MACHINE LEARNING DEPARTMENT

7.1 Directed Graphical Models

7 Graphical Models

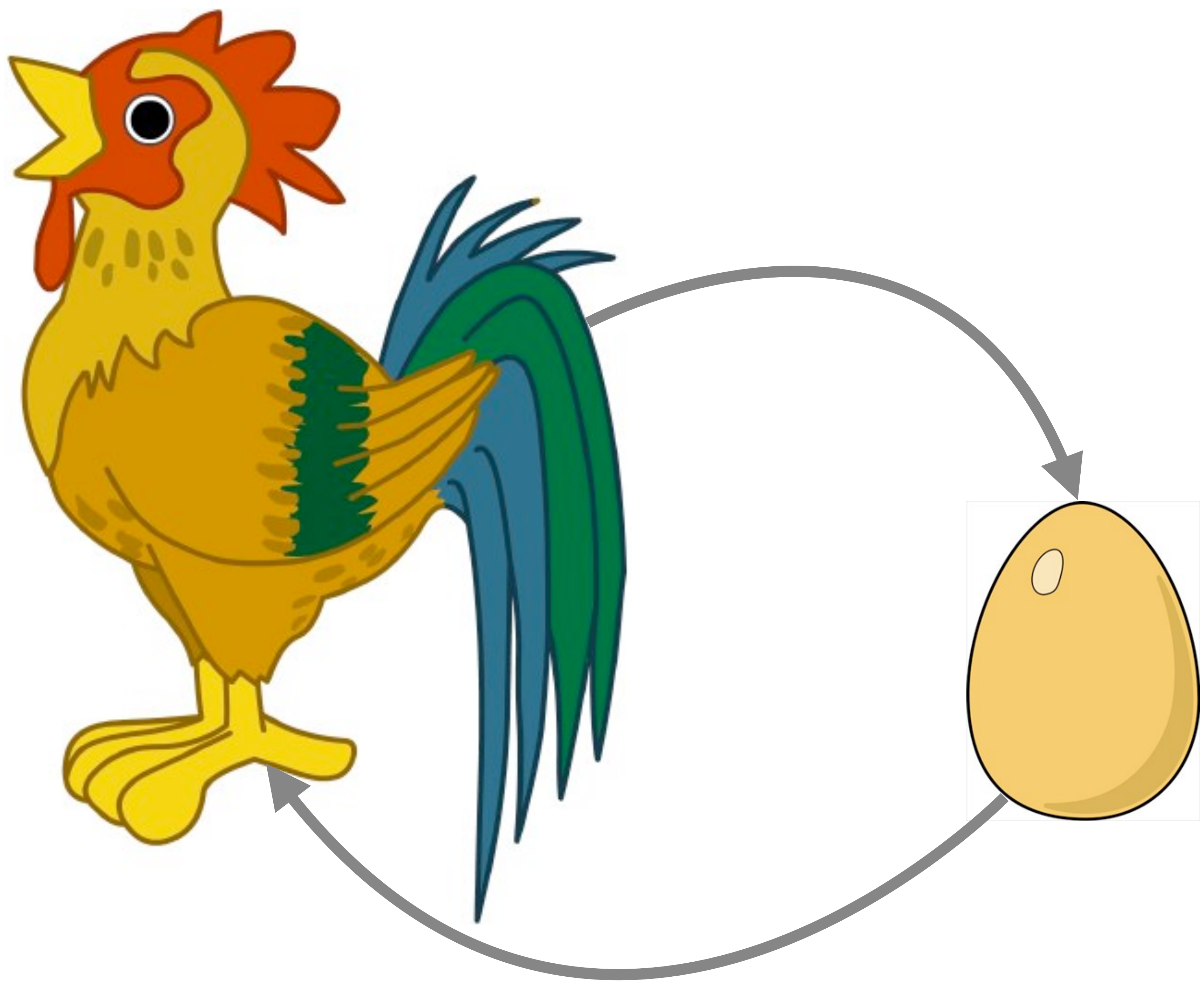
Alexander Smola

Introduction to Machine Learning 10-701

http://alex.smola.org/teaching/10-701-15

Carnegie Mellon University

Directed Graphical Models



Carnegie Mellon University

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

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

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

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

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

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

Brain & Brawn

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

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

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

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

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

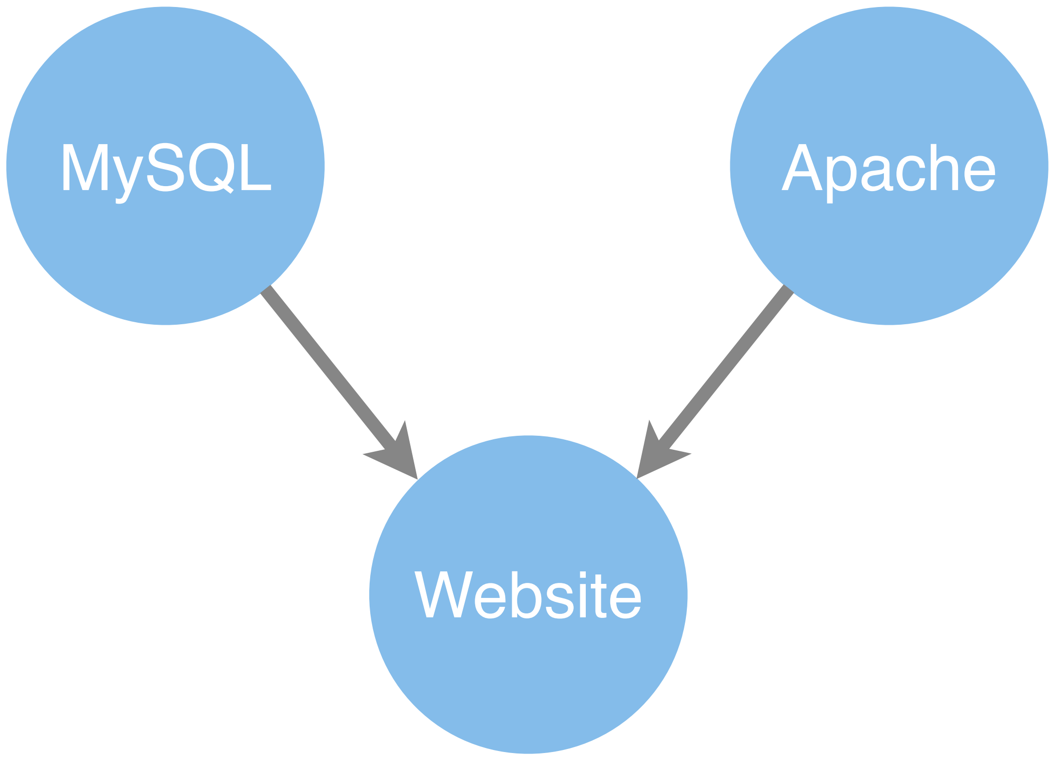
Brain & Brawn

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

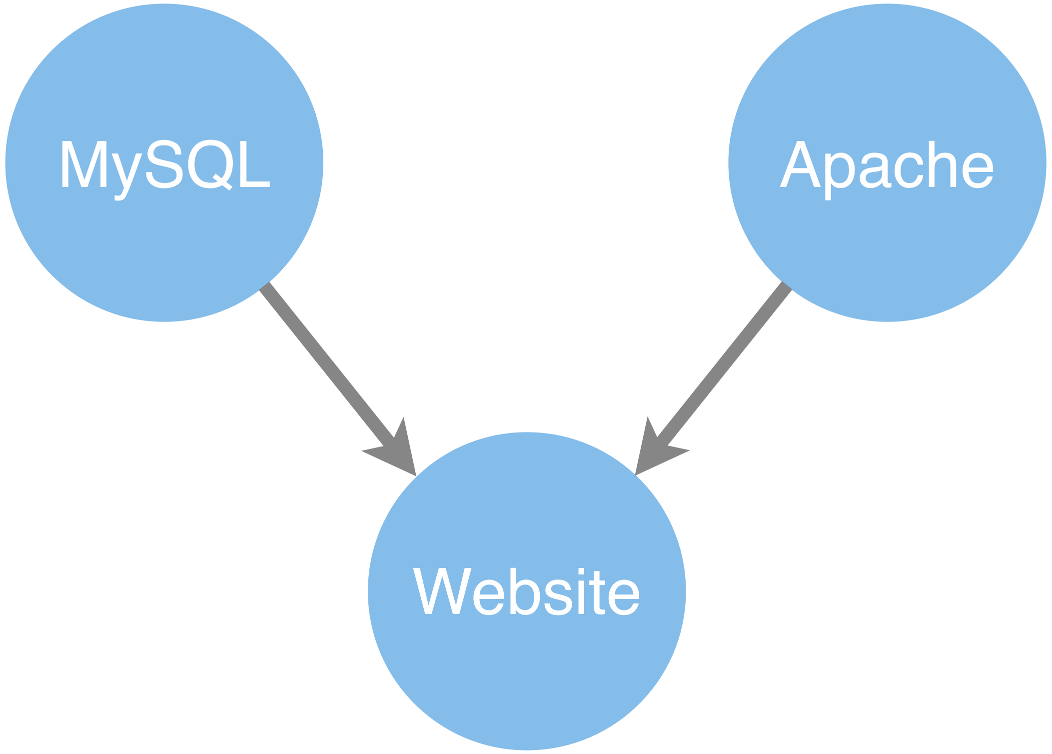
$$p(s, b) = \sum_g p(s|g)p(b|g)p(g)$$

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

... some Web 2.0 service

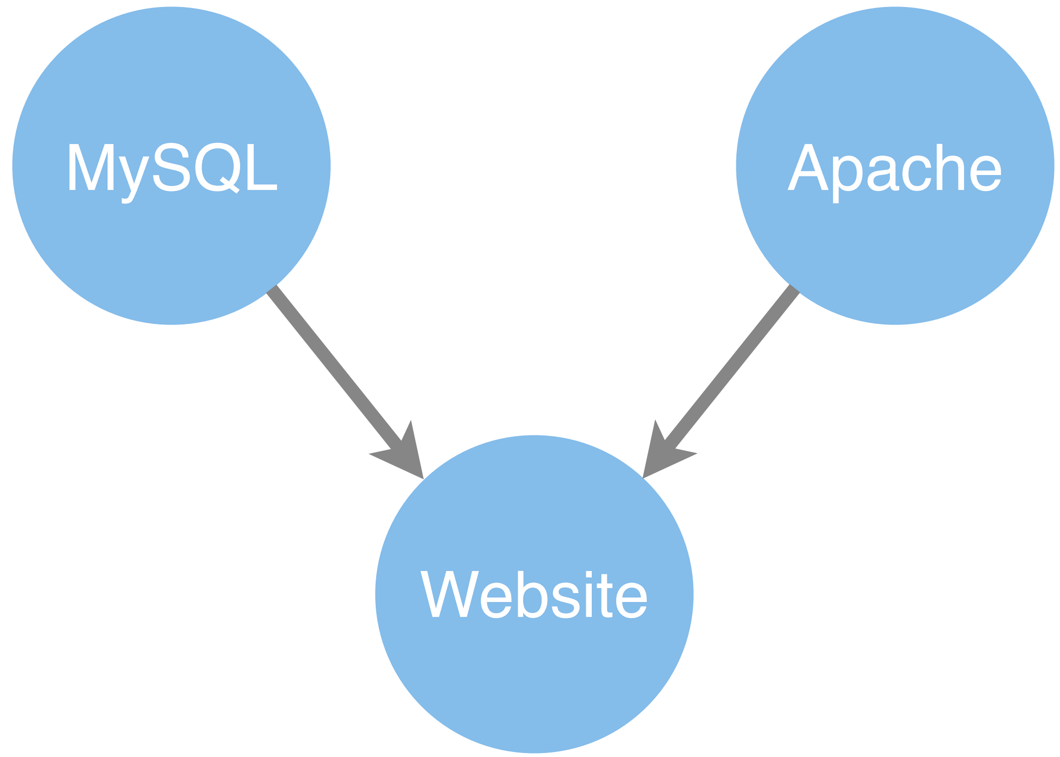


... some Web 2.0 service



- Joint distribution (assume a and m are independent)

... some Web 2.0 service



- Joint distribution (assume a and m are independent)

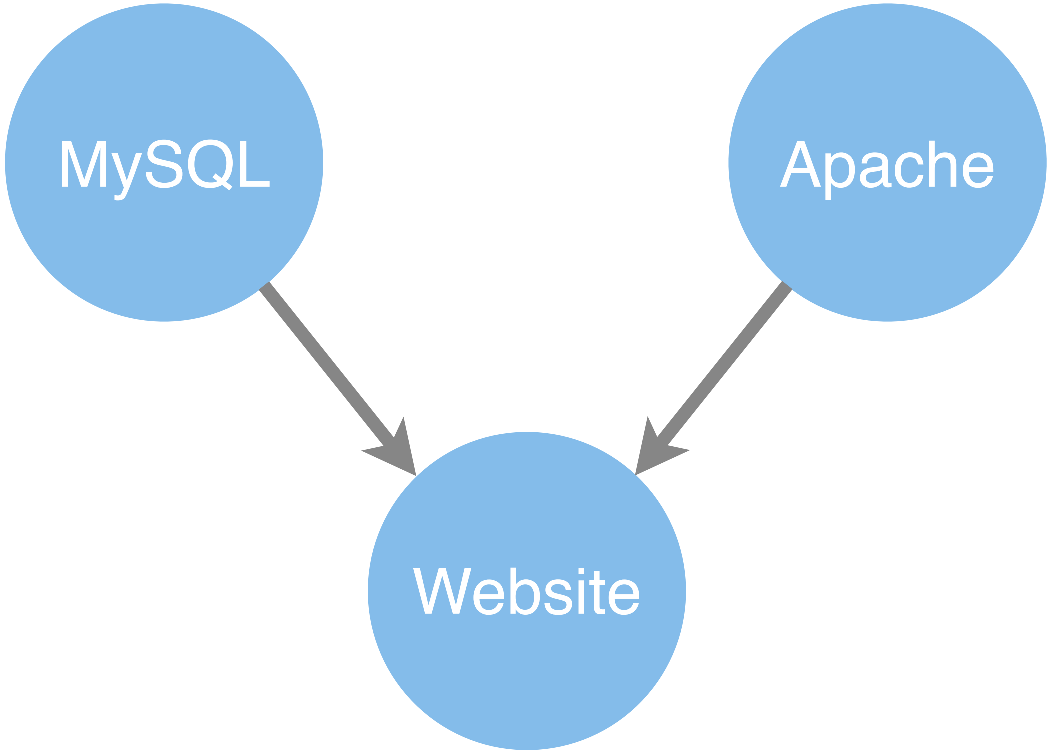
$$p(m, a, w) = p(w|m, a)p(m)p(a)$$

- Explaining away

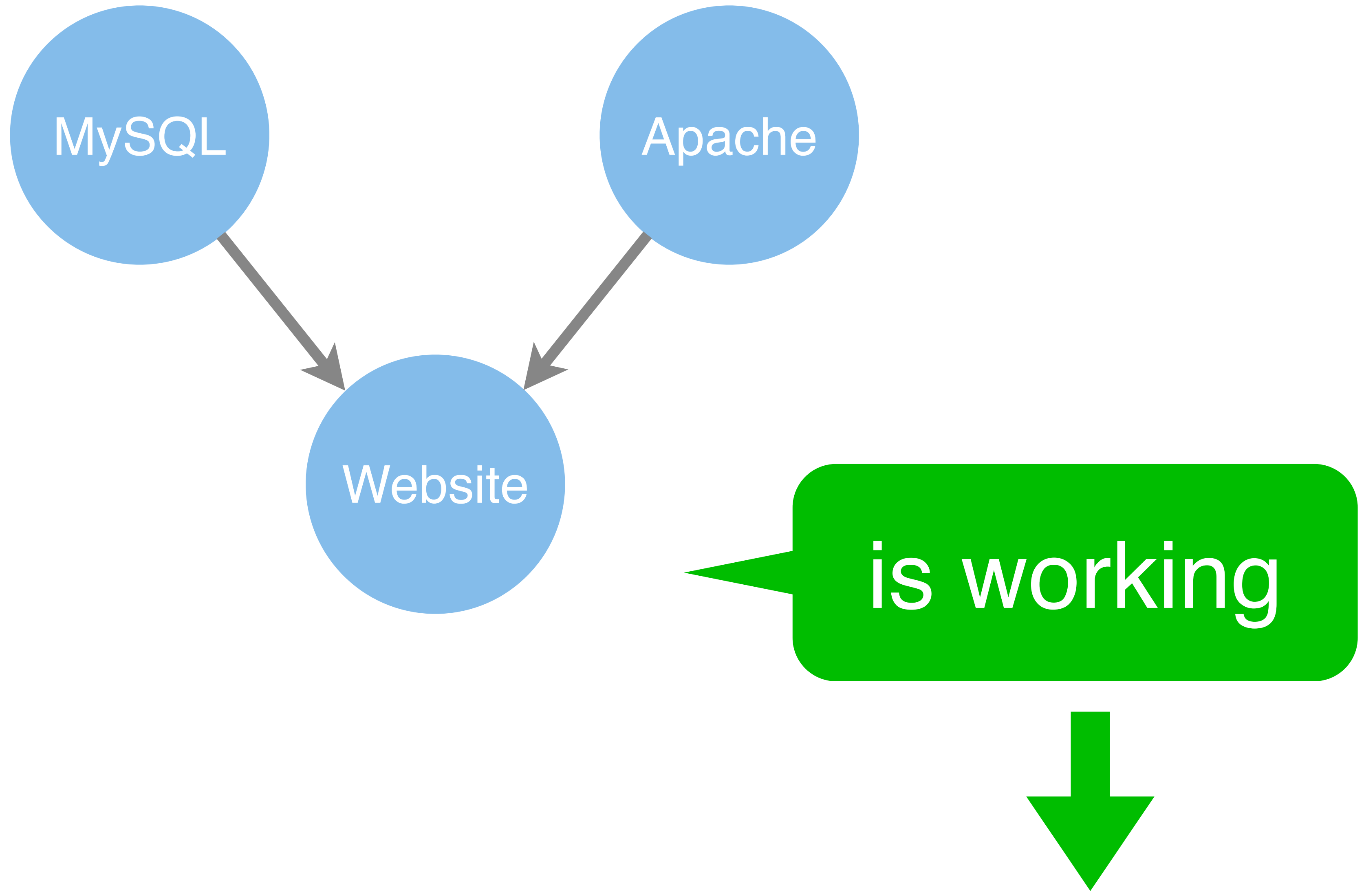
$$p(m, a|w) = \frac{p(w|m, a)p(m)p(a)}{\sum_{m', a'} p(w|m', a')p(m')p(a')}$$

a and m are dependent conditioned on w

... some Web 2.0 service

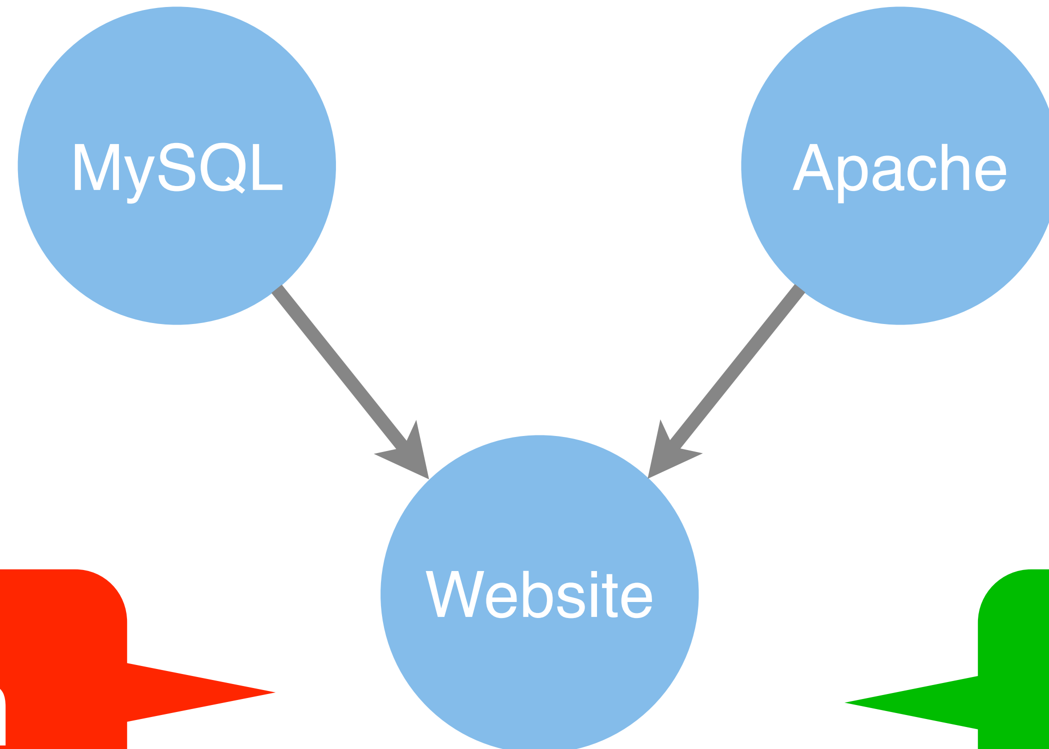


... some Web 2.0 service



MySQL is working
Apache is working

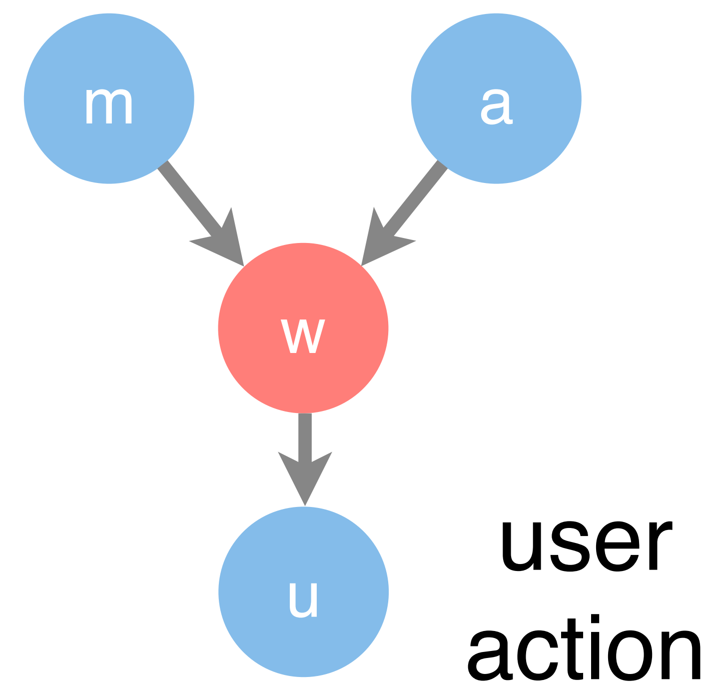
... some Web 2.0 service



is broken → At least one of the two services is broken (not independent)

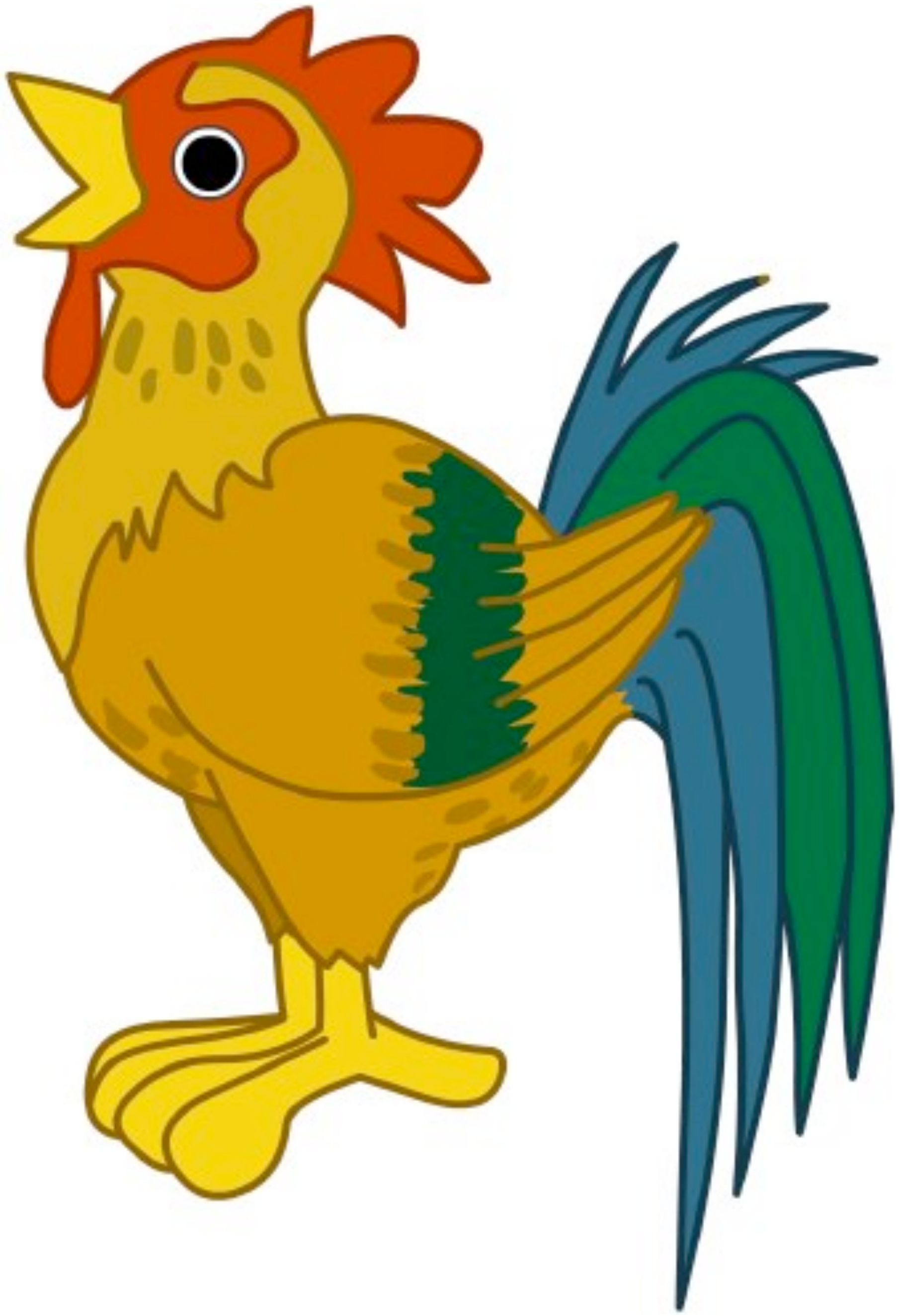
is working → MySQL is working
Apache is working

Directed graphical model

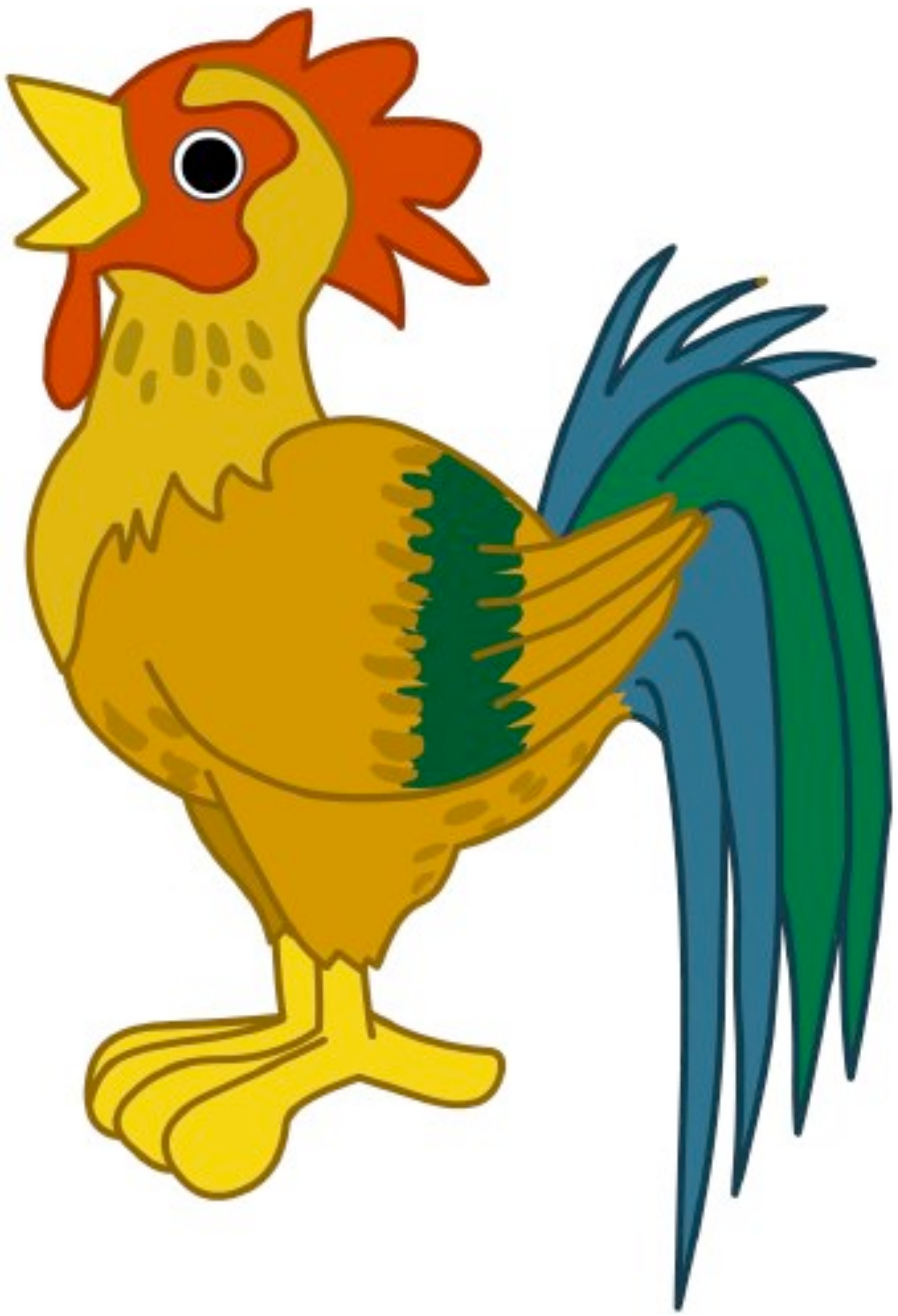


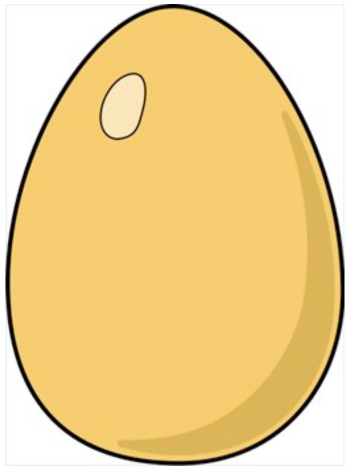
- Easier estimation
  - 15 parameters for full joint distribution
  - 1+1+4+1 for factorizing distribution
- Causal relations
- Inference for unobserved variables

No loops allowed

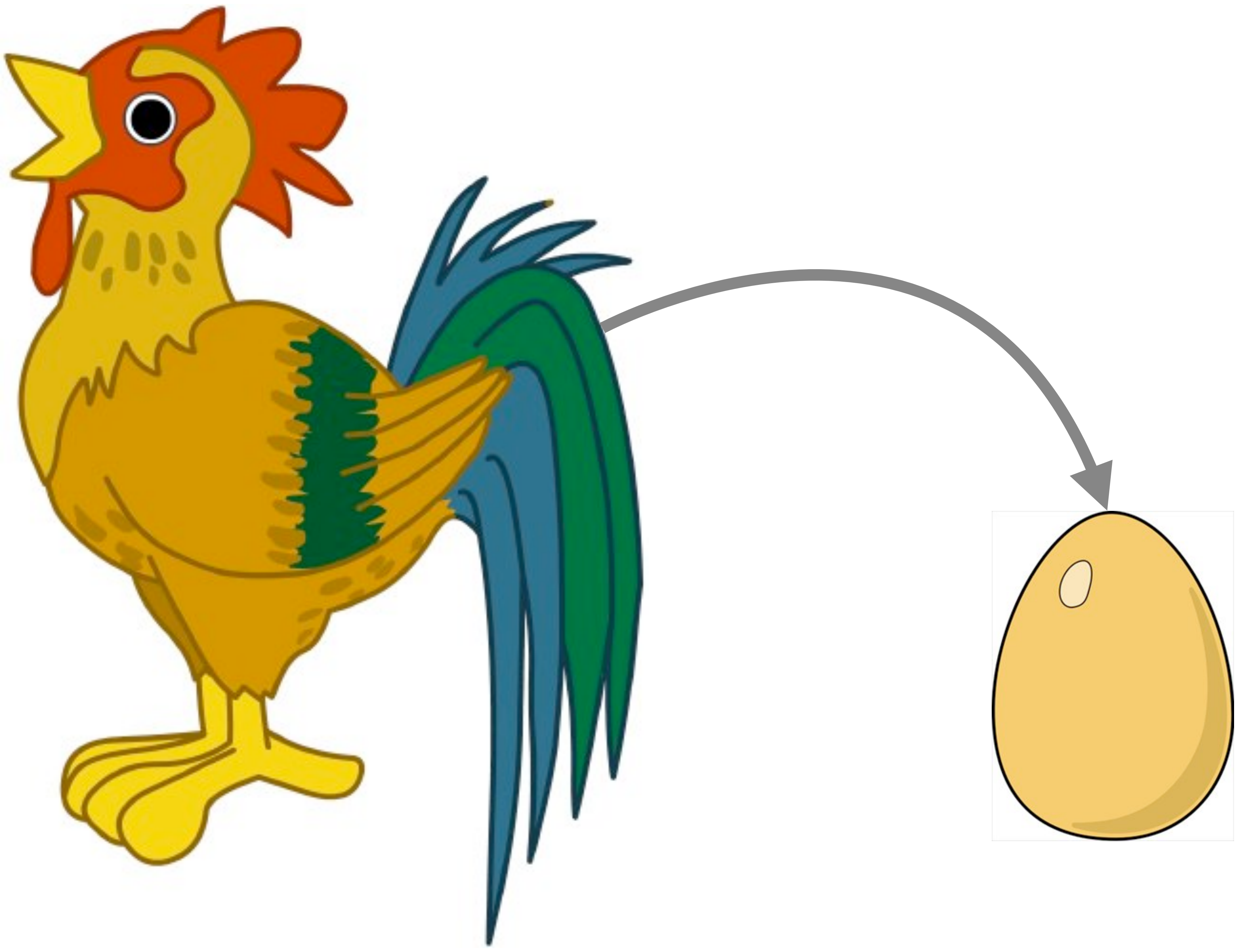


No loops allowed

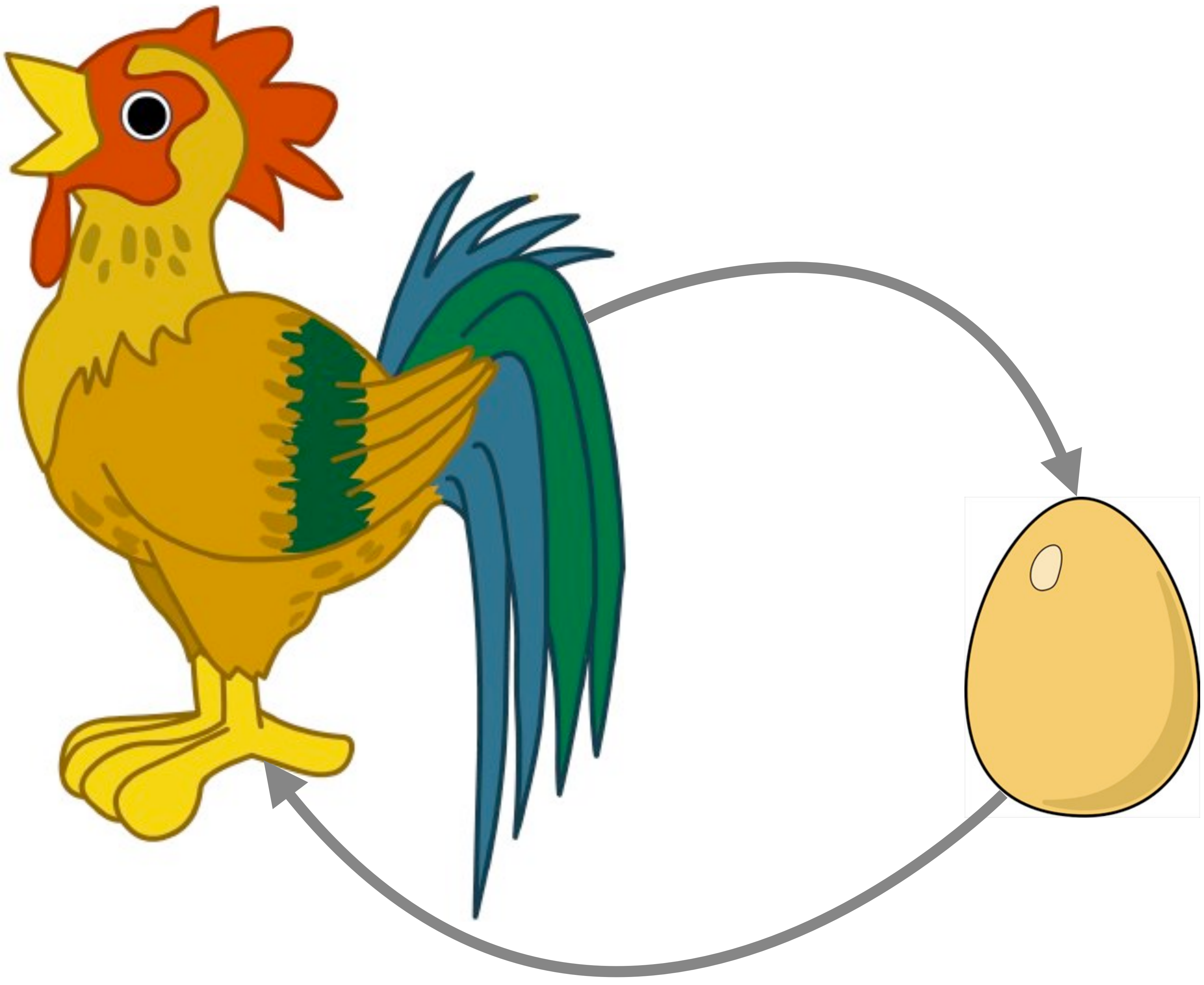




No loops allowed

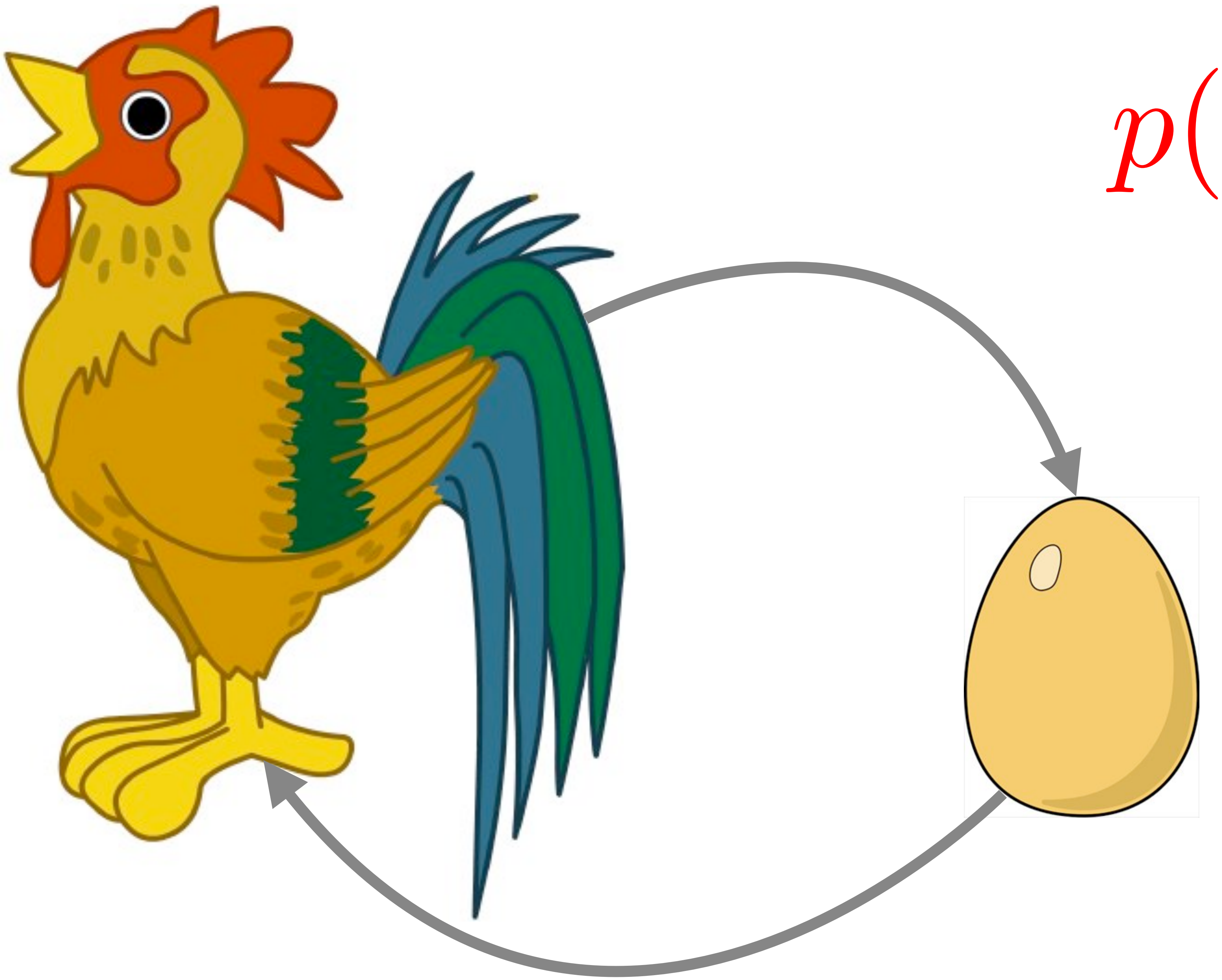


No loops allowed

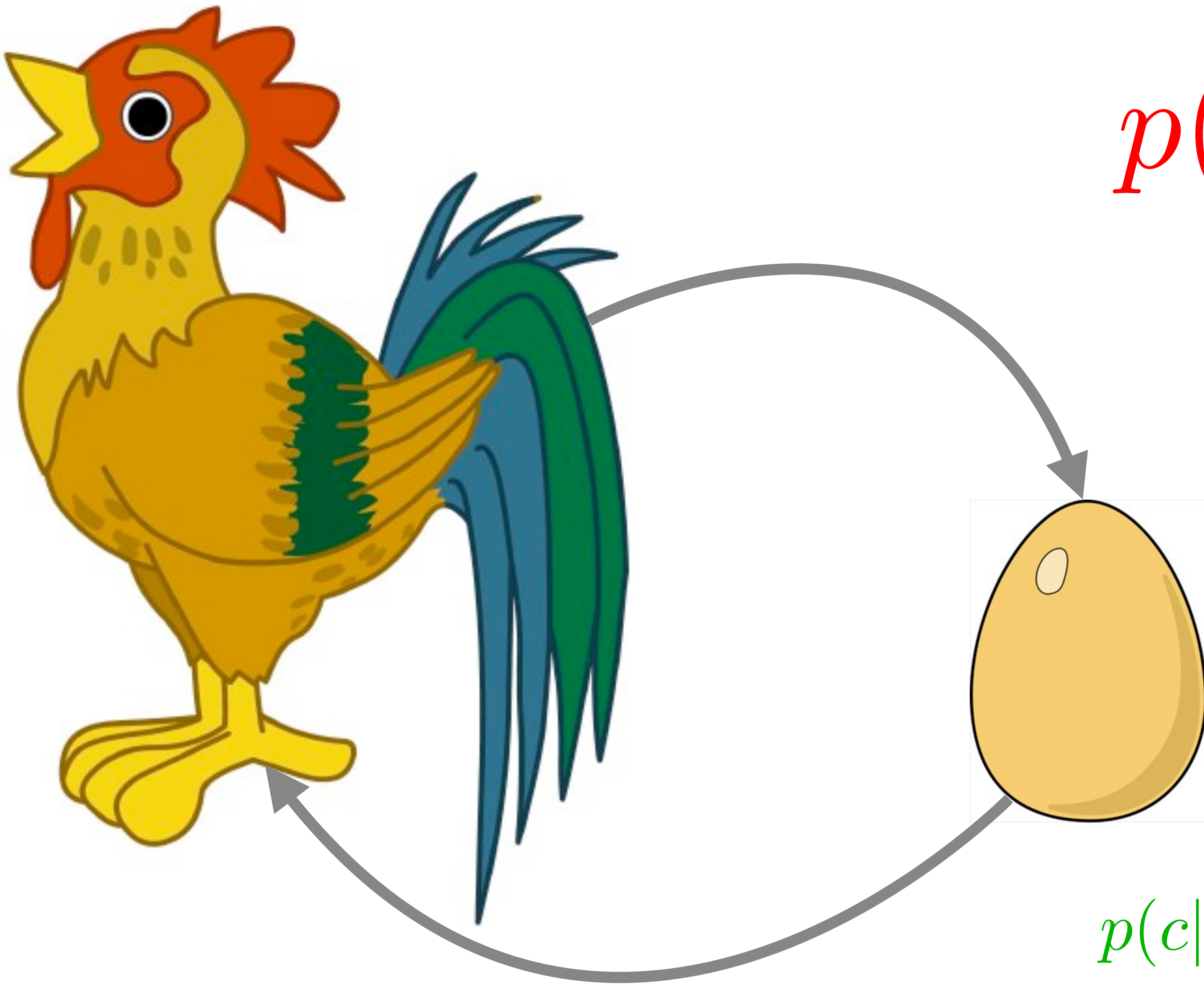


No loops allowed

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



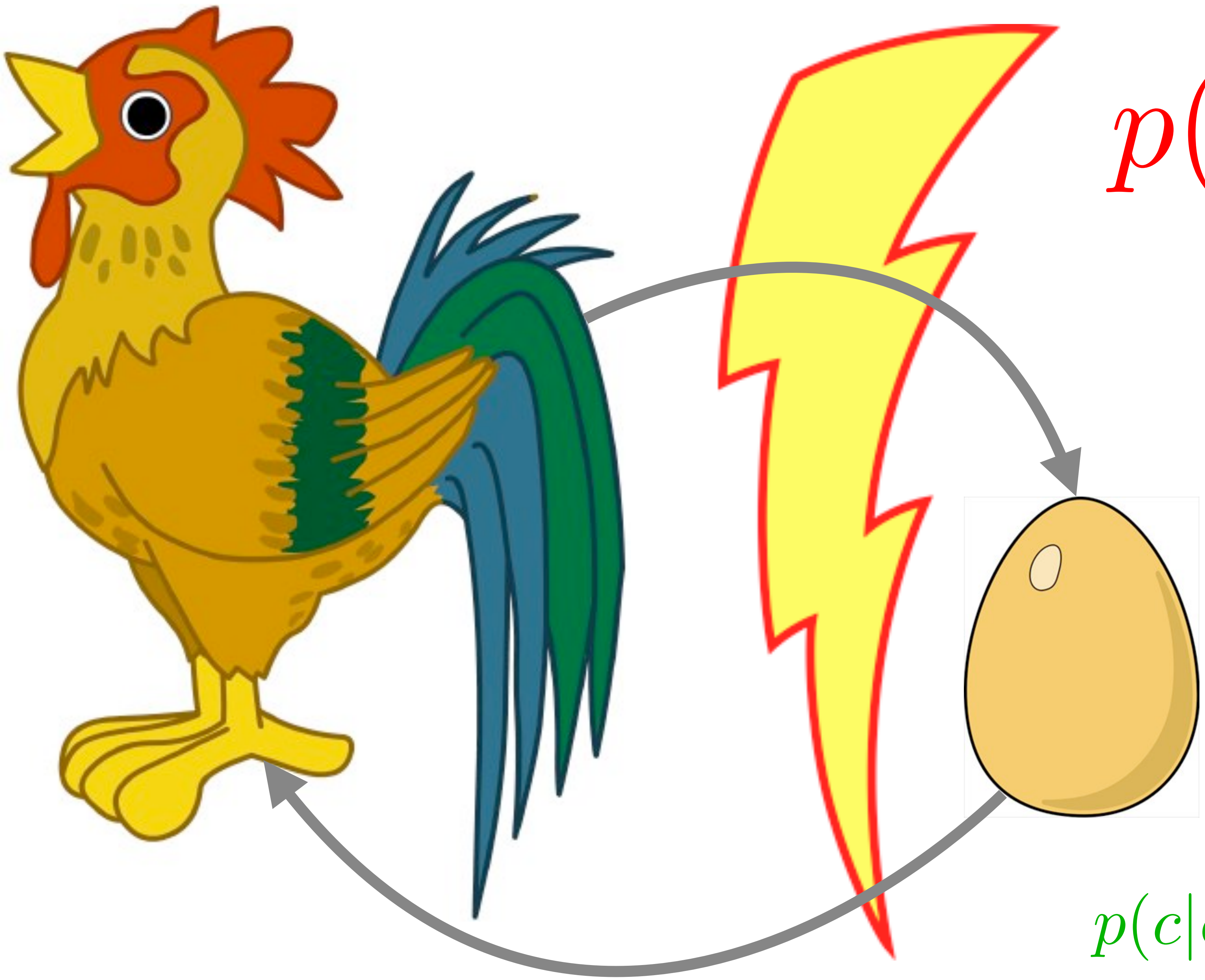
No loops allowed



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

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

No loops allowed

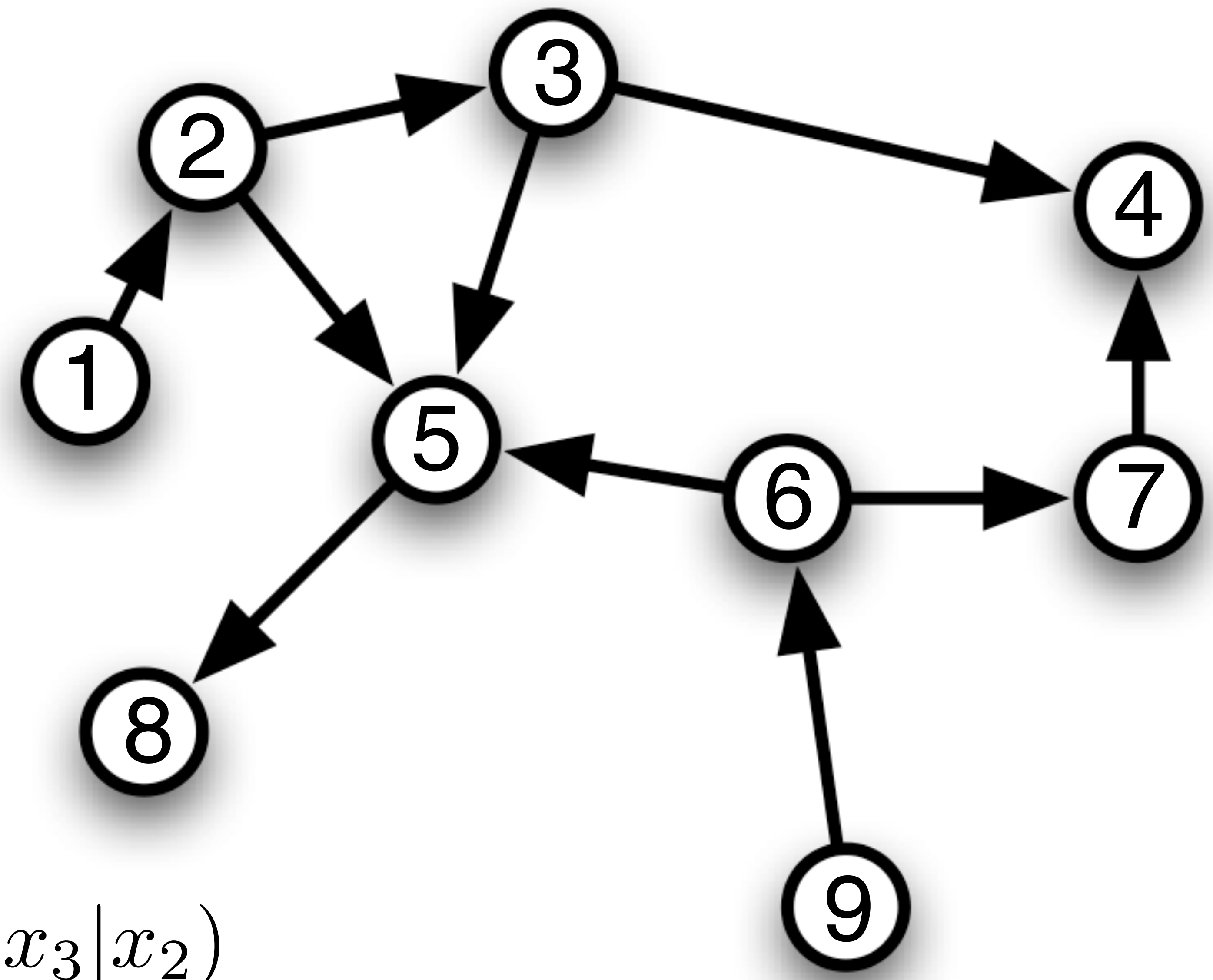


$$p(c|e)p(e|c)$$

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

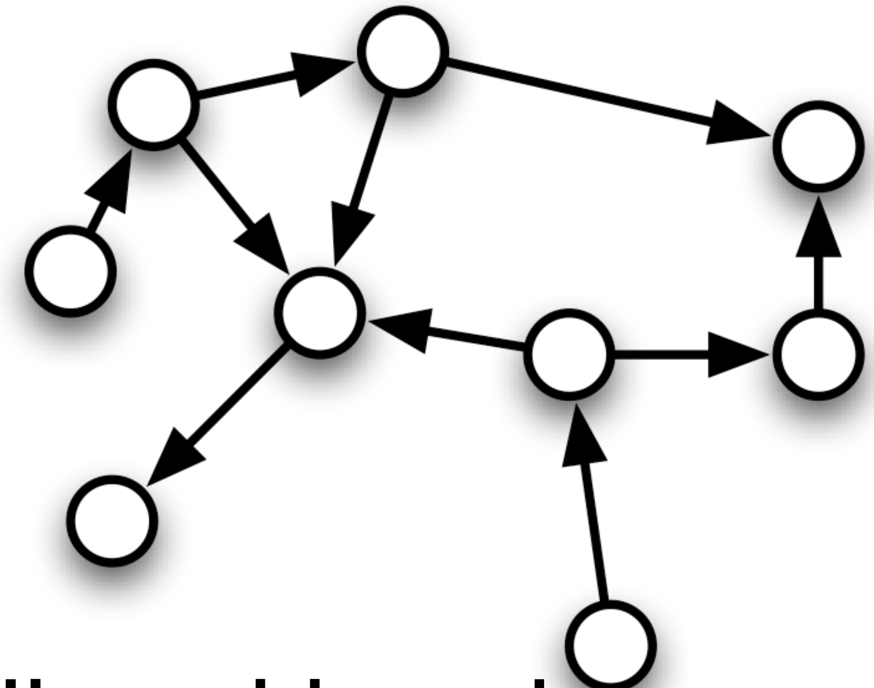
Directed Graphical Model

- Probability distribution
- Iterate over children|parents



$$
\begin{aligned}
p(x) = & p(x_1)p(x_2|x_1)p(x_3|x_2) \\
& p(x_4|x_3, x_7)p(x_5|x_2, x_3, x_6) \\
& p(x_6|x_9)p(x_7|x_6)p(x_8|x_5)p(x_9)
\end{aligned}
$$

Directed Graphical Model



- Joint probability distribution

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

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

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

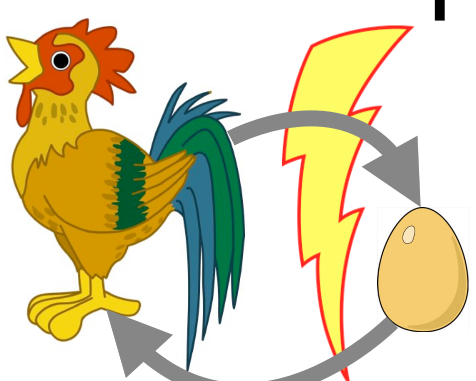
  - If x is partially observed things get interesting maximization, EM, variational, sampling ...
  - If we don't know the structure ...

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

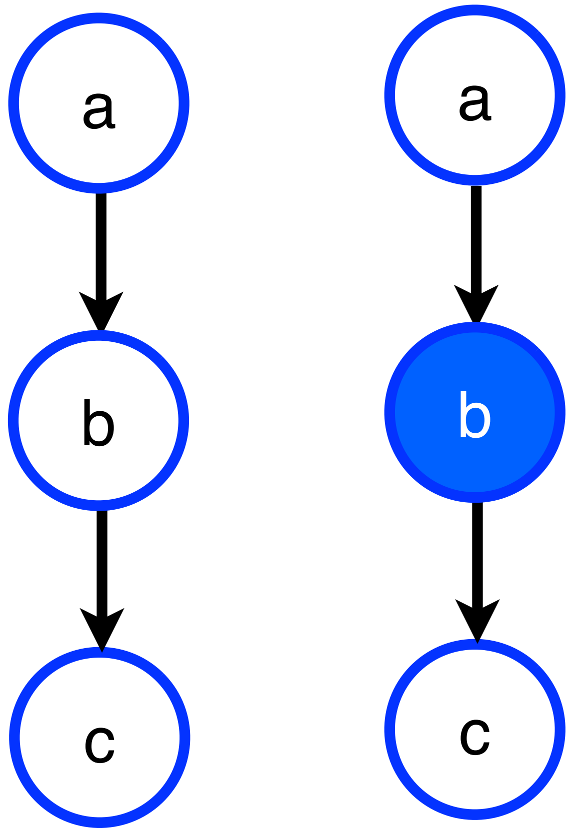
- Joint distribution

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

- 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

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

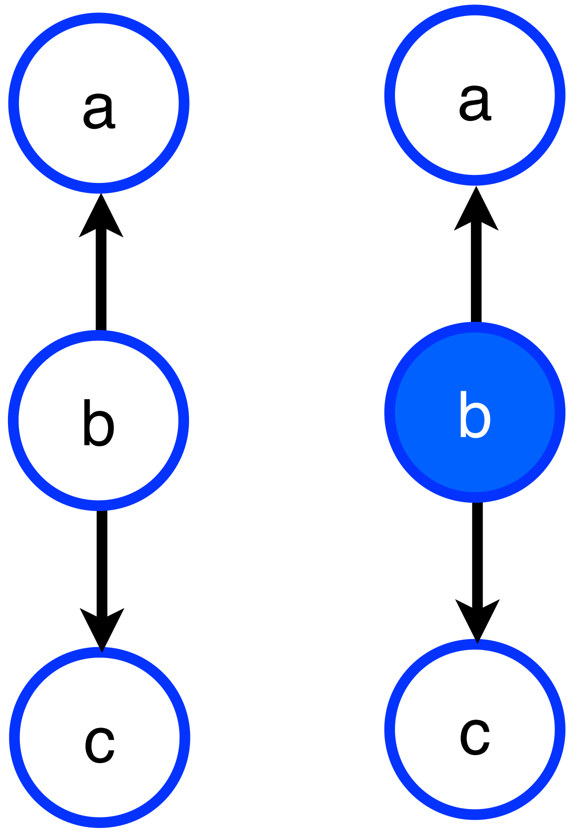
- a and c are dependent

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

- Conditioning on b creates independence

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

$$a \perp c|b$$



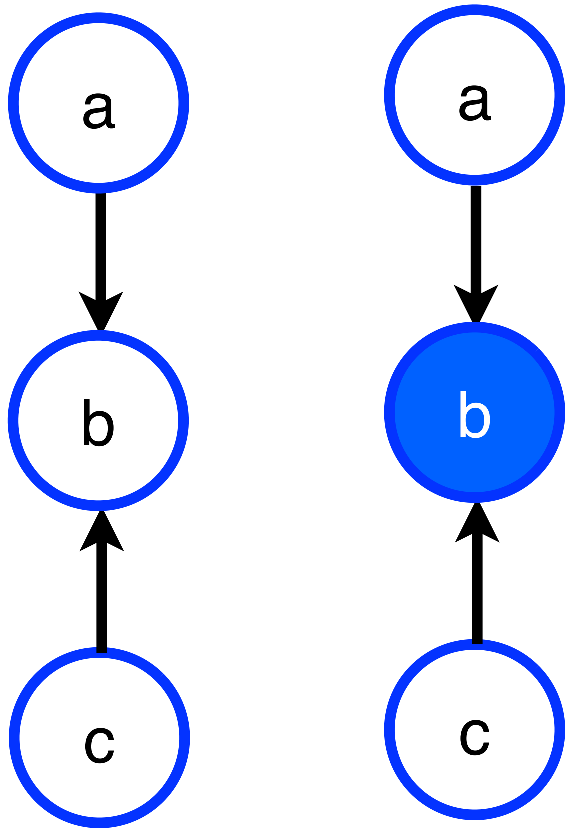
3 Explaining Away

- Joint distribution

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

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

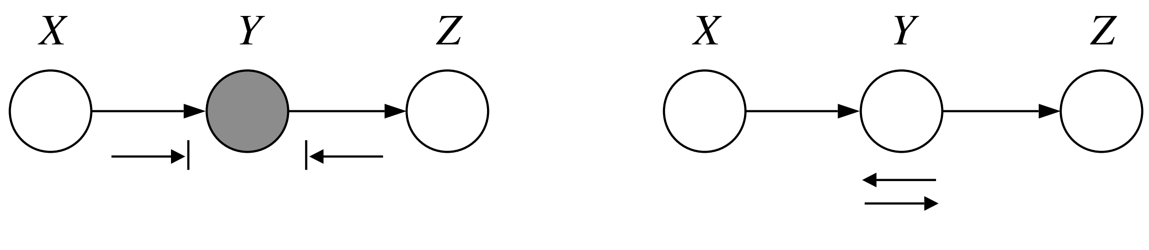
$$p(a, c|b) = \frac{p(a)p(b|a, c)p(c)}{\sum_{a', c'} p(a')p(b|a', c')p(c')}$$



d-Separation

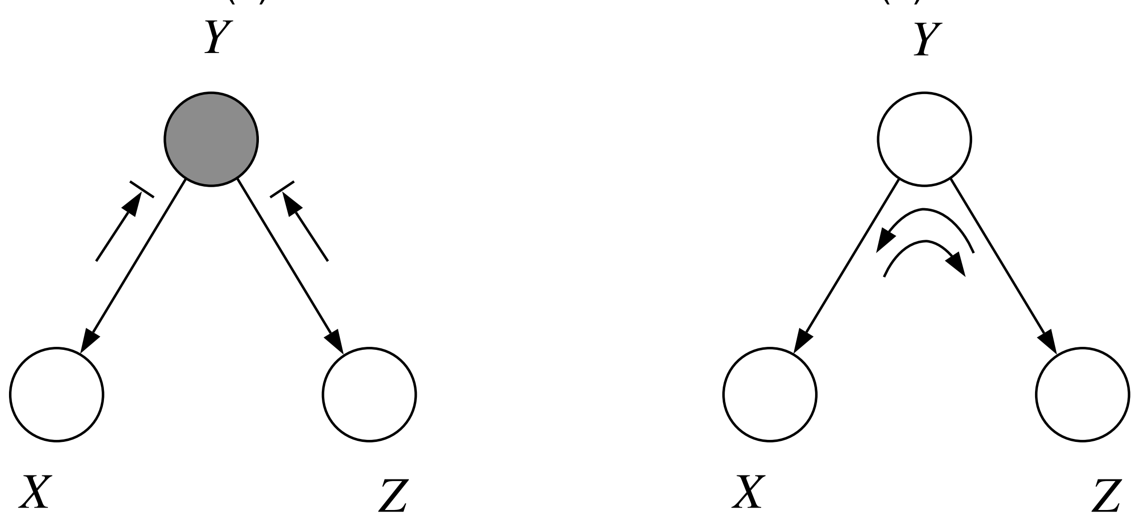
- Given general directed acyclic graph (DAG)
- Determine whether sets A, B of random variables are conditionally independent given C
- Simple algorithm - reachability
  - Start in in vertex of A
  - Check whether any vertex in B can be reached
  - If separated, we have conditional independence

Transition rules



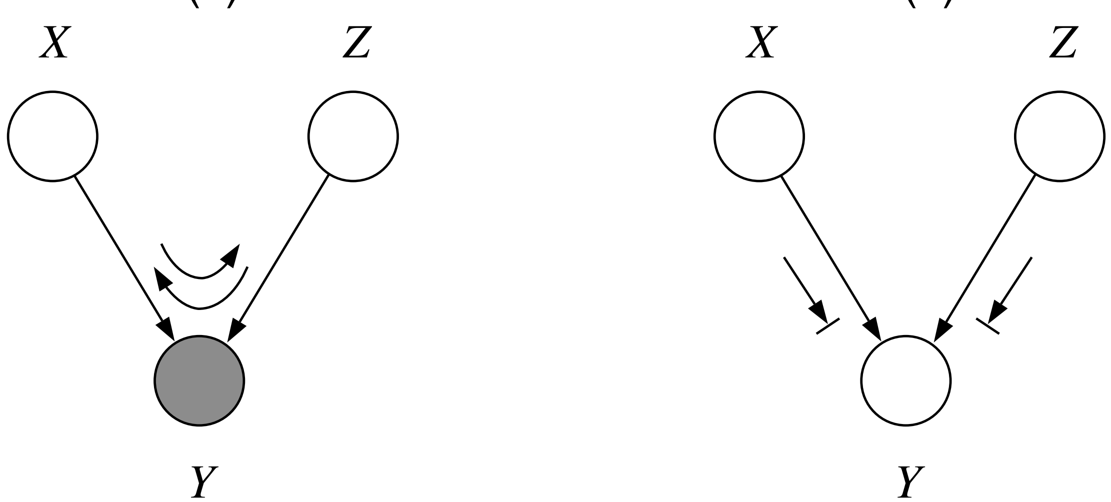
(a) (b)

(a) (b)



(a) (b)

(a) (b)

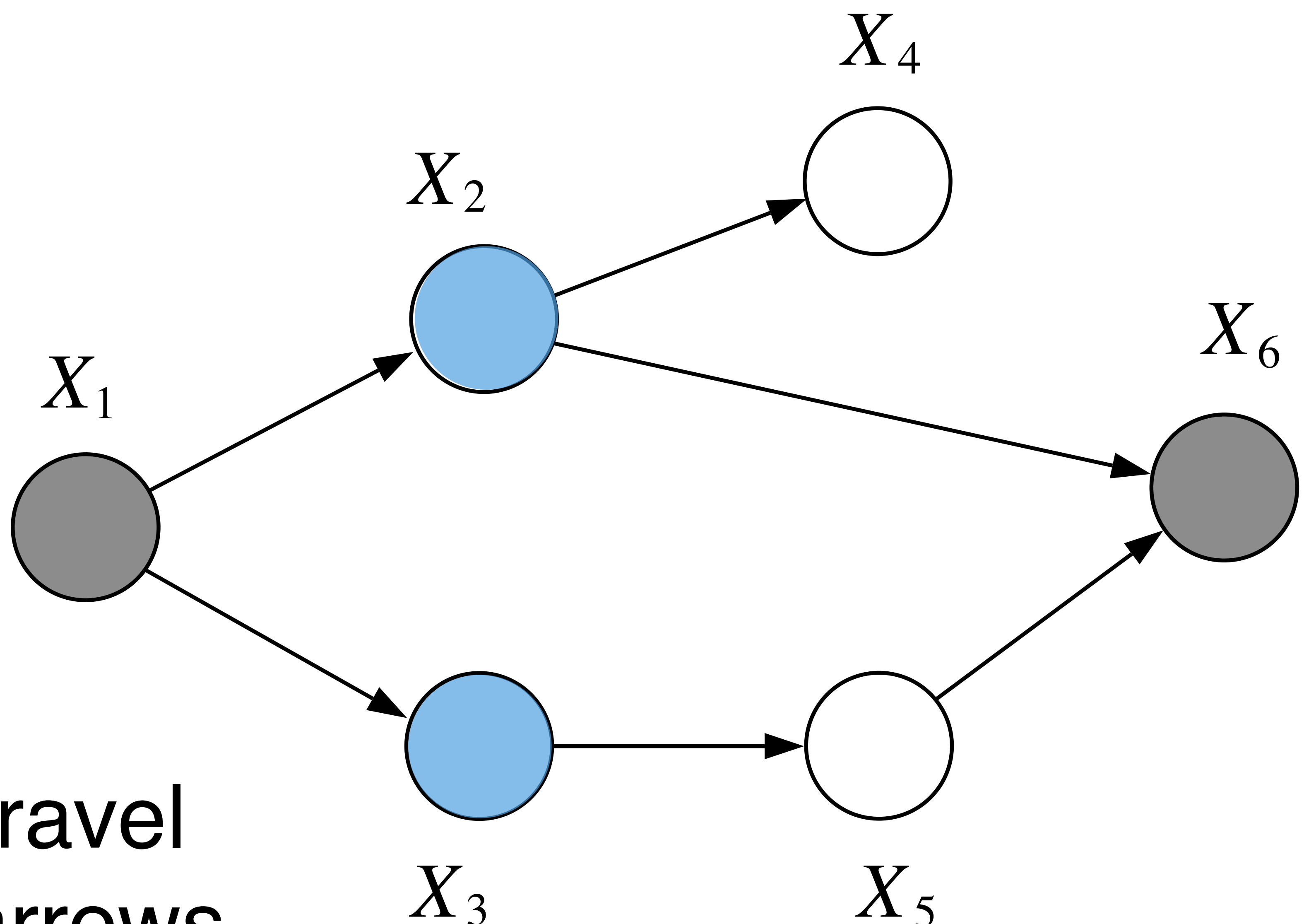


(a) (b)

Courtesy of Sam Roweis

Transition rules

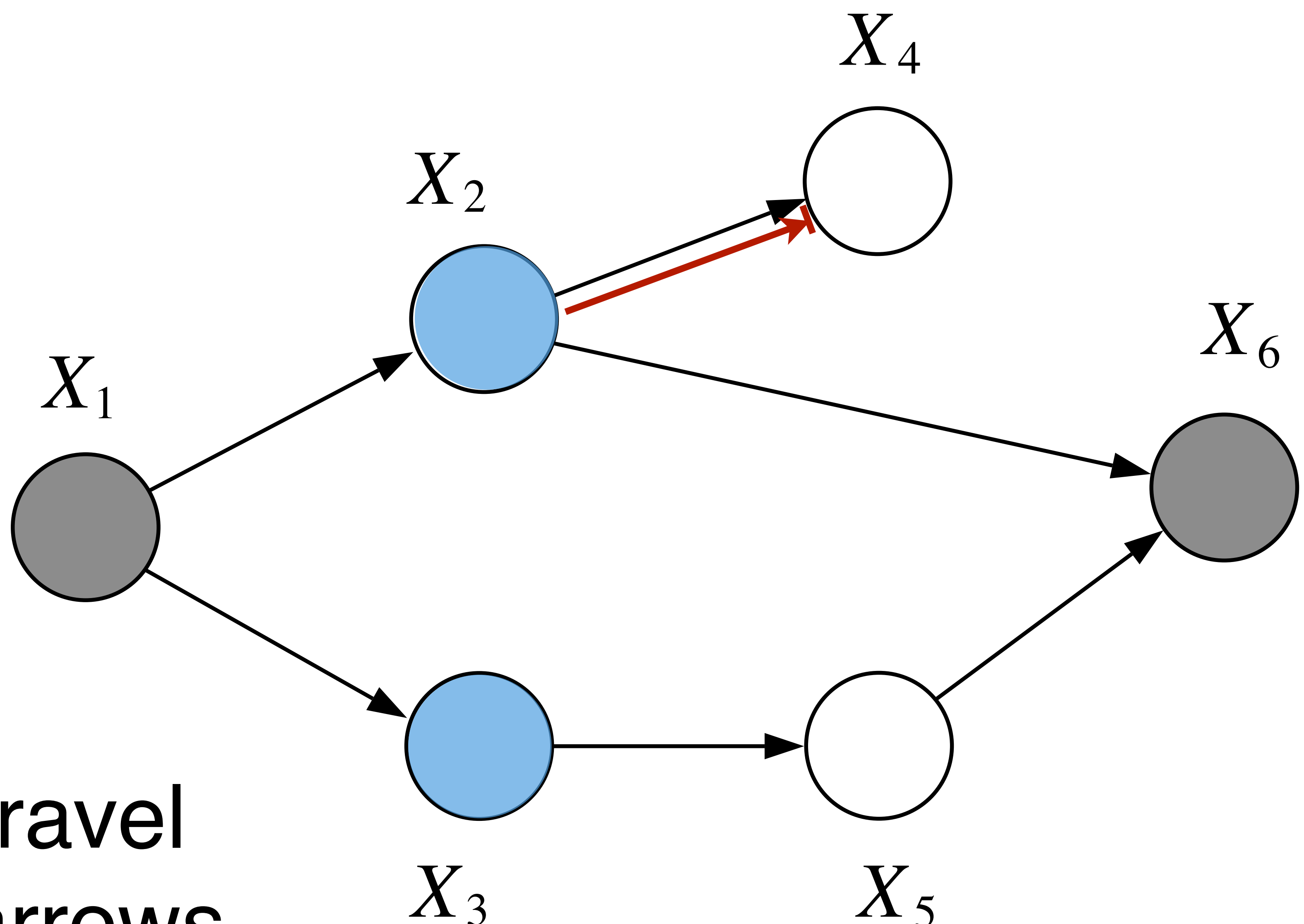
$$\mathbf{x}_2 \perp \mathbf{x}_3 | \{\mathbf{x}_1, \mathbf{x}_6\} \quad ?$$



ball can travel
opposite arrows

Transition rules

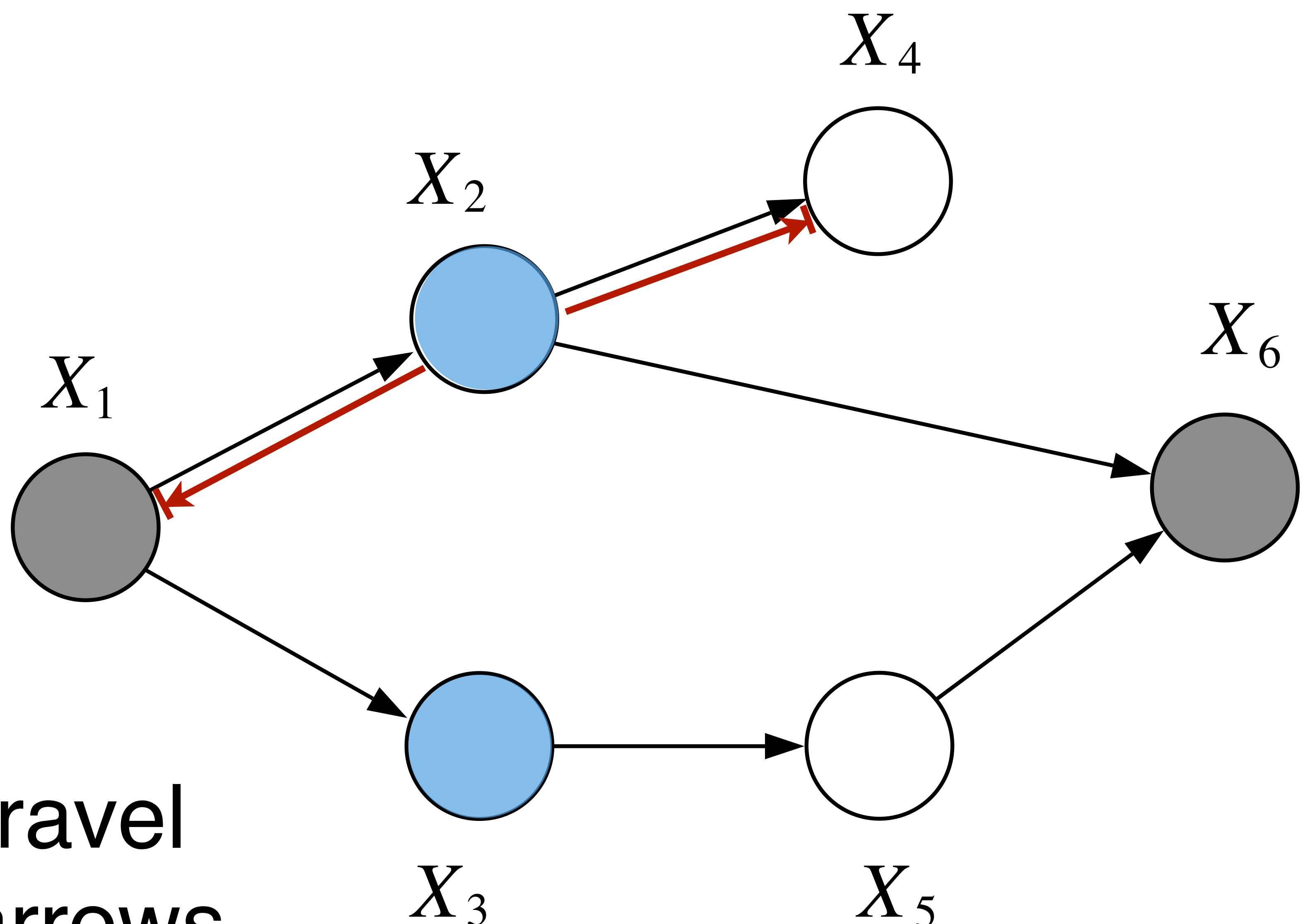
$$\mathbf{x}_2 \perp \mathbf{x}_3 | \{\mathbf{x}_1, \mathbf{x}_6\} \quad ?$$



ball can travel
opposite arrows

Transition rules

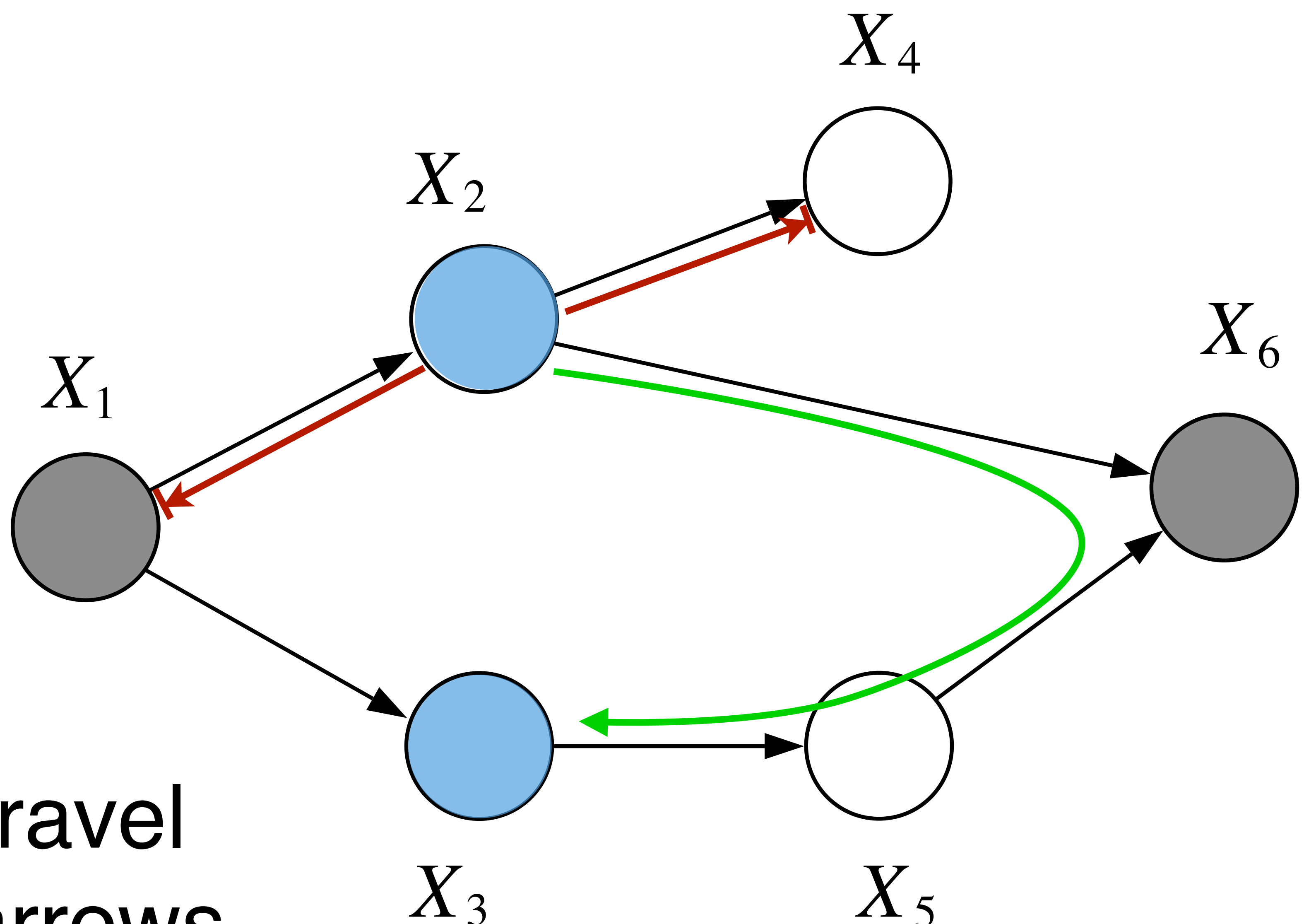
$$\mathbf{x}_2 \perp \mathbf{x}_3 | \{\mathbf{x}_1, \mathbf{x}_6\} \quad ?$$



ball can travel
opposite arrows

Transition rules

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

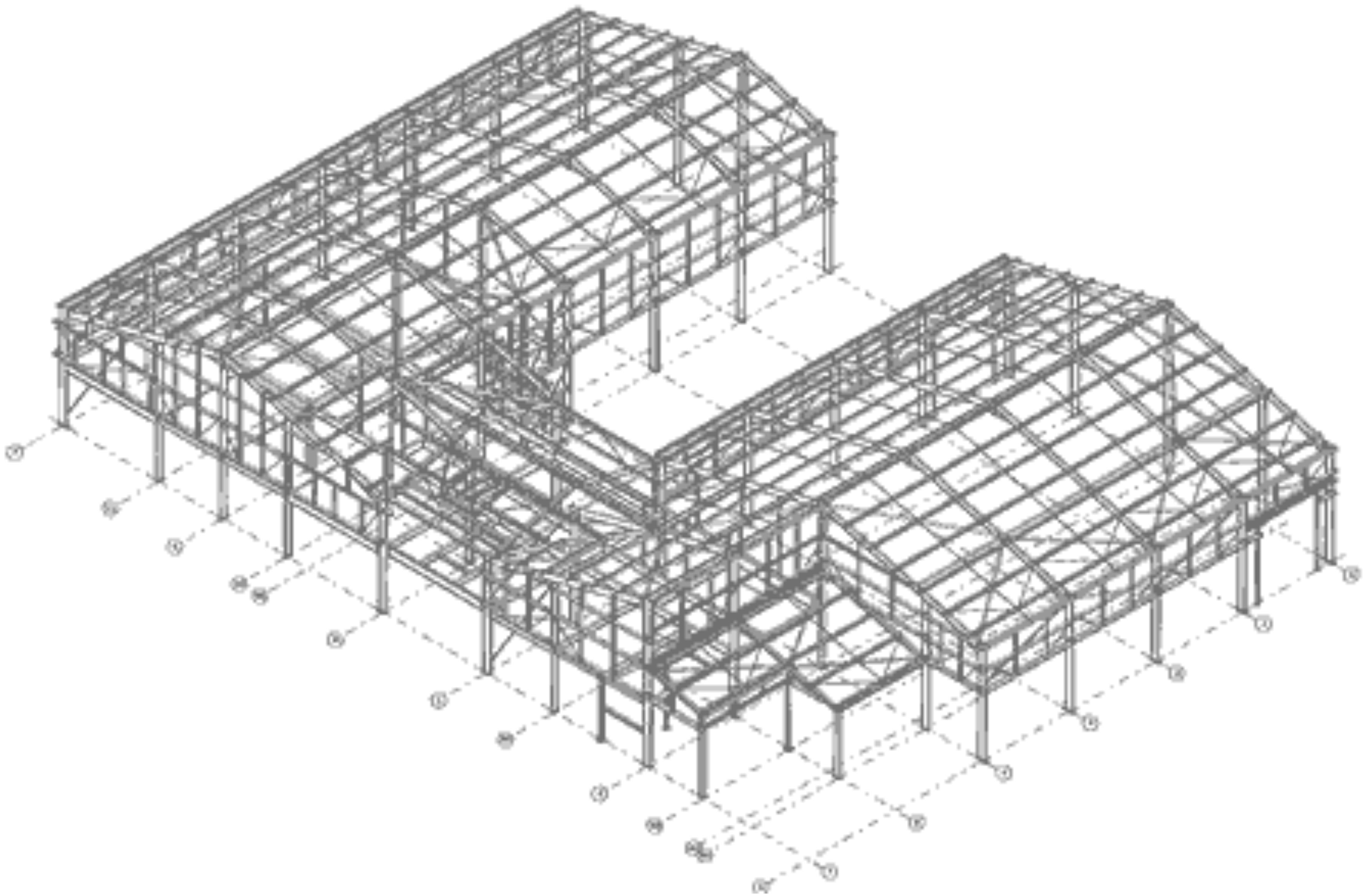


ball can travel
opposite arrows

Summary

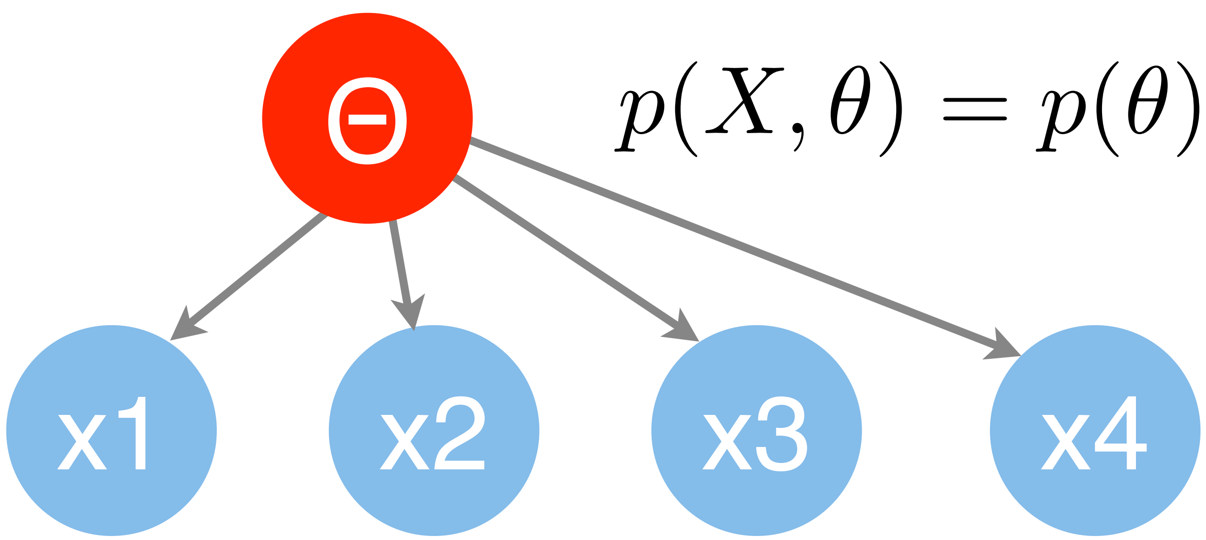
- Dependent random variables
- Observing can make things dependent or independent
- Conditional independence simplifies model
- Bayes ball to check properties
  - Chains (observing stops dependence)
  - Common causes (observing stops dependence)
  - Common children (observing creates dependence)

Structures

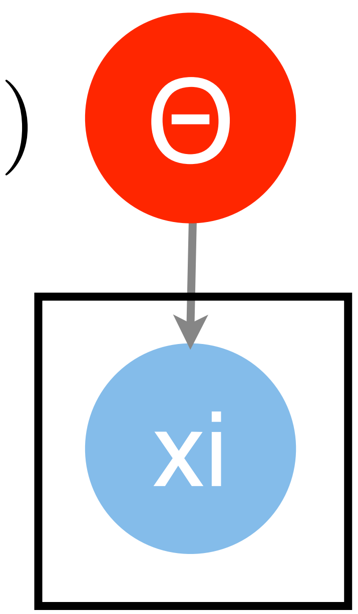


Plates: FOR loops for statisticians

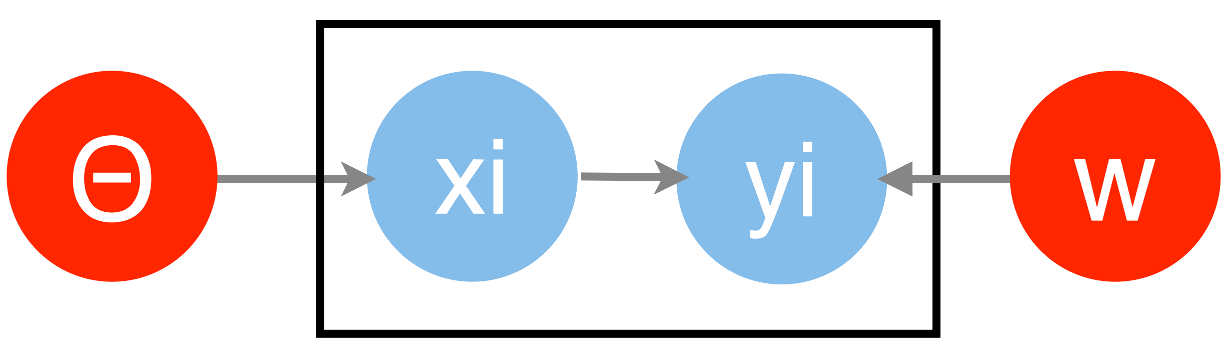
- Repeated dependency structure
  - Modeling iid observations



