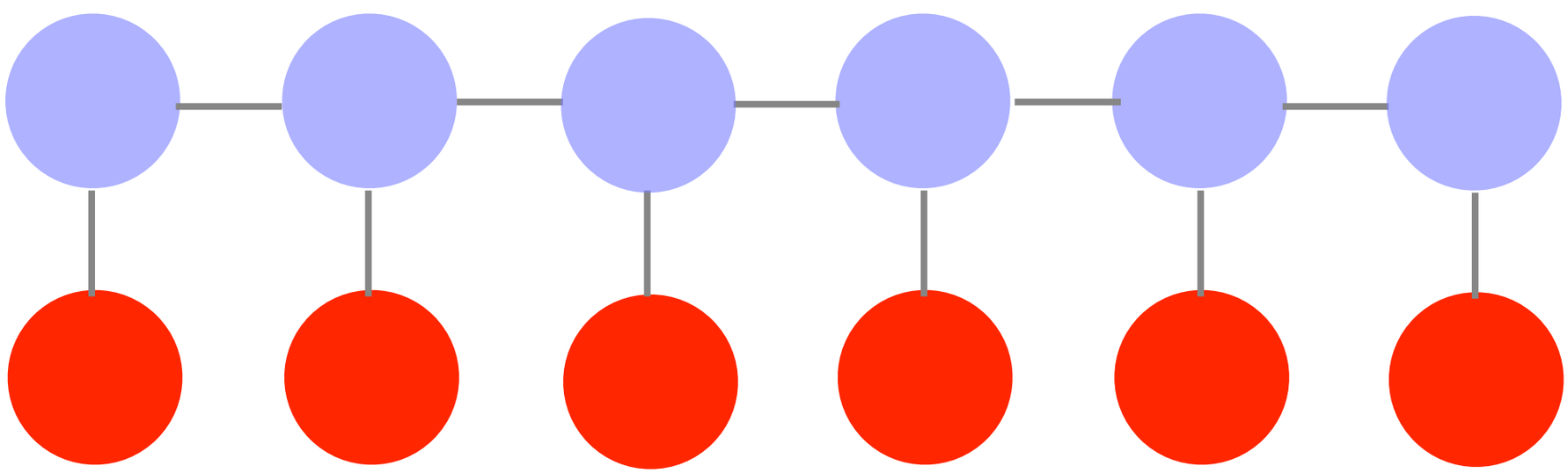
$$p(x) = \prod_i \psi_i(x_i, x_{i+1})$$

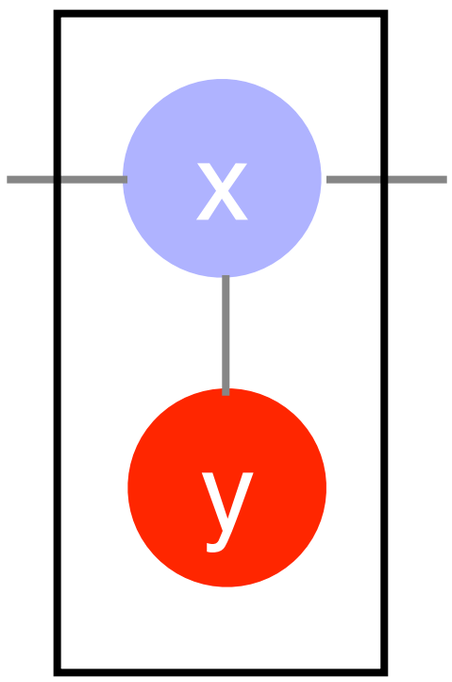
Chains





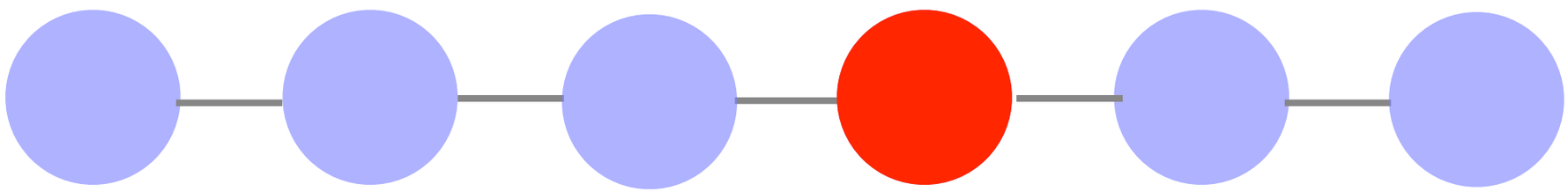
$$p(x) = \prod_i \psi_i(x_i, x_{i+1})$$

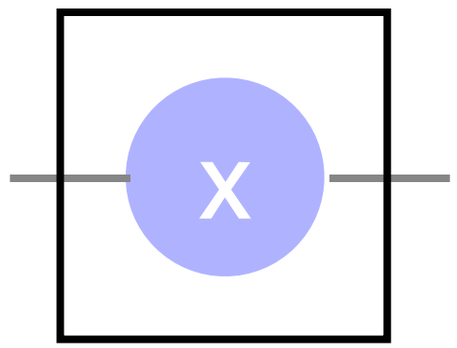




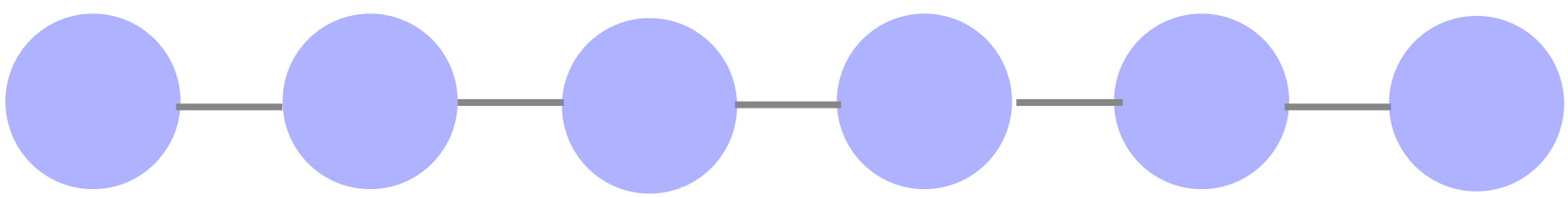
