Advanced Prediction Models

Deep Learning, Graphical Models and Reinforcement Learning

Recap: Why Graphical Models

- We have seen deep learning techniques for unstructured data
  - Predominantly vision and text/audio
  - We will see control in the last part of the course
    - (Reinforcement Learning)

- For structured data, graphical models are the most versatile framework
  - Successfully applications:
    - Kalman filtering in engineering
    - Decoding in cell phones (channel codes)
    - Hidden Markov models for time series
    - Clustering, regression, classification …

Recap: Graphical Models Landscape

- Three key parts:
  - Representation
    - Capture uncertainty (joint distribution)
    - Capture conditional independences (metadata)
    - Visualization of metadata for a distribution
  - Inference
    - Efficient methods for computing marginal or conditional distributions quickly
  - Learning
    - Learning the parameters of the distribution can deal with prior knowledge and missing data

Today's Outline

- Inference
  - Factor Graph
  - Variable Elimination
- Inference using Belief Propagation
- Inference using Markov Chain Monte Carlo

Inference

Based on notes from Bjoern Andres and Bernt Schiele (2016)

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

Inference Objectives

- Let $\bar{X} = X_1, \dots.., X_D$ be a random vector.
- Let $\bar{X} \in \mathfrak{X}$ and $\mathrm{X_i} \in \mathfrak{X}_i$
- Given $P(\bar{X})$ compute functions of it

Inference Objectives

- Let $\bar{X} = X_1, \dots, X_D$ be a random vector.
- Let $\bar{X} \in \mathfrak{X}$ and $\mathrm{X_i} \in \mathfrak{X}_i$
- Given $P(\bar{X})$ compute functions of it
  - Example, find
    - Mode $\bar{x}^* \in \mathrm{argmax}_{\bar{x} \in \mathfrak{X}} P(\bar{x})$
    - Mean $\mathrm{E}[g(\bar{x})] = \sum_{\bar{x} \in \mathfrak{X}} g(\bar{x}) P(\bar{x})$
    - A marginal $\mathrm{argmax}_{x_i \in \mathfrak{X}_i} \sum_{x_1,..,x_{i-1} x_{i+1},..,x_D} P(\bar{x})$
    - A conditional $P(X_i | x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_D)$

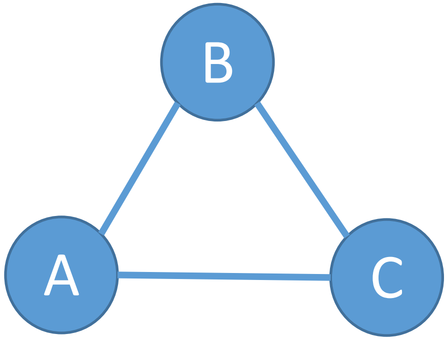
Algorithms for Inference

- Variable Elimination
- Belief Propagation
- Sampling based methods (MCMC)

[1]Note: There are others, but we will not discuss them here

Factor Graphs

- For both DPGM and UPGMs, factorization is simply not specified by the graph!
- Consider the following example graph



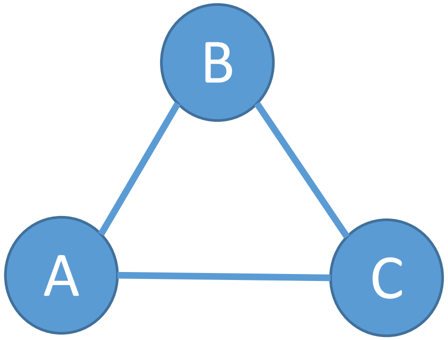
- It could be $P(a, b, c) = \frac{1}{Z}\phi(a, b, c)$
- Or it could be $P(a, b, c) = \frac{1}{Z}\phi_1(a, b)\phi_2(b, c)\phi_3(c, a)$

[1]Reference: Daphne Koller (2011)

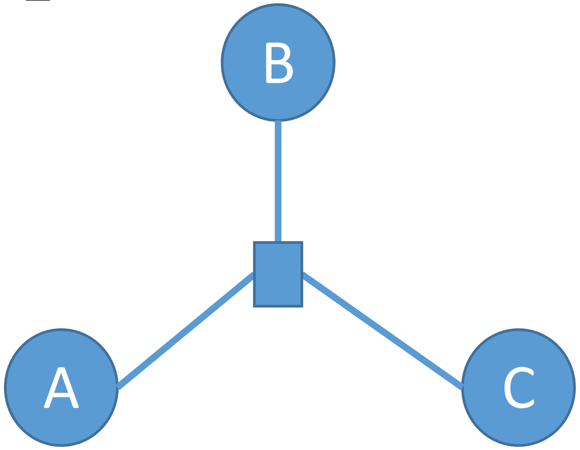
Factor Graph for UPGM

- Hence, we define new graphs called factor graphs

- Consider a square node for each factor



- Then, $P(a,b,c) = \frac{1}{Z}\phi(a,b,c)$ can be represented by



[1]Reference: Daphne Koller (2011)

Factor Graph

- factor graphs capture the factorization in the graph itself

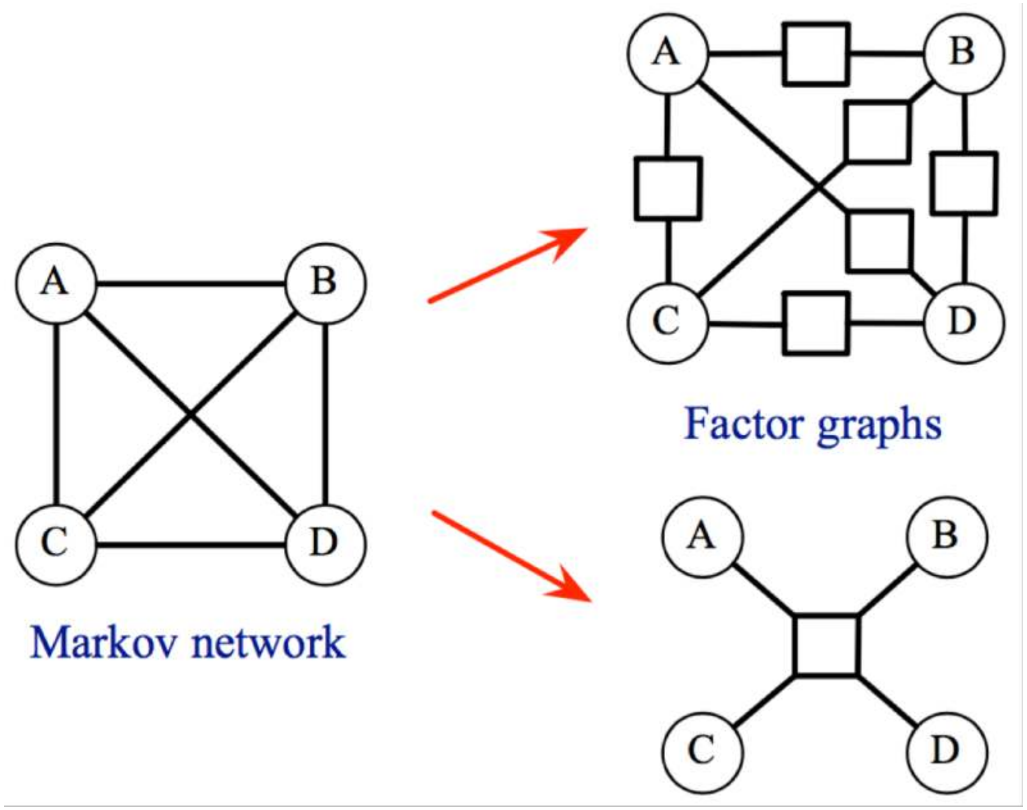
- For a function $f(x_1, .., x_D) = \prod_i \phi_i(\mathcal{X}_i)$ the factor graph has a square node for each factor $\phi_i(\mathcal{X}_i)$ and a circular variable node for each variable x_j

- Factor graphs will allow us to define inference algorithms for both DPGMs and UPGMs
  - Just a more richer way of drawing graphs for $P(X)$

[1]Reference: Daphne Koller (2011)

Factor Graphs for a UPGM

- The following example shows two factor graphs for the same UPGM

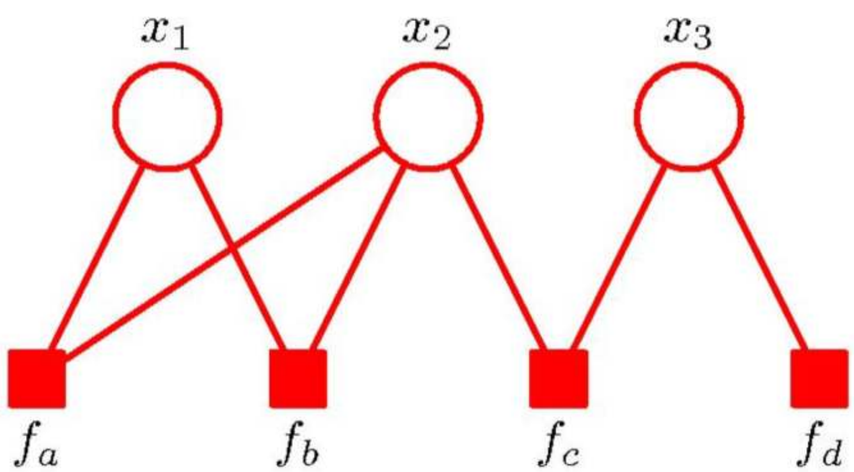


[1]Reference: David Sontag (2013)

Factor Graph Example (I)

- Which distribution does the following graph correspond to?

We will use f or ϕ to denote factors

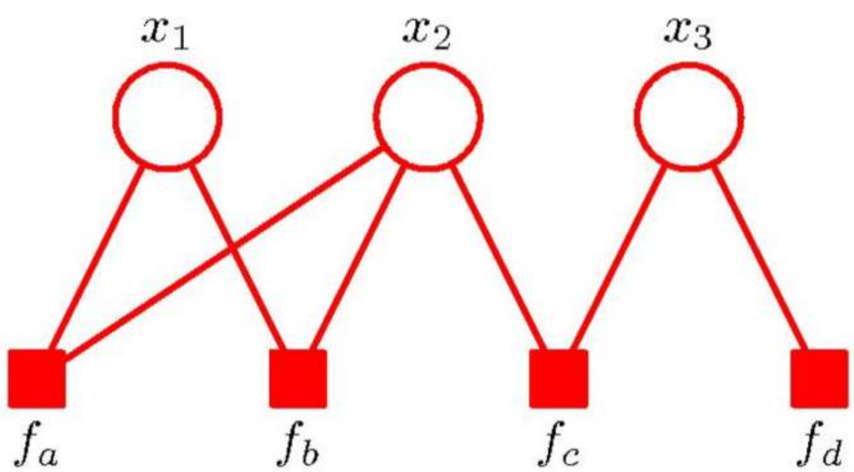


We will use lower case to minimize notation clutter

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

Factor Graph Example (I)

- Which distribution does the following graph correspond to?