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



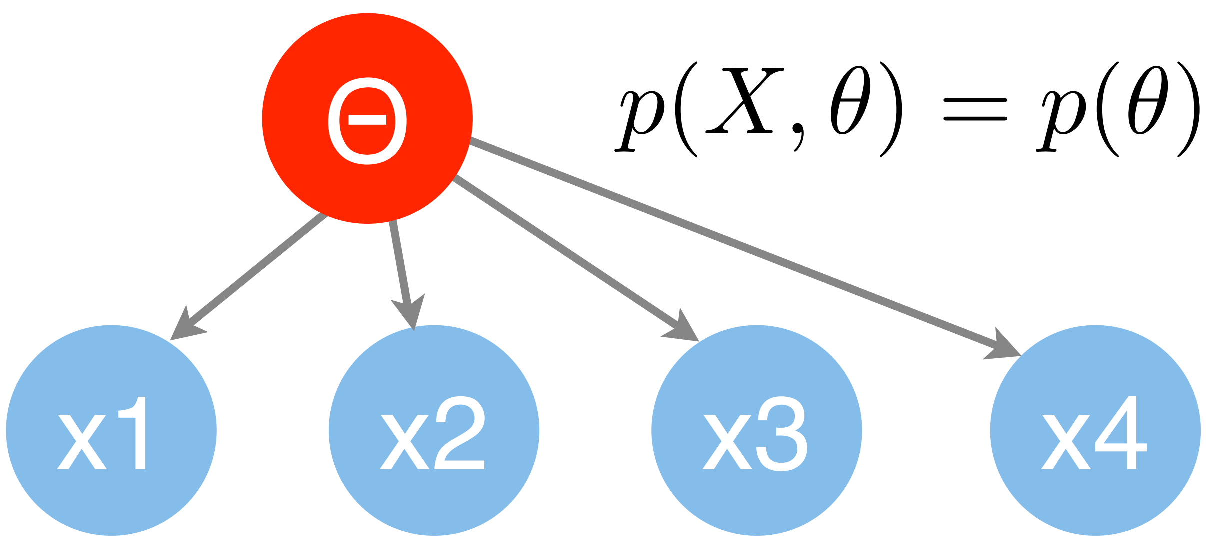
  - Supervised learning



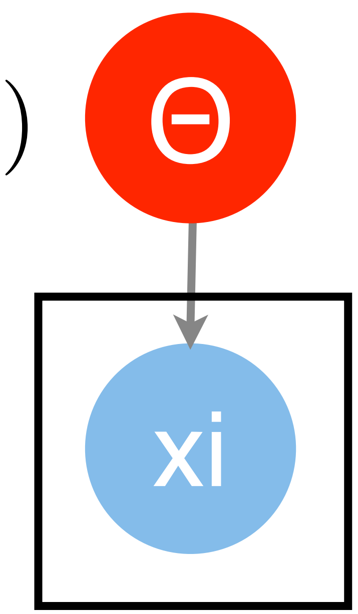
$$p(X, Y, \theta, w)$$
$$= p(\theta) p(w) \prod_i p(x_i | \theta) p(y_i | x_i, w)$$

Plates: FOR loops for statisticians

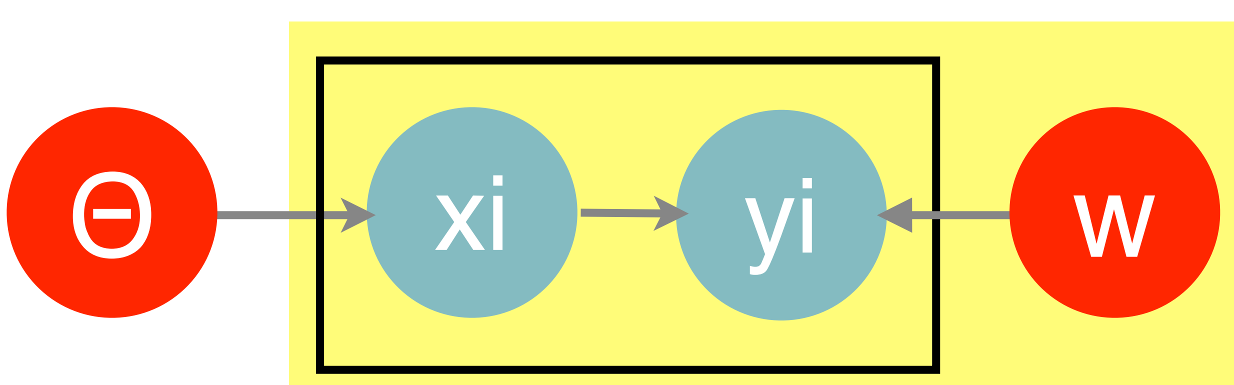
- Repeated dependency structure
  - Modeling iid observations



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



  - Supervised learning

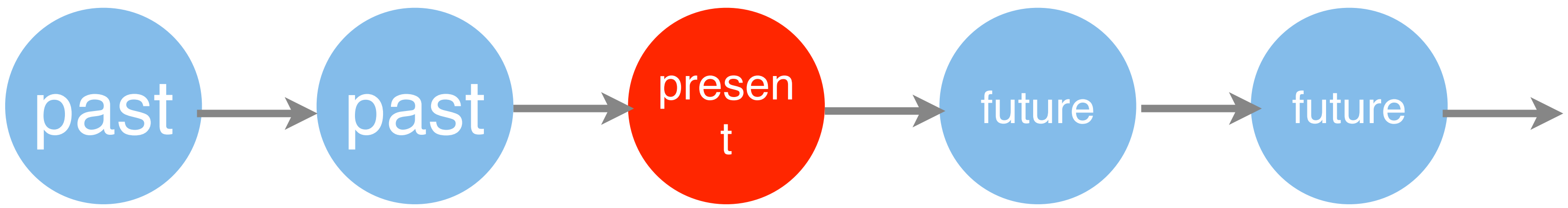


$$p(X, Y, \theta, w)$$

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

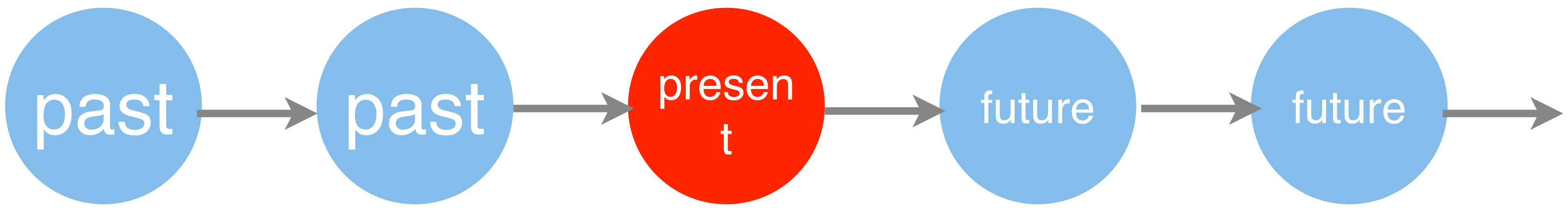
Chains

Markov Chain

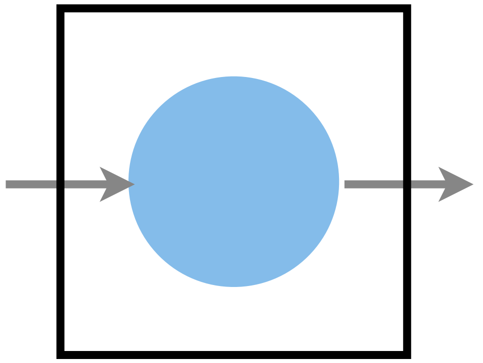


Chains

Markov Chain

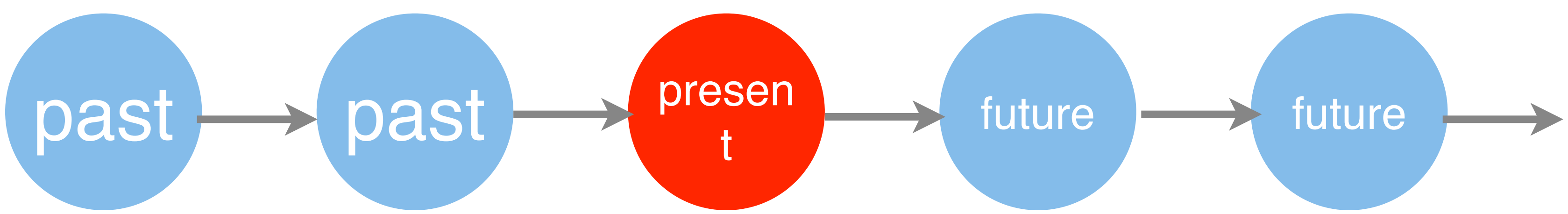


Plate

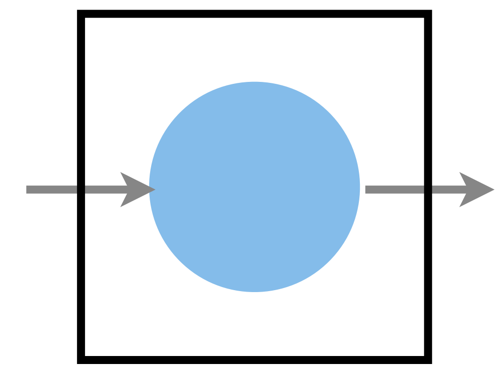


Chains

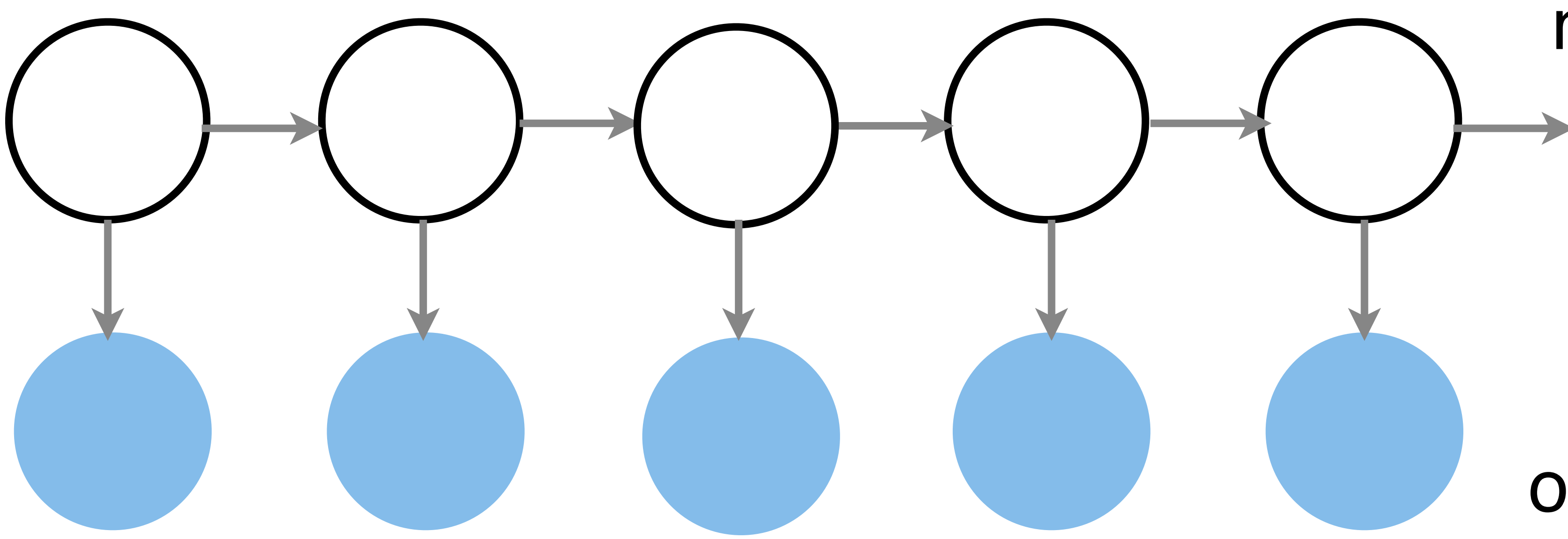
Markov Chain



Plate

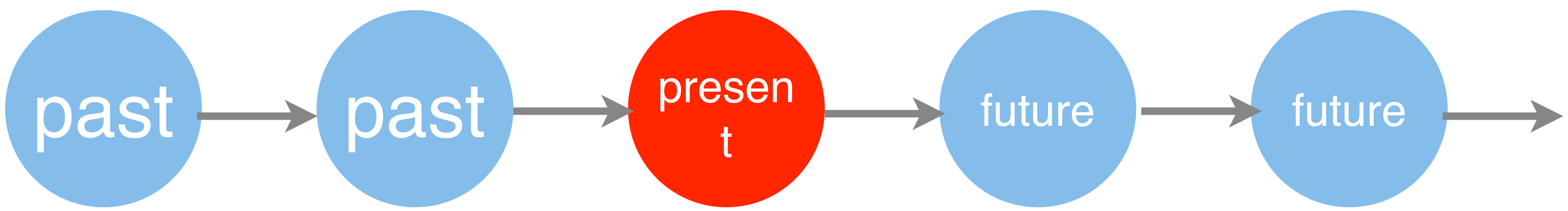


Hidden Markov Chain

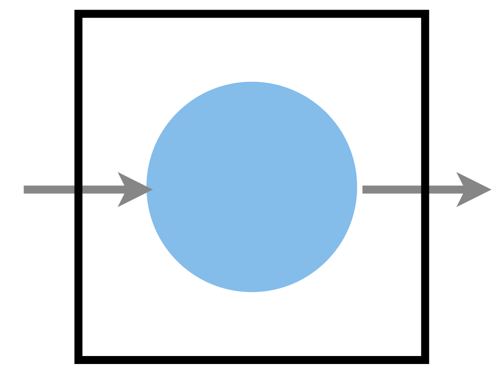


Chains

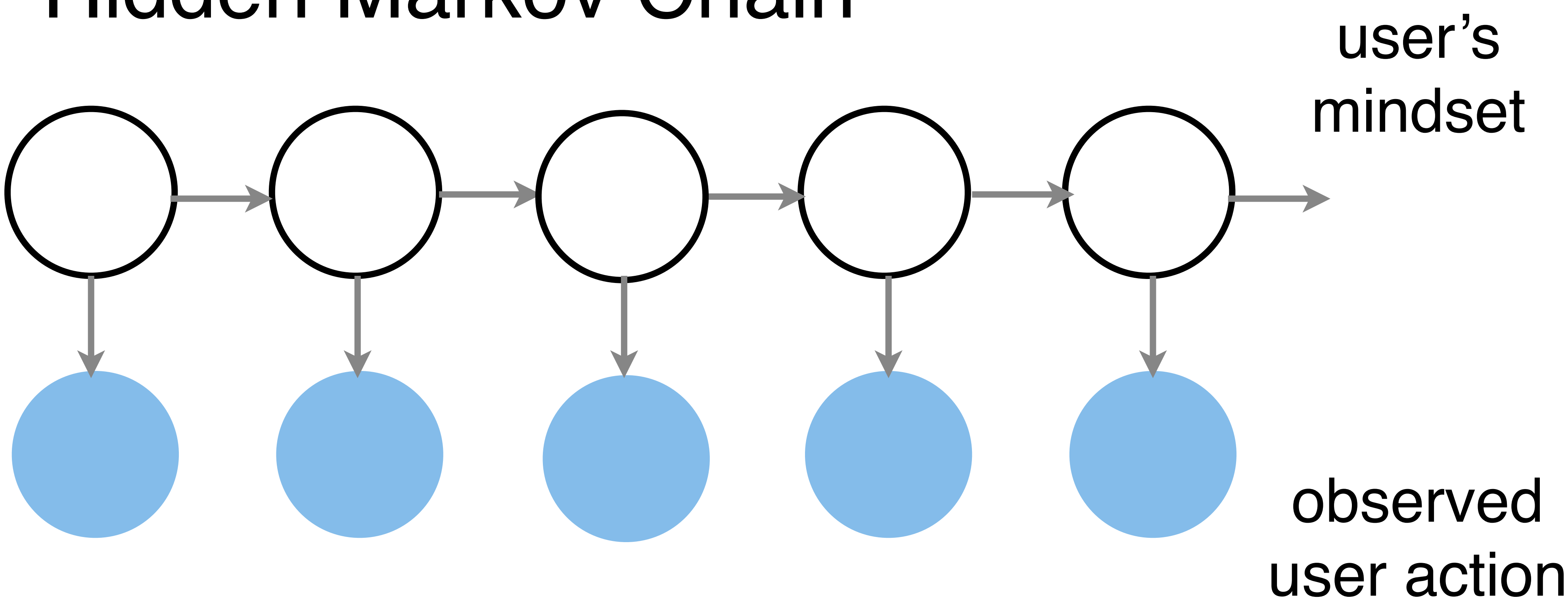
Markov Chain



Plate



Hidden Markov Chain

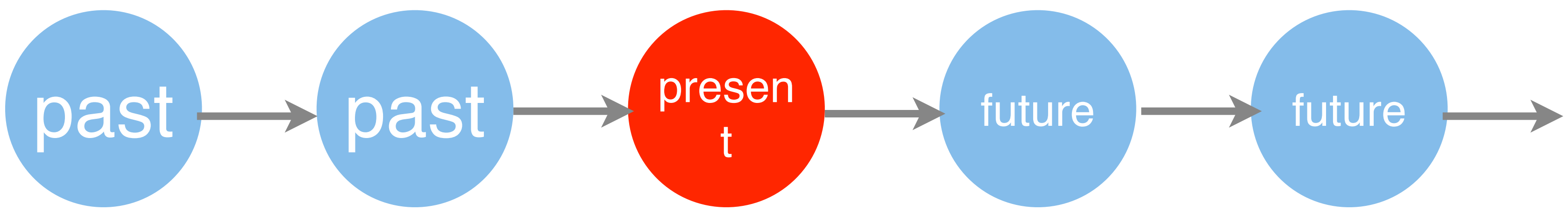


user model for traversal through search results

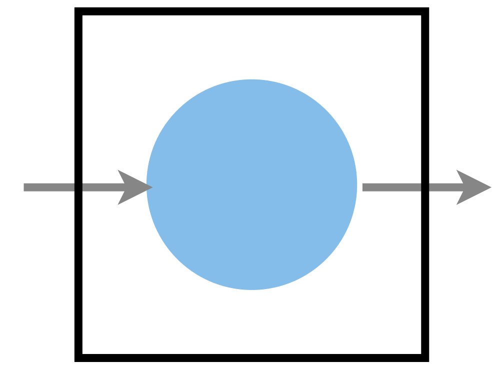
Carnegie Mellon University

Chains

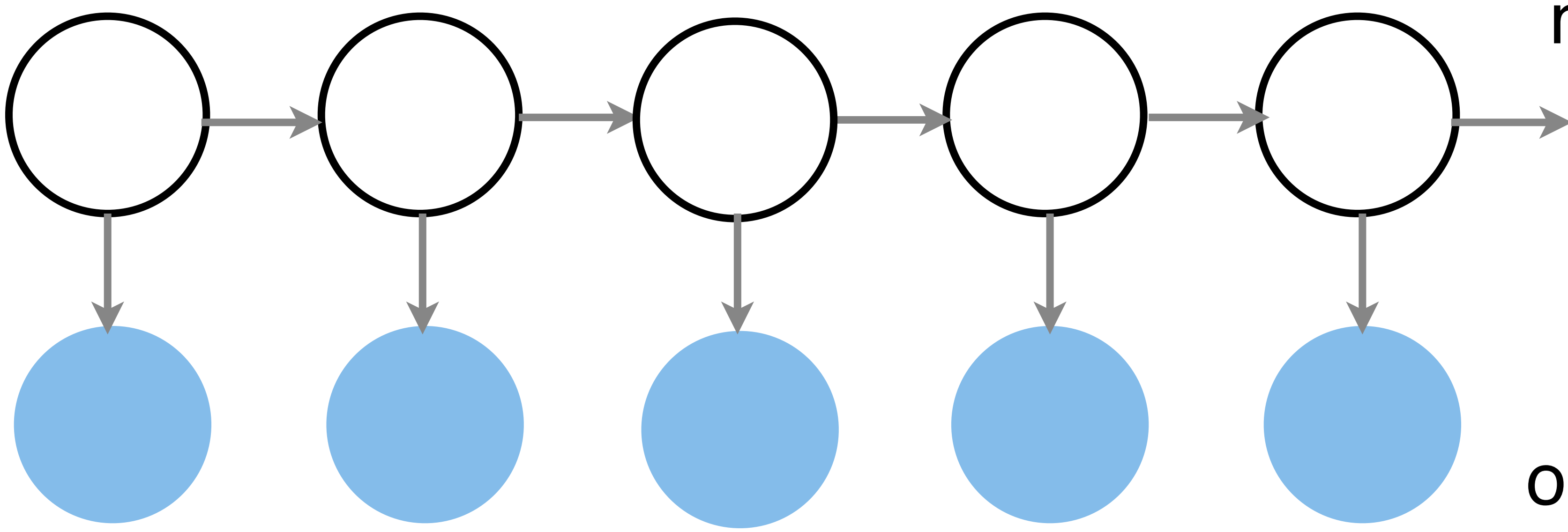
Markov Chain



Plate

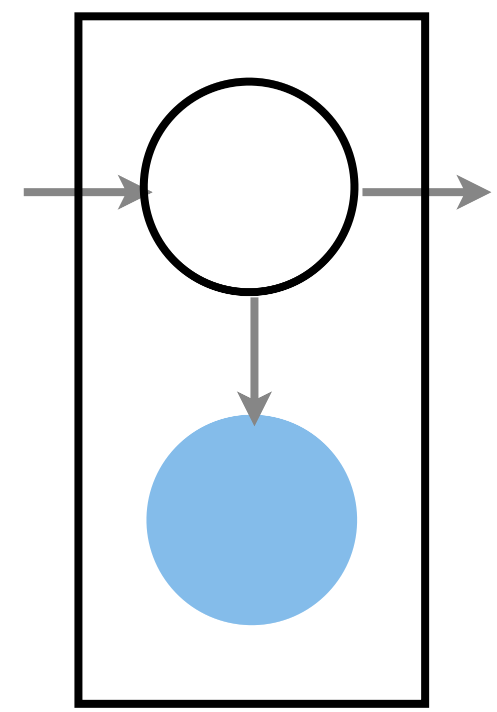


Hidden Markov Chain



user's
mindset

observed
user action



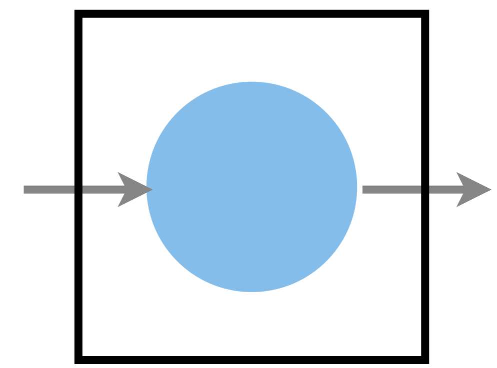
user model for traversal through search results

Chains

Markov Chain

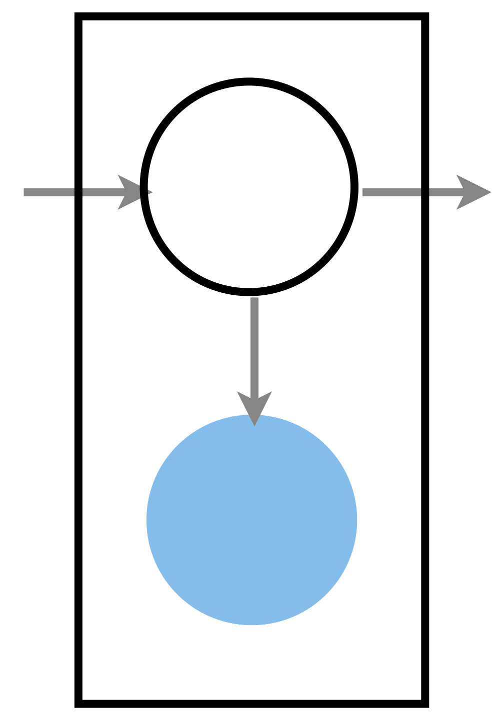
$$p(x;\theta) = p(x_0;\theta)\prod_{i=1}^{n-1} p(x_{i+1}|x_i;\theta)$$

Plate



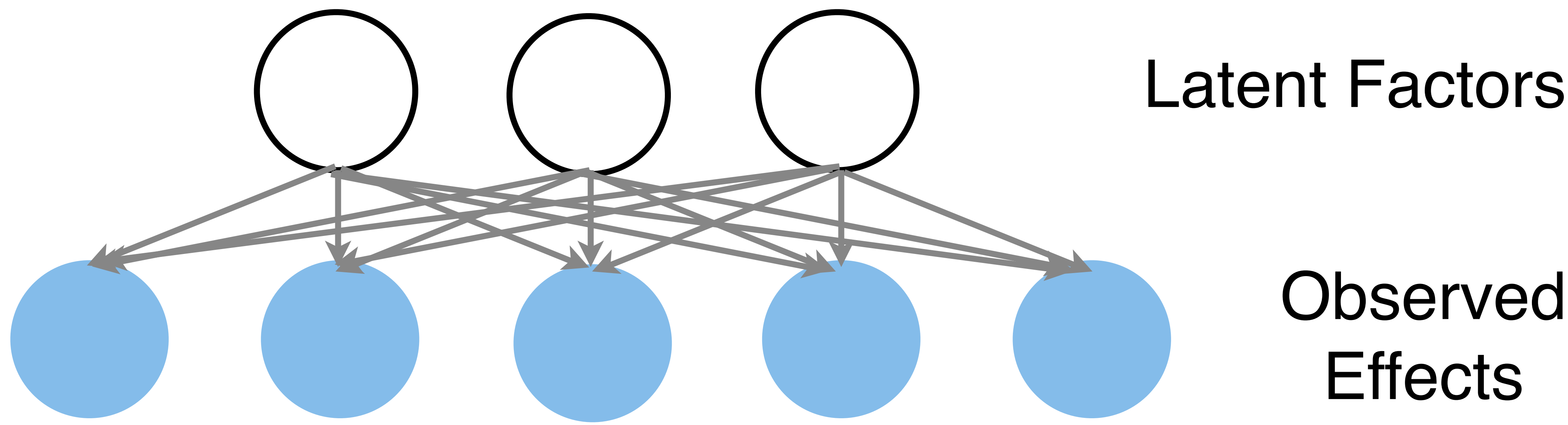
Hidden Markov Chain

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

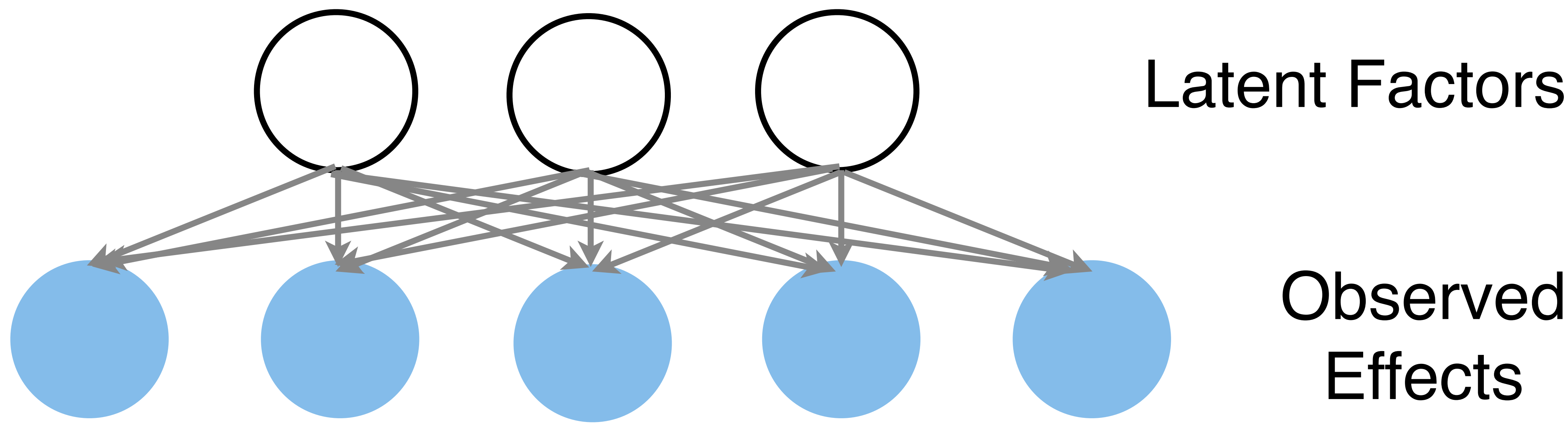


user model for traversal through search results

Factor Graphs

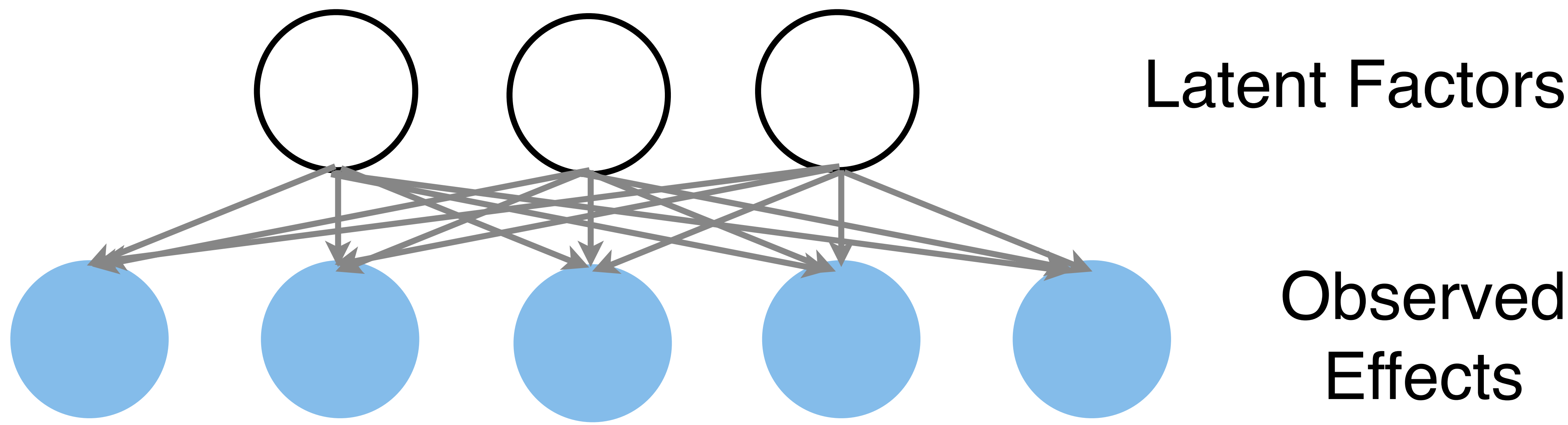


Factor Graphs



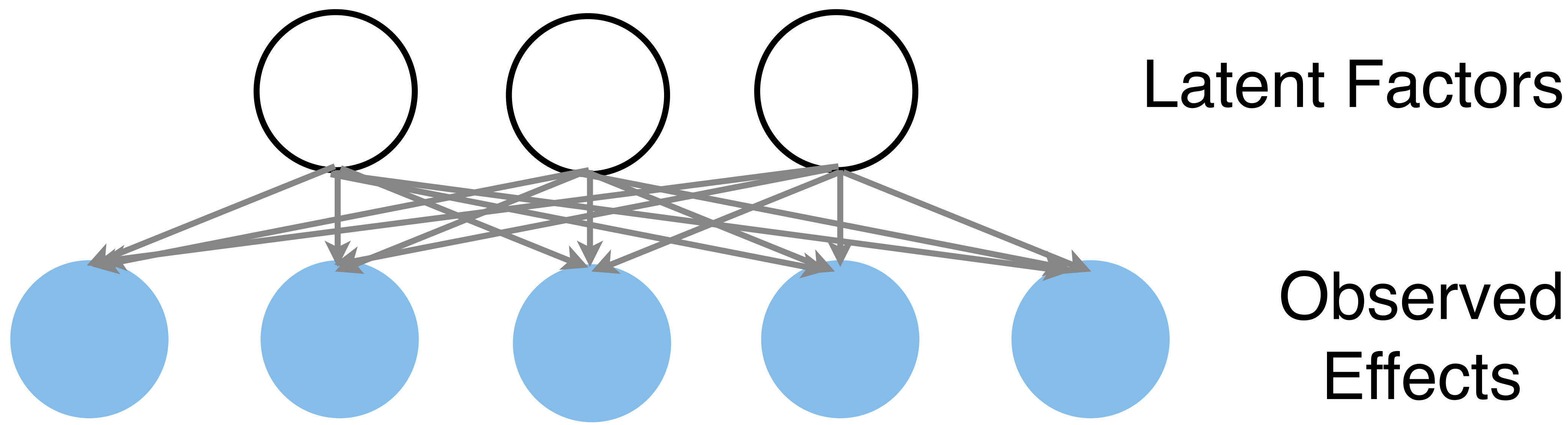
- Observed effects
  Click behavior, queries, watched news, emails

Factor Graphs



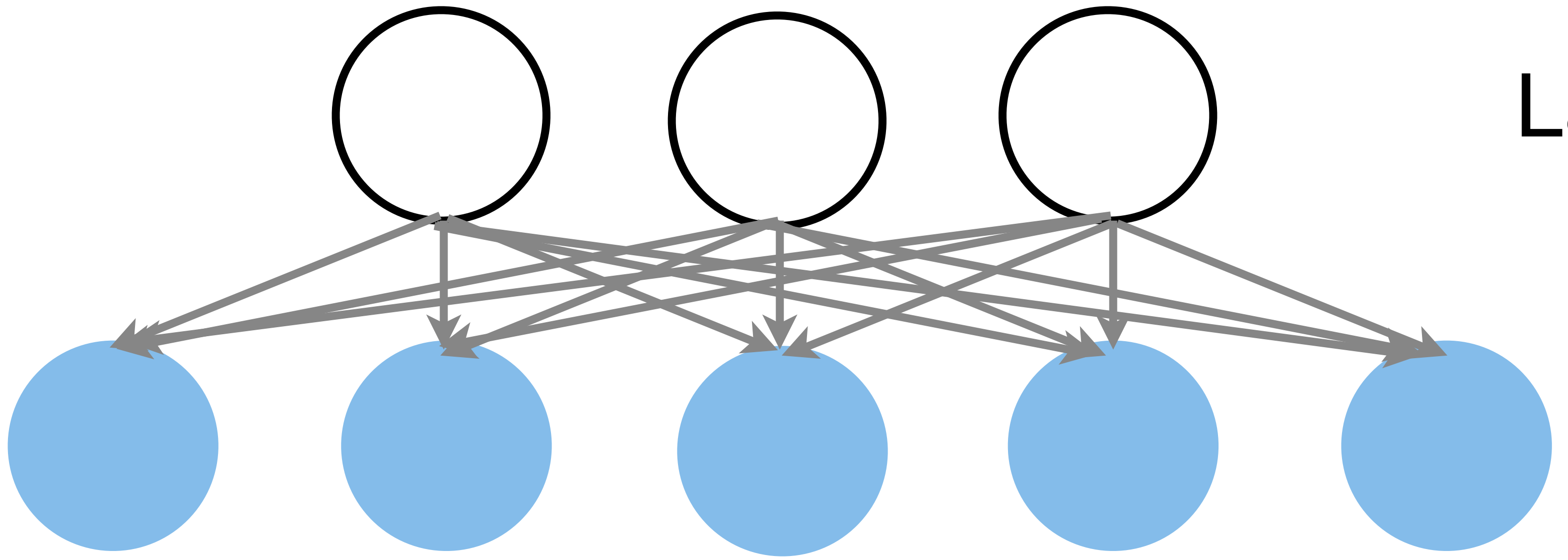
- Observed effects
  Click behavior, queries, watched news, emails
- Latent factors
  User profile, news content, hot keywords, social connectivity graph, events

Example - PCA/ICA



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

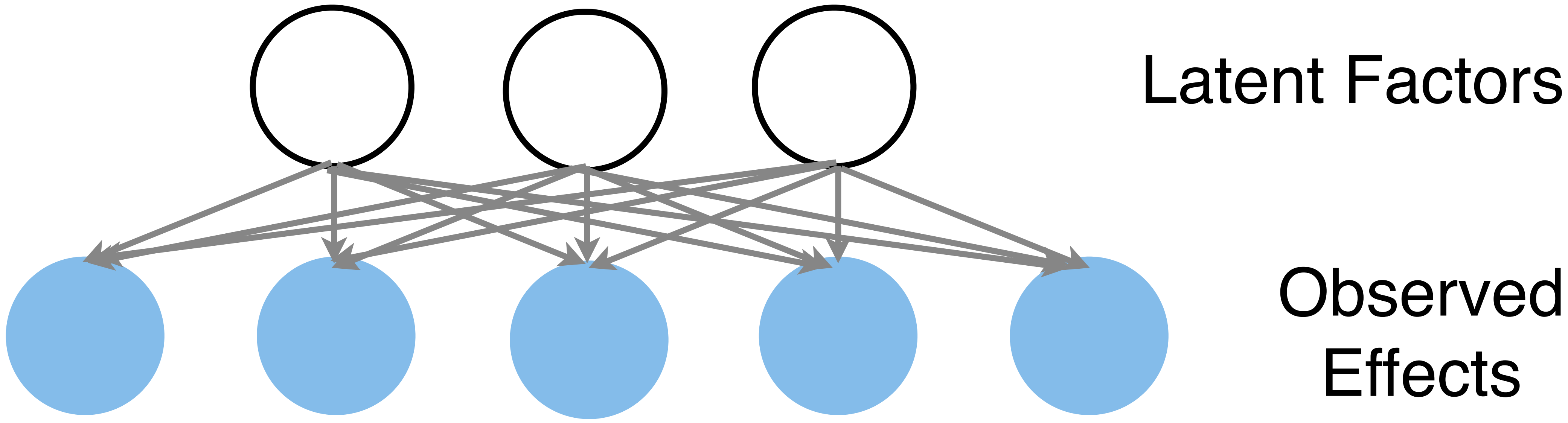
Example - PCA/ICA



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

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

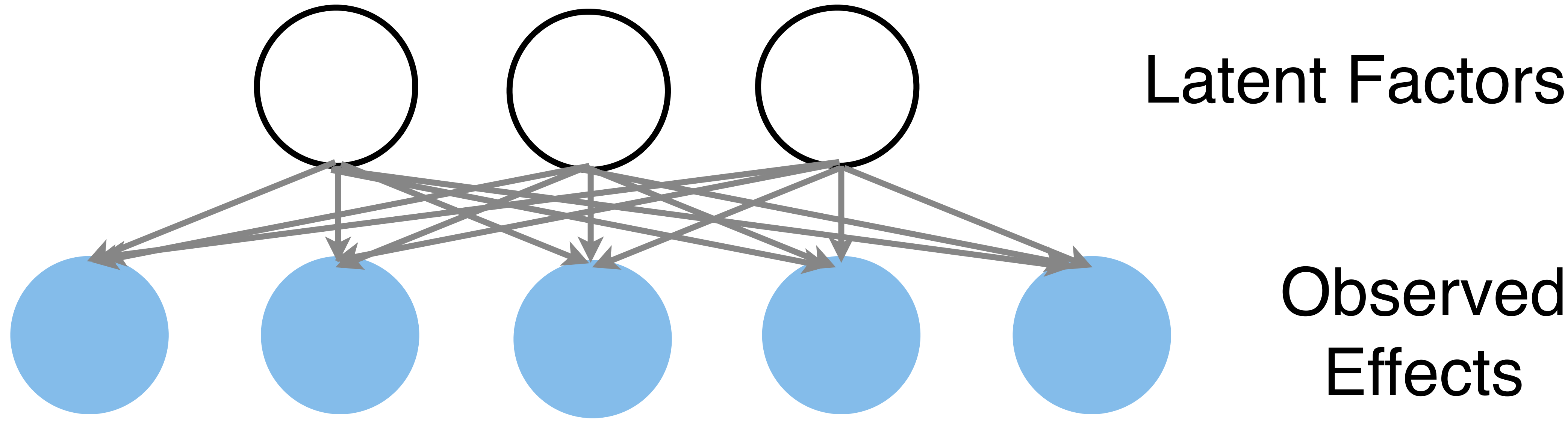
Example - PCA/ICA



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

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

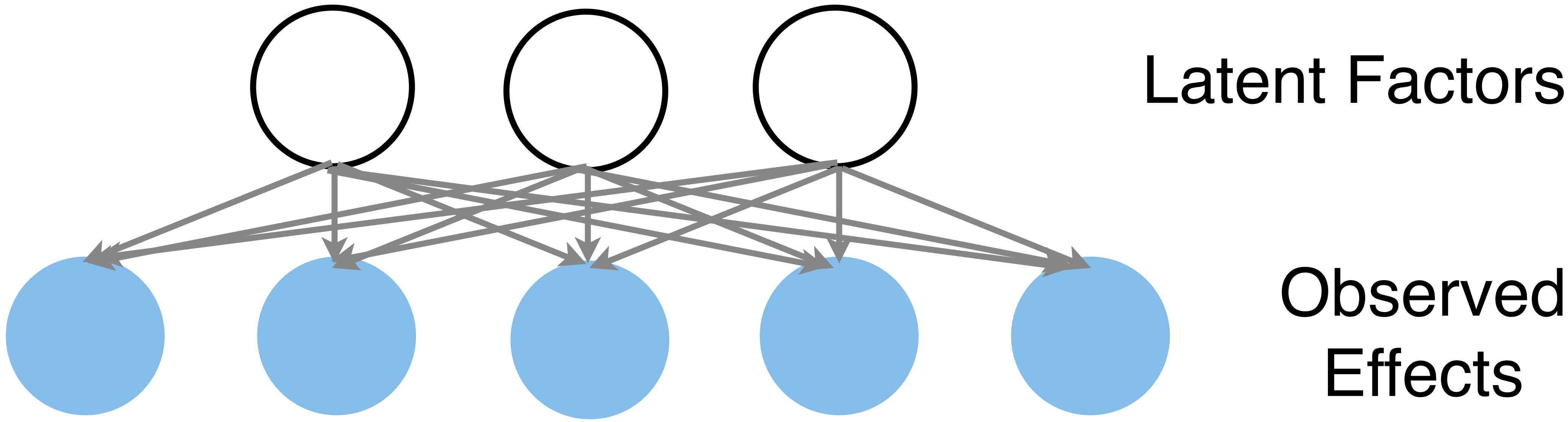
Example - PCA/ICA



- 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



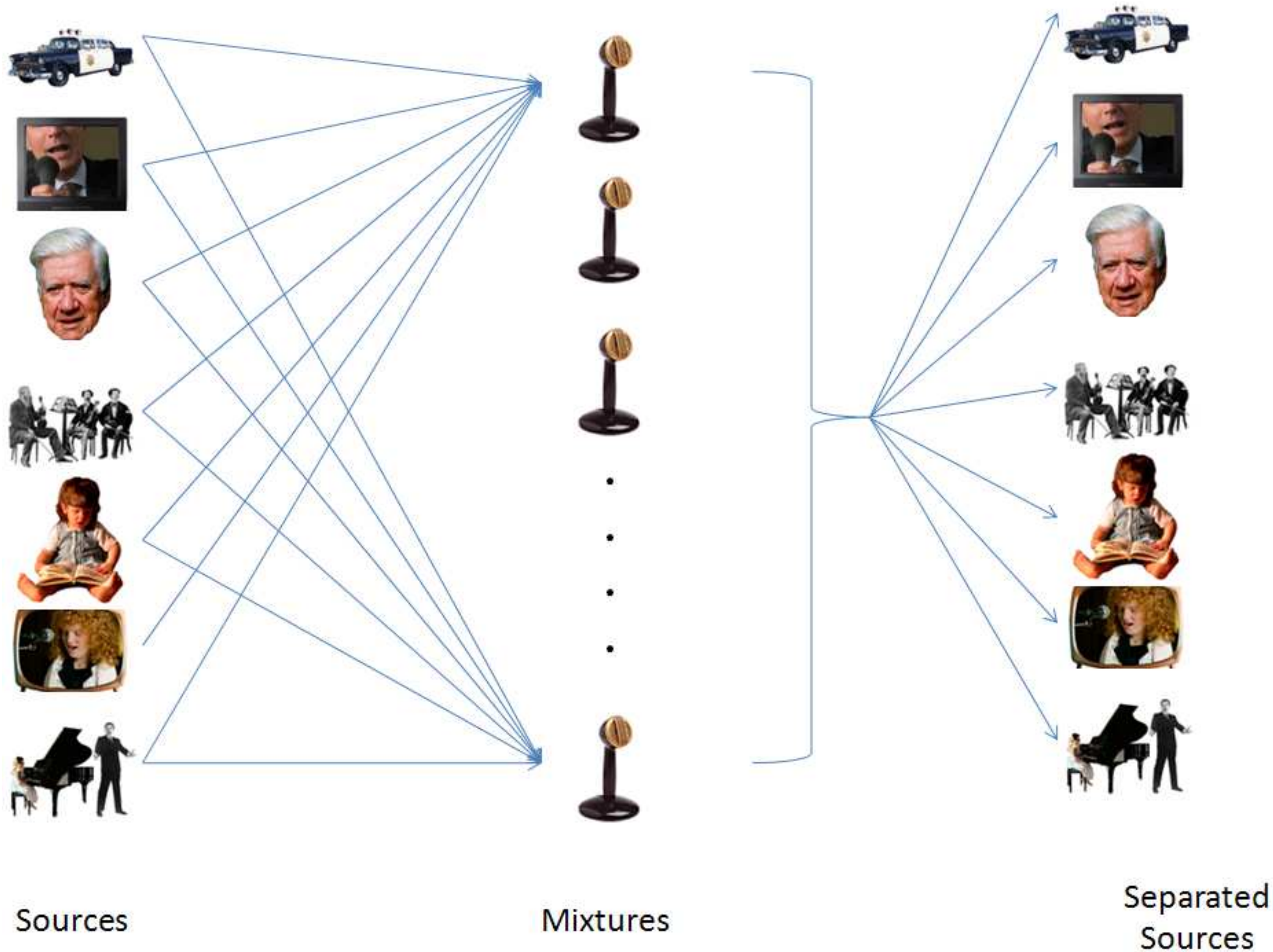
- Observed effects
  Click behavior, queries, watched news, emails

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

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

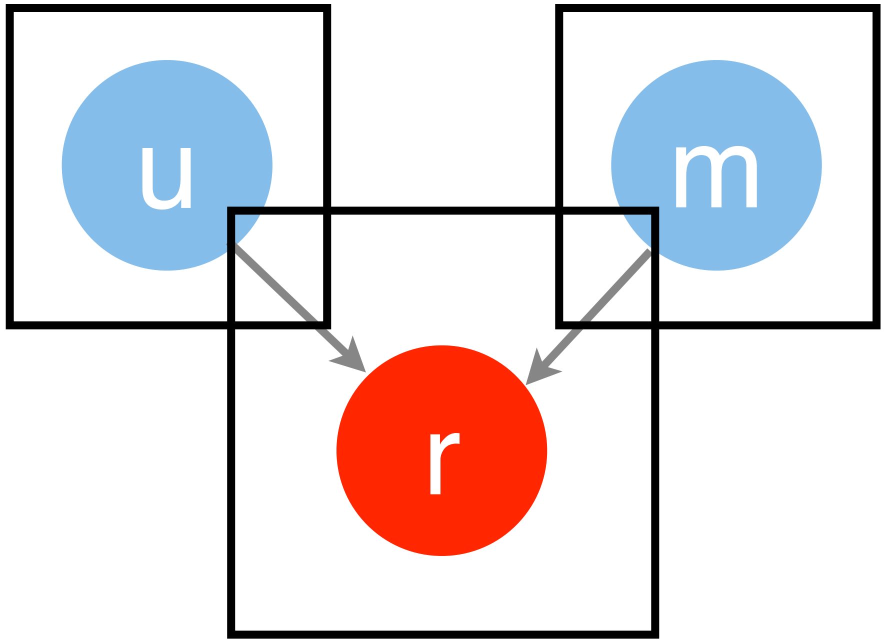
Carnegie Mellon University

Cocktail party problem

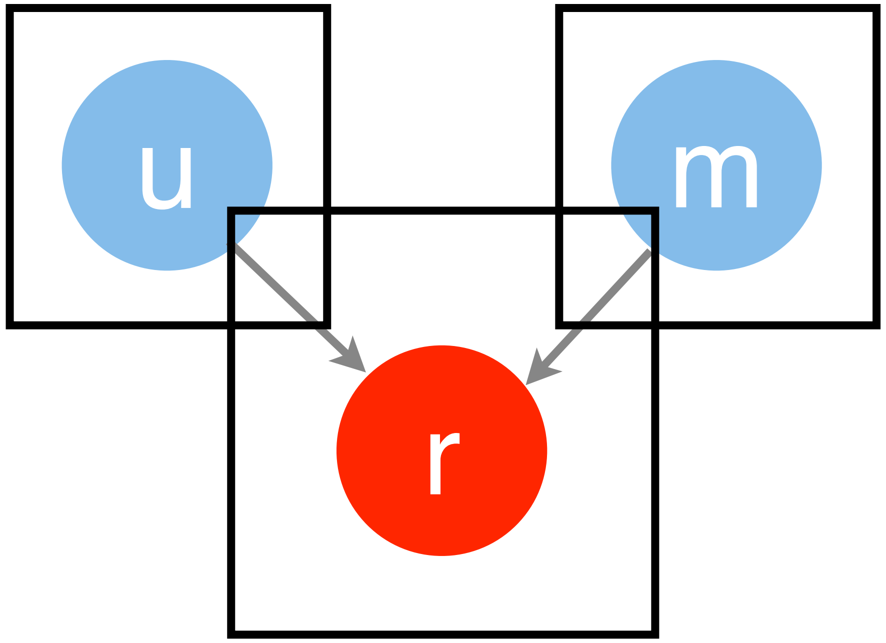


Carnegie Mellon University

Recommender Systems

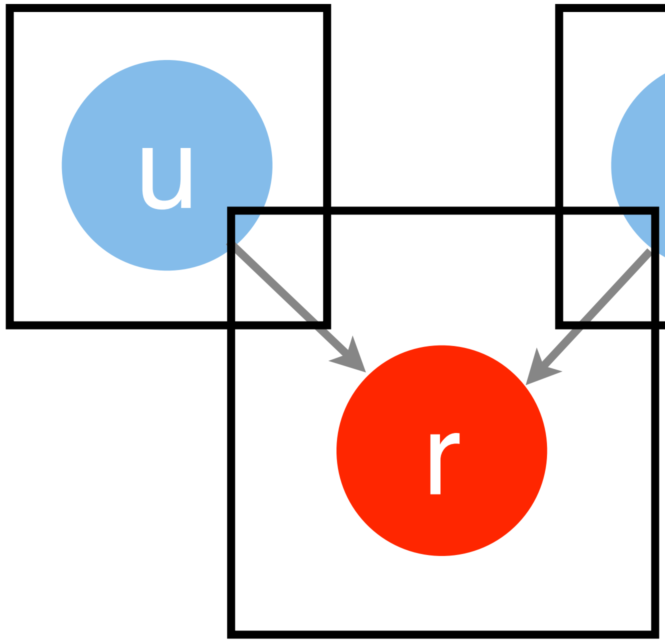


Recommender Systems



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

Recommender Systems

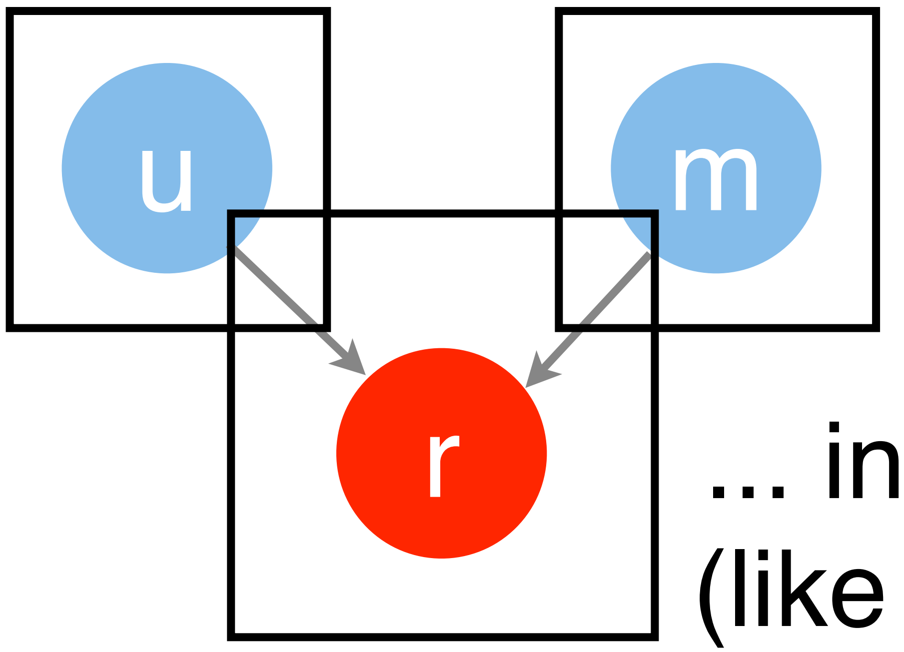


... intersecting plates ...
(like nested FOR loops)

- 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

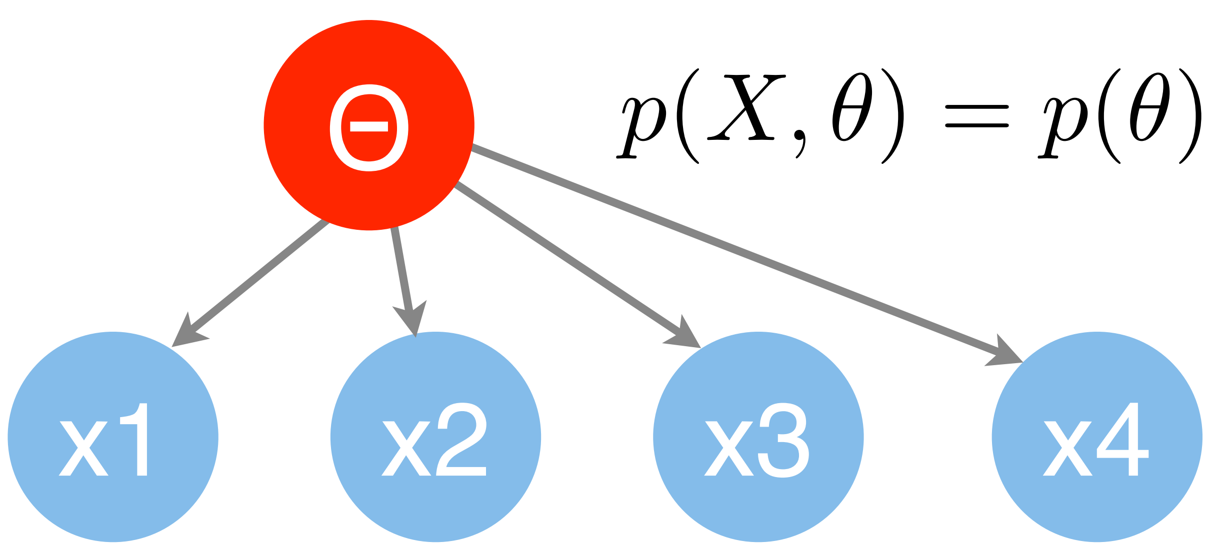
- How to design models
  - Common (engineering) sense
  - Computational tractability
- Dependency analysis
- Inference
  - Easy for fully observed situations
  - Many algorithms if not fully observed
  - Dynamic programming / message passing

engineering

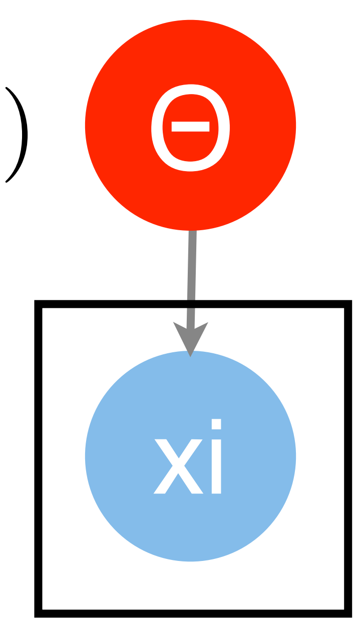
machine learning

Summary

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



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



7.2 Dynamic Programming

7 Graphical Models

Alexander Smola

Introduction to Machine Learning 10-701

http://alex.smola.org/teaching/10-701-15

MACHINE LEARNING DEPARTMENT

Carnegie Mellon University


CHAIN
TREE
WAYAN
The Land of Spicy Hills
DISTRICT TOURISM PROMOTION COUNCIL

Chains

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

$$x0 \rightarrow x1 \rightarrow x2 \rightarrow x3$$

Transition Matrices

x0	
0	0.4
1	0.6

x1 \ x0	0	1
0	0.2	0.1
1	0.8	0.9

x2 \ x1	0	1
0	0.8	0.5
1	0.2	0.5

x3 \ x2	0	1
0	0	1
1	1	0

Unraveling the chain

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

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

Chains

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

$$\text{x0} \to \text{x1} \to \text{x2} \to \text{x3}$$

- Transition matrices

x0	
0	0.4
1	0.6

x1 \ x0	0	1
0	0.2	0.1
1	0.8	0.9

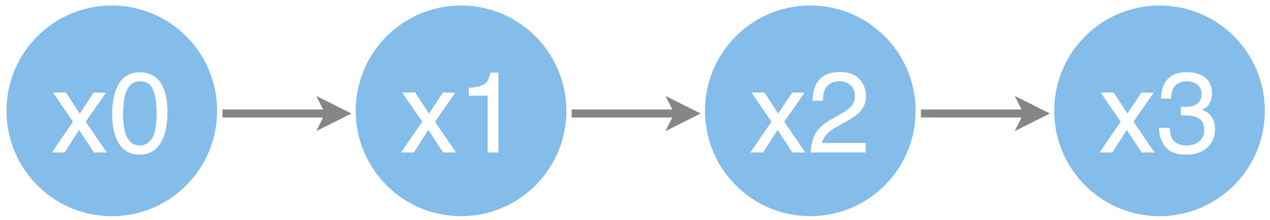
x2 \ x1	0	1
0	0.8	0.5
1	0.2	0.5

x3 \ x2	0	1
0	0	1
1	1	0

```
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]
```

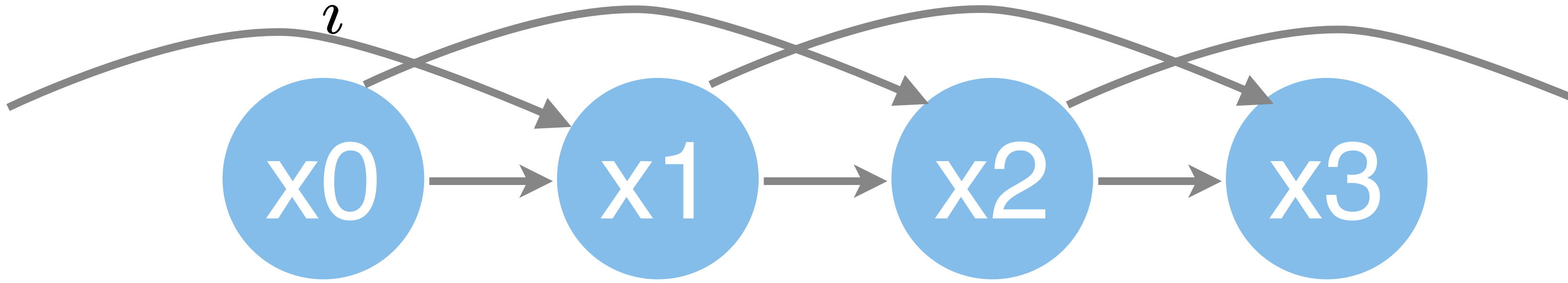
Markov Chains

- First order chain



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

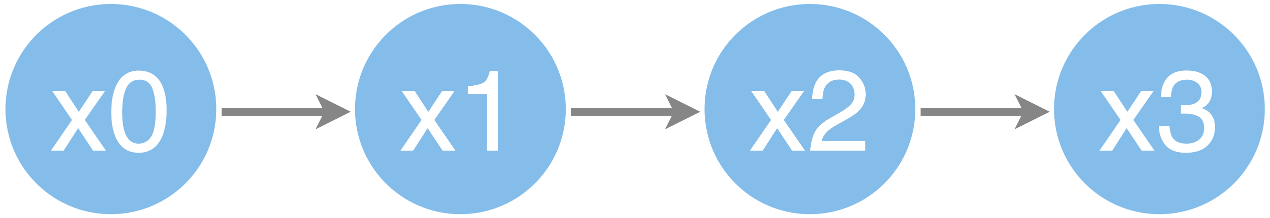
- Second order



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

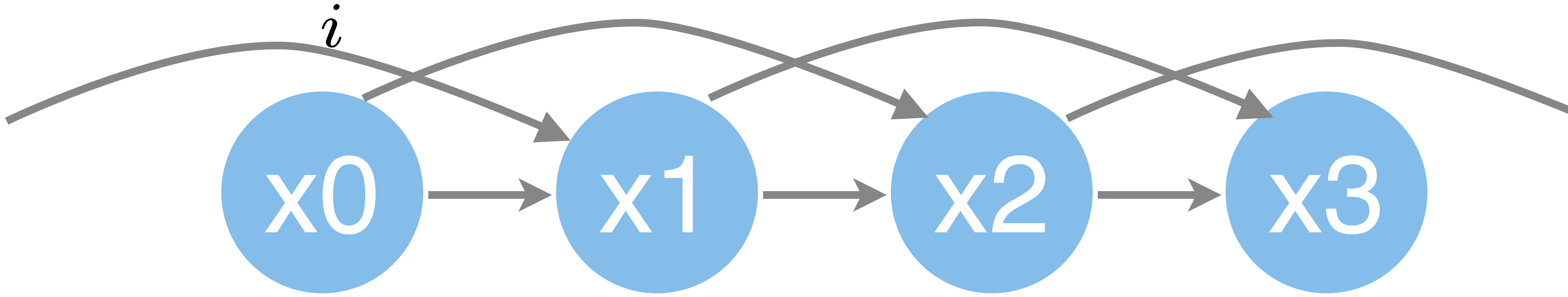
Markov Chains

- 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

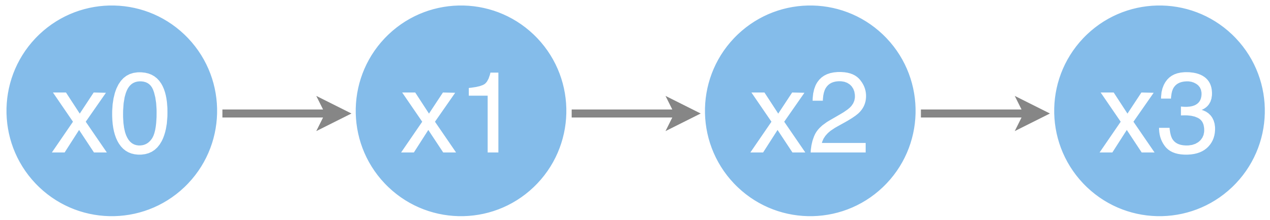
Markov In Chains #MLBandNames

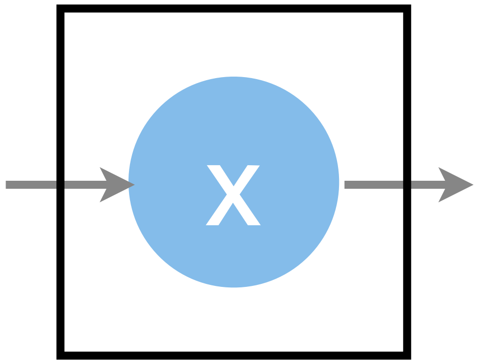
Collapse Reply Retweet Favorite ••• More

9	4
RETWEETS	FAVORITES

Chains

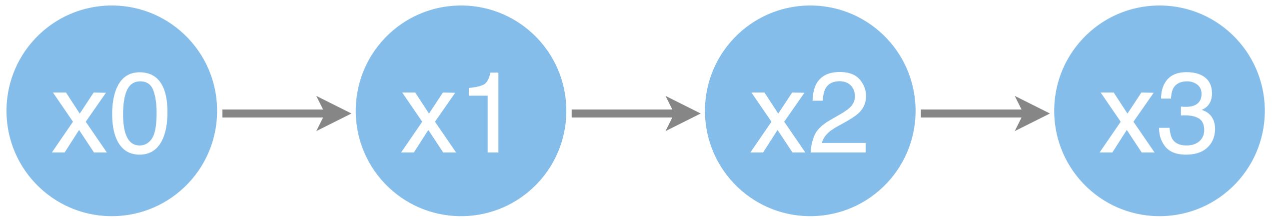
$$p(x;\theta) = p(x_0;\theta)\prod_{i=1}^{n-1} p(x_{i+1}|x_i;\theta)$$



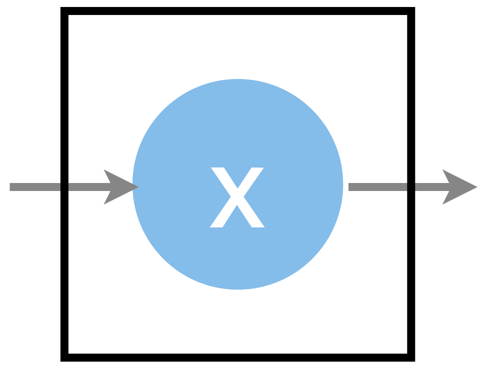


Chains

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

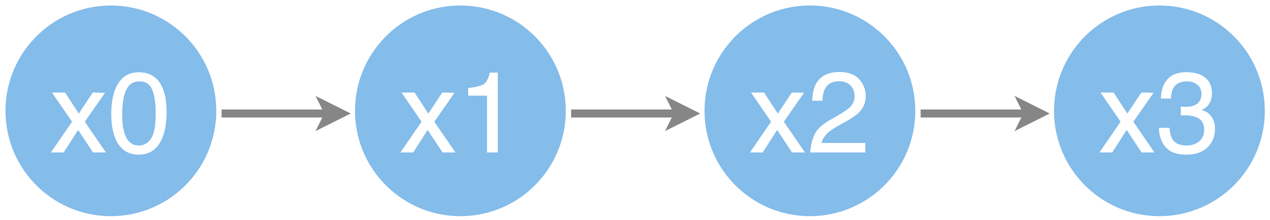


$$p(x_i) = \sum_{x_0,\dots x_{i-1},x_{i+1}\dots x_n} \underbrace{p(x_0)}_{:=l_0(x_0)} \prod_{j=1}^{n} p(x_j|x_{j-1})$$



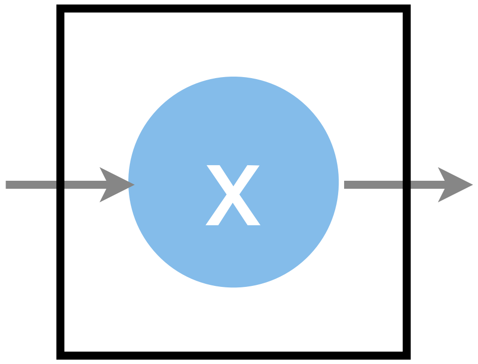
Chains

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



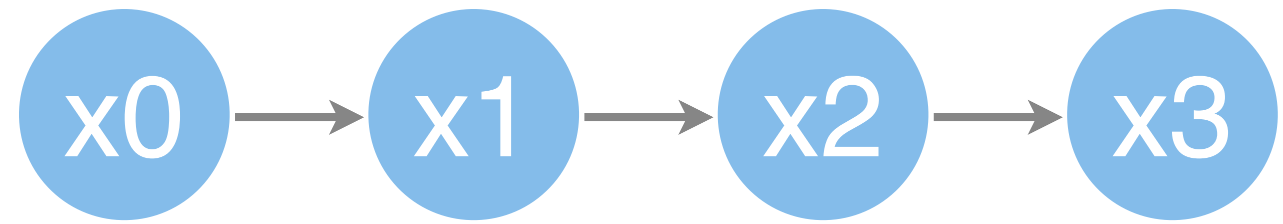
$$p(x_i) = \sum_{x_0,\dots x_{i-1},x_{i+1}\dots x_n} \underbrace{p(x_0)}_{:=l_0(x_0)} \prod_{j=1}^{n} p(x_j|x_{j-1})$$

$$= \sum_{x_1,\dots x_{i-1},x_{i+1}\dots x_n} \underbrace{\sum_{x_0}[l_0(x_0)p(x_1|x_0)]}_{:=l_1(x_1)} \prod_{j=2}^{n} p(x_j|x_{j-1})$$



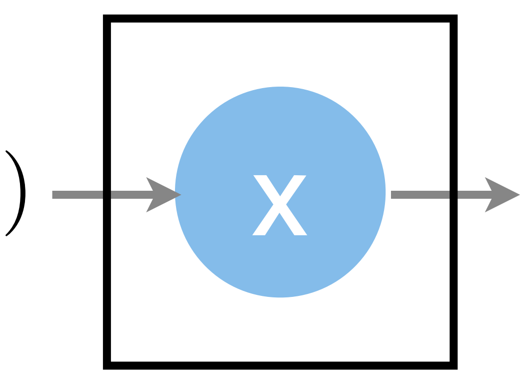
Chains

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



$$p(x_i) = \sum_{x_0,\ldots x_{i-1},x_{i+1}\ldots x_n} \underbrace{p(x_0)}_{:=l_0(x_0)} \prod_{j=1}^{n} p(x_j|x_{j-1})$$

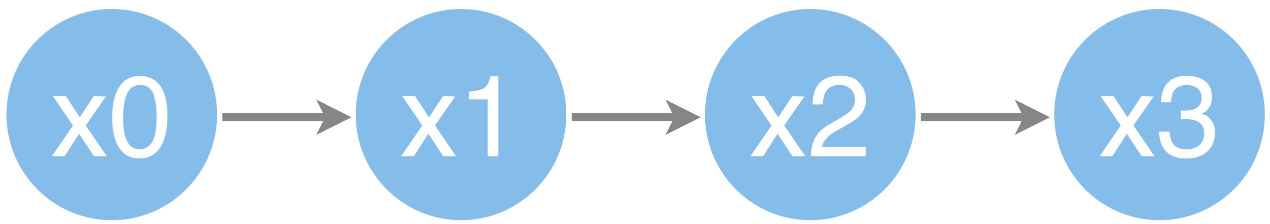
$$= \sum_{x_1,\ldots x_{i-1},x_{i+1}\ldots x_n} \underbrace{\sum_{x_0}\left[l_0(x_0)p(x_1|x_0)\right]}_{:=l_1(x_1)} \prod_{j=2}^{n} p(x_j|x_{j-1})$$



$$= \sum_{x_2,\ldots x_{i-1},x_{i+1}\ldots x_n} \underbrace{\sum_{x_1}\left[l_1(x_1)p(x_2|x_1)\right]}_{:=l_2(x_2)} \prod_{j=3}^{n} p(x_j|x_{j-1})$$

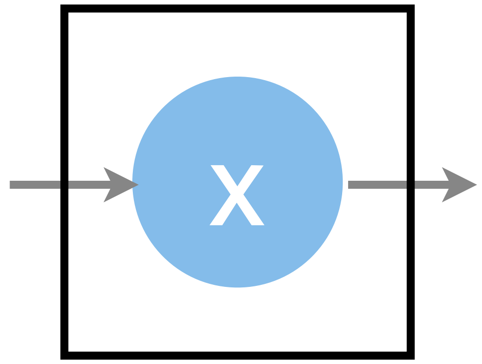
Chains

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



$$p(x_i) = l_i(x_i)\sum_{x_{i+1}\ldots x_n}\prod_{j=i}^{n-1} p(x_{j+1}|x_j)$$

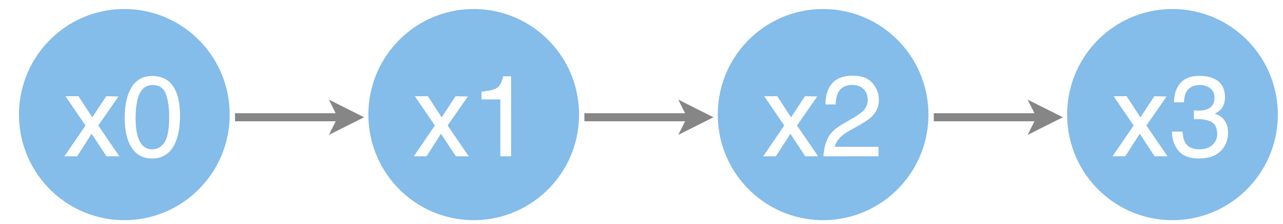
$$= l_i(x_i)\sum_{x_{i+1}\ldots x_{n-1}}\prod_{j=i}^{n-2} p(x_{j+1}|x_j)\underbrace{\sum_{x_n} p(x_n|x_{n-1})}_{:=r_{n-1}(x_{n-1})}$$



$$= l_i(x_i)\sum_{x_{i+1}\ldots x_{n-2}}\prod_{j=i}^{n-3} p(x_{j+1}|x_j)\underbrace{\sum_{x_{n-1}} p(x_{n-1}|x_{n-2})r_{n-1}(x_{n-1})}_{:=r_{n-2}(x_{n-2})}$$

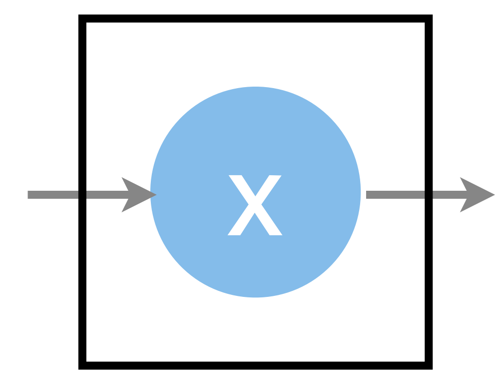
Chains

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



$$p(x_i) = l_i(x_i) \sum_{x_{i+1}\ldots x_n} \prod_{j=i}^{n-1} p(x_{j+1}|x_j)$$

$$= l_i(x_i) \sum_{x_{i+1}\ldots x_{n-1}} \prod_{j=i}^{n-2} p(x_{j+1}|x_j) \underbrace{\sum_{x_n} p(x_n|x_{n-1})}_{:=r_{n-1}(x_{n-1})}$$

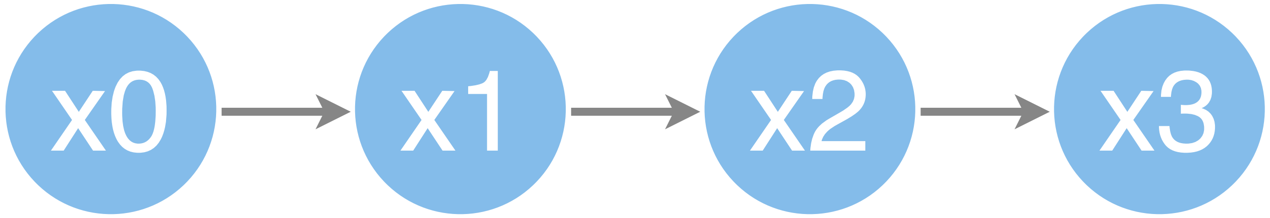


$$= l_i(x_i) \sum_{x_{i+1}\ldots x_{n-2}} \prod_{j=i}^{n-3} p(x_{j+1}|x_j) \underbrace{\sum_{x_{n-1}} p(x_{n-1}|x_{n-2}) r_{n-1}(x_{n-1})}_{:=r_{n-2}(x_{n-2})}$$

$$\pi_i = \prod_{j=1}^{i} \Pi_{j-1\to j}\pi_0$$

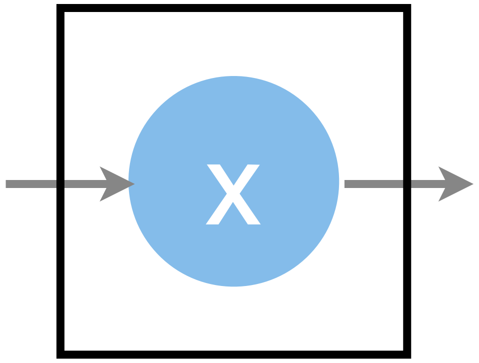
Chains

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



$$p(x_i) = l_i(x \ldots$$

$$= l_i(x \ldots$$

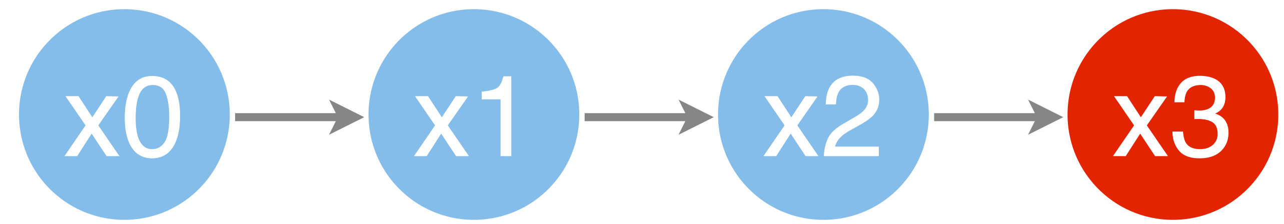


$$= l_i(x \ldots \sum_{x_{i+1}\ldots x_{n-2}} \prod_{j=i} \ldots \underbrace{\sum_{x_{n-1}} \ldots {}_{-1}(x_{n-1})}_{:=r_{n-2}(x_{n-2})}$$

not needed for directed graphs
that are already normalized
... but good to know ...

$$\pi_i = \prod_{j=1}^{i} \Pi_{j-1\to j}\pi_0$$

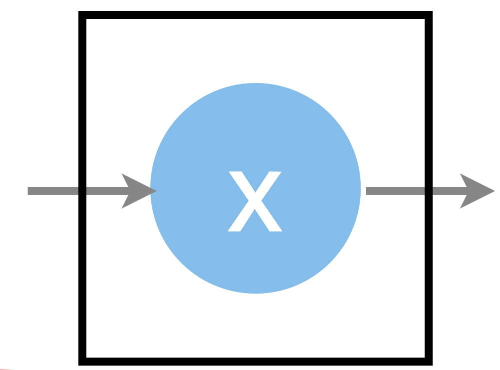
Chains



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

$$p(x_i|x_n) = l_i(x_i)\sum_{x_{i+1}\dots x_{n-1}}\prod_{j=i}^{n-1} p(x_{j+1}|x_j)$$

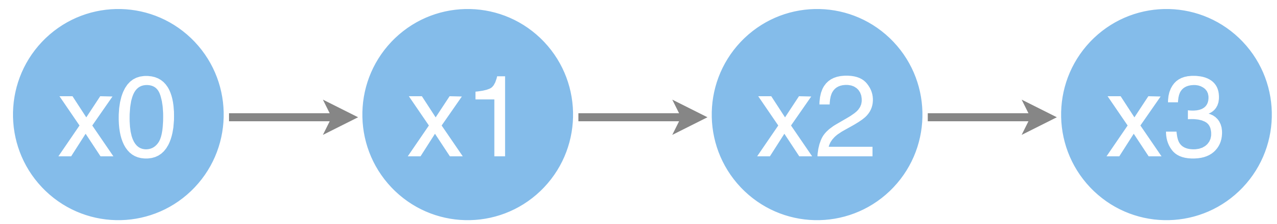
$$= l_i(x_i)\sum_{x_{i+1}\dots x_{n-1}}\prod_{j=i}^{n-2} p(x_{j+1}|x_j)\underbrace{p(x_n|x_{n-1})}_{:=r_{n-1}(x_{n-1})}$$



$$= l_i(x_i)\sum_{x_{i+1}\dots x_{n-2}}\prod_{j=i}^{n-3} p(x_{j+1}|x_j)\underbrace{\sum_{x_{n-1}} p(x_{n-1}|x_{n-2})\, r_{n-1}(x_{n-1})}_{:=r_{n-2}(x_{n-2})}$$

Chains

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



- Forward recursion

$$l_0(x_0) := p(x_0) \text{ and } l_i(x_i) := \sum_{x_{i-1}} l_{i-1}(x_{i-1})p(x_i|x_{i-1})$$

- Backward recursion

$$r_n(x_n) := 1 \text{ and } r_i(x_i) := \sum_{x_{i+1}} r_{i+1}(x_{i+1})p(x_{i+1}|x_i)$$

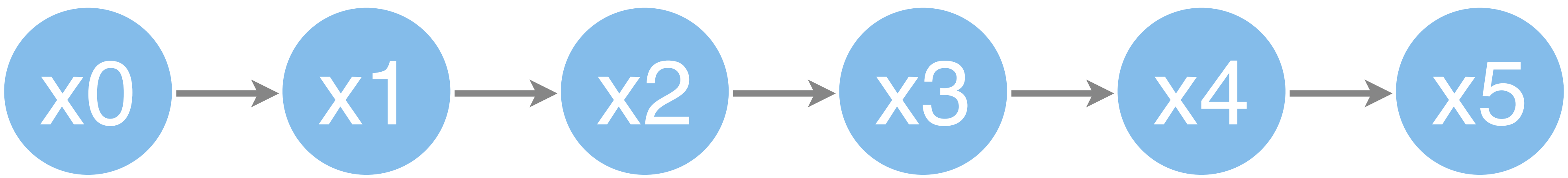
- Marginalization & conditioning

$$p(x_i) = l_i(x_i)r_i(x_i)$$

$$p(x_{-i}|x_i) = \frac{p(x)}{p(x_i)}$$

$$p(x_i, x_{i+1}) = l_i(x_i)p(x_{i+1}|x_i)r_i(x_{i+1})$$

Chains



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

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

- 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

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

Example - inferring lunch

current

		
	0.9	0.2
	0.1	0.8

- Initial probability p(x0=t)=p(x0=b) = 0.5
- Stationary transition matrix
- On fifth day observed at Tazza d'oro p(x5=t)=1
- Distribution on day 3
  - Left messages to 3
  - Right messages to 3
  - Renormalize

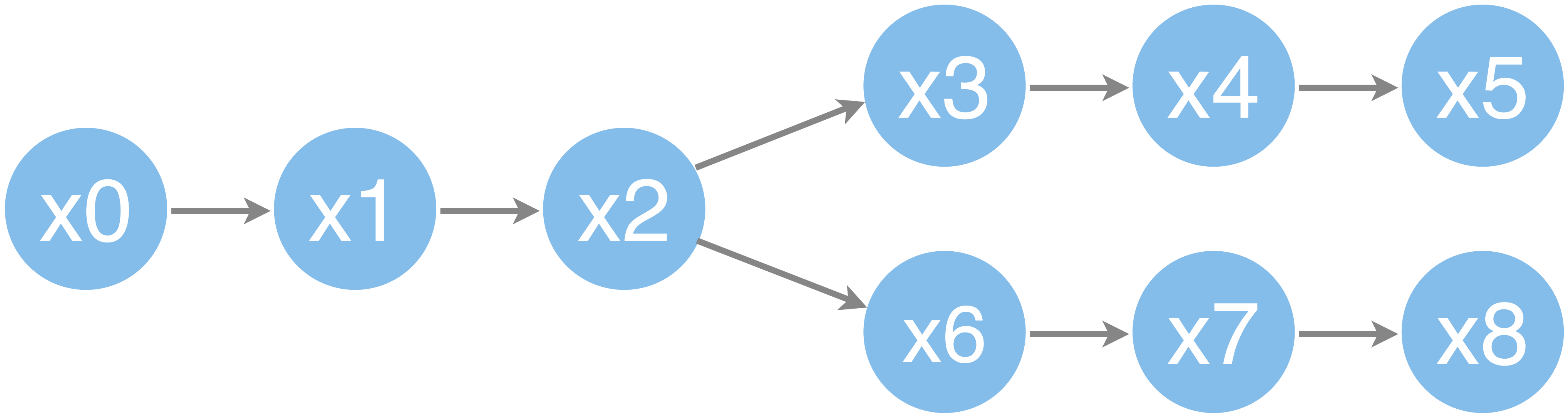
Example - inferring lunch

current

		
	0.9	0.2
	0.1	0.8

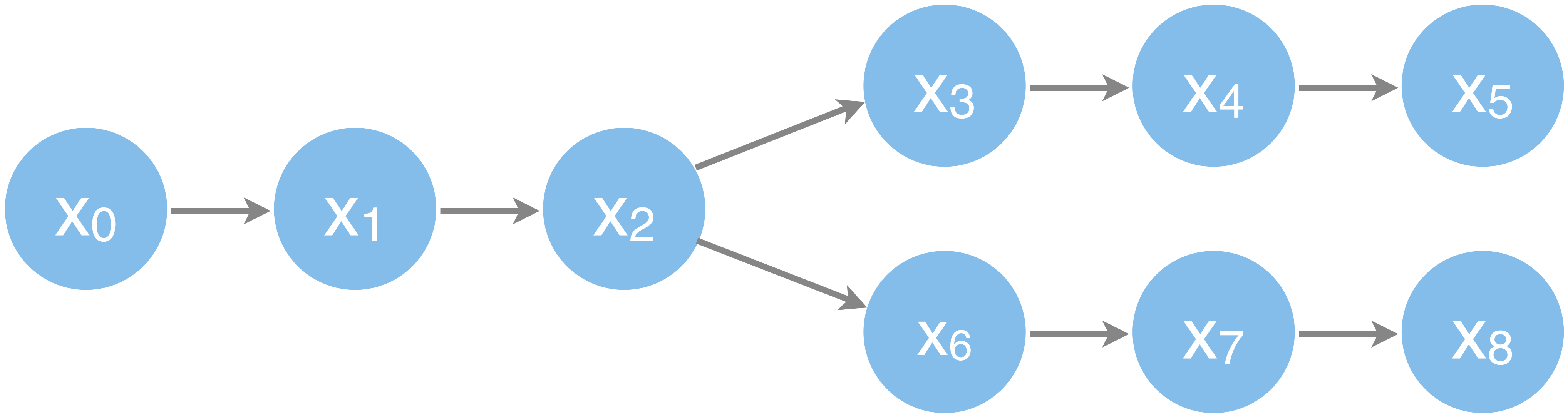
```
> Pi = [0.9, 0.2; 0.1 0.8]
Pi =
   0.90000   0.20000
   0.10000   0.80000
> l1 = [0.5; 0.5];
> l3 = Pi * Pi * l1
l3 =
   0.58500
   0.41500
> r5 = [1; 0];
> r3 = Pi' * Pi' * r5
r3 =
   0.83000
   0.34000
> (l3 .* r3) / sum(l3 .* r3)
ans =
   0.77483
   0.22517
```

Trees

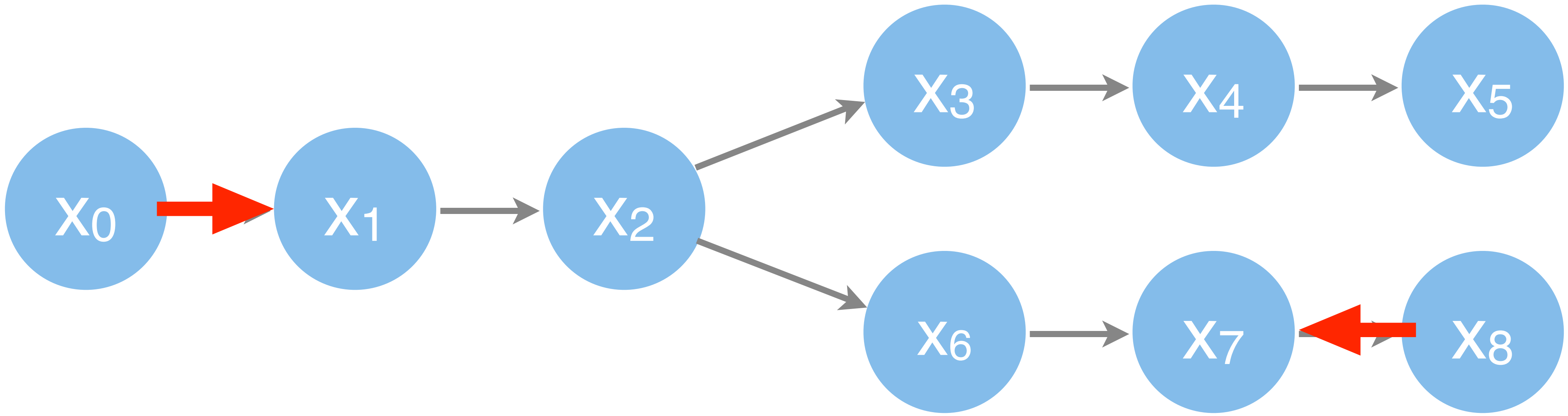


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

Trees

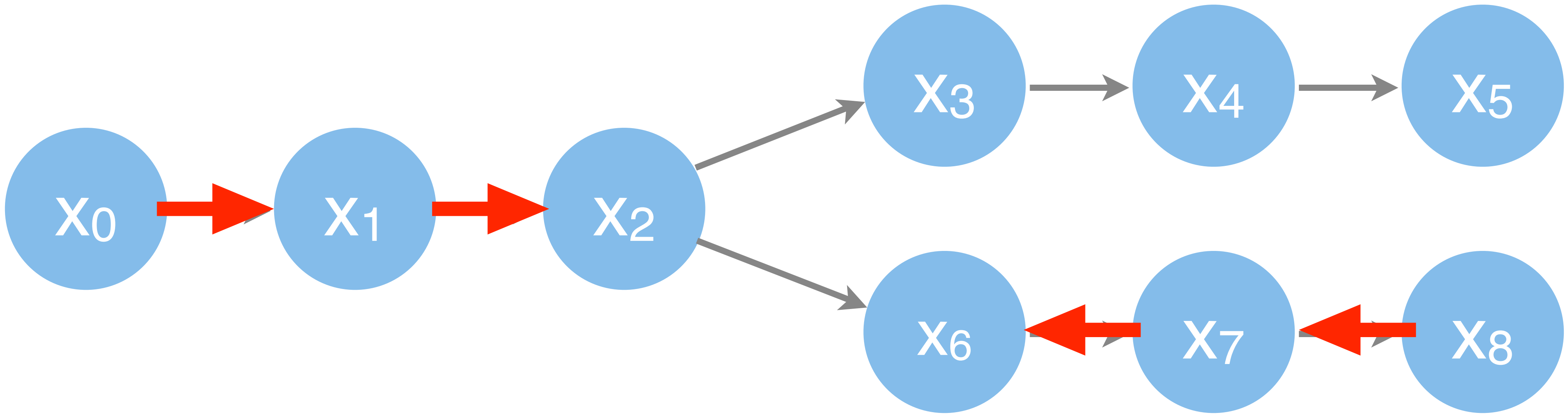


Trees



$$l_1(x_1) = \sum_{x_0} p(x_0)p(x_1|x_0) \qquad r_7(x_7) = \sum_{x_8} p(x_8|x_7)$$

Trees



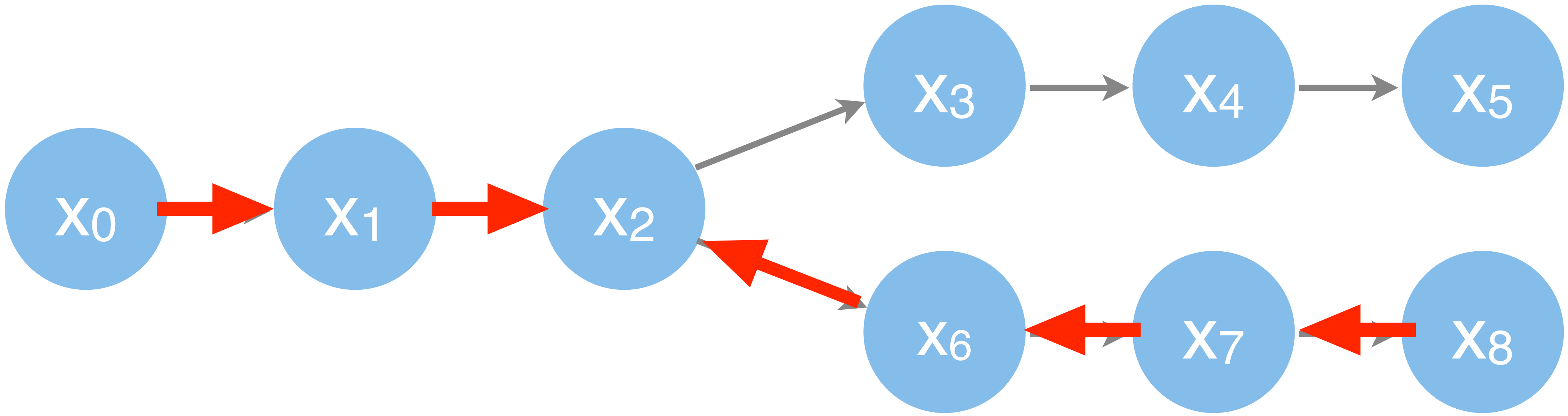
$$l_1(x_1) = \sum_{x_0} p(x_0) p(x_1 | x_0)$$