$$p(x, y) = \prod_i \psi_i^x(x_i, x_{i+1})\psi_i^{xy}(x_i, y_i)$$

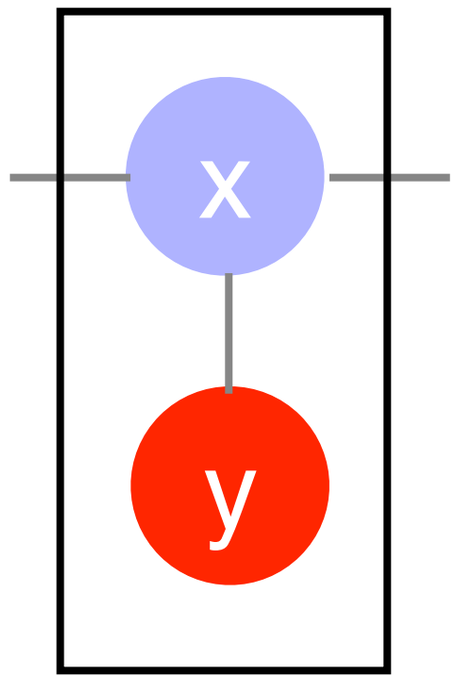
Chains





$$p(x) = \prod_i \psi_i(x_i, x_{i+1})$$

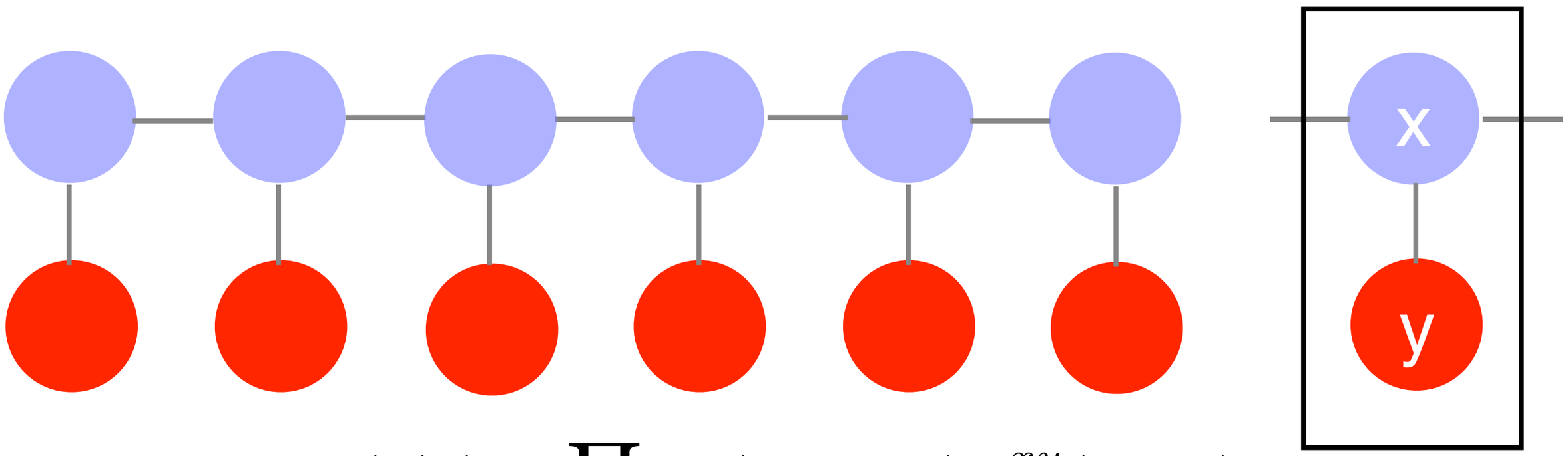




$$p(x|y) \propto \prod_i \underbrace{\psi_i^x(x_i, x_{i+1})\psi_i^{xy}(x_i, y_i)}_{=: f_i(x_i, x_{i+1})}$$