- It corresponds to
  - $P(x_1, x_2, x_3) = \frac{1}{Z} f_a(x_1, x_2) f_b(x_1, x_2)\ f_c(x_2, x_3)\ f_d(x_3)$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

Factor Graph Example (II)

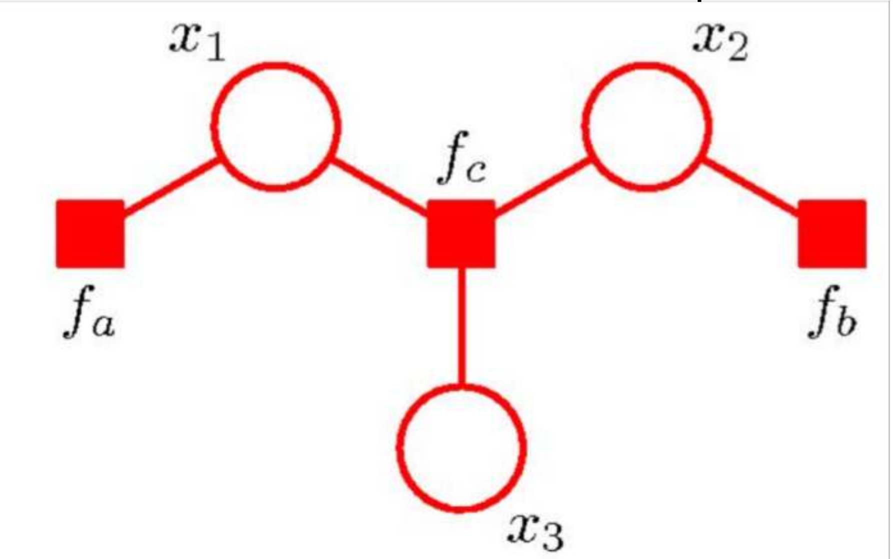
- What is the factor graph for the distribution
  - $P(x_1, x_2, x_3) = \frac{1}{Z} f_c(x_3 | x_1, x_2) f_a(x_1) f_b(x_2)$

Factor Graph Example (II)

- What is the factor graph for the distribution
  - $P(x_1, x_2, x_3) = \frac{1}{Z} f_c(x_3 | x_1, x_2) f_a(x_1) f_b(x_2)$

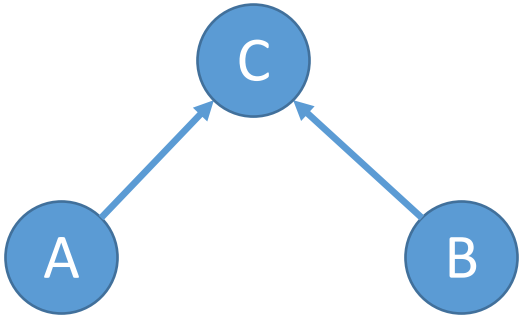
- The following is the desired factor graph

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

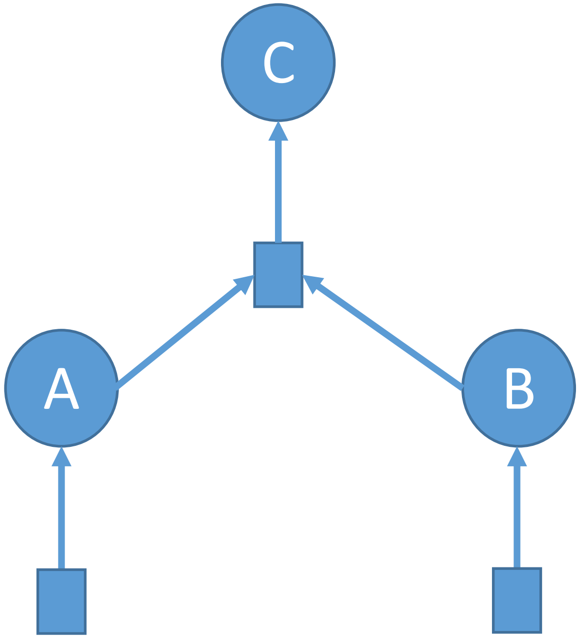


Factor Graph for DPGM

- We can do this for DPGMs as well (although redundant)
- Consider the graph on the right



- Its factor graph representation is shown below



[1]Reference: Daphne Koller (2011)

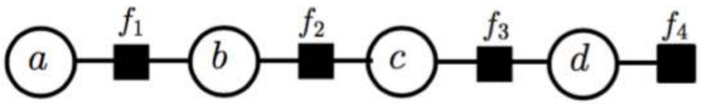
Inference using Variable Elimination

- It is a very simple idea, which is
  - Don’t sum over all configurations simultaneously
  - Do it one variable at a time

- Works for DPGMs and UPGMs

Variable Elimination Example

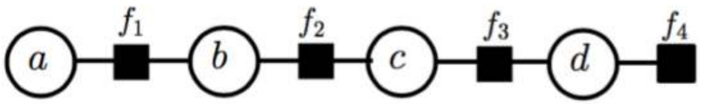
We will use lower case to minimize notation clutter



This can be for a DPGM or a UPGM

Variable Elimination Example

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)



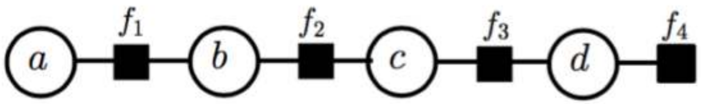
This can be for a DPGM or a UPGM

$$p(a, b, c, d) = \frac{1}{Z} f_1(a, b) f_2(b, c) f_3(c, d) f_4(d)$$

Objective: Find $p(a, b)$

Variable Elimination Example

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)



$$p(a,b,c,d) = \frac{1}{Z} f_1(a,b) f_2(b,c) f_3(c,d) f_4(d)$$

Objective: *Find* $p(a,b)$

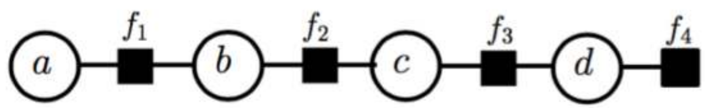
$$p(a,b,c) = \sum_d p(a,b,c,d)$$

$$= \frac{1}{Z} f_1(a,b) f_2(b,c) \underbrace{\sum_d f_3(c,d) f_4(d)}_{\mu_{d\to c}(c)}$$

(compute this for all c)

Variable Elimination Example

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)



$$p(a,b,c,d) = \frac{1}{Z} f_1(a,b) f_2(b,c) f_3(c,d) f_4(d)$$

Objective: *Find* $p(a,b)$

$$p(a,b,c) = \sum_d p(a,b,c,d)$$

$$= \frac{1}{Z} f_1(a,b) f_2(b,c) \underbrace{\sum_d f_3(c,d) f_4(d)}_{\mu_{d \to c}(c)}$$

(compute this for all c)

$$p(a,b) = \sum_c p(a,b,c) = \frac{1}{Z} f_1(a,b) \underbrace{\sum_c f_2(b,c) \mu_{d \to c}(c)}_{\mu_{c \to b}(b)}$$

(compute this for all b)

Questions?

Today's Outline

- Inference
  - Factor Graph
  - Variable Elimination
- Inference using Belief Propagation
- Inference using Markov Chain Monte Carlo

Inference using Belief Propagation

Belief Propagation (BP)

- Generalizes the idea of Variable Elimination
- Also called the Sum-Product Algorithm

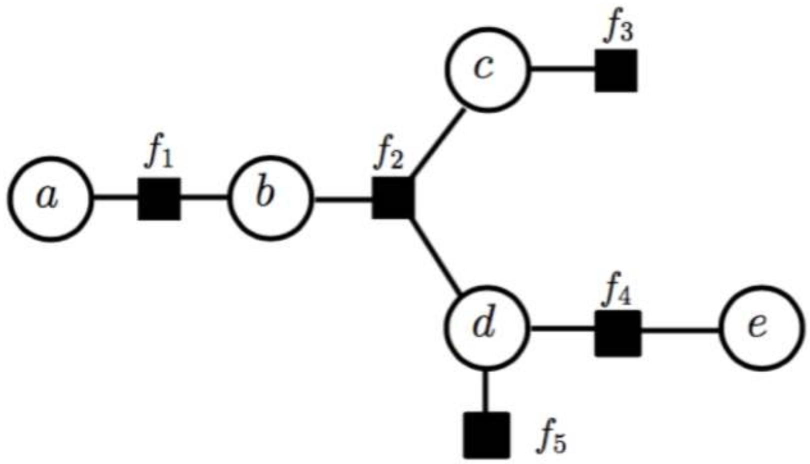
- Will give exact answers (marginals, conditionals) on factor graphs that are trees

- Can also be used for general graphs but may give wrong answers

[1]Reference: See https://en.wikipedia.org/wiki/Belief_propagation

BP Example: Compute a Marginal

consider a branching graph:



with factors

$$f_1(a,b)f_2(b,c,d)f_3(c)f_4(d,e)f_5(d)$$

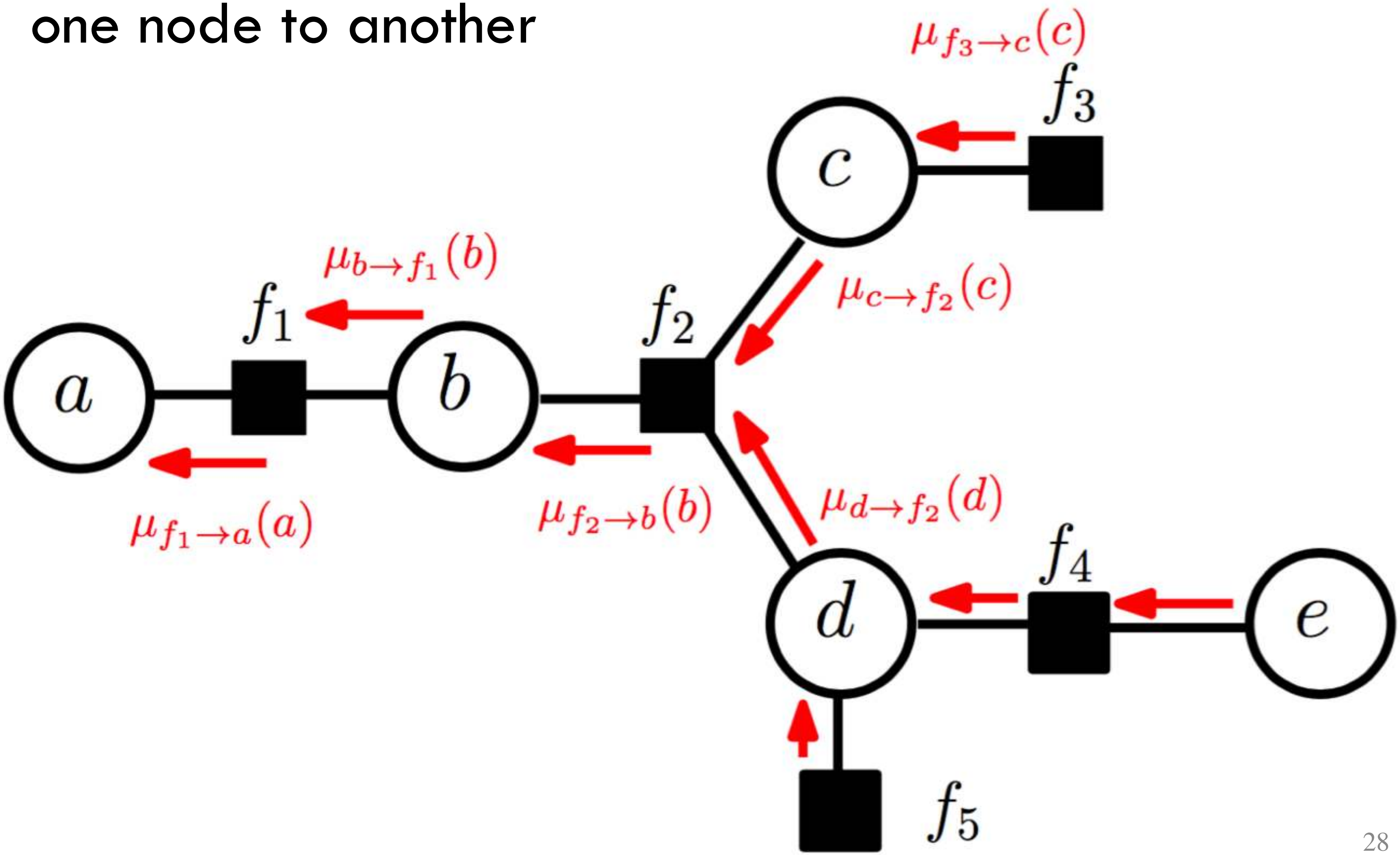
For example: find marginal $p(a,b)$

- We will introduce the notion of
  - messages, and
  - message passing

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

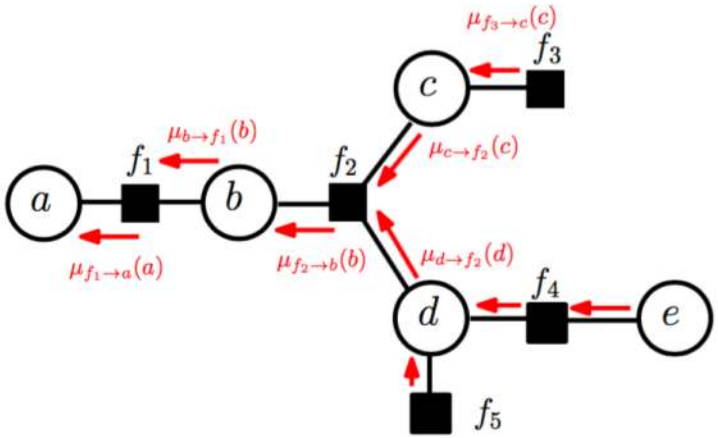
BP Example: Messages

- Messages are functions (vectors) that are passed from one node to another



[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP Example: Messages



$$p(a,b) = \frac{1}{Z} f_1(a,b) \underbrace{\sum_{c,d,e} f_2(b,c,d) f_3(c) f_5(d) f_4(d,e)}_{\mu_{f_2\to b}(b)}$$

$$\mu_{f_2\to b}(b) = \sum_{c,d} f_2(b,c,d) \underbrace{f_3(c)}_{\mu_{c\to f_2}(c)} \underbrace{f_5(d) \sum_{e} f_4(d,e)}_{\mu_{d\to f_2}(d)}$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP Example: Message from Factor to Variable

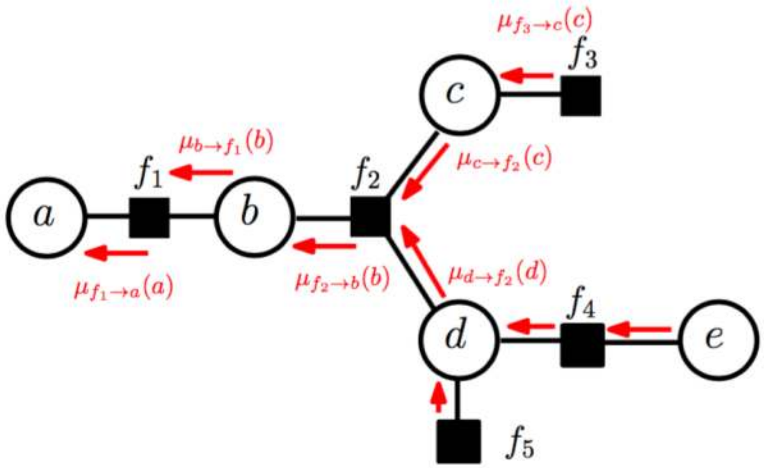
Here (repeated from last slide):

$$\mu_{f_2 \to b}(b) = \sum_{c,d} f_2(b,c,d) \underbrace{f_3(c)}_{\mu_{c \to f_2}(c)} \underbrace{f_5(d) \sum_{e} f_4(d,e)}_{\mu_{d \to f_2}(d)}$$

$$\mu_{f_2 \to b}(b) = \sum_{c,d} f_2(b,c,d)\mu_{c \to f_2}(c)\mu_{d \to f_2}(d)$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP Example: Message from Factor to Variable



Here (repeated from last slide):

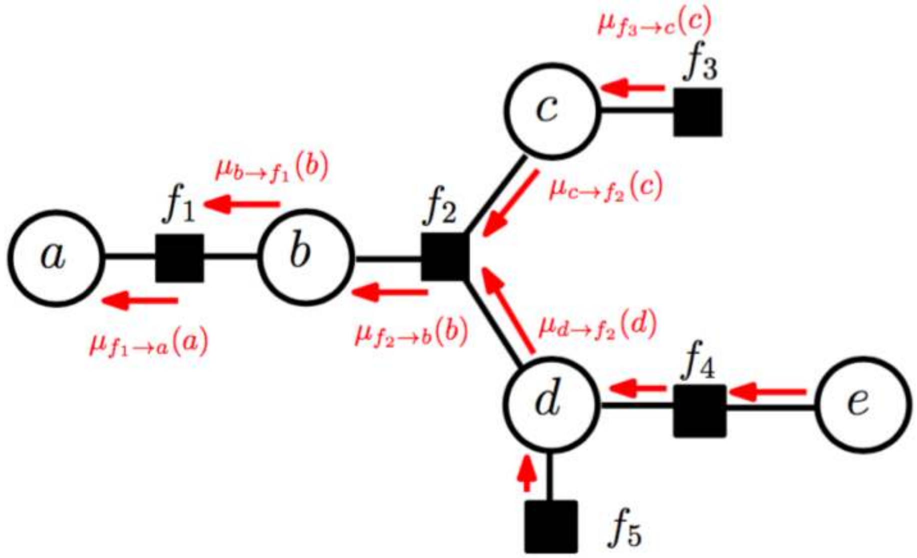
$$\mu_{f_2\to b}(b) = \sum_{c,d} f_2(b,c,d)\mu_{c\to f_2}(c)\mu_{d\to f_2}(d)$$

more general:

$$\mu_{f\to x}(x) = \sum_{y\in\mathcal{X}_f\setminus x} \phi_f(\mathcal{X}_f) \prod_{y\in\{\text{ne}(f)\setminus x\}} \mu_{y\to f}(y)$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP Example: Message from Variable to Factor

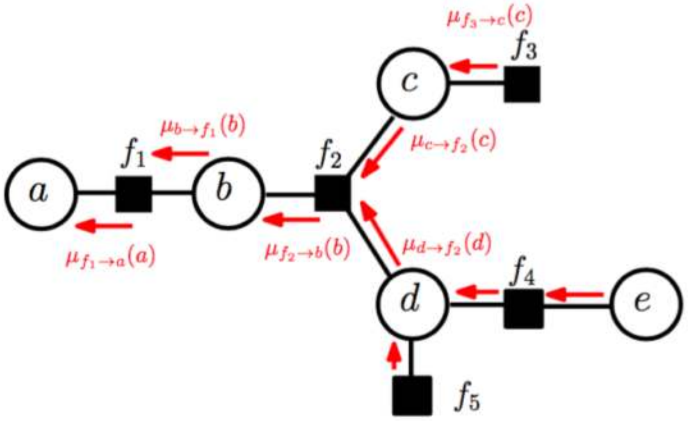


$$\mu_{d \to f_2}(d) = \underbrace{f_5(d)}_{\mu_{f_5 \to d}(d)} \underbrace{\sum_e f_4(d, e)}_{\mu_{f_4 \to d}(d)}$$

$$\mu_{d \to f_2}(d) = \mu_{f_5 \to d}(d) \mu_{f_4 \to d}(d)$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP Example: Message from Variable to Factor



Here (repeated from last slide):

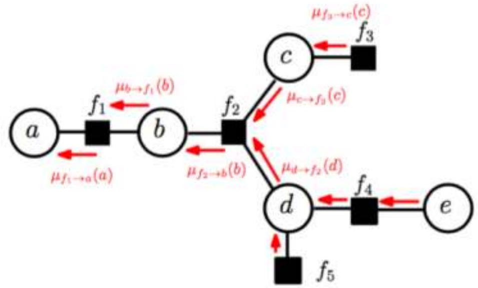
$$\mu_{d\to f_2}(d) = \mu_{f_5\to d}(d)\mu_{f_4\to d}(d)$$

General:

$$\mu_{x\to f}(x) = \prod_{g\in\{\text{ne}(x)\setminus f\}} \mu_{g\to x}(x)$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP Example: Compute a Different Marginal



If we want to compute the marginal $p(a)$ (use factor-to-variable message):

$$p(a) = \frac{1}{Z}\mu_{f_1 \to a}(a) = \underbrace{\sum_b f_1(a,b)\mu_{b \to f_1}(b)}_{\mu_{f_1 \to a}(a)} \frac{1}{Z}$$

which we could also view as

$$p(a) = \frac{1}{Z}\sum_b f_1(a,b)\underbrace{\mu_{b \to f_1}(b)}_{\mu_{f_2 \to b}(b)}$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

Belief Propagation Algorithm

- We described the concept of ‘messages’ via an example (computing marginals for a given factor graph)

- Now we will summarize the algorithm in general
- It has three key ingredients
  - Initialization
  - Variable to factor message
  - Factor to variable message

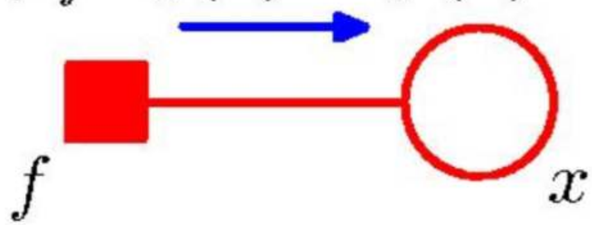
- Don’t forget the original objective: efficient inference