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

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

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

Trees



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

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

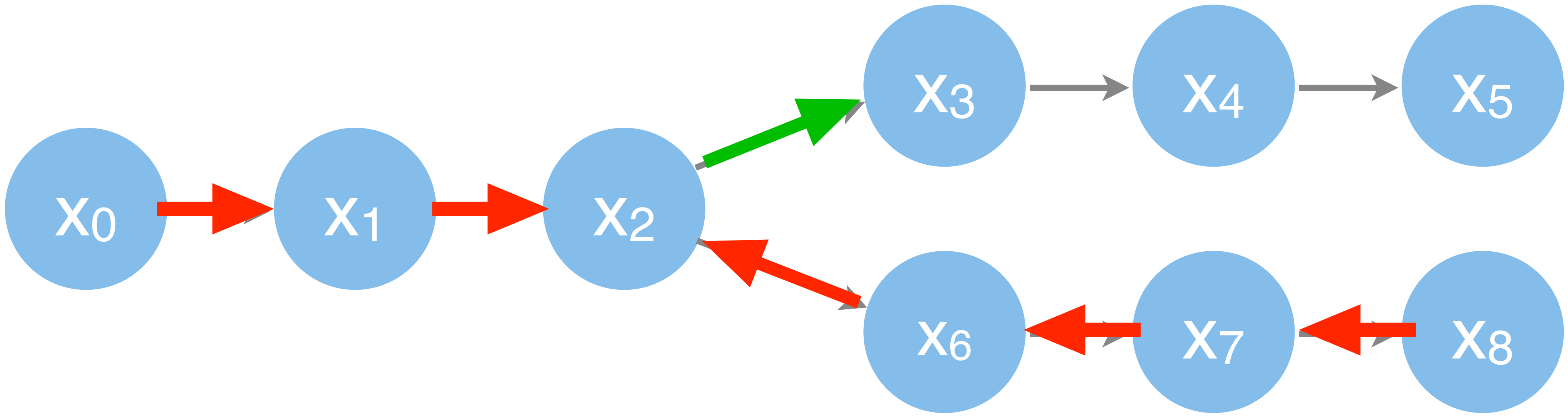
$$r_7(x_7) = \sum_{x_8} p(x_8|x_7)$$

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

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

Carnegie Mellon University

Trees



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

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

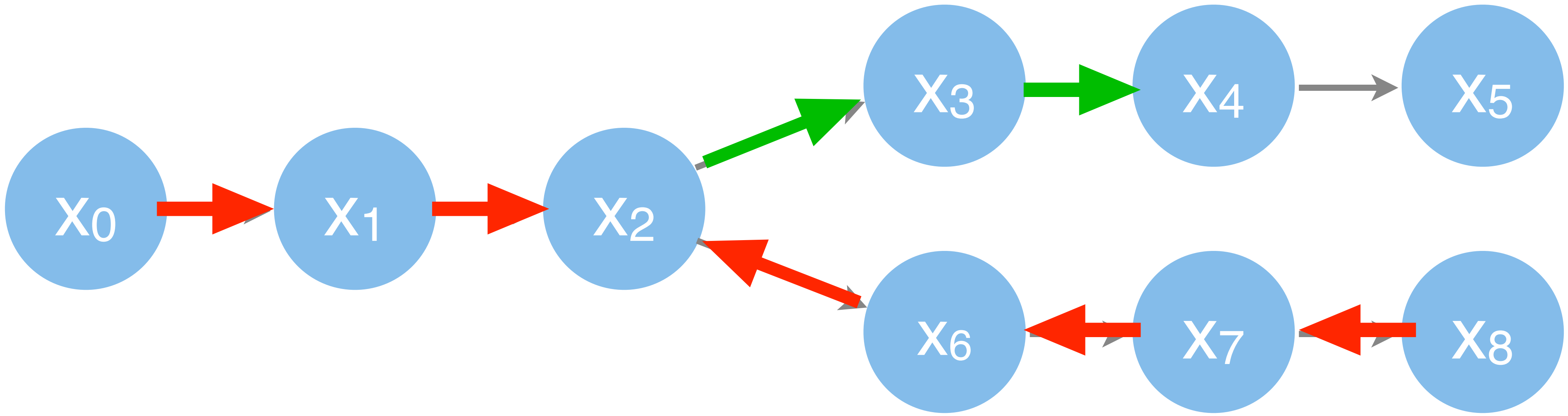
$$l_3(x_3) = \sum_{x_2} l_2(x_2)p(x_3|x_2)r_2(x_2)$$

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

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

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

Trees



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

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

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

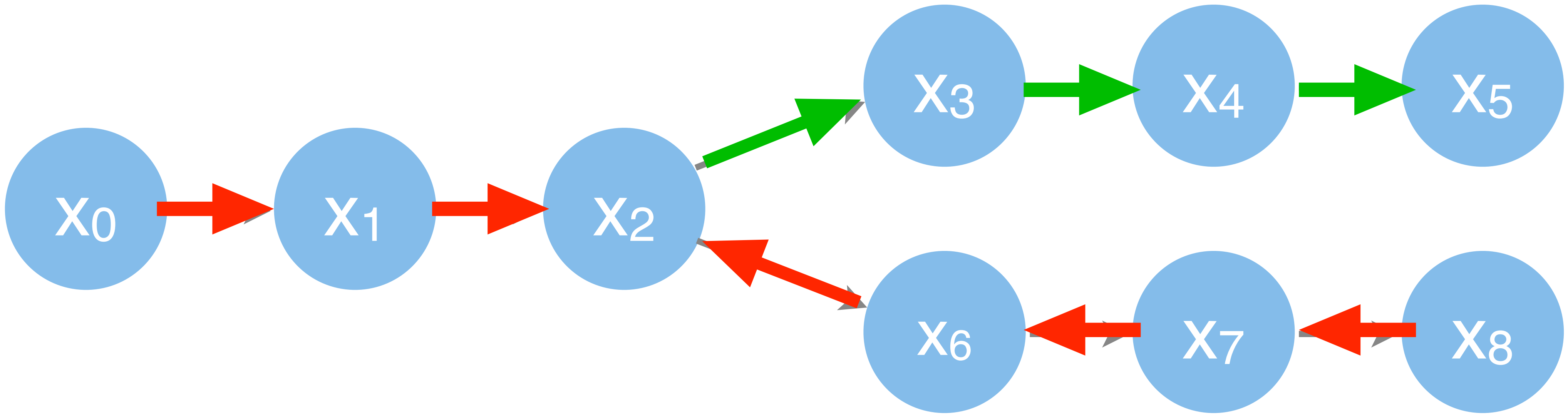
$$r_6(x_6) = \sum_{x_7} r_7(x_7)p(x_7|x_6)$$

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

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

Carnegie Mellon University

Trees



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

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

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

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

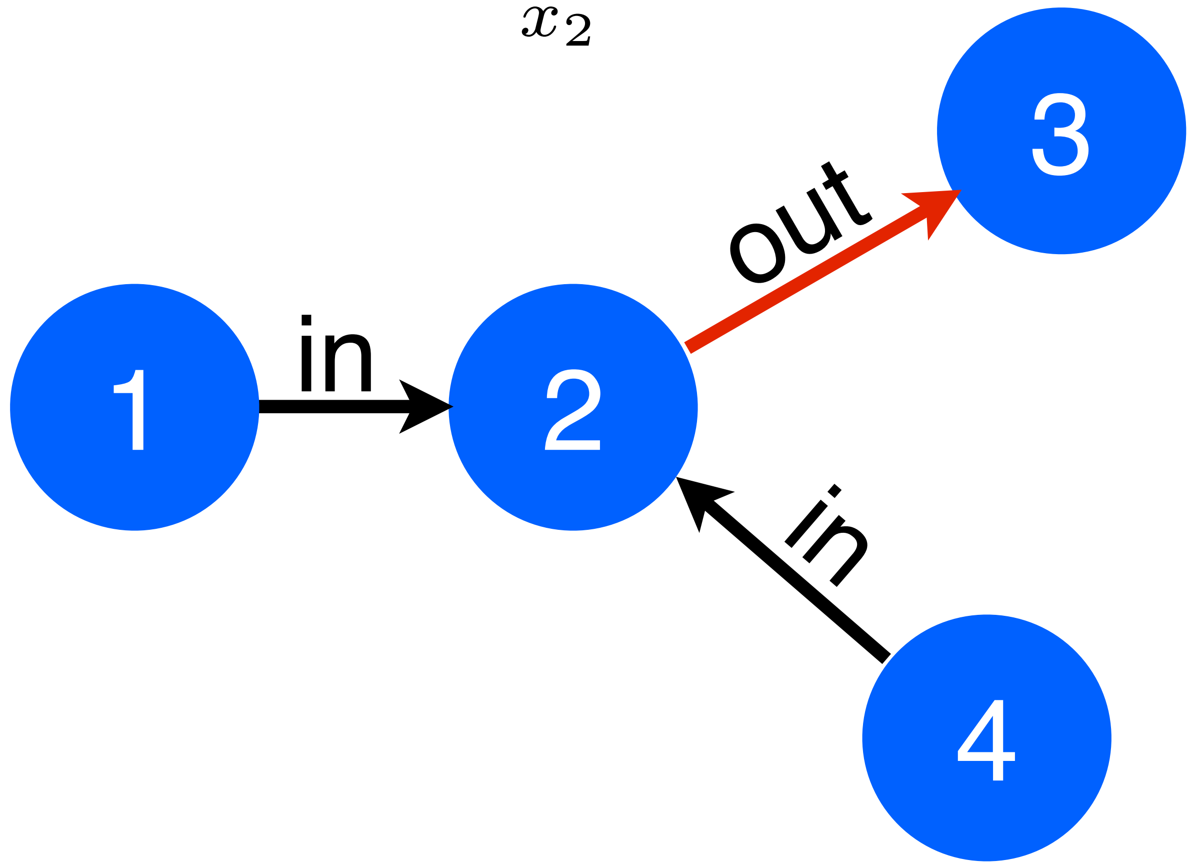
$$r_2(x_2) = \sum_{x_6} r_6(x_6)p(x_6|x_2)$$

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

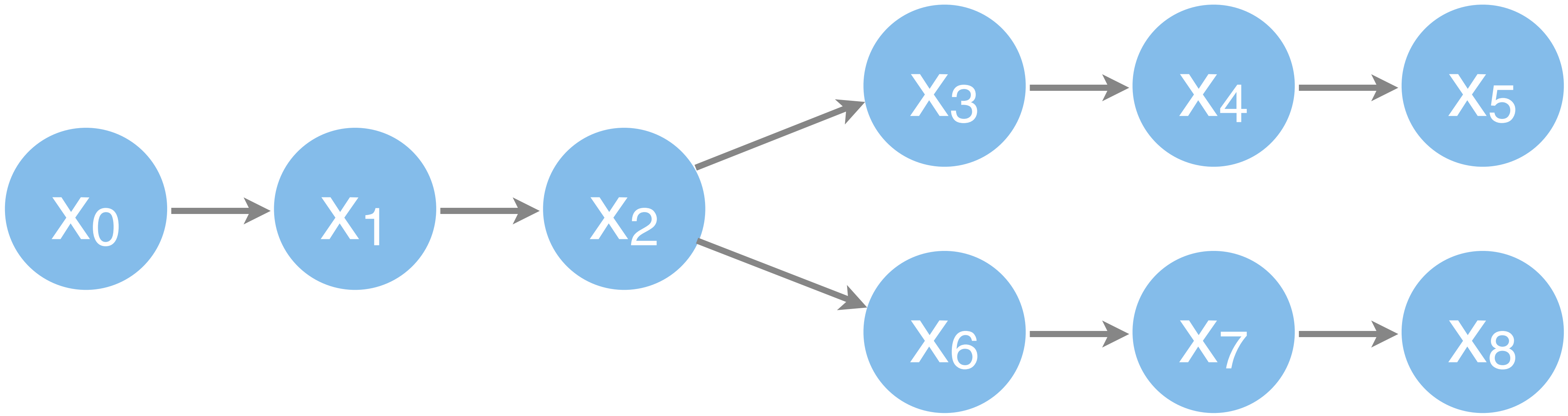
Junction Template

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

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



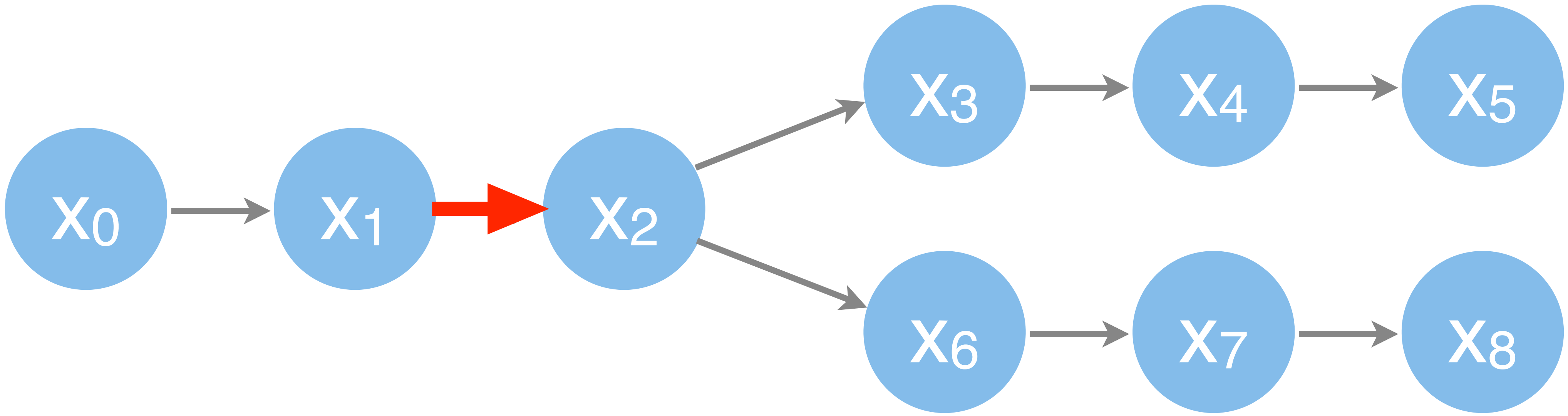
Trees



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

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

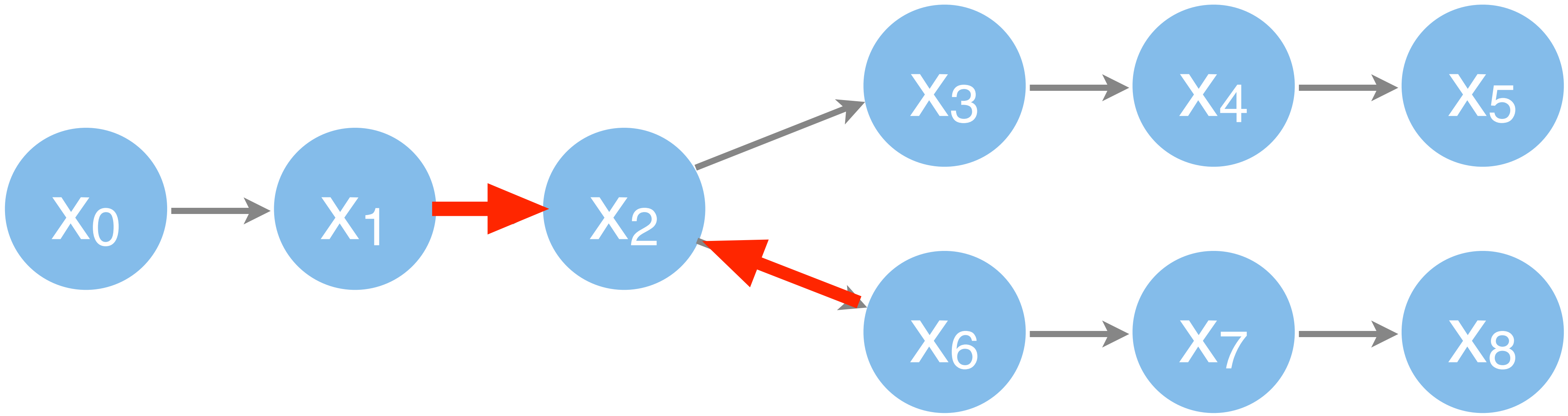
Trees



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

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

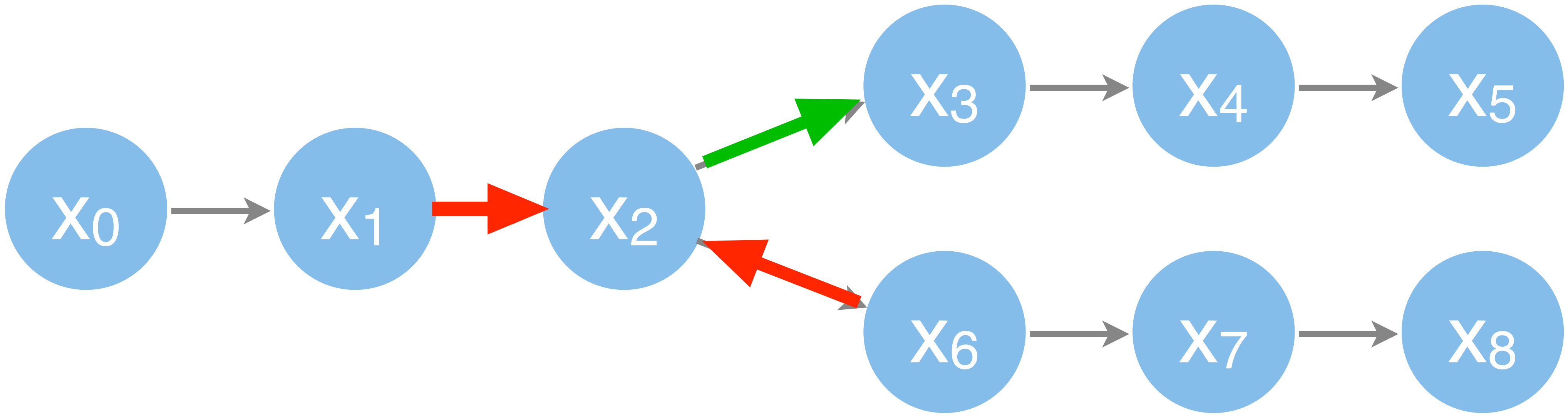
Trees



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

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

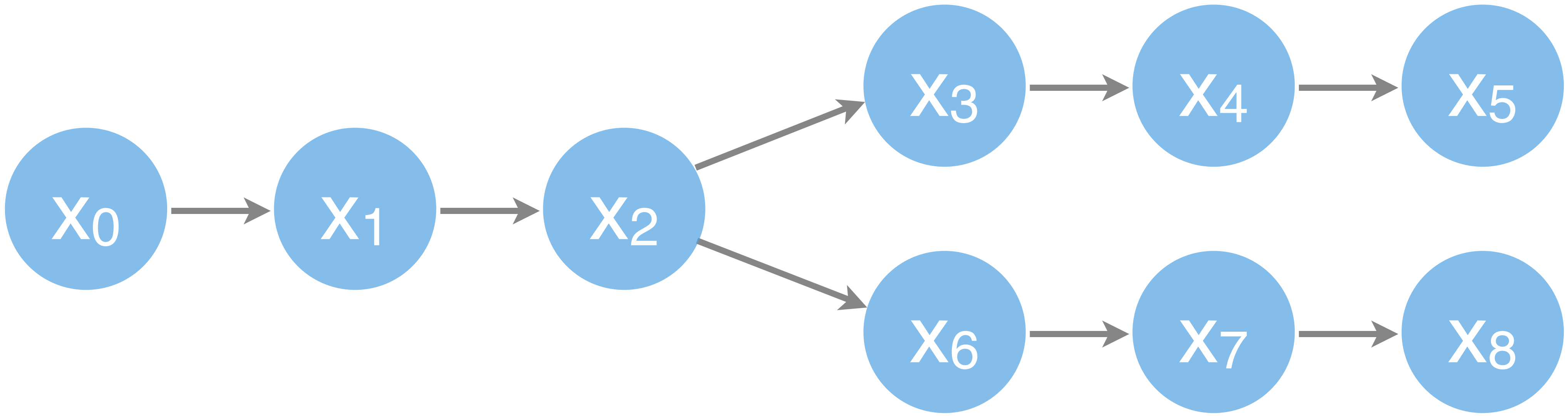
Trees



- 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

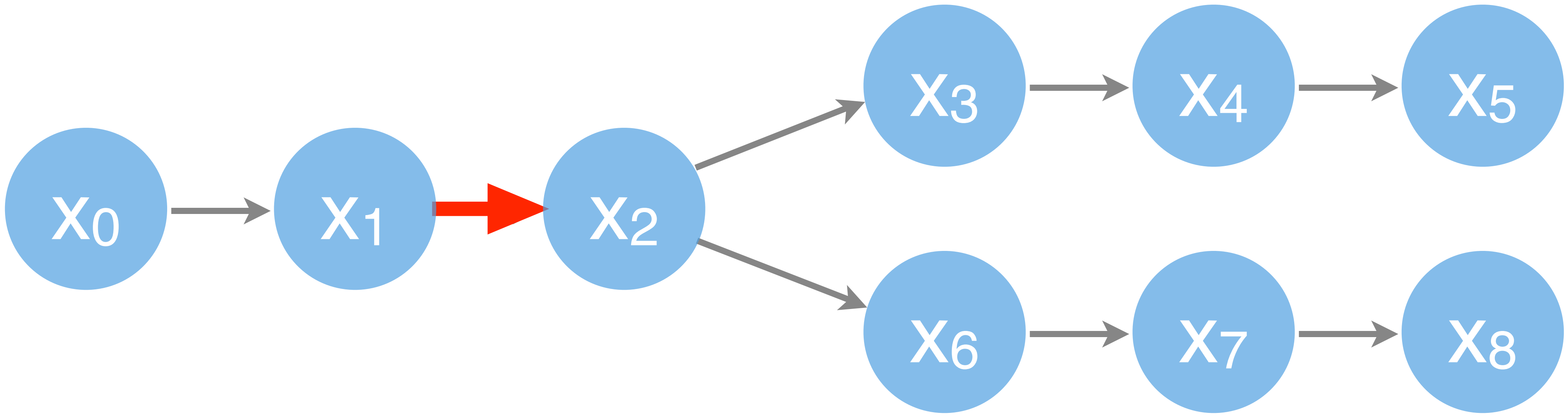


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

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

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

Trees

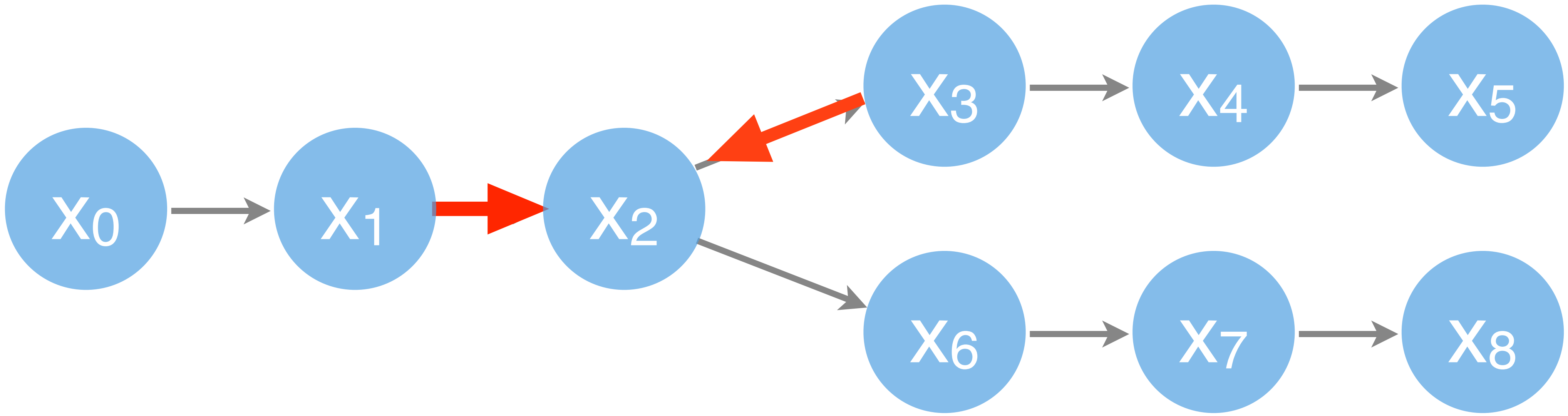


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

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

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

Trees

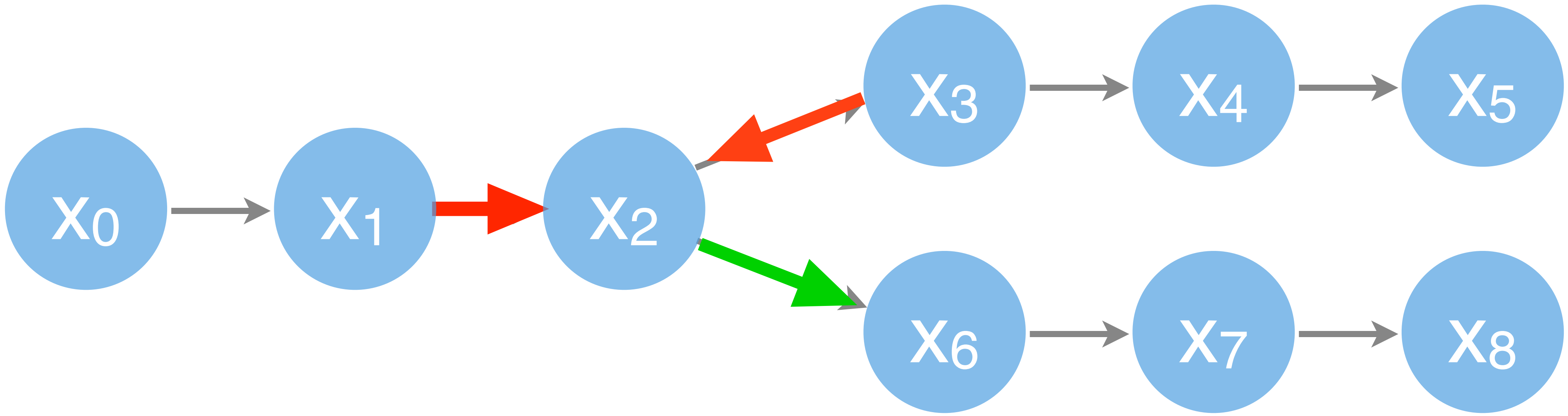


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

$$m_{2\to3}(x_3) = \sum_{x_2} m_{1\to2}(x_2) m_{6\to2}(x_2) f(x_2, x_3)$$

$$m_{2\to6}(x_6) = \sum_{x_2} m_{1\to2}(x_2) m_{3\to2}(x_2) f(x_2, x_6)$$

Trees

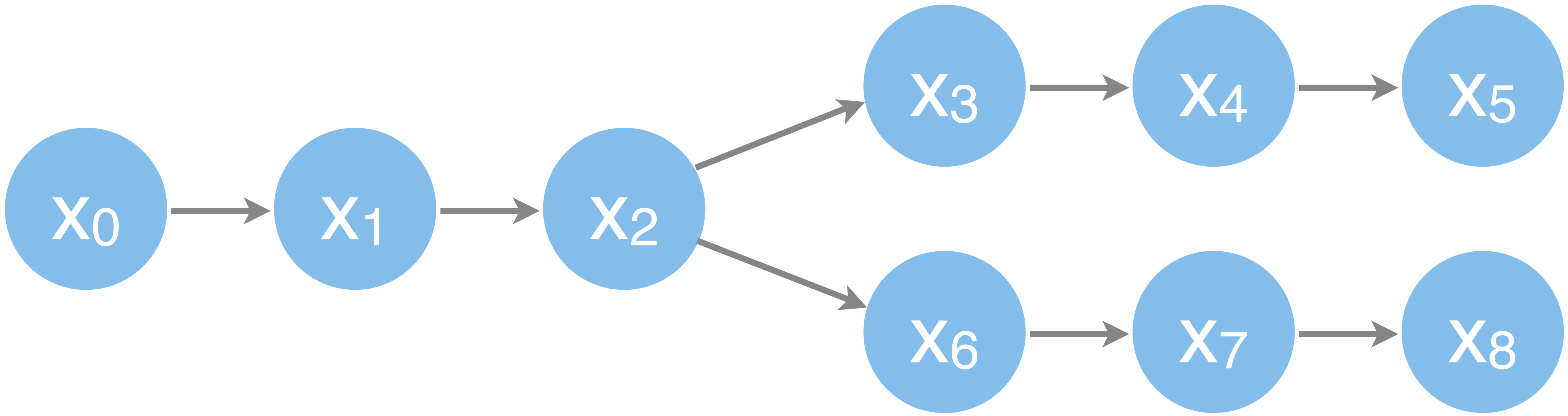


- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

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

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

Trees



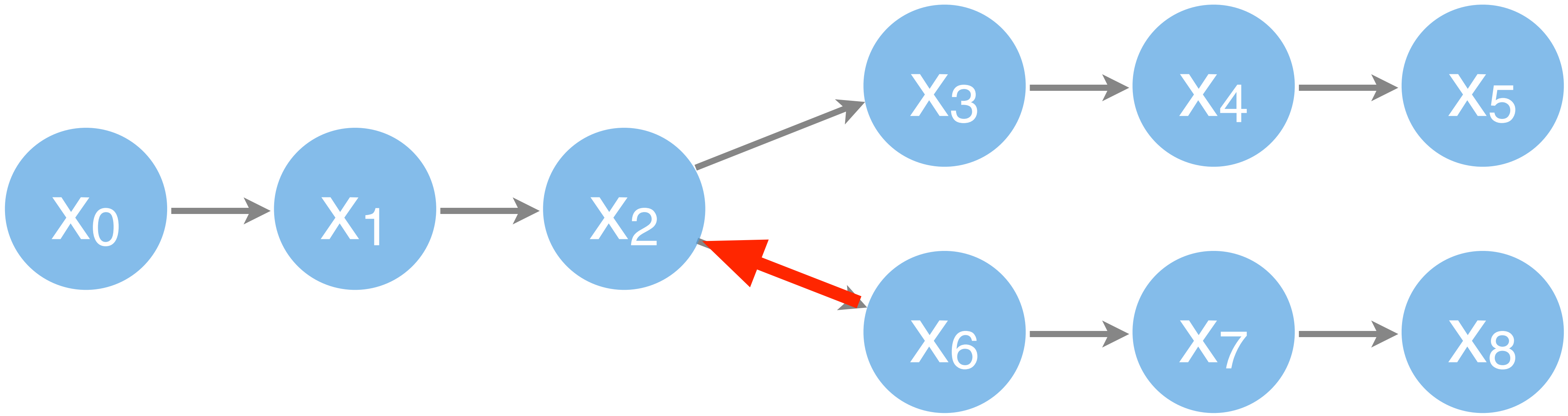
- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

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

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

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

Trees



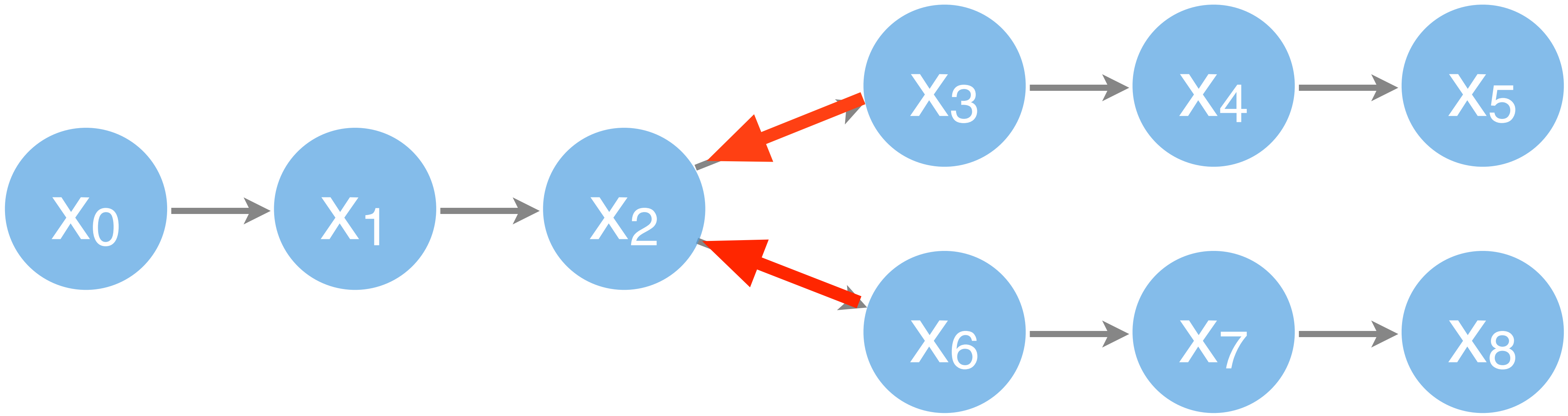
- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

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

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

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

Trees



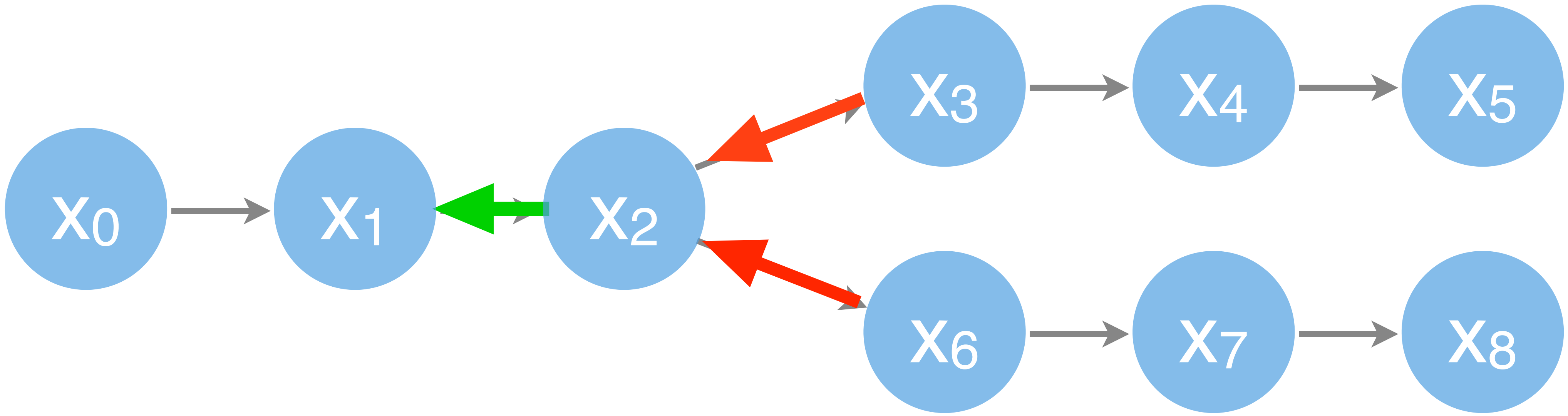
- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

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

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

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

Trees



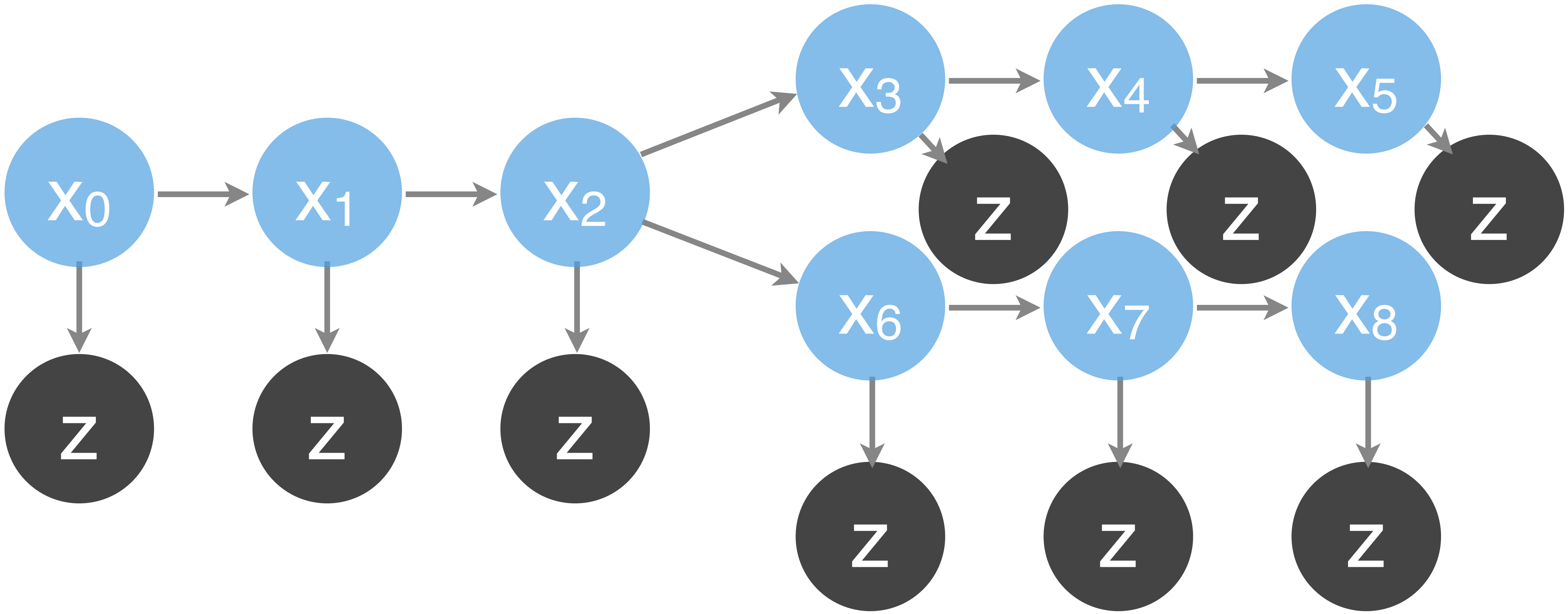
- Forward/Backward messages as normal for chain
- When we have more edges for a vertex use ...

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

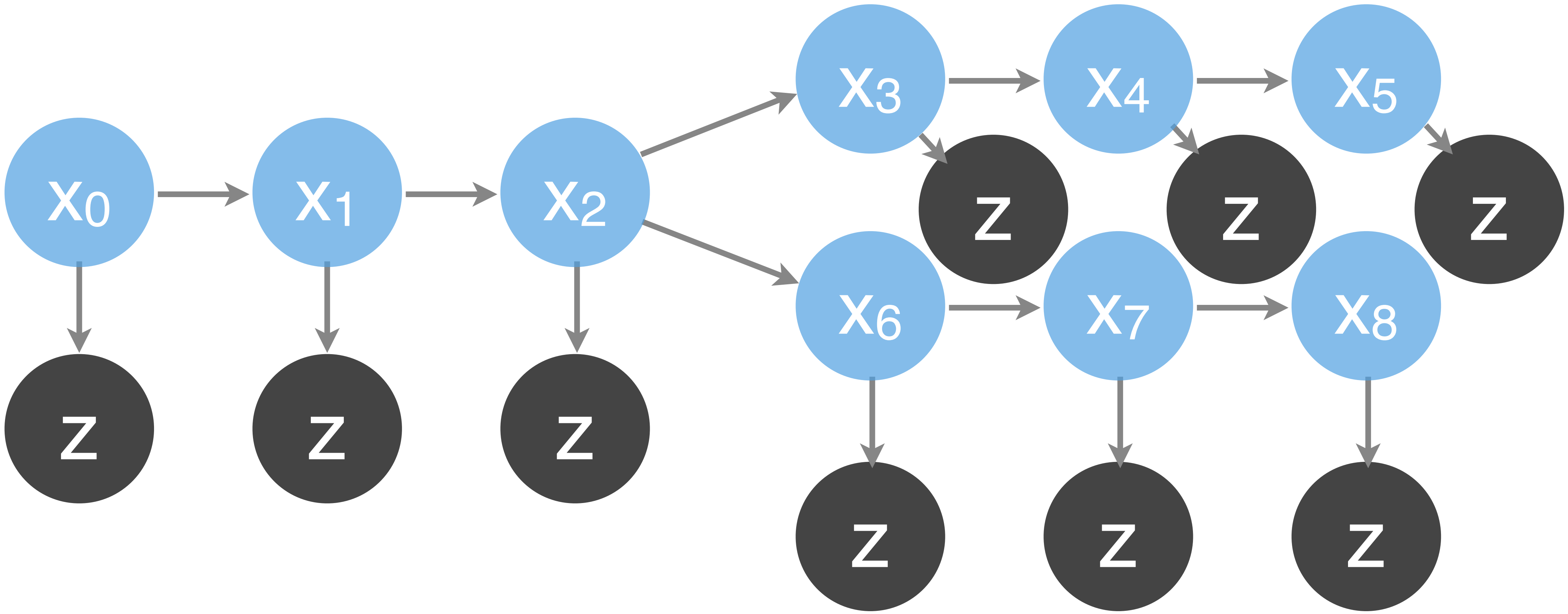
$$m_{2\to 6}(x_6) = \sum_{x_2} m_{1\to 2}(x_2) m_{3\to 2}(x_2) f(x_2, x_6)$$

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

Trees



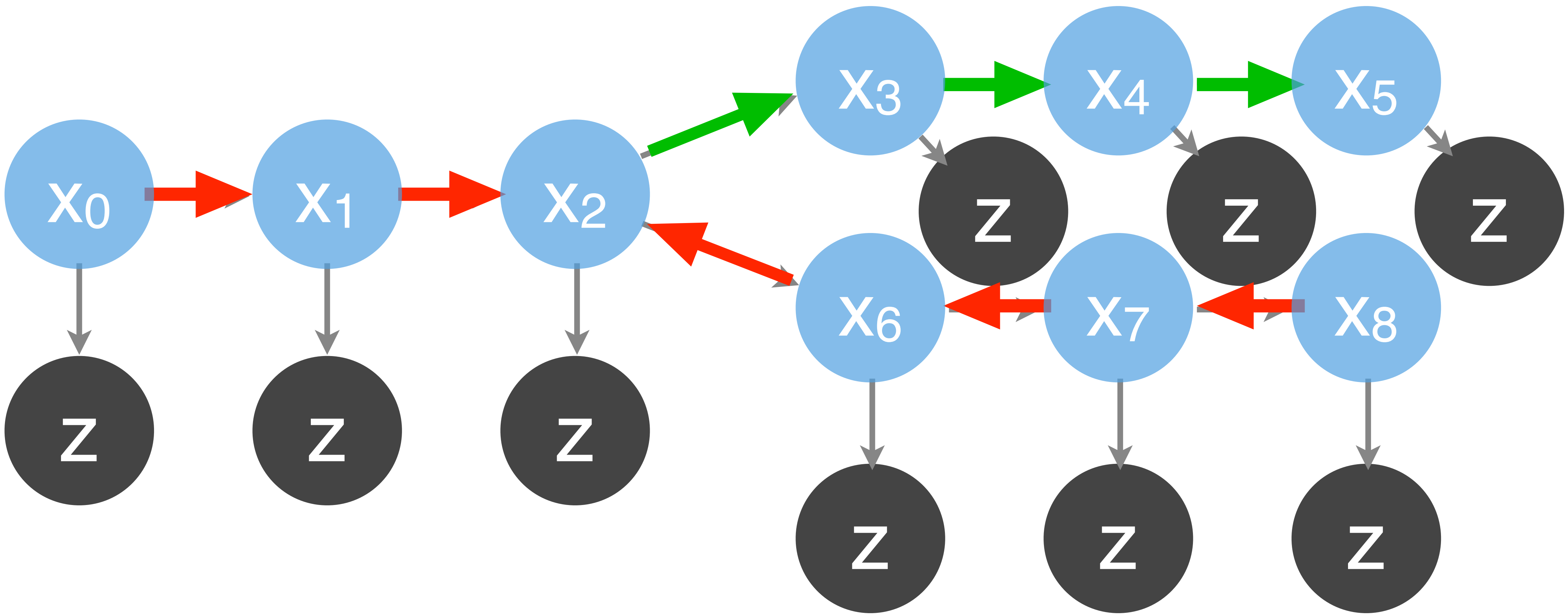
Trees



- Joint distribution over latent state and observations
- To compute conditional probability we normalize

$$p(x, z) = p(x) \prod_i p(z_i | x_i) = \prod_{i,j \in T} f(x_i, x_j) \prod_i g(x_i, z_i)$$

Trees

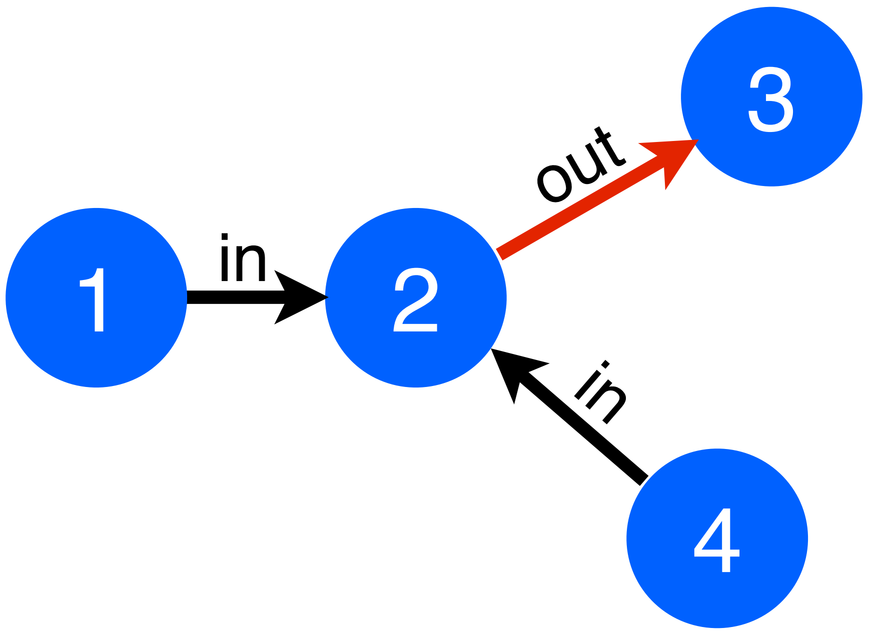


$$p(x_i|\text{rest}) \propto \sum_{x^{-i}} \left[\prod_{j,k \in T} f(x_j, x_k) \prod_j g(x_j) \right]$$

$$= g(x_i) \prod_{(j,i)} m_{j \to i}(x_i)$$

Summary

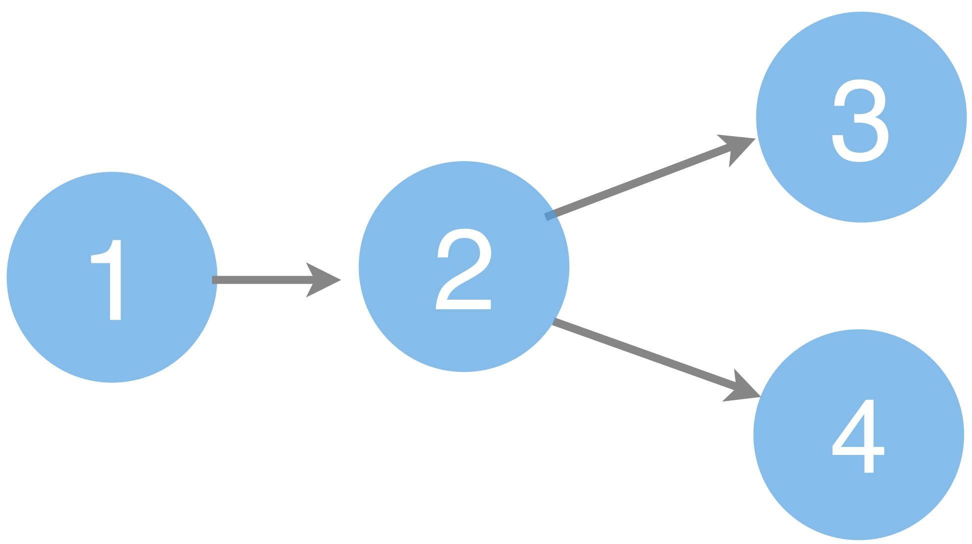
- Markov chains
  - Present only depends on recent past
  - Higher order - longer history.
- Dynamic programming
  - Exponential if brute force.
  - Linear in chain if we iterate.
  - For junctions treat like chains but integrate signals from all sources.
  - Exponential in the history size.

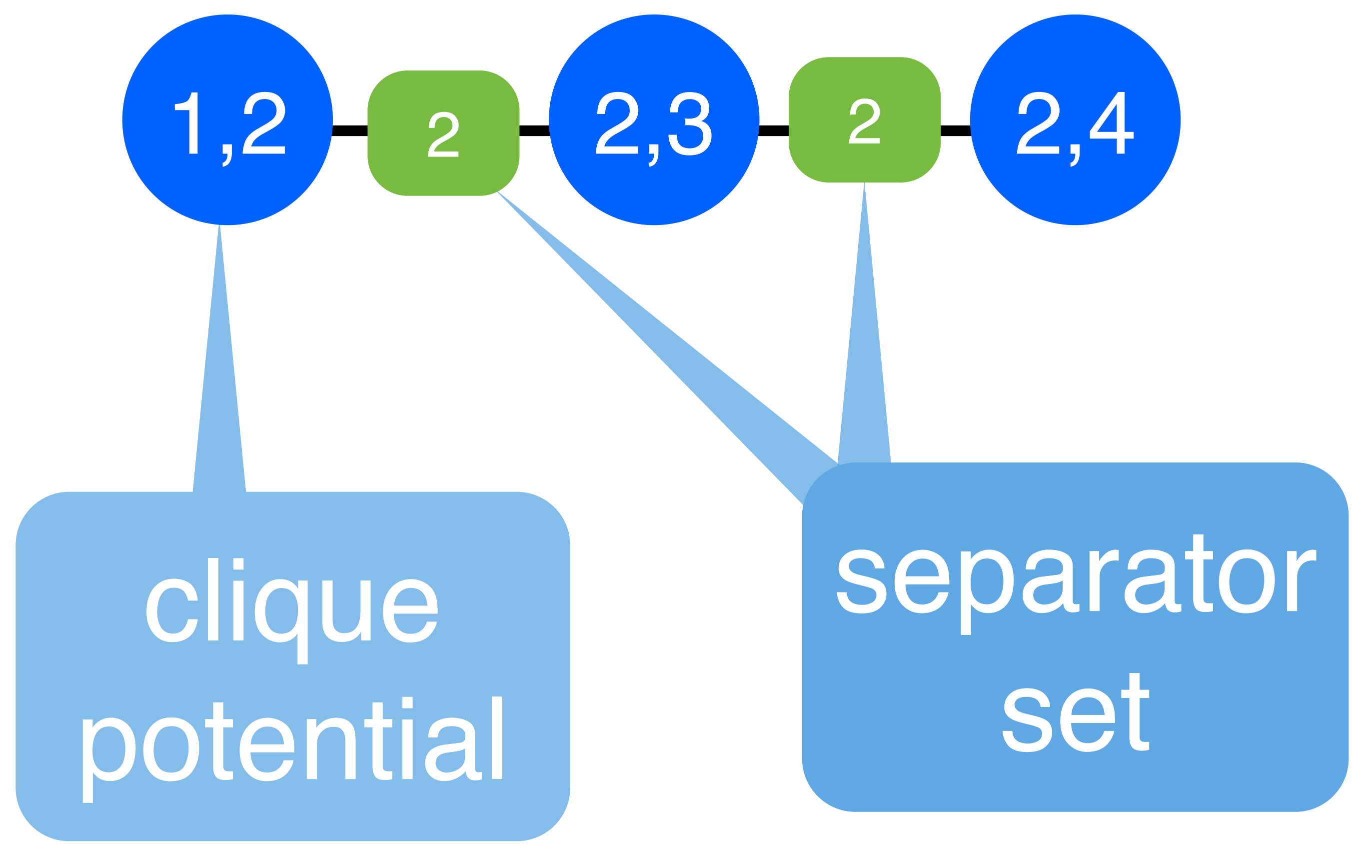



Junction Trees
OVERSIZE

Junction Trees

$$f(x_1, x_2) f(x_2, x_3) f(x_2, x_4)$$



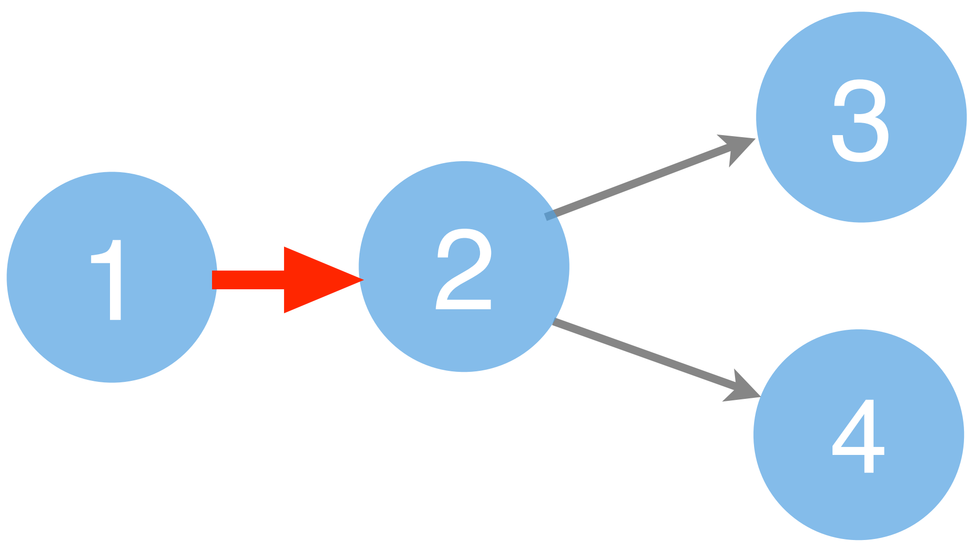


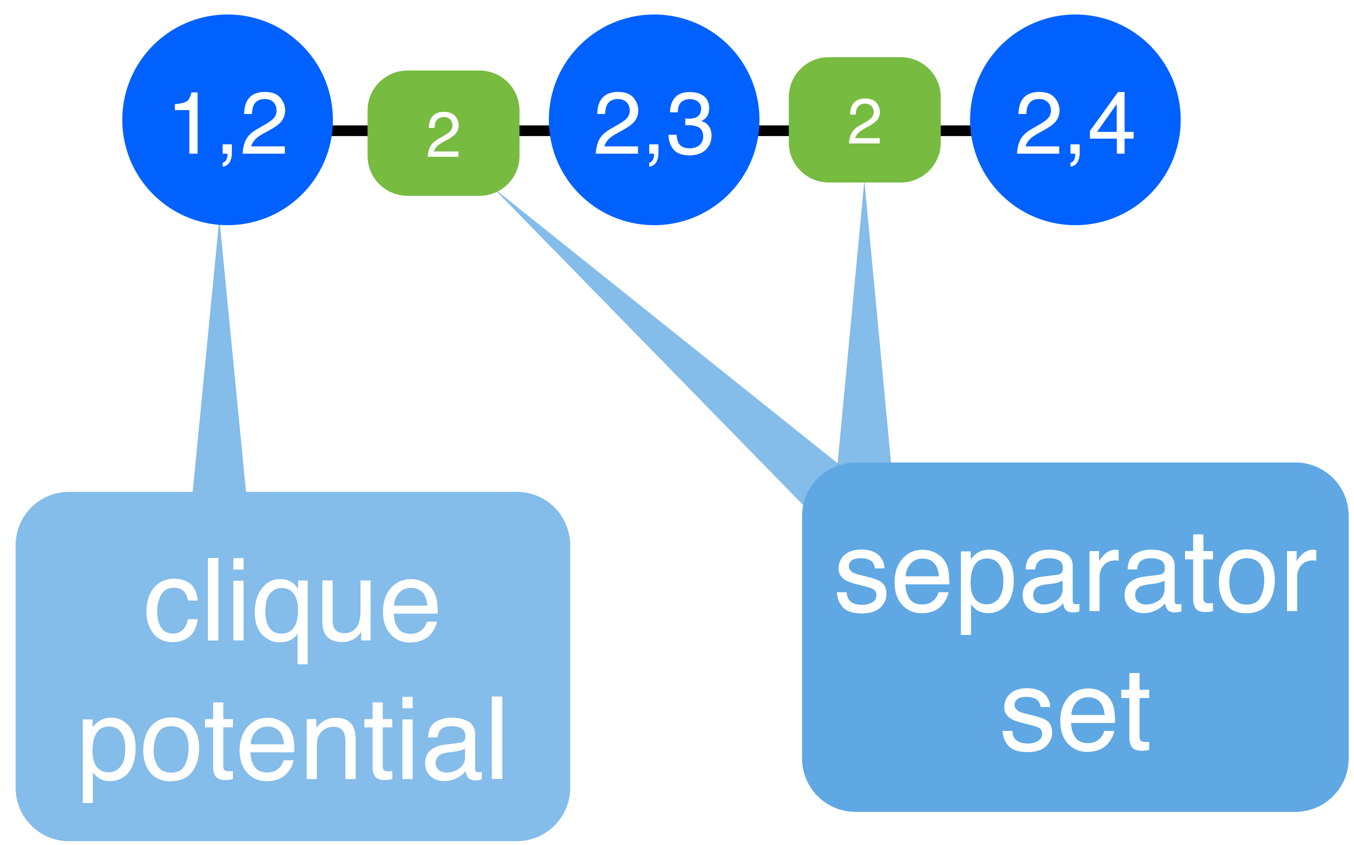
$$m_{i \to j}(x_j) = \sum_{x_i} f(x_i, x_j) \prod_{l \neq j} m_{l \to i}(x_j)$$

clique potential

Junction Trees

$$f(x_1, x_2) f(x_2, x_3) f(x_2, x_4)$$



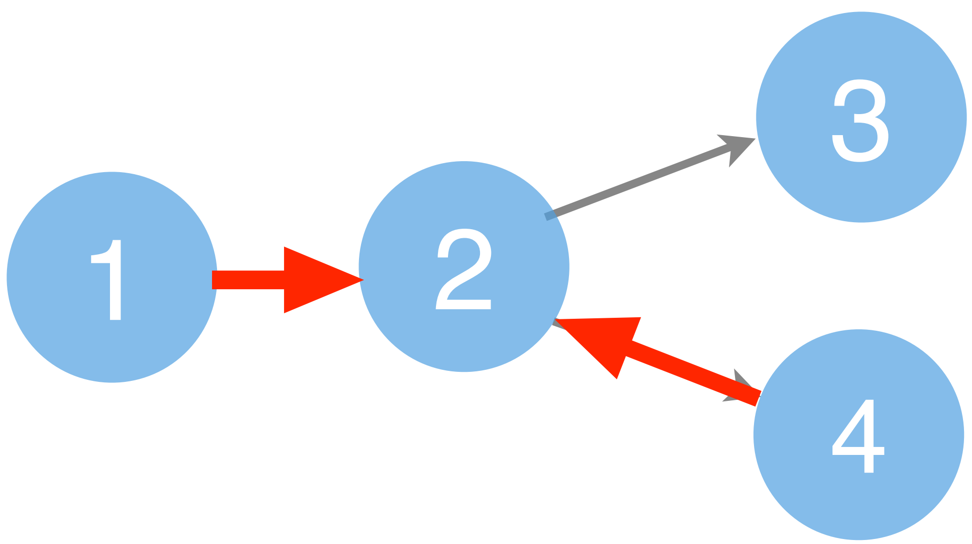


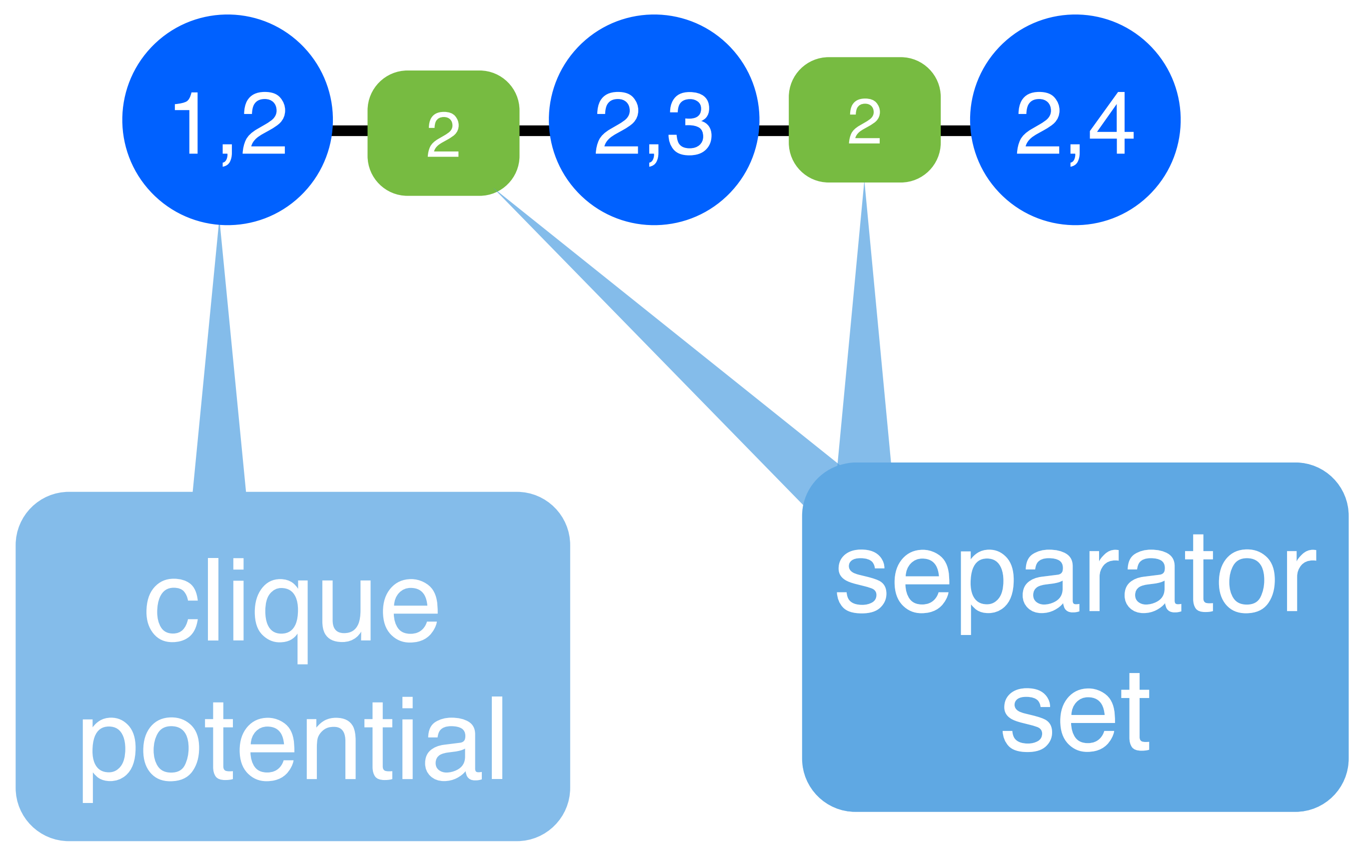
$$m_{i \to j}(x_j) = \sum_{x_i} f(x_i, x_j) \prod_{l \neq j} m_{l \to i}(x_j)$$

clique potential

Junction Trees

$$f(x_1, x_2) f(x_2, x_3) f(x_2, x_4)$$



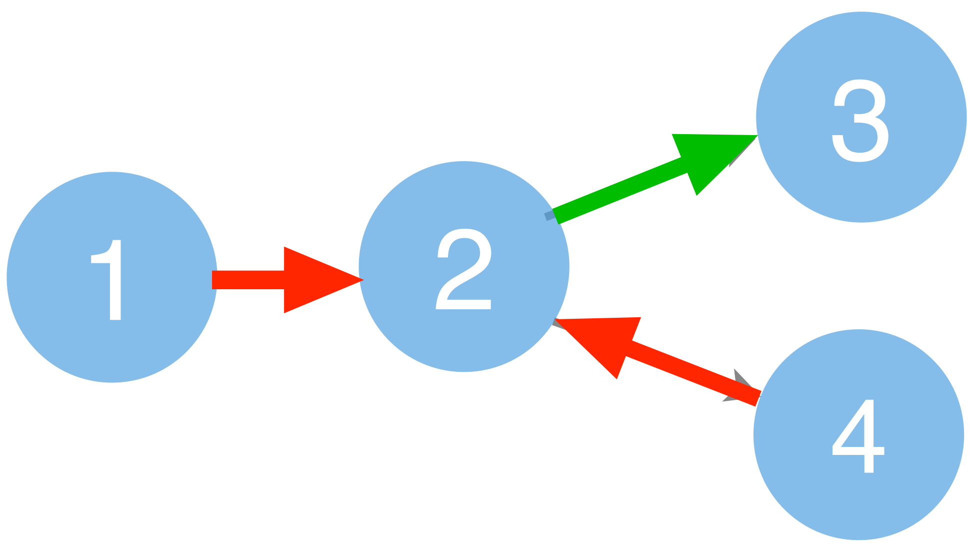


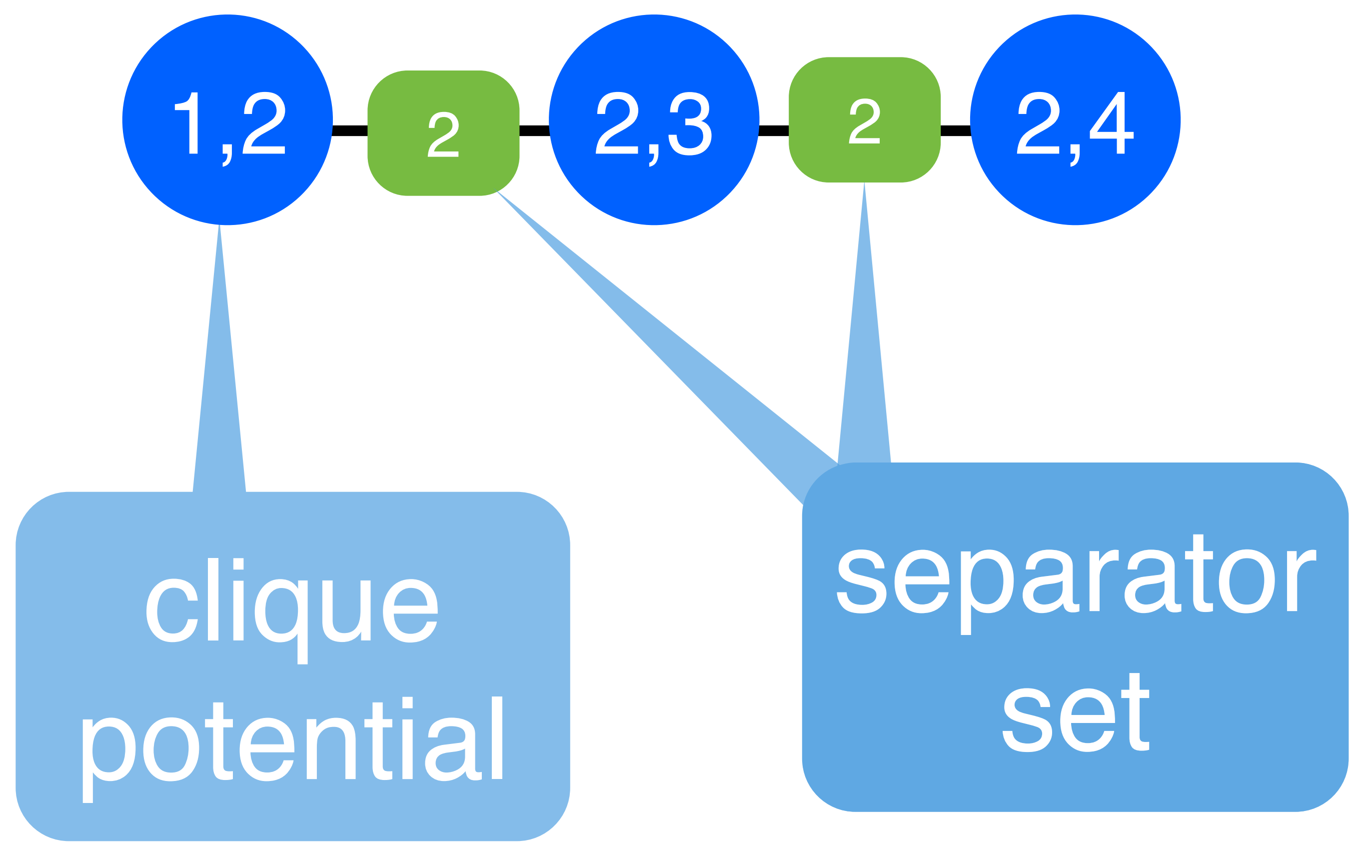
$$m_{i \to j}(x_j) = \sum_{x_i} f(x_i, x_j) \prod_{l \neq j} m_{l \to i}(x_j)$$

clique potential

Junction Trees

$$f(x_1, x_2) f(x_2, x_3) f(x_2, x_4)$$

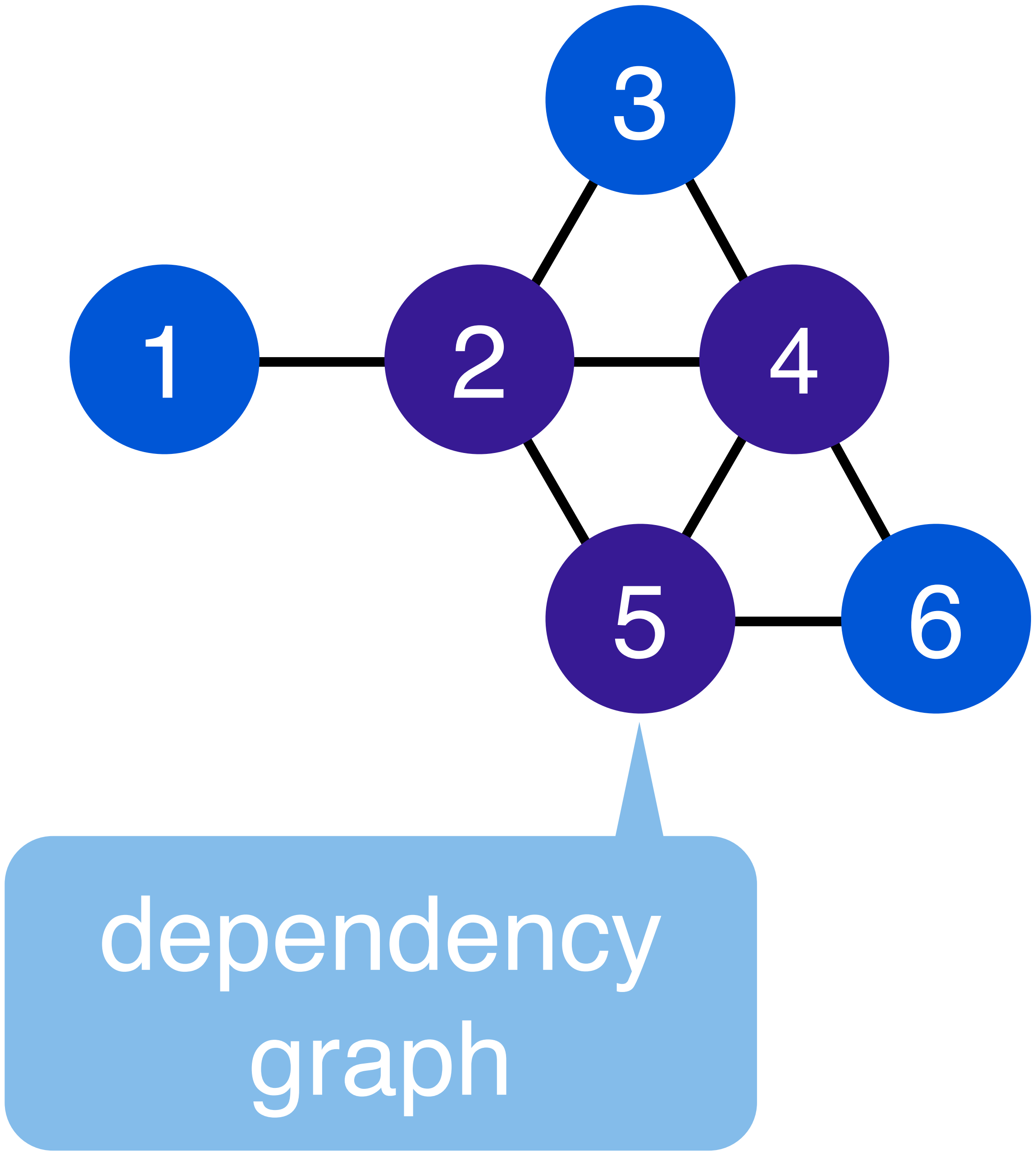




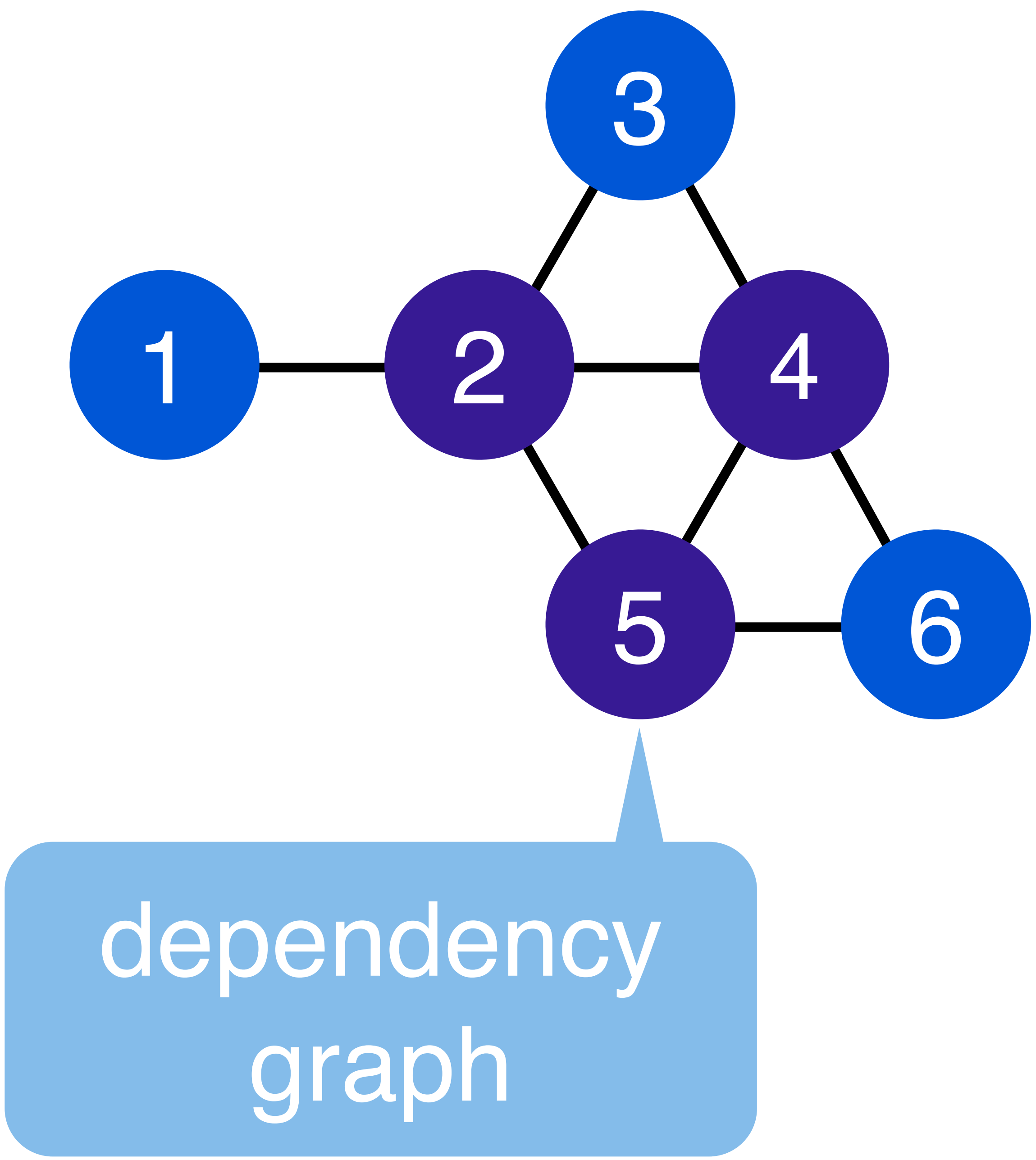
$$m_{i \to j}(x_j) = \sum_{x_i} f(x_i, x_j) \prod_{l \neq j} m_{l \to i}(x_j)$$

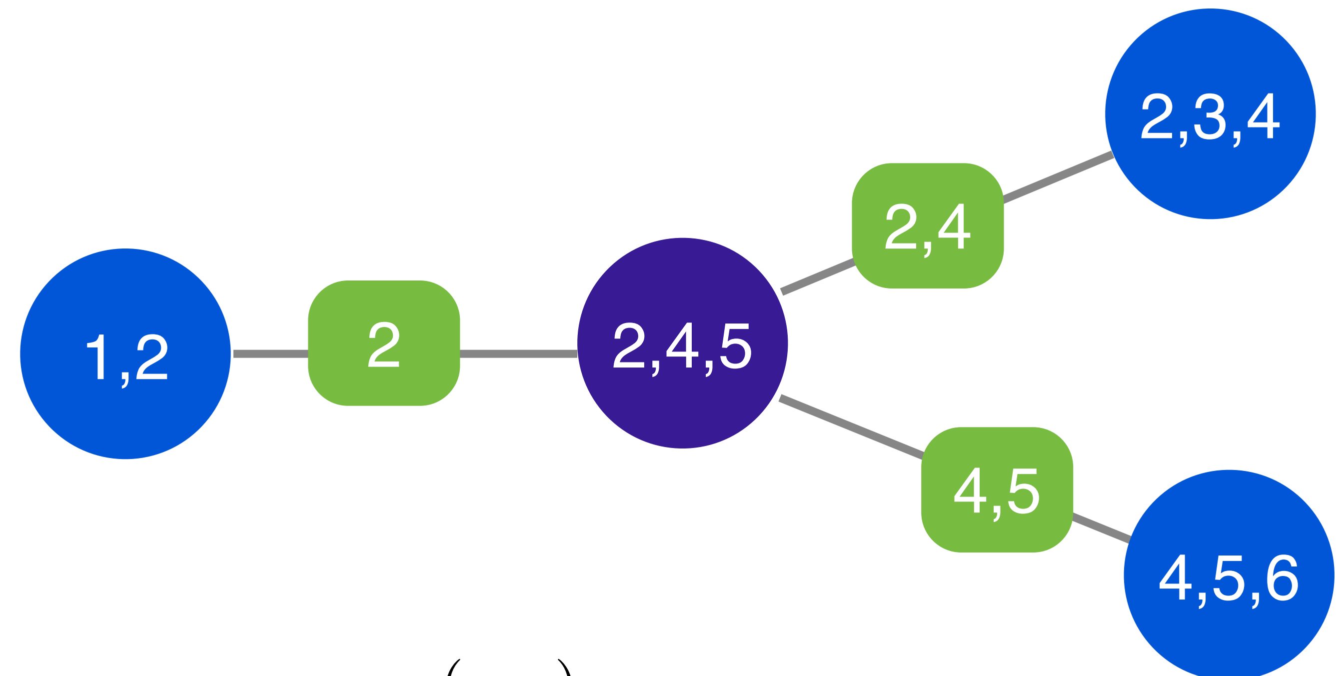
clique potential

Junction Trees



Junction Trees



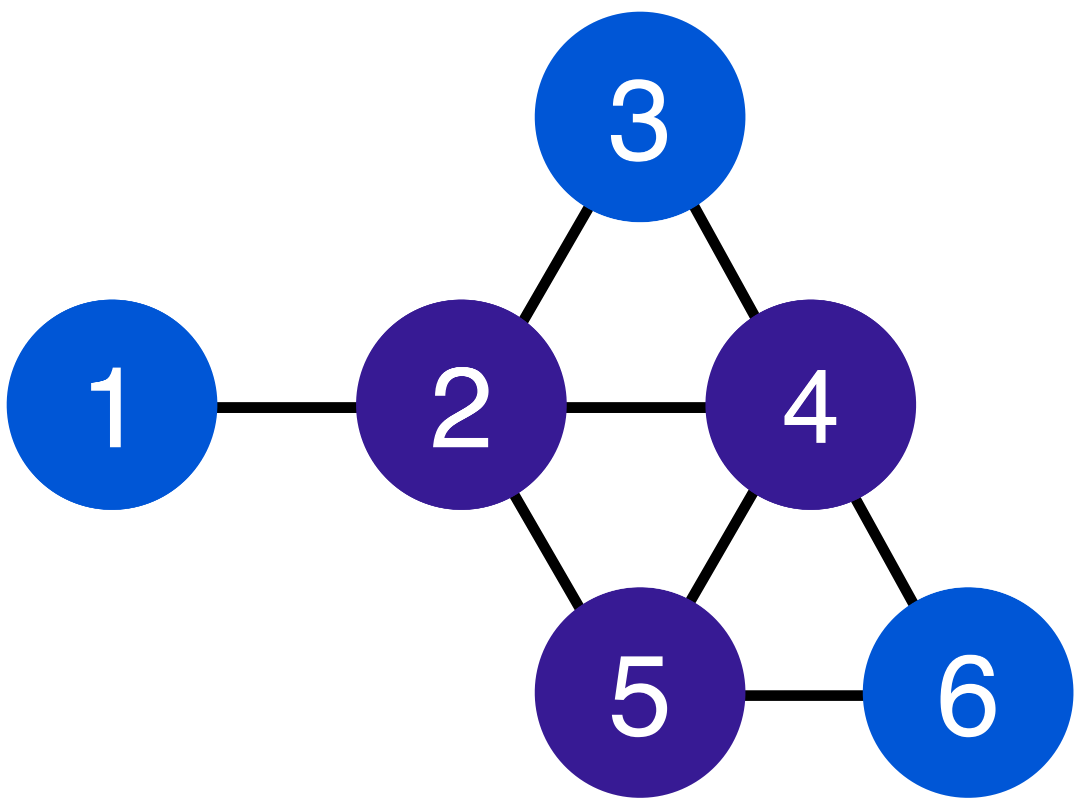


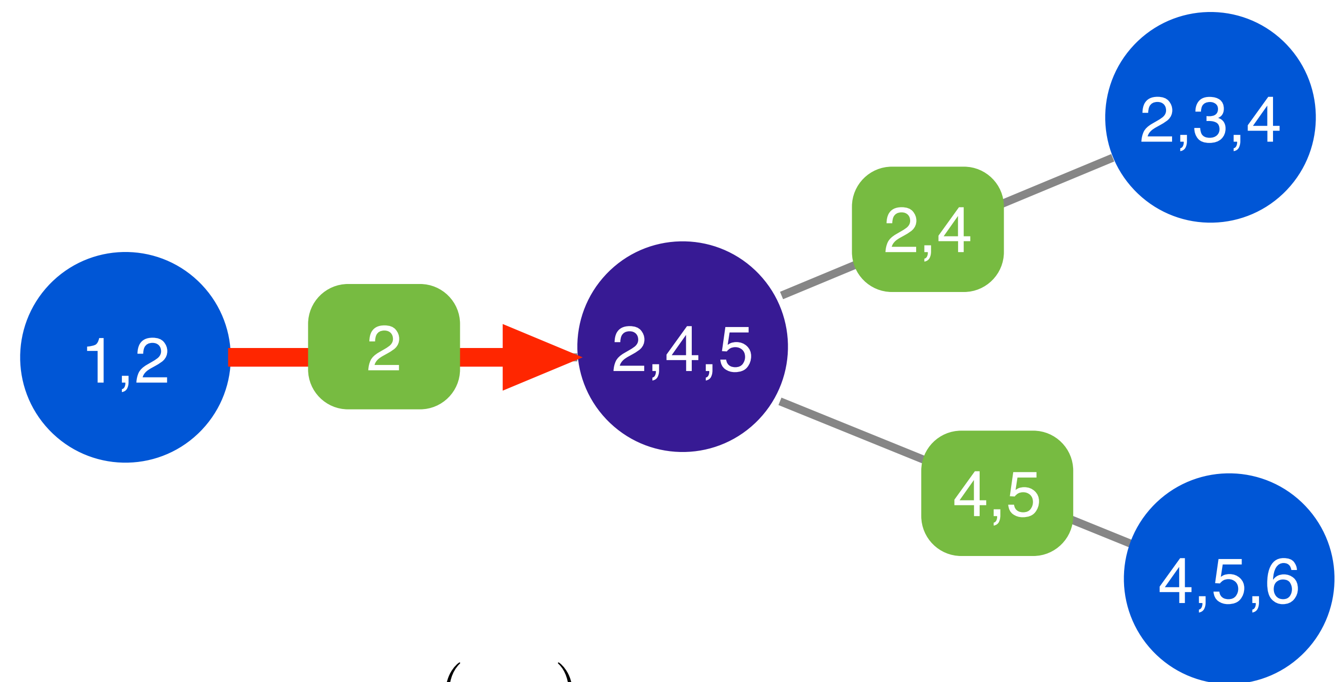
$$m_{245 \to 234}(x_{24}) = \sum_{x_5} f(x_{245}) m_{12 \to 245}(x_2) m_{457 \to 245}(x_{45})$$

clique potential

separator set

Junction Trees





$$m_{245 \to 234}(x_{24})$$
$$= \sum_{x_5} f(x_{245}) m_{12 \to 245}(x_2) m_{457 \to 245}(x_{45})$$

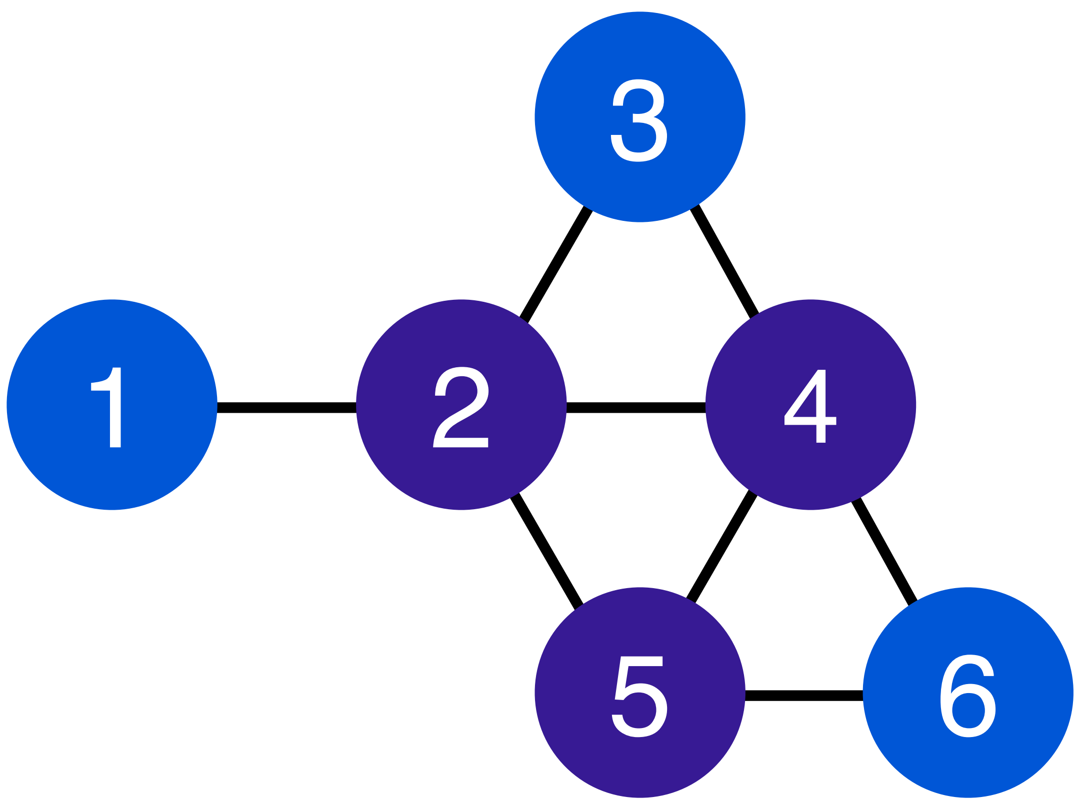
dependency graph

clique potential

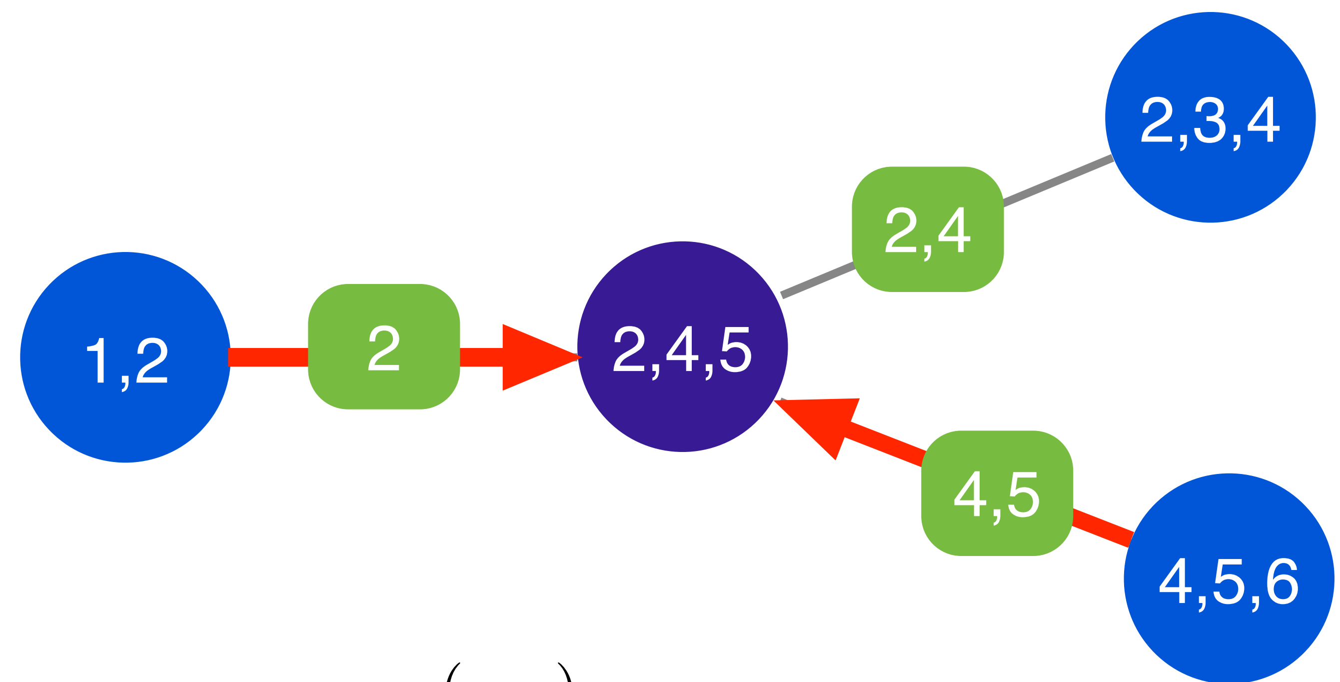
separator set

Carnegie Mellon University

Junction Trees



dependency graph

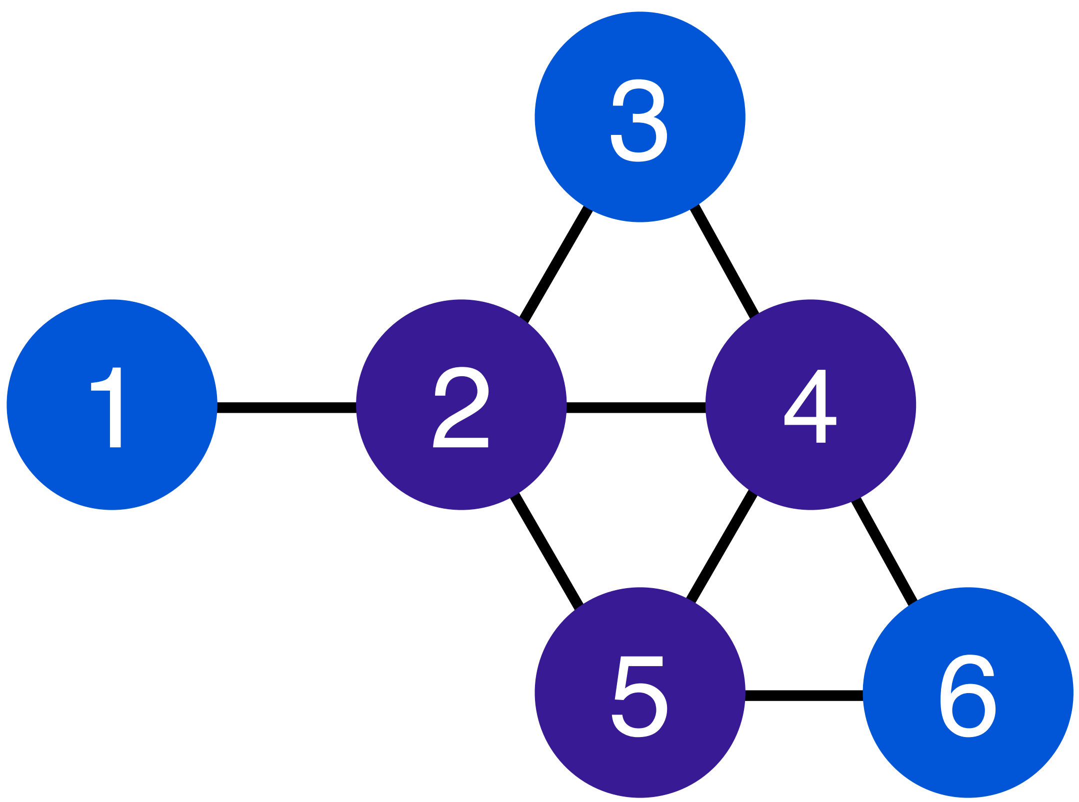


$$m_{245\to 234}(x_{24}) = \sum_{x_5} f(x_{245})\, m_{12\to 245}(x_2)\, m_{457\to 245}(x_{45})$$