$$p(x, y) = \prod_i \psi_i^x(x_i, x_{i+1})\psi_i^{xy}(x_i, y_i)$$

Chains



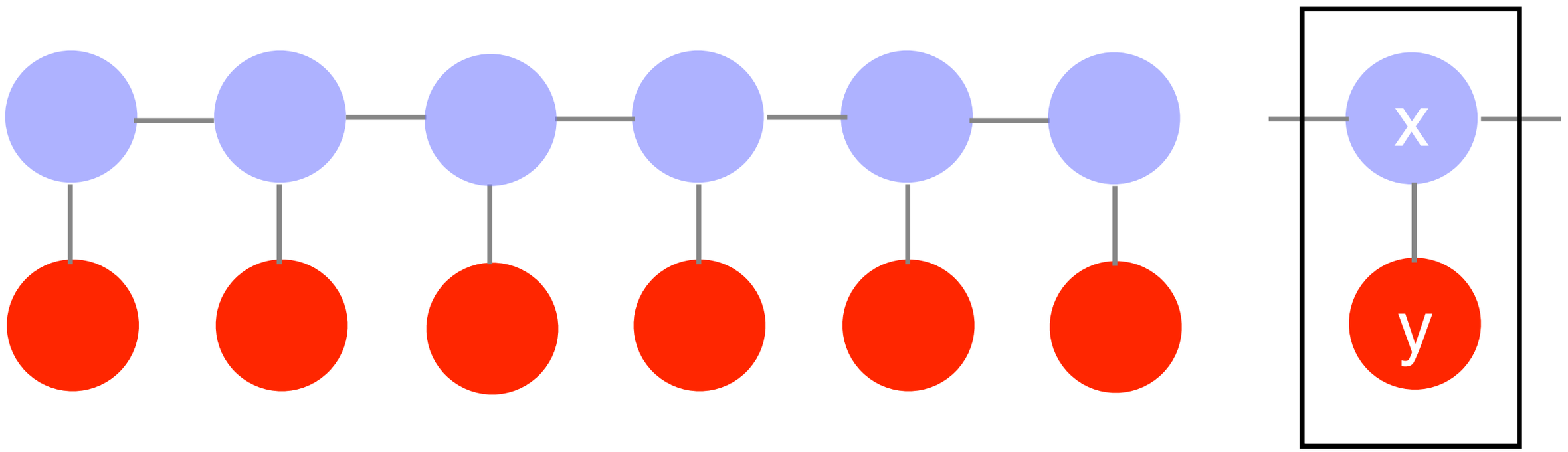
$$p(x|y) \propto \prod_i \underbrace{\psi_i^x(x_i, x_{i+1})\psi_i^{xy}(x_i, y_i)}_{=: f_i(x_i, x_{i+1})}$$

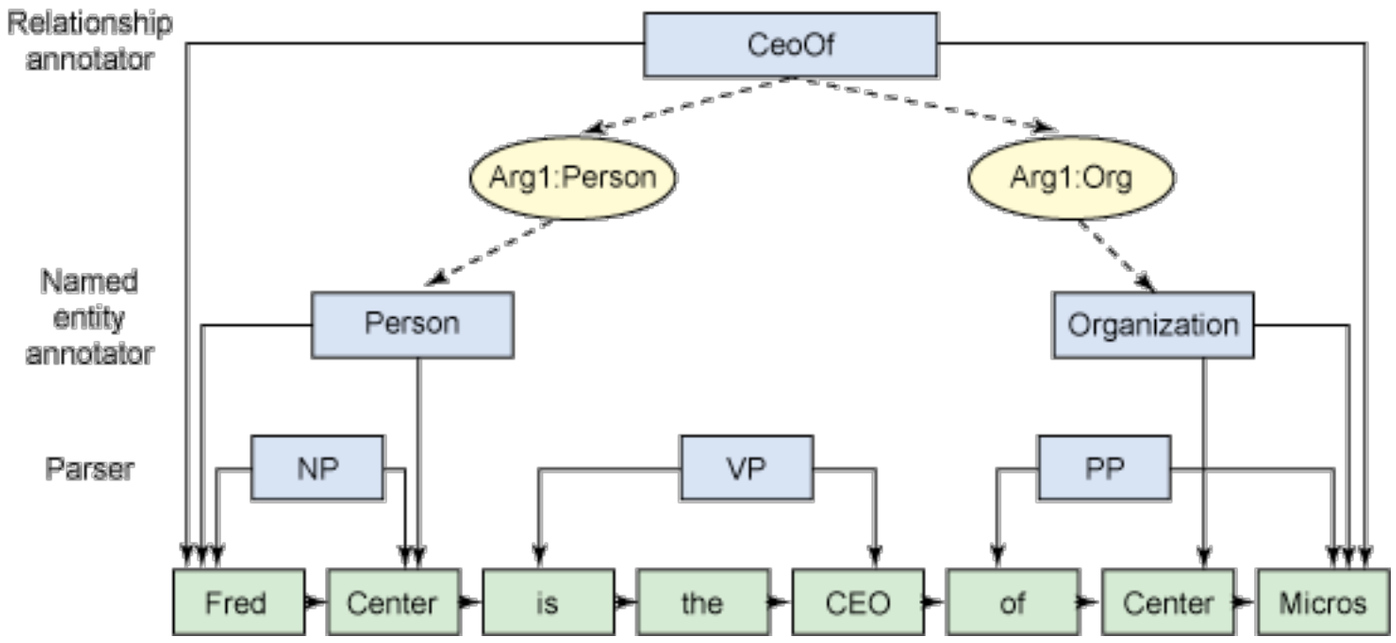
Dynamic Programming

$$l_1(x_1) = 1 \text{ and } l_{i+1}(x_{i+1}) = \sum_{x_i} l_i(x_i) f_i(x_i, x_{i+1})$$

$$r_n(x_n) = 1 \text{ and } r_i(x_i) = \sum_{x_{i+1}} r_{i+1}(x_{i+1}) f_i(x_i, x_{i+1})$$