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP: Initialization

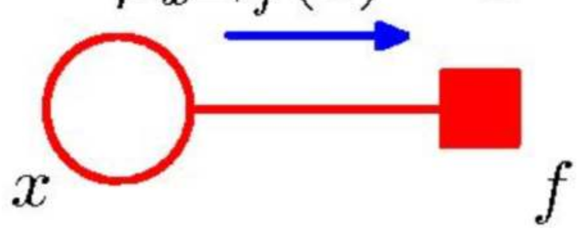
- Messages from extremal/leaf node factors are initialized to be the factor itself

$$\mu_{f \to x}(x) = f(x)$$



- Messages from extremal/leaf node variables are initialized to value 1

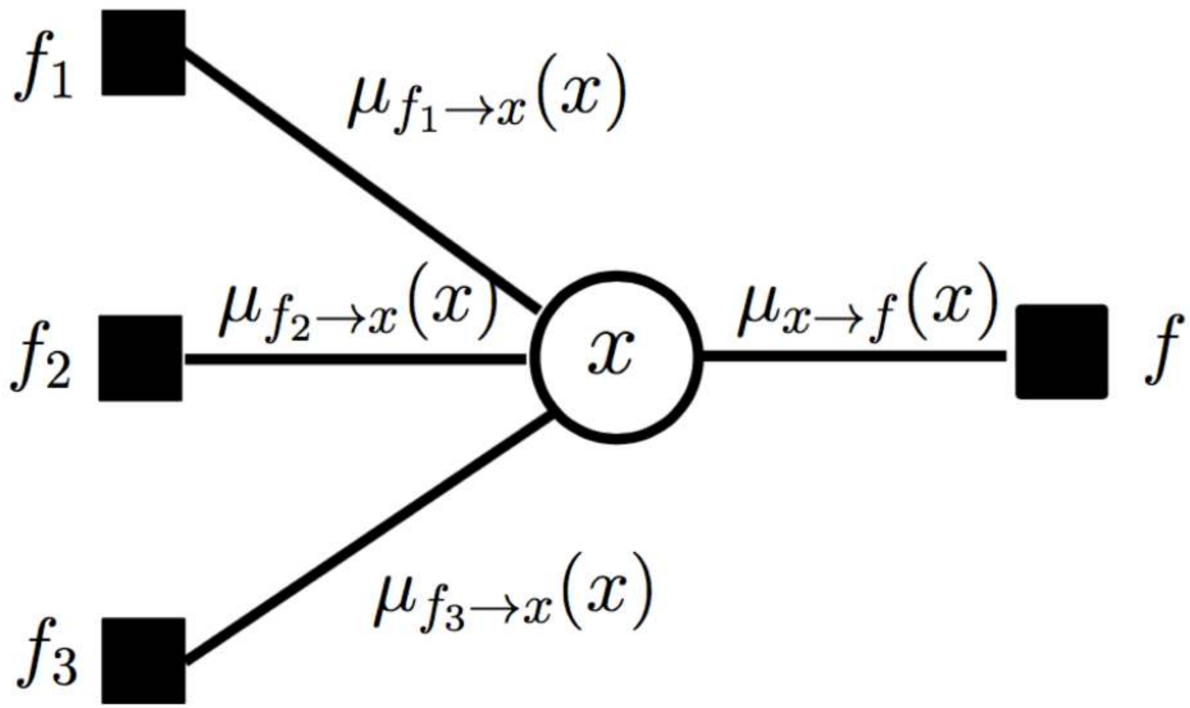
$$\mu_{x \to f}(x) = 1$$



[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP: Variable to Factor Message

$$\mu_{x\to f}(x) = \prod_{g \in \{\mathrm{ne}(x) \setminus f\}} \mu_{g\to x}(x)$$

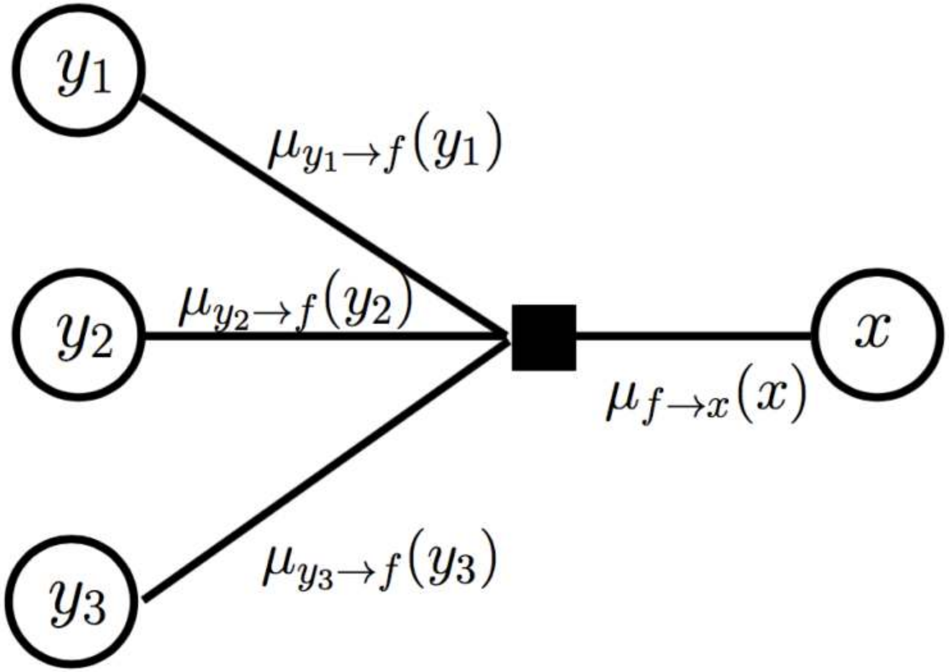


[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP: Factor to Variable Message

- We sum over all values possible in the scope of the factor

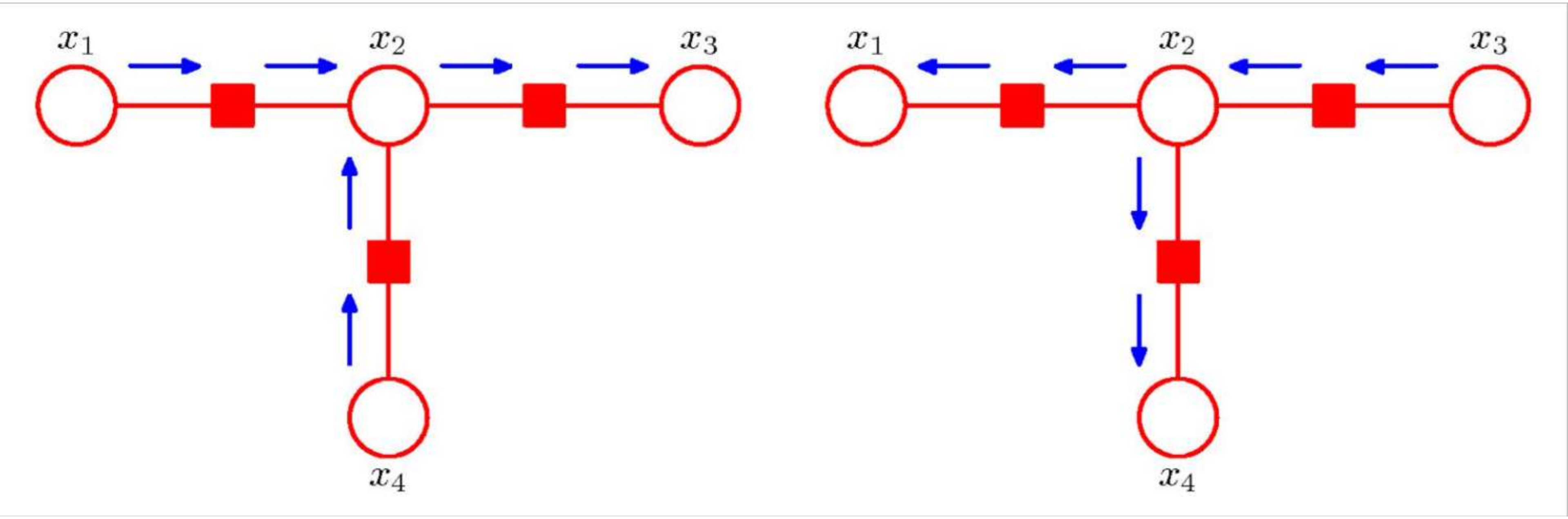
$$\mu_{f \to x}(x) = \sum_{y \in \mathcal{X}_f \setminus x} \phi_f(\mathcal{X}_f) \prod_{y \in \{\mathrm{ne}(f) \setminus x\}} \mu_{y \to f}(y)$$



[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP: Ordering of Messages

- Messages depend on all incoming messages
- To compute all messages
  - Go from leaves to a designated root (say x_3)
  - Go from the designated root back to leaves



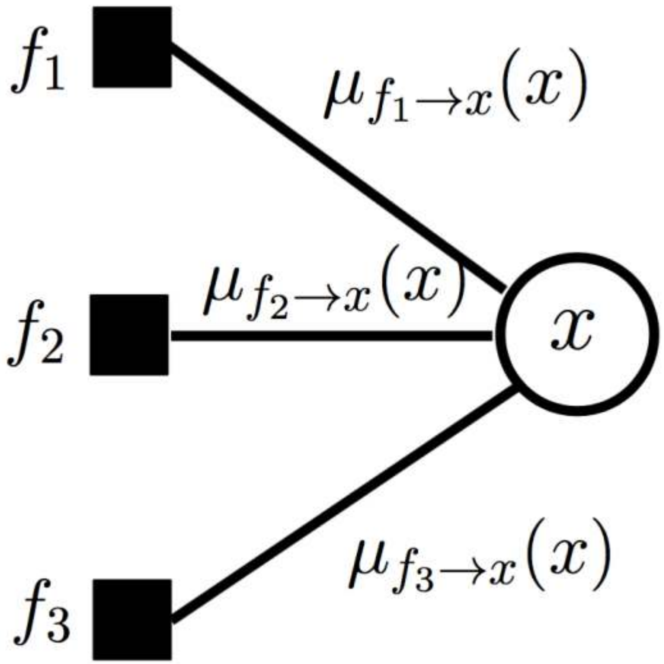
Designated root: x_3

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP: Computing a Marginal

- Marginal is simply the product of messages the variable of interest receives

$$p(x) \propto \prod_{f \in \text{ne}(x)} \mu_{f \to x}(x)$$



[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP: General Factor Graphs

- Is in-exact

- Since it is not clear whether BP is a clear winner for inference with general graphs (among competing algorithms), we will not explore this further.