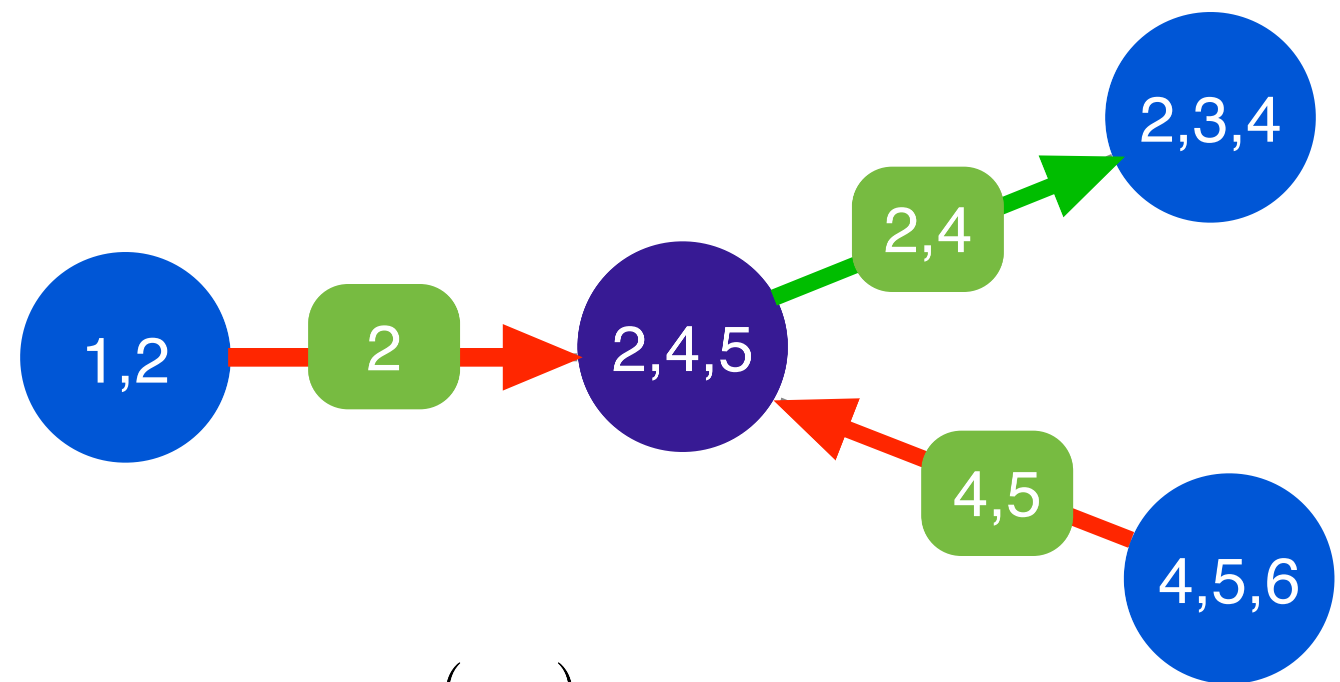
clique potential

separator set

Junction Trees



dependency graph

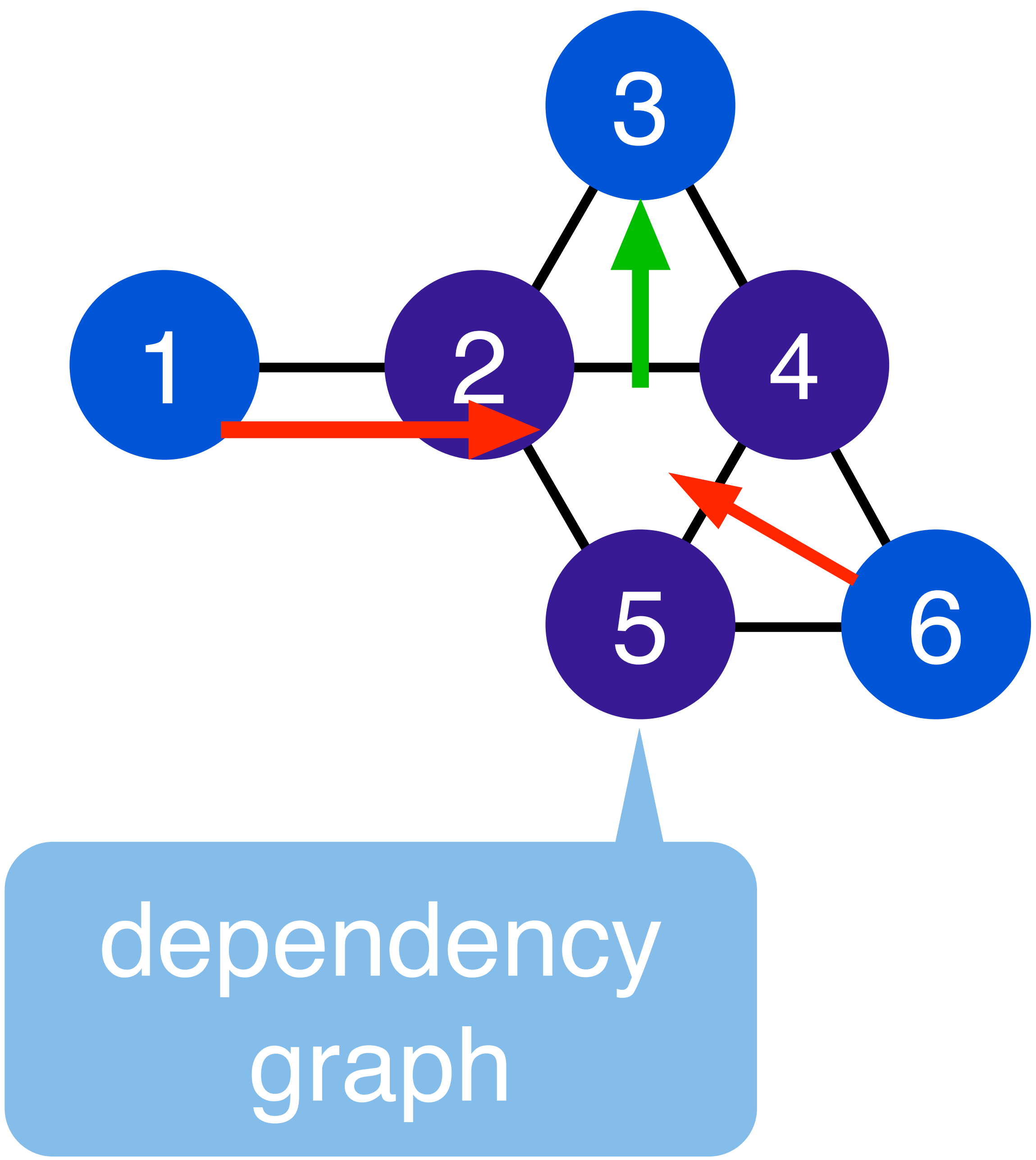


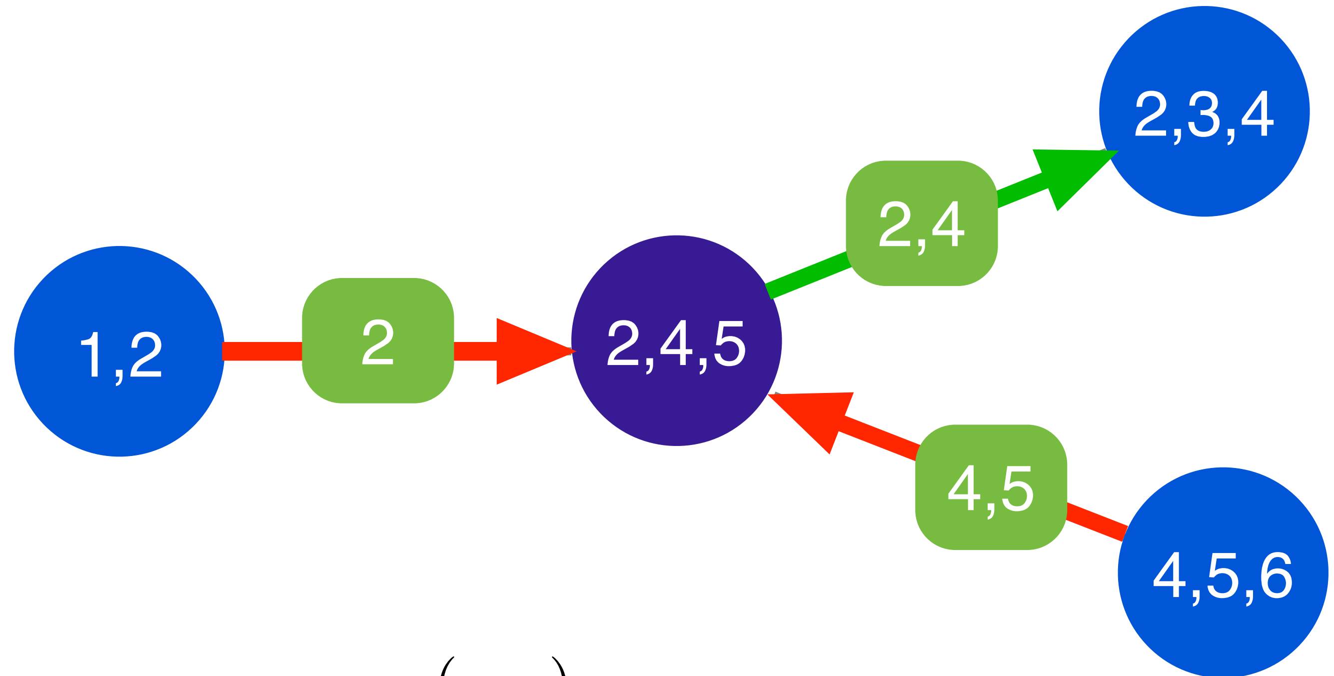
$$m_{245\rightarrow 234}(x_{24}) = \sum_{x_5} f(x_{245}) m_{12\rightarrow 245}(x_2) m_{457\rightarrow 245}(x_{45})$$

clique potential

separator set

Junction Trees



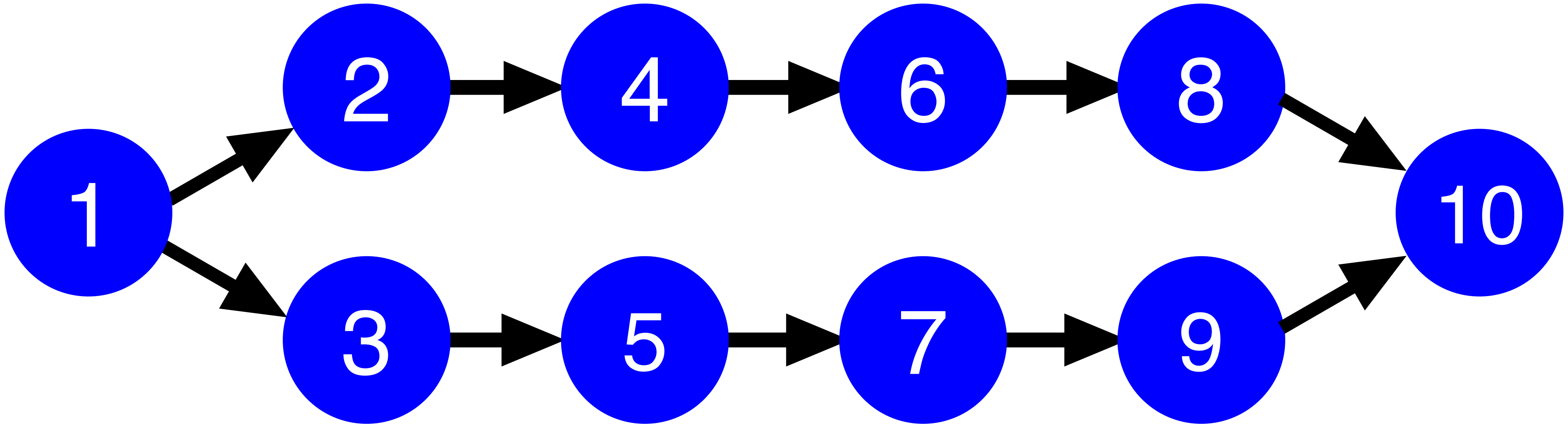


$$m_{245 \to 234}(x_{24}) = \sum_{x_5} f(x_{245}) m_{12 \to 245}(x_2) m_{457 \to 245}(x_{45})$$

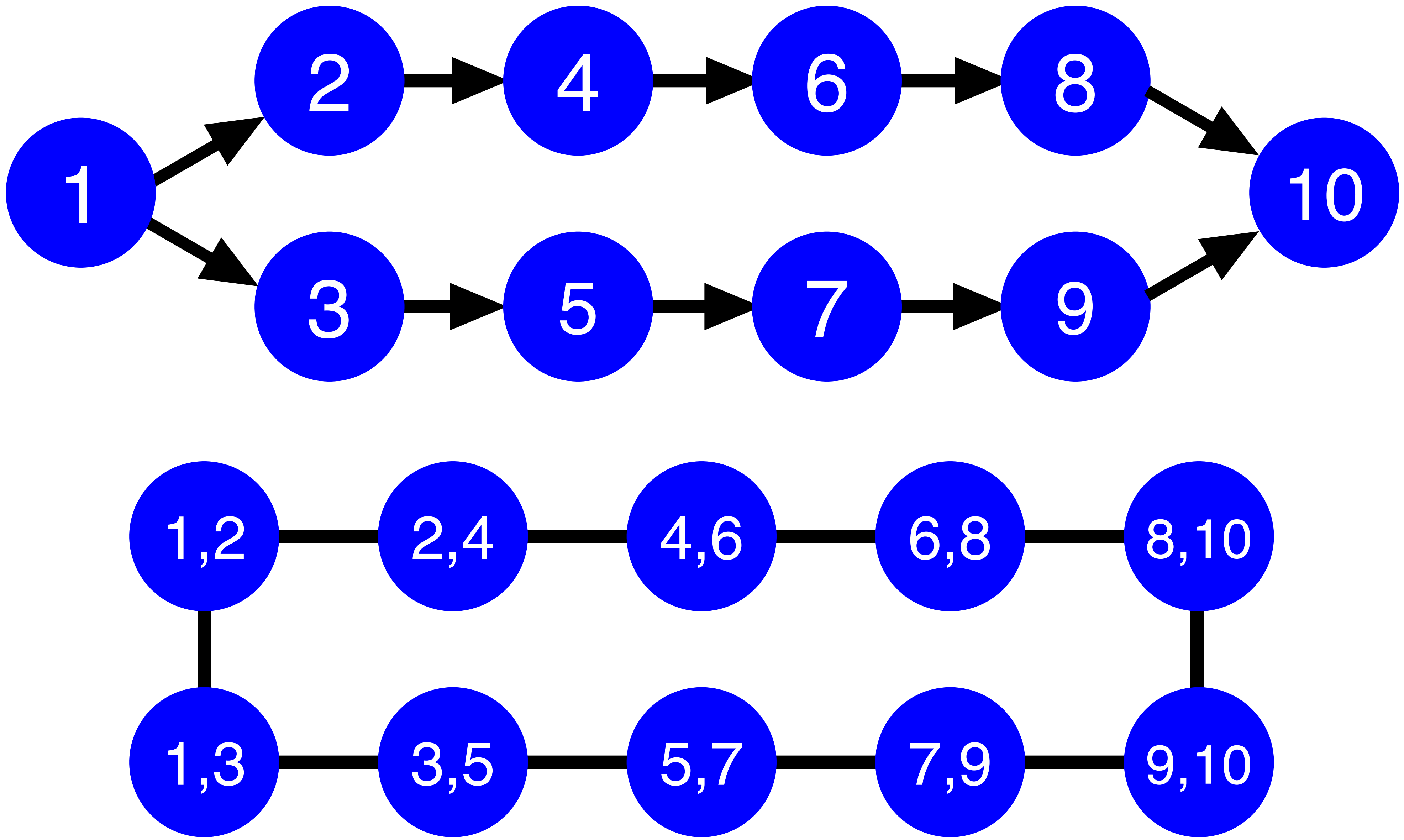
clique potential

separator set

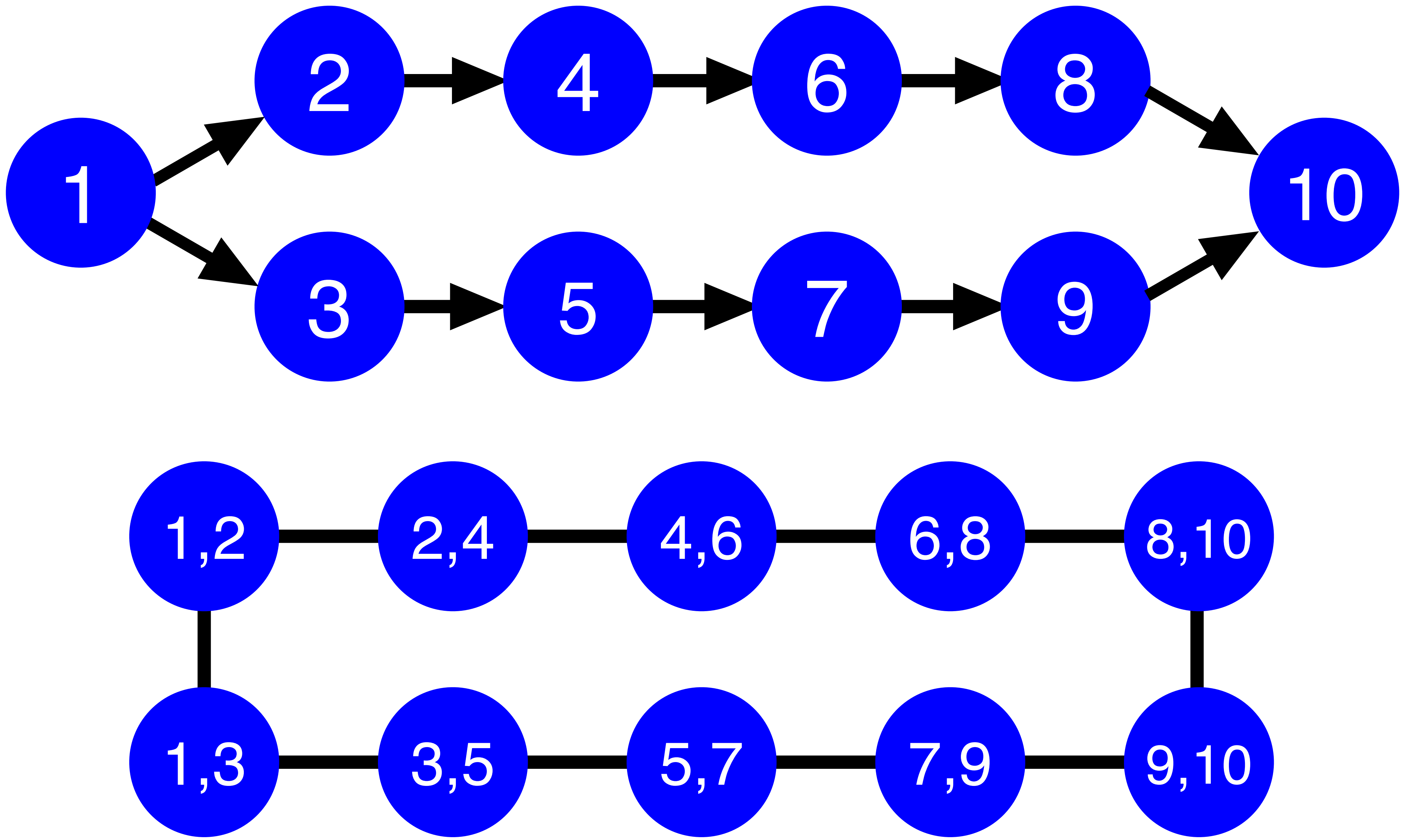
Caution



Caution

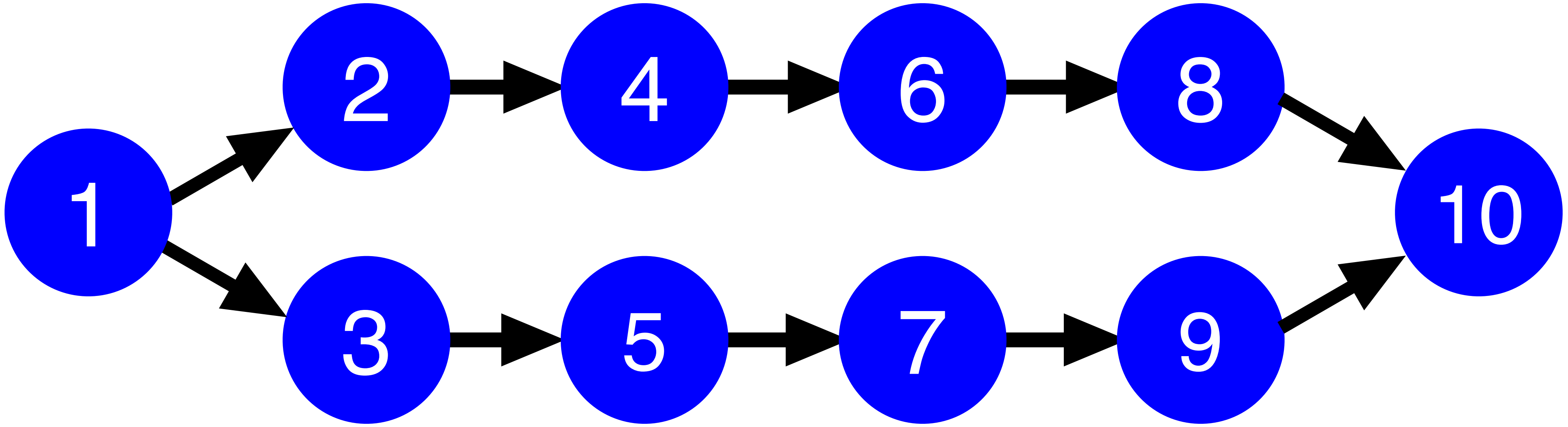


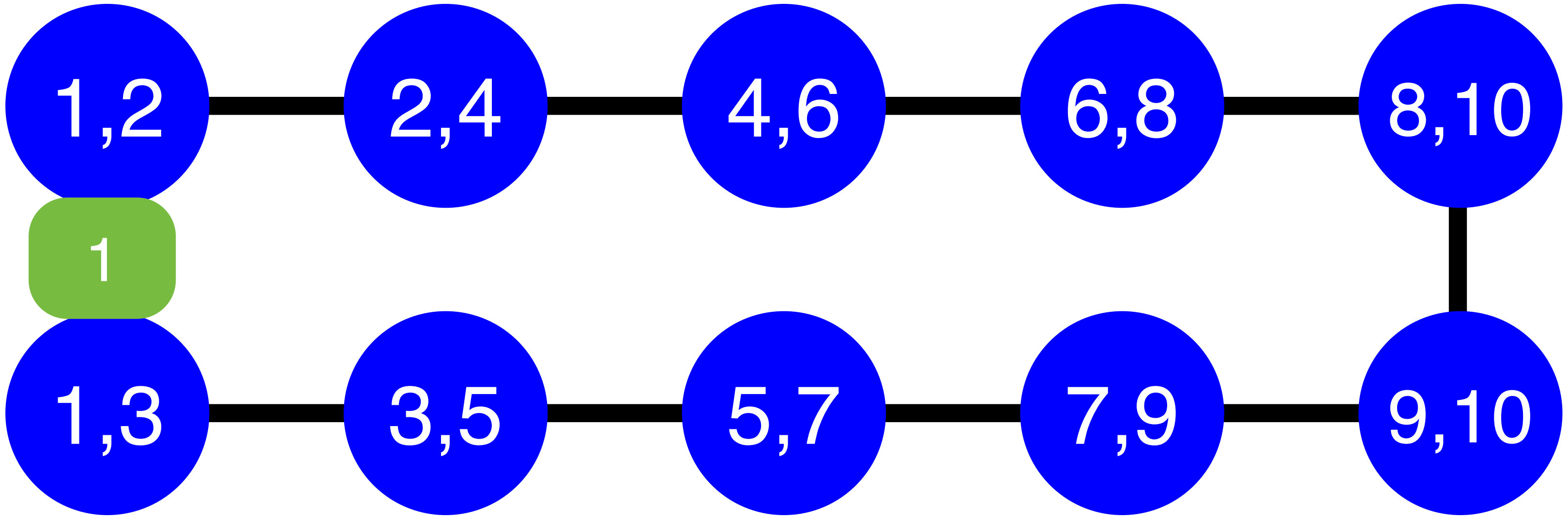
Caution



This is not a tree

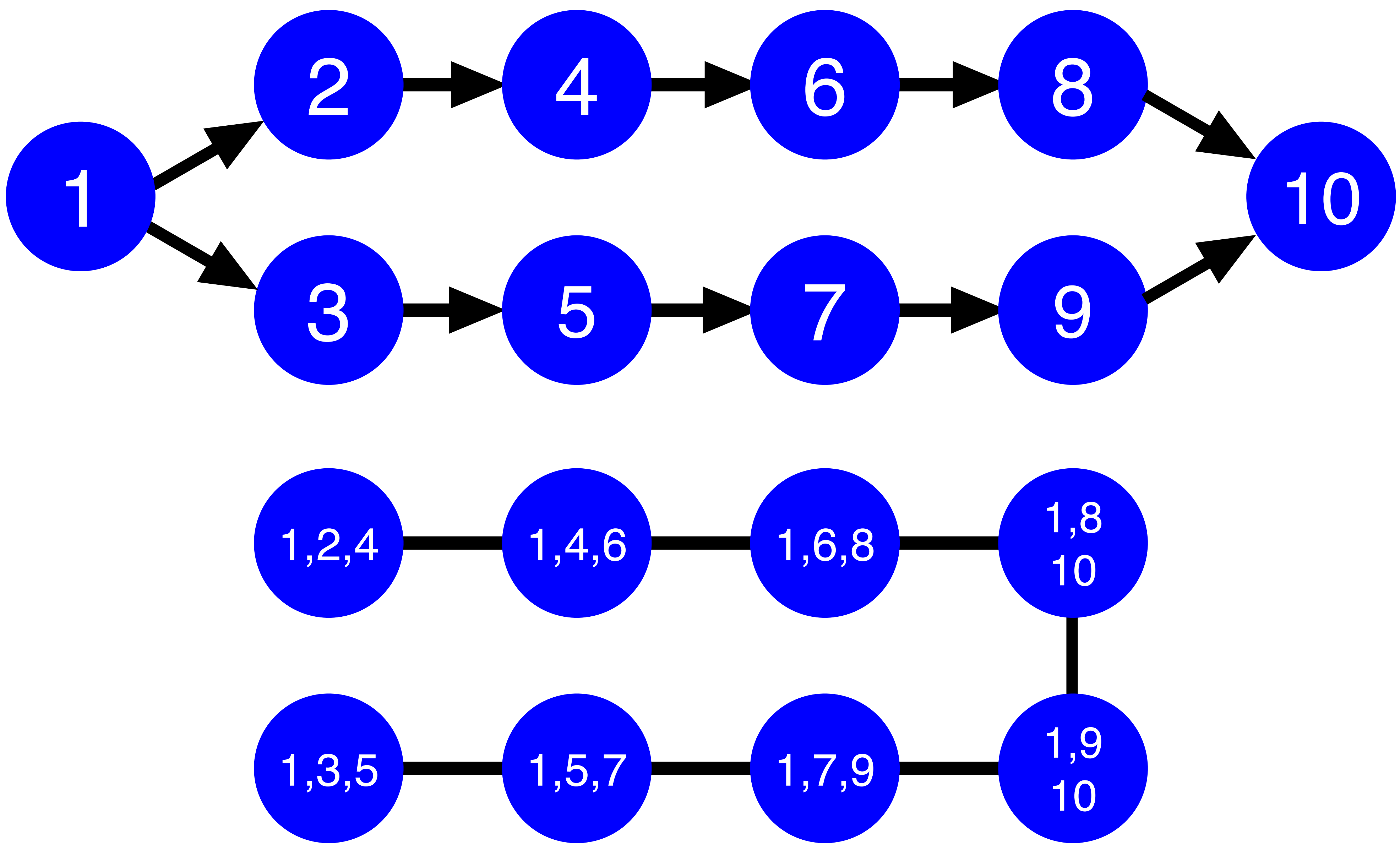
Caution



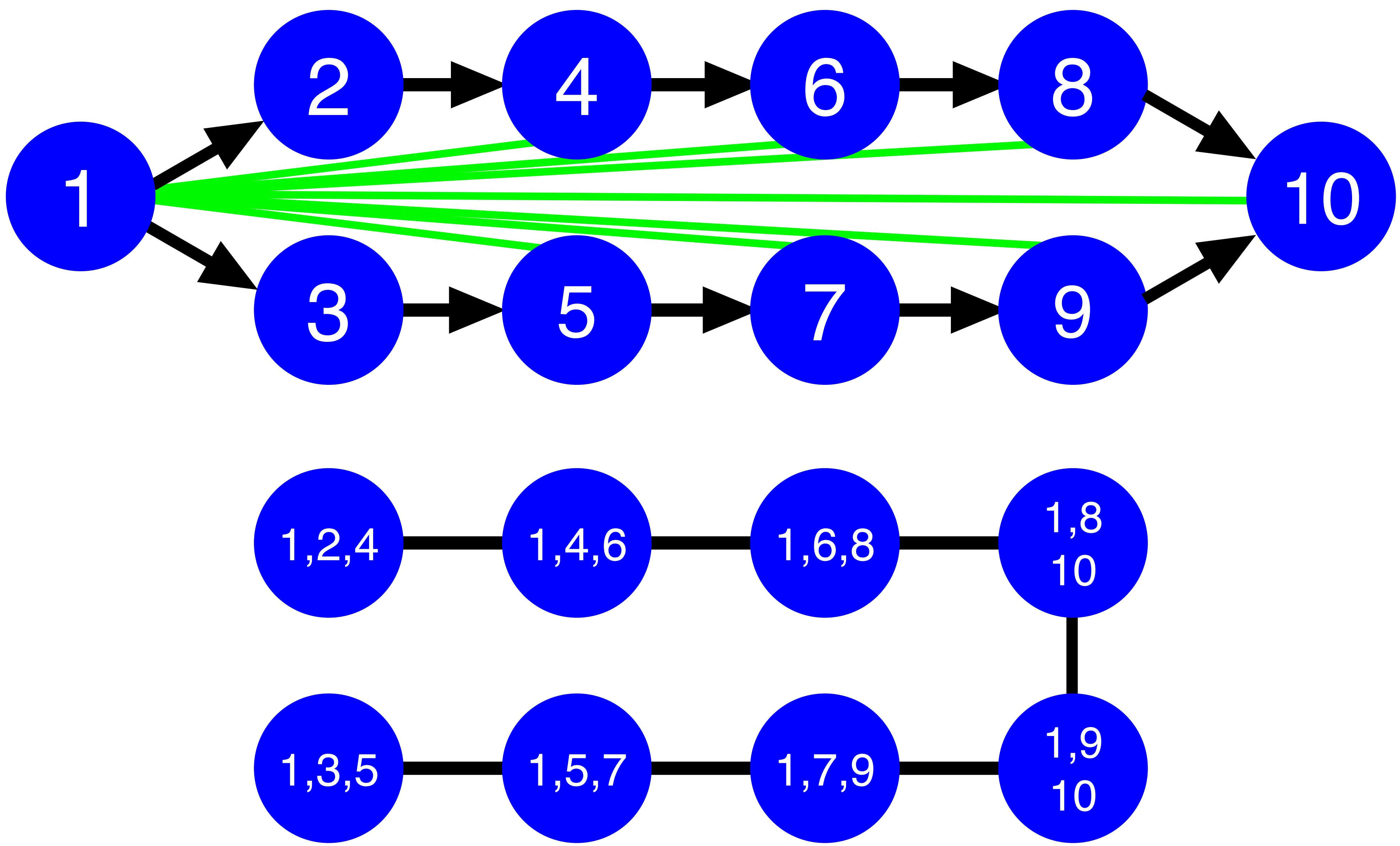


This is not a tree

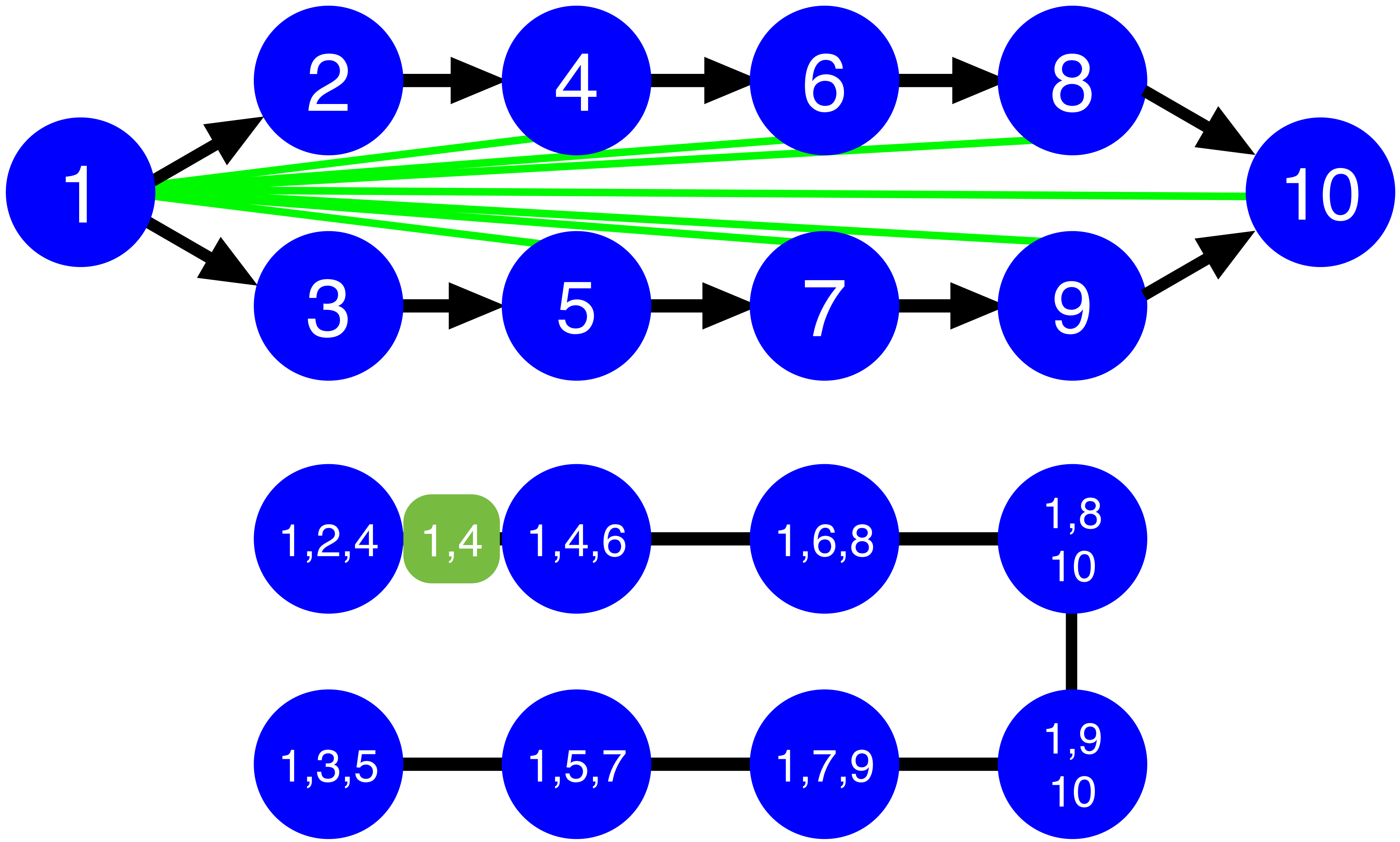
Graph triangulation



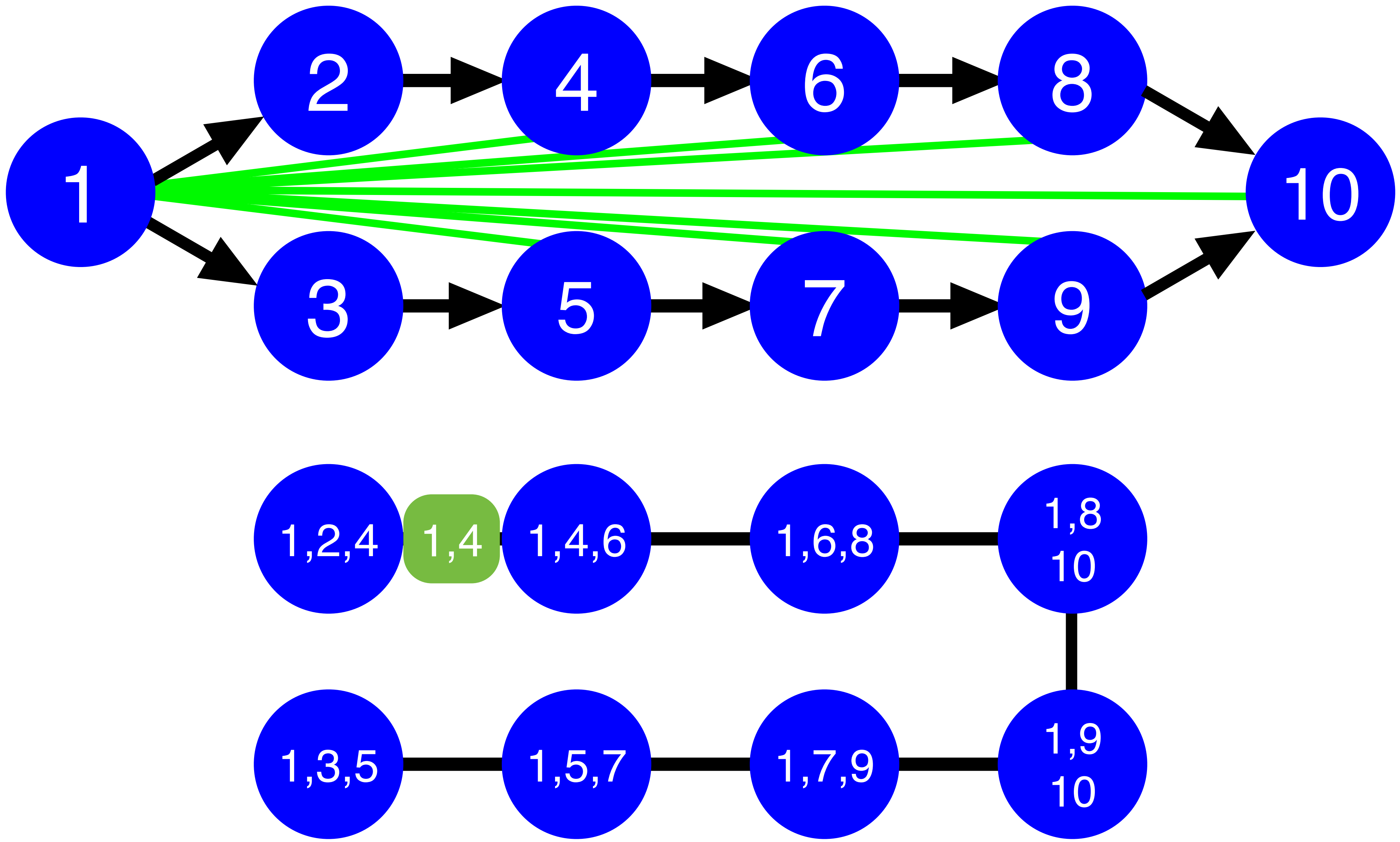
Graph triangulation



Graph triangulation

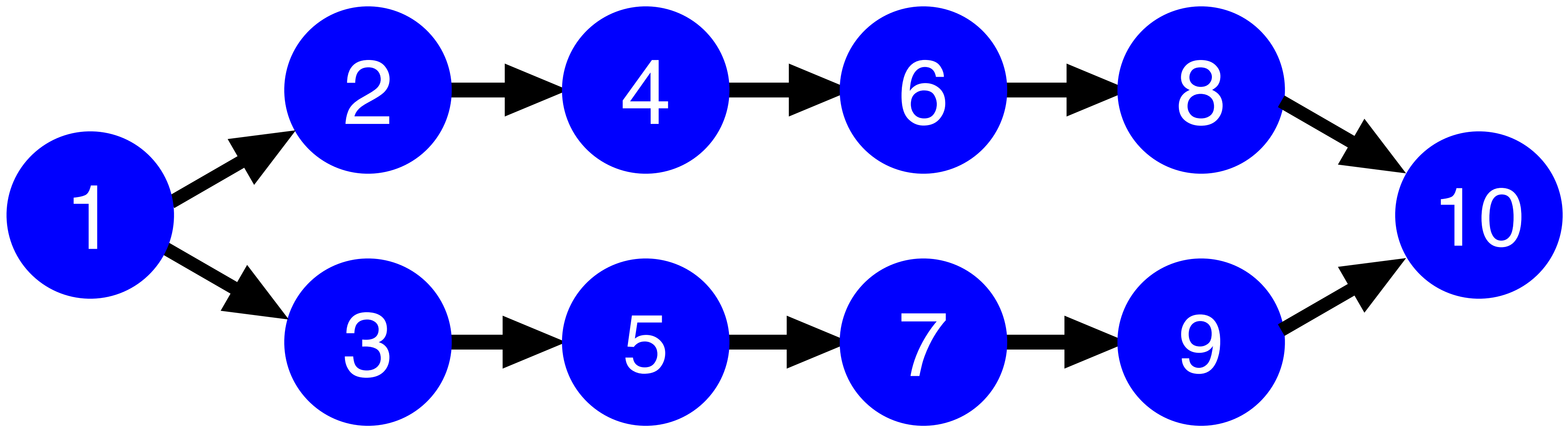


Graph triangulation



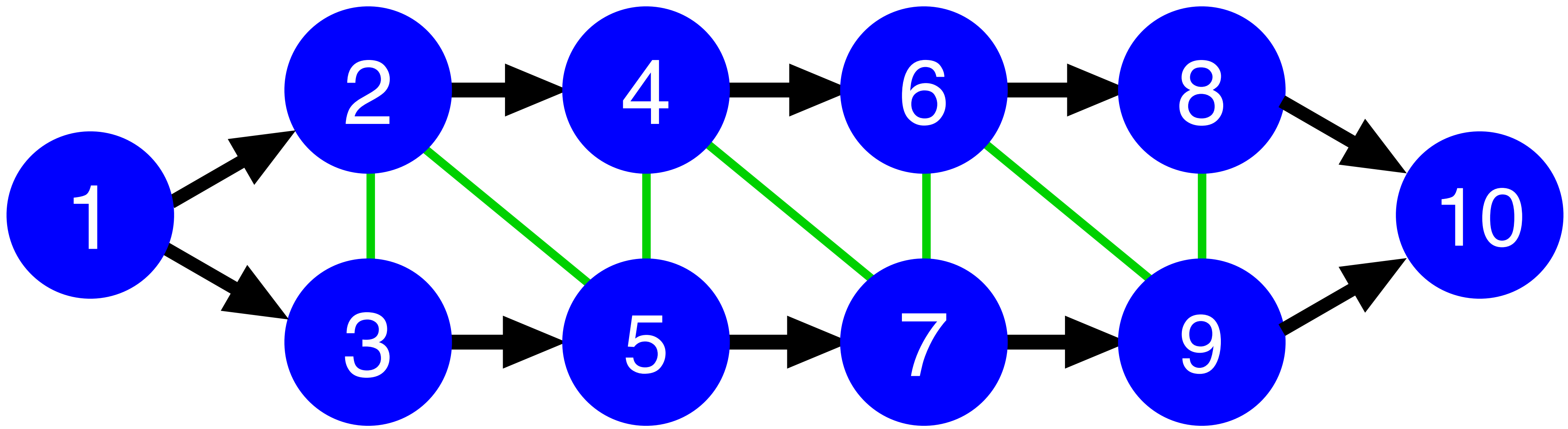
Separator set increases

Graph triangulation



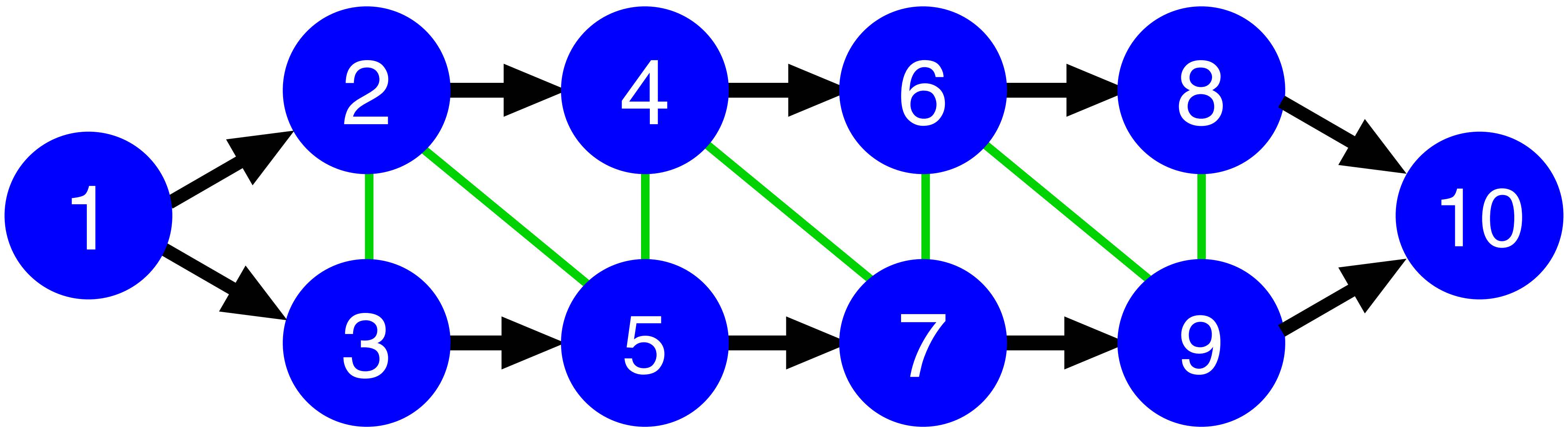
Separator set increases

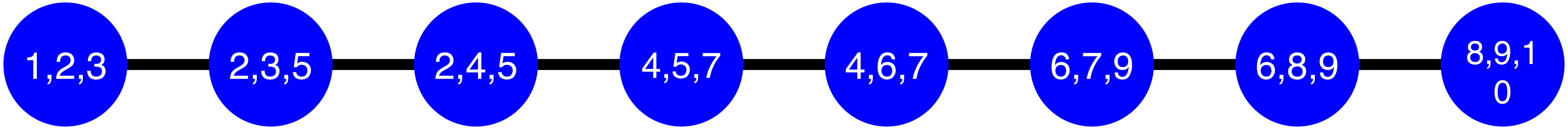
Graph triangulation



Separator set increases

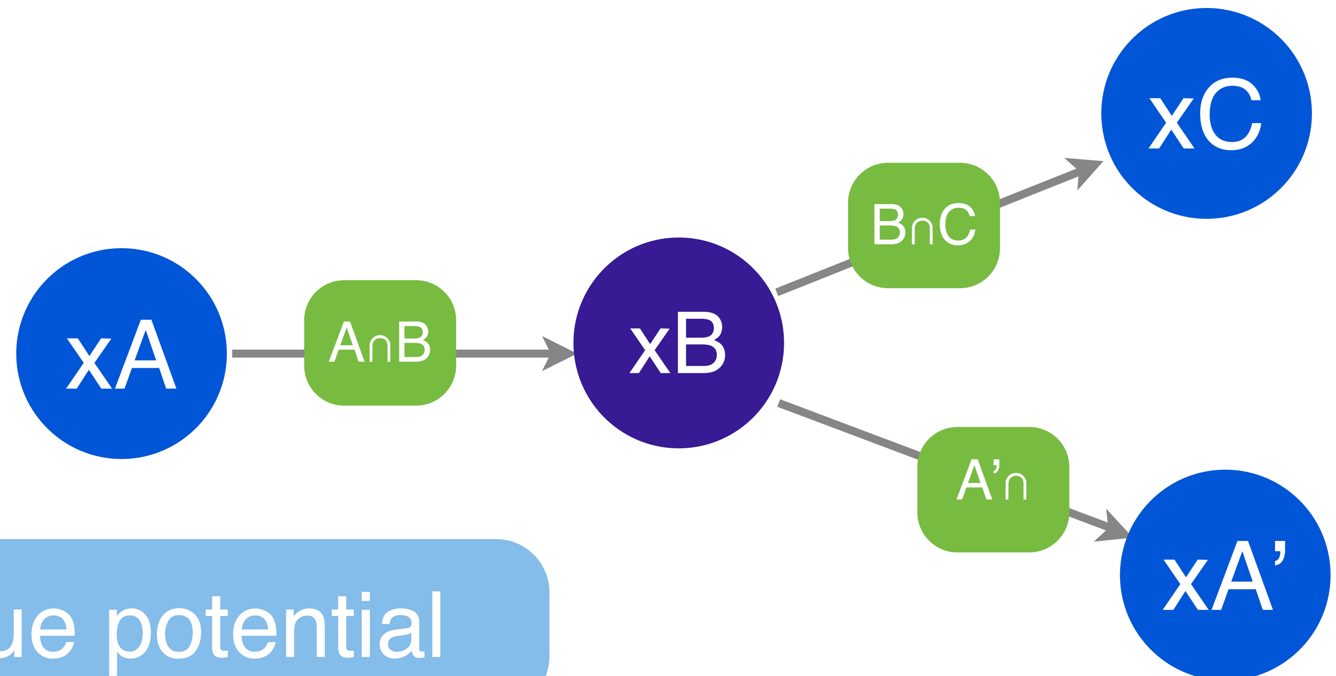
Graph triangulation





Separator set increases

Update equations



clique potential

incoming messages

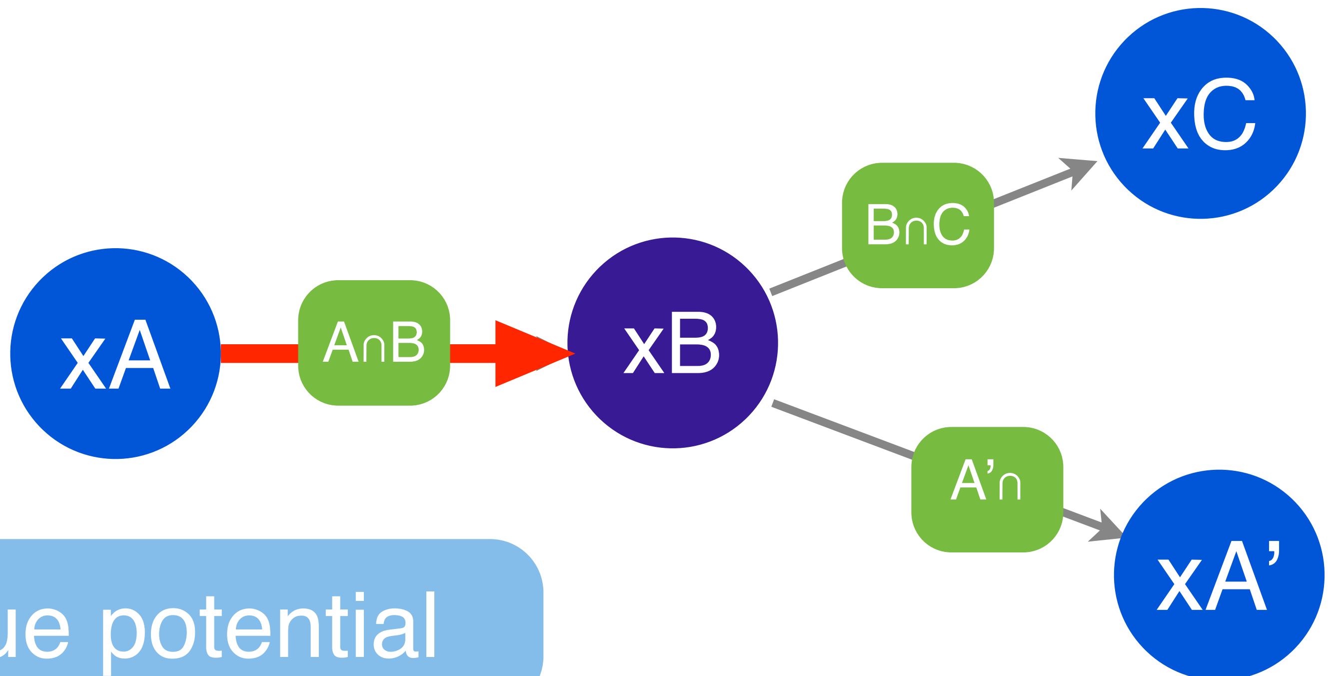
$$m_{B\to C}(x_{B\cap C}) = \sum_{x_{B\setminus C}} f(x_B) \prod_{A\neq C | A\sim B} m_{A\to B}(x_{A\cap B})$$

all but separator set

$$p(x_B) \propto f(x_B) \prod_{A\sim B} m_{A\to B}(x_{A\cap B})$$

unnormalized

Update equations



incoming messages

clique potential

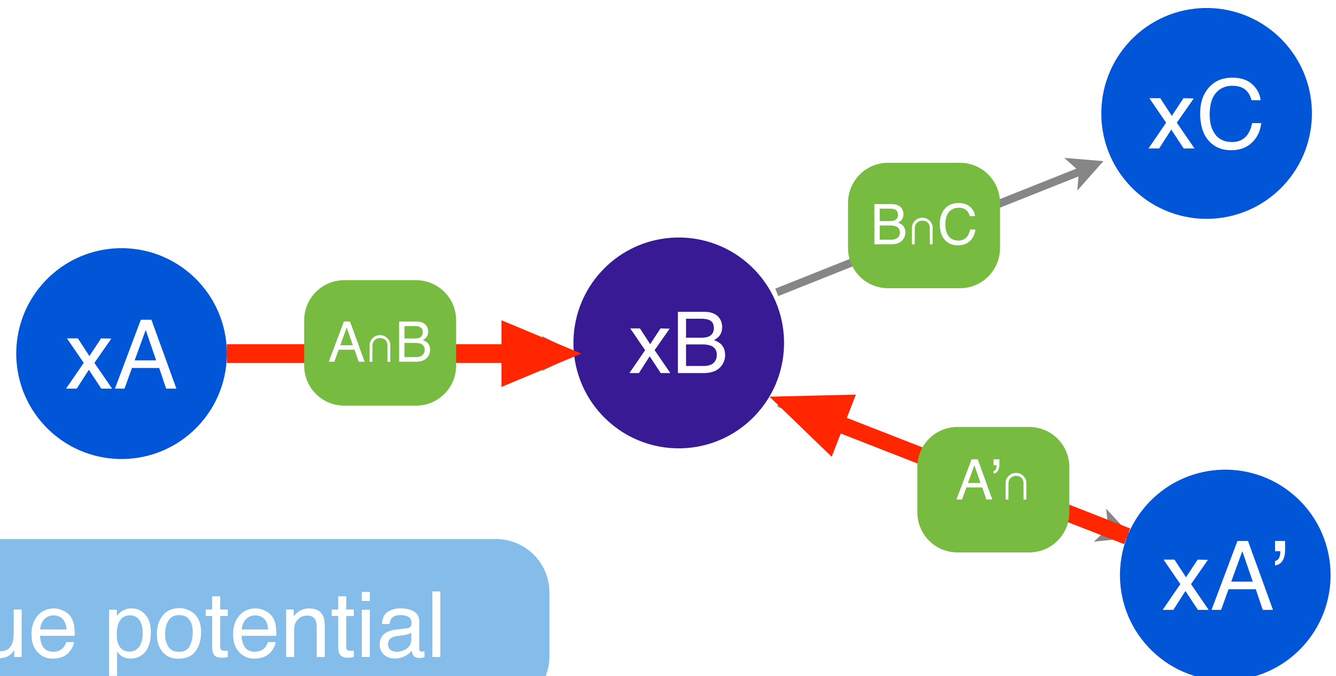
$$m_{B \to C}(x_{B \cap C}) = \sum_{x_{B \setminus C}} f(x_B) \prod_{A \neq C | A \sim B} m_{A \to B}(x_{A \cap B})$$

all but separator set

$$p(x_B) \propto f(x_B) \prod_{A \sim B} m_{A \to B}(x_{A \cap B})$$

unnormalized

Update equations



incoming messages

clique potential

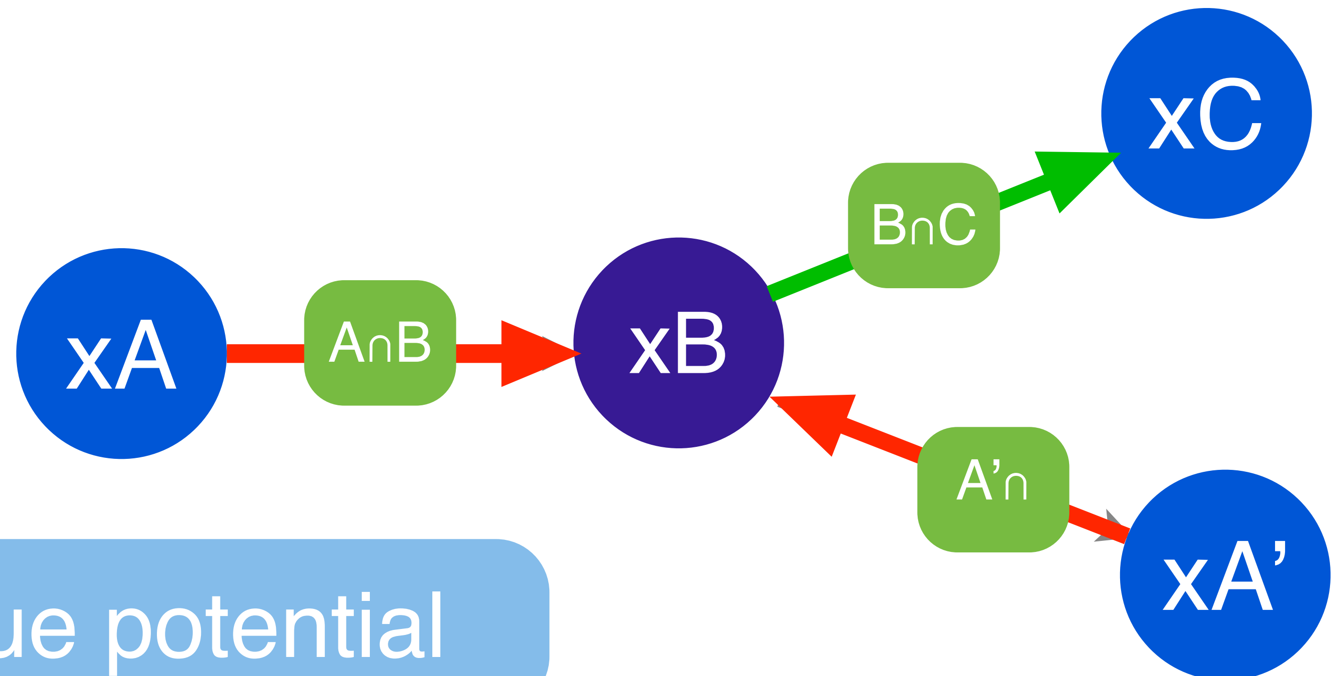
$$m_{B \to C}\left(x_{B \cap C}\right) = \sum_{x_{B \setminus C}} f(x_B) \prod_{A \neq C | A \sim B} m_{A \to B}(x_{A \cap B})$$

all but separator set

$$p(x_B) \propto f(x_B) \prod_{A \sim B} m_{A \to B}(x_{A \cap B})$$

unnormalized

Update equations



incoming messages

clique potential

$$m_{B\to C}(x_{B\cap C}) = \sum_{x_{B\setminus C}} f(x_B) \prod_{A\neq C | A\sim B} m_{A\to B}(x_{A\cap B})$$

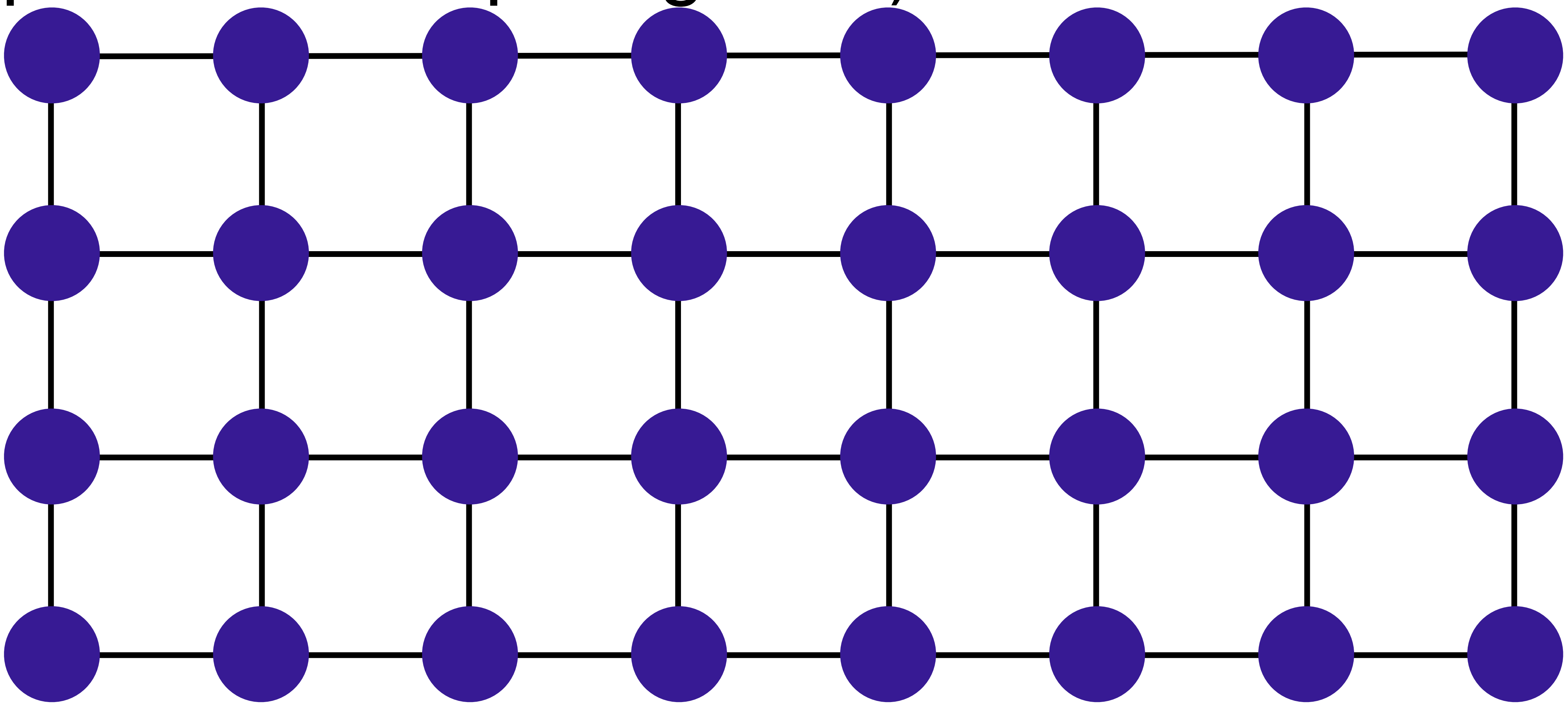
all but separator set

$$p(x_B) \propto f(x_B) \prod_{A\sim B} m_{A\to B}(x_{A\cap B})$$

unnormalized

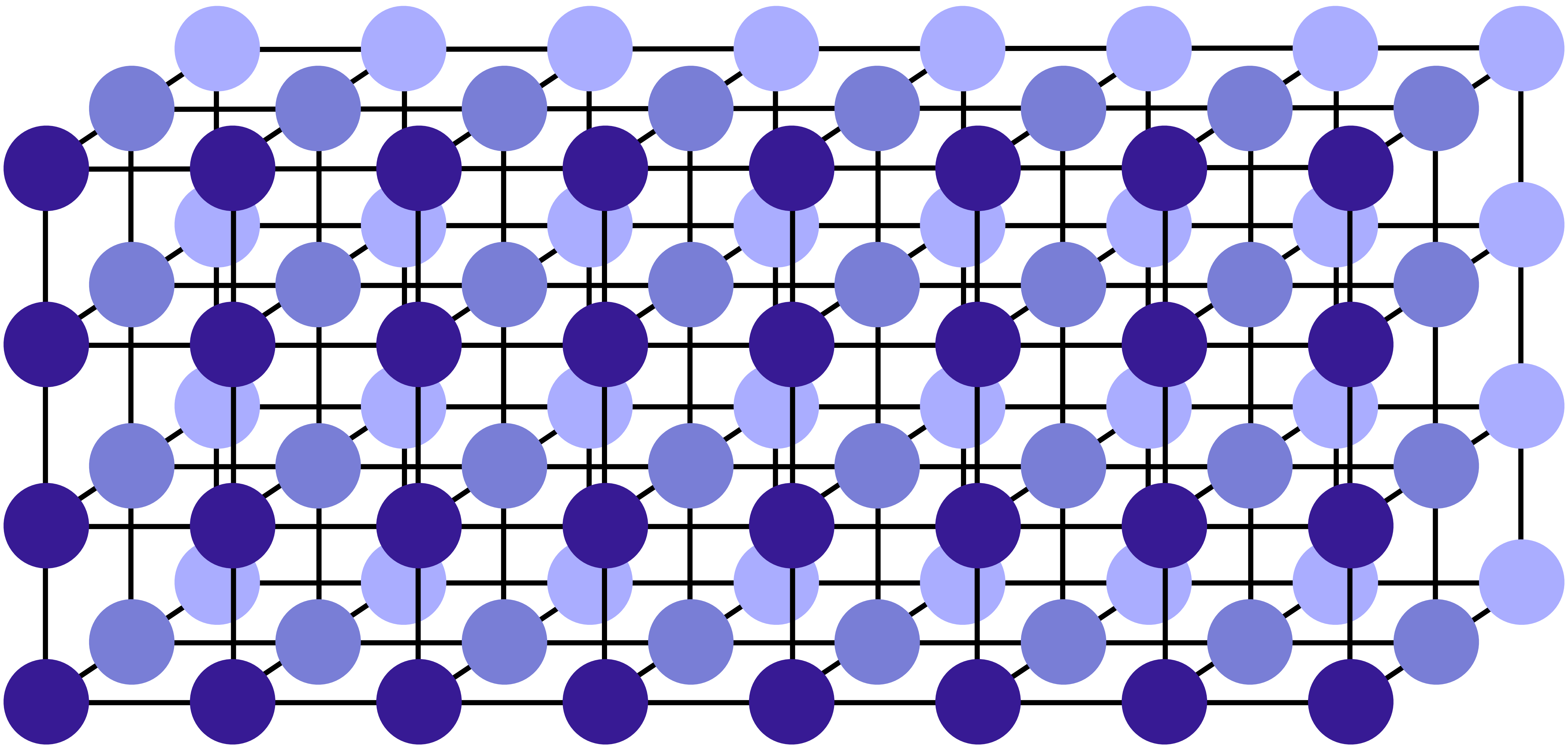
2D grid

- Nontrivial to generate junction tree (problem clumps together)



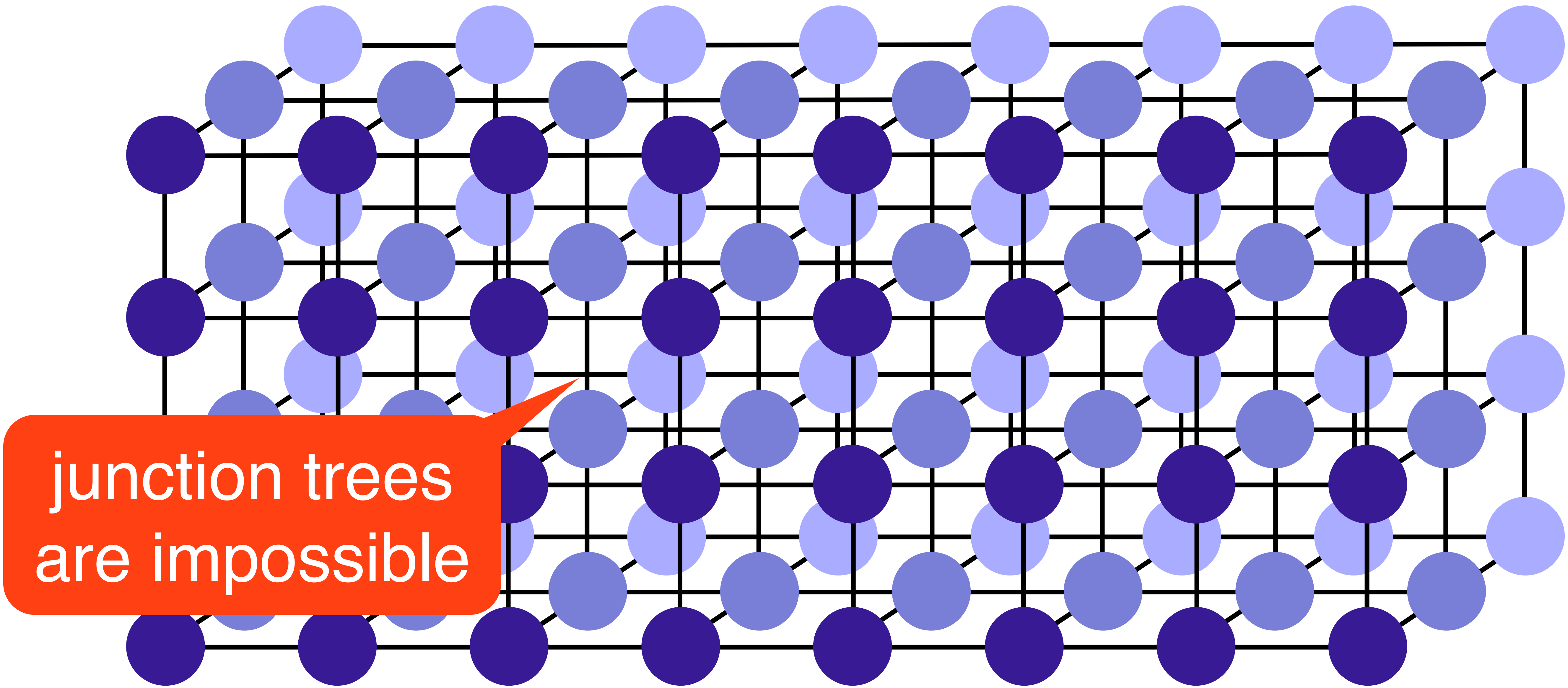
images

3D grid



movies, CAT scans

3D grid



movies, CAT scans

Summary

- (Directed) graphical model
- Build clique graph
  - Luck if it's a tree
  - If not, need to add edges to make it a tree
  - Tree width increases
  - In many realistic cases exact inference is not possible - need approximation techniques.
- Same operations as for tree.
  Just now with more variables


Generalized Distributive Law

Recall - dynamic programming

$$
\begin{aligned}
p(x_i|x_n) &= l_i(x_i) \sum_{x_{i+1}\dots x_{n-1}} \prod_{j=i}^{n-1} p(x_{j+1}|x_j) \\
&= l_i(x_i) \sum_{x_{i+1}\dots x_{n-1}} \prod_{j=i}^{n-2} p(x_{j+1}|x_j) \underbrace{p(x_n|x_{n-1})}_{:=r_{n-1}(x_{n-1})} \\
&= l_i(x_i) \sum_{x_{i+1}\dots x_{n-2}} \prod_{j=i}^{n-3} p(x_{j+1}|x_j) \underbrace{\sum_{x_{n-1}} p(x_{n-1}|x_{n-2}) r_{n-1}(x_{n-1})}_{:=r_{n-2}(x_{n-2})}
\end{aligned}
$$

- The reason for efficient computation is the fact that we can swap multiplication and addition.
- Are there other such pairs?

Generalized Distributive Law

- Dynamic programming uses only additions and multiplications,
- Replace them with equivalent operations from other **semirings**

- Semiring
  - ‘addition’ and ‘summation’ equivalent
  - Associative law $(a+b)+c = a+(b+c)$
  - Distributive law $a(b+c) = ab+ac$

Generalized Distributive Law

- Integrating out probabilities (sum, product)

$$a \cdot (b + c) = a \cdot b + a \cdot c$$

- Finding the maximum (max, +)

$$a + \max(b, c) = \max(a + b, a + c)$$

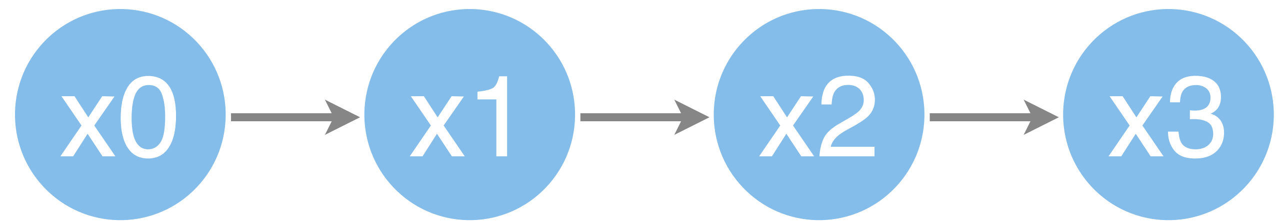
- Set algebra (union, intersection)

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

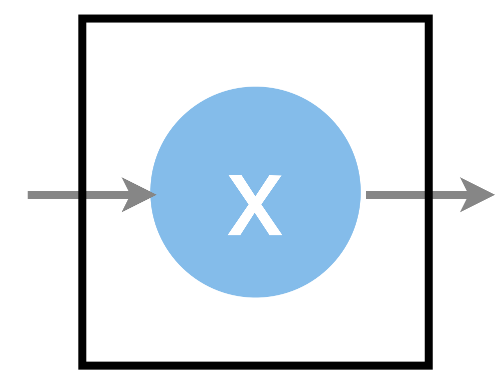
- Boolean semiring (AND, OR)
- Probability semiring (log +, +)
- Tropical semiring (min, +)

Chains ... again

$$\bar{s} = \max_x s(x_0) + \sum_{i=1}^{n-1} s(x_{i+1}|x_i)$$



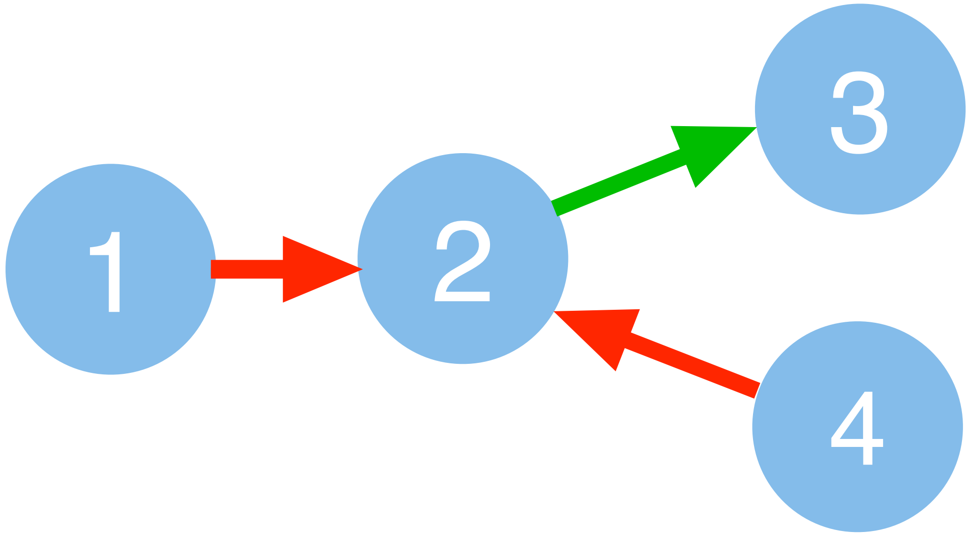
$$\bar{s} = \max_{x_{0\ldots n}} \underbrace{s(x_0)}_{:=l_0(x_0)} + \sum_{j=1}^{n} s(x_j|x_{j-1})$$



$$= \max_{x_{1\ldots n}} \underbrace{\max_{x_0} [l_0(x_0)s(x_1|x_0)]}_{:=l_1(x_1)} + \sum_{j=2}^{n} s(x_j|x_{j-1})$$

$$= \max_{x_{2\ldots n}} \underbrace{\max_{x_1} [l_1(x_1)s(x_2|x_1)]}_{:=l_2(x_2)} + \sum_{j=3}^{n} s(x_j|x_{j-1})$$

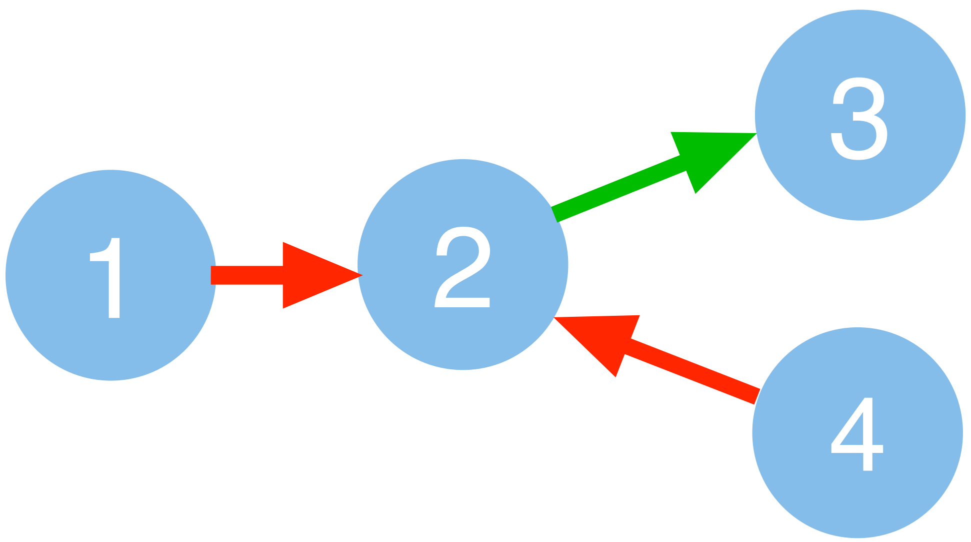
Junction Trees

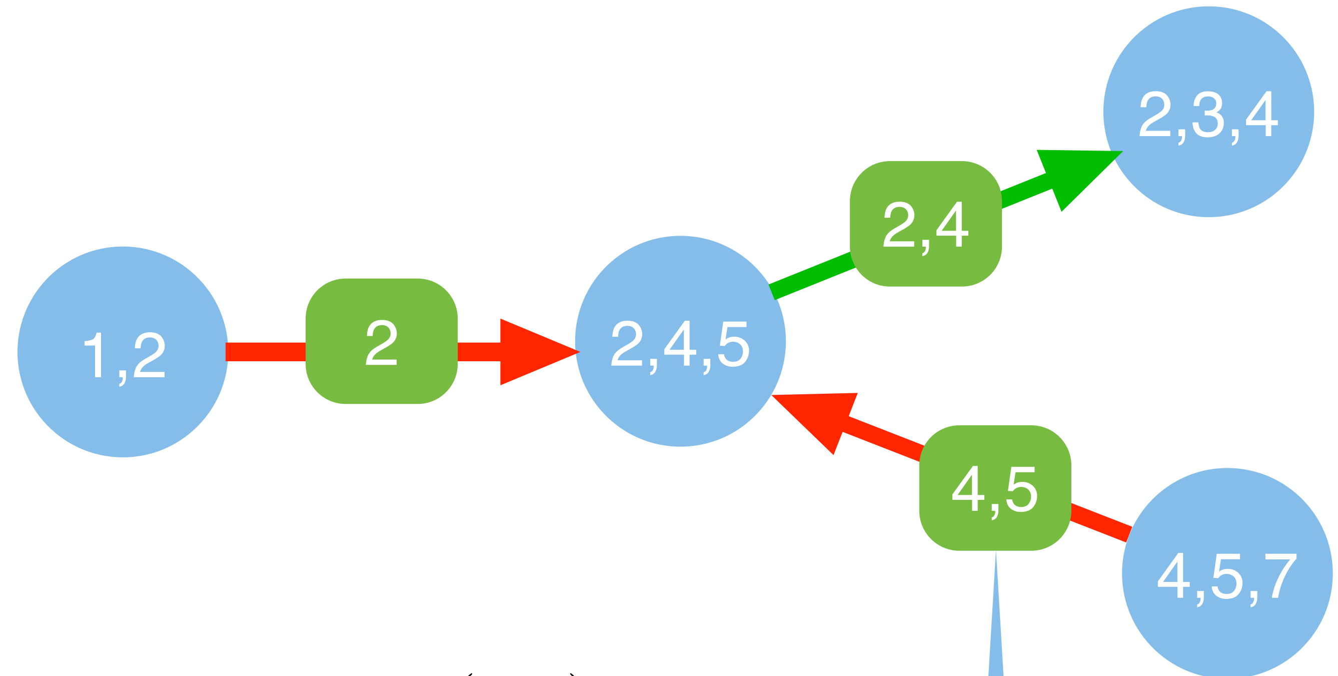


$$m_{i\to j}(x_j) = \max_{x_i} f(x_i, x_j) + \sum_{l\neq j} m_{l\to i}(x_j)$$

clique potential

Junction Trees





$$m_{i \to j}(x_j) = \max_{x_i} f(x_i, x_j) + \sum_{l \neq j} m_{l \to i}(x_j)$$

clique potential

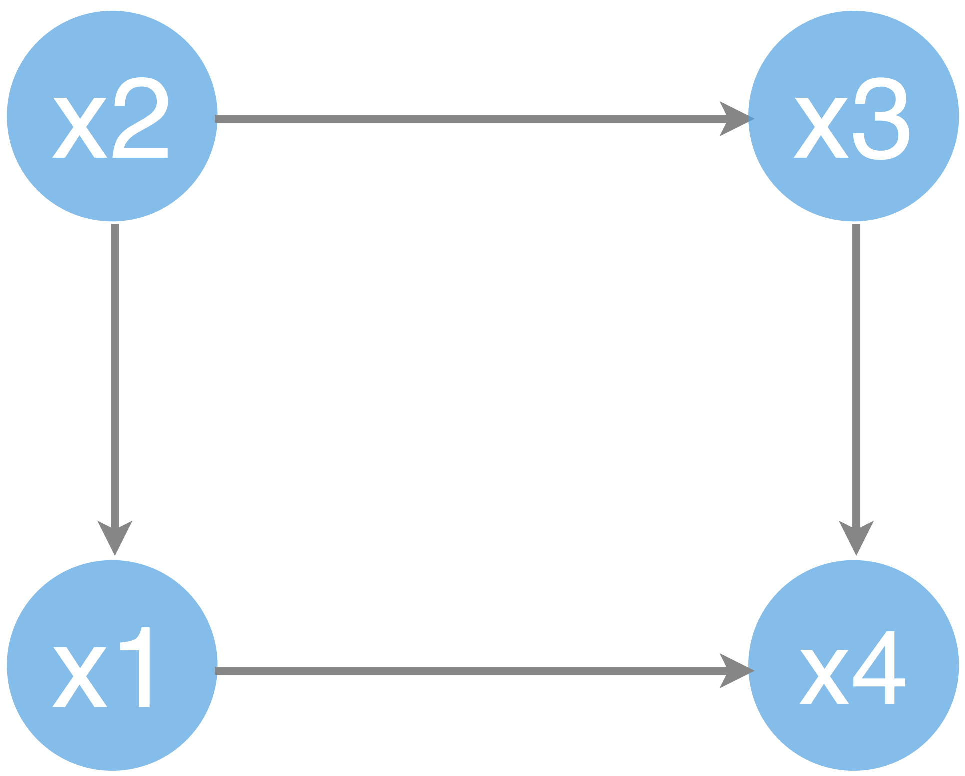
$$m_{245 \to 234}(x_{24})$$
$$= \max_{x_5} f(x_{245}) + m_{12 \to 245}(x_2) + m_{457 \to 245}(x_{45})$$

clique potential

separator set

No loops allowed

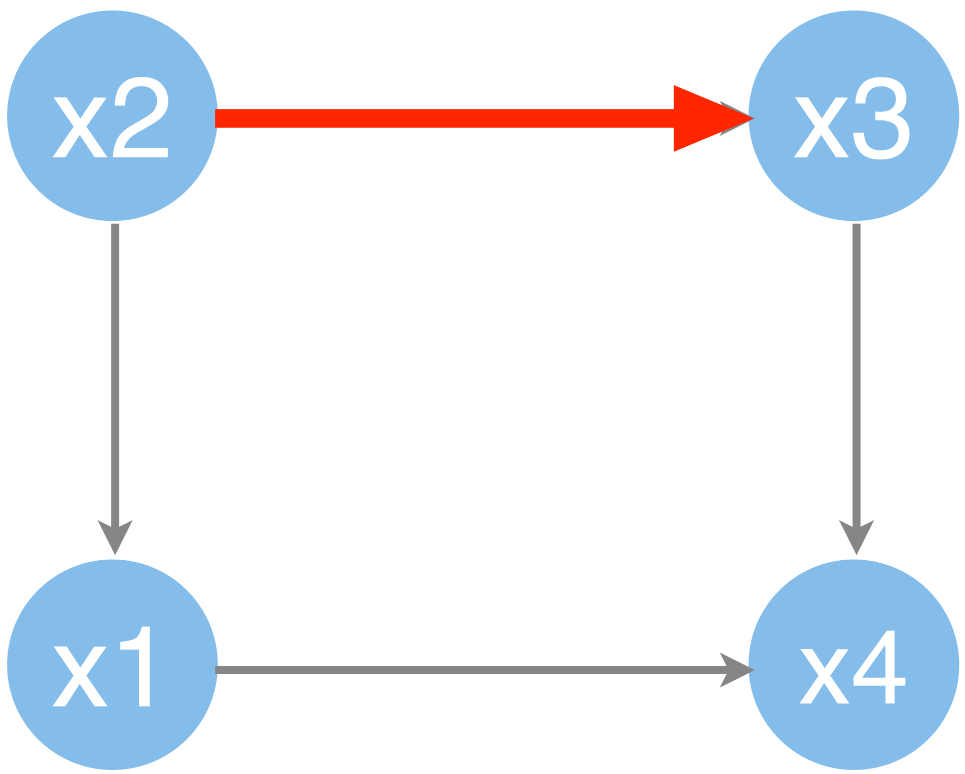
$$s(x_1, x_2) + s(x_2, x_3) + s(x_3, x_4) + s(x_4, x_1)$$



Often use it anyway --- Loopy Belief Propagation
(Turbo Codes, Markov Random Fields, etc.)

No loops allowed

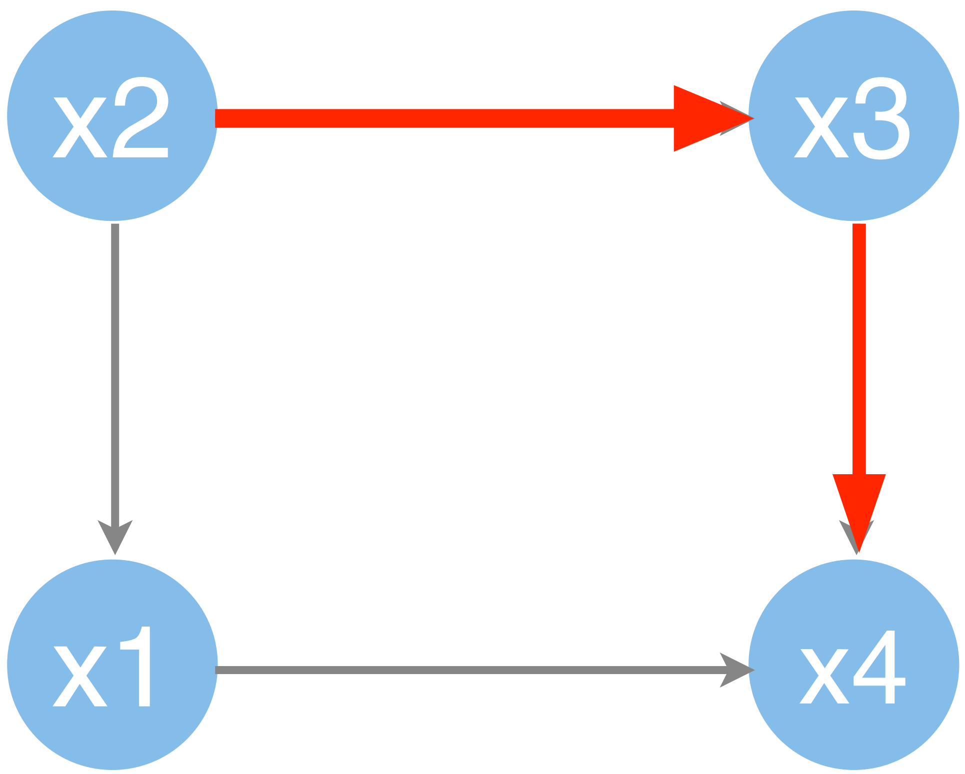
$$s(x_1, x_2) + s(x_2, x_3) + s(x_3, x_4) + s(x_4, x_1)$$



Often use it anyway --- Loopy Belief Propagation
(Turbo Codes, Markov Random Fields, etc.)