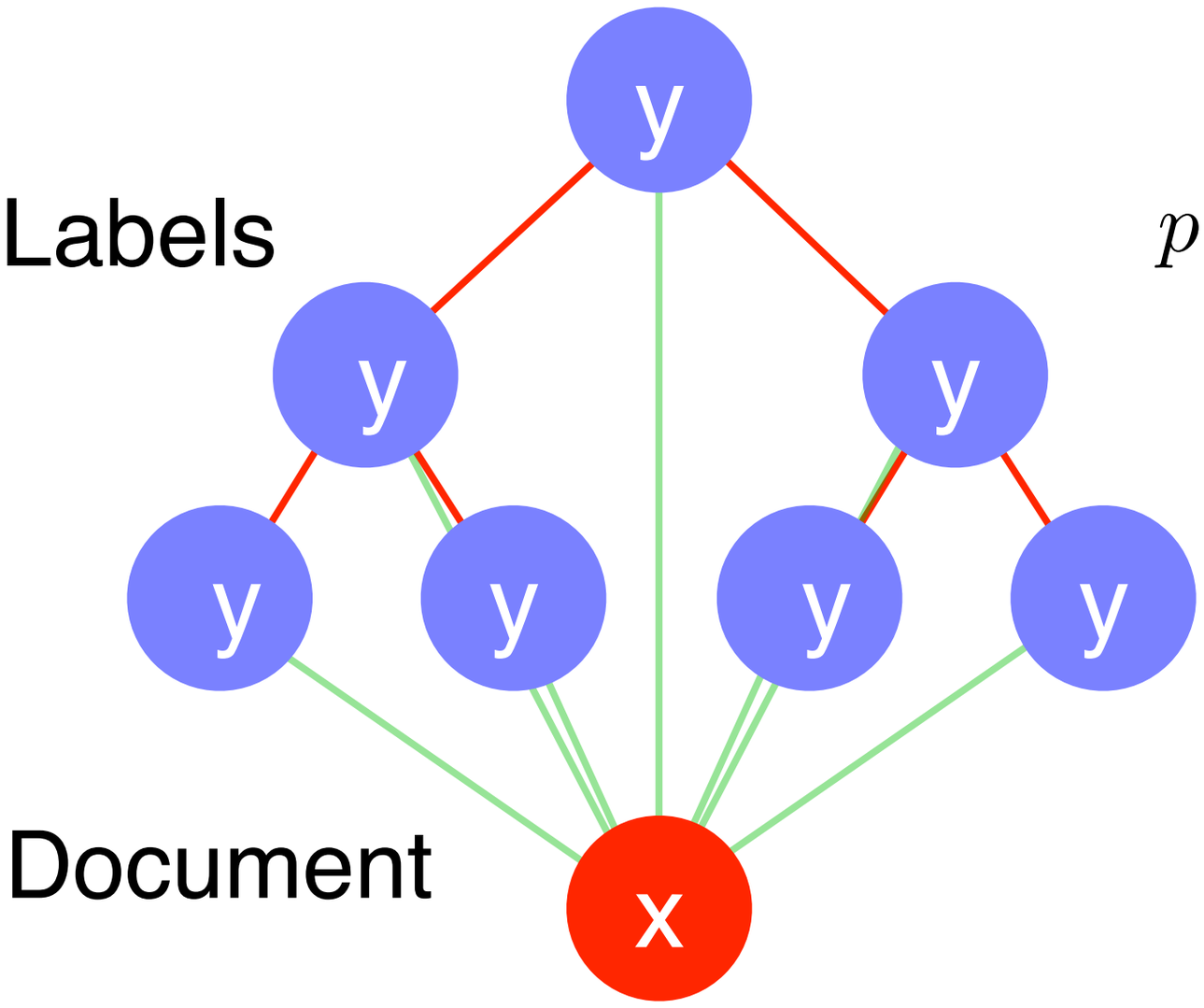
Named Entity Tagging





$$p(x|y) \propto \prod_i \underbrace{\psi_i^x(x_i, x_{i+1})\psi_i^{xy}(x_i, y_i)}_{=: f_i(x_i, x_{i+1})}$$