- See https://en.wikipedia.org/wiki/Belief_propagation for more details

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

Questions?

Today's Outline

- Inference
  - Factor Graphs
  - Variable Elimination
- Inference using Belief Propagation
- Inference using Markov Chain Monte Carlo

Inference using Markov Chain Monte Carlo

See https://en.wikipedia.org/wiki/Markov_chain_Monte_Carlo

Approximate Inference

- BP and Variable Elimination are exact algorithms
- They work for tree structured factor graphs

- We will resort to numerical sampling to perform approximate inference for general graphical models
  - Essentially, use random sampling to approximate

Sampling

- Many methods in the literature

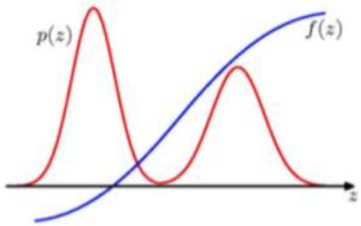
- Monte Carlo methods
  - MC Averaging and Importance sampling
  - Rejection sampling

- Markov Chain Monte Carlo methods
  - Gibbs sampling
  - Metropolis-Hastings sampling

- …

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

Monte Carlo Averaging



We want to evaluate

$$\mathbb{E}[f] = \int f(x)p(x)dx \quad \text{or } \mathbb{E}[f] = \sum_{x \in \mathcal{X}} f(x)p(x)$$

Sampling idea:

- draw L independent samples $x^1, x^2, \ldots, x^L$ from $p(\cdot)$: $x^l \sim p(\cdot)$
- replace the integral/sum with the finite set of samples

$$\hat{f} = \frac{1}{L} \sum_{l=1}^{L} f(x^l)$$

- as long as $x^l \sim p(\cdot)$ then

$$\mathbb{E}[\hat{f}] = \mathbb{E}[f]$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Monte_Carlo_method

Importance Sampling

- Is a variance reduction technique for MC averaging

use a proposal distribution $q(z)$ from which it is easy to draw samples

express expectation in the form of a finite sum over samples $\{z^l\}$ drawn from $q(z)$:

$$\mathbb{E}[f] = \int f(z)p(z)dz = \int f(z)\frac{p(z)}{q(z)}q(z)dz$$
$$\simeq \frac{1}{L}\sum_{l=1}^{L}\frac{p(z^l)}{q(z^l)}f(z^l)$$

with importance weights: $r^l = \frac{p(z^l)}{q(z^l)}$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Importance_sampling

Importance Sampling

- If we can only evaluate up to a normalizing constant, then additional tricks needed.

$p(z)$ can be only evaluated up to a normalization constant (unkown):

$$p(z) = \tilde{p}(z)/Z_p$$

$q(z)$ can be also treated in a similar way:

$$q(z) = \tilde{q}(z)/Z_q$$

then:

$$\mathbb{E}[f] = \int f(z)p(z)dz = \frac{Z_q}{Z_p}\int f(z)\frac{\tilde{p}(z)}{\tilde{q}(z)}q(z)dz$$

$$\simeq \frac{Z_q}{Z_p}\frac{1}{L}\sum_{l=1}^{L}\tilde{r}^l f(z^l)$$

with: $\tilde{r}^l = \frac{\tilde{p}(z^l)}{\tilde{q}(z^l)}$

For example, $\frac{Z_p}{Z_q} \simeq \frac{1}{L}\sum_{l=1}^{L}\tilde{r}^l$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Importance_sampling

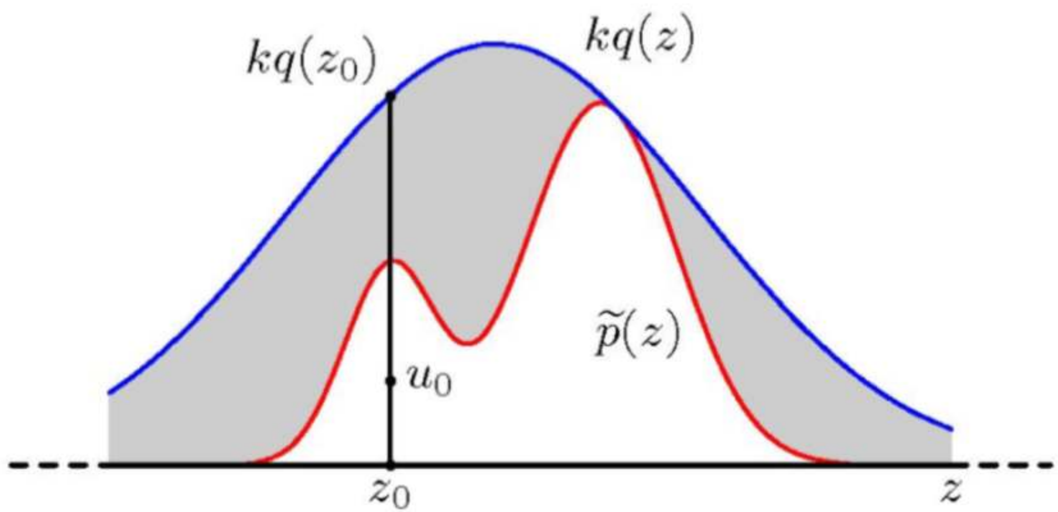
Rejection Sampling

Sample two random variables:

1. $z_0 \sim q(x)$
2. $u_0 \sim [0, kq(z_0)]$ uniform

reject sample z_0 if $u_0 > \tilde{p}(z_0)$

$q(x)$ is a proposal distribution such that $kq(x) \geq p(x) \forall x$



[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Rejection_sampling

Rejection Sampling

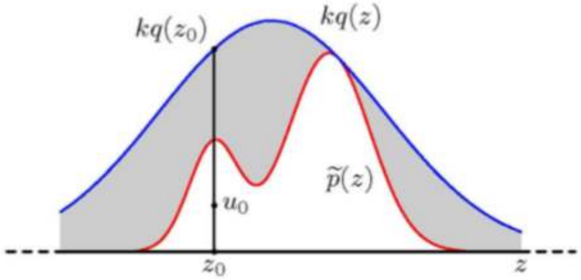
Sample z drawn from q and accepted with probability $\tilde{p}(z)/kq(z)$

So (overall) acceptance probability

$$p(accept) = \int \frac{\tilde{p}(z)}{kq(z)} q(z) dz = \frac{1}{k} \int \tilde{p}(z) dz$$

So the lower k the better (more acceptance)

- subject to constraint $kq(z) \geq \tilde{p}(z)$



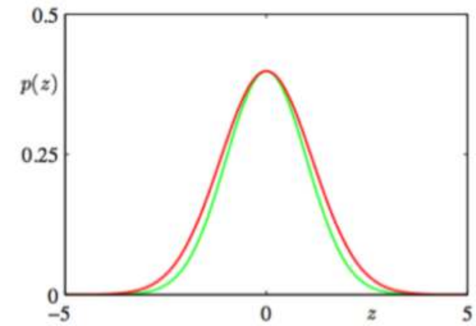
- Impractical in high dimensions (lots of samples will get rejected)

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

Rejection Sampling

Example:

- assume $p(x)$ is Gaussian with covariance matrix: $\sigma_p^2 I$
- assume $q(x)$ is Gaussian with covariance matrix: $\sigma_q^2 I$
- clearly: $\sigma_q^2 \geq \sigma_p^2$
- in D dimensions: $k = \left(\frac{\sigma_q}{\sigma_p}\right)^D$



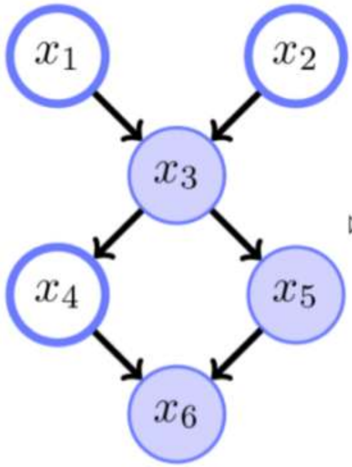
assume:

- σ_q is 1% larger than σ_p, $D = 1000$
- then $k = 1.01^{1000} \geq 20000$
- and $p(accept) \leq \frac{1}{20000}$

therefore: often impractical to find good proposal distribution $q(x)$ for high dimensions

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

Gibbs Sampling: Markov Blanket



Sample from this distribution $p(x)$

Idea: Sample sequence $x^0, x^1, x^2, \ldots$ by updating one variable at a time

Eg. update x_4 by conditioning on the set of shaded variables Markov blanket

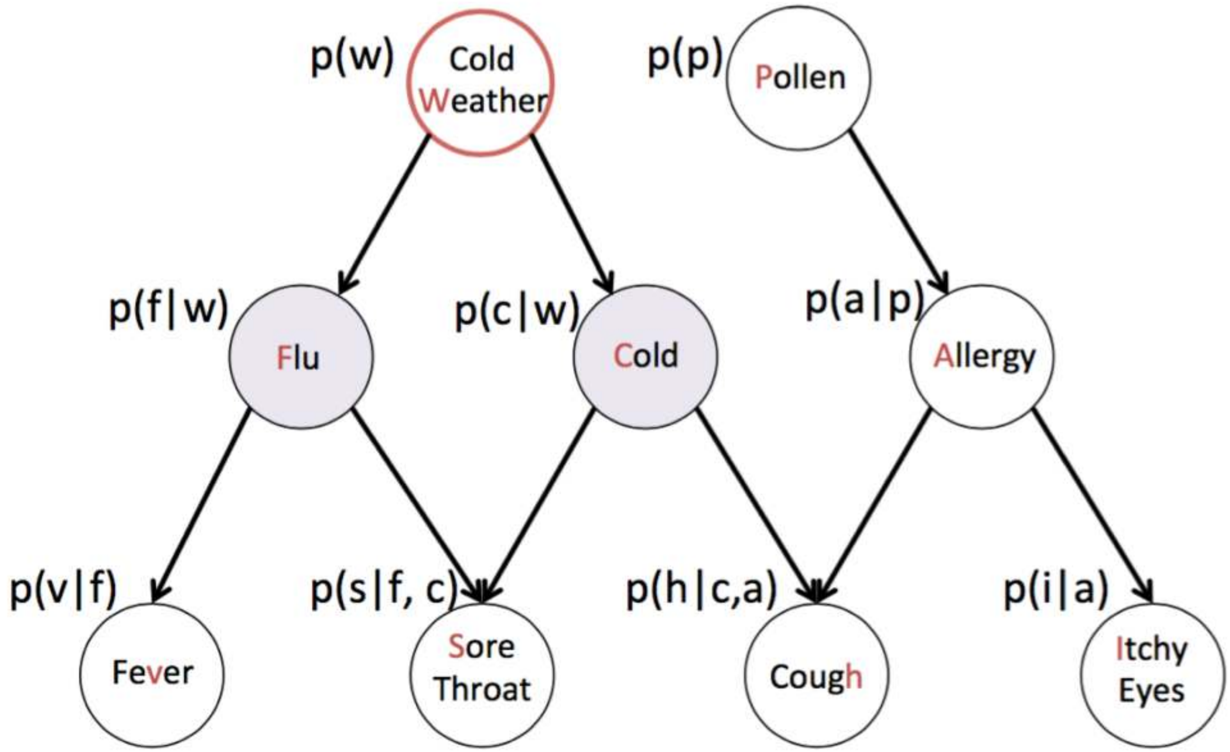
$$p(x_4 \mid x_1, x_2, x_3, x_5, x_6) = p(x_4 \mid x_3, x_5, x_6)$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Gibbs_sampling

Gibbs Sampling Example I

How do we sample a new value for W?