No loops allowed

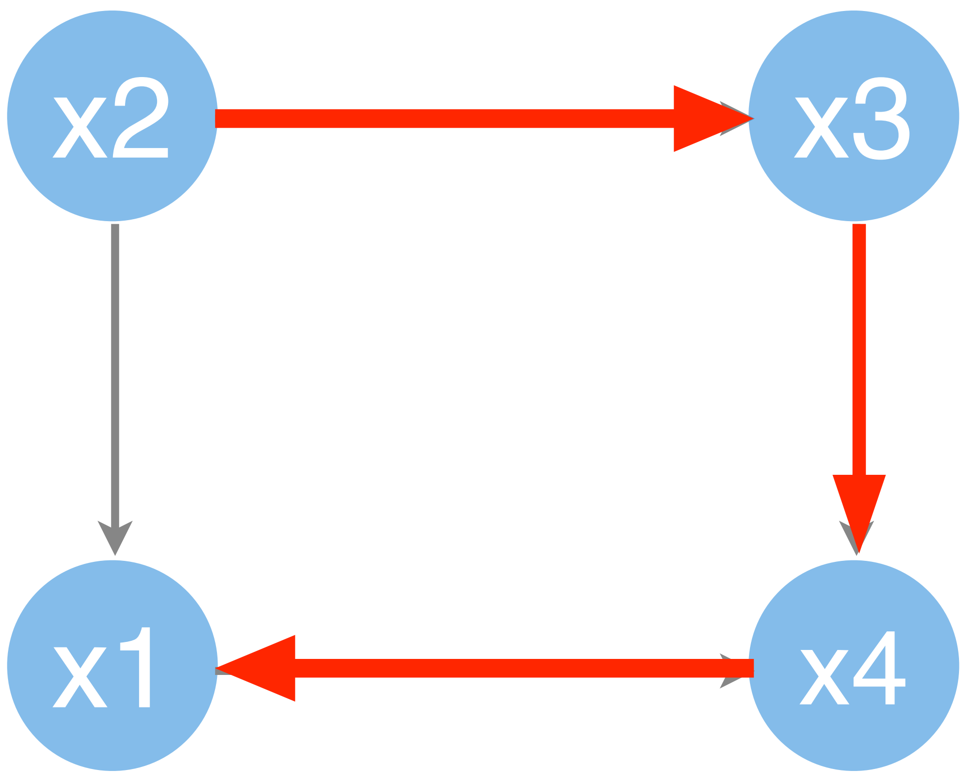
$$s(x_1, x_2) + s(x_2, x_3) + s(x_3, x_4) + s(x_4, x_1)$$



Often use it anyway --- Loopy Belief Propagation
(Turbo Codes, Markov Random Fields, etc.)

No loops allowed

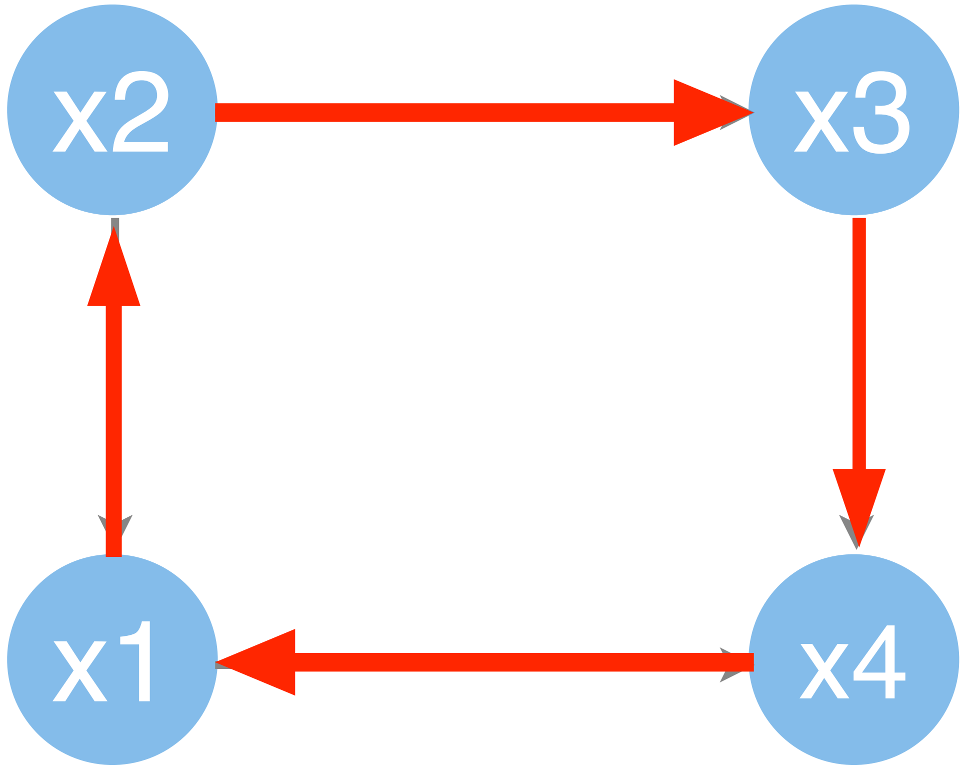
$$s(x_1, x_2) + s(x_2, x_3) + s(x_3, x_4) + s(x_4, x_1)$$



Often use it anyway --- Loopy Belief Propagation
(Turbo Codes, Markov Random Fields, etc.)

No loops allowed

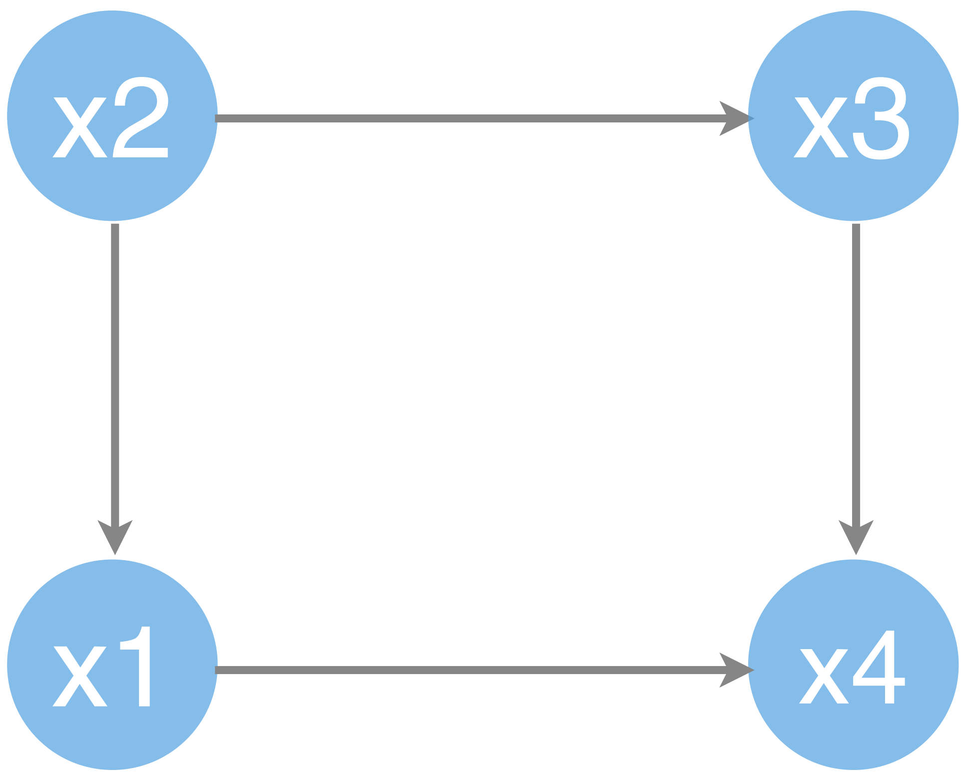
$$s(x_1, x_2) + s(x_2, x_3) + s(x_3, x_4) + s(x_4, x_1)$$



Often use it anyway --- Loopy Belief Propagation
(Turbo Codes, Markov Random Fields, etc.)

No loops allowed

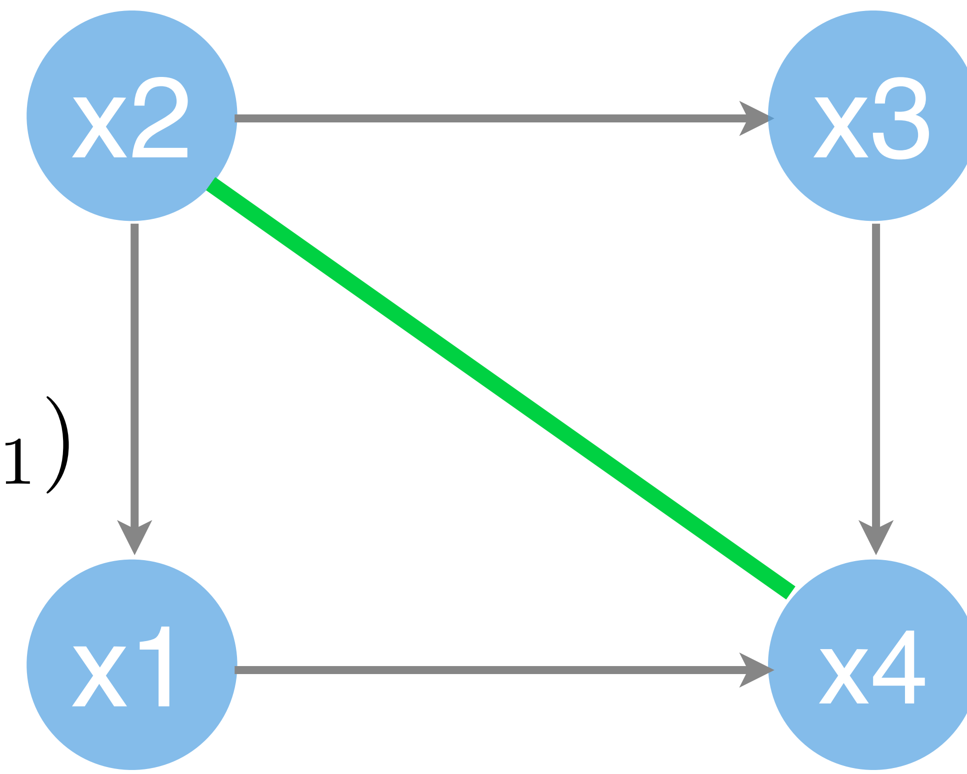
$$s(x_1, x_2) + s(x_2, x_3) + s(x_3, x_4) + s(x_4, x_1)$$



Often use it anyway --- Loopy Belief Propagation
(Turbo Codes, Markov Random Fields, etc.)

No loops allowed

$$s(x_1, x_2) + s(x_2, x_3) + s(x_3, x_4) + s(x_4, x_1)$$



$$s(x_2, x_3) + s(x_3, x_4)$$

$$s(x_1, x_2) + s(x_4, x_1)$$

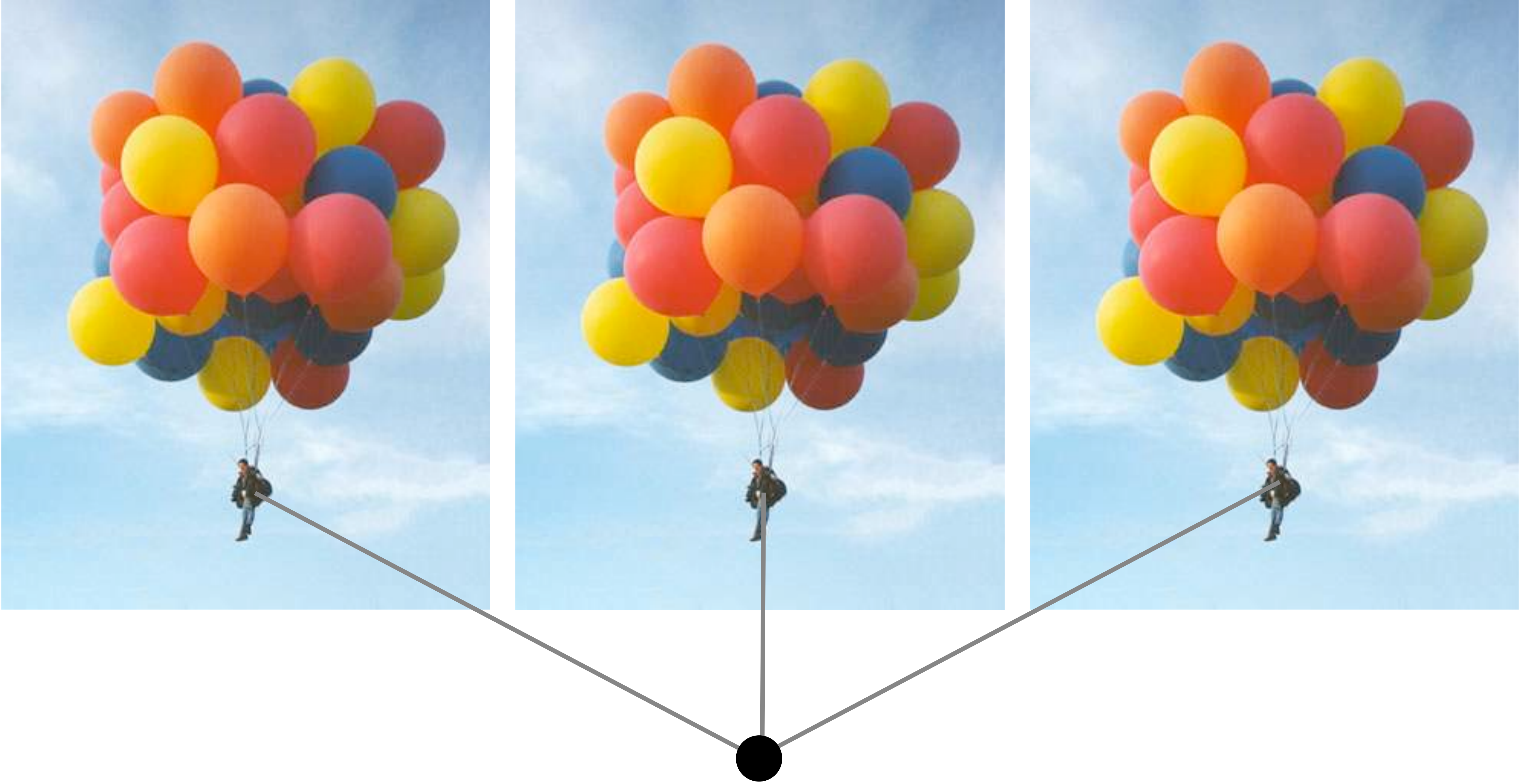
Often use it anyway --- Loopy Belief Propagation
(Turbo Codes, Markov Random Fields, etc.)

7.3 Practical Inference

7 Graphical Models

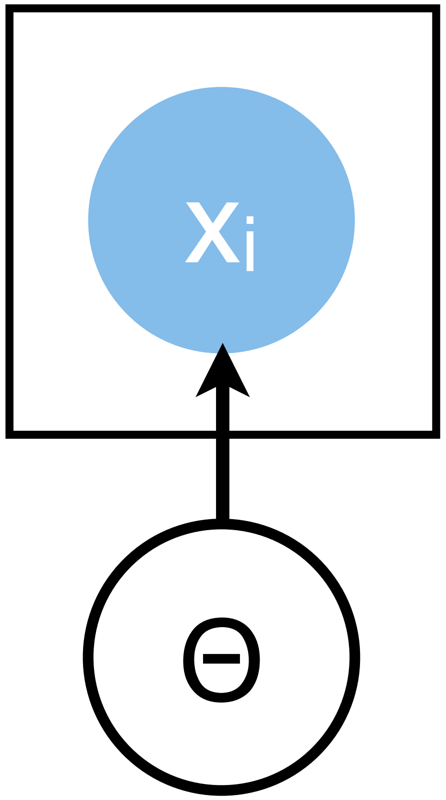
Alexander Smola
Introduction to Machine Learning 10-701
http://alex.smola.org/teaching/10-701-15

MACHINE LEARNING DEPARTMENT
Carnegie Mellon University



Clustering

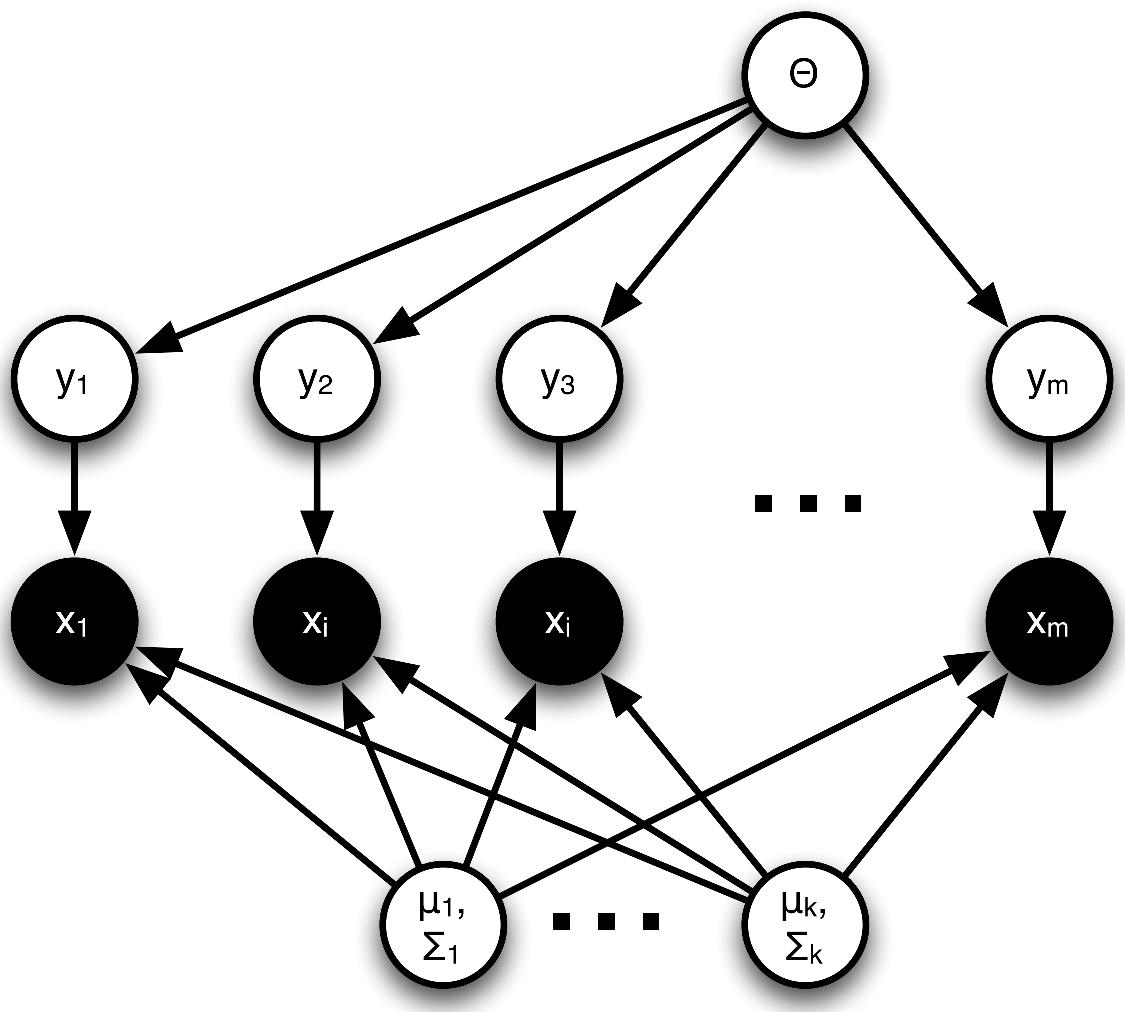
Density Estimation



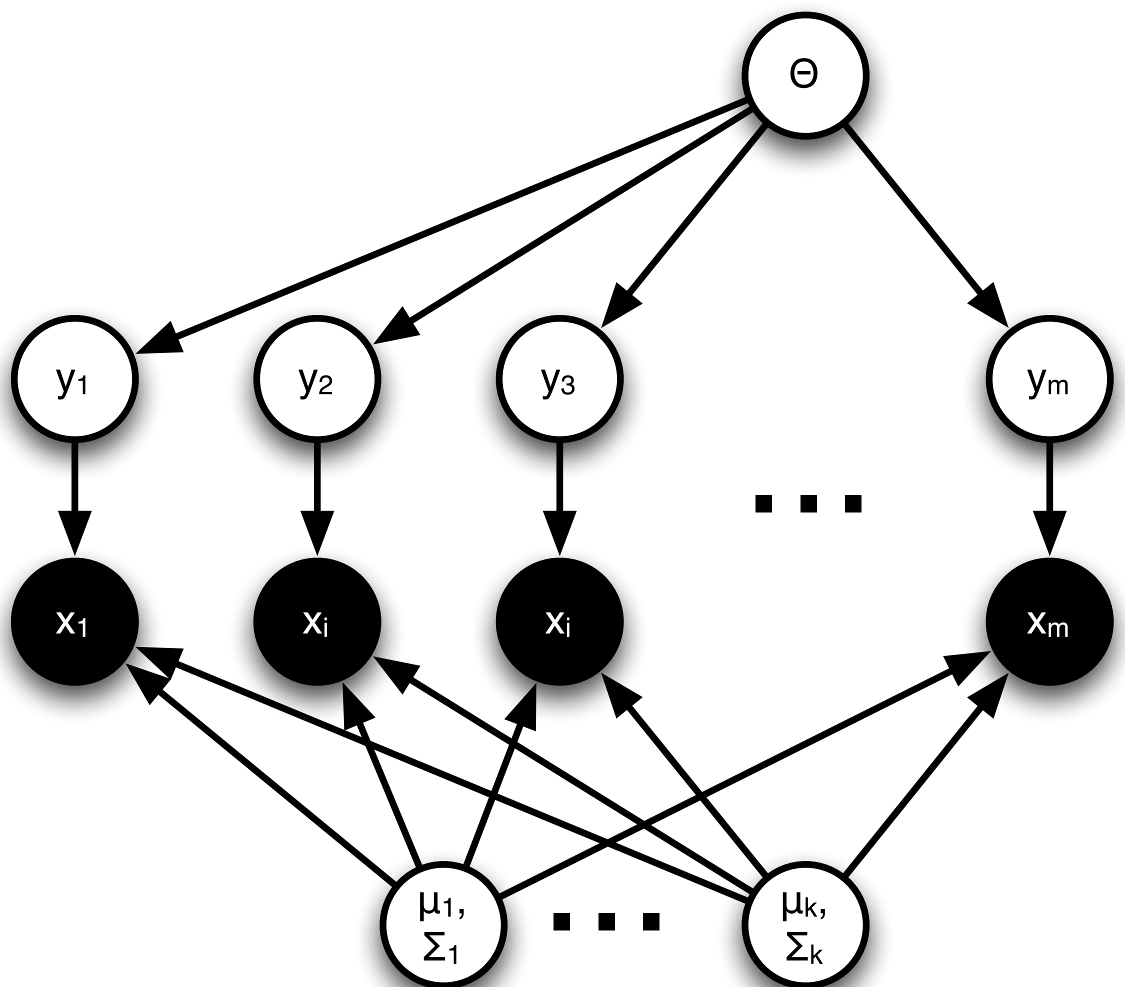
$$p(X|\theta) = \prod_{i=1}^{m} p(x_i|\theta)$$

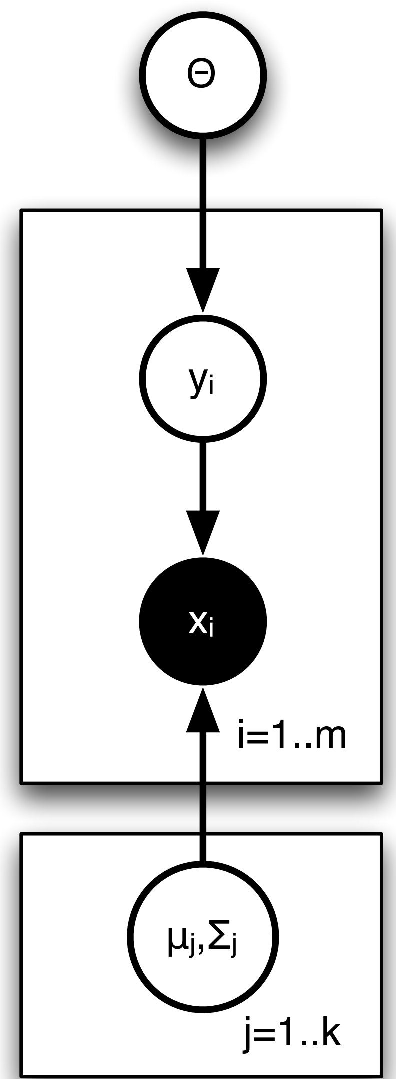
- Draw latent parameter Θ
- For all i draw observed x_i given Θ
- What if the basic model doesn't fit all data?

One size doesn't fit all

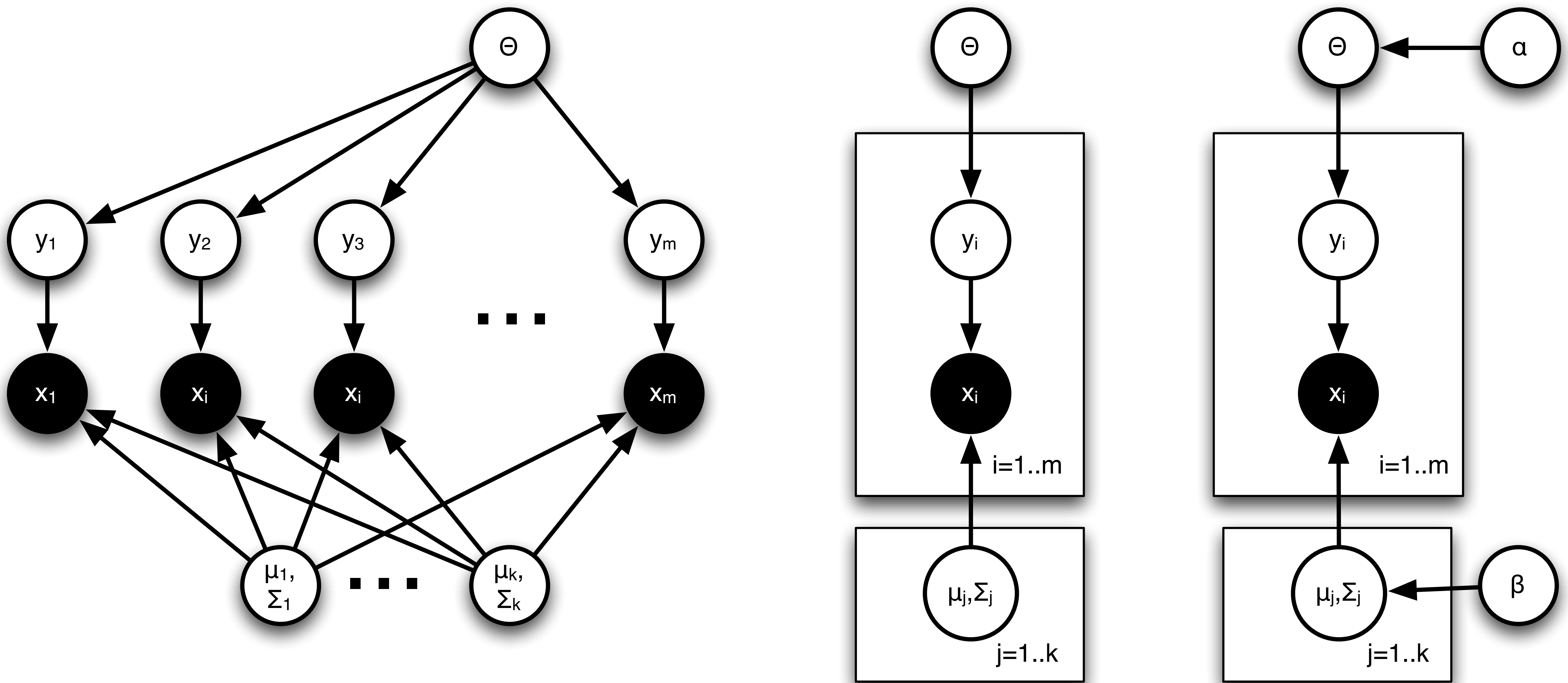


One size doesn't fit all

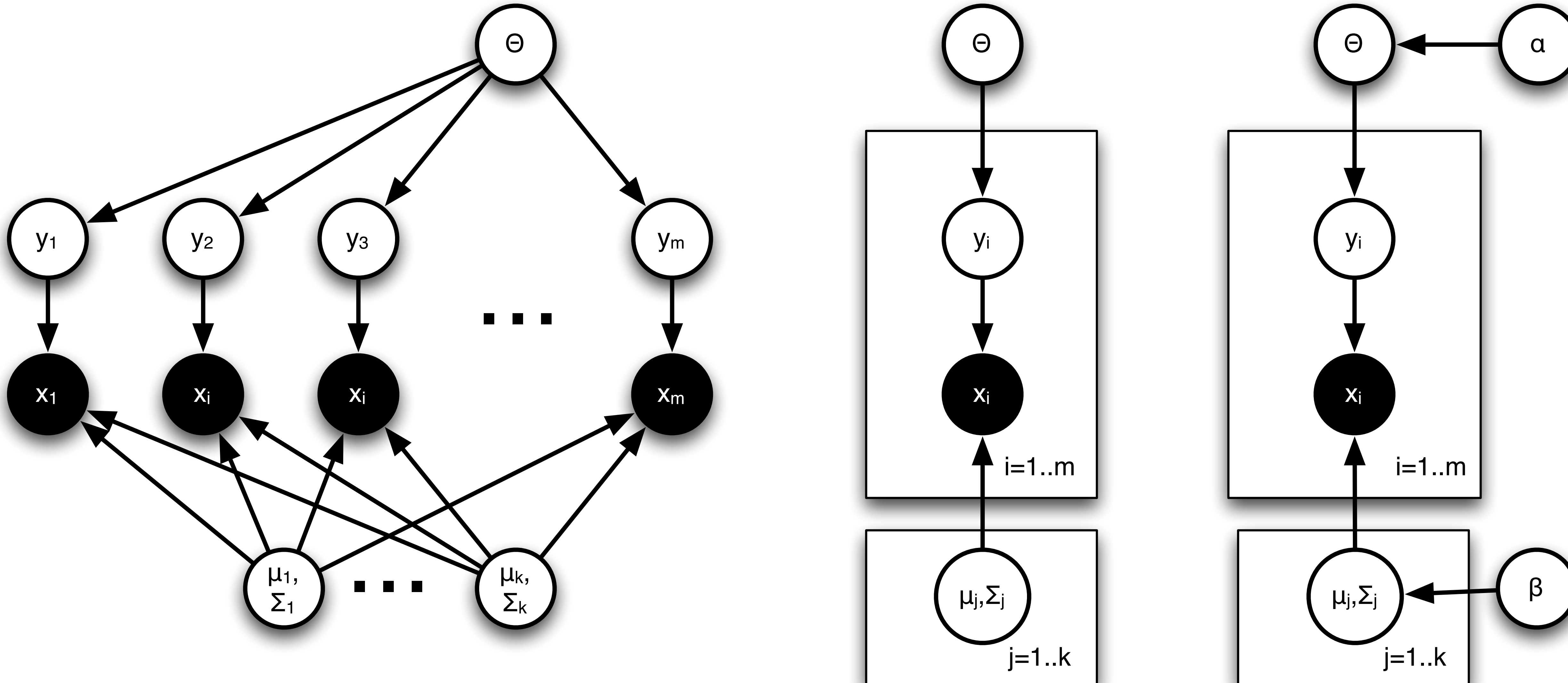




One size doesn't fit all



One size doesn't fit all



$$p(X, Y|\theta, \sigma, \mu) = \prod_{i=1}^{n} p(x_i|y_i, \sigma, \mu)p(y_i|\theta)$$

What can we cluster?

What can we cluster?

products

urls

mails

text

news

users

queries

locations

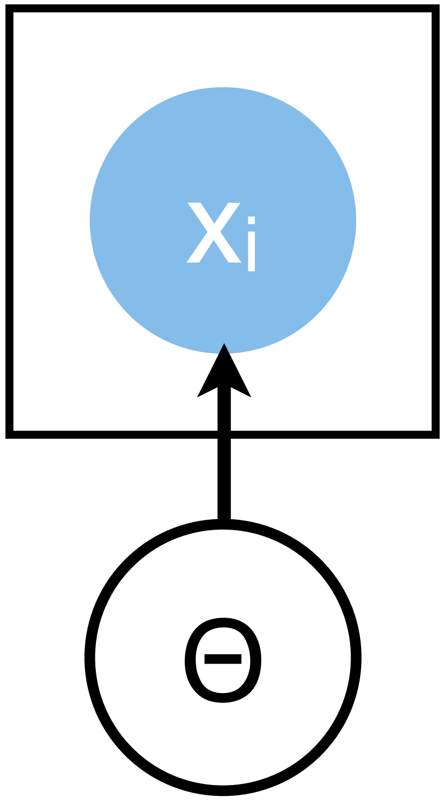
spammers

ads

events

abuse

Mixture of Gaussians



- Draw cluster ID y from discrete distribution
- Draw data x from Gaussian for cluster y

- Prior for discrete distribution - Dirichlet
- Prior for Gaussians - Gauss-Wishart
- Problem: we don't know y
  - If we knew the parameters we could get y

$$p(y|x,\theta) \propto p(x|y,\theta)p(y|\theta)$$

  - If we knew y we could get the parameters (estimate normal distribution)

k-means

- Fixed uniform variance for all Gaussians
- Fixed uniform distribution over clusters
- Initialize centers with random subset of points
- Find most likely cluster y for x (ignores p(y) …)

$$y_i = \operatorname*{argmax}_y p(x_i | y, \sigma, \mu)$$

- Find most likely center for given cluster

$$\mu_y = \frac{1}{n_y} \sum_i \{y_i = y\} x_i$$

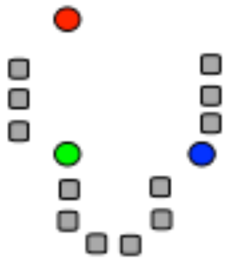
- Repeat until converged

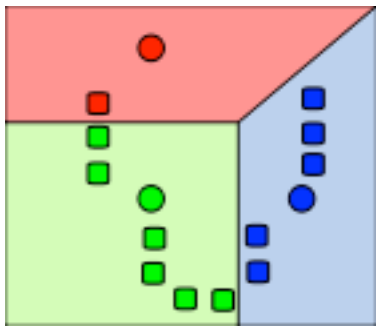
k-means

- Pro
  - simple algorithm
  - can be implemented by MapReduce passes
- Con
  - no proper probabilistic representation
  - can get stuck easily in local minima

k-means

partitioning





initialization

k-means

partitioning

partitioning

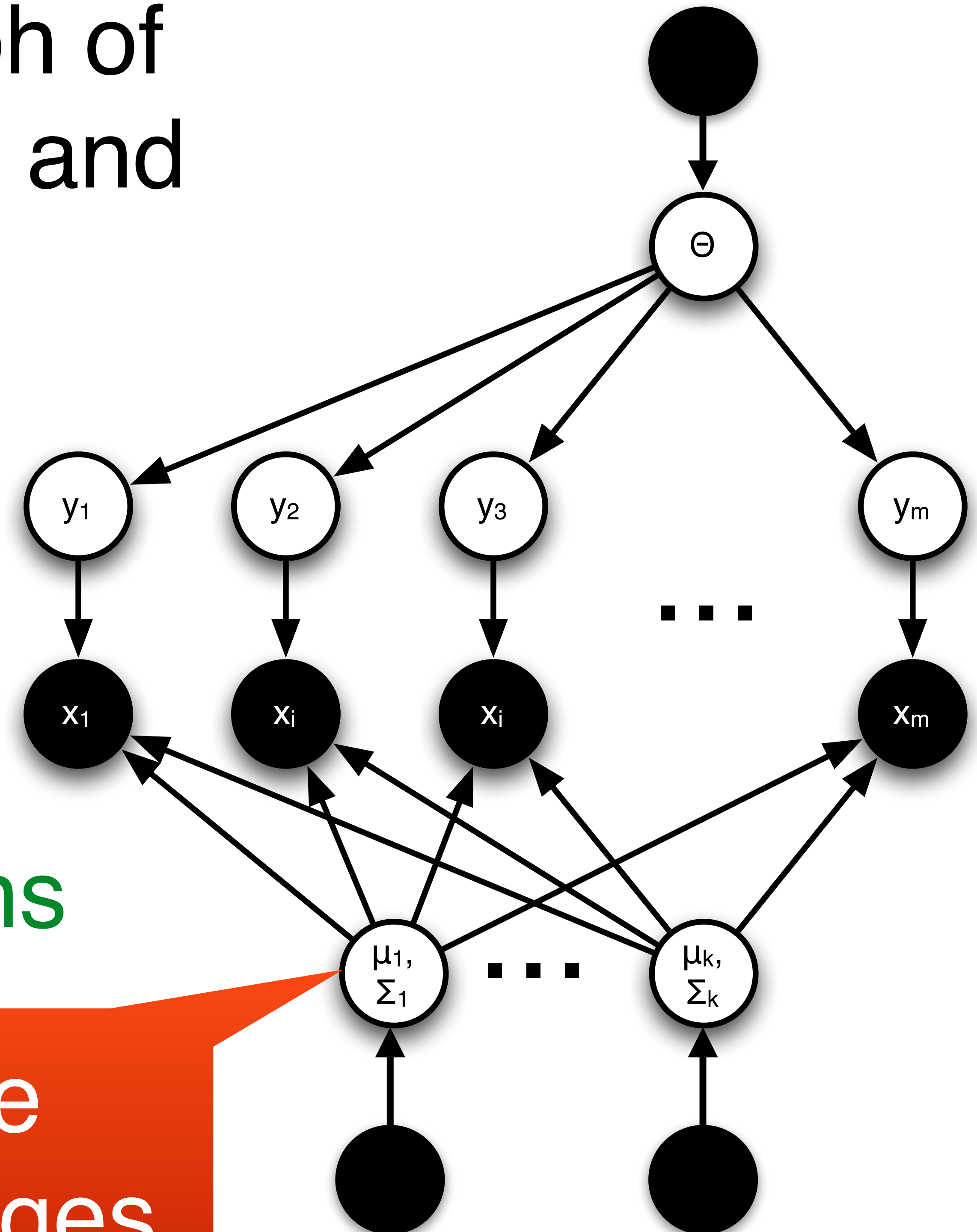
initialization

update

Inference Overview

Bayesian Inference

- Complete bipartite graph of dependence between y and the model parameters.
- Cannot generate a thin junction tree.
- Exact inference is impossible.
- We need approximations

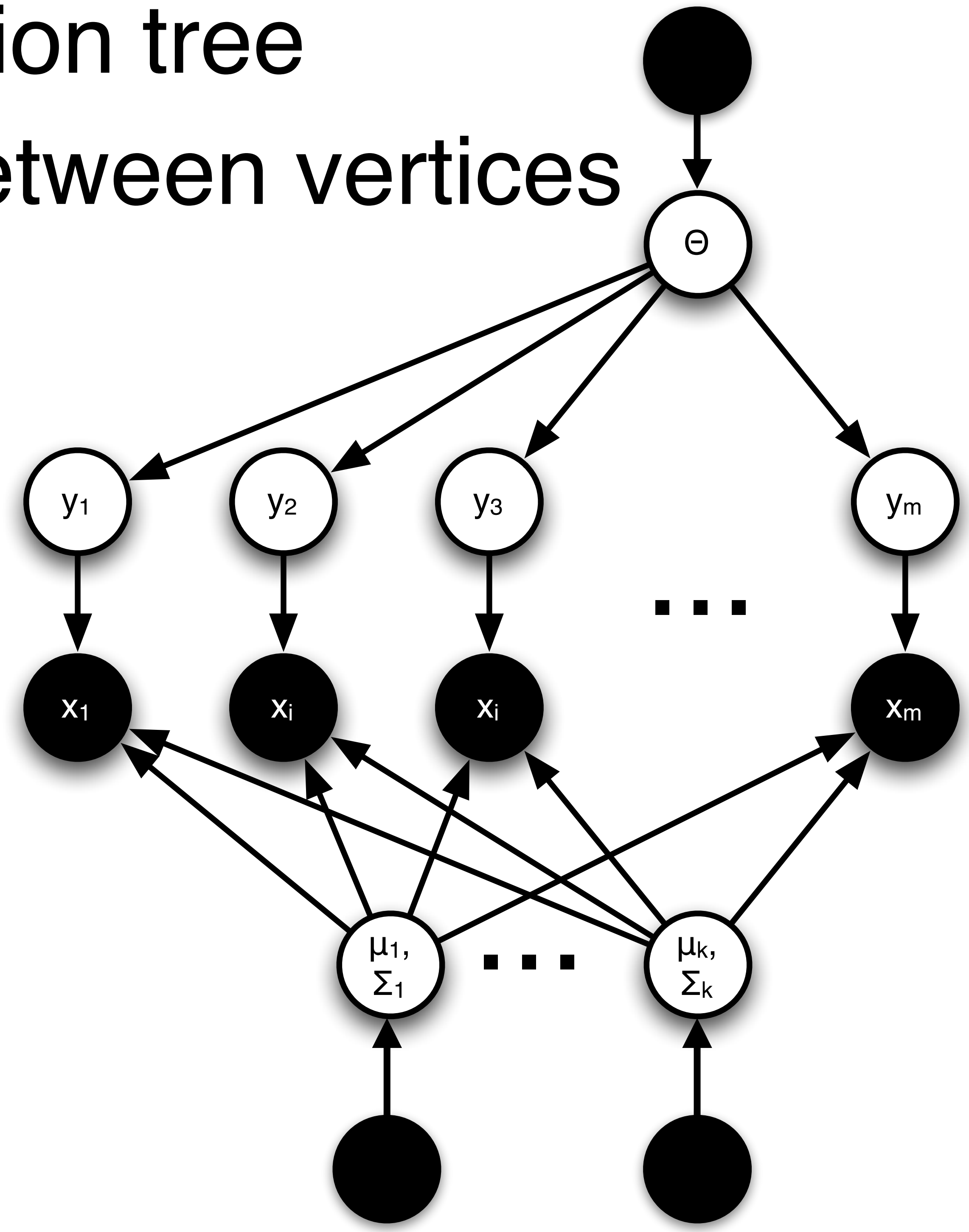


huge messages

Loopy belief propagation

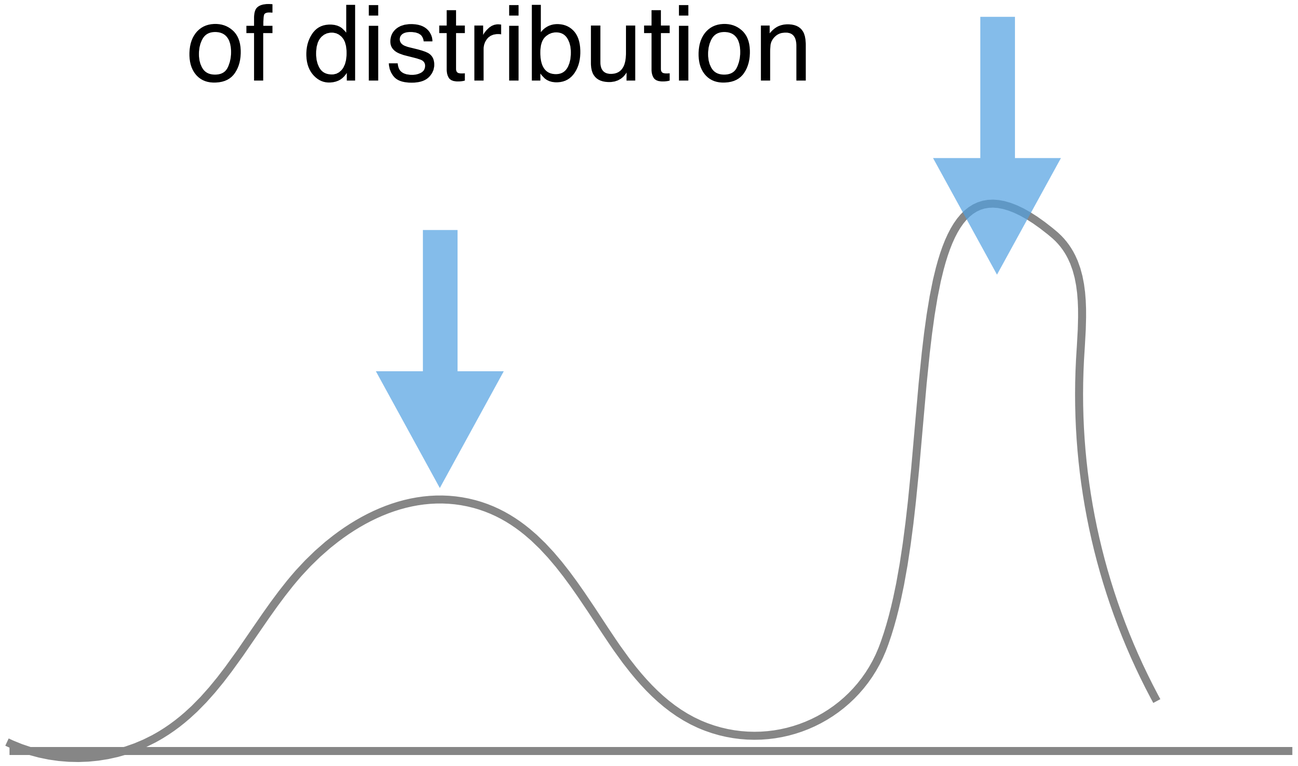
- Don't worry about junction tree
- Just send messages between vertices
- Still expensive for high degree vertices such as clusters
- Exact messages and potentials are too complicated

$$\left[\prod_{i=1}^{m} \mu_{y_i \to \theta}(\theta)\right] \cdot \psi(\theta)$$

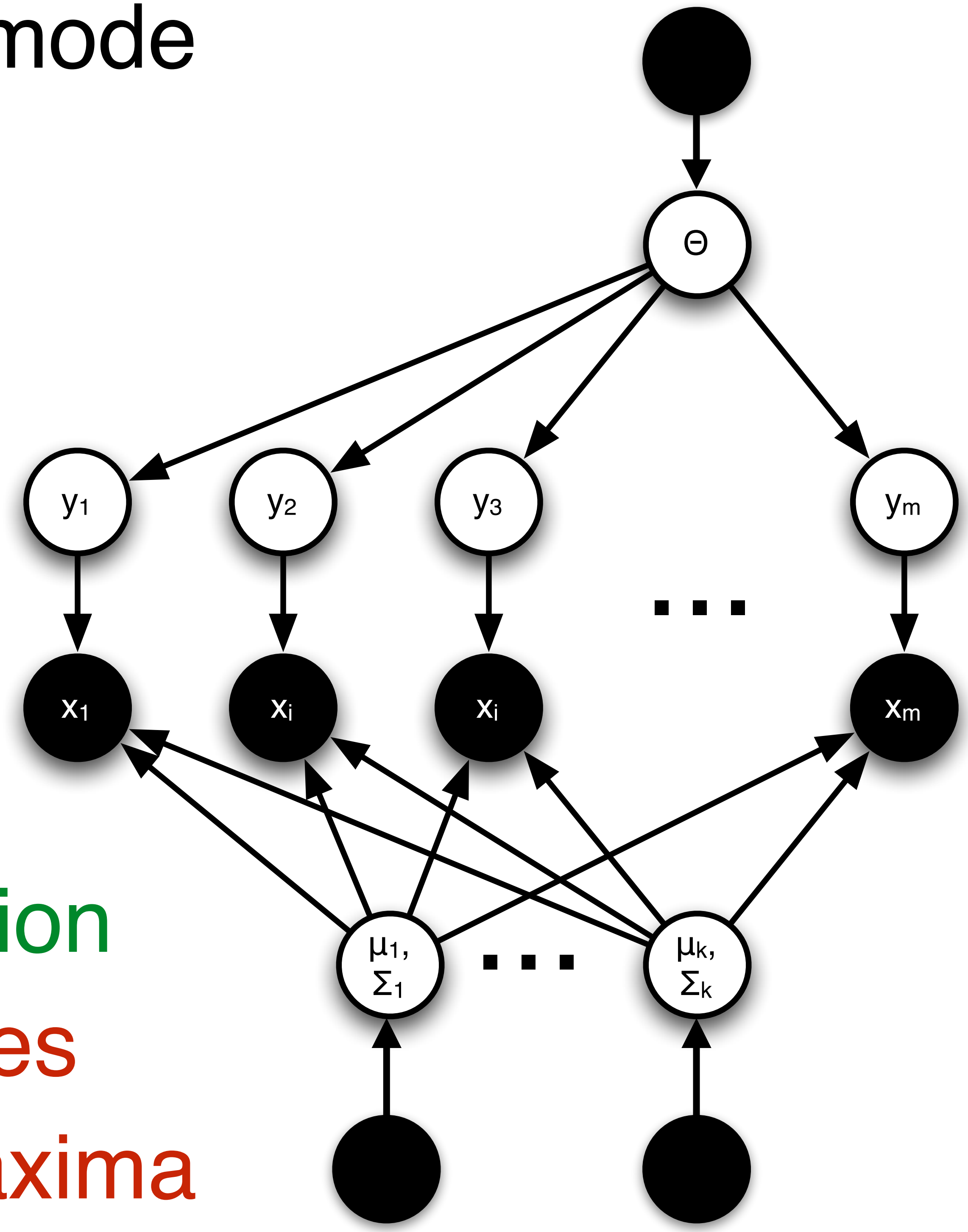


Maximum a posteriori

- Approximate integral by mode of distribution



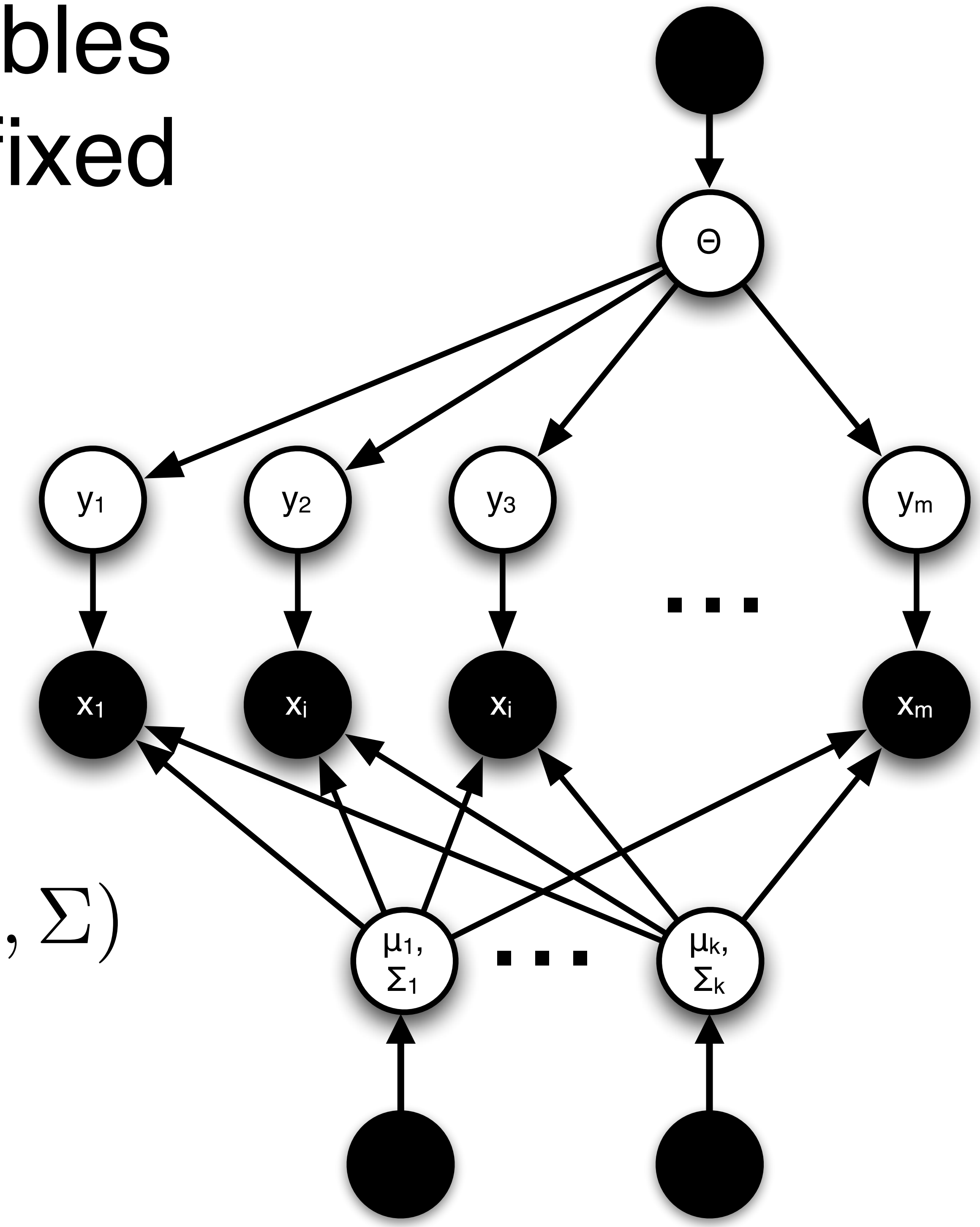
- Easy (see k-means)
- OK for unimodal distribution
- Misses out on large modes
- Can get stuck in local maxima



Sampling

- Sample subset of variables while keeping the rest fixed
- Iterate until converged
- Draw several samples
- Gibbs sampler
  Draw one group
  at a time and iterate

$$y_i \sim p(y_i | X, Y^{-i}, \theta, \mu, \Sigma)$$
$$\theta \sim p(\theta | X, Y, \mu, \Sigma)$$
$$(\mu, \Sigma) \sim p(\mu, \Sigma | X, \theta, Y)$$

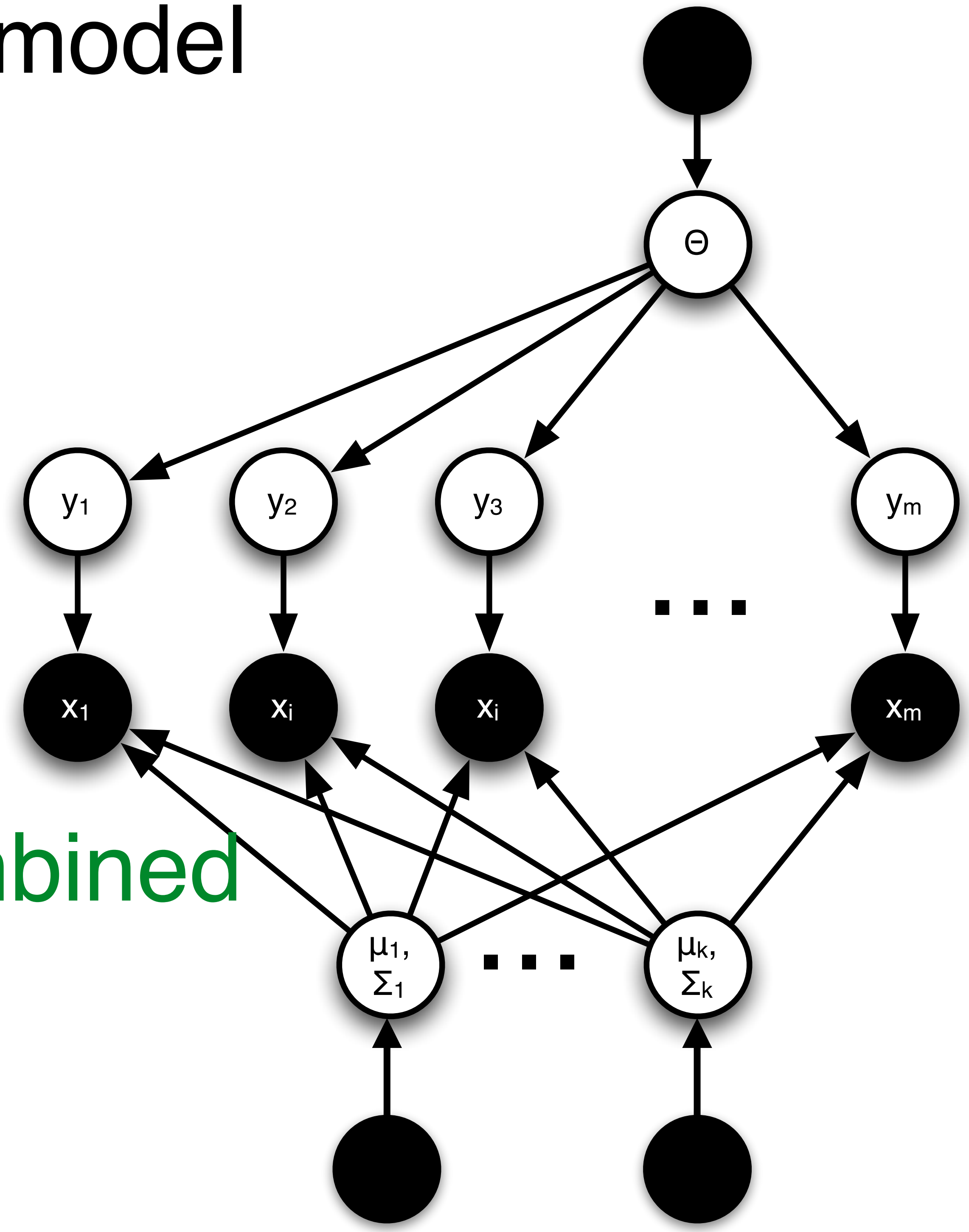


Variational Inference

- Approximate graphical model by simpler one

$$q(\theta)\prod_{i=1}^{m} q(y_i|\theta)\prod_{j=1}^{k} q(\mu_j,\Sigma_j)$$

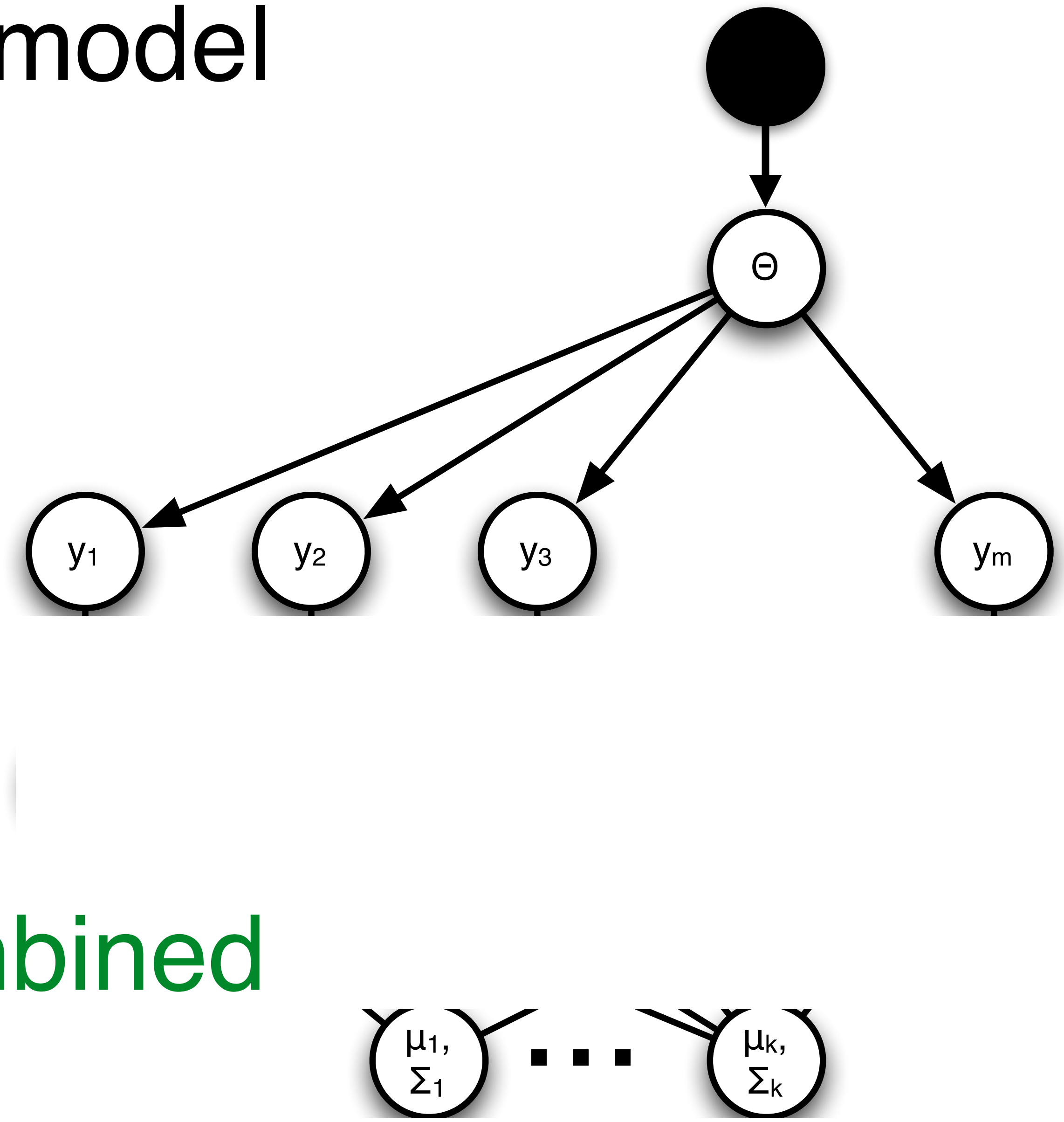
- Minimize 'distance' between models
- Often methods are combined into hybrid approach



Variational Inference

- Approximate graphical model by simpler one

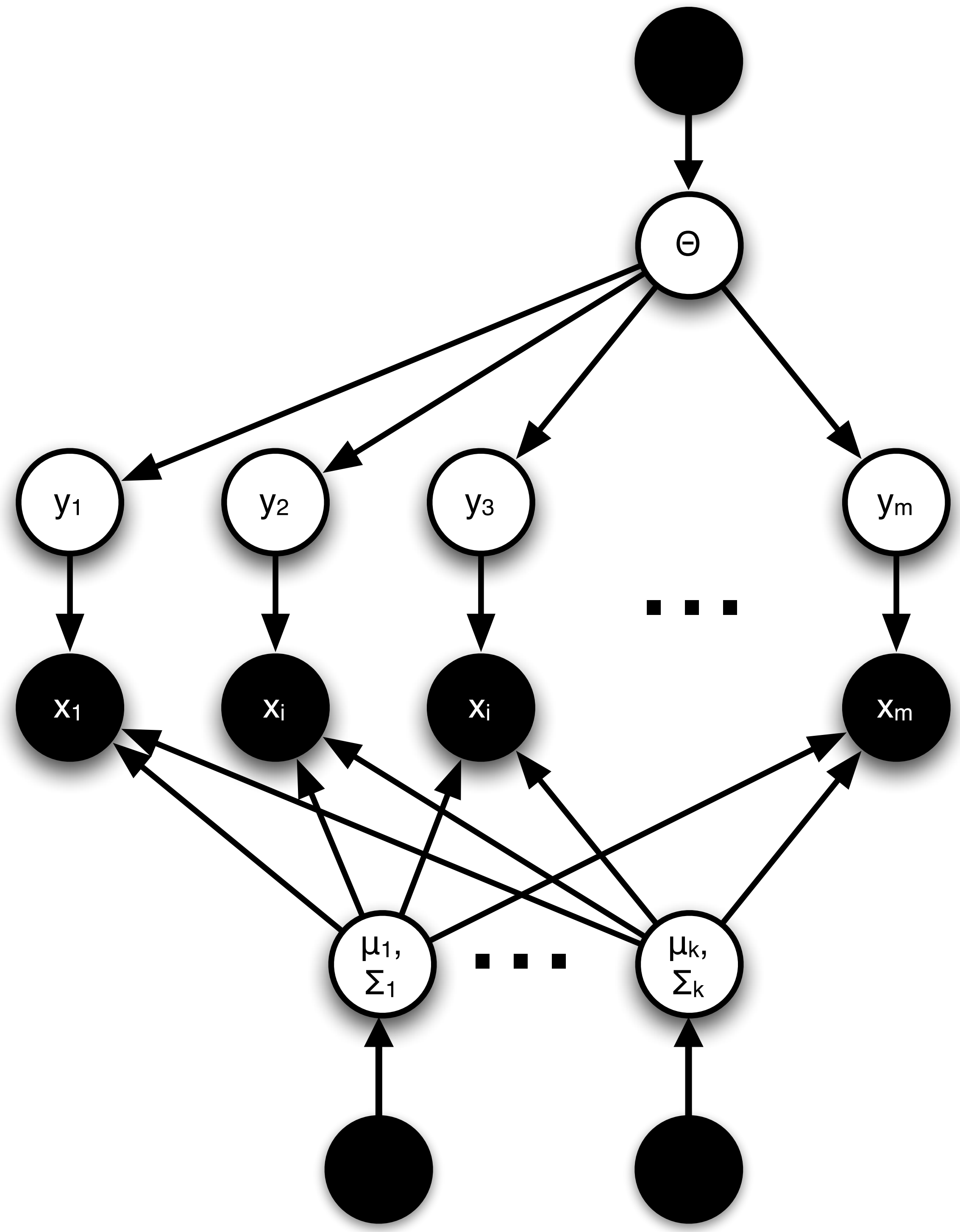
$$q(\theta) \prod_{i=1}^{m} q(y_i|\theta) \prod_{j=1}^{k} q(\mu_j, \Sigma_j)$$

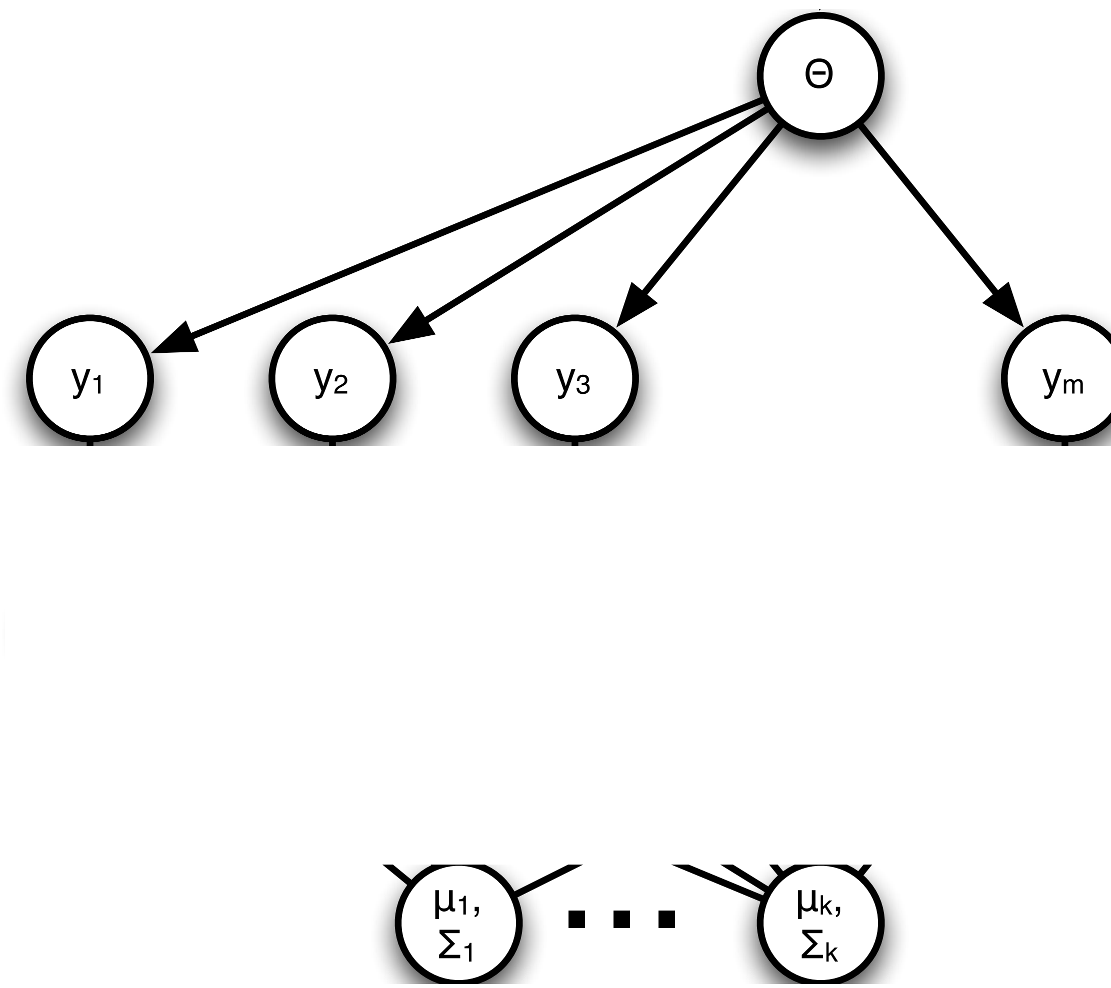


- Minimize 'distance' between models
- Often methods are combined into hybrid approach

$$\underset{\gamma}{\text{minimize}}\, D(q_\gamma(Y, \theta, \mu, \Sigma) | p(Y, \theta, \mu, \Sigma | X))$$

Variational Inference and EM



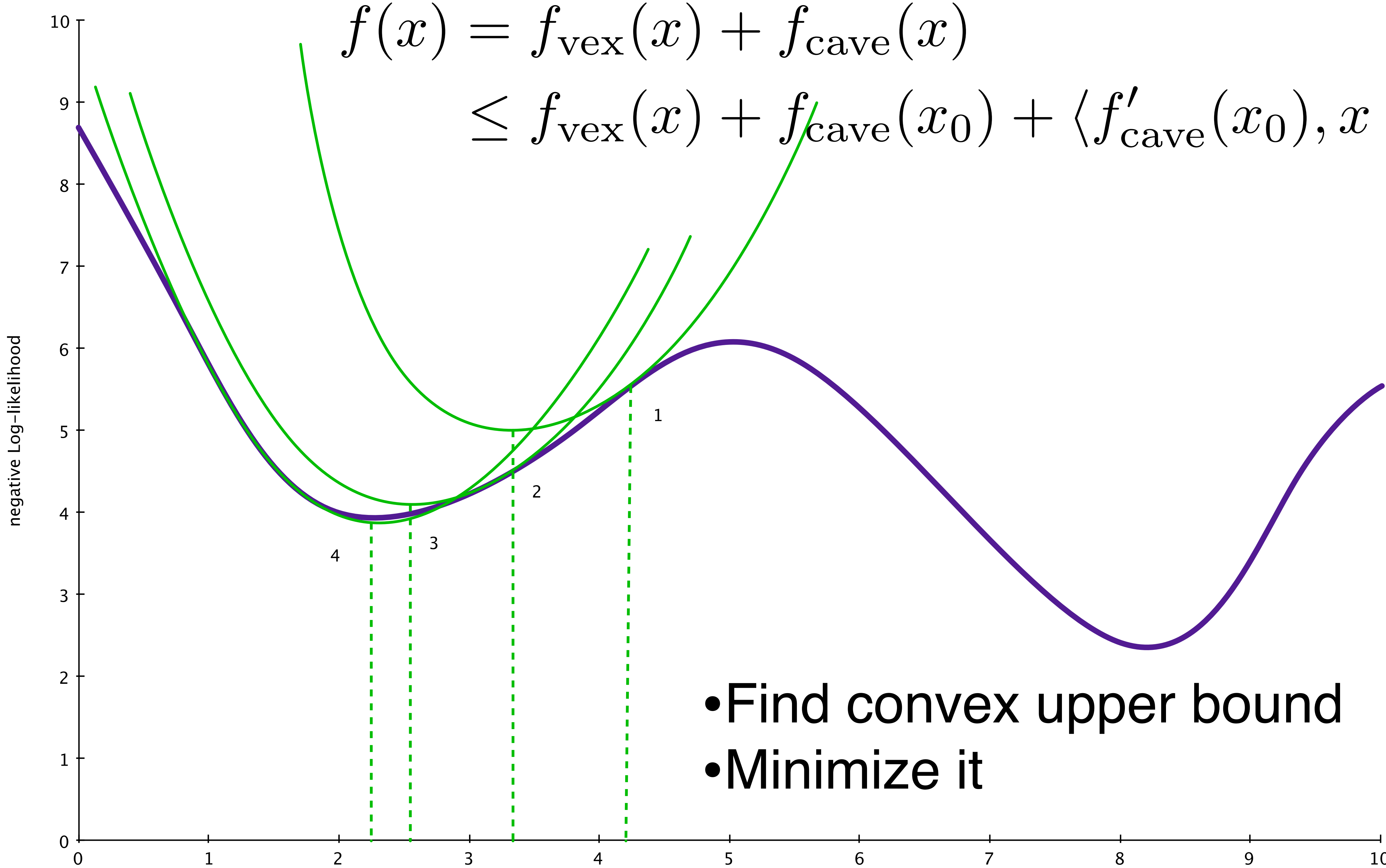


Nonconvex Optimization

- **Optimization Problem**
  Find the parameters (clusters, probabilities) for the mixture of Gaussians problem
  $$\underset{\theta,\mu,\sigma}{\text{maximize}}\, p(X|\theta,\sigma,\mu) = \underset{\theta,\mu,\sigma}{\text{maximize}} \sum_{Y} \prod_{i=1}^{n} p(x_i|y_i,\sigma,\mu)p(y_i|\theta)$$
  This problem is nonconvex and difficult to solve
- **Key idea**
  If we knew p(ylx) we could estimate the remaining parameters easily and vice versa

DC Programming

$$f(x) = f_{\text{vex}}(x) + f_{\text{cave}}(x)$$
$$\leq f_{\text{vex}}(x) + f_{\text{cave}}(x_0) + \langle f'_{\text{cave}}(x_0), x - x_0 \rangle$$



- Find convex upper bound
- Minimize it

Expectation Maximization

Expectation Maximization

- Variational Bound

$$\begin{aligned}\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\ &= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\ &= \int dq(y) \log p(x,y;\theta) - \int dq(y) \log q(y)\end{aligned}$$

Expectation Maximization

- Variational Bound

$$
\begin{aligned}
\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\
&= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\
&= \int dq(y) \log p(x,y;\theta) - \int dq(y) \log q(y)
\end{aligned}
$$

$$q(y) = p(y|x;\theta)$$

Expectation Maximization

- Variational Bound

$$
\begin{aligned}
\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\
&= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\
&= \int dq(y) \log p(x,y;\theta) - \int dq(y) \log q(y)
\end{aligned}
$$

$$q(y) = p(y|x;\theta)$$

find bound

Expectation Maximization

- Variational Bound

$$
\begin{aligned}
\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\
&= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\
&= \int dq(y)\log p(x,y;\theta) - \int dq(y)\log q(y)
\end{aligned}
$$

$$q(y) = p(y|x;\theta)$$

find bound

Expectation Maximization

- Variational Bound

$$\begin{aligned}\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\ &= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\ &= \int dq(y)\log p(x,y;\theta) - \int dq(y)\log q(y)\end{aligned}$$

$$q(y) = p(y|x;\theta)$$

find bound

Expectation Maximization

- Variational Bound

$$\begin{aligned}\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\ &= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\ &= \int dq(y)\log p(x,y;\theta) - \int dq(y)\log q(y)\end{aligned}$$

- This inequality is tight for p(ylx) = q(y)

$$q(y) = p(y|x;\theta)$$

find bound

Expectation Maximization

- Variational Bound

$$\begin{aligned}\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\ &= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\ &= \int dq(y) \log p(x,y;\theta) - \int dq(y) \log q(y)\end{aligned}$$

- This inequality is tight for p(y|x) = q(y)
- Expectation step

$$q(y) = p(y|x;\theta)$$

find bound

Expectation Maximization

- Variational Bound

$$\begin{aligned}\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\ &= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\ &= \int dq(y) \log p(x,y;\theta) - \int dq(y) \log q(y)\end{aligned}$$

- This inequality is tight for p(y|x) = q(y)
- Expectation step

$$q(y) = p(y|x;\theta)$$

find bound

- Maximization step

$$\theta^* = \operatorname*{argmax}_\theta \int dq(y) \log p(x,y;\theta)$$

Expectation Maximization

- Variational Bound

$$\begin{aligned}\log p(x;\theta) &\geq \log p(x;\theta) - D(q(y)\|p(y|x;\theta)) \\ &= \int dq(y)\left[\log p(x;\theta) + \log p(y|x;\theta) - \log q(y)\right] \\ &= \int dq(y) \log p(x,y;\theta) - \int dq(y) \log q(y)\end{aligned}$$

- This inequality is tight for p(y|x) = q(y)
- Expectation step

$$q(y) = p(y|x;\theta)$$

find bound

- Maximization step

$$\theta^* = \operatorname*{argmax}_\theta \int dq(y) \log p(x,y;\theta)$$

maximize it

Expectation Step

- Factorizing distribution

$$q(Y) = \prod_i q_i(y)$$

- E-Step

$$q_i(y) \propto p(x_i | y_i, \mu, \sigma) p(y_i | \theta) \text{ hence}$$

$$m_{iy} := \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma_y|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(x_i - \mu_y)\Sigma_y^{-1}(x_i - \mu_y)\right] p(y)$$

$$\boxed{q_i(y) = \frac{m_{iy}}{\sum_{y'} m_{iy'}}}$$

Maximization Step

- Log-likelihood

$$\log p(X, Y|\theta, \mu, \sigma) = \sum_{i=1}^{n} \log p(x_i|y_i, \mu, \sigma) + \log p(y_i|\theta)$$

- Cluster distribution (weighted Gaussian MLE)

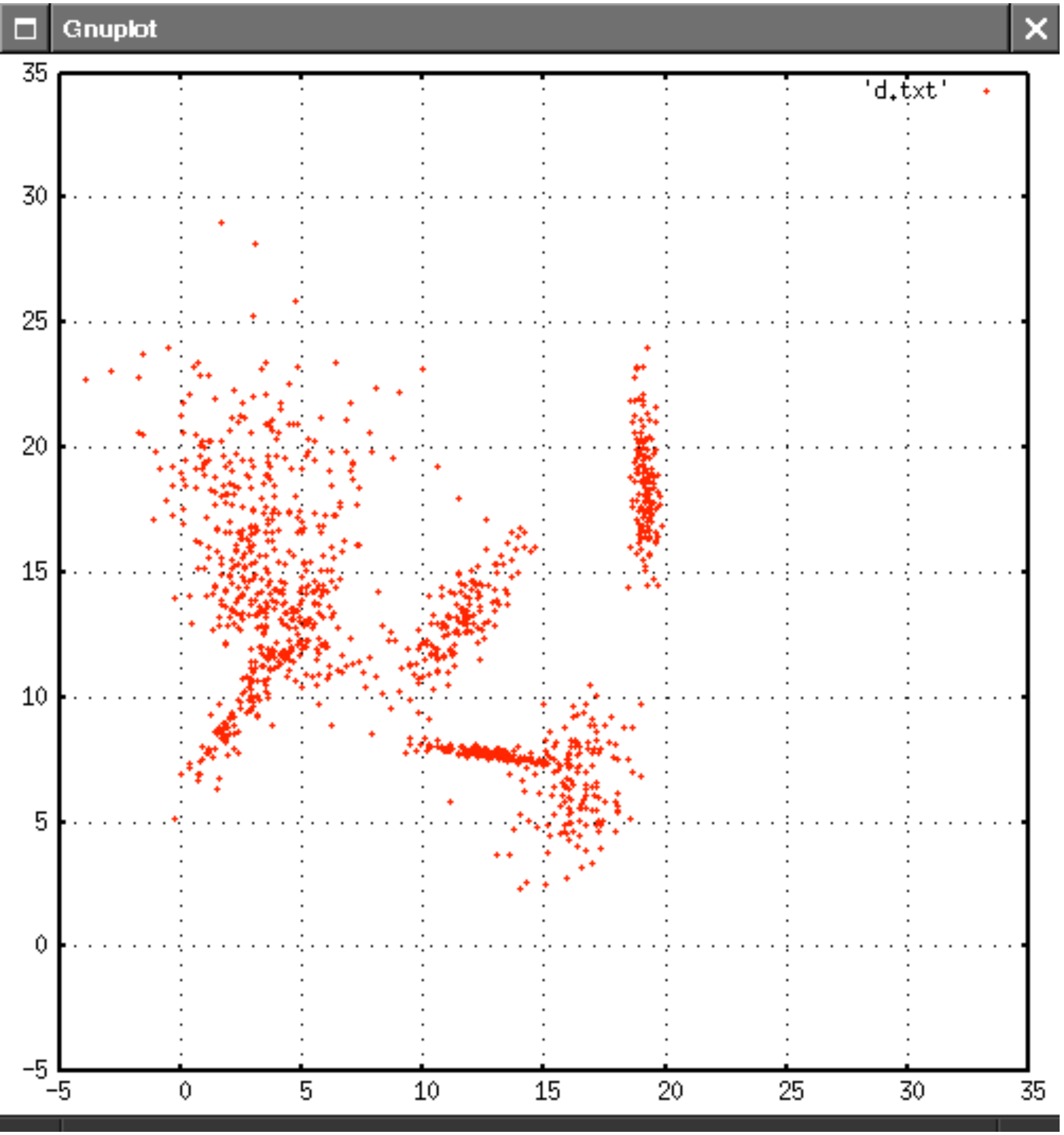
$$n_y = \sum_i q_i(y)$$

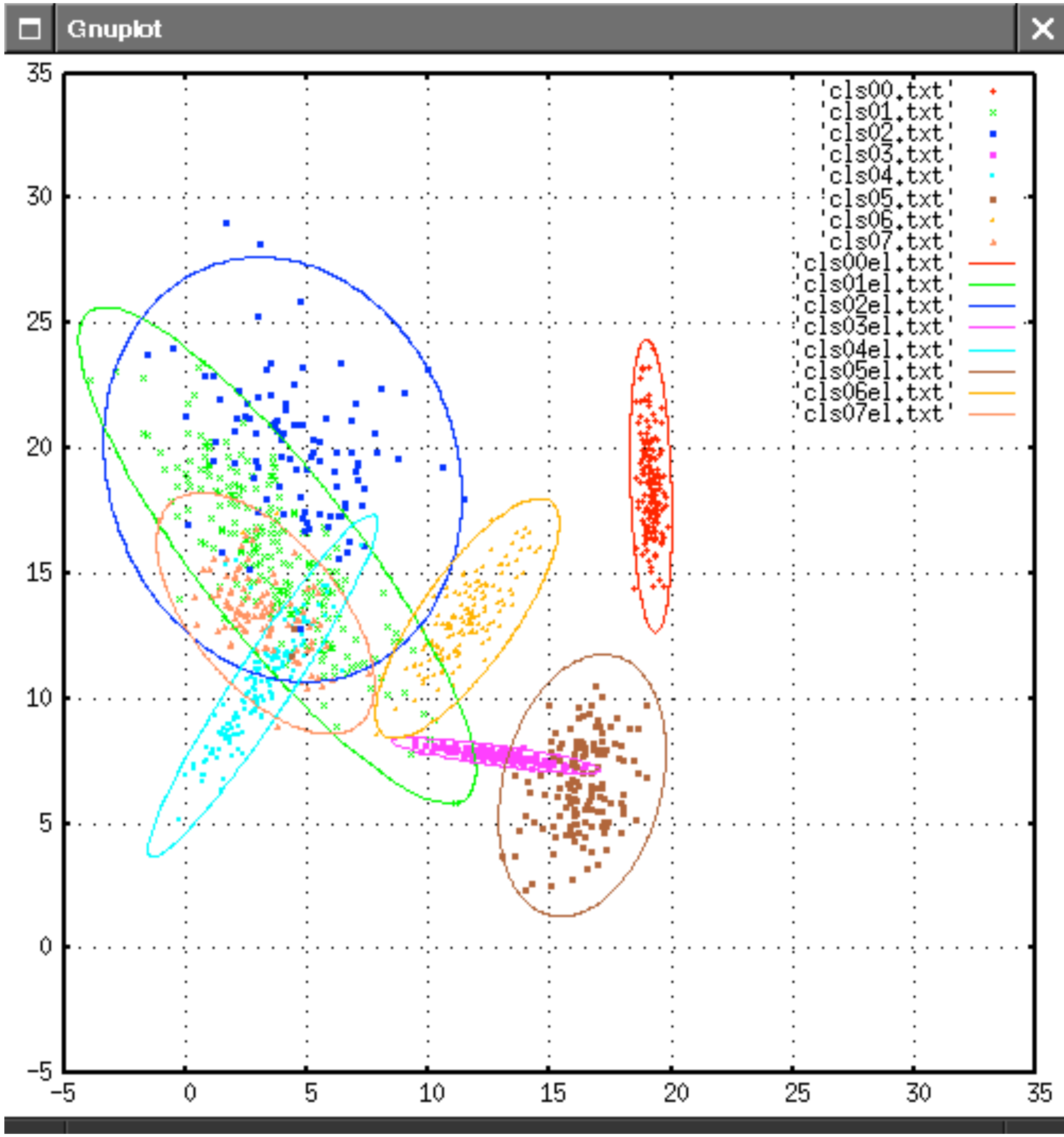
$$\boxed{\begin{aligned}\mu_y &= \frac{1}{n_y}\sum_{i=1}^{n} q_i(y) x_i \\ \Sigma_y &= \frac{1}{n_y}\sum_{i=1}^{n} q_i(y) x_i x_i^\top - \mu_y \mu_y^\top\end{aligned}}$$

- Cluster probabilities

$$\theta^* = \operatorname*{argmax}_\theta \sum_{i=1}^{n}\sum_y q_i(y) \log p(y_i|\theta) \text{ hence } \boxed{p(y|\theta) = \frac{n_y}{n}}$$

EM Clustering in action





Problem

Estimates will diverge
(infinite variance, zero probability, tiny clusters)

Solution

- Use priors for μ, σ, θ
  - Dirichlet distribution for cluster probabilities
  - Gauss-Wishart for Gaussian
- Cluster distribution

$$n_y = n_0 + \sum_i q_i(y)$$

$$\mu_y = \frac{1}{n_y} \sum_{i=1}^{n} q_i(y) x_i$$

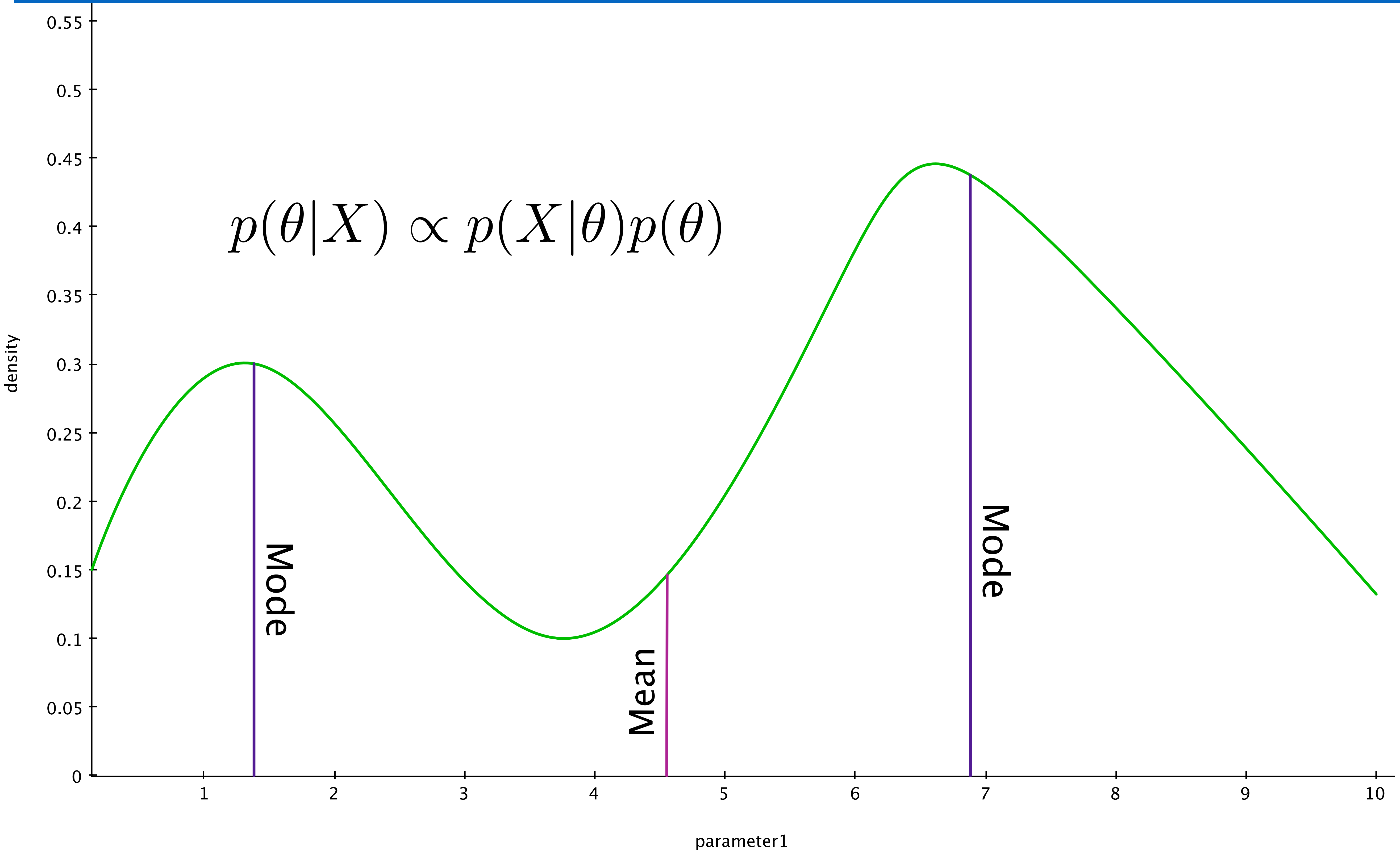
$$\Sigma_y = \frac{1}{n_y} \sum_{i=1}^{n} q_i(y) x_i x_i^\top + \frac{n_0}{n_y} \mathbf{1} - \mu_y \mu_y^\top$$

- Cluster probabilities

$$p(y|\theta) = \frac{n_y}{n + k \cdot n_0}$$

Sampling

Is maximization (always) good?