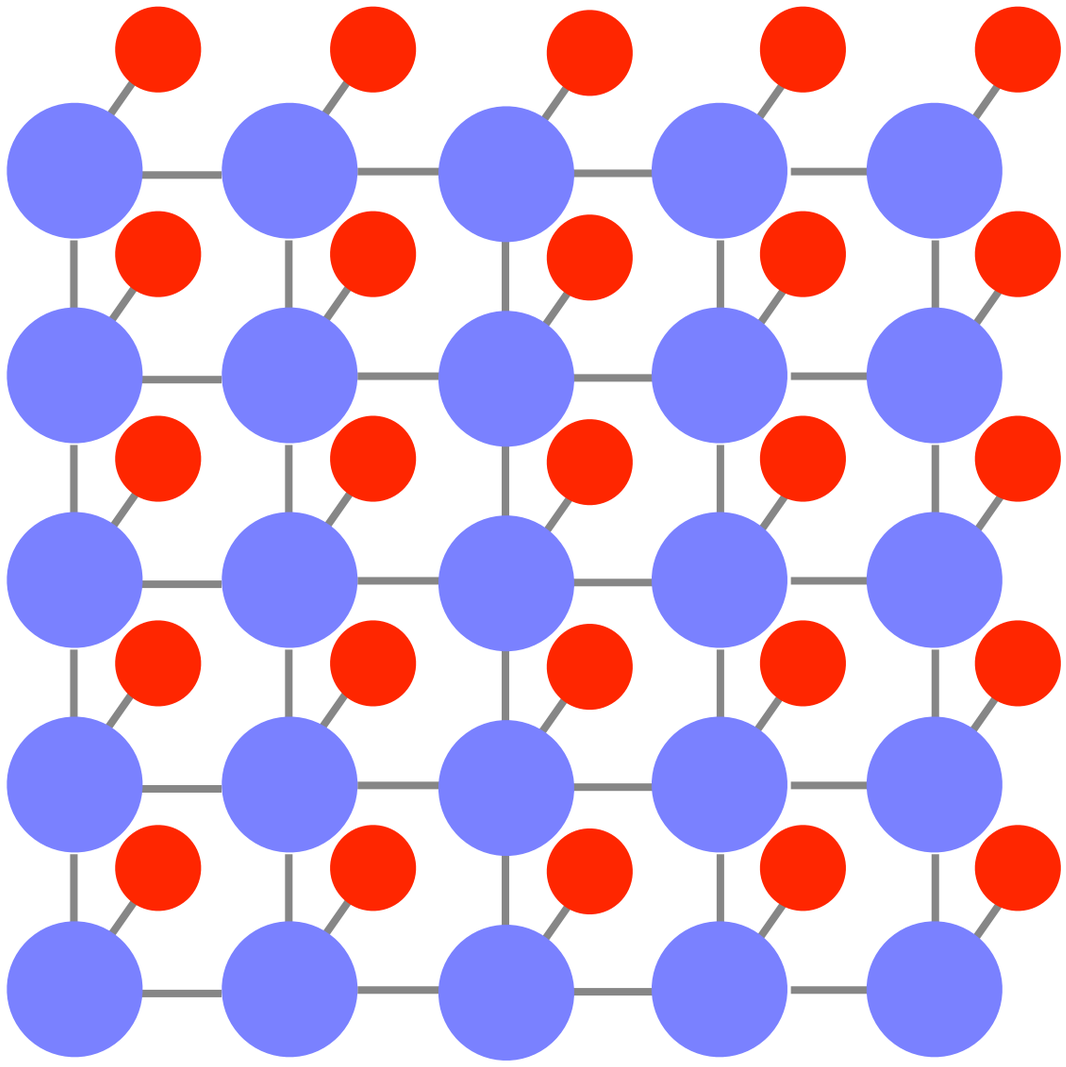
Trees + Ontologies

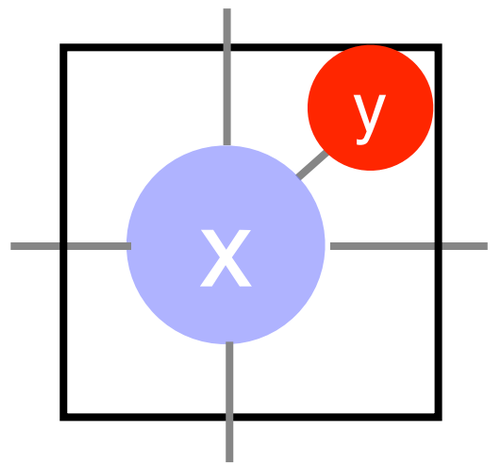


$$p(y|x) = \prod_i \psi(y_i, y_{\text{parent}(i)}, x)$$

- Ontology classification (e.g. YDir, DMOZ)