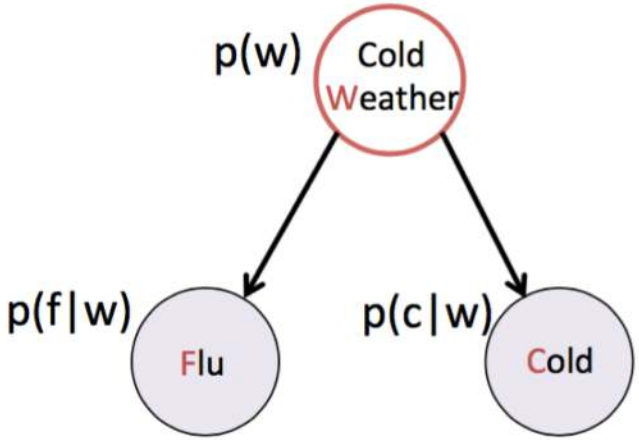
$$P(W=w|F=1, P=1, C=0, \ldots, I=0)$$

$$= P(W=w|F=1, C=0)$$

Markov Blanket!

[1]Reference: Percy Liang, CS221 (2015)

Gibbs Sampling Example I



w	p(w)
0	0.4
1	0.6

w	f	p(f|w)
0	0	0.95
0	1	0.05
1	0	0.80
1	1	0.20

w	c	p(f|w)
0	0	0.88
0	1	0.12
1	0	0.70
1	1	0.30

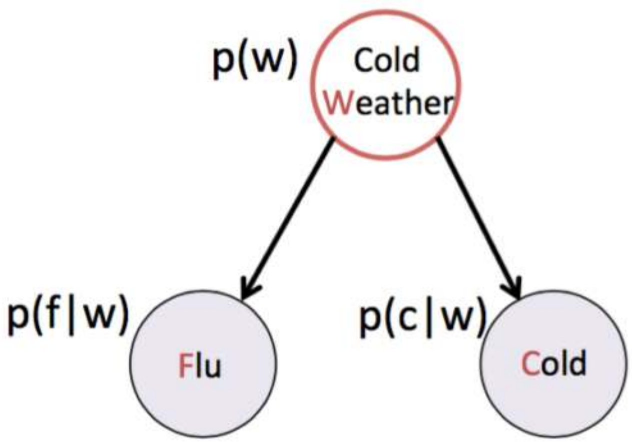
$$P(W=w|F=1, P=1, C=0, \ldots, I=0)$$

$$= P(W=w|F=1, C=0)$$

$$\propto P(F=1|W=w) * P(C=0|W=w) * P(W=w)$$

[1] Reference: Percy Liang, CS221 (2015)

Gibbs Sampling Example I



w	p(w)
0	0.40
1	0.60

w	f	p(f|w)
0	0	0.95
0	1	0.05
1	0	0.80
1	1	0.20

w	c	p(f|w)
0	0	0.88
0	1	0.12
1	0	0.70
1	1	0.30

$$P(W{=}w \mid F{=}1, P{=}1, C{=}0, \ldots, I{=}0)$$

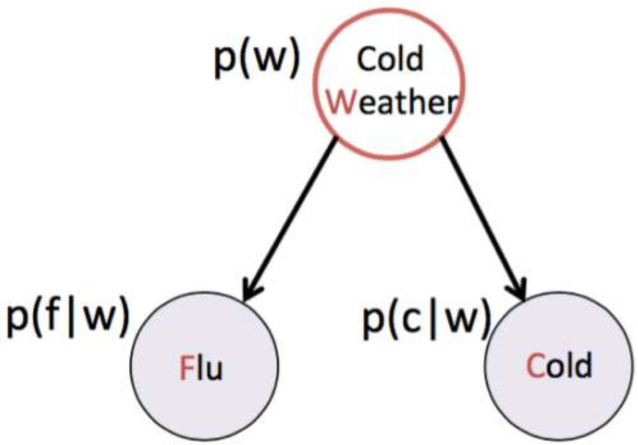
$$= P(W{=}w \mid F{=}1, C{=}0)$$

$$\propto P(F{=}1 \mid W{=}w) * P(C{=}0 \mid W{=}w) * P(W{=}w)$$

$$= \begin{cases} 0.05 * 0.88 * 0.40, & W = 0 \\ & \end{cases}$$

[1]Reference: Percy Liang, CS221 (2015)

Gibbs Sampling Example I



w	p(w)
0	0.40
1	0.60

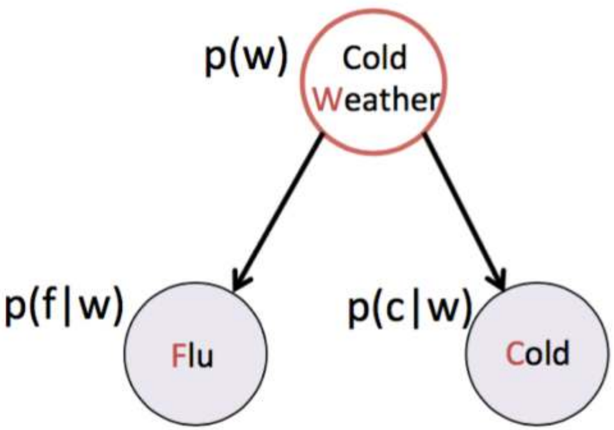
w	f	p(f|w)
0	0	0.95
0	1	0.05
1	0	0.80
1	1	0.20

w	c	p(f|w)
0	0	0.88
0	1	0.12
1	0	0.70
1	1	0.30

$$
\begin{aligned}
& P(W=w|F=1, P=1, C=0, \ldots, I=0) \\
= \; & P(W=w|F=1, C=0) \\
\propto \; & P(F=1|W=w)*P(C=0|W=w)*P(W=w) \\
= \; & \begin{cases} 0.05*0.88*0.40, & W=0 \\ 0.20*0.70*0.60, & W=1 \end{cases}
\end{aligned}
$$

[1]Reference: Percy Liang, CS221 (2015)

Gibbs Sampling Example I



w	p(w)
0	0.40
1	0.60

w	f	p(f|w)
0	0	0.95
0	1	0.05
1	0	0.80
1	1	0.20

w	c	p(f|w)
0	0	0.88
0	1	0.12
1	0	0.70
1	1	0.30

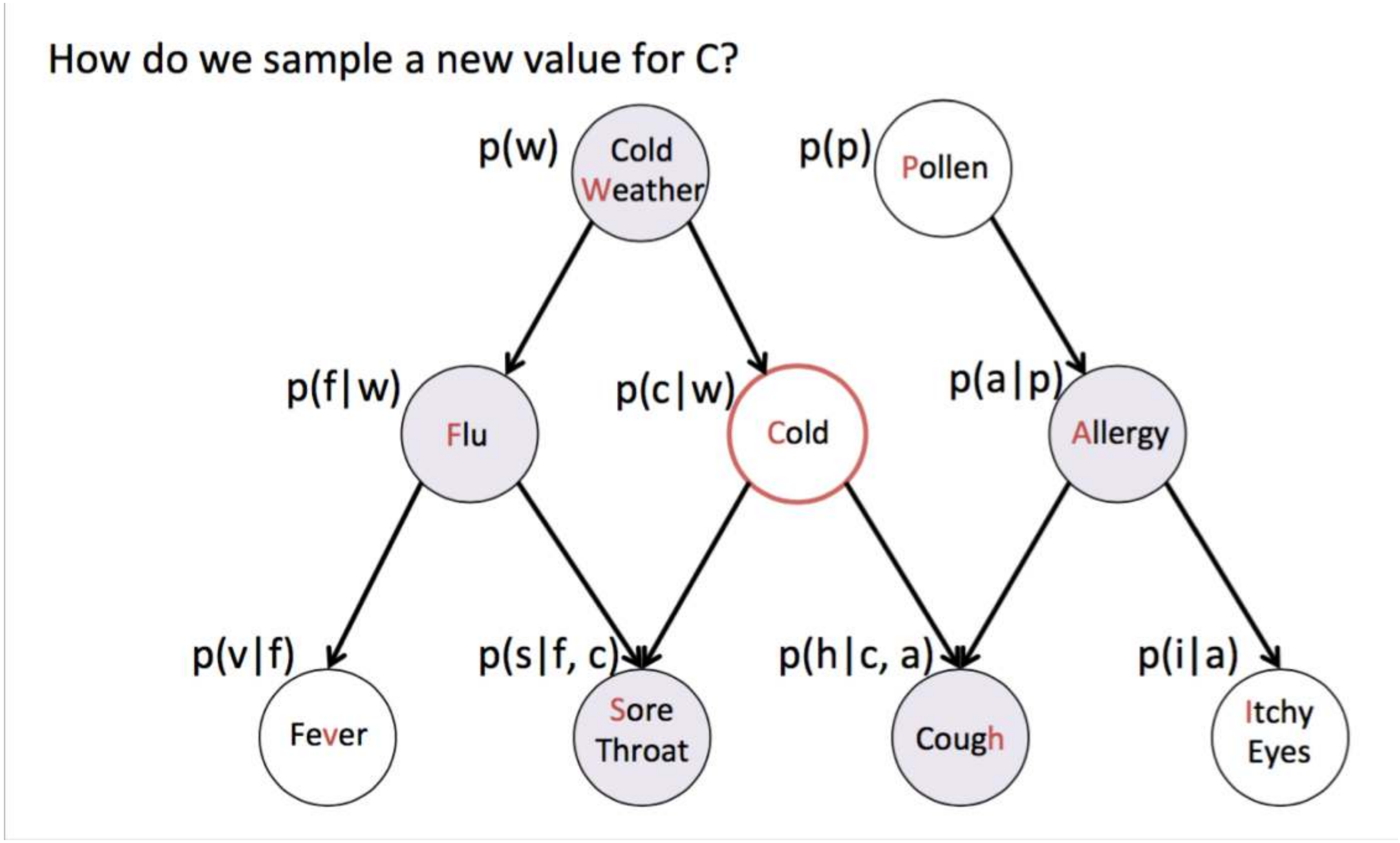
$$P(W{=}w \mid F{=}1, P{=}1, C{=}0, \ldots, I{=}0)$$
$$= P(W{=}w \mid F{=}1, C{=}0)$$
$$\propto P(F{=}1 \mid W{=}w) * P(C{=}0 \mid W{=}w) * P(W{=}w)$$
$$= \begin{cases} 0.05 * 0.88 * 0.40, & W = 0 \\ 0.20 * 0.70 * 0.60, & W = 1 \end{cases}$$

$$P(W = w \mid F = 1, C = 0)$$
$$= \begin{cases} 0.0176/(0.0176+0.084), & w = 0 \\ 0.084/(0.0176+0.084), & w = 1 \end{cases}$$
$$= \begin{cases} 0.173, & w = 0 \\ 0.827, & w = 1 \end{cases}$$

Sample a new w!