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

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

		
	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)
(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,.)

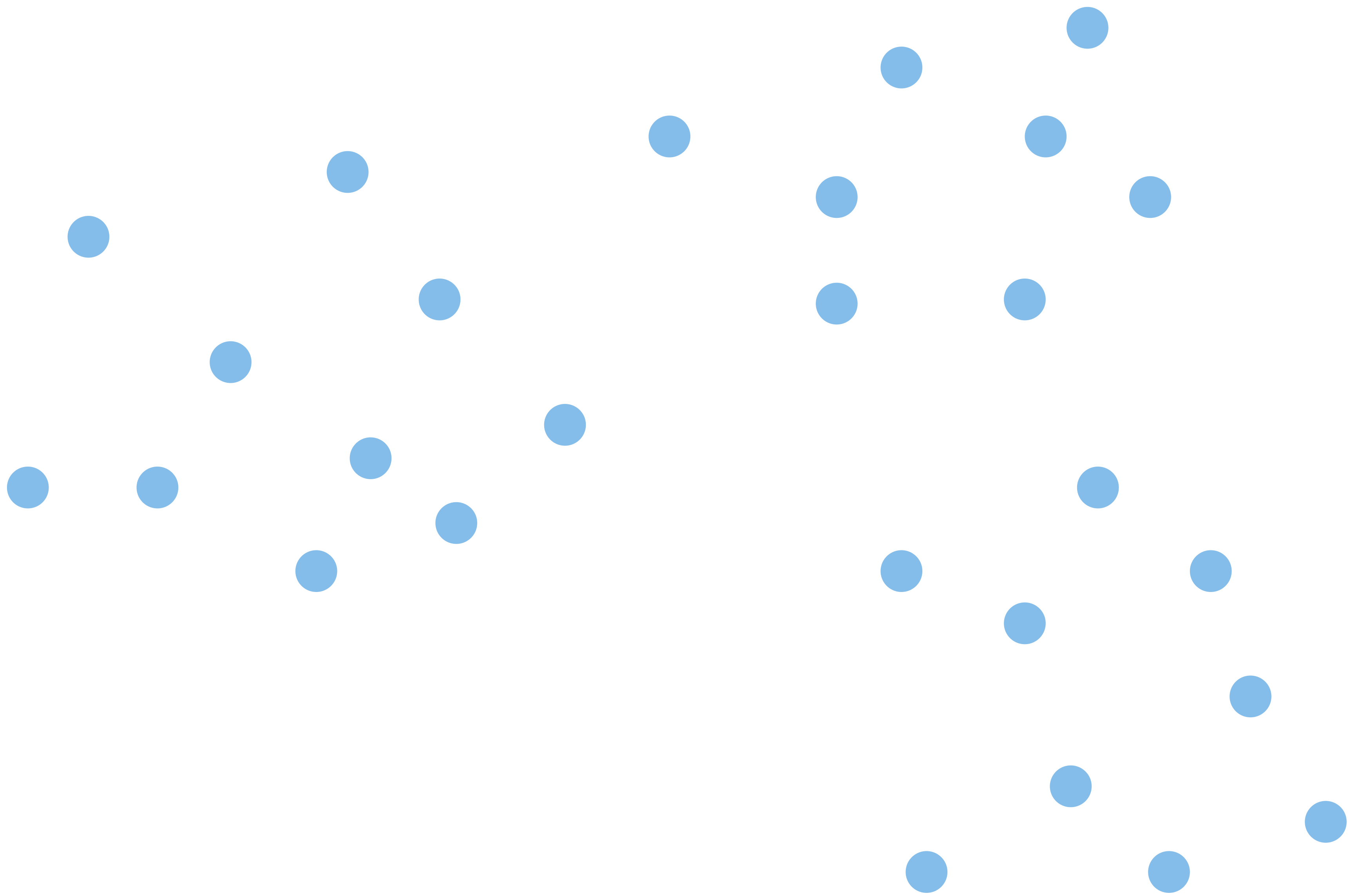
Gibbs sampling

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

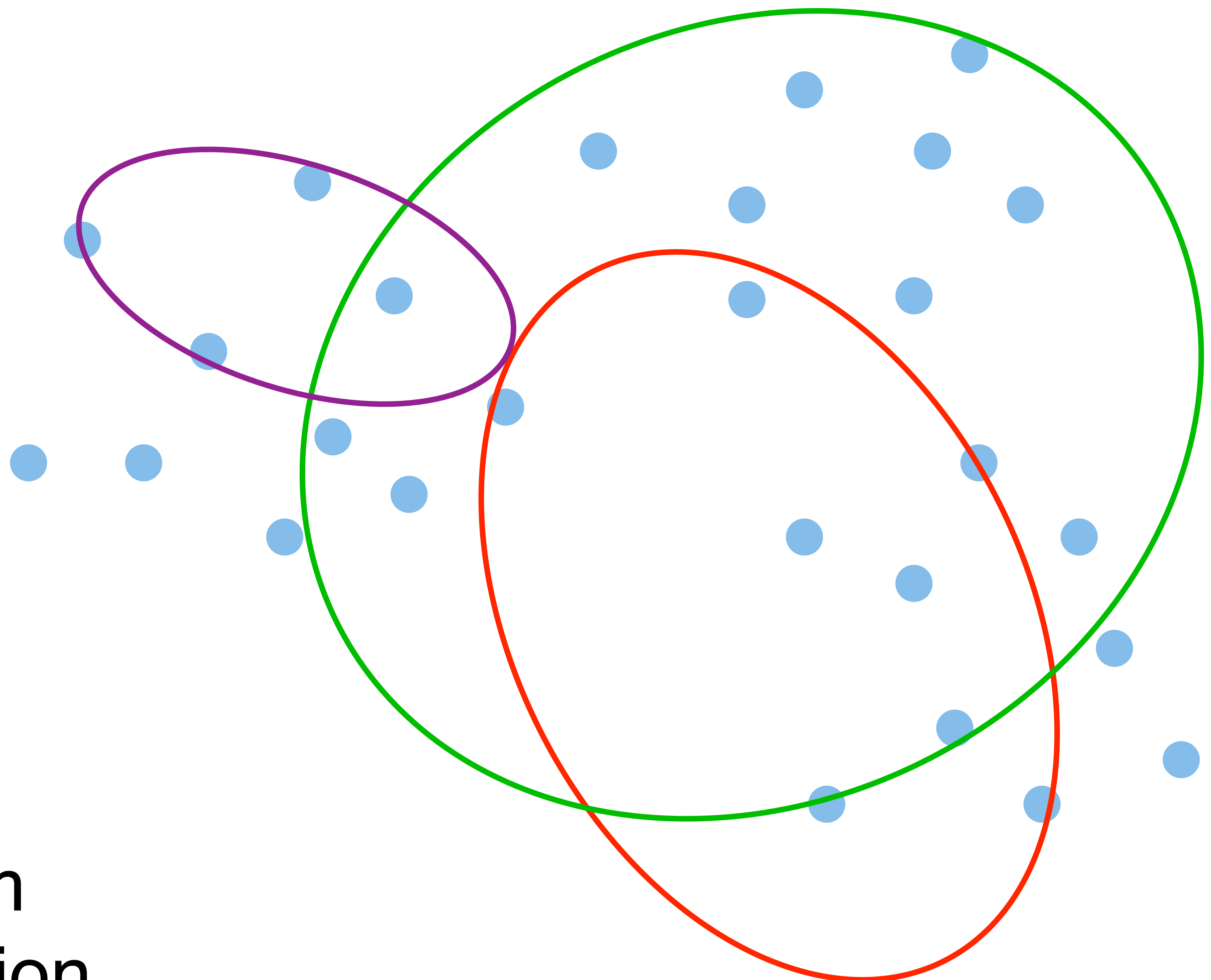
		
	0.45	0.05
	0.05	0.45

(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

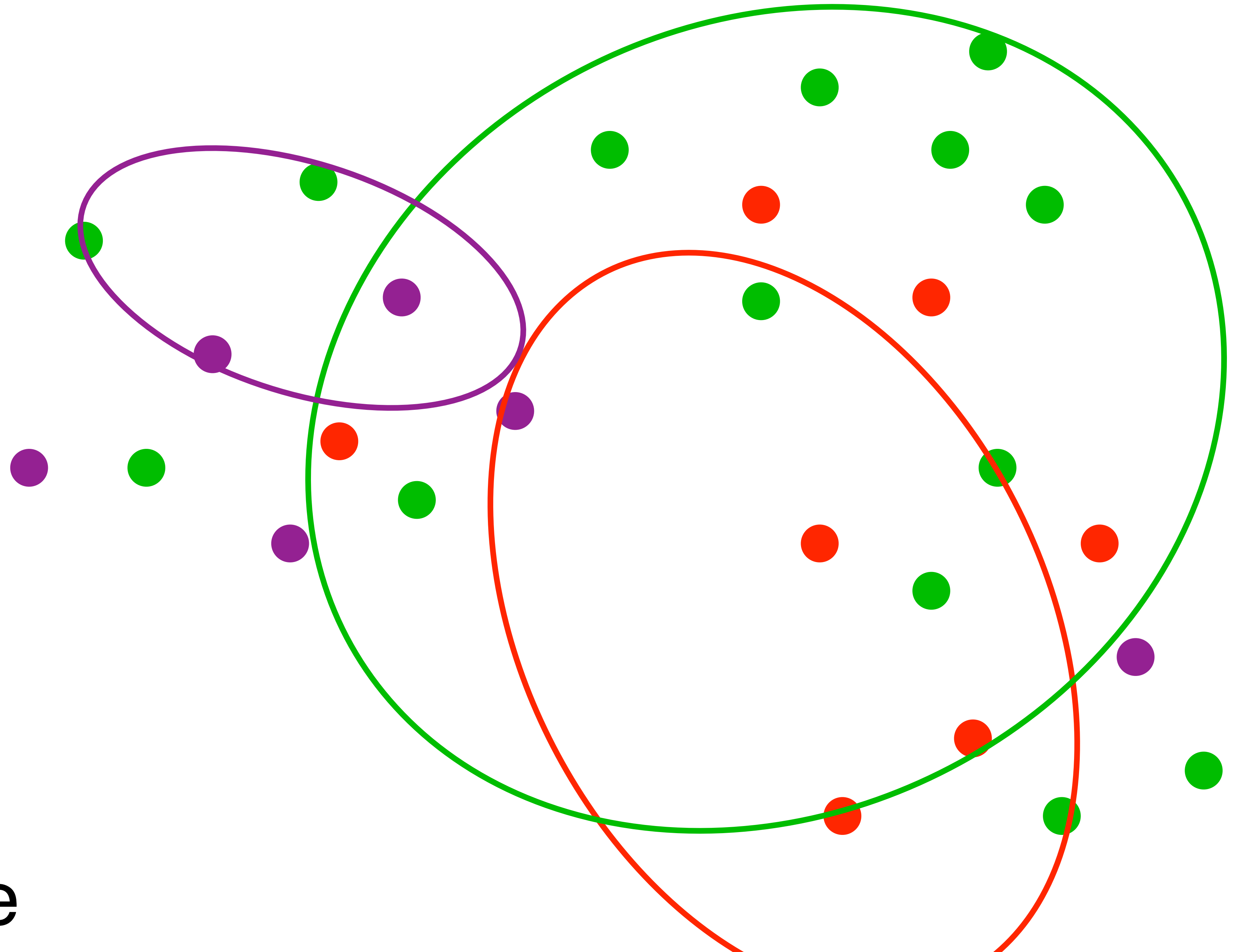


Gibbs sampling for clustering



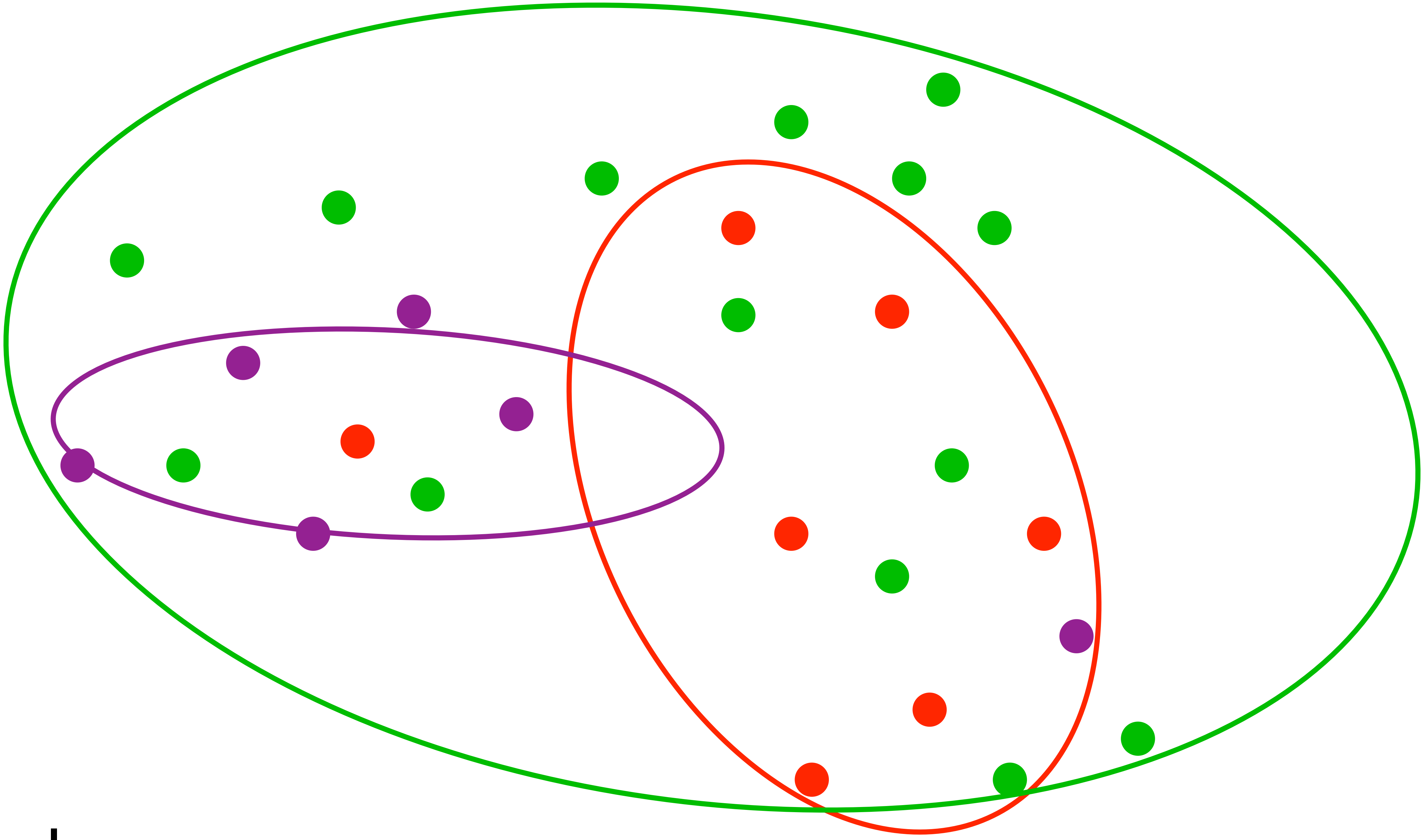
random
initialization

Gibbs sampling for clustering



sample
cluster labels

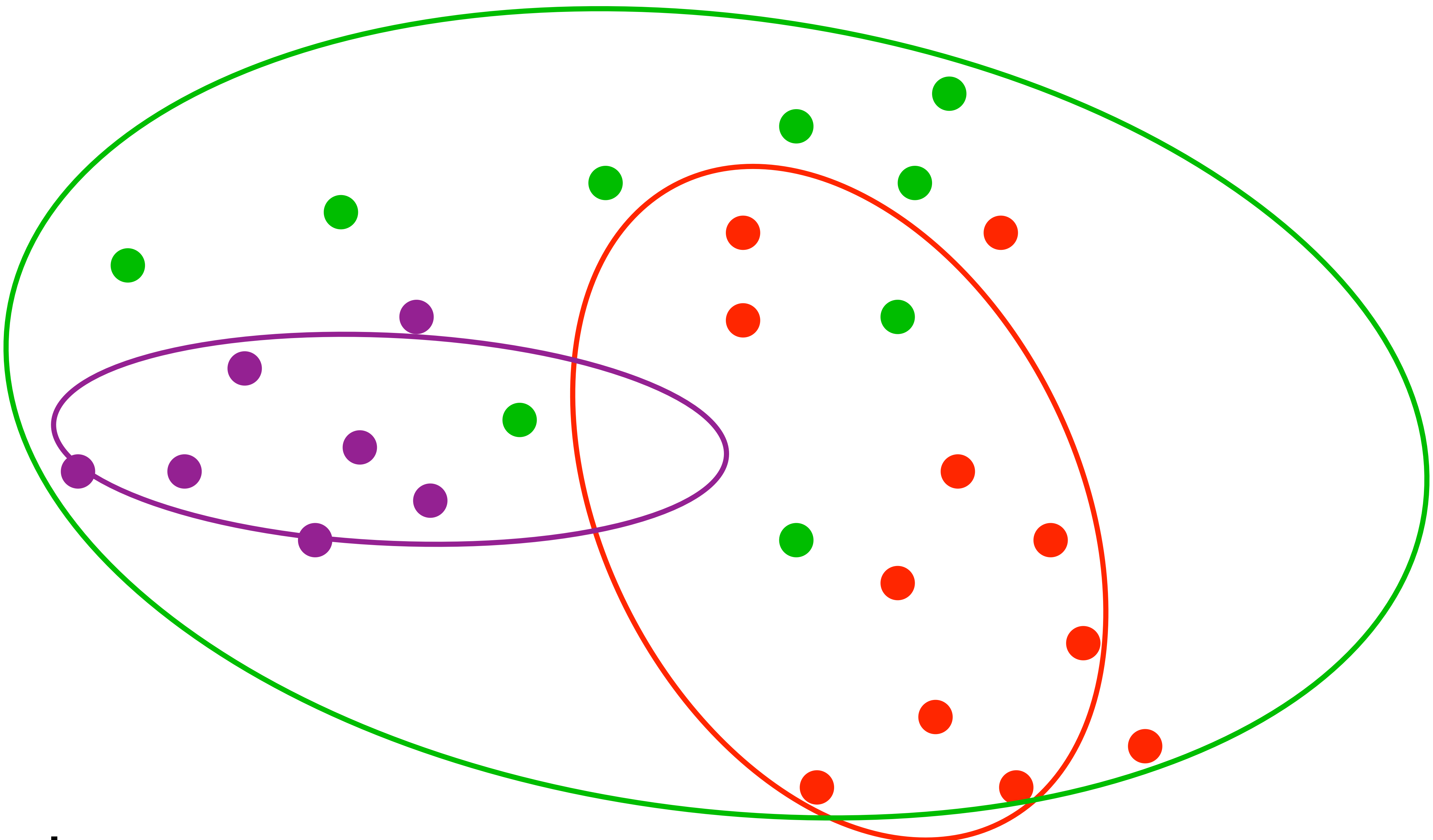
Gibbs sampling for clustering



resample
cluster model

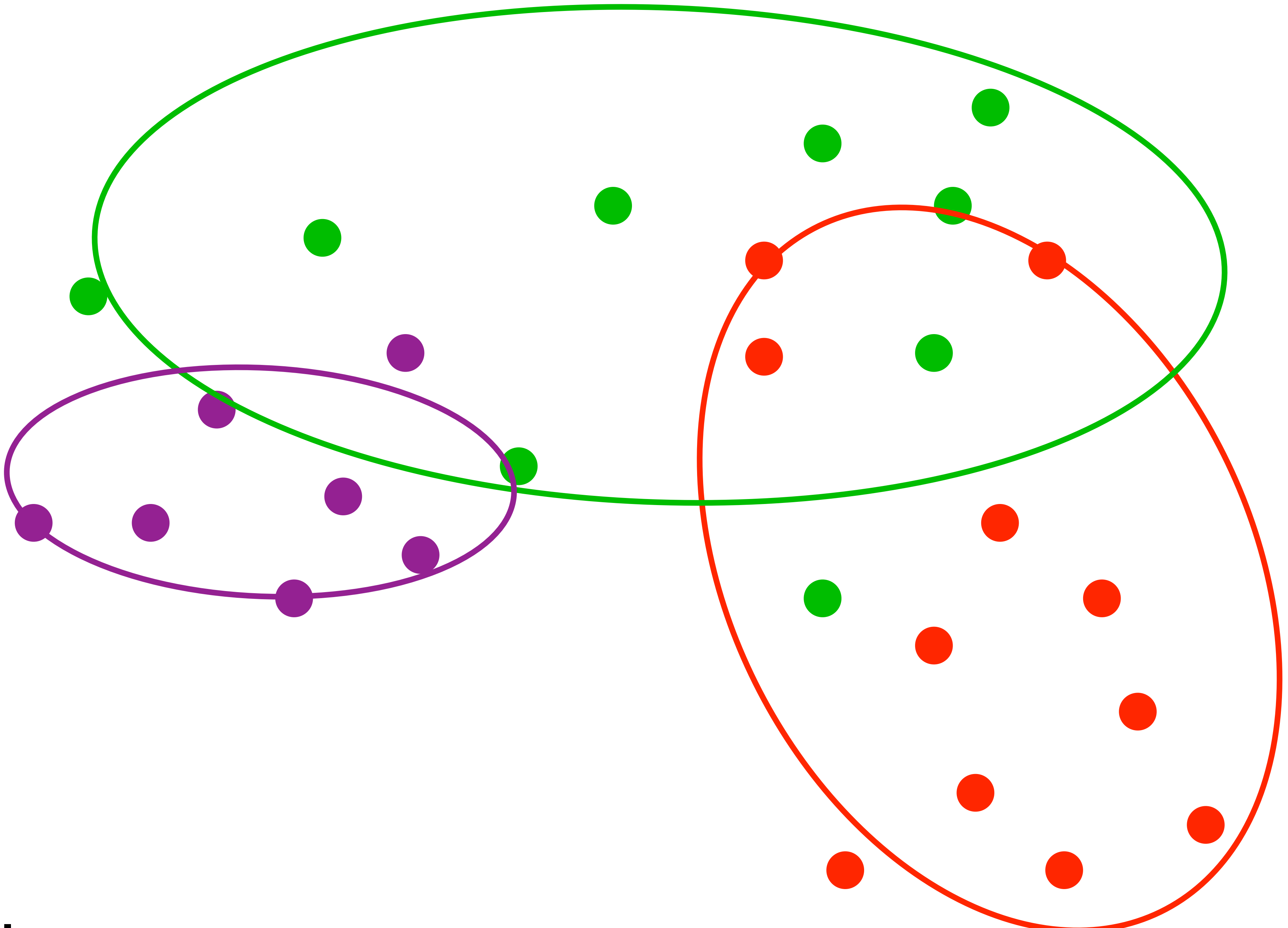
Carnegie Mellon University

Gibbs sampling for clustering



resample
cluster labels

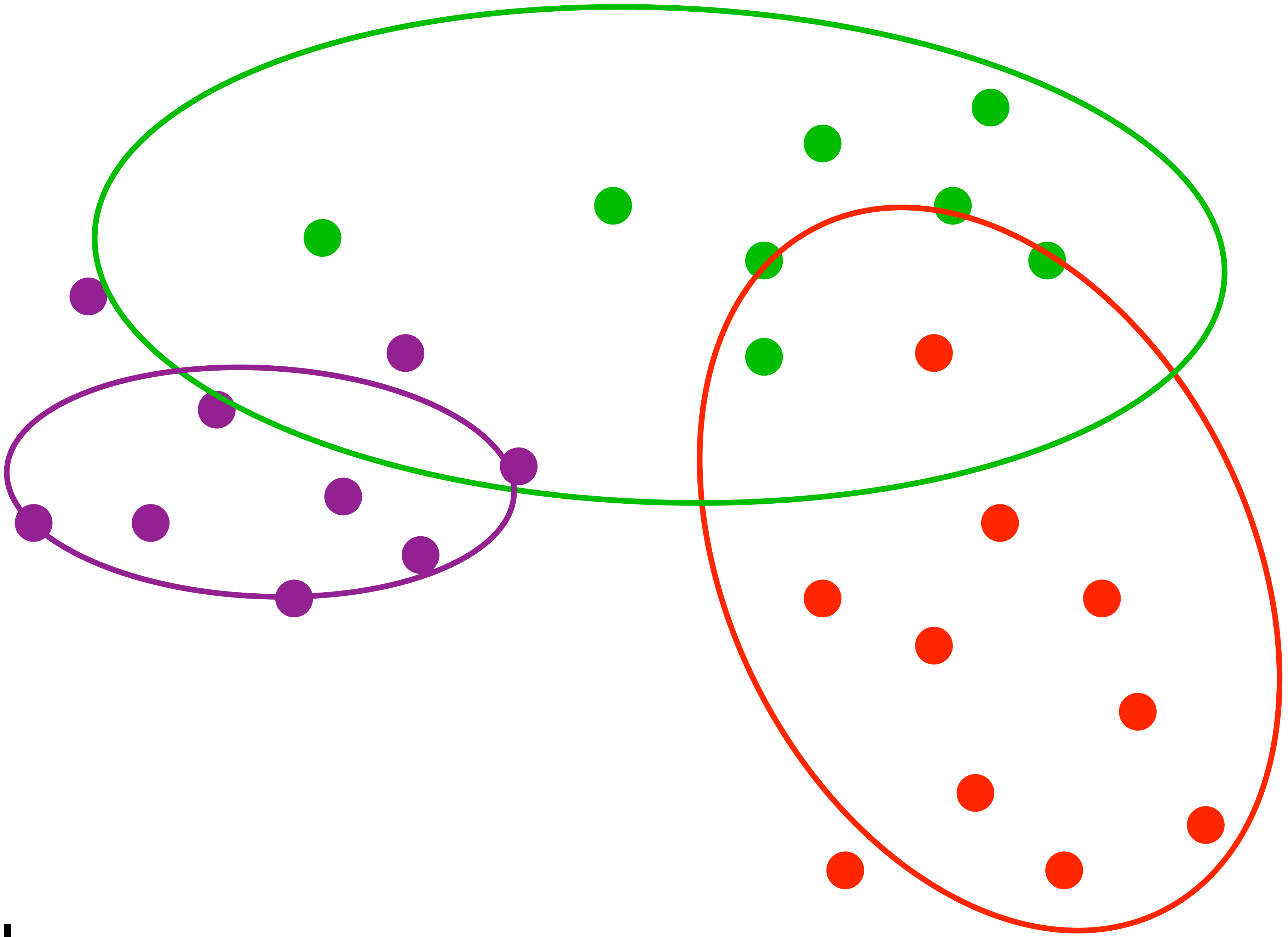
Gibbs sampling for clustering



resample
cluster model

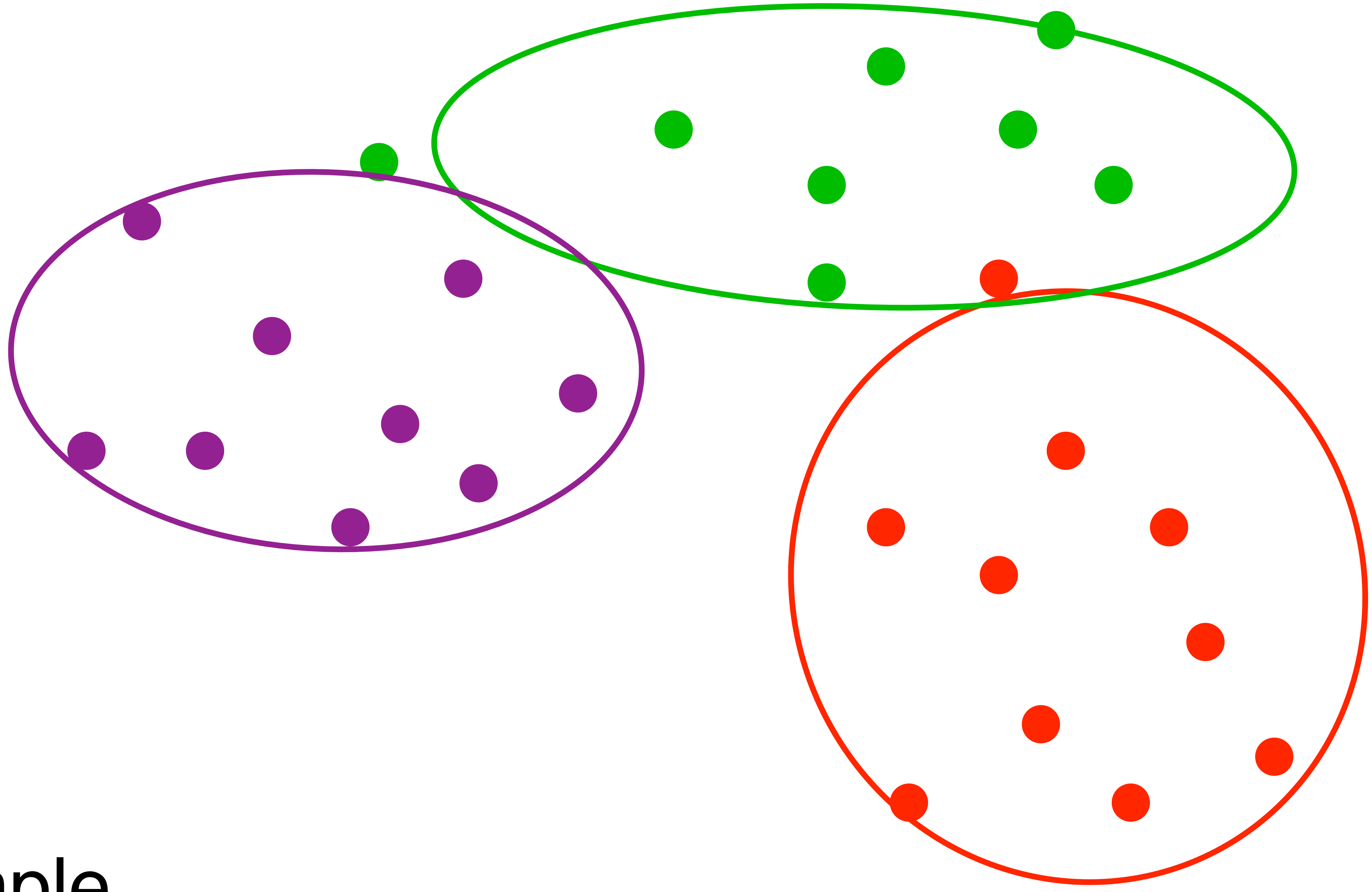
Carnegie Mellon University

Gibbs sampling for clustering



resample
cluster labels

Gibbs sampling for clustering



resample
cluster model

e.g. Mahout Dirichlet Process Clustering

Inference Algorithm ≠ Model

Corollary: EM ≠ Clustering

… but some algorithms and models are good match …

7.4 Models

7 Graphical Models

Alexander Smola

Introduction to Machine Learning 10-701

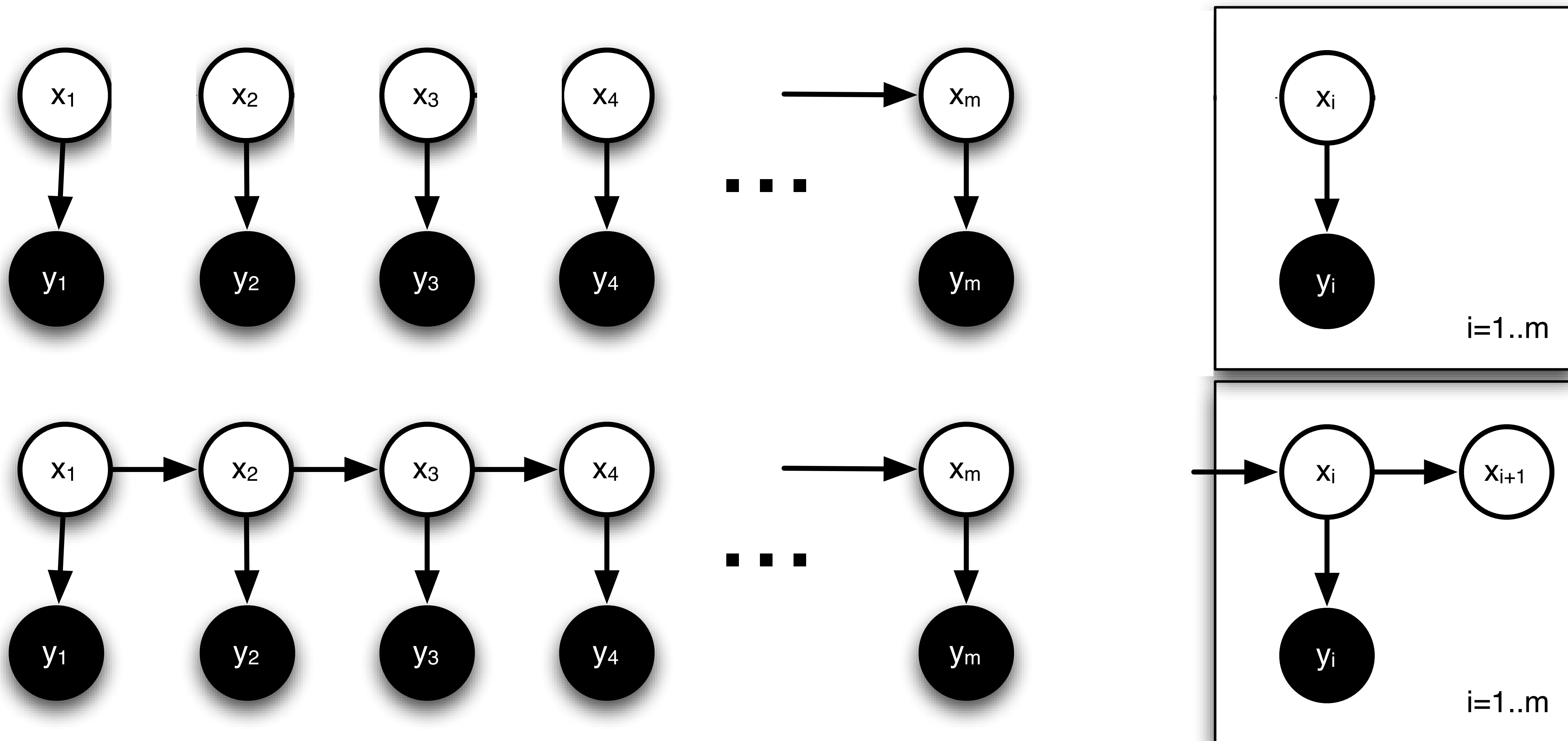
http://alex.smola.org/teaching/10-701-15

MACHINE LEARNING DEPARTMENT

Carnegie Mellon University

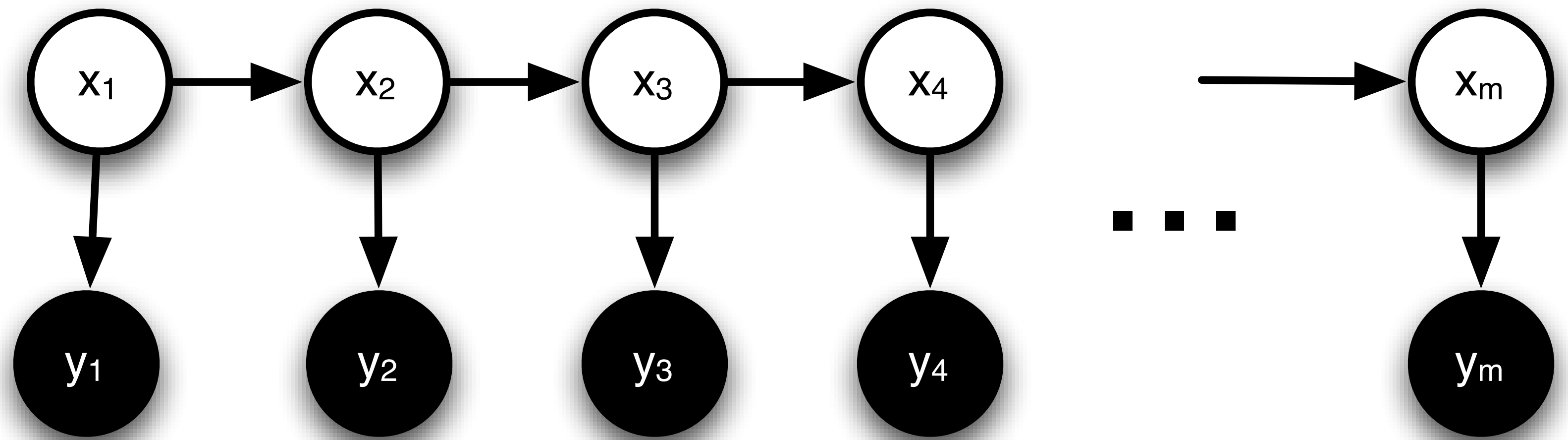
Hidden Markov Models

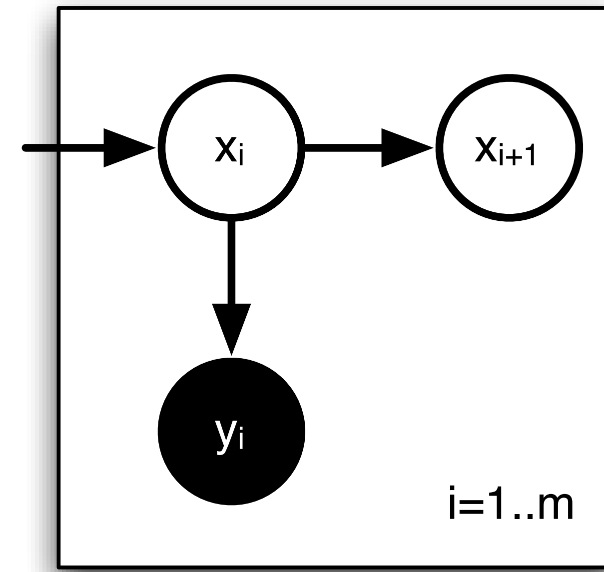
Clustering and Hidden Markov Models



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

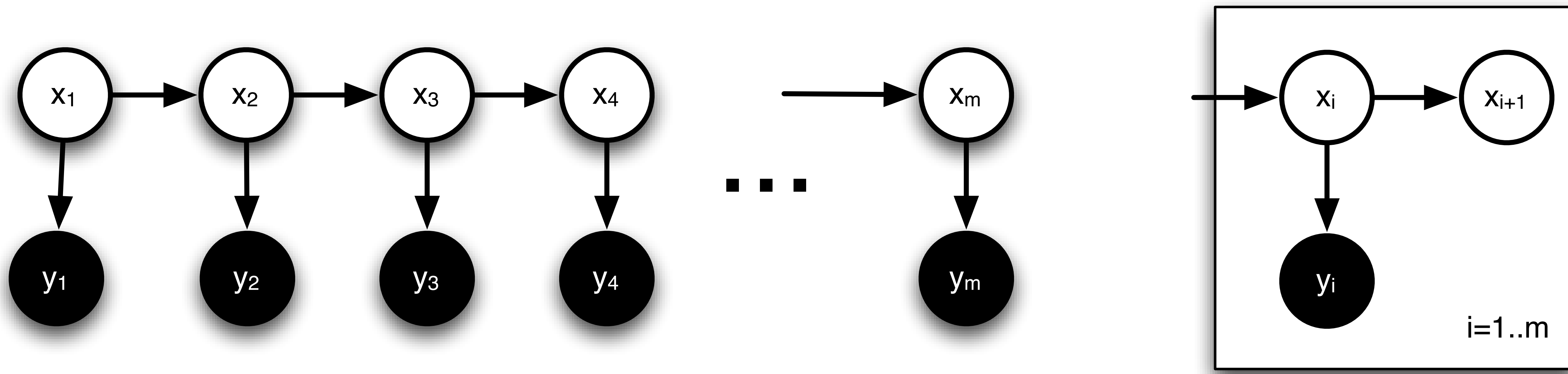
Applications





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

Inference



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

- Summing over y possible via dynamic programming
- Log-likelihood is nonconvex

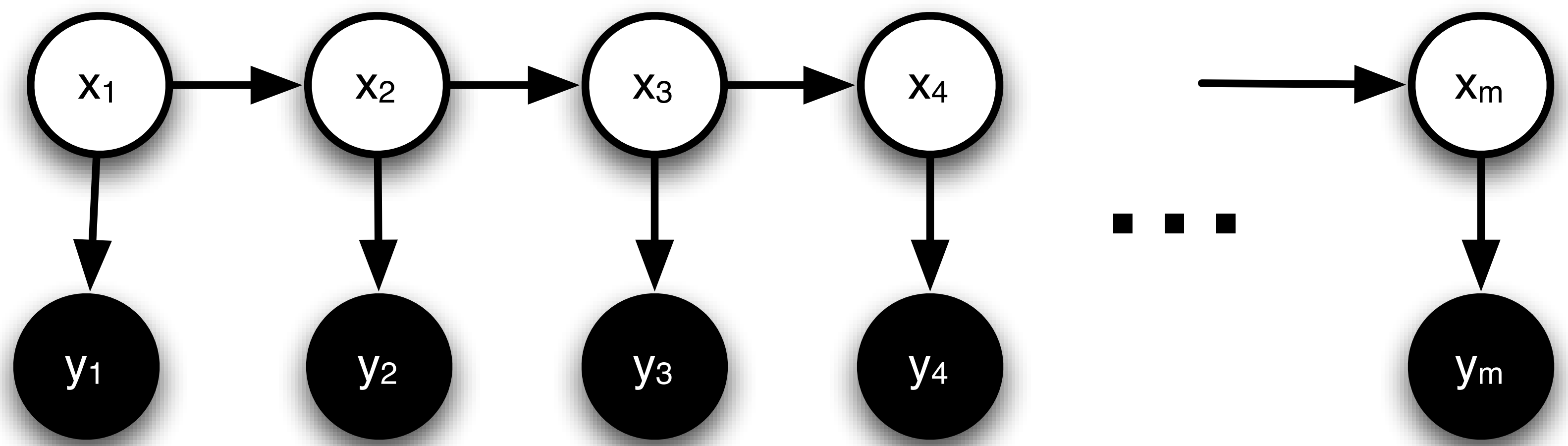
Variational Approximation

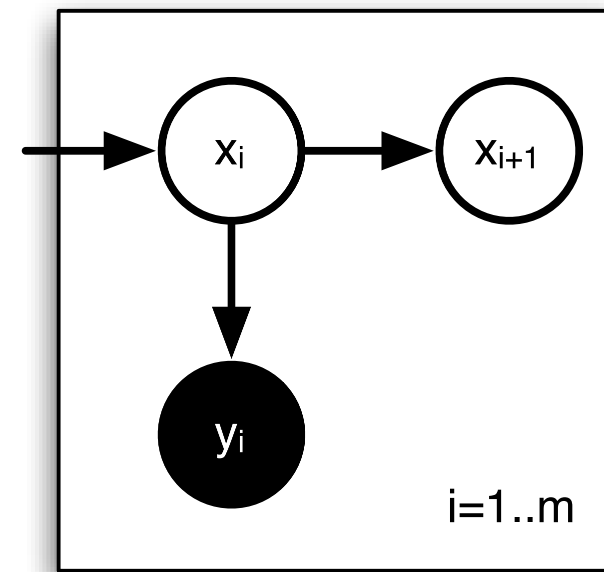
- Lower bound on log-likelihood

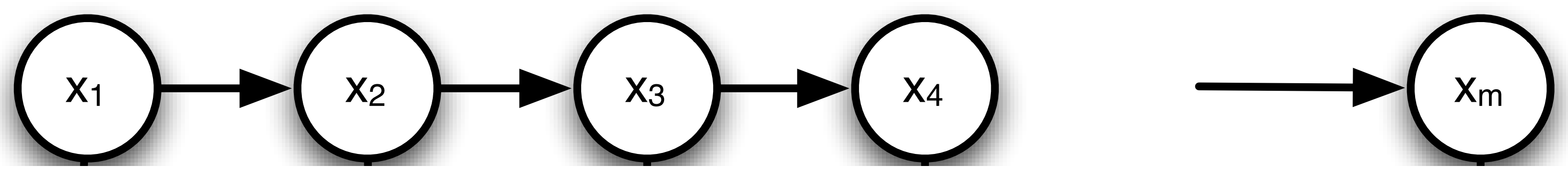
$$\log p(x;\theta) \geq \int dq(y) \log p(x,y;\theta) - \int dq(y) \log q(y)$$

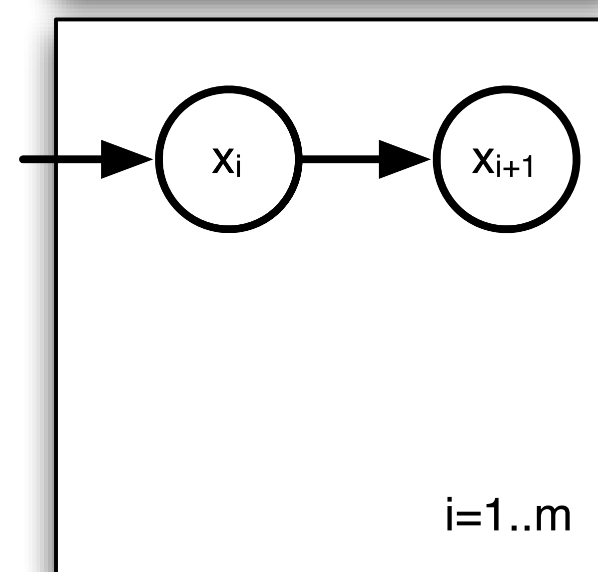
- Inequality holds for any q
  - Find q within subset Q to tighten inequality
  - Find parameters to maximize for fixed q
- Inference for graphical models where joint probability computation is infeasible

Variational Approximation









- Variational approximation via

$$q(y) = q(y_1) \prod_{x=2}^{m} q(y_i | y_{i-1})$$

- Compute p(xly) via dynamic programming

Variational Method

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

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

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

q(y1)

q(yi+1|yi)

q(yi)

Parameter Estimation

$$\mathbf{E}_{y\sim q}\left[\log p(x,y;\theta)\right] = \mathbf{E}_{y_1\sim q}\log p(y_1;\theta) + \sum_{i=1}^{\cdots}\mathbf{E}_{y_i\sim q}\log p(x_i|y_i;\theta)$$

$$+ \sum_{i=1}^{m-1}\mathbf{E}_{y_{i+1},y_i\sim q}\log p(y_{i+1}|y_i;\theta)$$

- $p(y_1)$
  Since we have $\mathbf{E}_{q(y_1)}[\log p(y_1)]$ set $p(y_1) = q(y_1)$
- $p(x_i|y_i)$
  Same as clustering
  e.g. for Gaussians

$$\mu_y = \frac{1}{n_y}\sum_{i=1}^{n} q_i(y)x_i$$

$$\Sigma_y = \frac{1}{n_y}\sum_{i=1}^{n} q_i(y)x_i x_i^\top - \mu_y\mu_y^\top$$

Parameter Estimation

$$\mathbf{E}_{y\sim q}\left[\log p(x, y; \theta)\right] = \mathbf{E}_{y_1 \sim q} \log p(y_1; \theta) + \sum_{i=1} \mathbf{E}_{y_i \sim q} \log p(x_i | y_i; \theta)$$

$$+ \sum_{i=1}^{m-1} \mathbf{E}_{y_{i+1}, y_i \sim q} \log p(y_{i+1} | y_i; \theta)$$

- Maximum likelihood estimate for p(y'ly)

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

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

effective sample

Smoothed Estimates

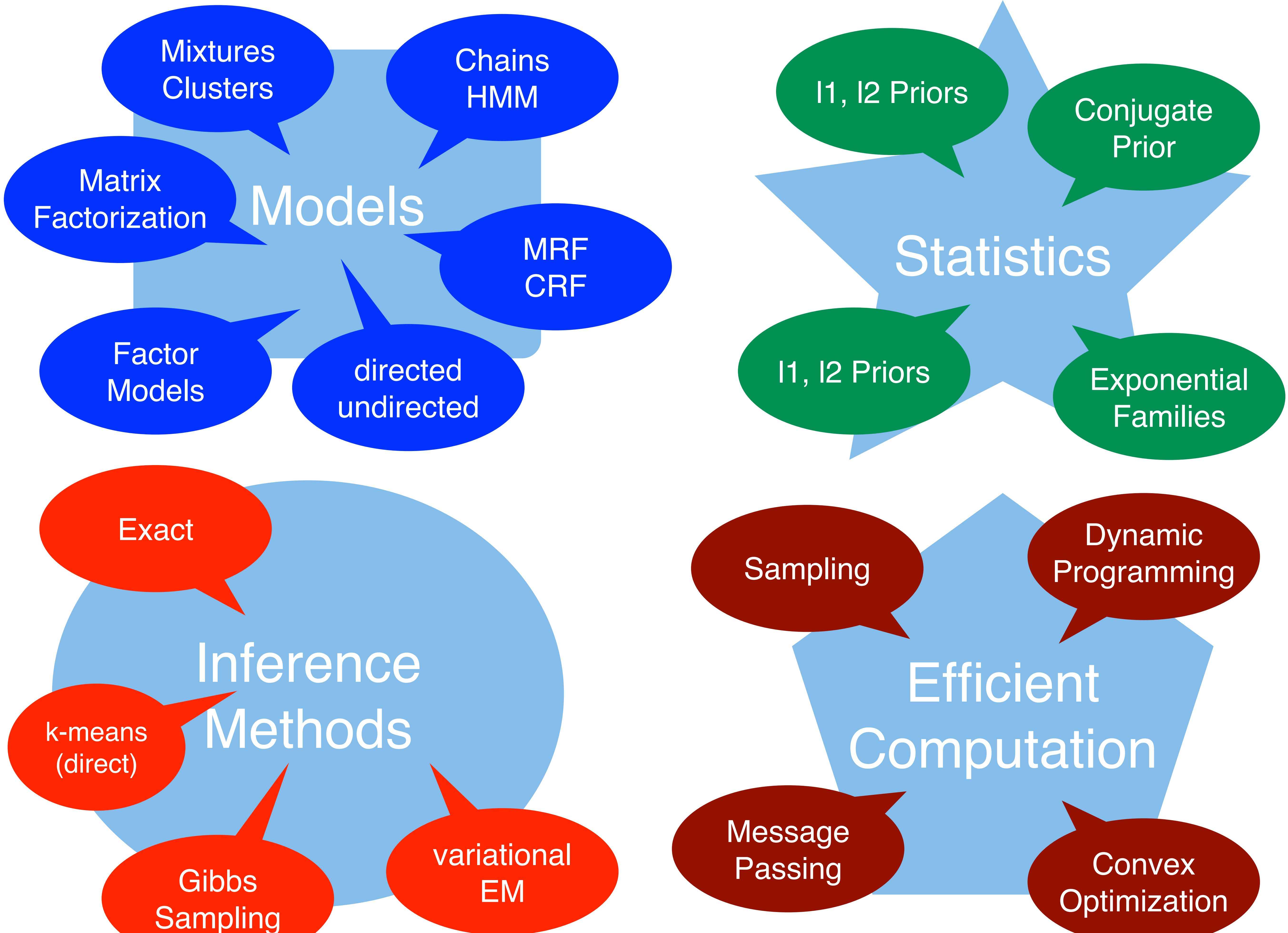
- Laplace prior on latent state distribution
  - Uniform distribution over states
  - Alternatively assume that state remains

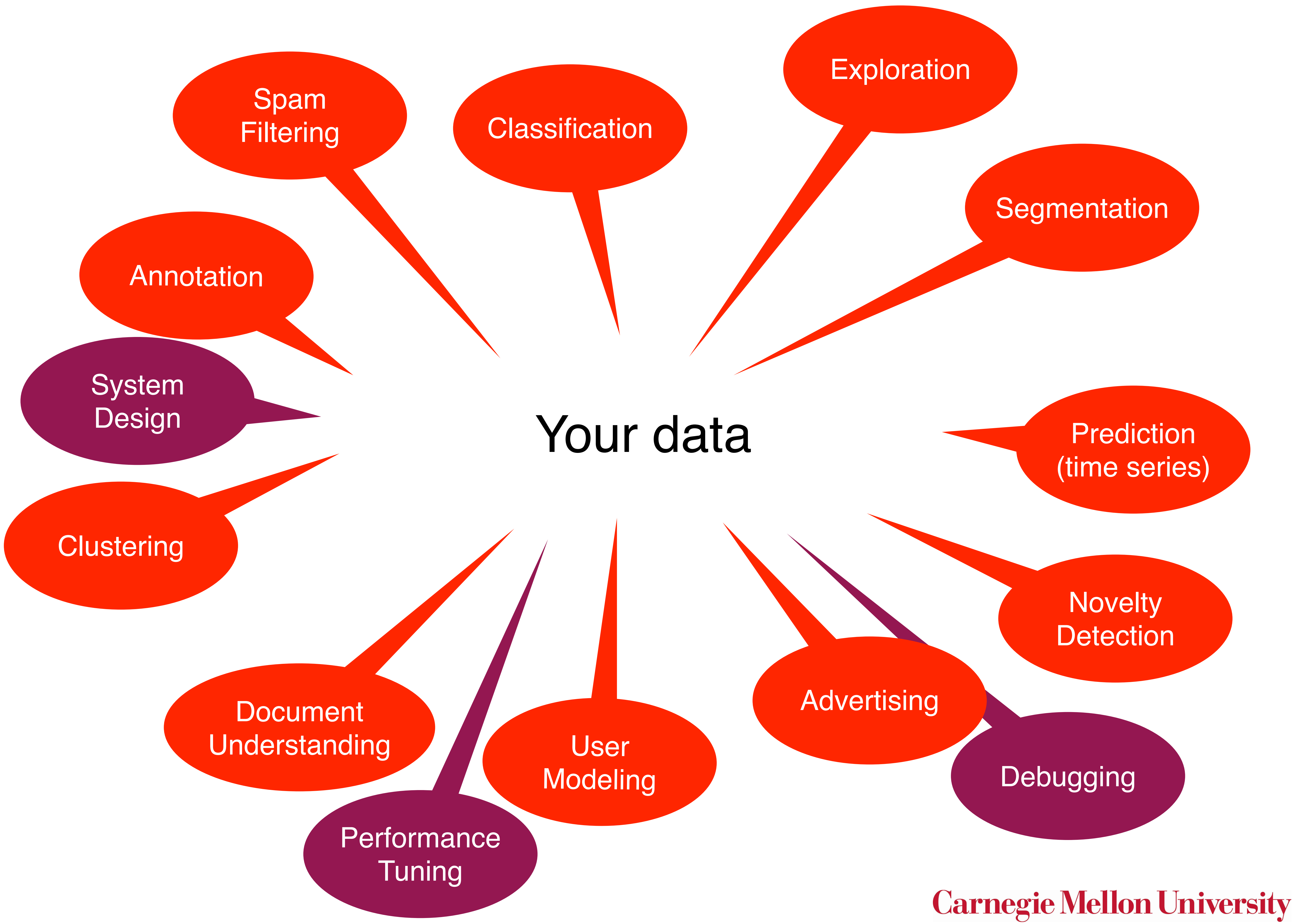
$$p(a|b) = \frac{n_{a|b} + \sum_{i=1}^{m-1} q(y_{i+1} = a, y_i = b)}{n_b + \sum_{i=1}^{m-1} q(y_i = b)}$$

transition smoother

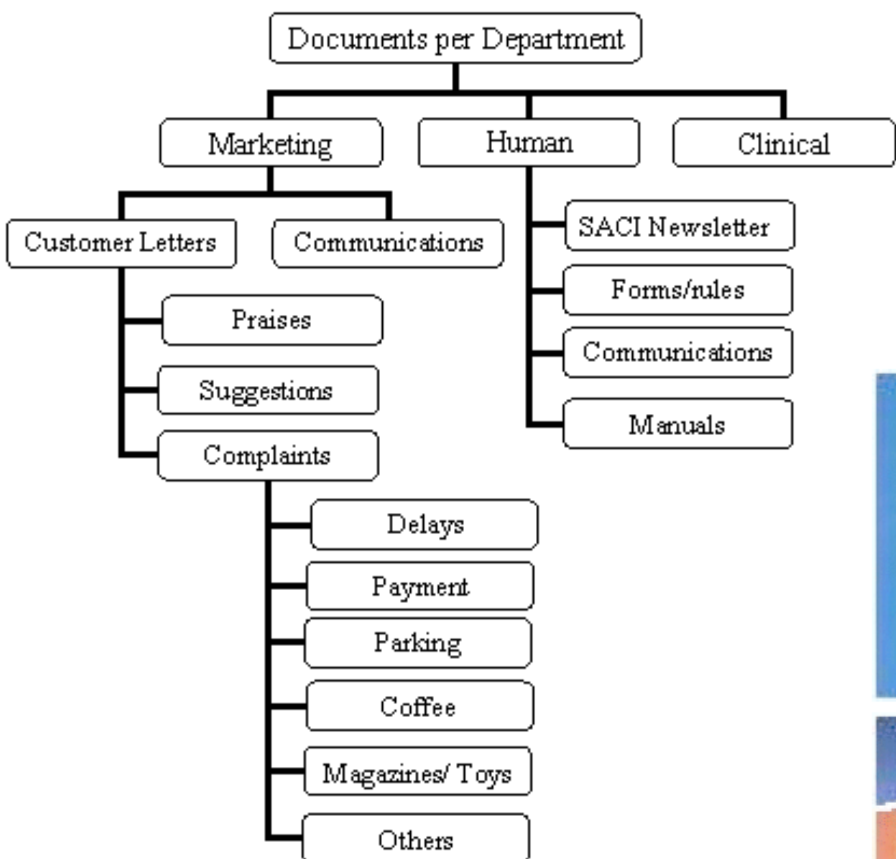
aggregate mass

Graphical Model Zoology

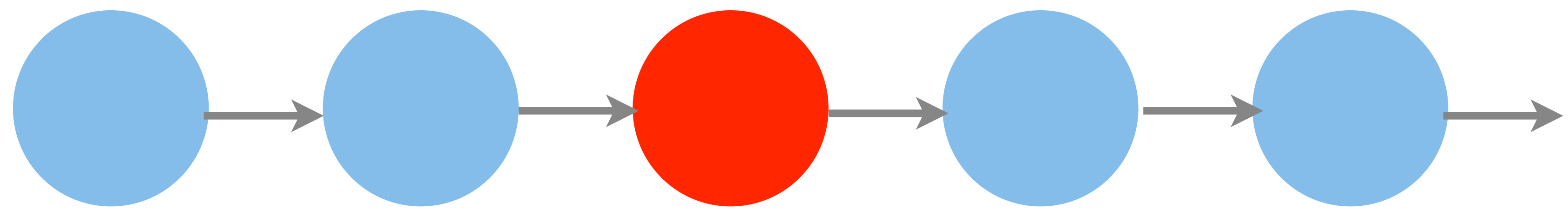

Models
Mixtures
Clusters
Chains
HMM
Matrix
Factorization
MRF
CRF
Factor
Models
directed
undirected
Statistics
l1, l2 Priors
Conjugate
Prior
l1, l2 Priors
Exponential
Families
Inference
Methods
Exact
k-means
(direct)
Gibbs
Sampling
variational
EM
Efficient
Computation
Sampling
Dynamic
Programming
Message
Passing
Convex
Optimization


Your data
Spam Filtering
Classification
Exploration
Segmentation
Annotation
System Design
Prediction (time series)
Clustering
Novelty Detection
Document Understanding
Performance Tuning
User Modeling
Advertising
Debugging
Carnegie Mellon University

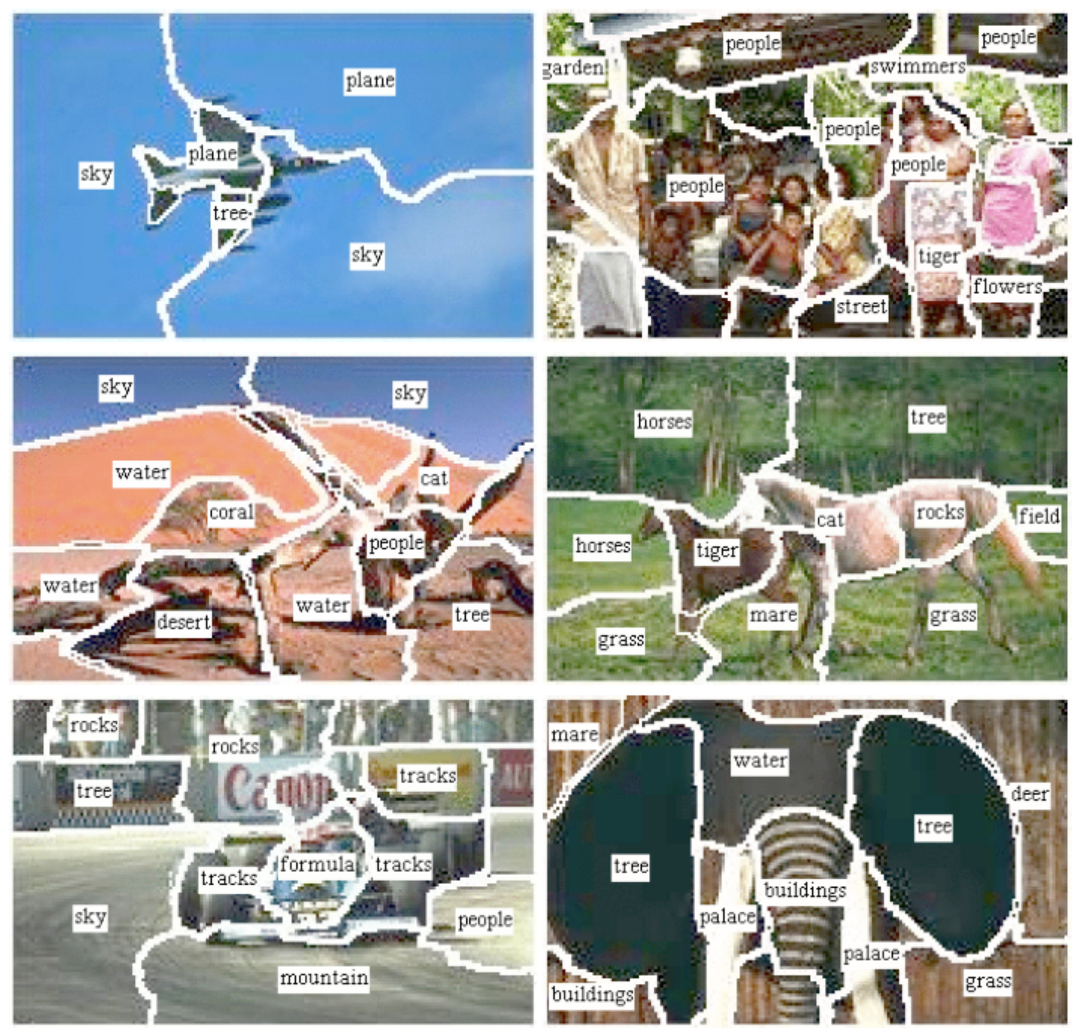
Beyond mixtures



taxonomies



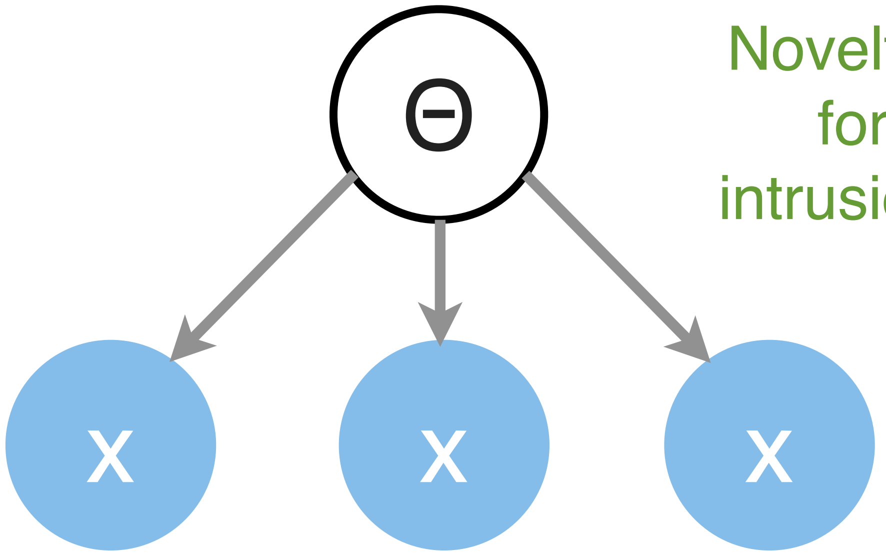
chains



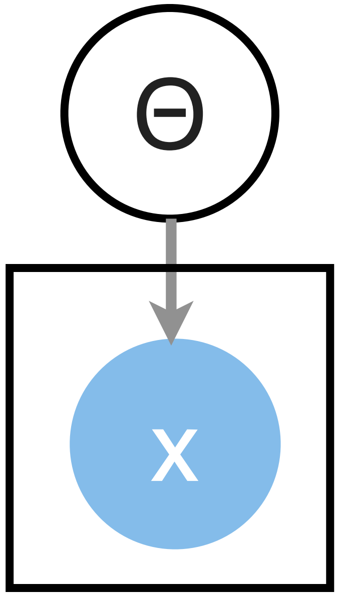
topics

‘Unsupervised’ Models

Density Estimation



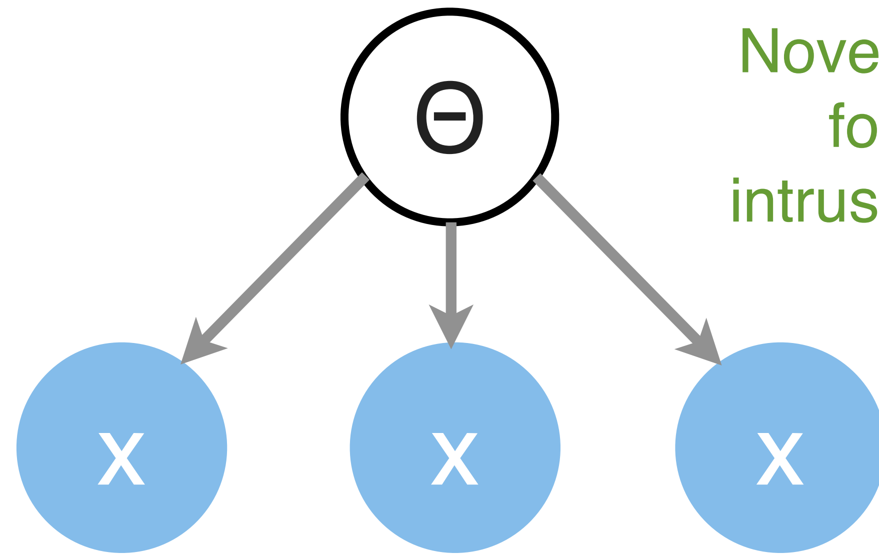
Novelty Detection
forecasting
intrusion detection



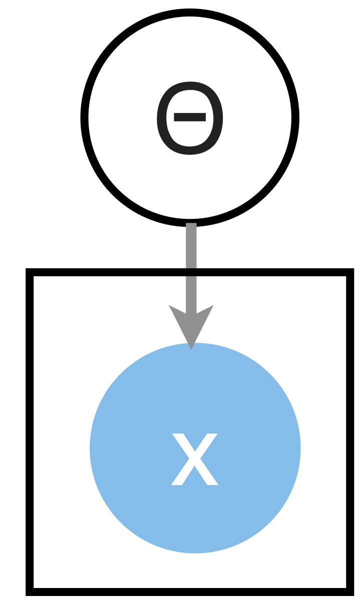
webpages
news
users
ads
queries
images

‘Unsupervised’ Models

Density Estimation

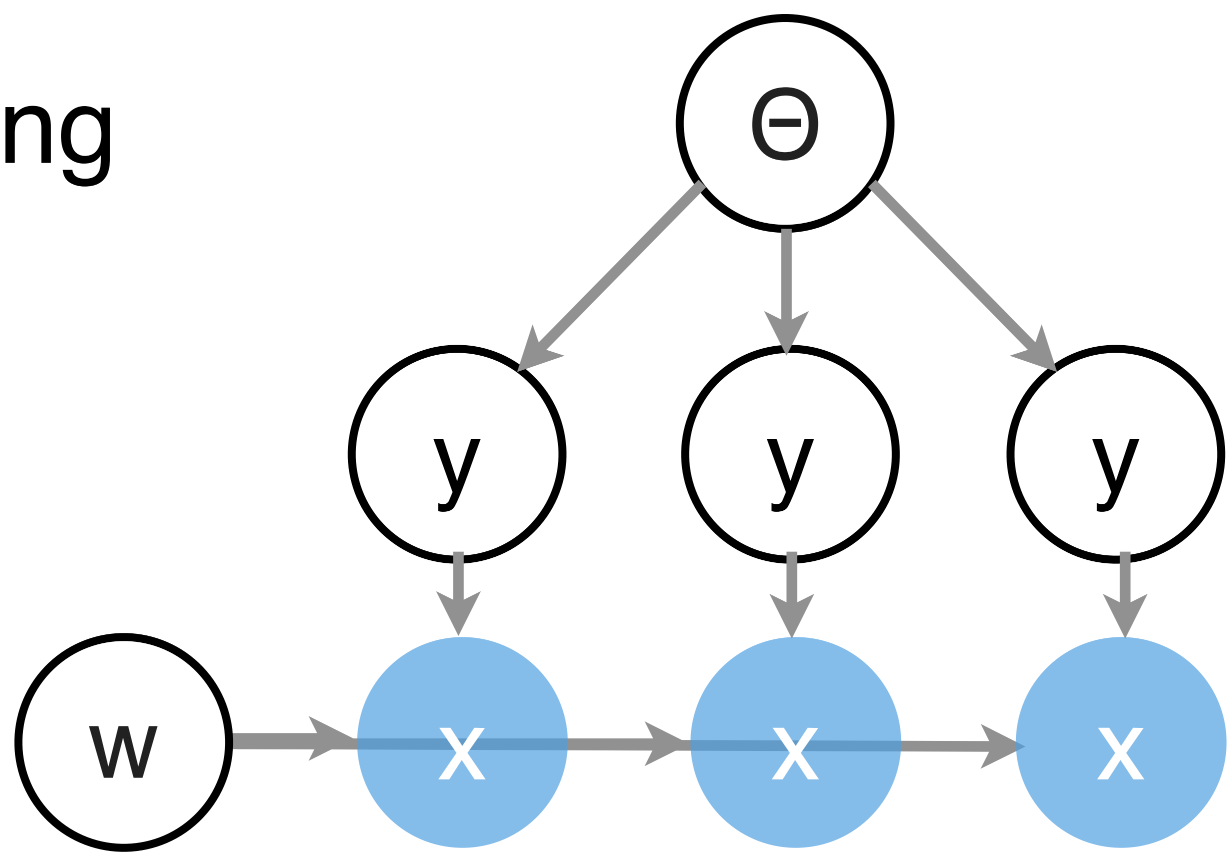


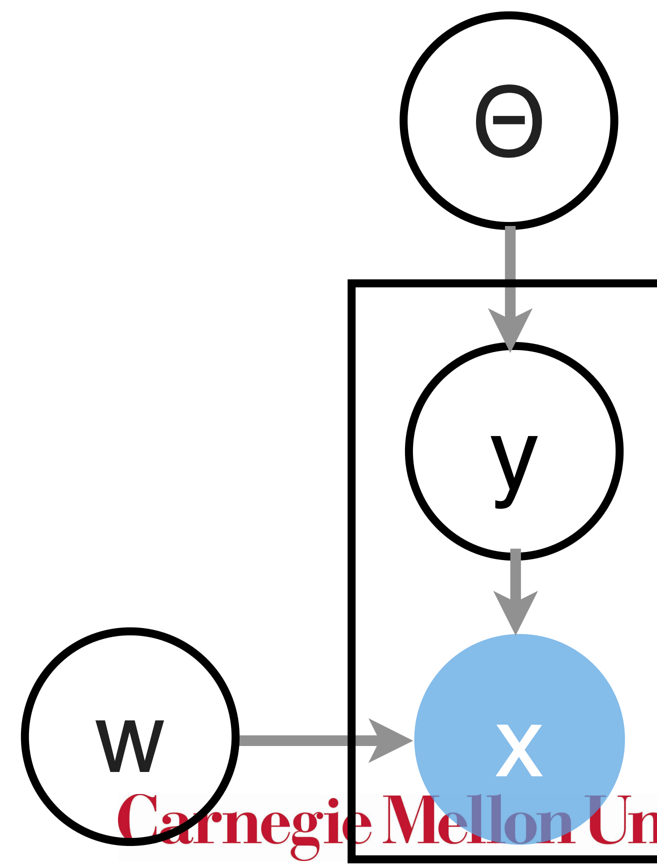
Novelty Detection
forecasting
intrusion detection



Clustering

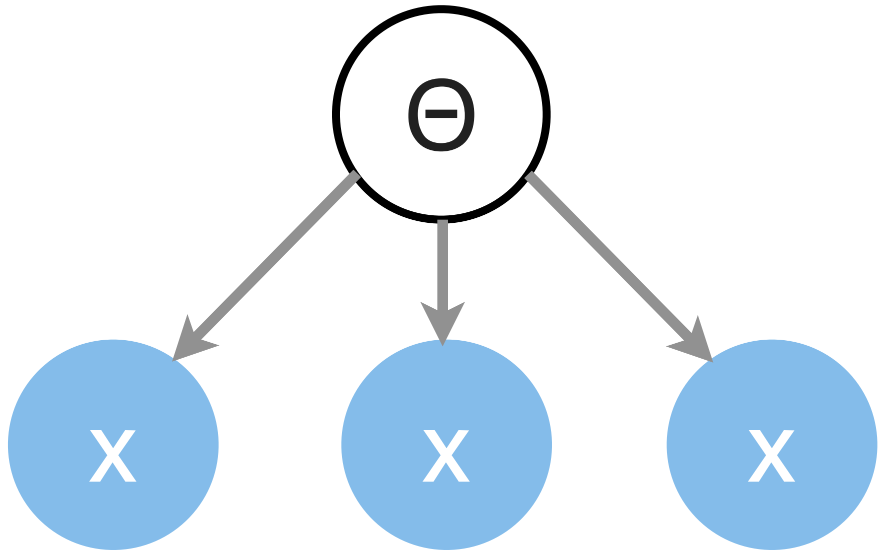
webpages
news
users
ads
queries
images

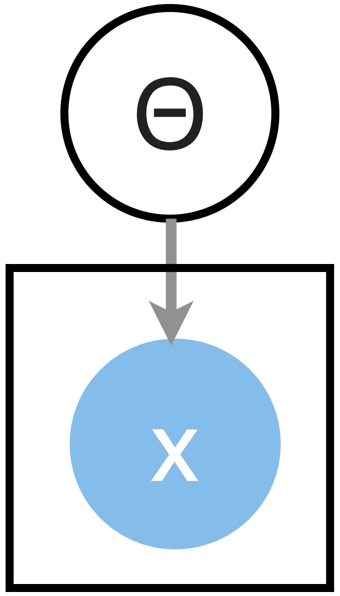




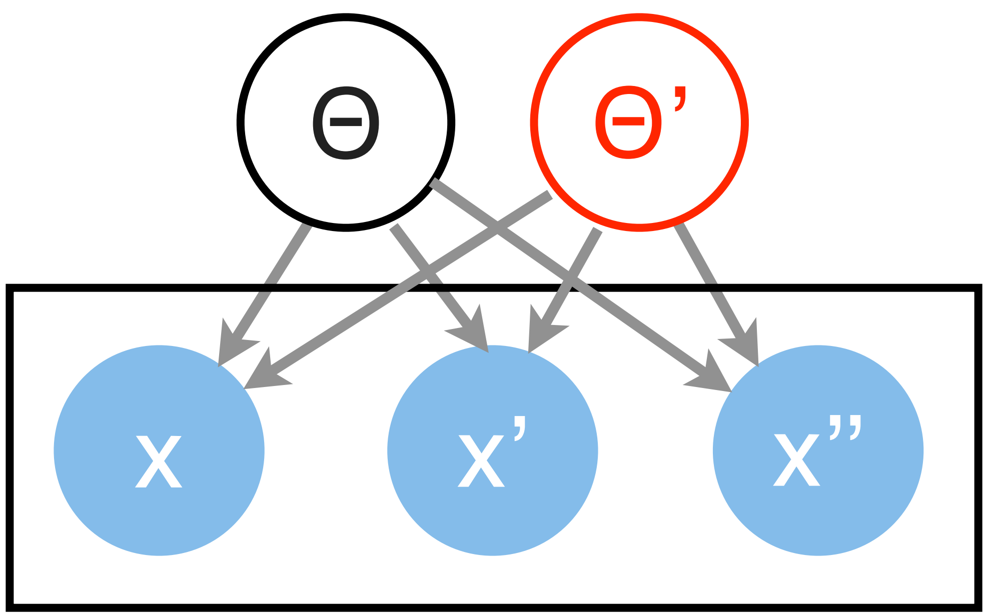
‘Unsupervised’ Models

Density Estimation





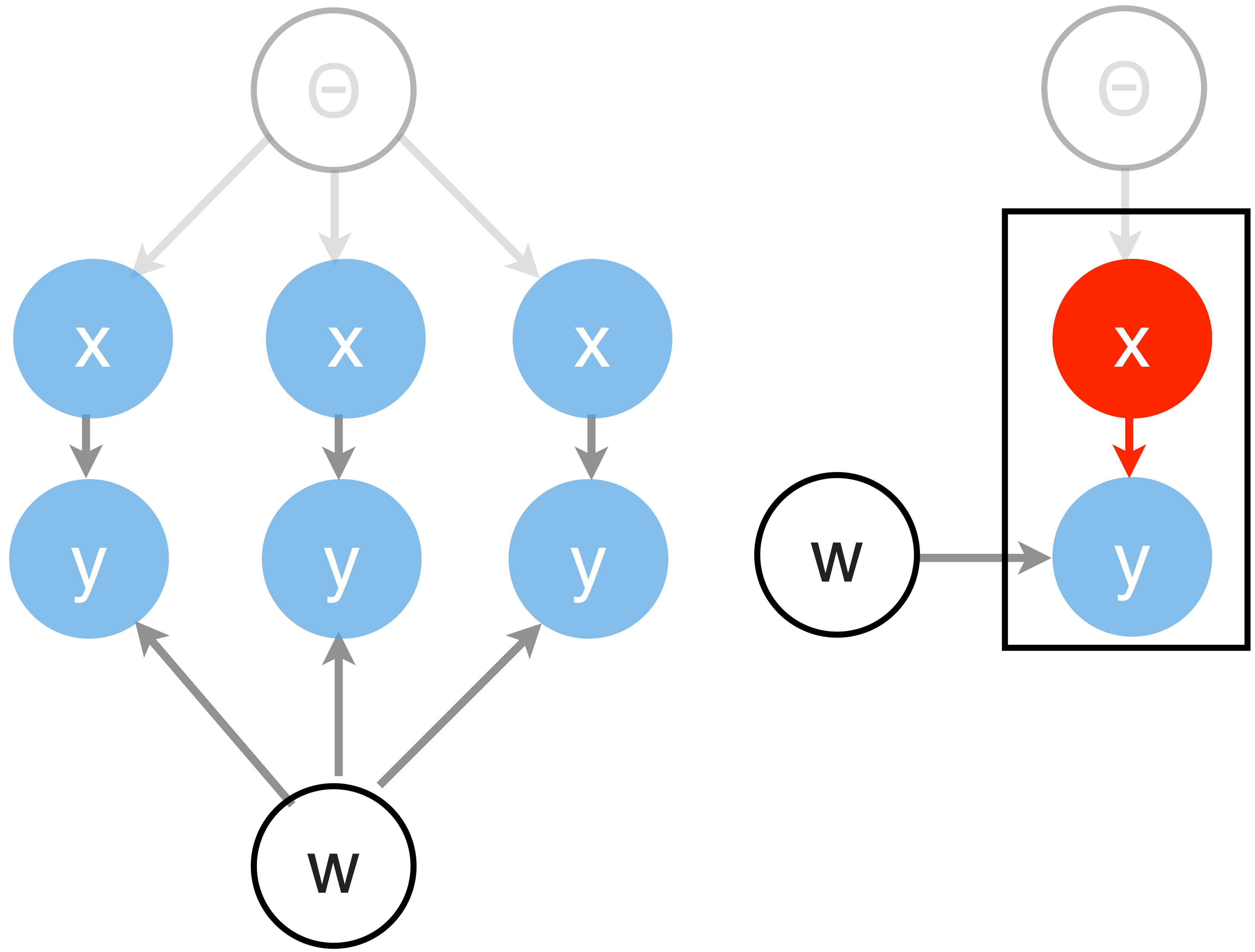
Factor Analysis



‘Supervised’ Models

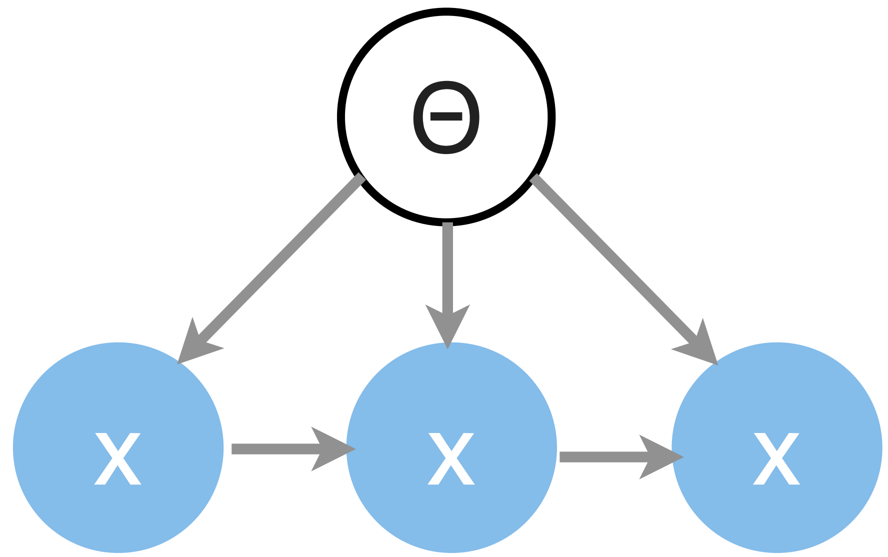
Classification
Regression

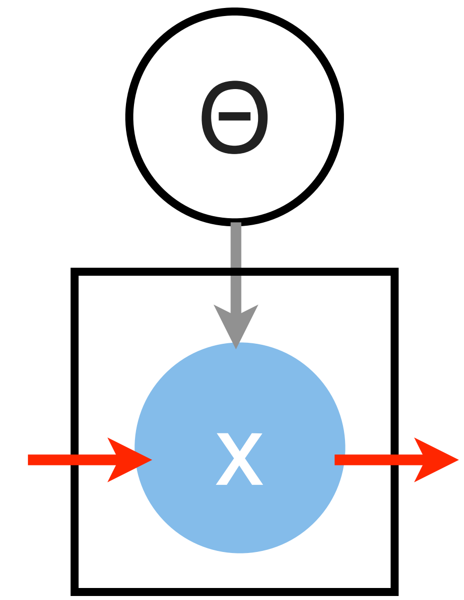
spam filtering
tiering
crawling
categorization
bid estimation
tagging