Spin Glasses + Images



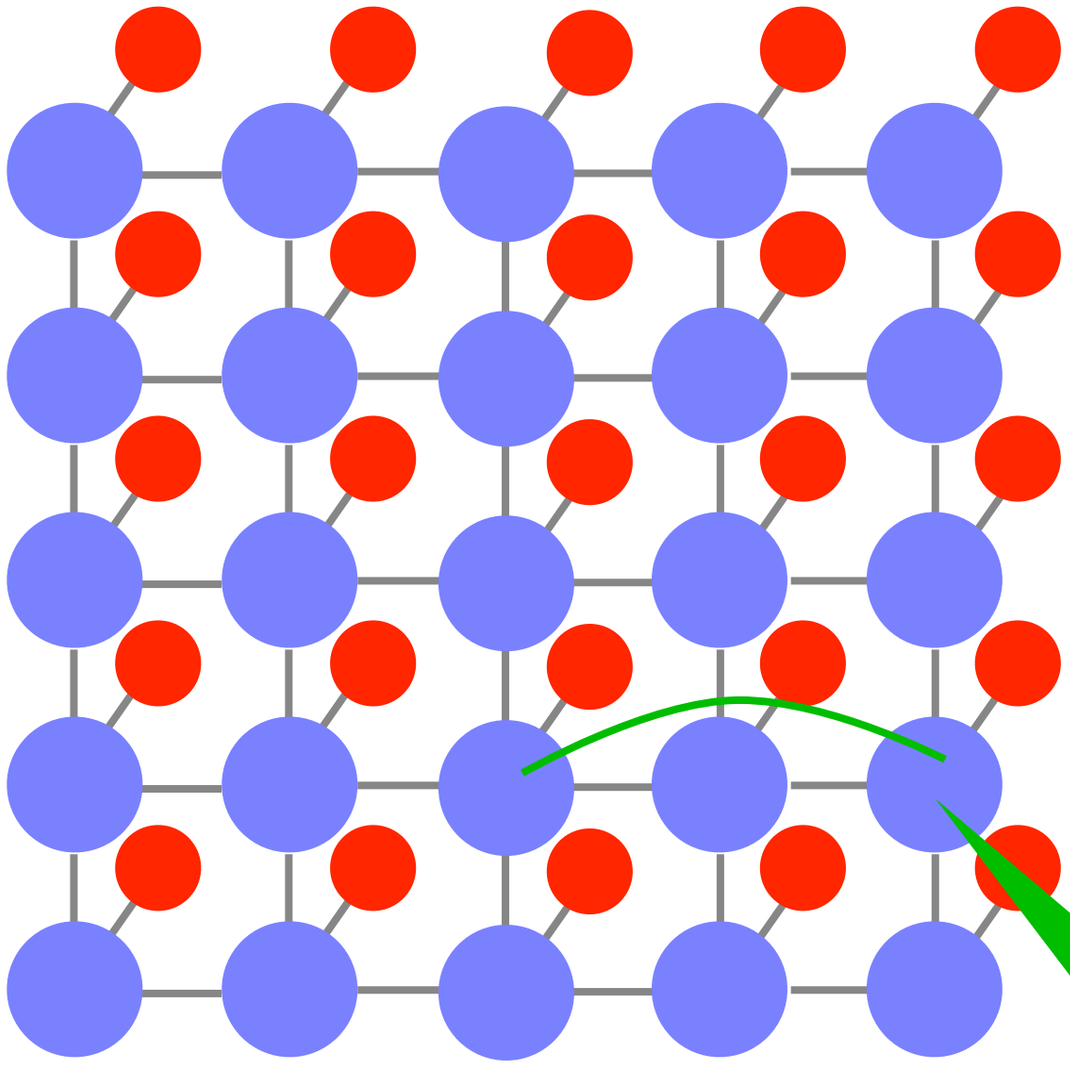


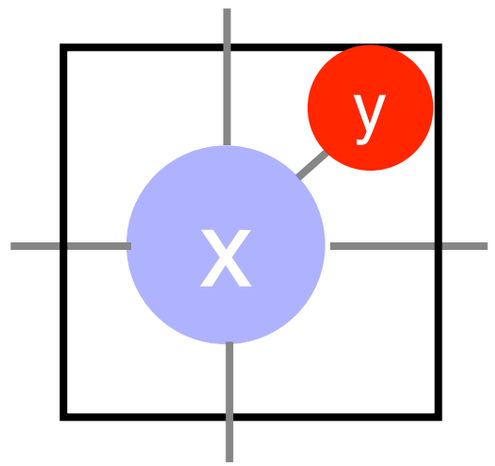
observed pixels

real image

$$p(x|y) = \prod_{ij} \psi^{\text{right}}(x_{ij}, x_{i+1,j}) \psi^{\text{up}}(x_{ij}, x_{i,j+1}) \psi^{xy}(x_{ij}, y_{ij})$$