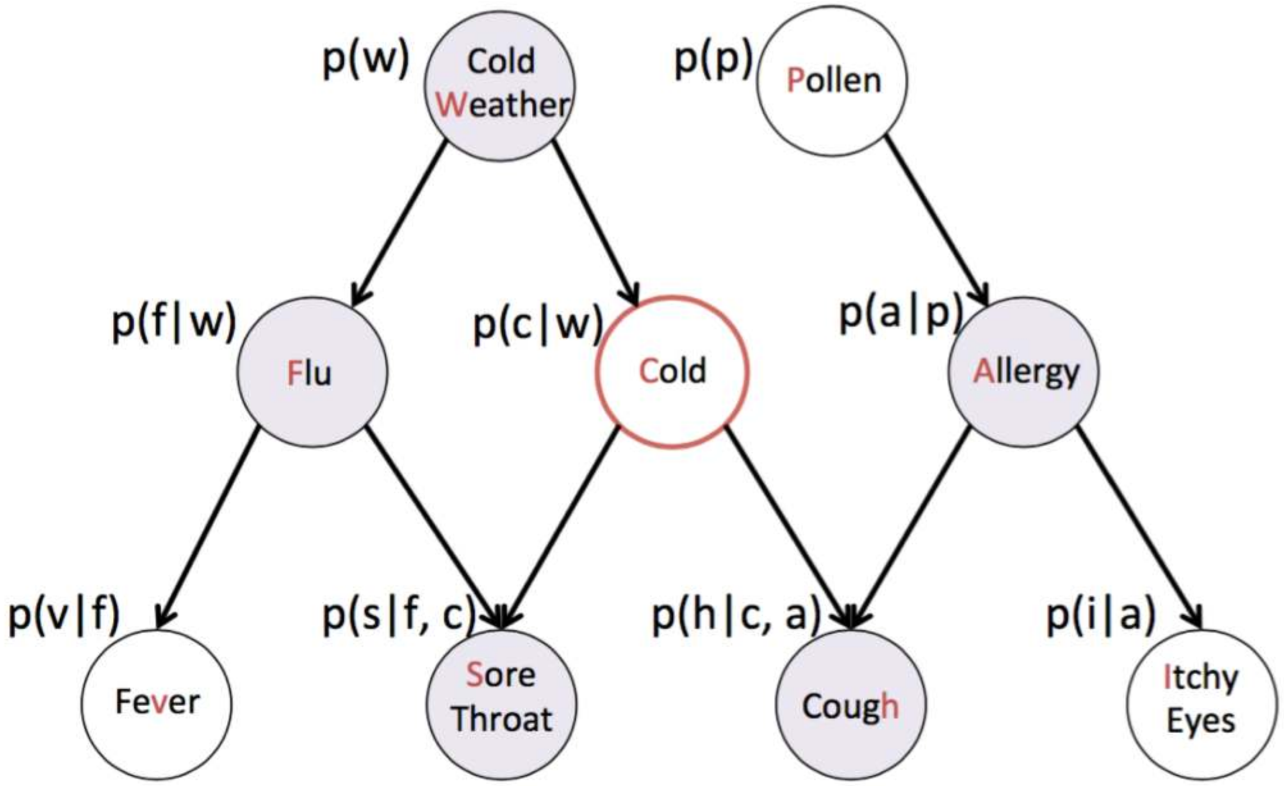
[1] Reference: Percy Liang, CS221 (2015)

Gibbs Sampling Example II



[1]Reference: Percy Liang, CS221 (2015)

Gibbs Sampling Example II

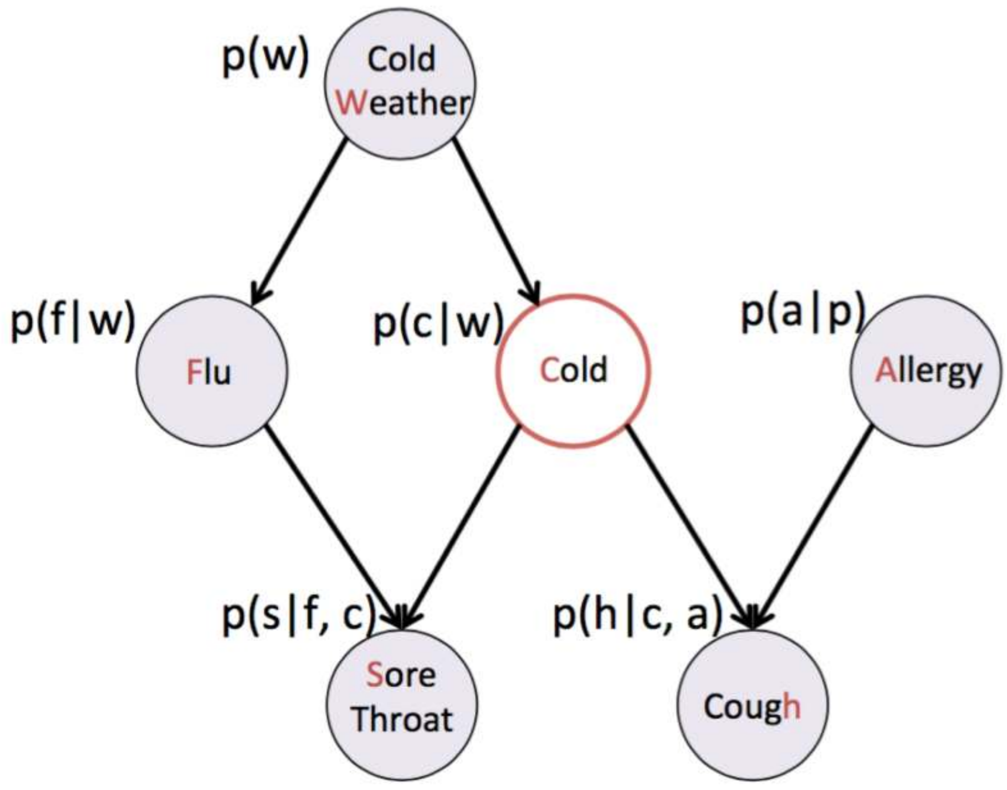


$$P(C{=}c \mid W{=}1, F{=}1, P{=}1, \ldots, I{=}0)$$

$$= P(C{=}c \mid W{=}1, F{=}1, S{=}0, H{=}1, A{=}1)$$ Markov Blanket!

[1]Reference: Percy Liang, CS221 (2015)

Gibbs Sampling Example II

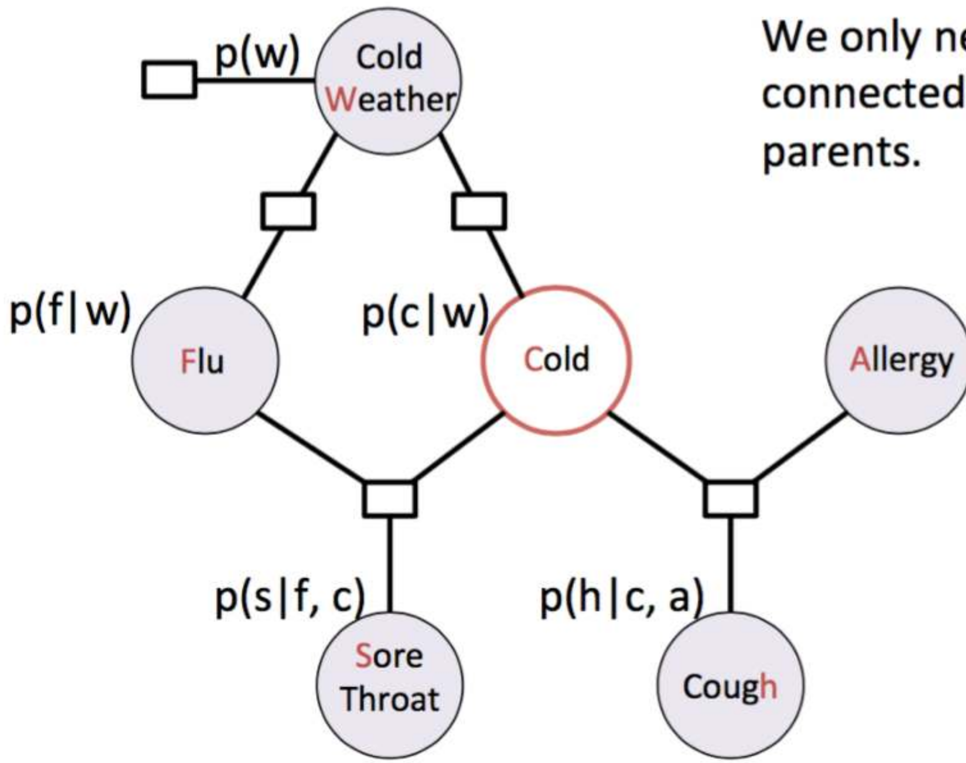


$$P(C=c \mid W=1, F=1, P=1, \ldots, I=0)$$

$$= P(C=c \mid W=1, F=1, S=0, H=1, A=1)$$

[1] Reference: Percy Liang, CS221 (2015)

Gibbs Sampling Example II



We only need look at the factors connected to the children and the parents.

$$P(C{=}c \mid W{=}1, F{=}1, P{=}1, \ldots, I{=}0$$

$$= P(C{=}c \mid W{=}1, F{=}1, S{=}0, H{=}1, A{=}1)$$

$$= p(w)\, p(f \mid w)\, p(c \mid w)\, p(s \mid f, c)\, p(h \mid c, a)$$

[1]Reference: Percy Liang, CS221 (2015)

Gibbs Sampling: Conditional Probability

Update x_i

$$p(x_i \mid x_{\setminus i}) = \frac{1}{Z} p(x_i \mid pa(x_i)) \prod_{j \in \mathsf{ch}(i)} p(x_j \mid \mathsf{pa}(x_j))$$

and the normalisation constant is

$$Z = \sum_{x_i} p(x_i \mid pa(x_i)) \prod_{j \in \mathsf{ch}(i)} p(x_j \mid \mathsf{pa}(x_j))$$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Gibbs_sampling