Chains

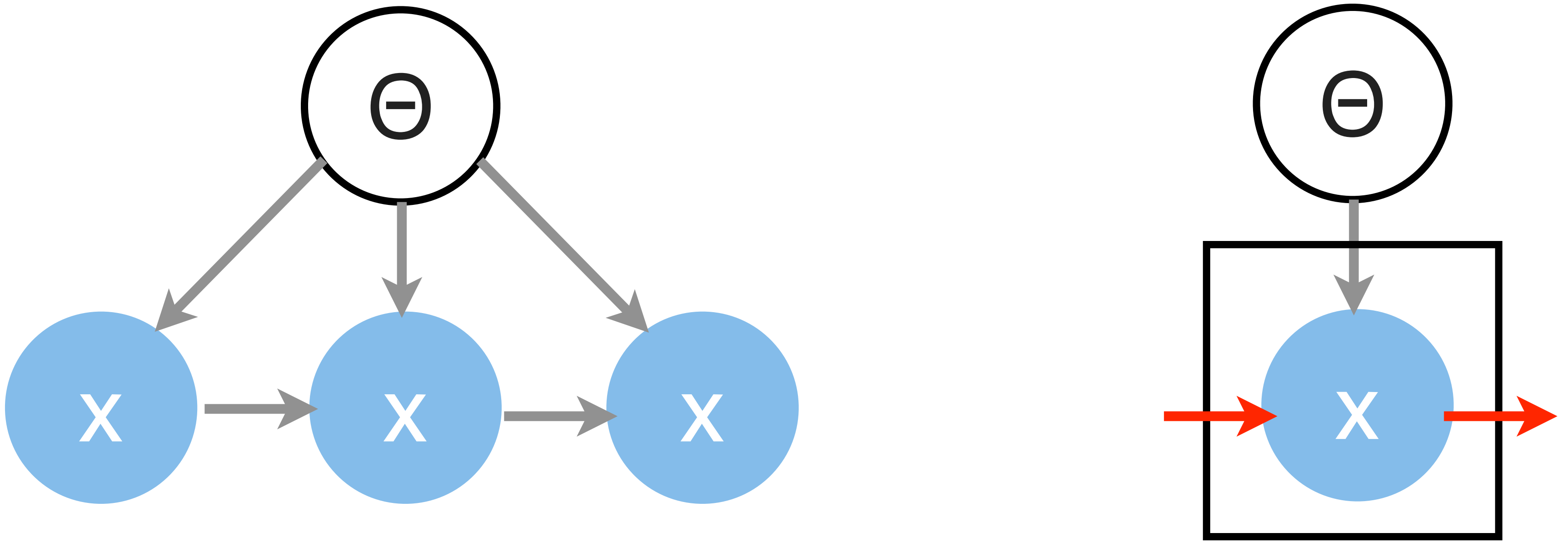
Markov Chain



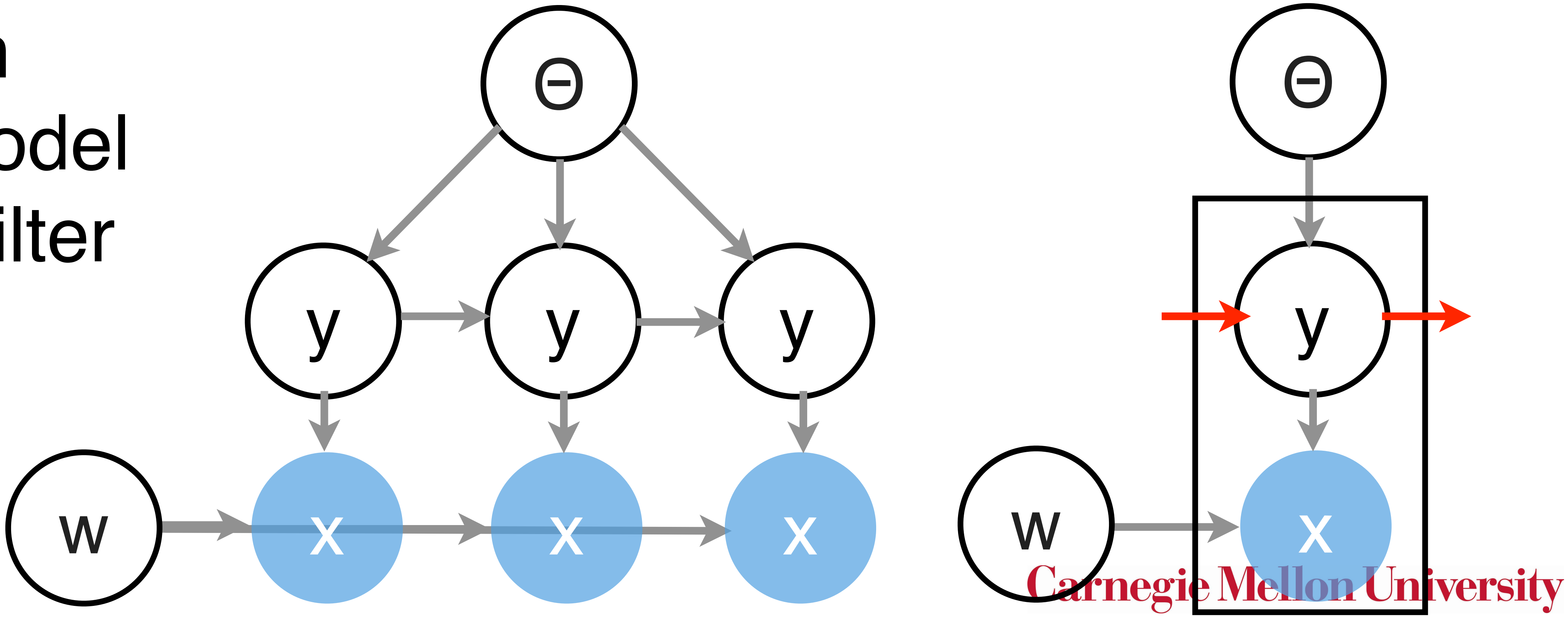


Chains

Markov Chain



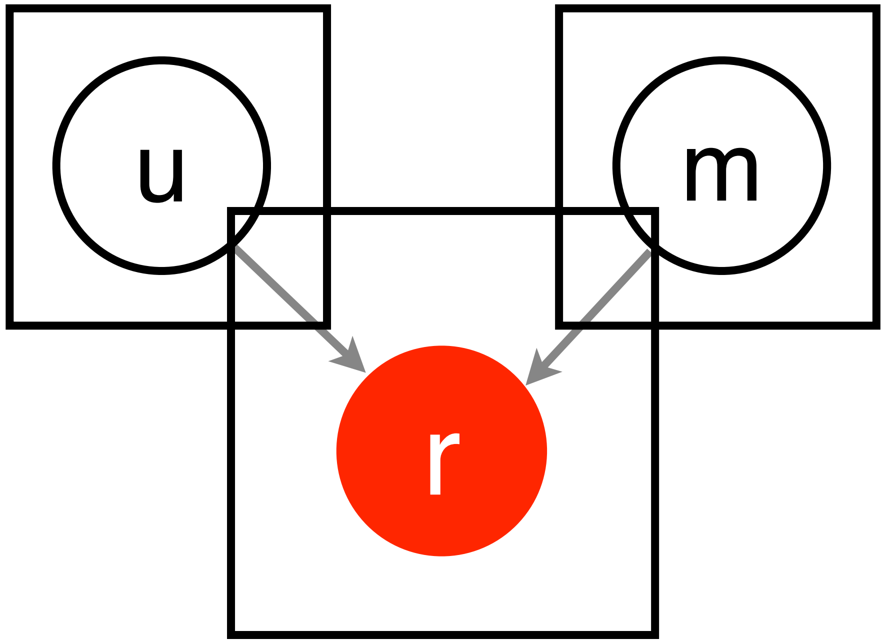
Hidden Markov Model
Kalman Filter



Collaborative Models

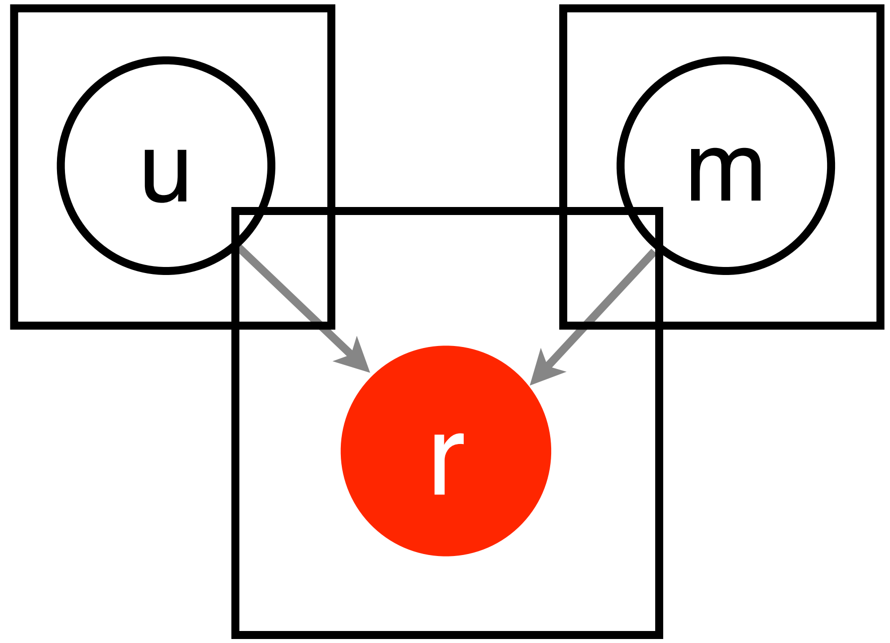
Collaborative Models

Collaborative Filtering

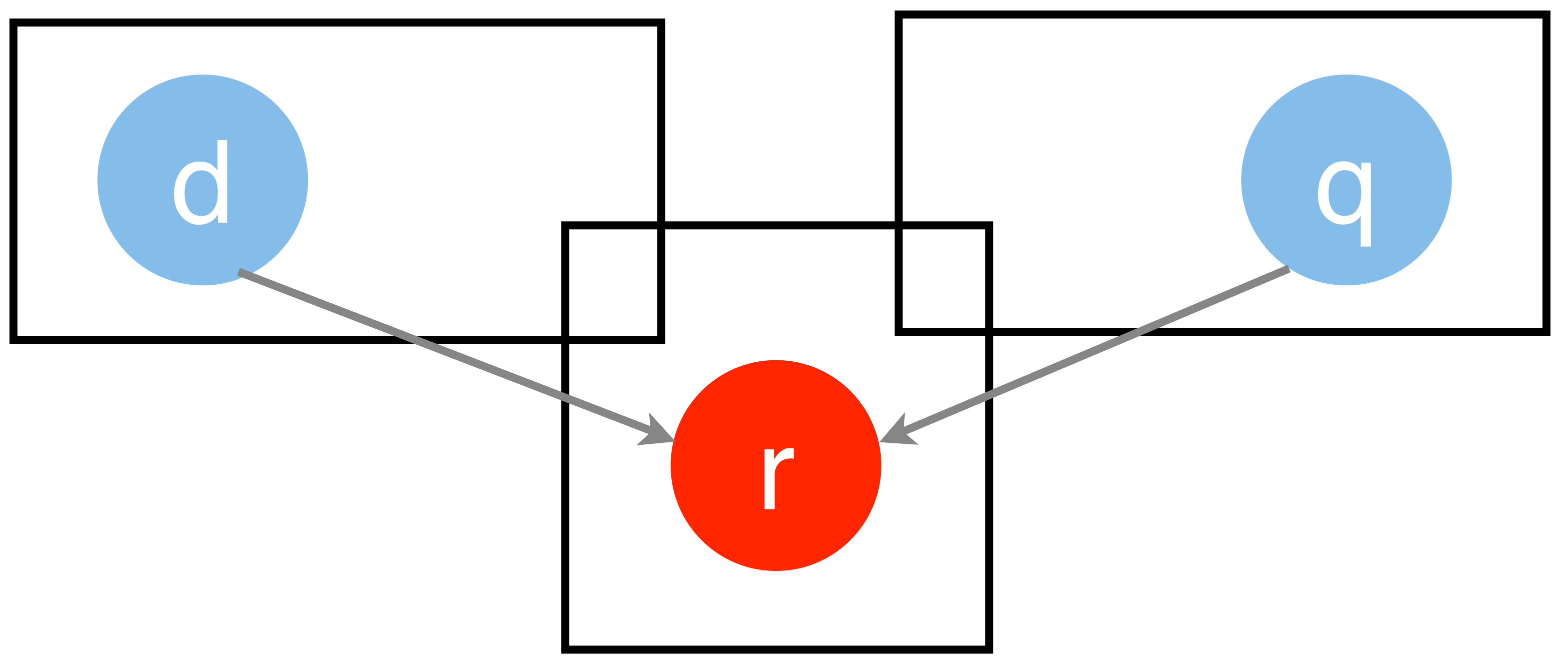


Collaborative Models

Collaborative Filtering

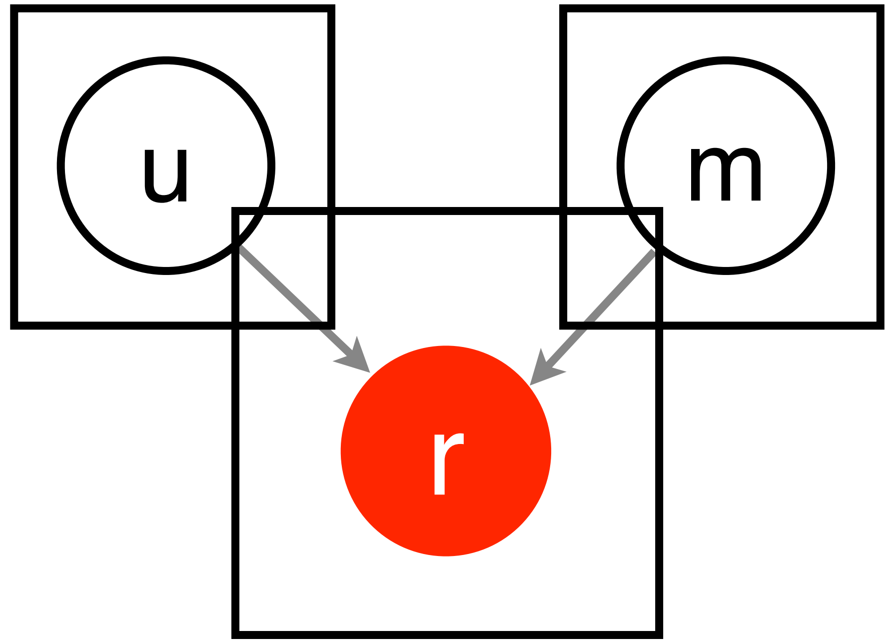


Current Webpage Ranking

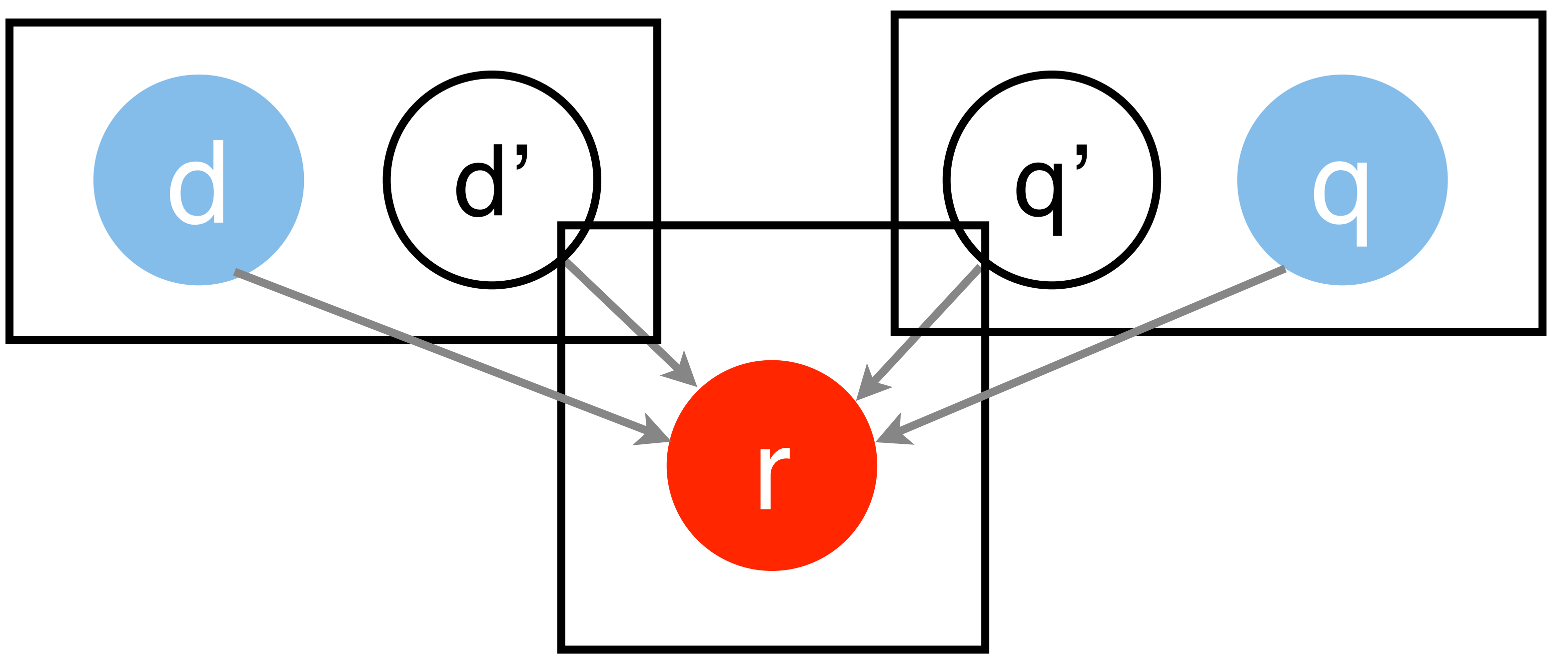


Collaborative Models

Collaborative Filtering

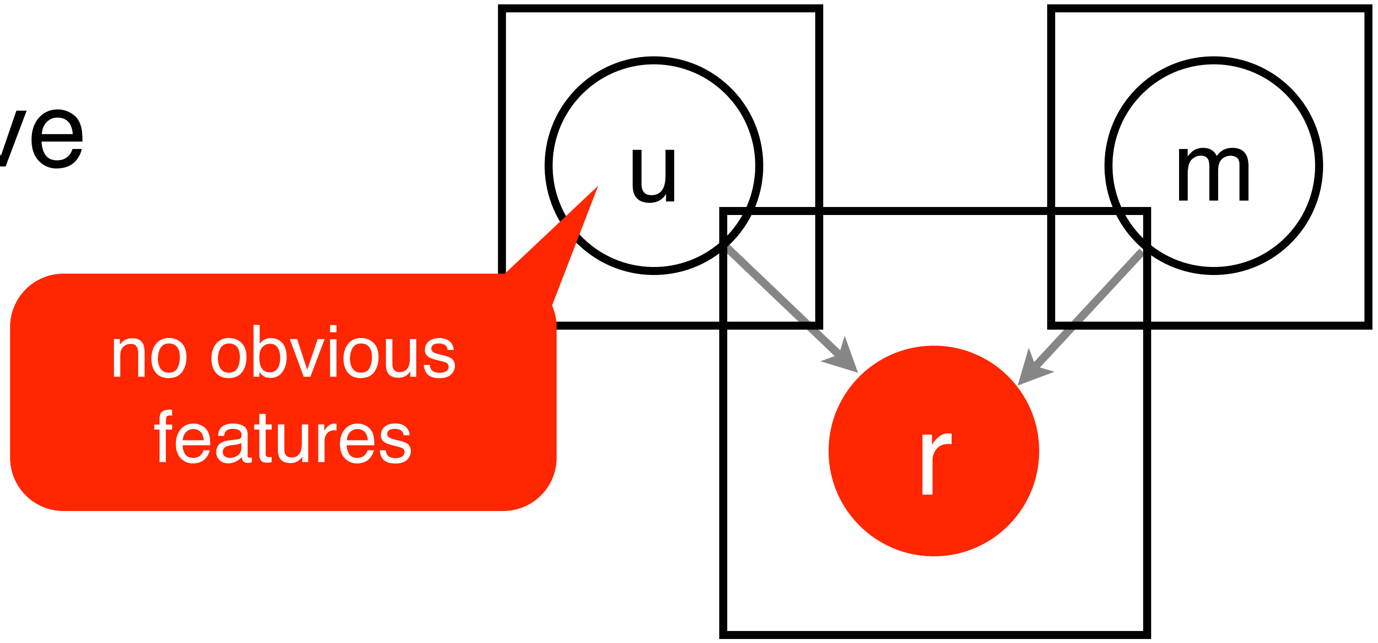


Webpage Ranking

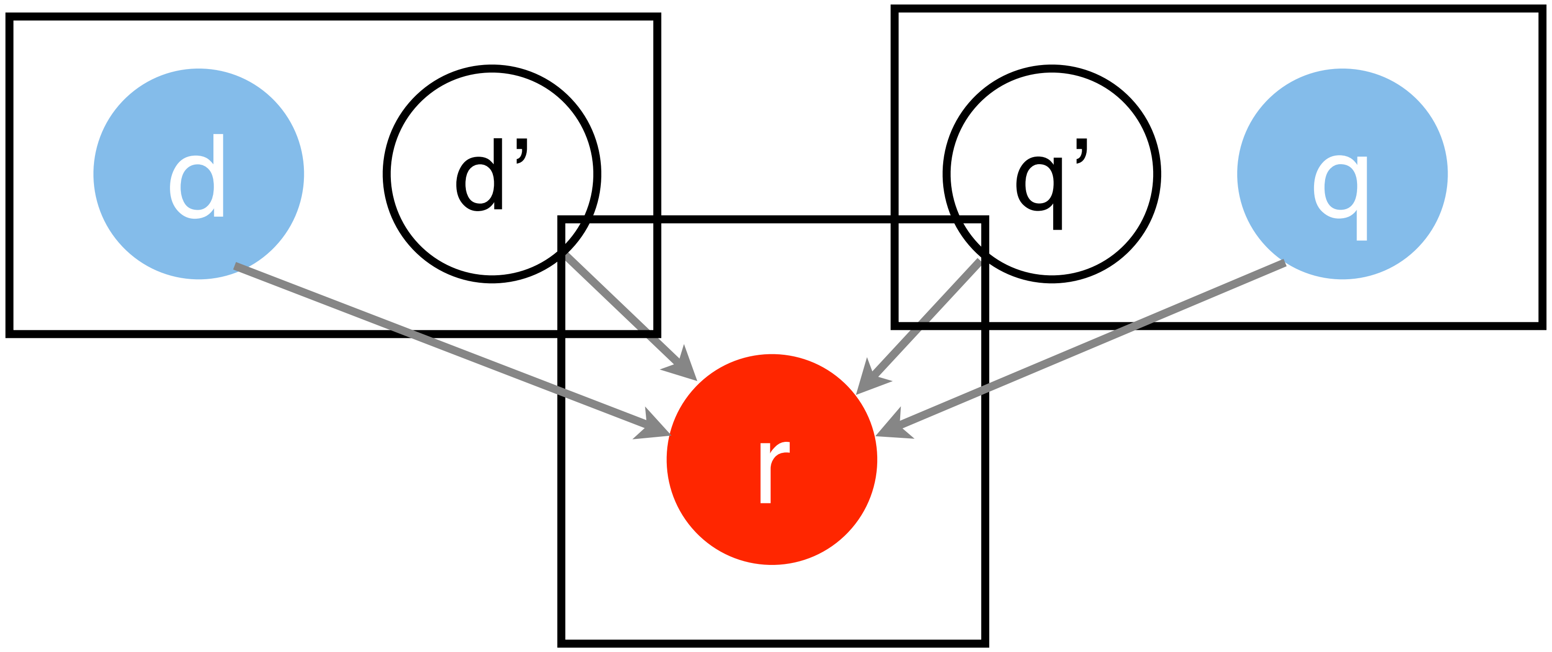


Collaborative Models

Collaborative Filtering

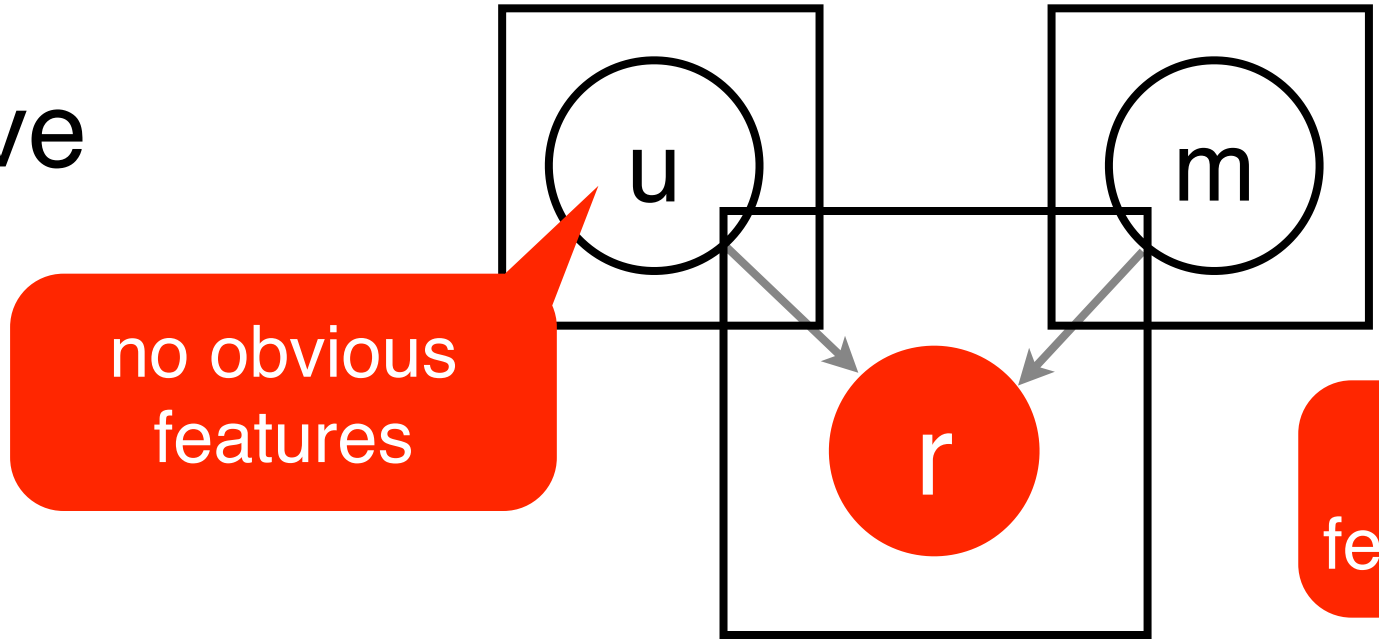


Webpage Ranking

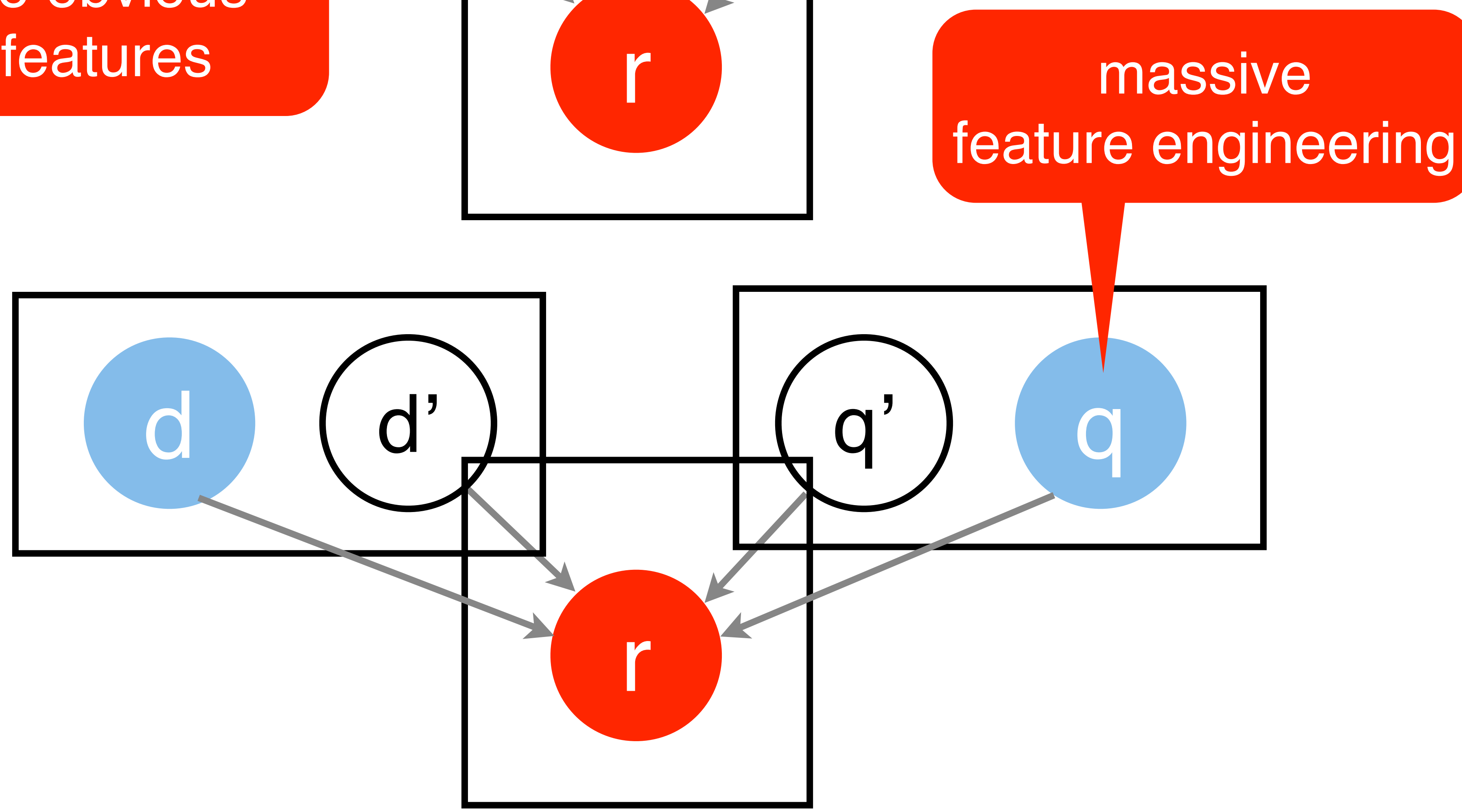


Collaborative Models

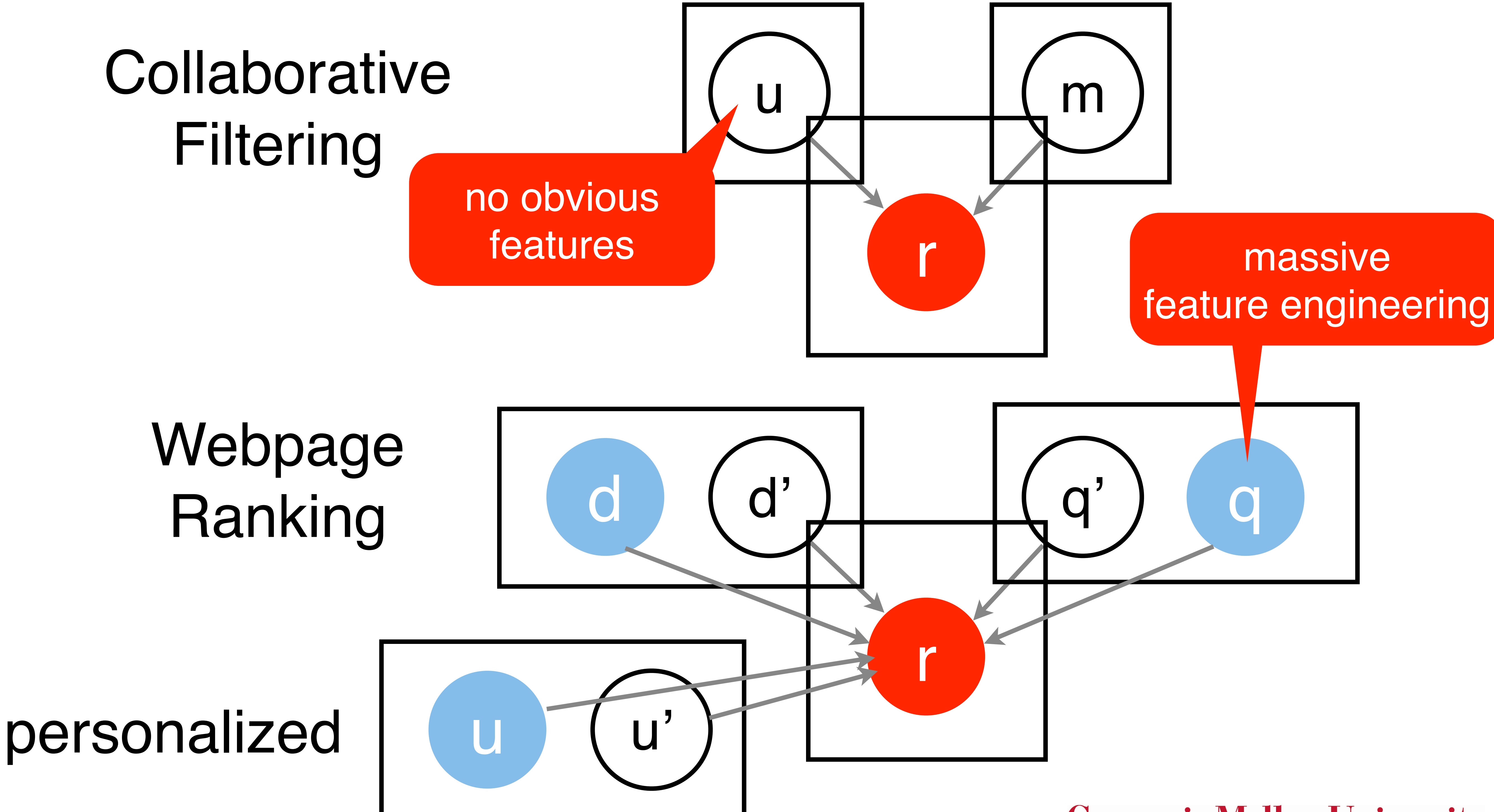
Collaborative Filtering



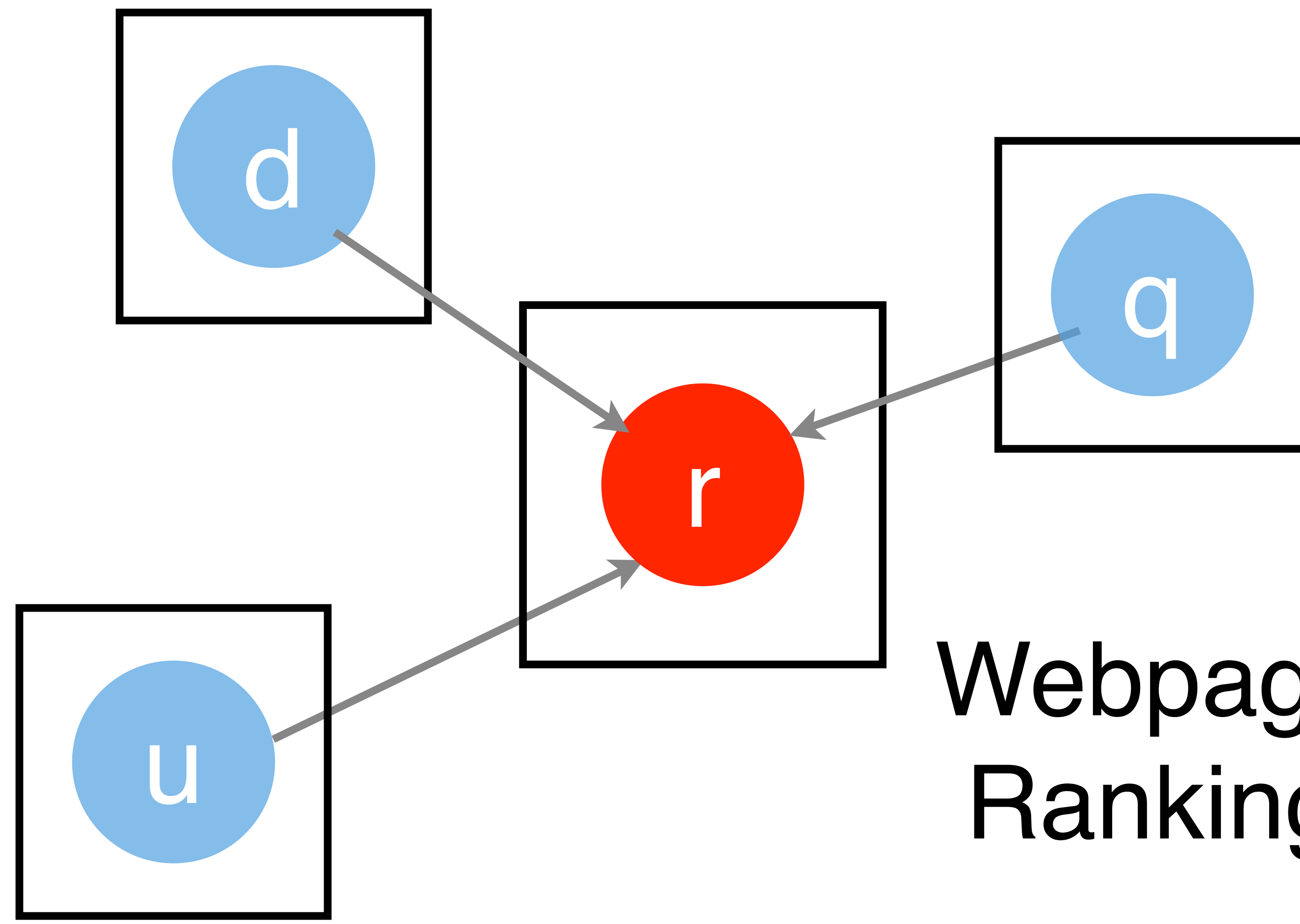
Webpage Ranking



Collaborative Models

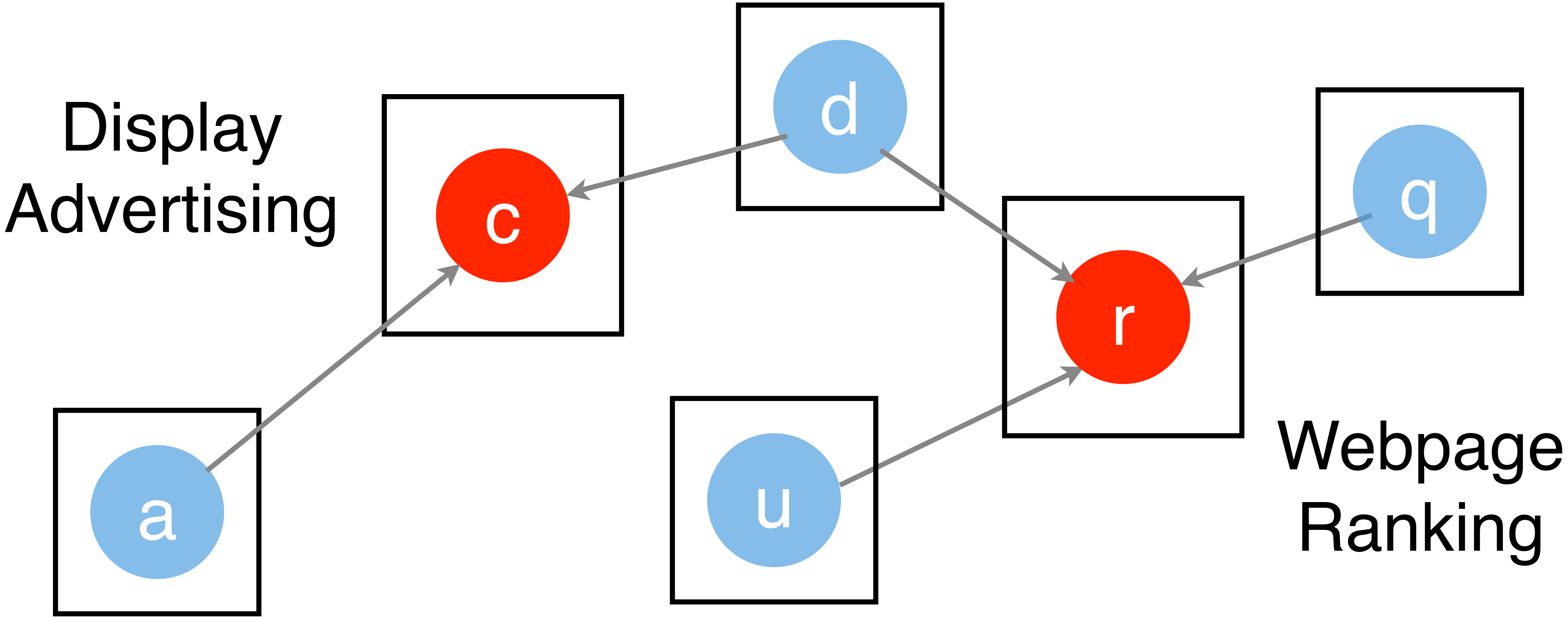


Data Integration

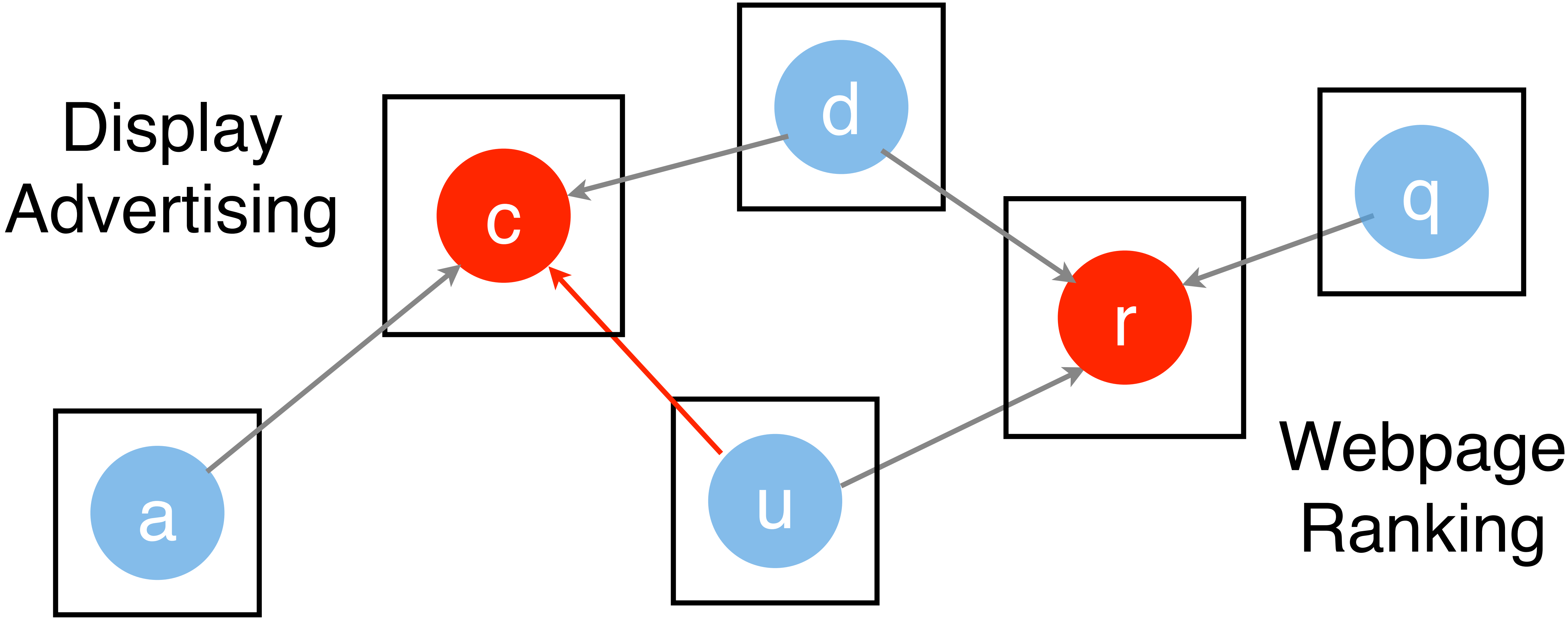


Webpage Ranking

Data Integration

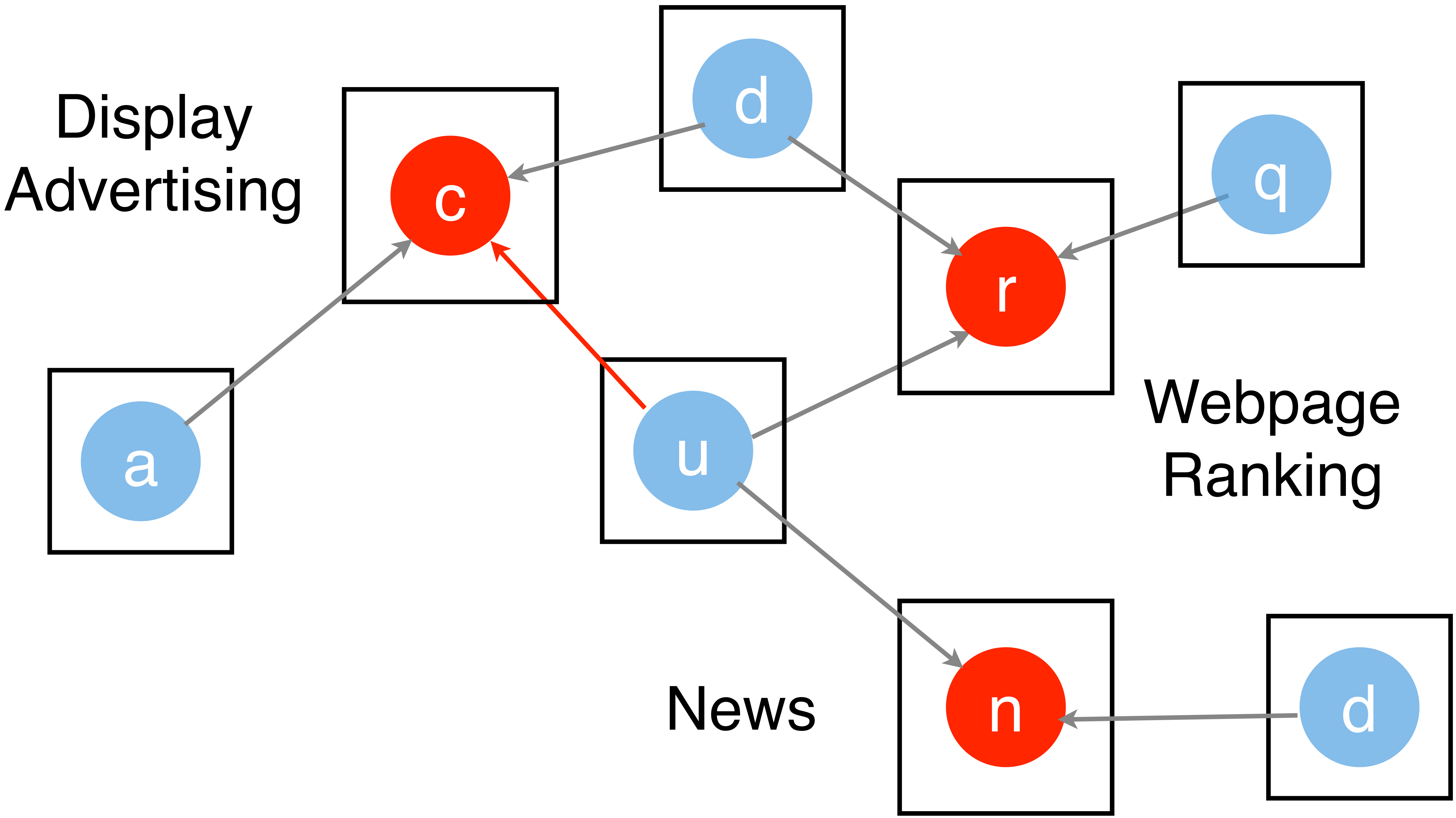


Data Integration

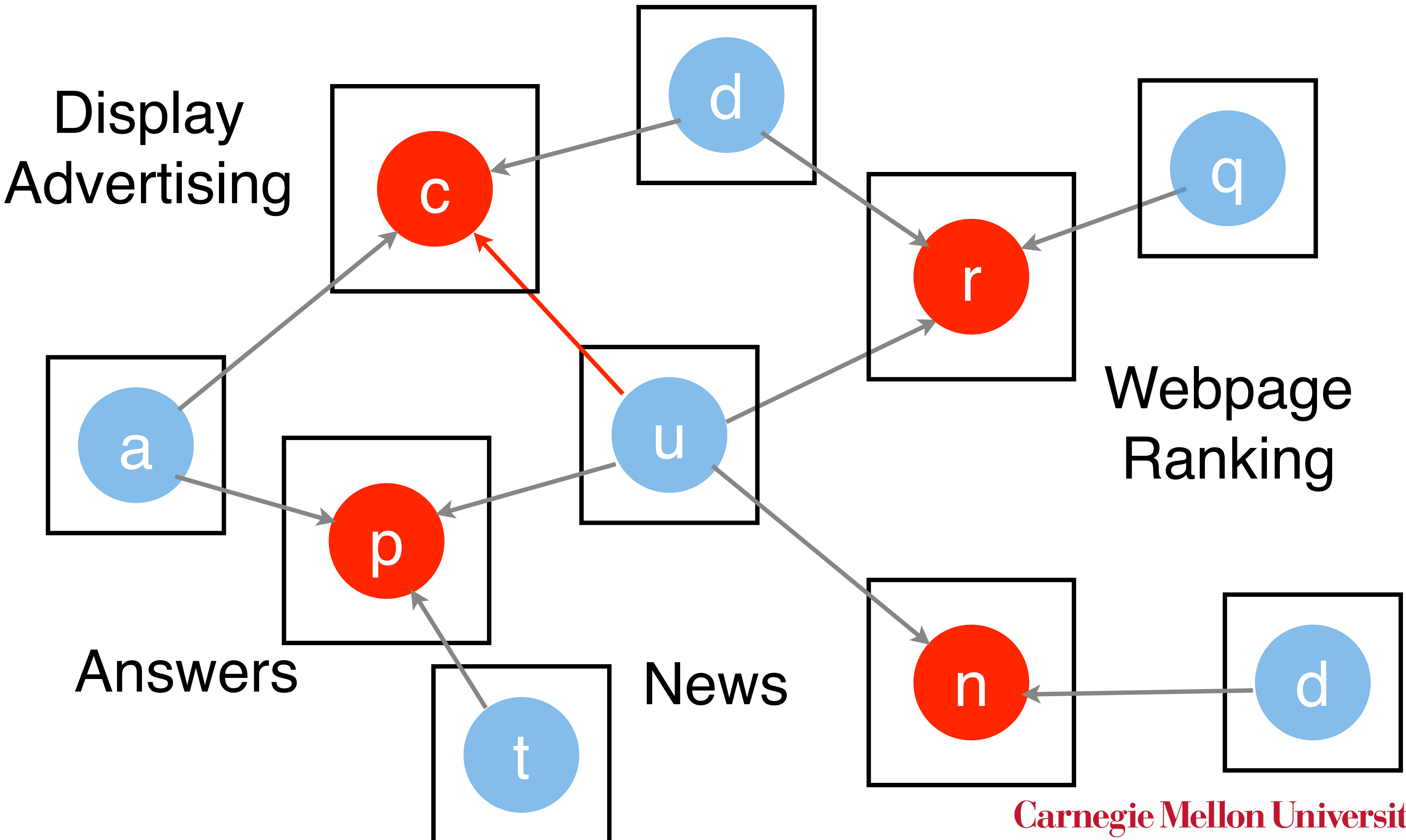


Carnegie Mellon University

Data Integration



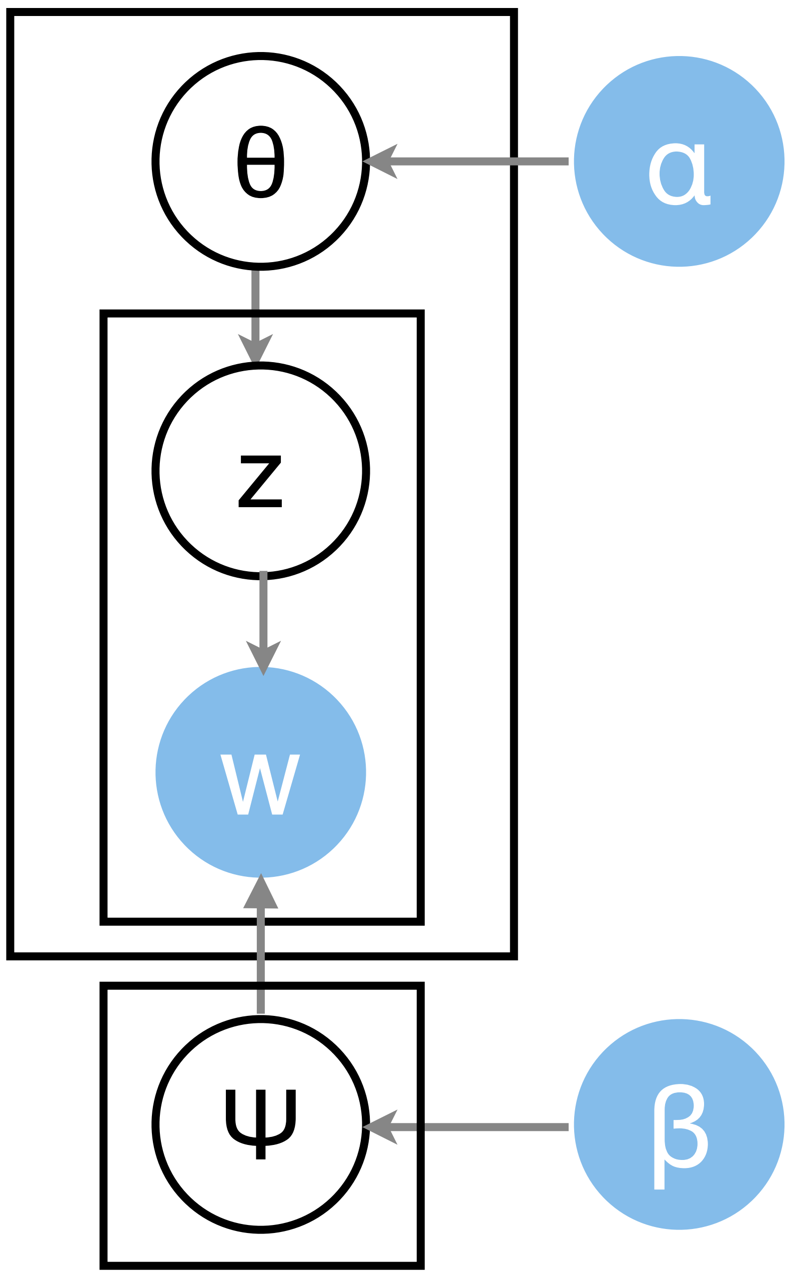
Data Integration



Topic Models

Topic Models

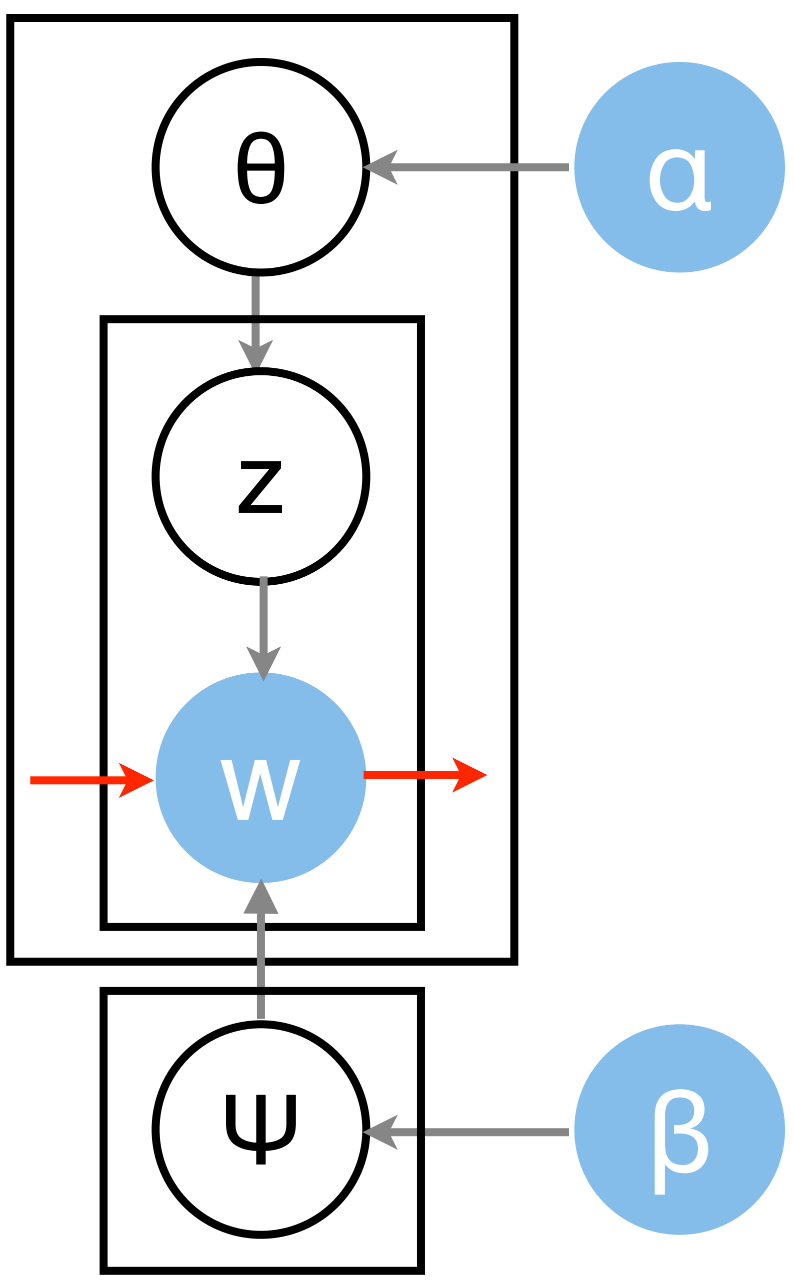
Topic Models



Topic Models

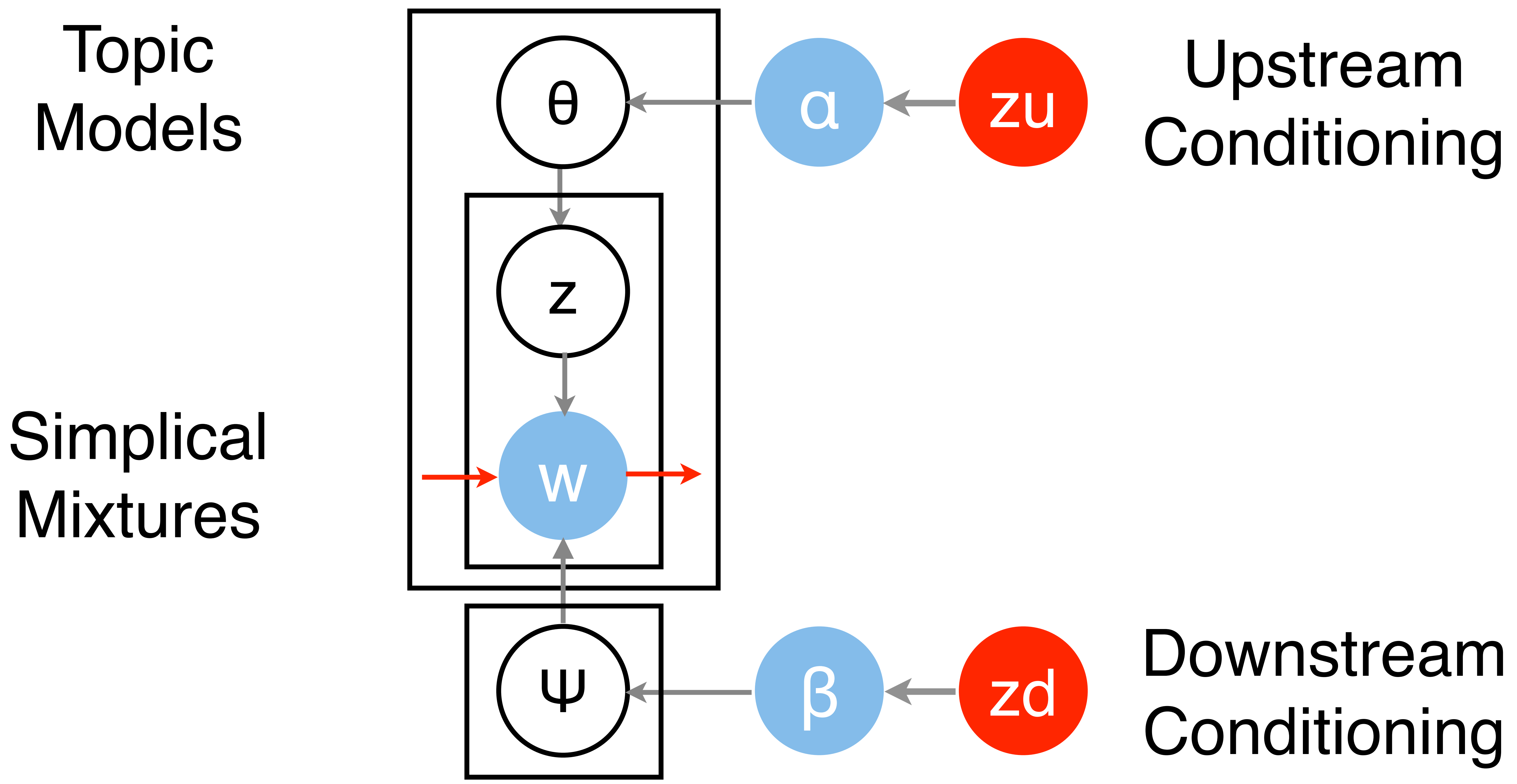
Topic Models

Simplical Mixtures



Carnegie Mellon University

Topic Models



7.5 Undirected Graphical Models

7 Graphical Models

Alexander Smola

Introduction to Machine Learning 10-701

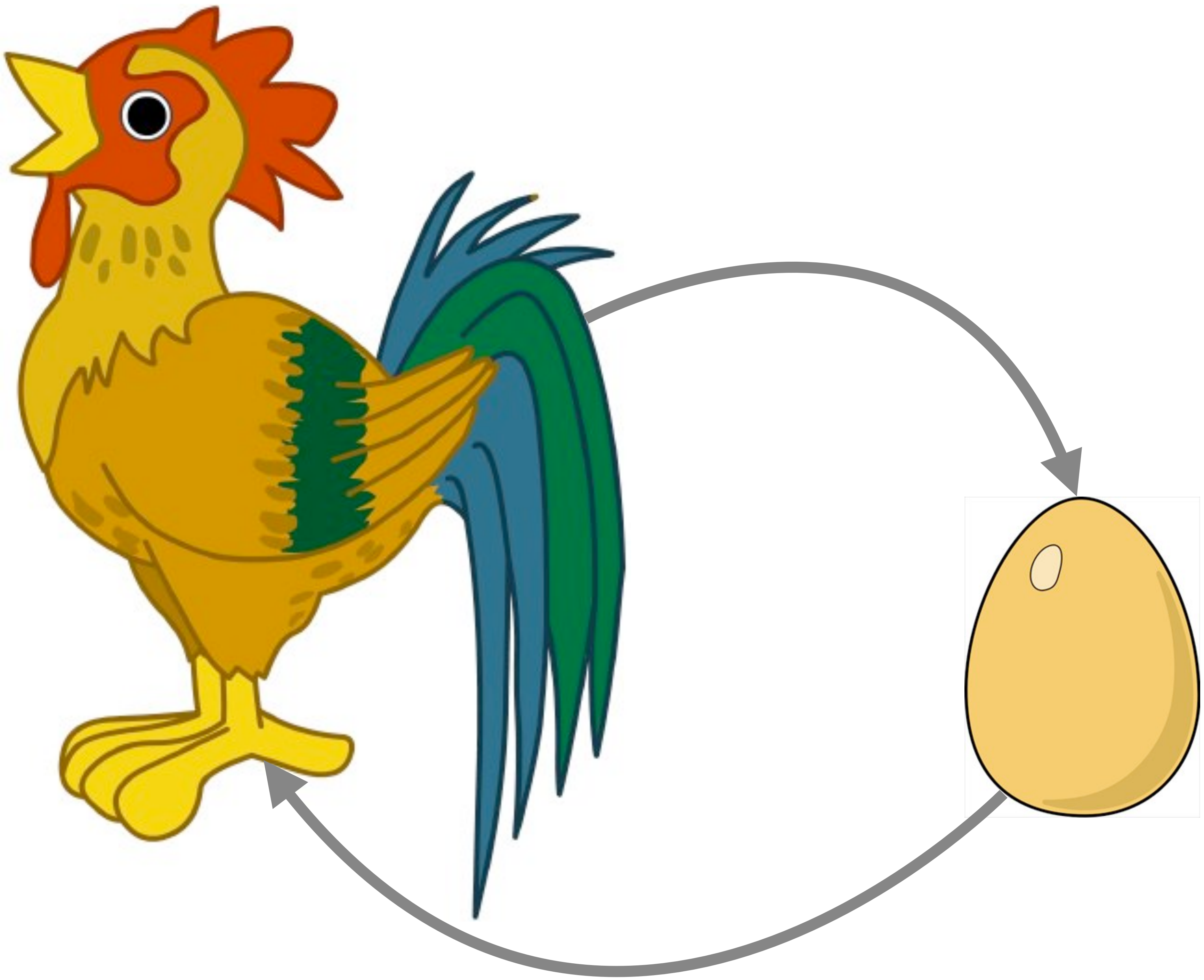
http://alex.smola.org/teaching/10-701-15

MACHINE LEARNING DEPARTMENT

Carnegie Mellon University

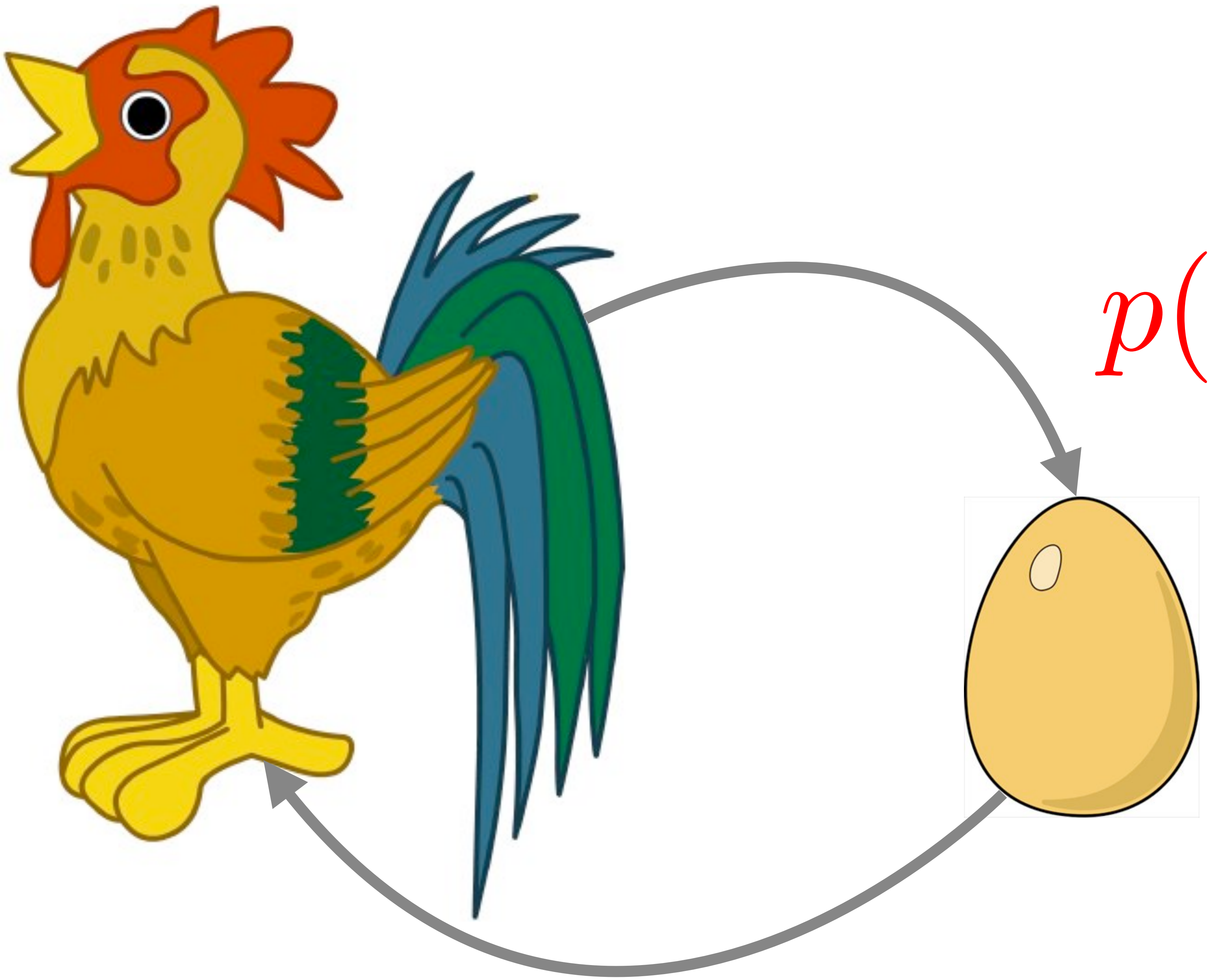
Blunting the arrows

Chicken and Egg



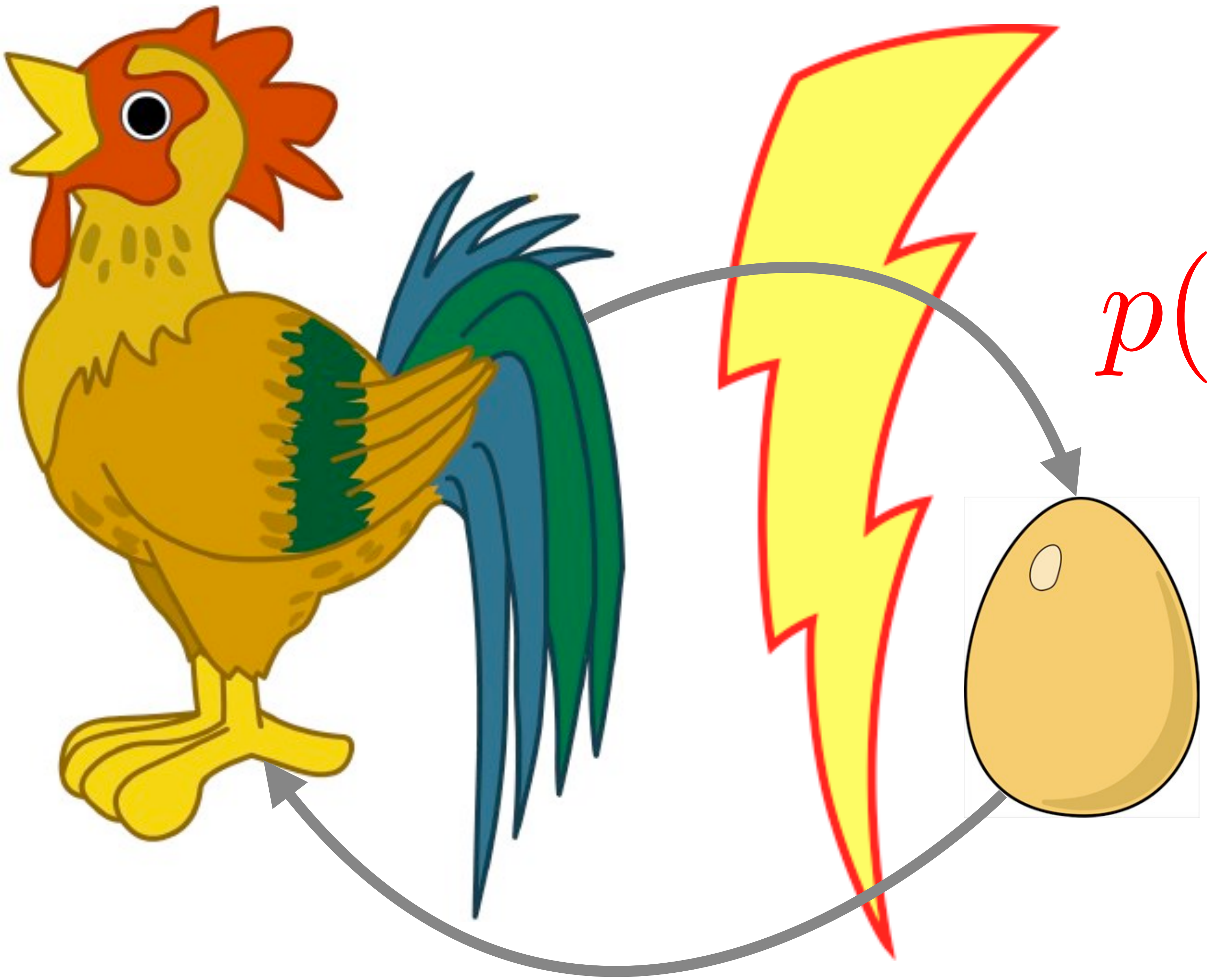
Carnegie Mellon University

Chicken and Egg



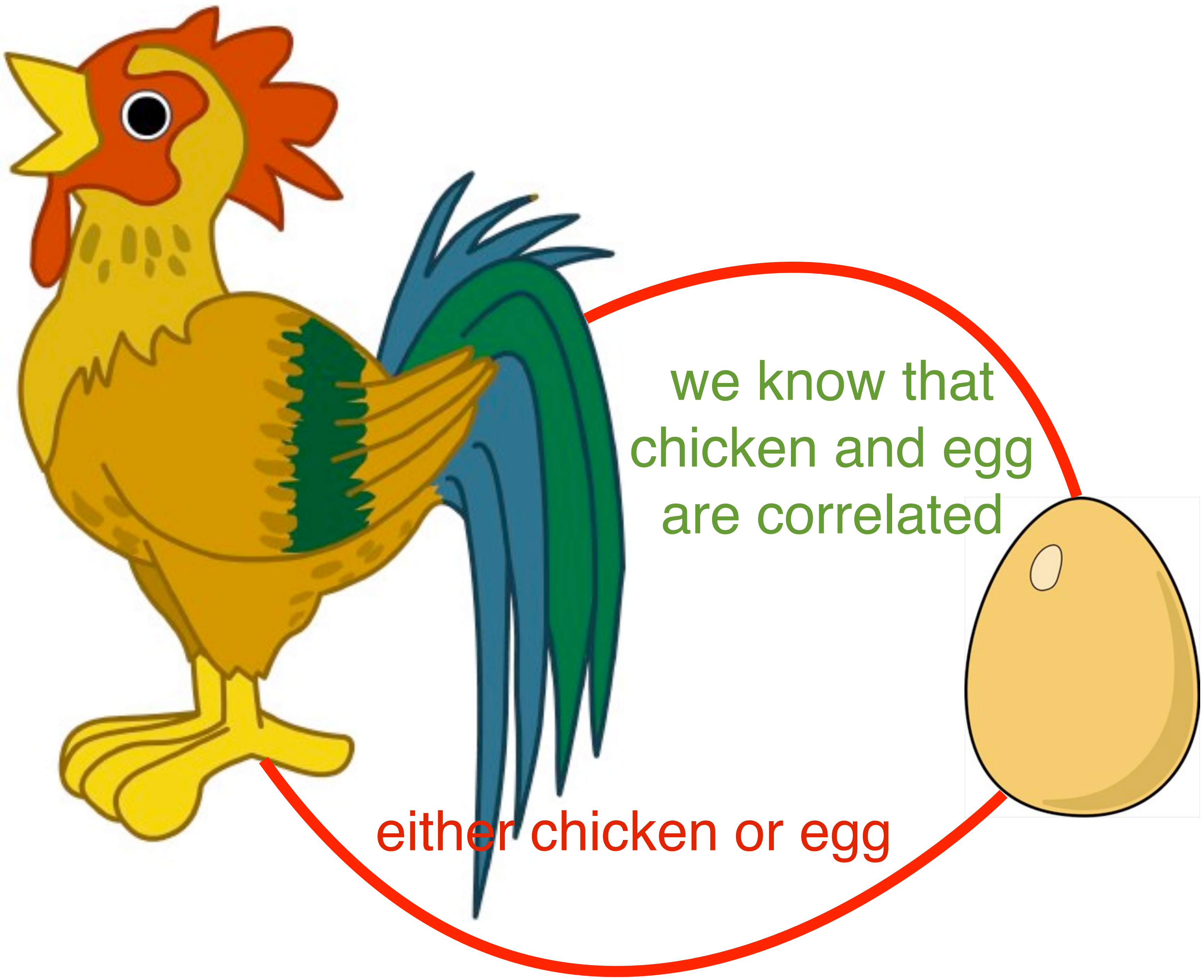
$$p(c|e)p(e|c)$$

Chicken and Egg

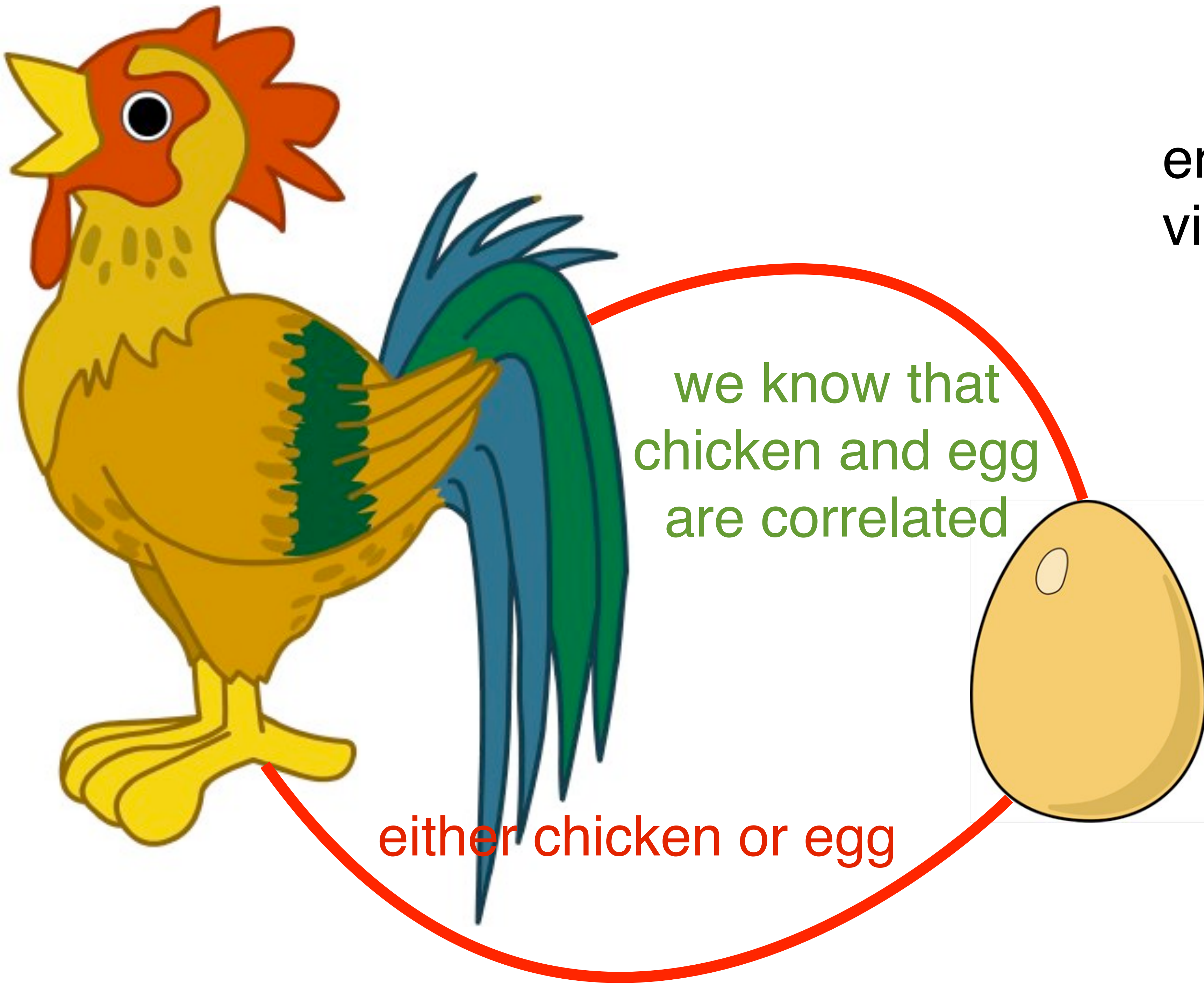


$$p(c|e)p(e|c)$$

Chicken and Egg



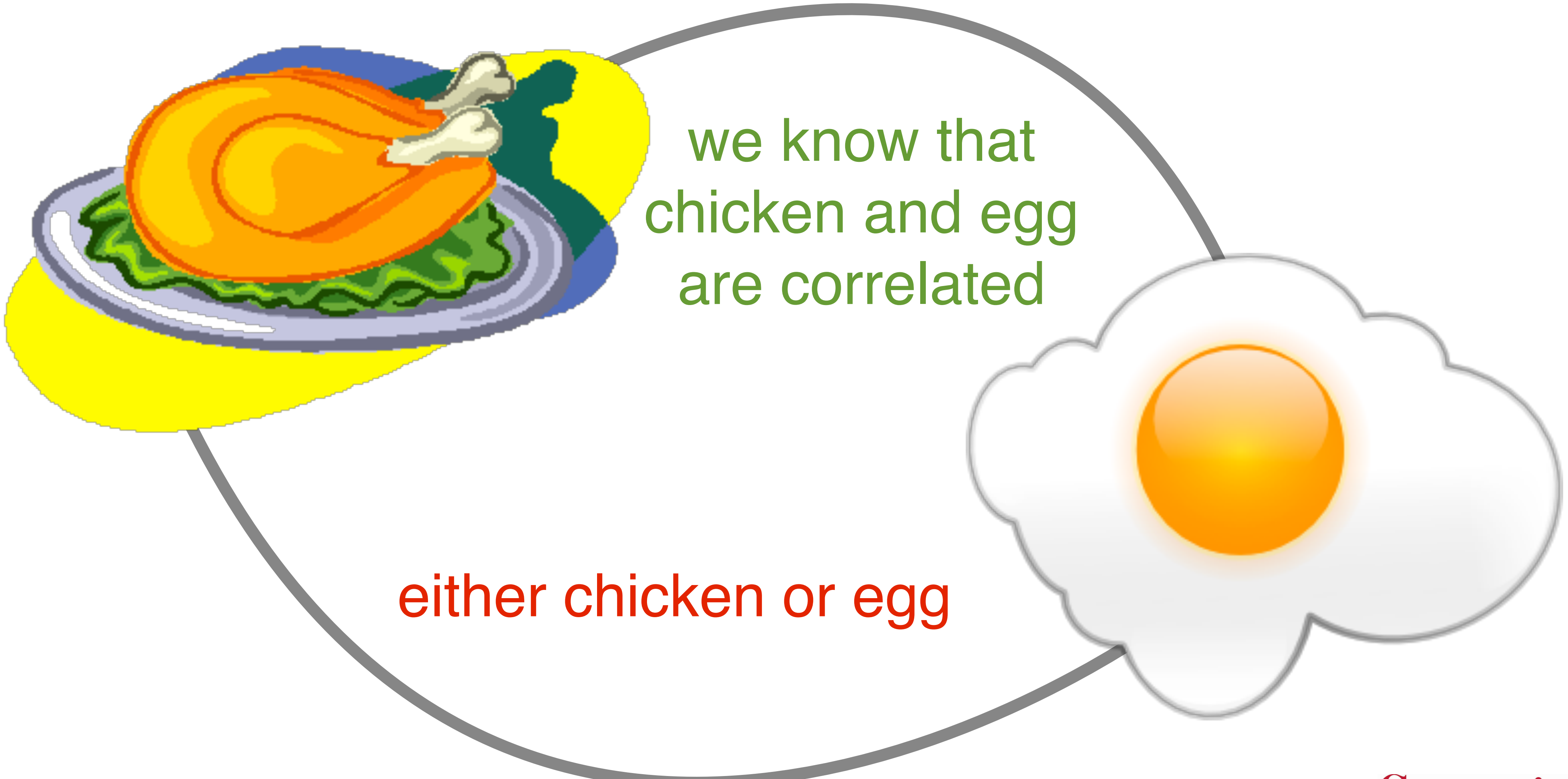
Chicken and Egg



encode the correlation
via the clique potential
between c and e

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

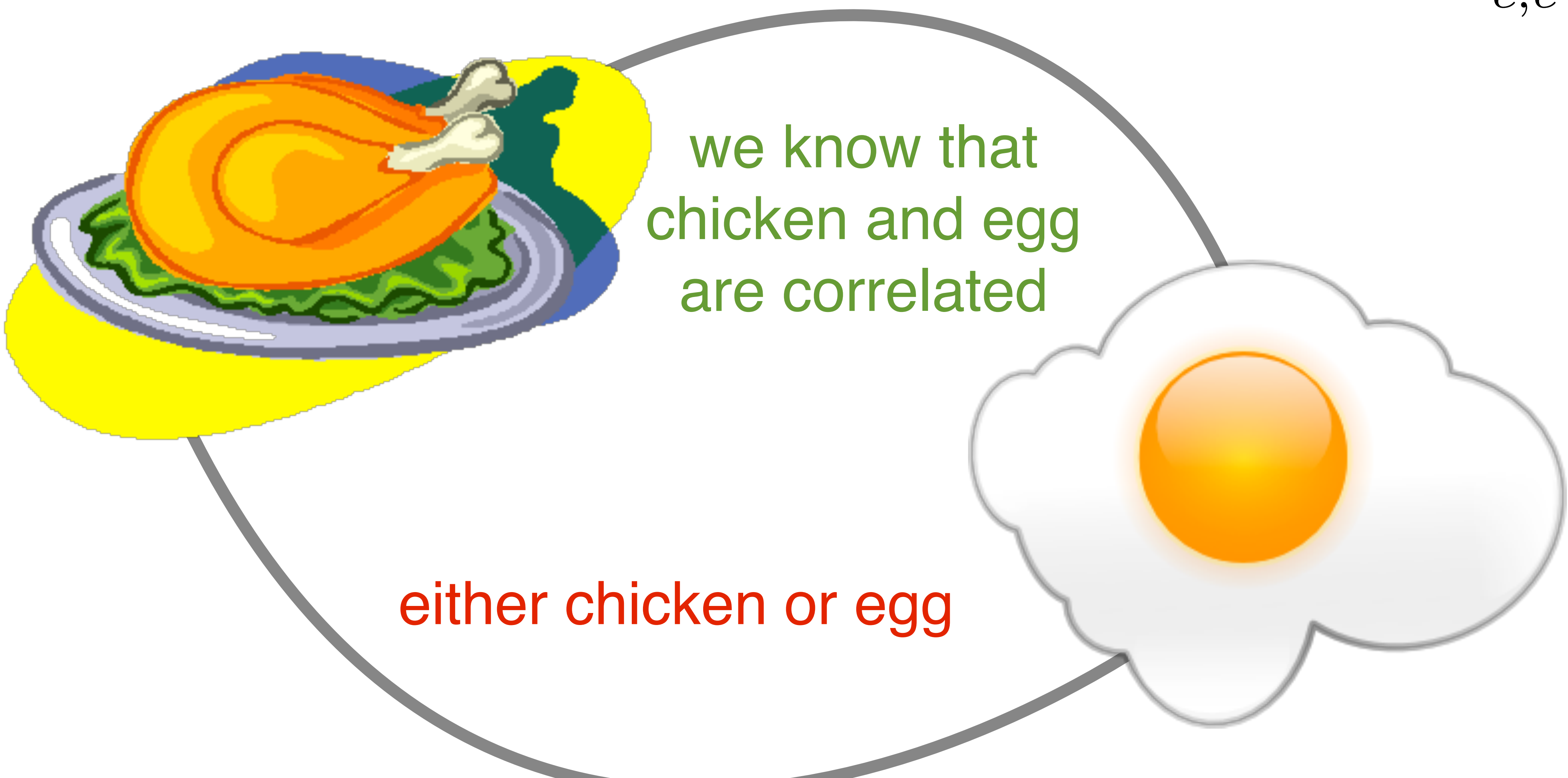
Chicken and Egg



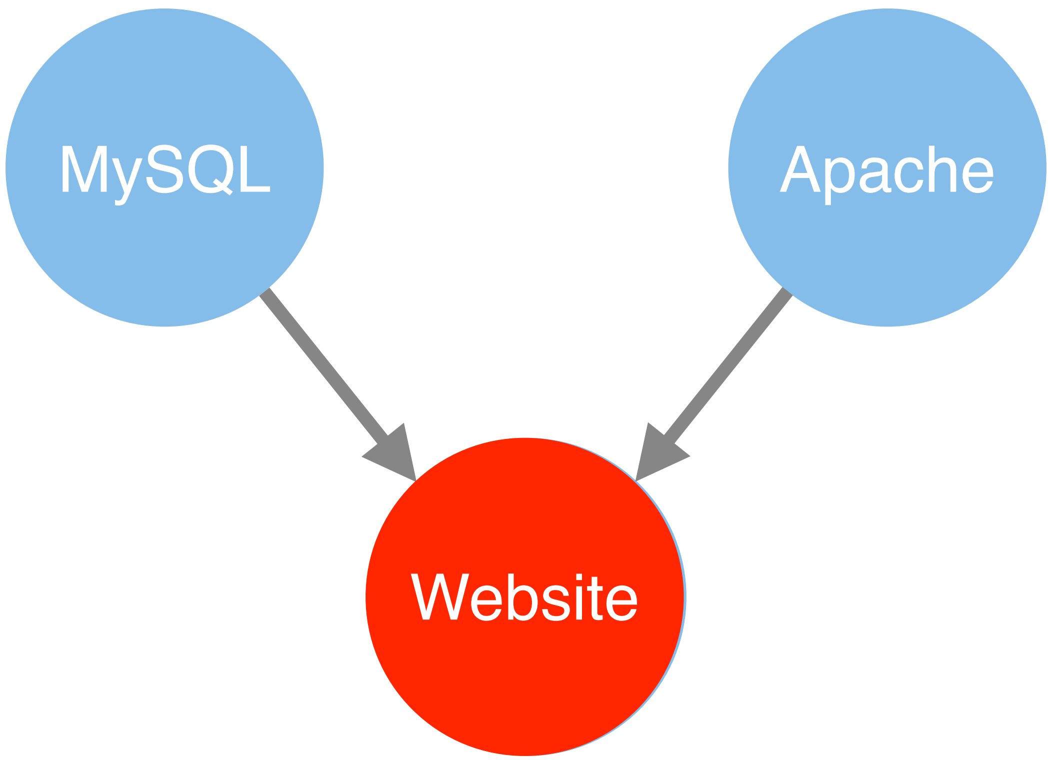
Chicken and Egg

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

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



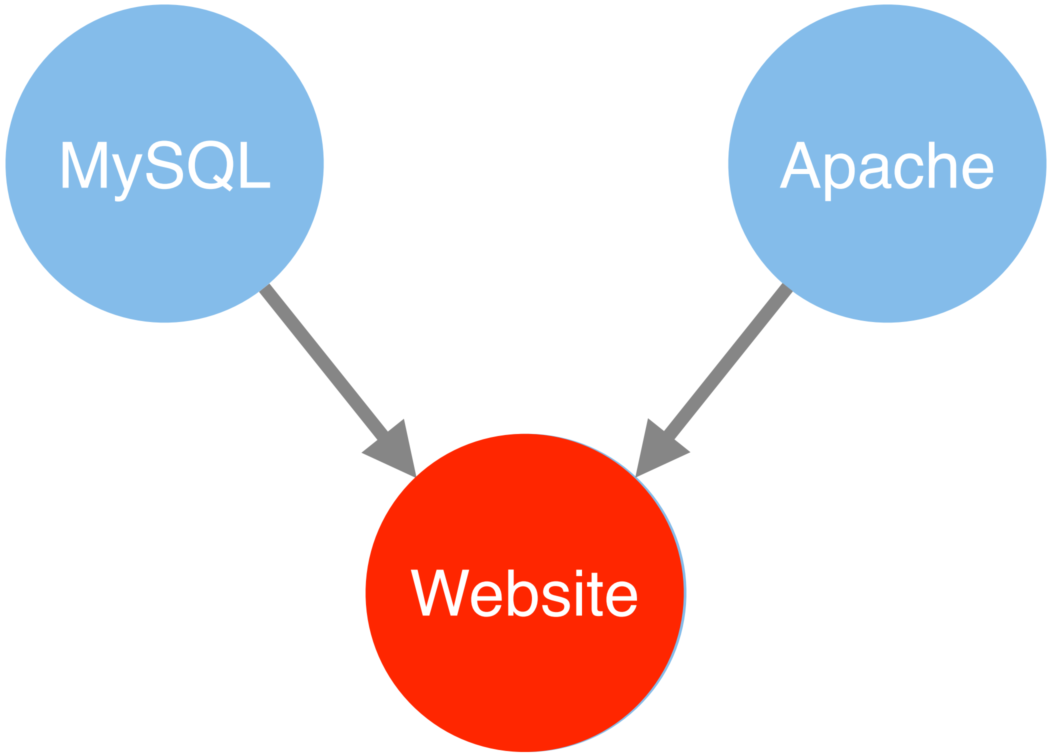
... some web service

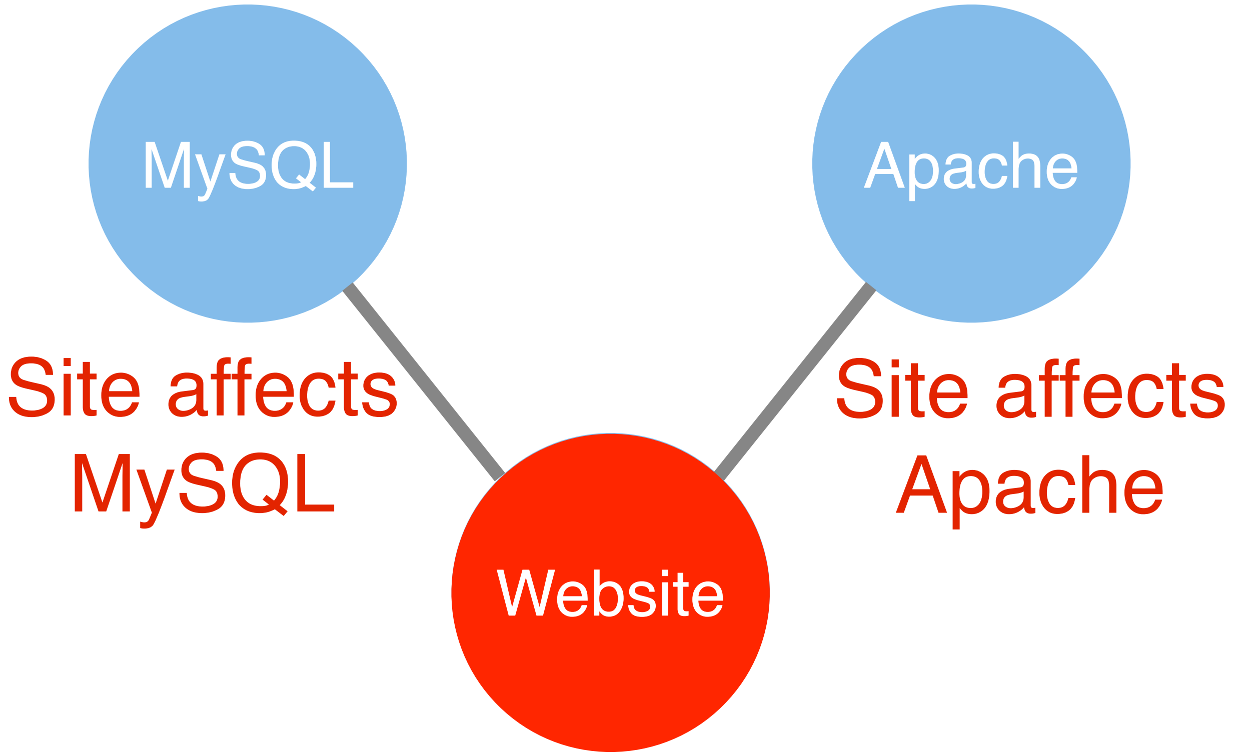


$$p(w|m,a)p(m)p(a)$$

$$m \not\perp\!\!\!\perp a|w$$

... some web service

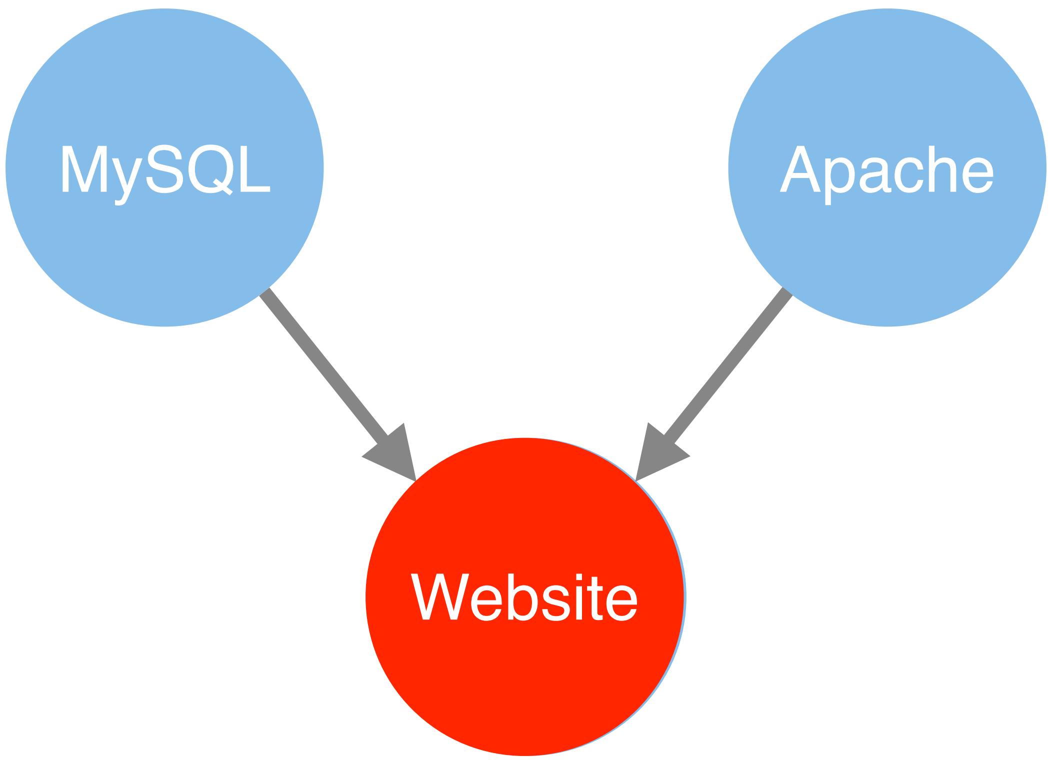




$$p(w|m,a)p(m)p(a)$$
$$m \not\perp\!\!\!\perp a|w$$

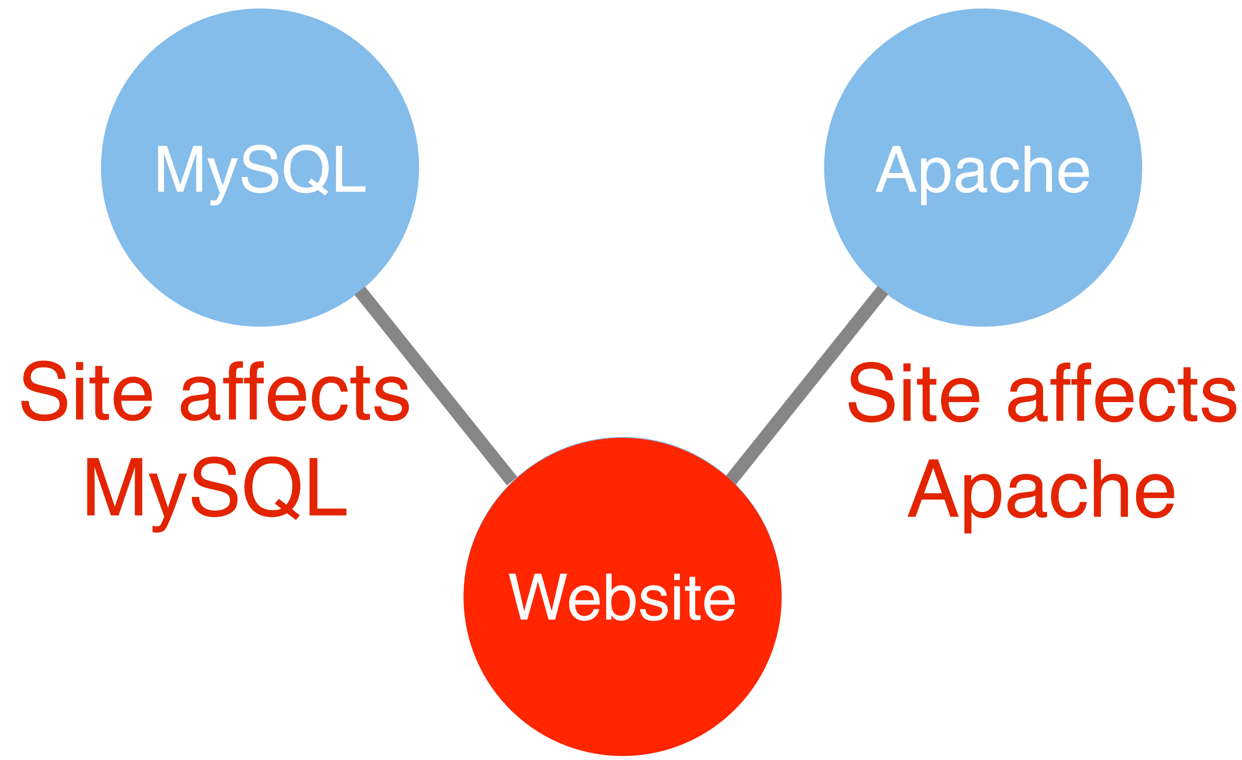
$$p(m,w,a) \propto \phi(m,w)\phi(w,a)$$
$$m \perp\!\!\!\perp a|w$$