Spin Glasses + Images





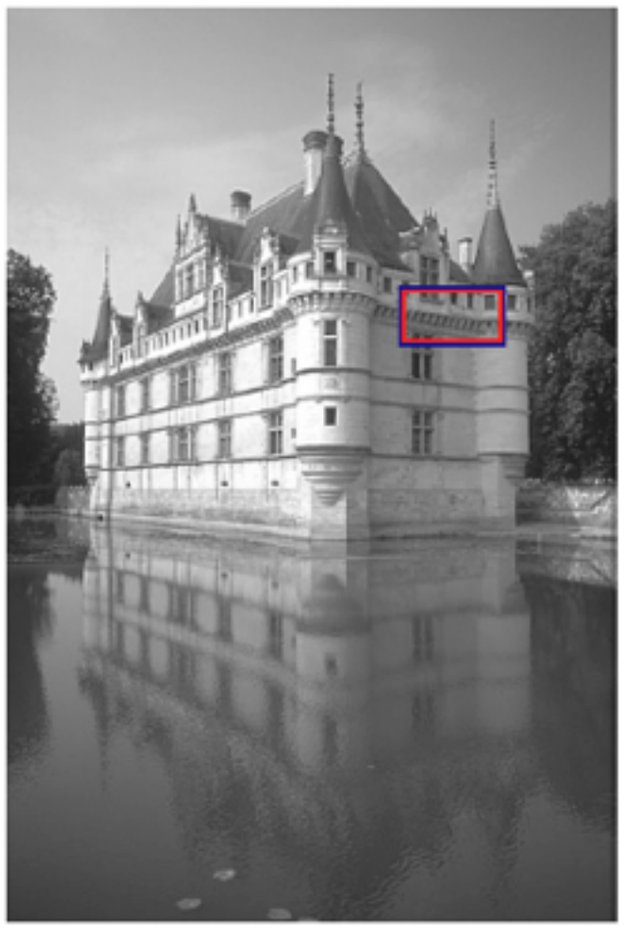
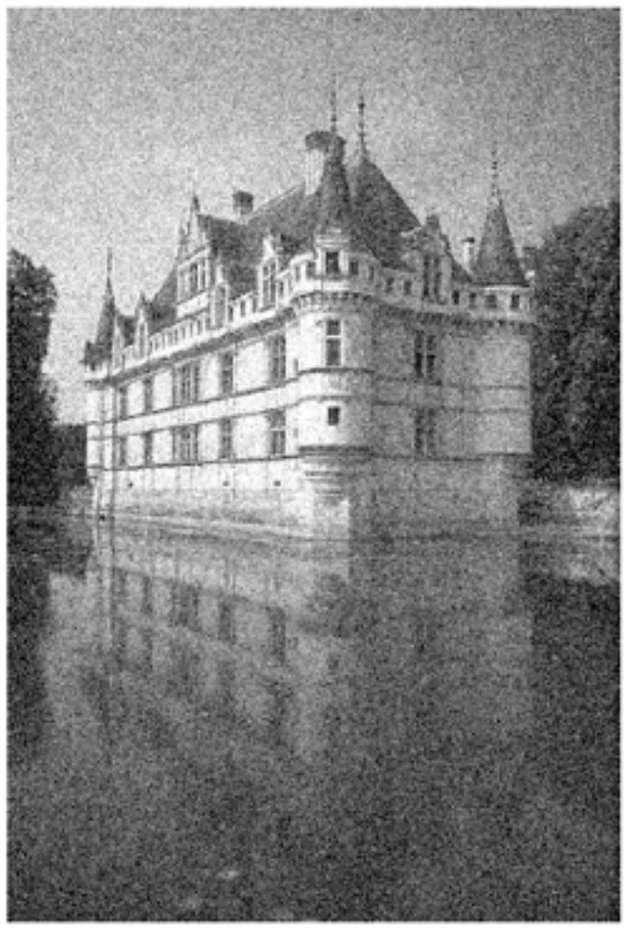
observed pixels

real image

long range interactions

$$p(x|y) = \prod_{ij} \psi^{\text{right}}(x_{ij}, x_{i+1,j}) \psi^{\text{up}}(x_{ij}, x_{i,j+1}) \psi^{xy}(x_{ij}, y_{ij})$$

Image Denoising

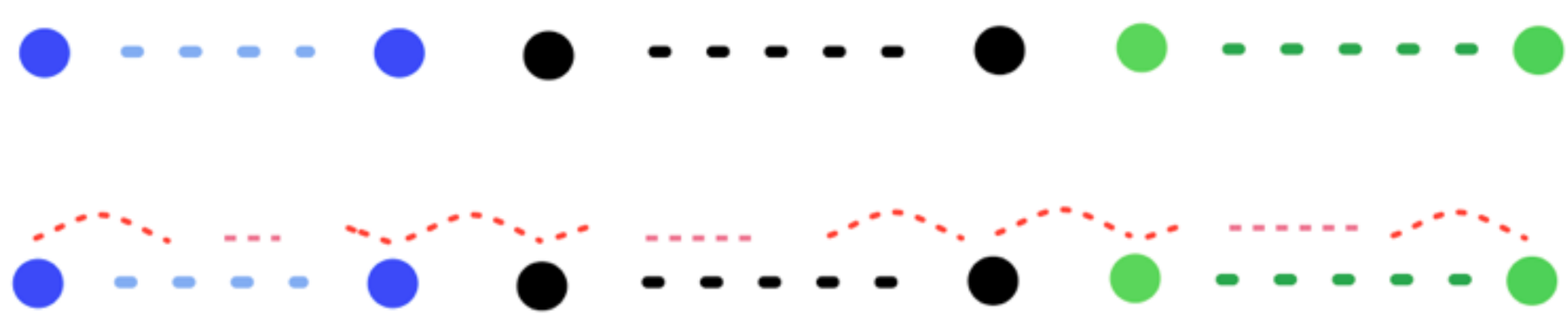







Li&Huttenlocher, ECCV'08

Semi-Markov Models



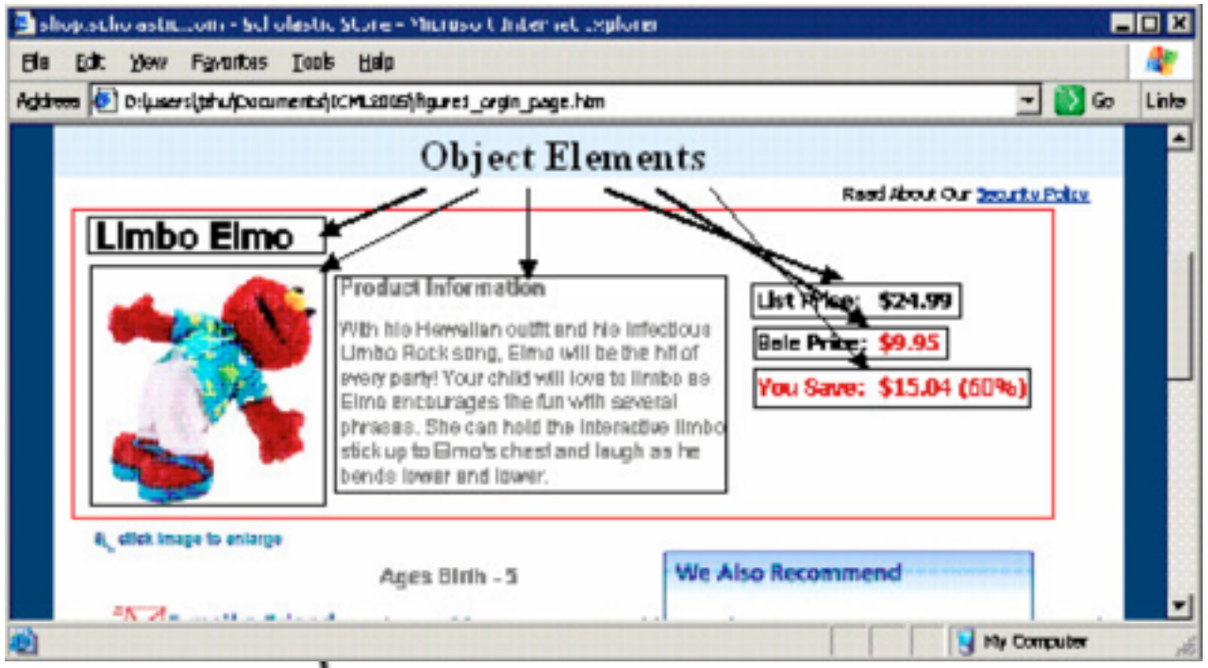
classification

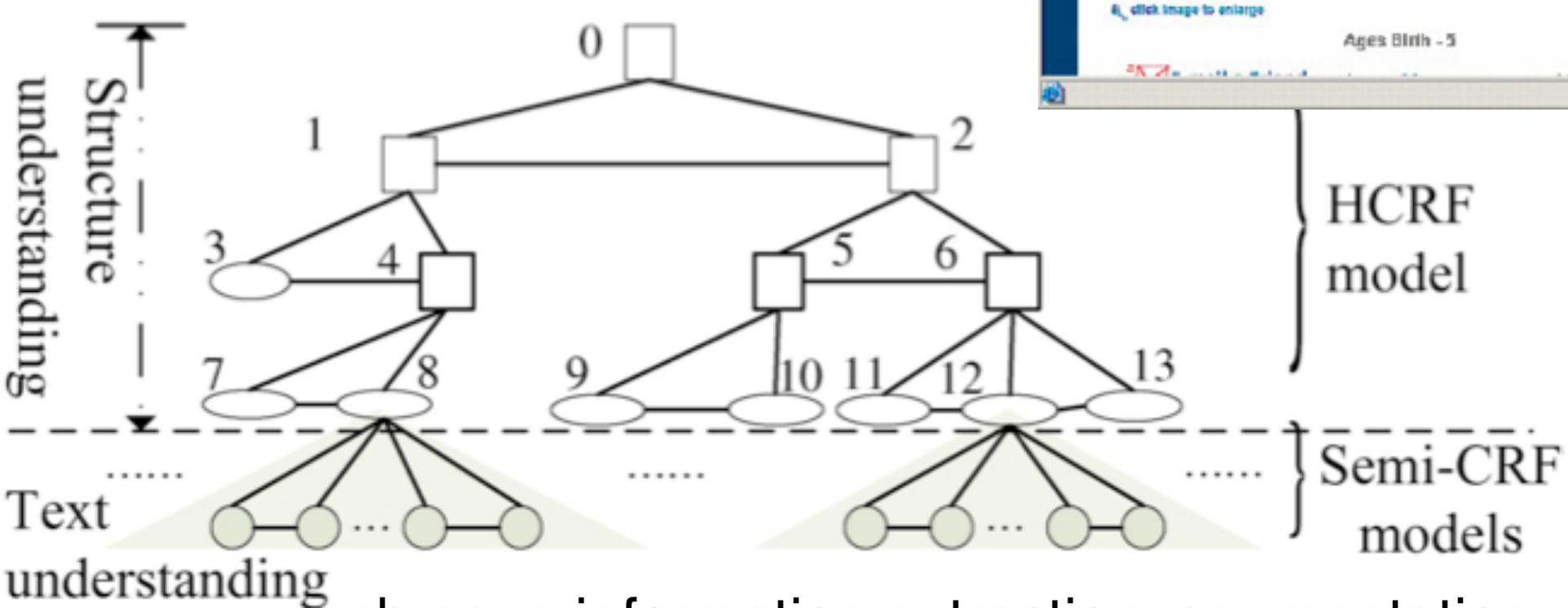
CRF

SMM

phrase segmentation, activity recognition, motion data analysis
Shi, Smola, Altun, Vishwanathan, Li, 2007-2009

2D CRF for Webpages





web page information extraction, segmentation, annotation

Bo, Zhu, Nie, Wen, Hon, 2005-2007

Carnegie Mellon University