... some web service



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

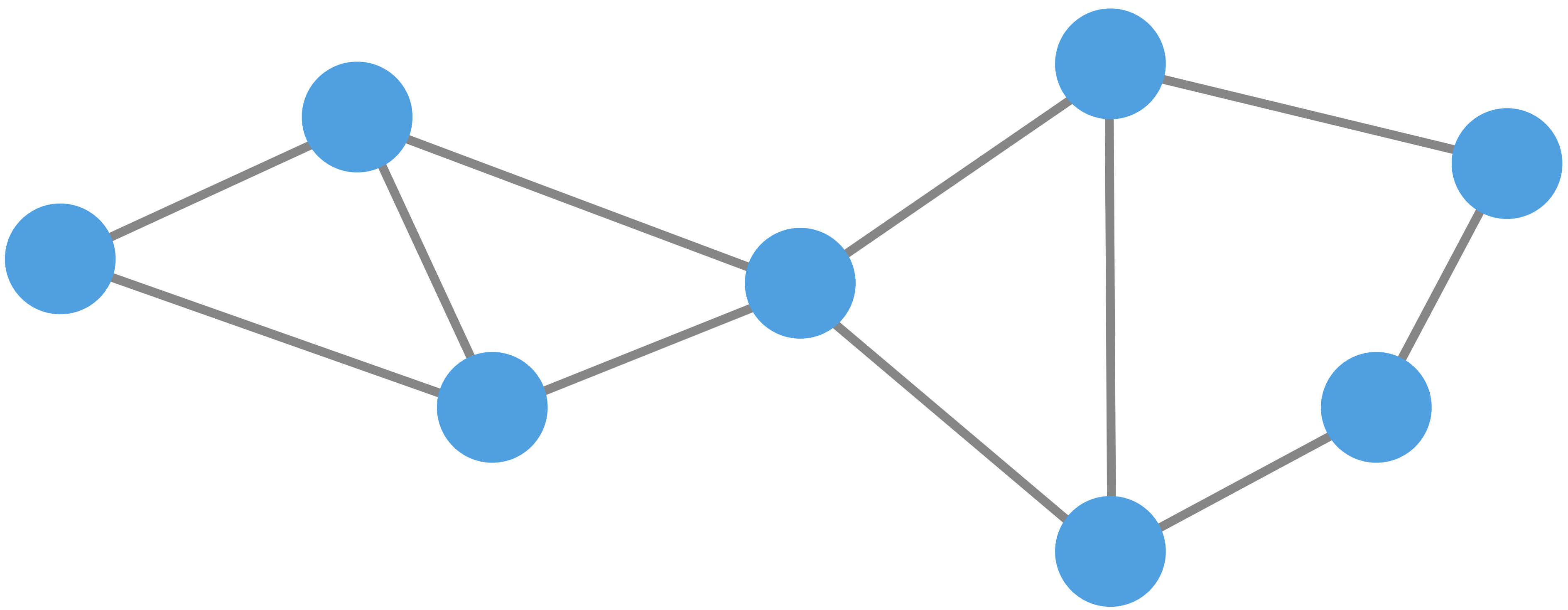
easier
"debugging"



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

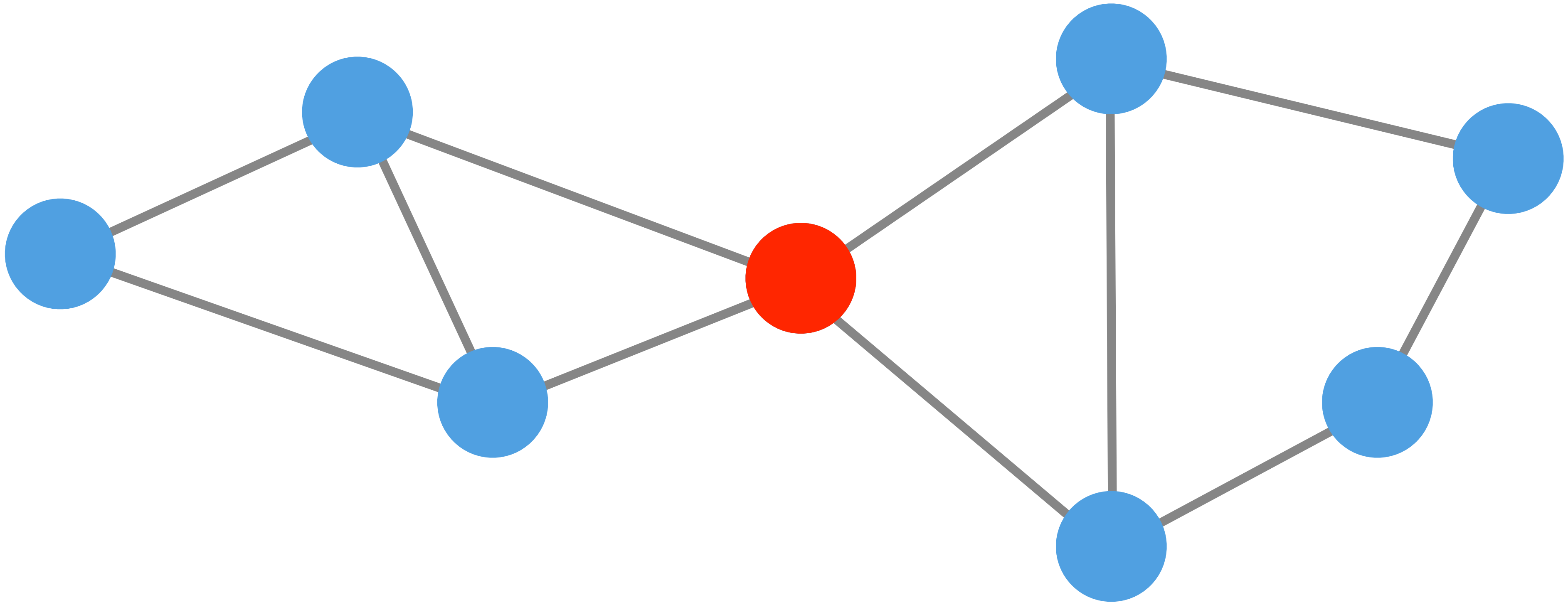
easier
"modeling"

Undirected Graphical Models



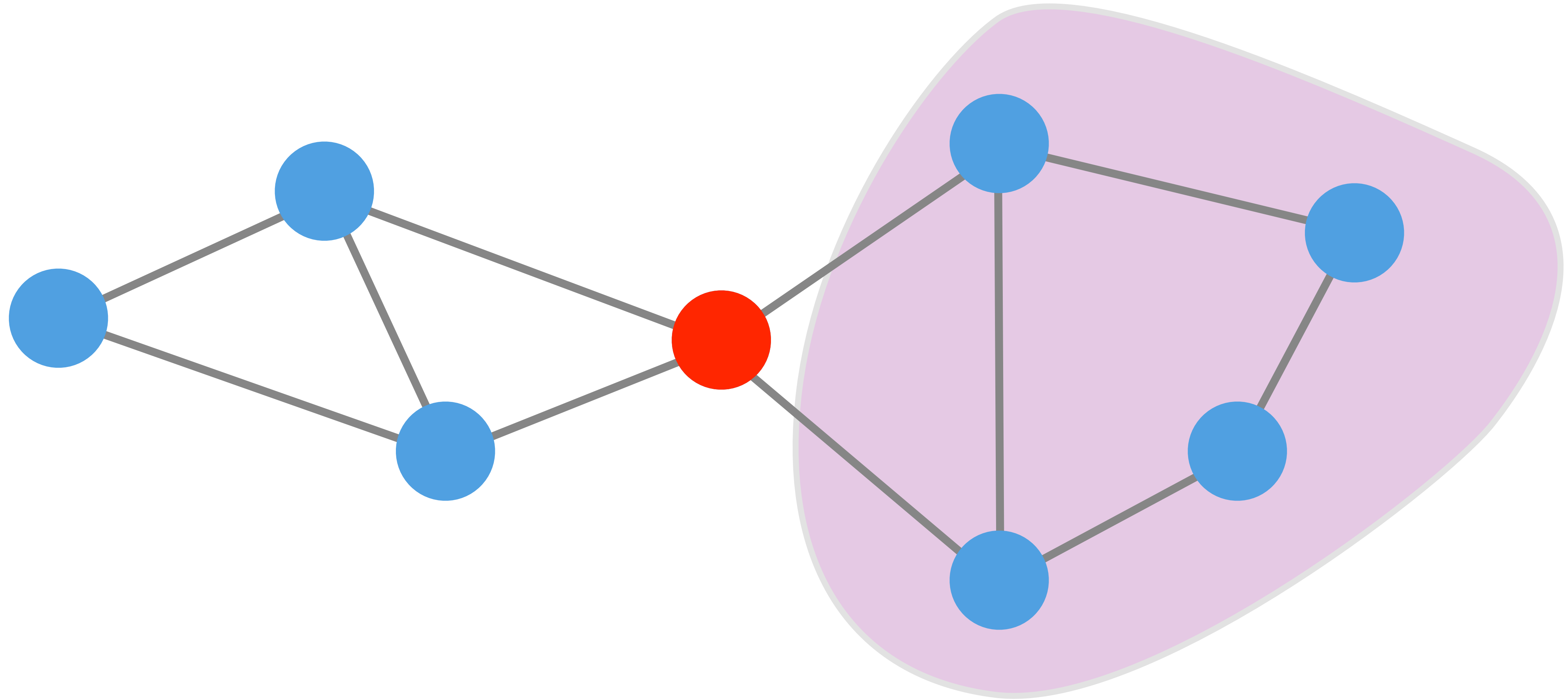
Key Concept
Observing nodes makes remainder
conditionally independent

Undirected Graphical Models



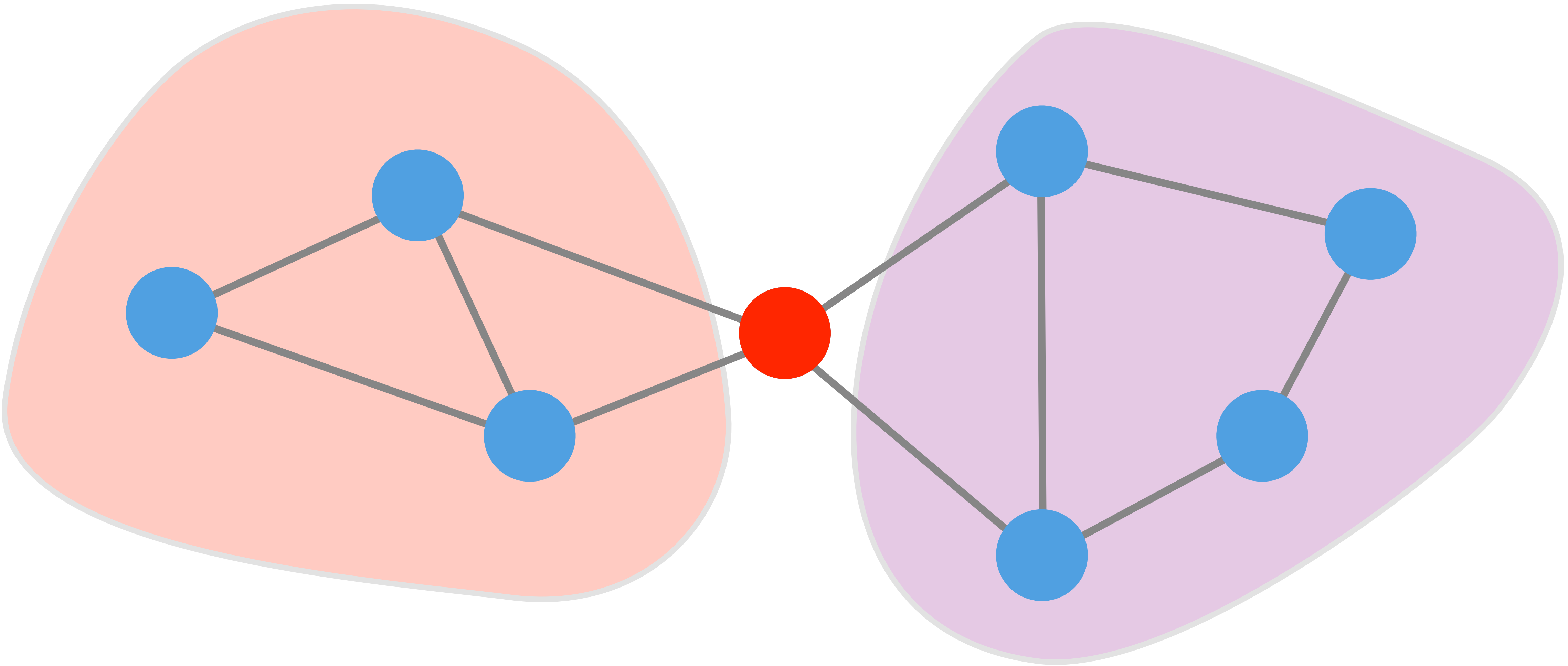
Key Concept
Observing nodes makes remainder
conditionally independent

Undirected Graphical Models



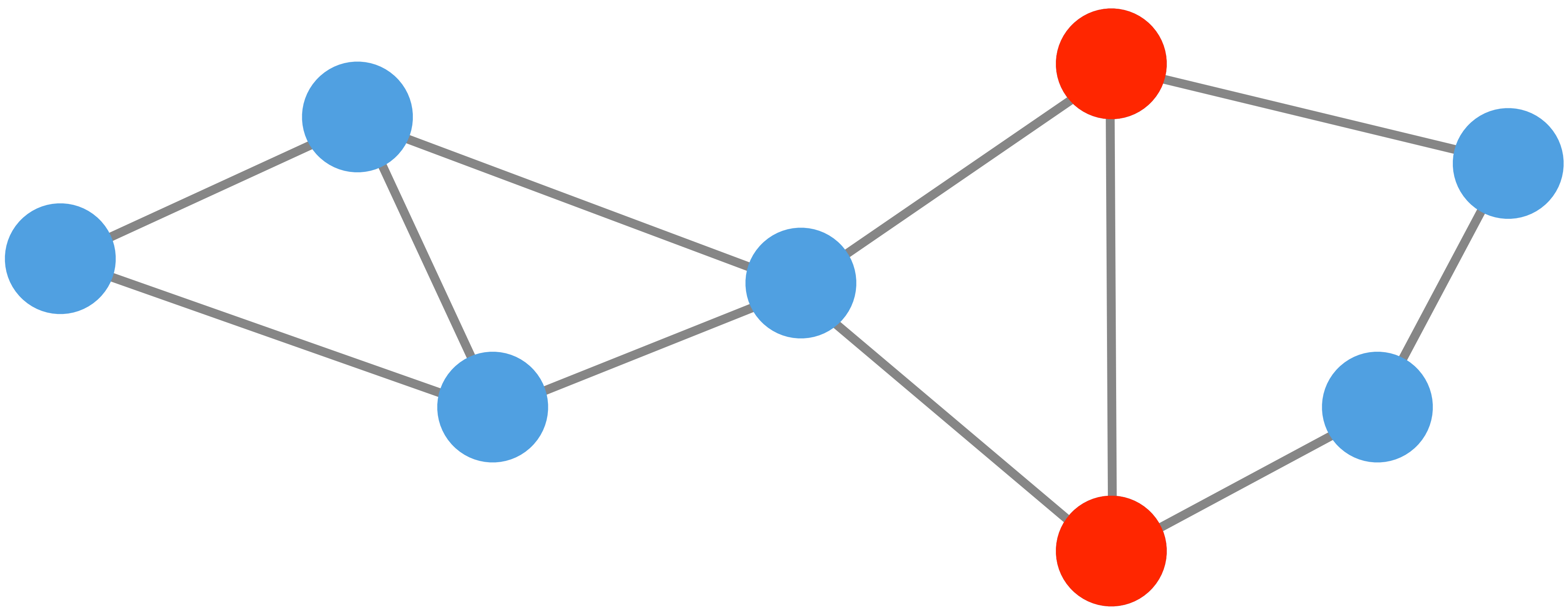
Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



Key Concept
Observing nodes makes remainder conditionally independent

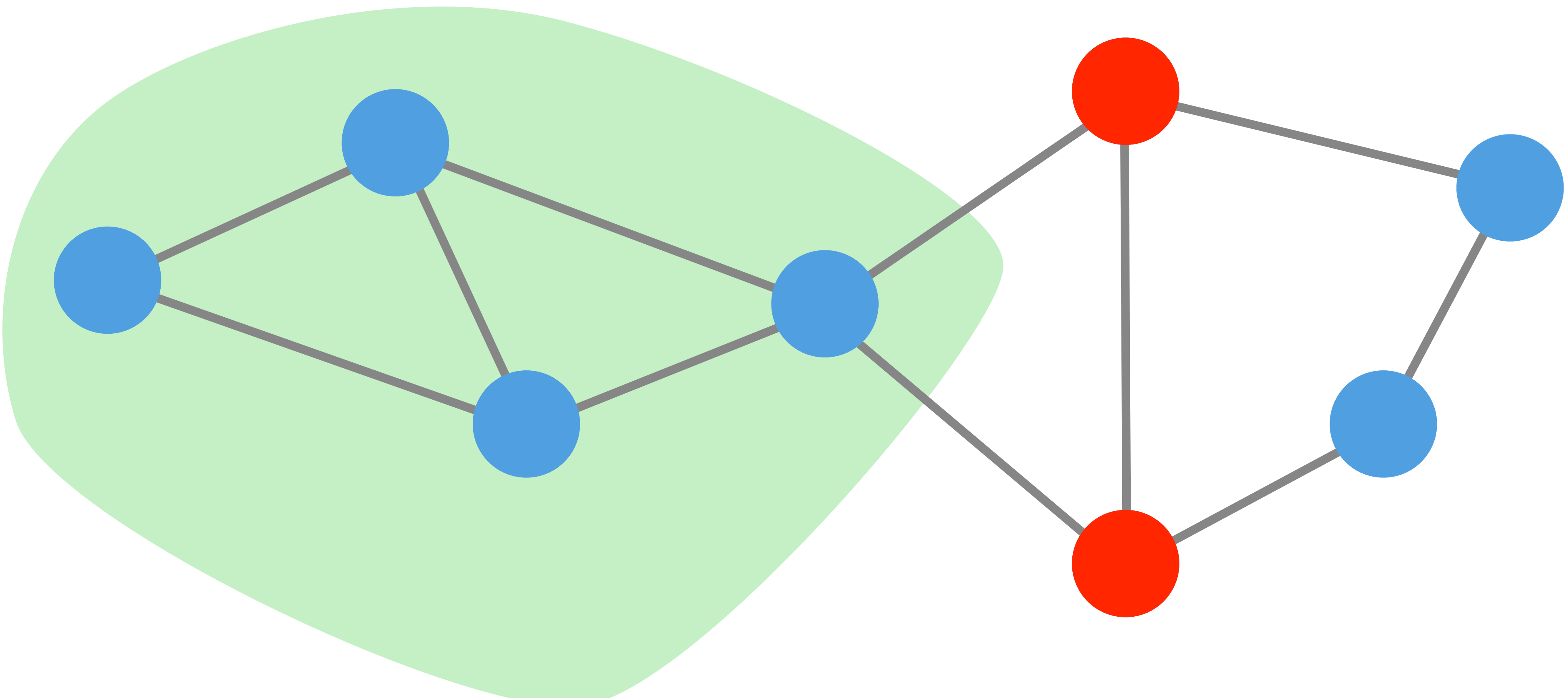
Undirected Graphical Models



Key Concept
Observing nodes makes remainder conditionally independent

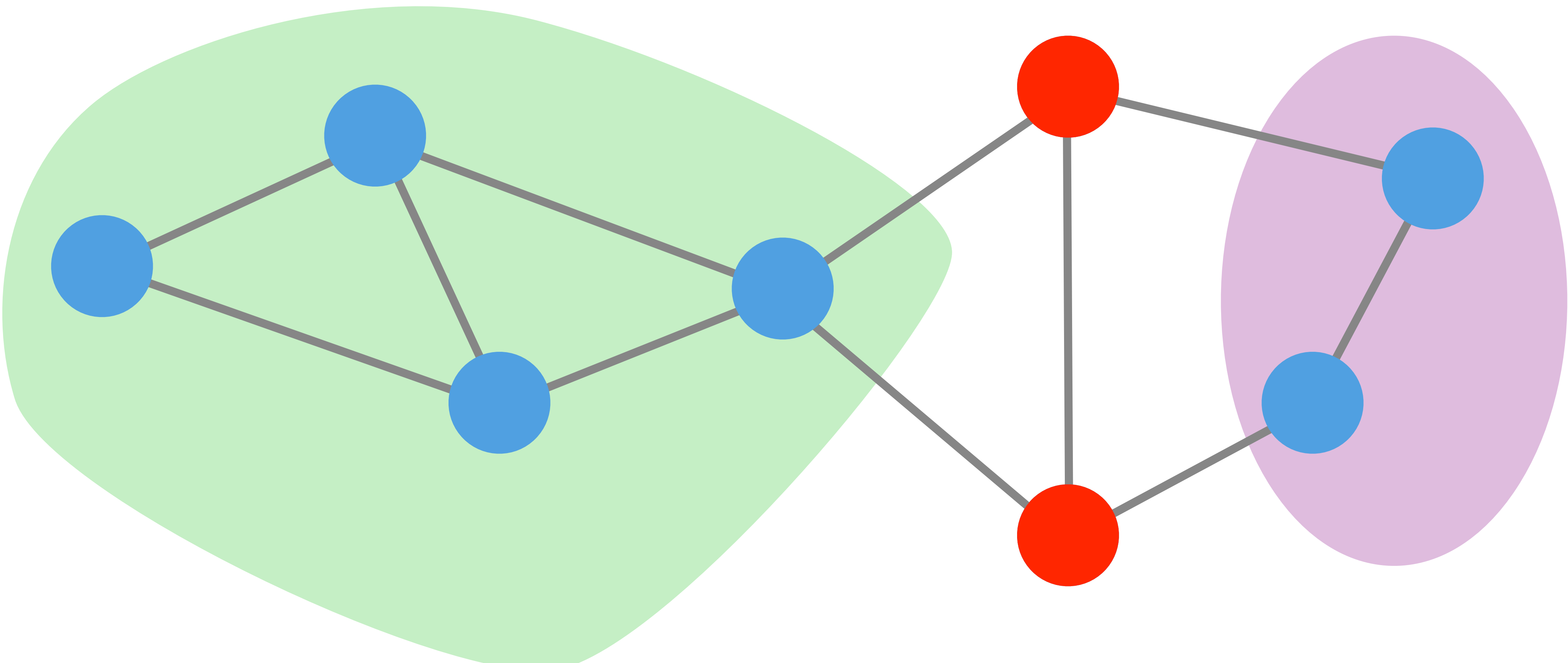
Carnegie Mellon University

Undirected Graphical Models



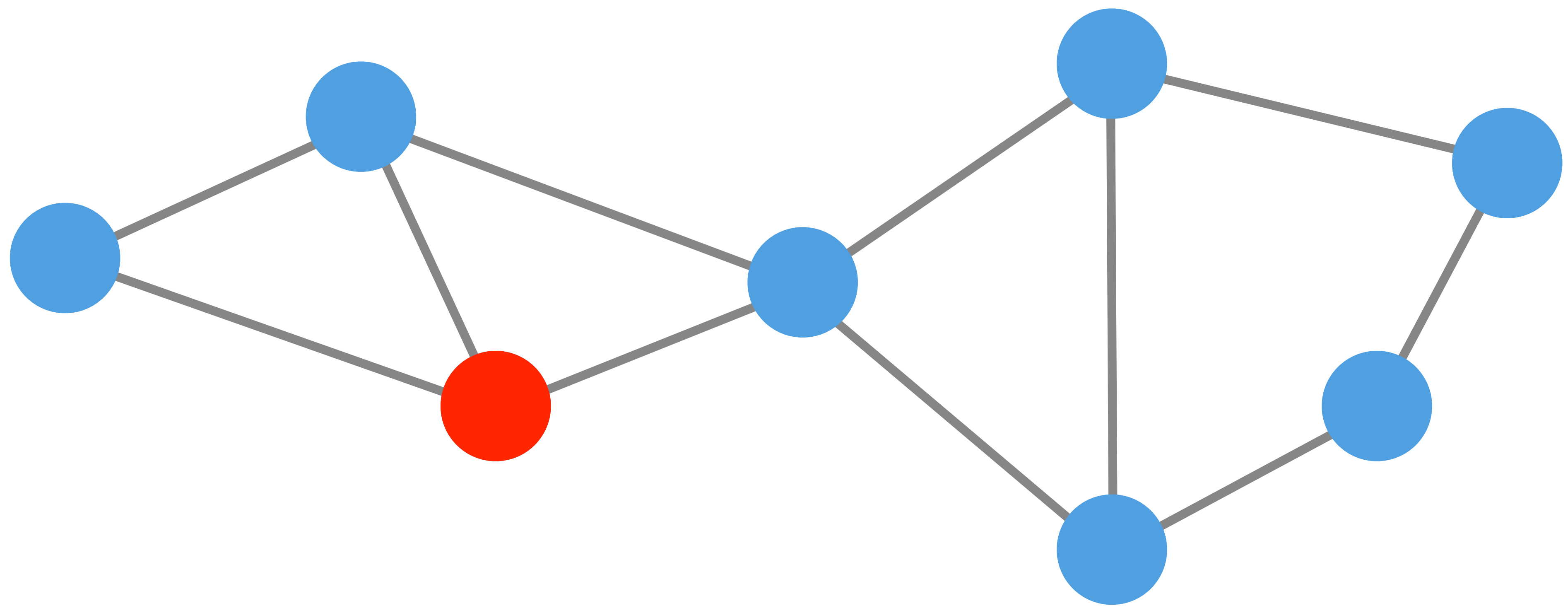
Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



Key Concept
Observing nodes makes remainder conditionally independent

Undirected Graphical Models



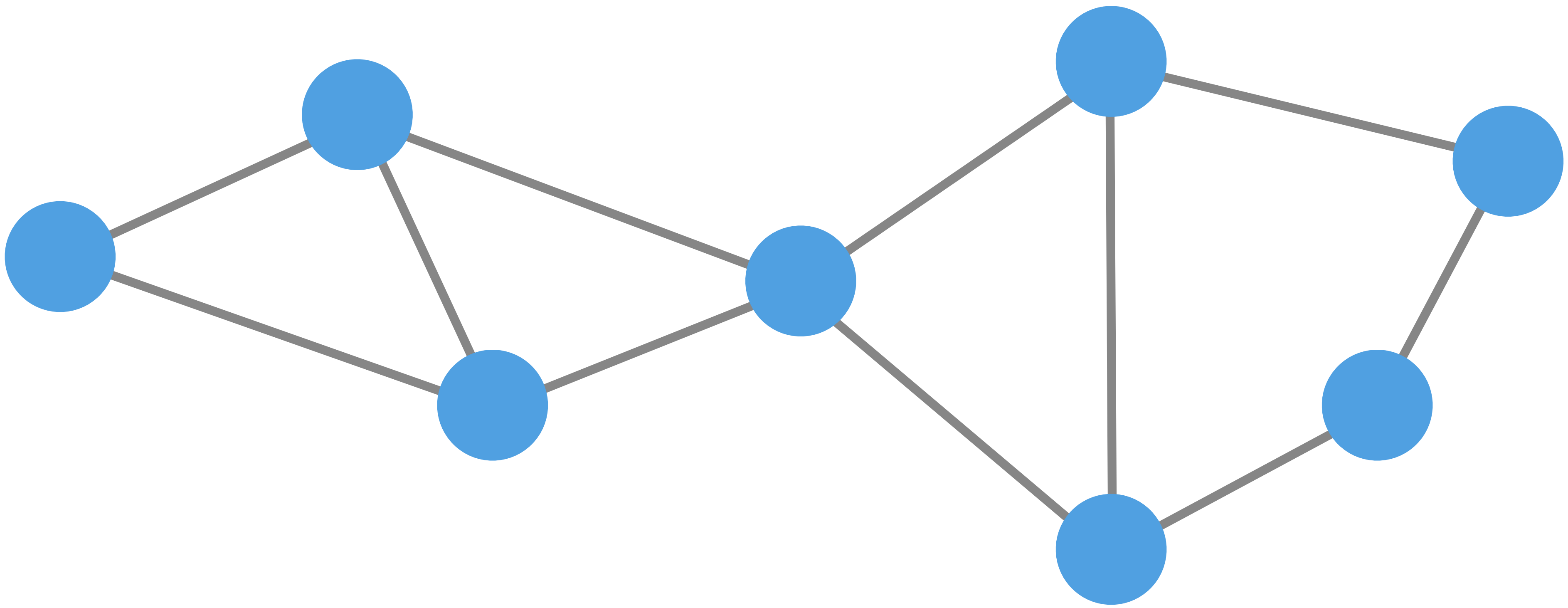
Key Concept
Observing nodes makes remainder
conditionally independent

Cliques

Cliques

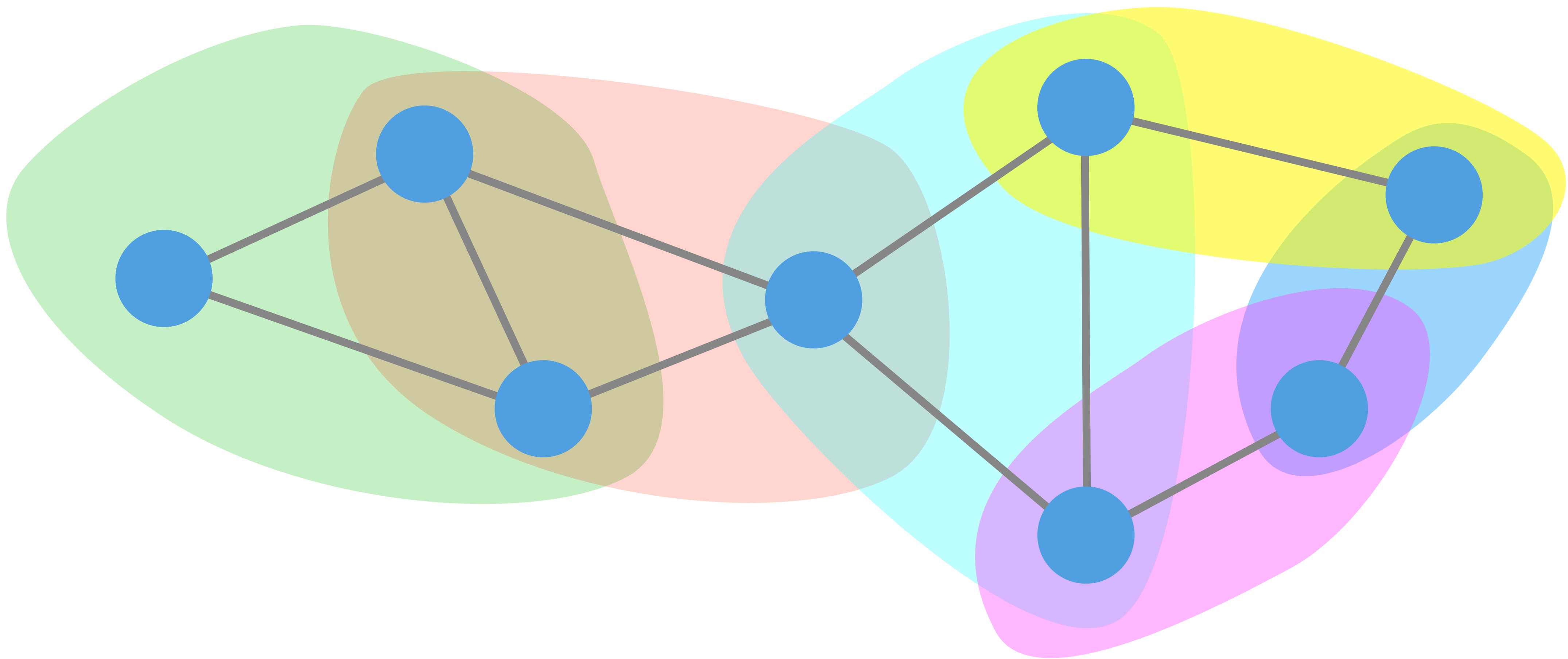
maximal fully connected subgraph

Cliques



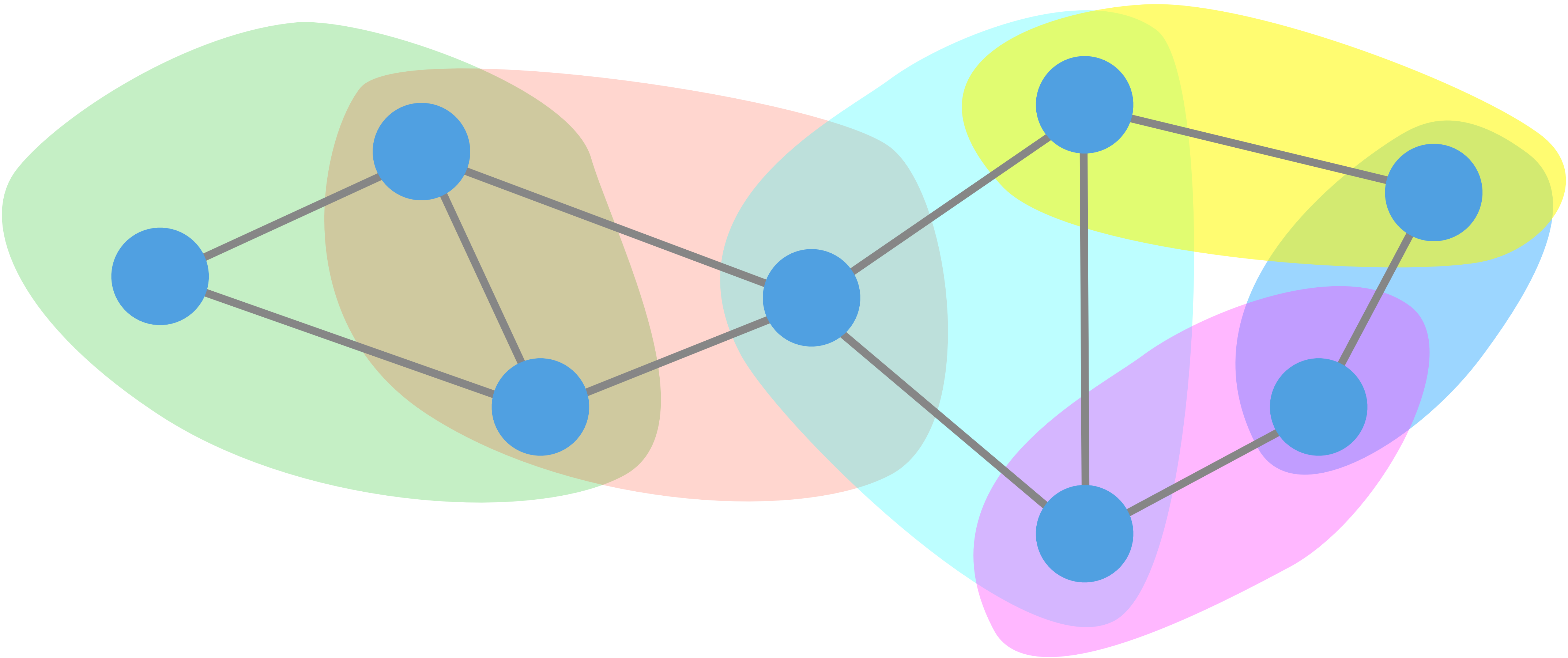
maximal fully connected subgraph

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

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

Unconditional model

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

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

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

Conditional model

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

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

Estimation

- Conditional log-likelihood

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

- Log-posterior (Gaussian Prior)

$$\log p(\theta|X,Y) = \sum_i \log(y_i|x_i;\theta) + \log p(\theta) + \text{const.}$$

$$= \left\langle \sum_i \phi(x_i,y_i), \theta \right\rangle - \sum_i g(\theta|x_i) - \frac{1}{2\sigma^2}\|\theta\|^2 + \text{const.}$$

- First order optimality conditions

expensive

prior

maxent model

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

Logistic Regression

- Label space

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

- Log-partition function

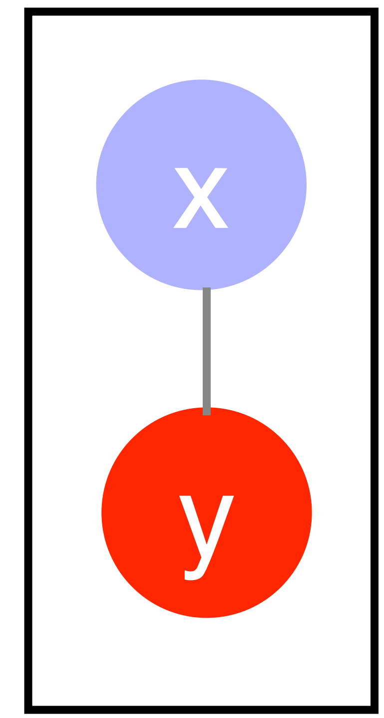
$$g(\theta|x) = \log \left[e^{1 \cdot \langle \phi(x), \theta \rangle} + e^{-1 \cdot \langle \phi(x), \theta \rangle} \right] = \log 2 \cosh \langle \phi(x), \theta \rangle$$

- 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

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



Carnegie Mellon University

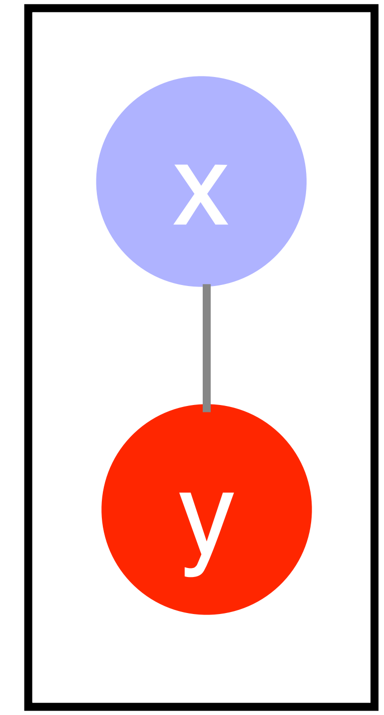
Logistic Regression

- Label space

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

- 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

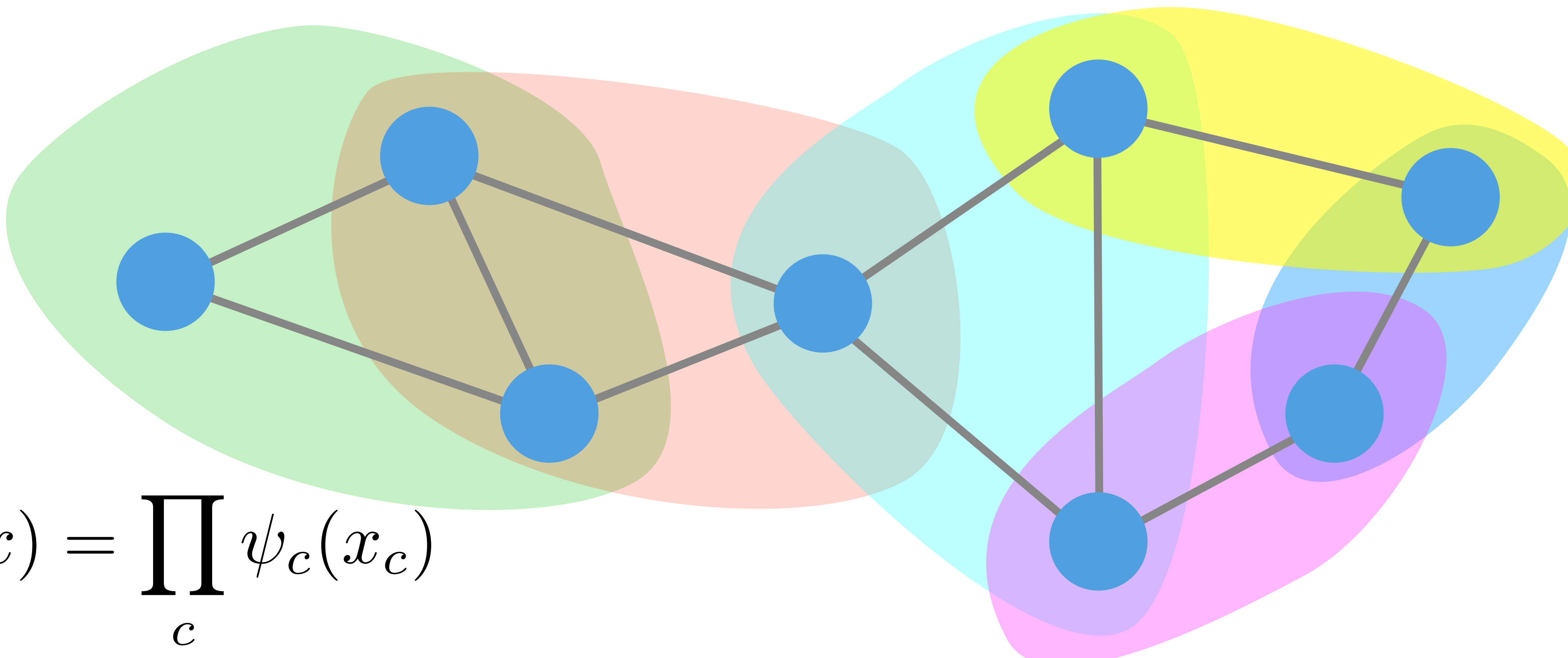
$$\underset{\theta}{\text{minimize}} \frac{1}{2\sigma^2}\|\theta\|^2 + \sum_i \log 2\cosh\langle\phi(x_i),\theta\rangle - y_i\langle\phi(x_i,\theta\rangle$$

- 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



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

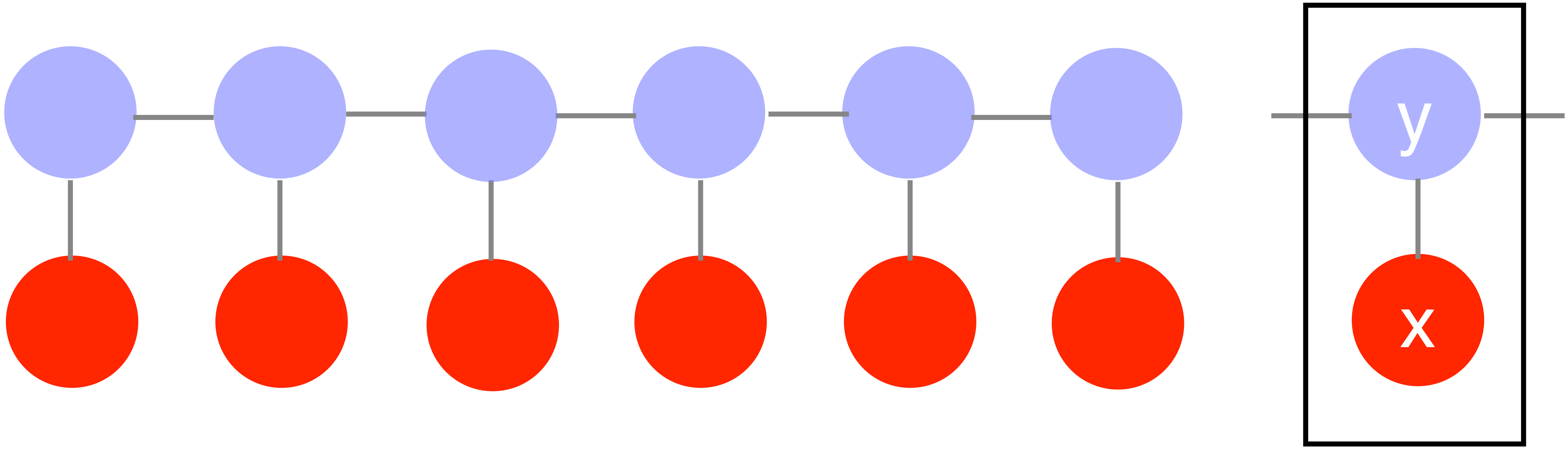
Theorem: Clique decomposition holds in sufficient statistics

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

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



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

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

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

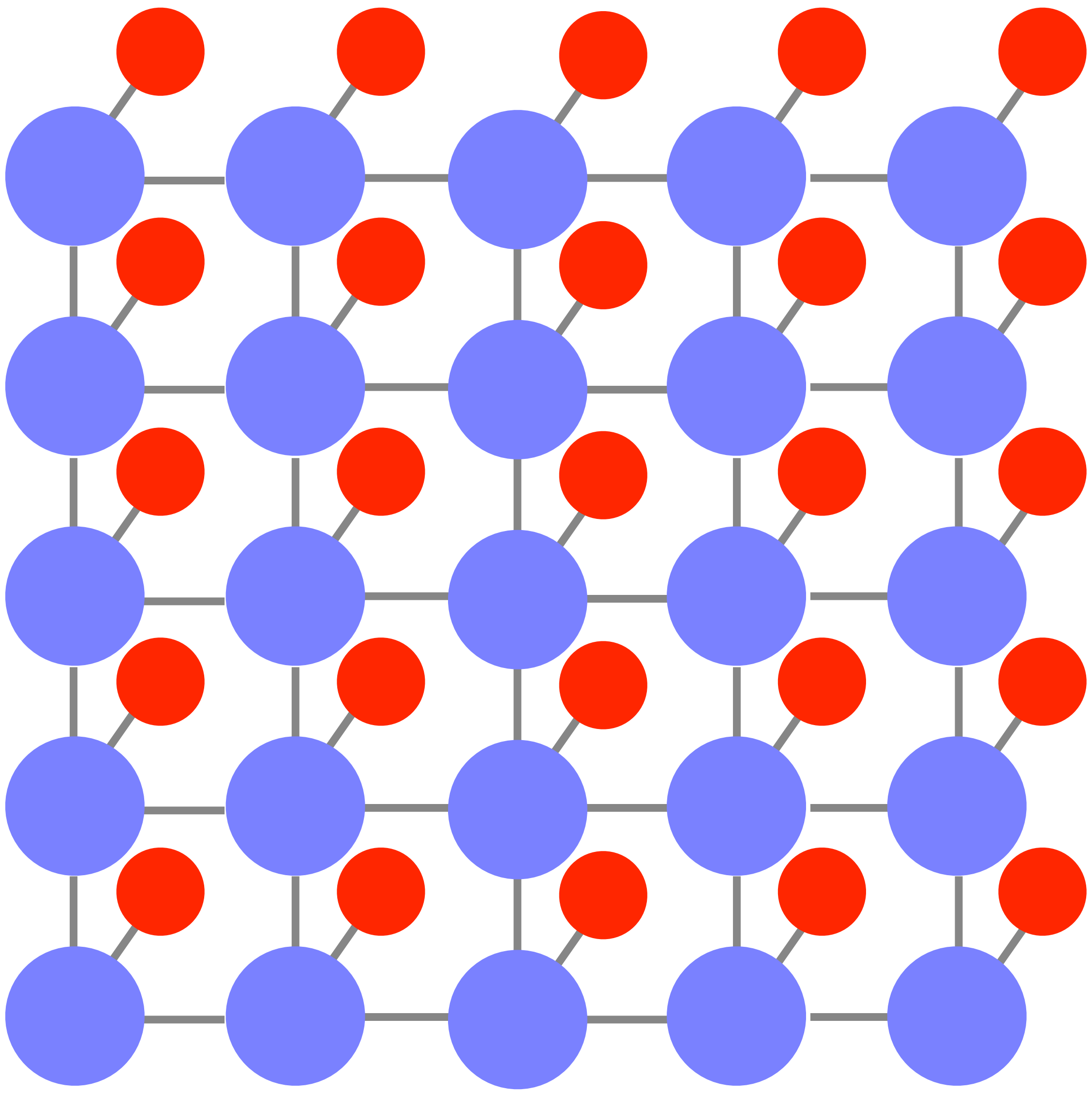
$$f_i(y_i, y_{i+1}) = e^{\langle \phi_x(x_i, y_i), \theta_x \rangle + \langle \phi_y(y_i, y_{i+1}), \theta_y \rangle}$$

dynamic programming

Conditional Random Fields

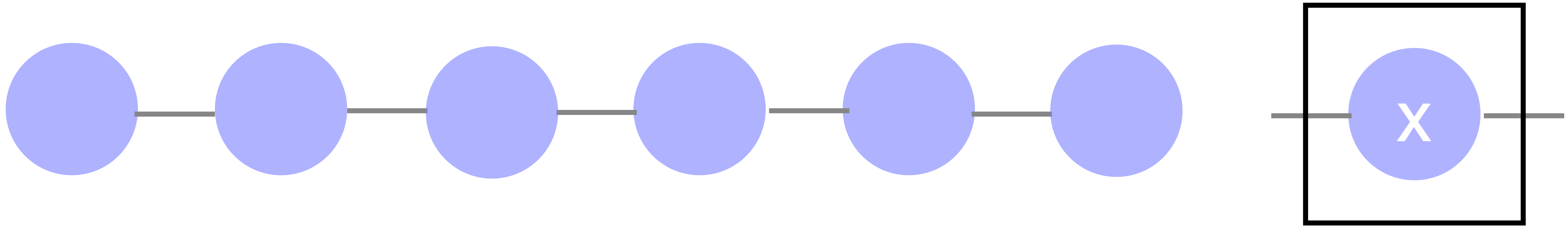
- Compute distribution over marginal and adjacent labels
- Take conditional expectations
- Take update step (batch or online)

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



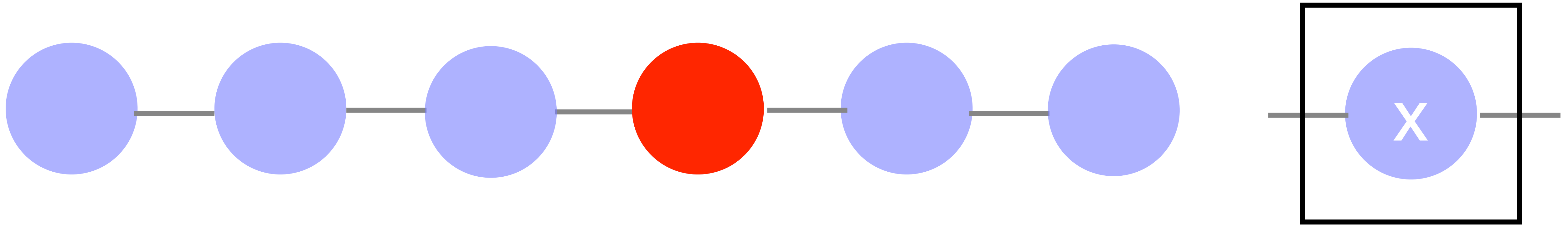
Examples

Chains



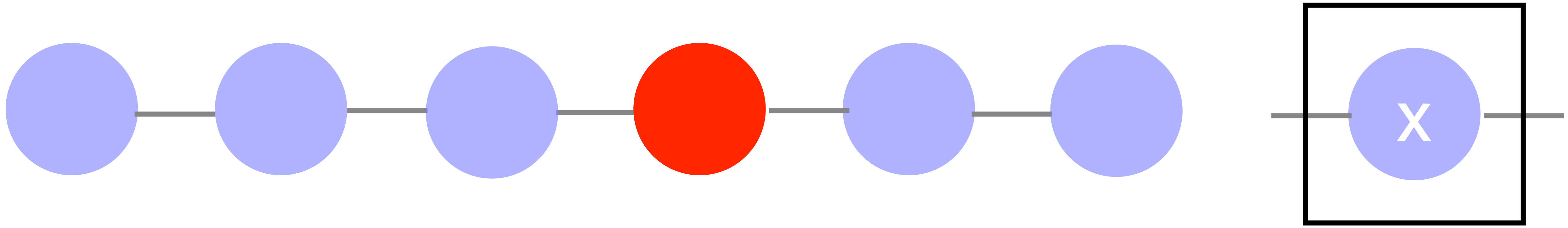
$$p(x) = \prod_i \psi_i(x_i, x_{i+1})$$

Chains

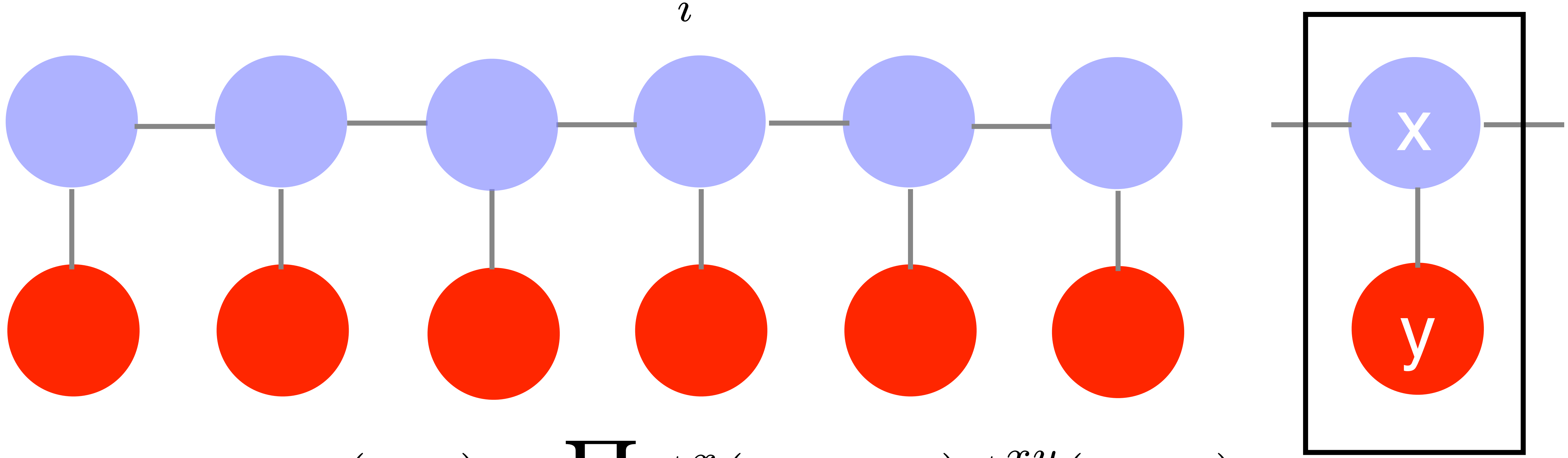


$$p(x) = \prod_i \psi_i(x_i, x_{i+1})$$

Chains

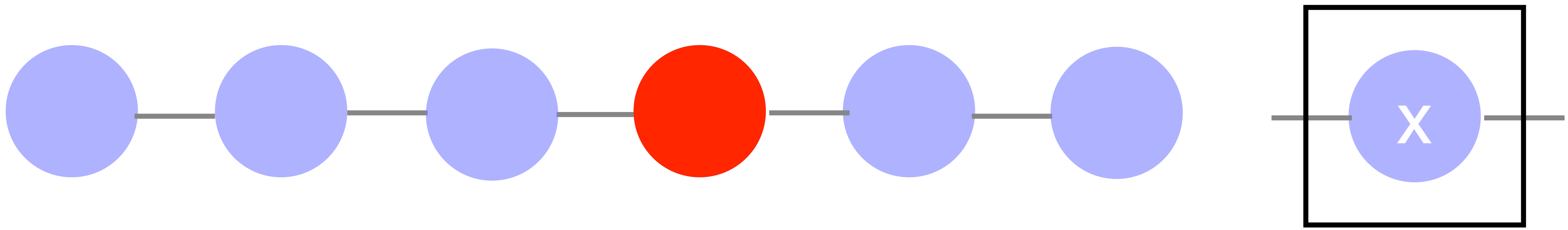


$$p(x) = \prod_i \psi_i(x_i, x_{i+1})$$

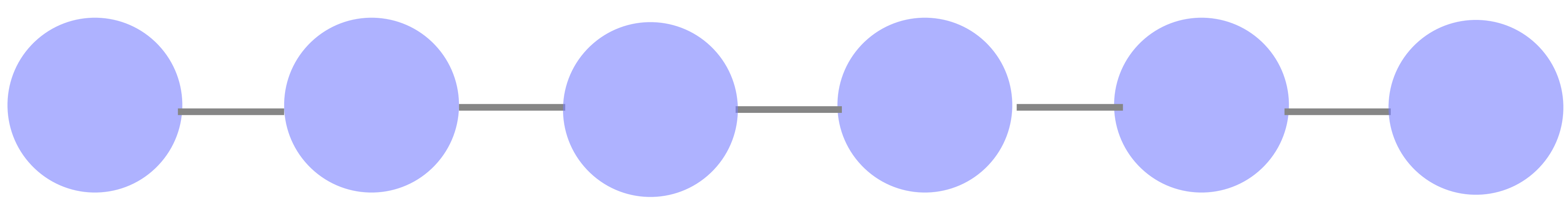


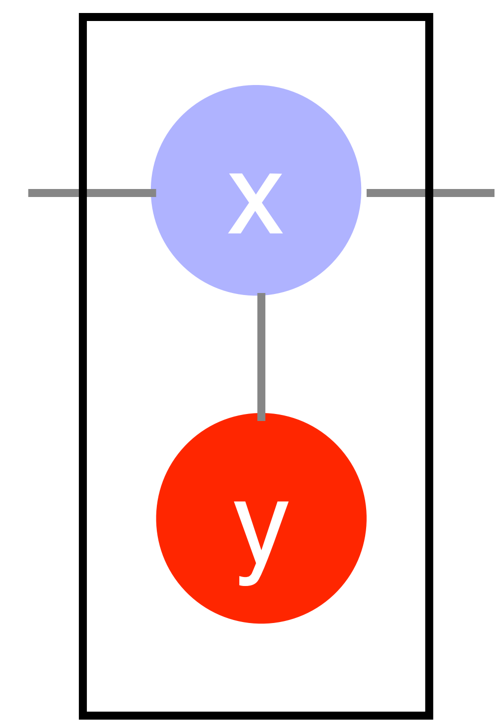
$$p(x, y) = \prod_i \psi_i^x(x_i, x_{i+1})\psi_i^{xy}(x_i, y_i)$$

Chains



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

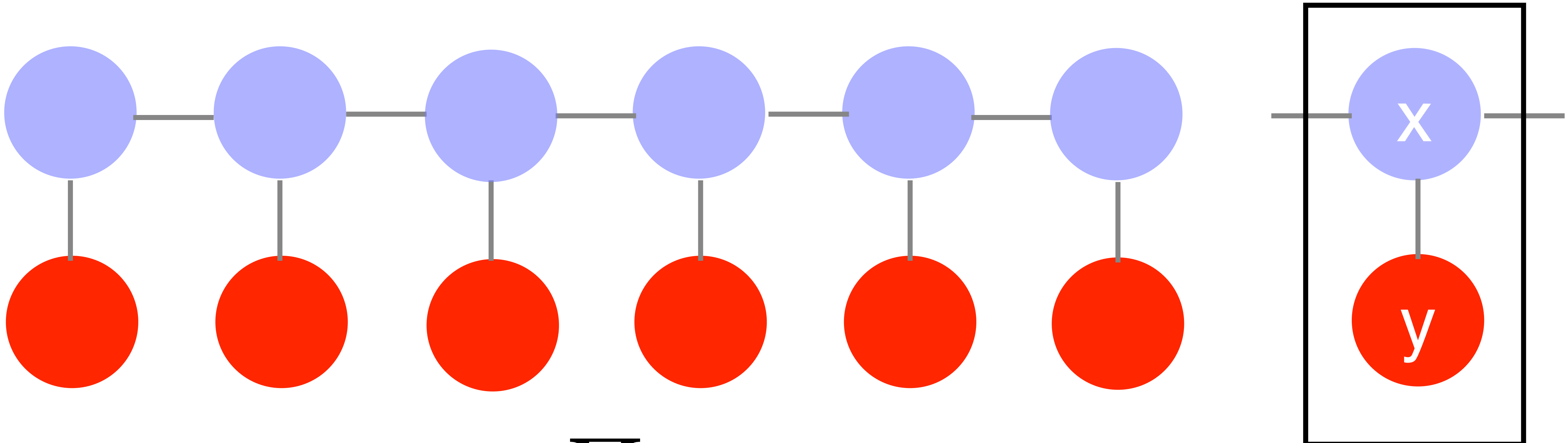




$$p(x|y) \propto \prod_i \underbrace{\psi_i^x(x_i, x_{i+1})\psi_i^{xy}(x_i, y_i)}_{=: f_i(x_i, x_{i+1})}$$

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

Chains



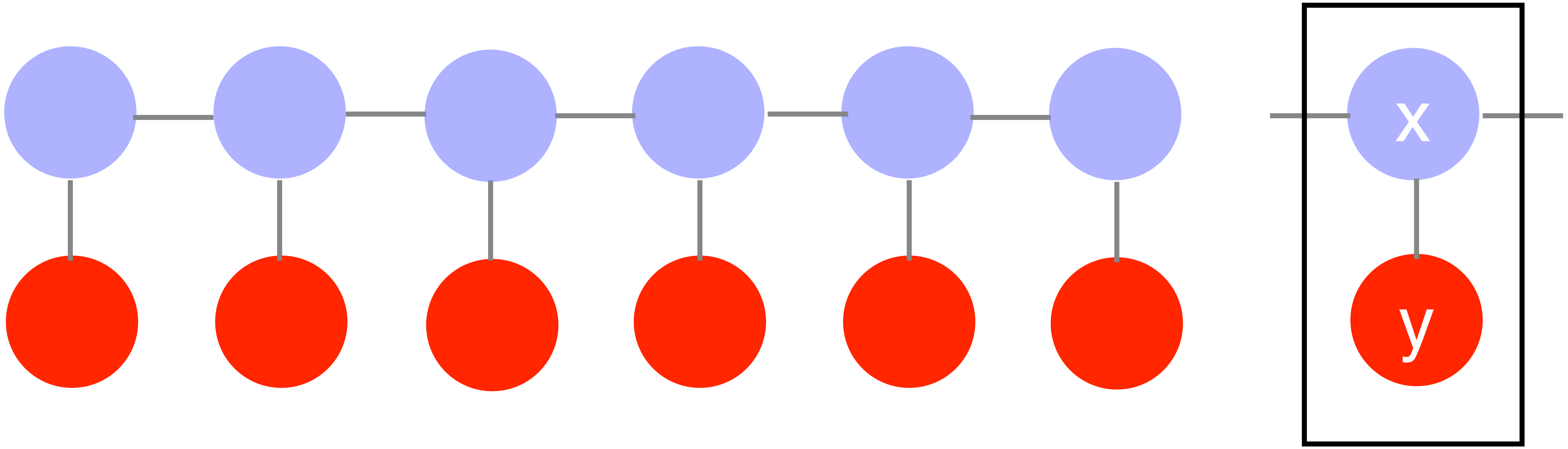
$$p(x|y) \propto \prod_i \underbrace{\psi_i^x(x_i, x_{i+1}) \psi_i^{xy}(x_i, y_i)}_{=: f_i(x_i, x_{i+1})}$$

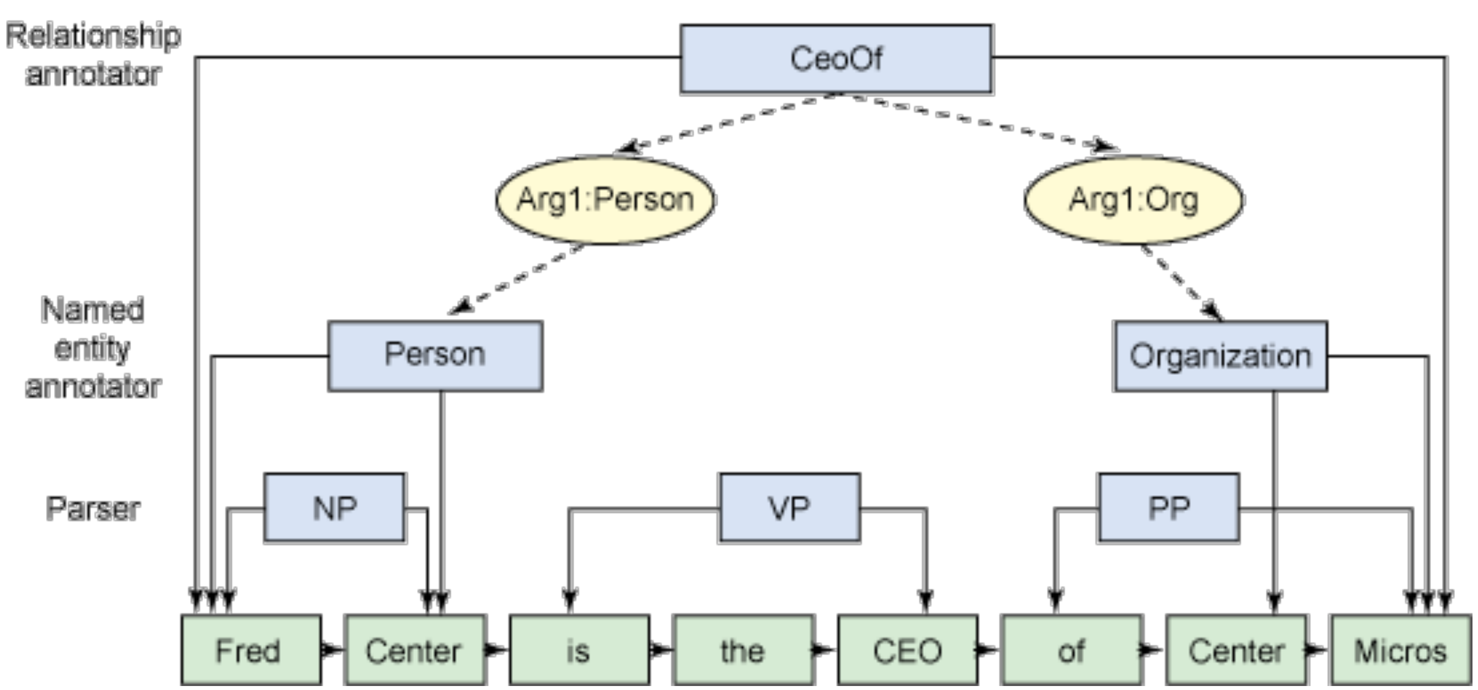
Dynamic Programming

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

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

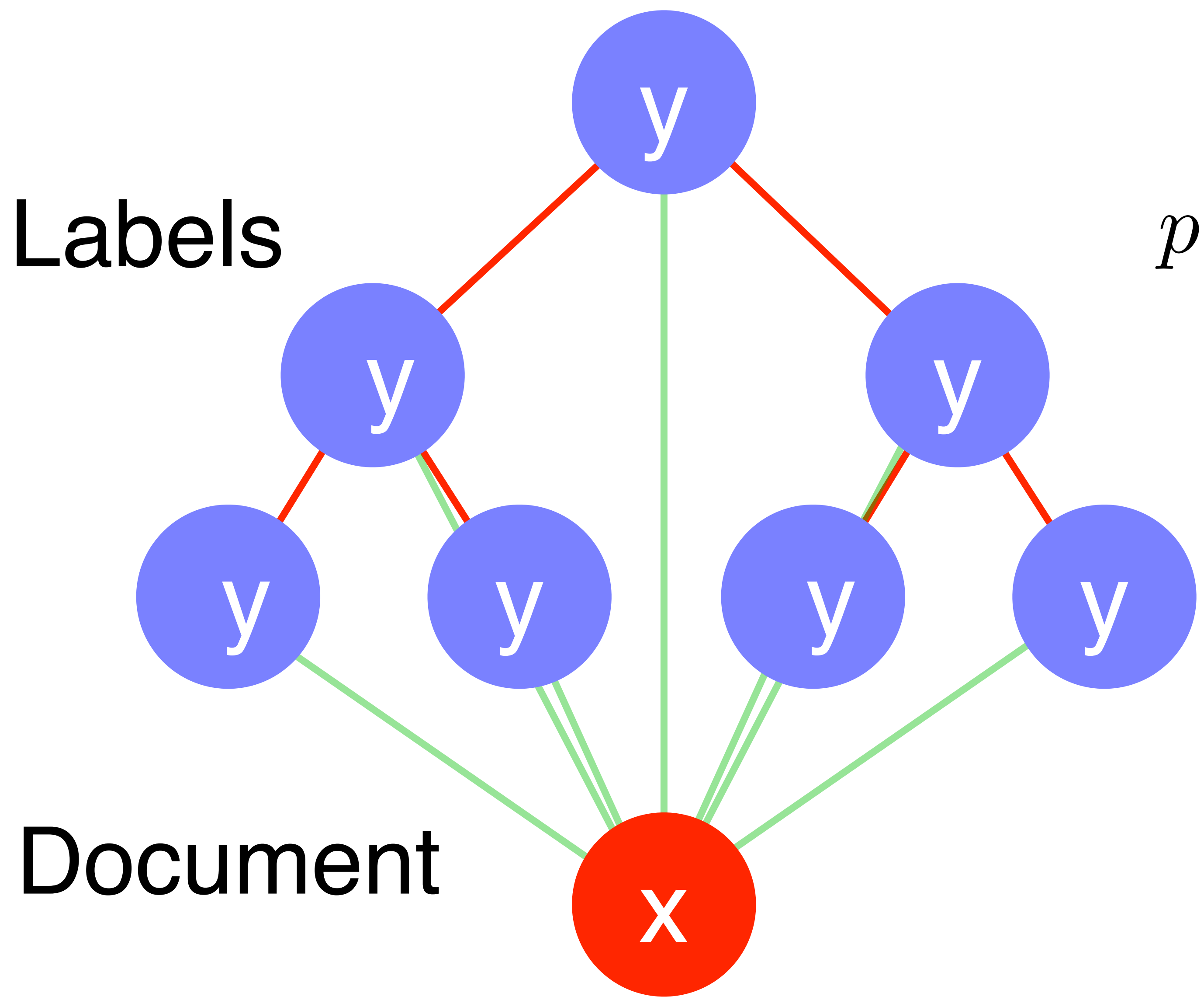
Named Entity Tagging





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

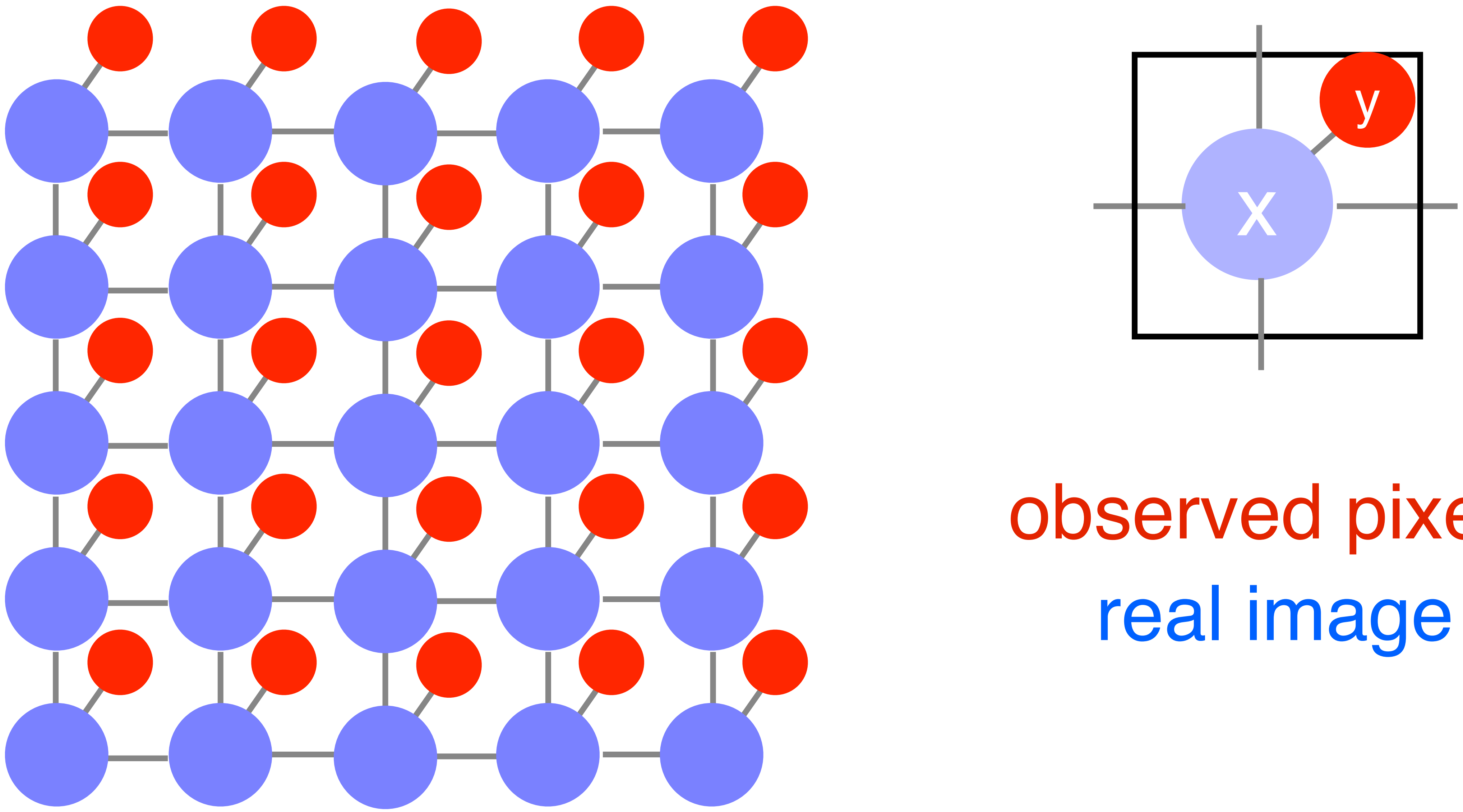
Trees + Ontologies



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

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

Spin Glasses + Images

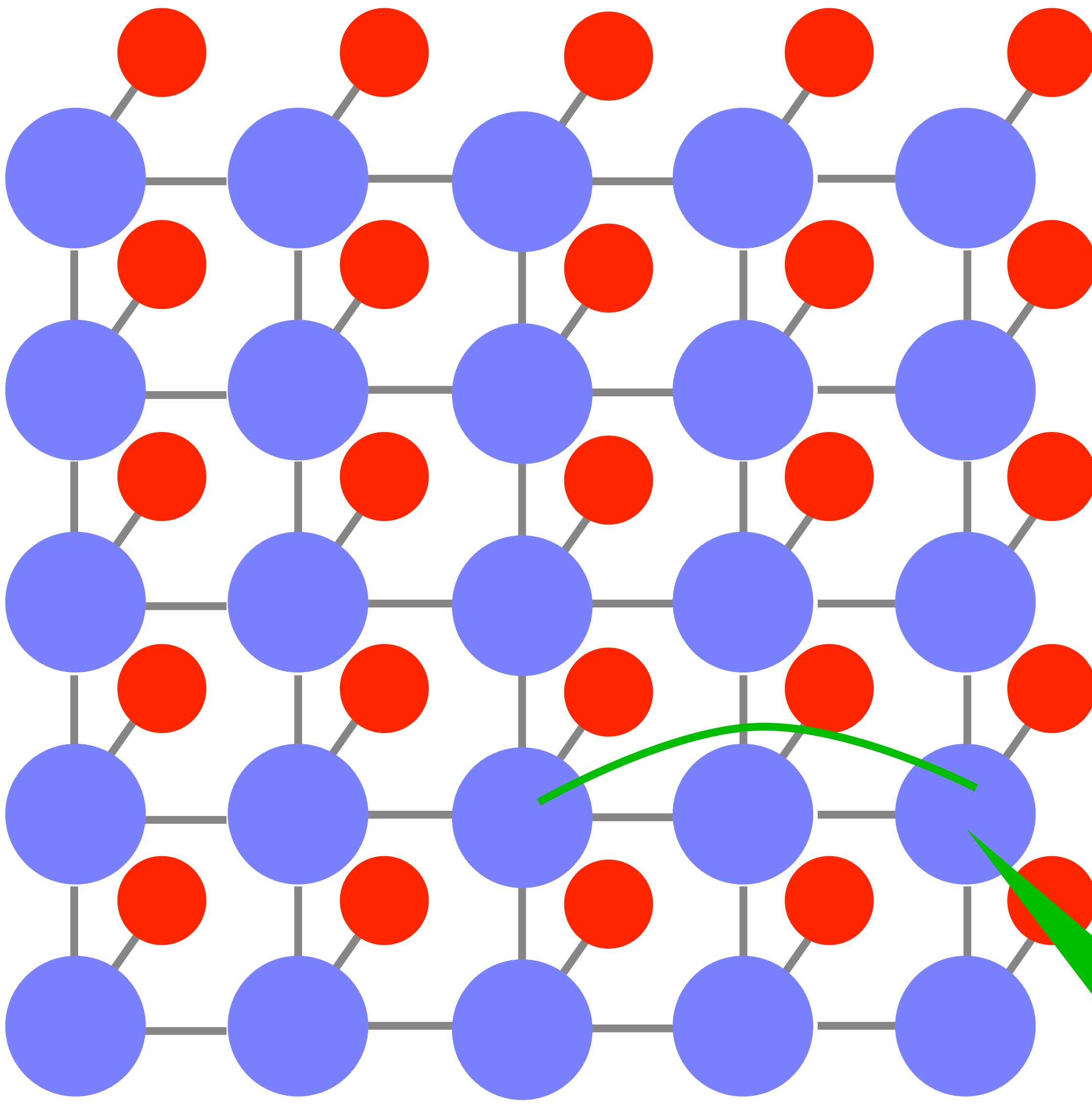


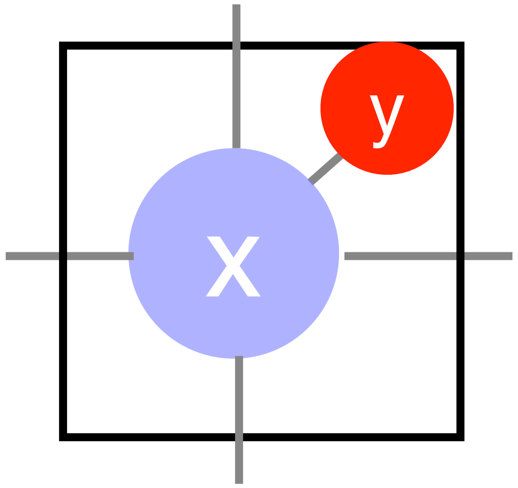
observed pixels

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



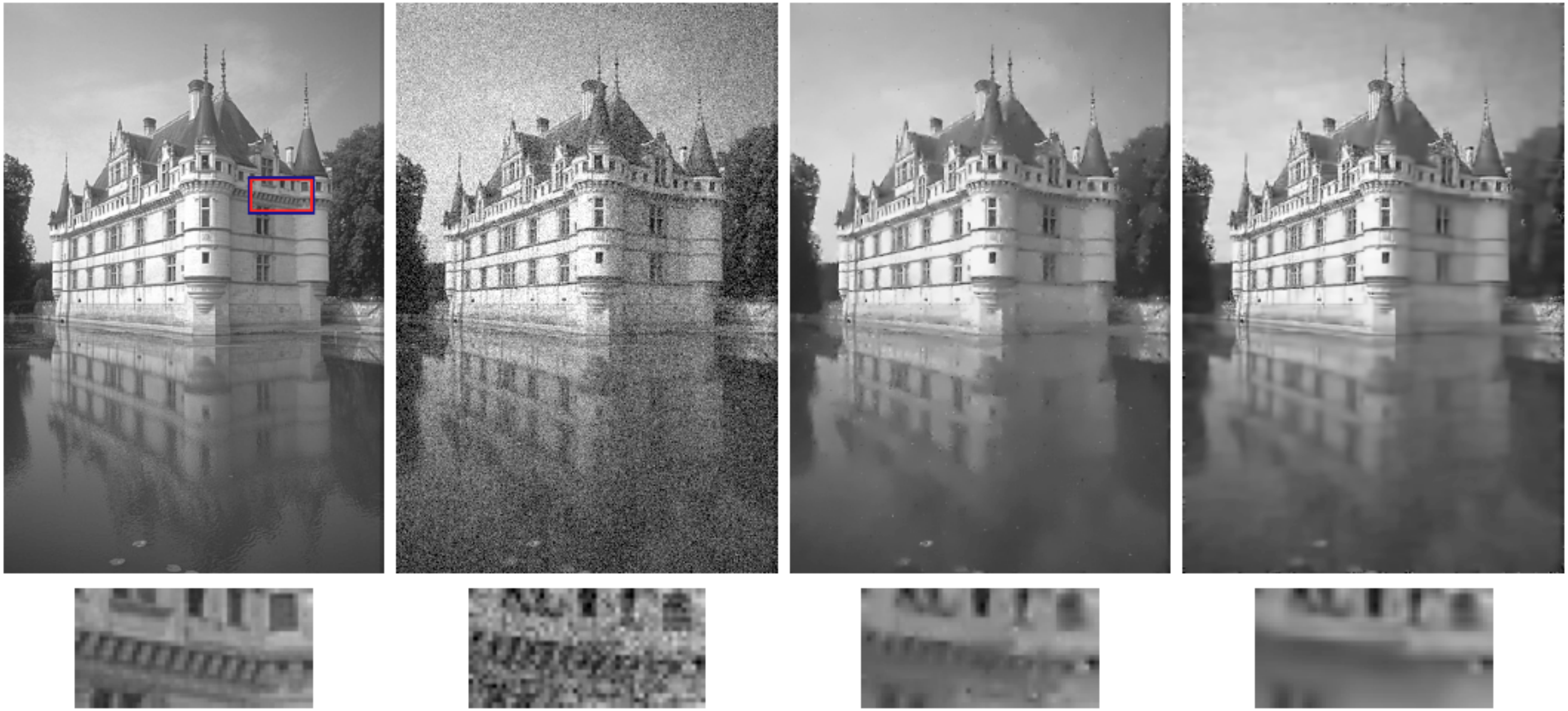


observed pixels

real image

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

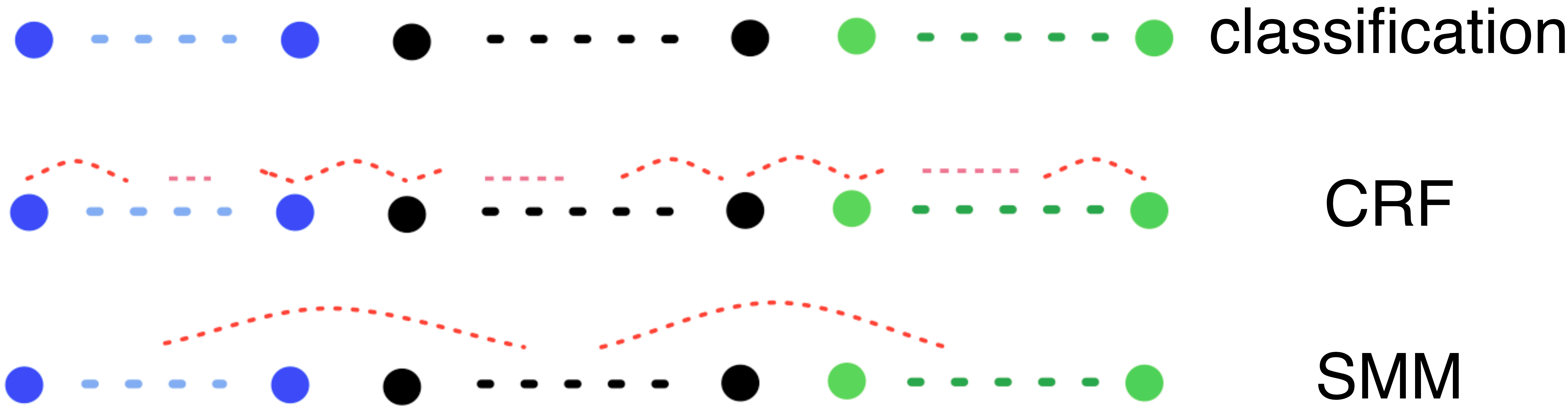
Image Denoising



Li&Huttenlocher, ECCV'08

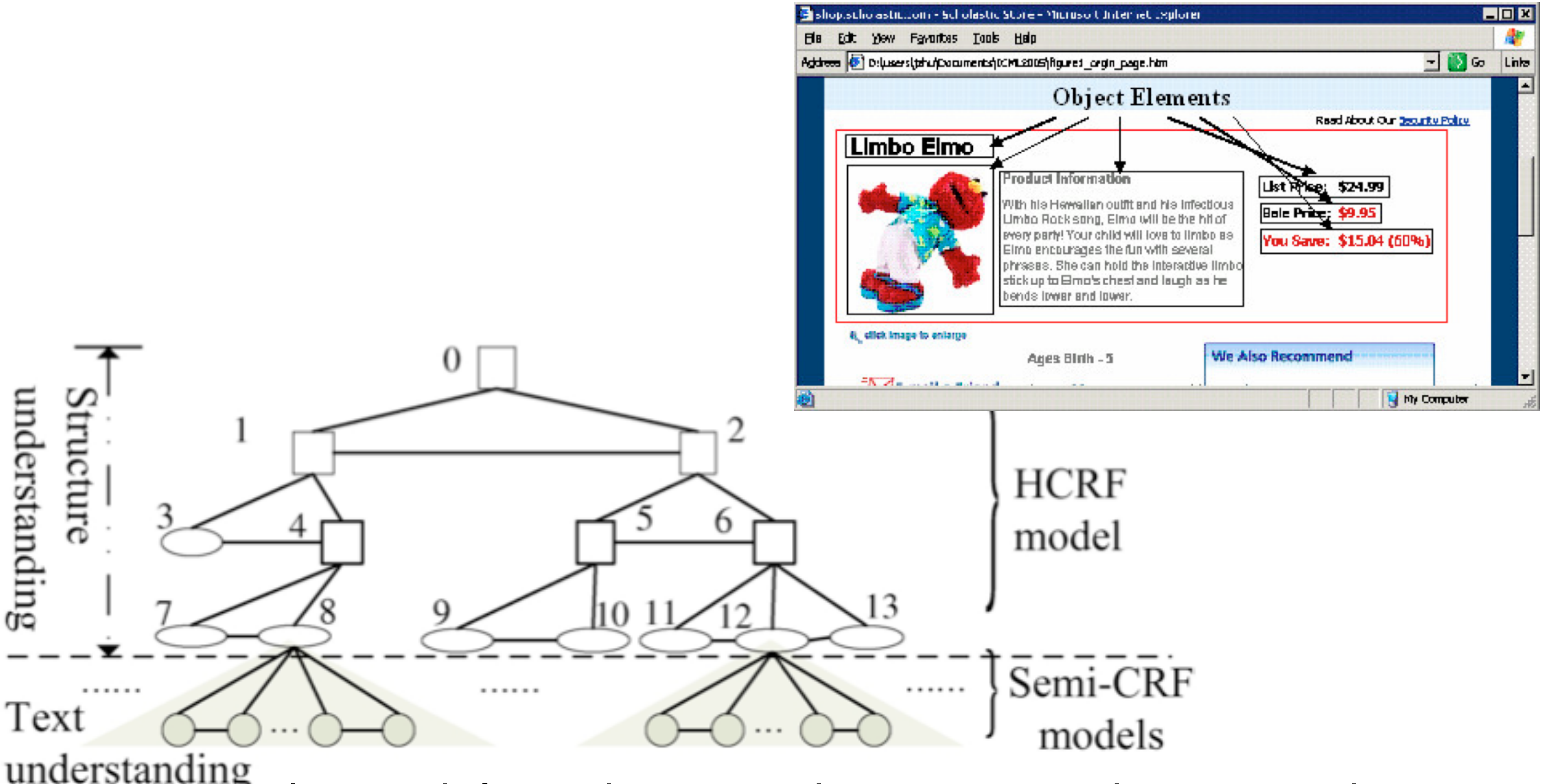
Carnegie Mellon University

Semi-Markov Models



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

Summary

- Directed Graphical Models
  - Dependence
  - Inference for fully observed models
  - Incomplete information / variational and sampling inference
- Undirected Graphical Models
  - Hammersley Clifford decomposition
  - Conditional independence
  - Junction trees
- Dynamic Programming
  - Generalized Distributive Law
  - Naive Message Passing
- Inference techniques
  - Sampling (Gibbs and Monte Carlo)
  - Variational methods (EM, extensions)