Understanding MCMC via Markov Chain Terminology

Sample from a multi-variate distribution

$$p(x) = \frac{1}{Z} p^*(x)$$

with Z intractable to calculate

Idea: Sample from some $q(\mathbf{x} \to \mathbf{x}')$ with a stationary distribution

$$\pi(\mathbf{x}') = \Sigma_{\mathbf{x}} \pi(\mathbf{x}) q(\mathbf{x} \to \mathbf{x}') \qquad \text{for all } \mathbf{x}'$$

Given $p(x)$ find $q(\mathbf{x} \to \mathbf{x}')$ such that $\pi(\mathbf{x}) = p(x)$

Gibbs sampling is one instance (that is why it is working)

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm

Understanding MCMC via Markov Chain Terminology

Transition probability $q(\mathbf{x} \rightarrow \mathbf{x}')$

Occupancy probability $\pi_t(\mathbf{x})$ at time t

Equilibrium condition on π_t defines stationary distribution $\pi(\mathbf{x})$

Note: stationary distribution depends on choice of $q(\mathbf{x} \rightarrow \mathbf{x}')$

Pairwise detailed balance on states guarantees equilibrium

Gibbs sampling transition probability:

sample each variable given current values of all others

$\Rightarrow$ detailed balance with the true posterior

For Bayesian networks, Gibbs sampling reduces to sampling conditioned on each variable's Markov blanket

[1]Reference: Pedro Domingos, CSE 515 (2017)

Stationary Distribution of a MC

$\pi_t(\mathbf{x})$ = probability in state $\mathbf{x}$ at time t
$\pi_{t+1}(\mathbf{x}')$ = probability in state $\mathbf{x}'$ at time $t+1$

π_{t+1} in terms of π_t and $q(\mathbf{x} \rightarrow \mathbf{x}')$

$$\pi_{t+1}(\mathbf{x}') = \Sigma_{\mathbf{x}} \pi_t(\mathbf{x}) q(\mathbf{x} \rightarrow \mathbf{x}')$$

Stationary distribution: $\pi_t = \pi_{t+1} = \pi$

$$\pi(\mathbf{x}') = \Sigma_{\mathbf{x}} \pi(\mathbf{x}) q(\mathbf{x} \rightarrow \mathbf{x}') \qquad \text{for all } \mathbf{x}'$$

If π exists, it is unique (specific to $q(\mathbf{x} \rightarrow \mathbf{x}')$)

In equilibrium, expected "outflow" = expected "inflow"

[1]Reference: Pedro Domingos, CSE 515 (2017)

Detailed Balance Equation

"Outflow" = "inflow" for each pair of states:

$$\pi(\mathbf{x})q(\mathbf{x} \to \mathbf{x}') = \pi(\mathbf{x}')q(\mathbf{x}' \to \mathbf{x}) \qquad \text{for all } \mathbf{x},\ \mathbf{x}'$$

Detailed balance $\Rightarrow$ stationarity:

$$\begin{aligned} \Sigma_{\mathbf{x}}\pi(\mathbf{x})q(\mathbf{x} \to \mathbf{x}') &= \Sigma_{\mathbf{x}}\pi(\mathbf{x}')q(\mathbf{x}' \to \mathbf{x}) \\ &= \pi(\mathbf{x}')\Sigma_{\mathbf{x}}q(\mathbf{x}' \to \mathbf{x}) \\ &= \pi(\mathbf{x}') \end{aligned}$$

MCMC algorithms typically constructed by designing a transition probability q that is in detailed balance with desired π

[1]Reference: Pedro Domingos, CSE 515 (2017)

Gibbs Satisfies Detailed Balance

Sample each variable in turn, given **all other variables**

Sampling X_i, let $\bar{\mathbf{X}}_i$ be all other nonevidence variables
Current values are x_i and $\bar{\mathbf{x}}_i$; $\mathbf{e}$ is fixed
Transition probability is given by

$$q(\mathbf{x} \rightarrow \mathbf{x}') = q(x_i, \bar{\mathbf{x}}_i \rightarrow x_i', \bar{\mathbf{x}}_i) = P(x_i'|\bar{\mathbf{x}}_i, \mathbf{e})$$

This gives detailed balance with true posterior $P(\mathbf{x}|\mathbf{e})$:

$$\begin{aligned}
\pi(\mathbf{x})q(\mathbf{x} \rightarrow \mathbf{x}') &= P(\mathbf{x}|\mathbf{e})P(x_i'|\bar{\mathbf{x}}_i, \mathbf{e}) = P(x_i, \bar{\mathbf{x}}_i|\mathbf{e})P(x_i'|\bar{\mathbf{x}}_i, \mathbf{e}) \\
&= P(x_i|\bar{\mathbf{x}}_i, \mathbf{e})P(\bar{\mathbf{x}}_i|\mathbf{e})P(x_i'|\bar{\mathbf{x}}_i, \mathbf{e}) \quad \text{(chain rule)} \\
&= P(x_i|\bar{\mathbf{x}}_i, \mathbf{e})P(x_i', \bar{\mathbf{x}}_i|\mathbf{e}) \quad \text{(chain rule backwards)} \\
&= q(\mathbf{x}' \rightarrow \mathbf{x})\pi(\mathbf{x}') = \pi(\mathbf{x}')q(\mathbf{x}' \rightarrow \mathbf{x})
\end{aligned}$$

[1] Reference: Pedro Domingos, CSE 515 (2017)

Gibbs Sampling: Performance

Think of Gibbs sampling as

$$x^{l+1} \sim q(\cdot \mid x^l)$$

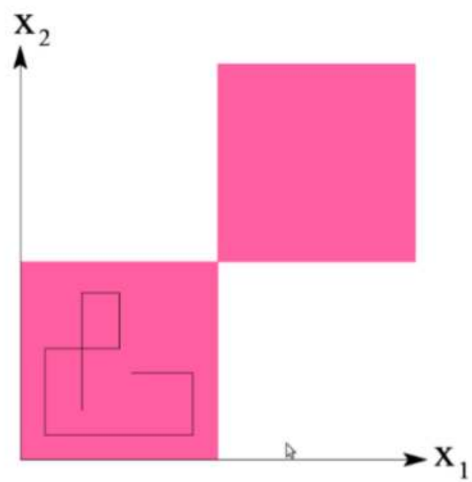
Problem: States are highly dependent $(x^1, x^2, \ldots)$

Need a long time to run Gibbs sampling to *forget* the initial state, this is called burn in phase

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Gibbs_sampling

Gibbs Sampling: Performance



In this example the samples stay in the lower left quadrant

Some technical requirements to Gibbs sampling

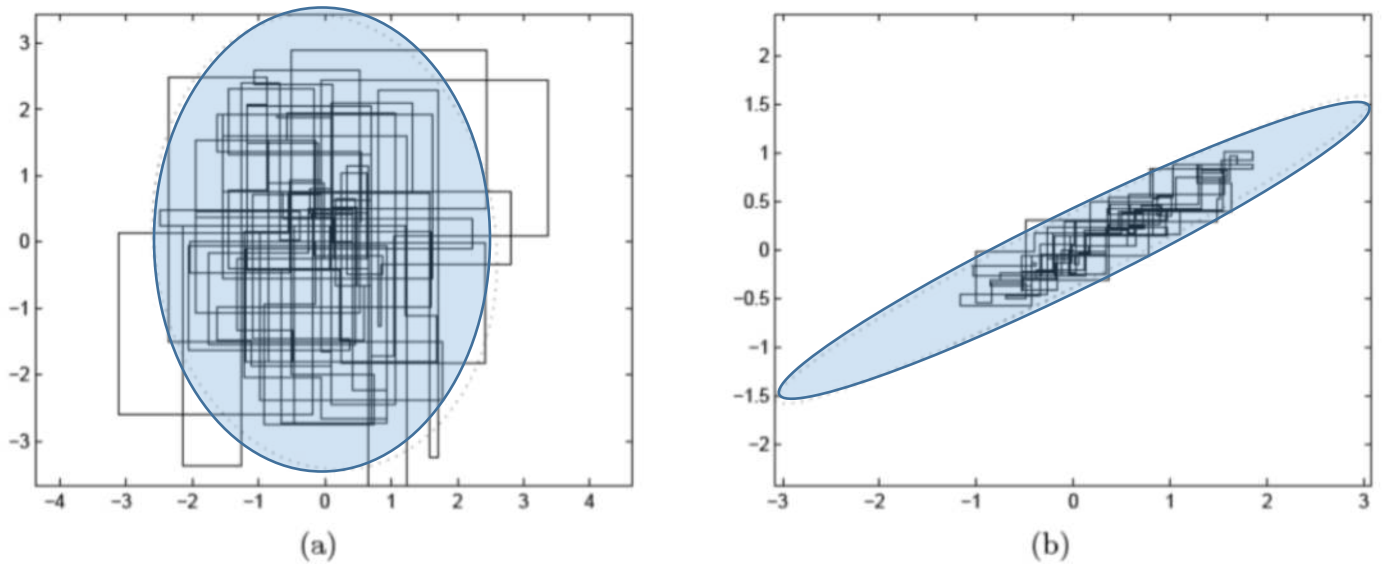
The Markov Chain $q(x^{l+1} \mid x^l)$ needs to be able to traverse the entire state-space (no matter where we start)

- This property is called irreducible
- Then $p(x)$ is the stationary distribution of $q(x' \mid x)$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Gibbs_sampling

Gibbs Sampling: Performance



Gibbs sampling is more efficient if states are not correlated

- Left: Almost isotropic Gaussian
- Right: correlated Gaussian

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Gibbs_sampling

Metropolis-Hasting MCMC

- We will now mention one other MCMC method in passing.
  - Metropolis-Hasting (MH)
    - A special case is called Metropolis sampling.

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm

MH MCMC Algorithm

Slightly more general MCMC method when the proposal distribution is *not* symmetric

Sample x' and accept with probability

$$A(x', x) = \min\left(1, \frac{\tilde{q}(x \mid x')p^*(x')}{\tilde{q}(x' \mid x)p^*(x)}\right)$$

Note: when the proposal distribution is symmetric, Metropolis-Hastings reduces to standard Metropolis sampling

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm

MH MCMC Special Case: Metropolis Sampling

Special case of MCMC method (proposal distribution) with the following proposal distribution

- symmetric: $q(x' \mid x) = q(x \mid x')$

Sample x' and accept with probability

$$A(x', x) = \min\left(1, \frac{p^*(x')}{p^*(x)}\right) \in [0, 1]$$

- If new state x' is more probable always accept
- If new state is less probable accept with $\frac{p^*(x')}{p^*(x)}$

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

[2]Reference: https://en.wikipedia.org/wiki/Metropolis%E2%80%93Hastings_algorithm

Questions?

Summary

- Inference computations on joint distributions is a hard problem

- Graphical models help us do this in efficient ways
  - For tree models, this is linear time!

- We discussed two exact methods
  - Variable Elimination
  - Belief propagation

- We discussed one family of approximate methods
  - Based on sampling via Markov Chain Monte Carlo techniques

Appendix

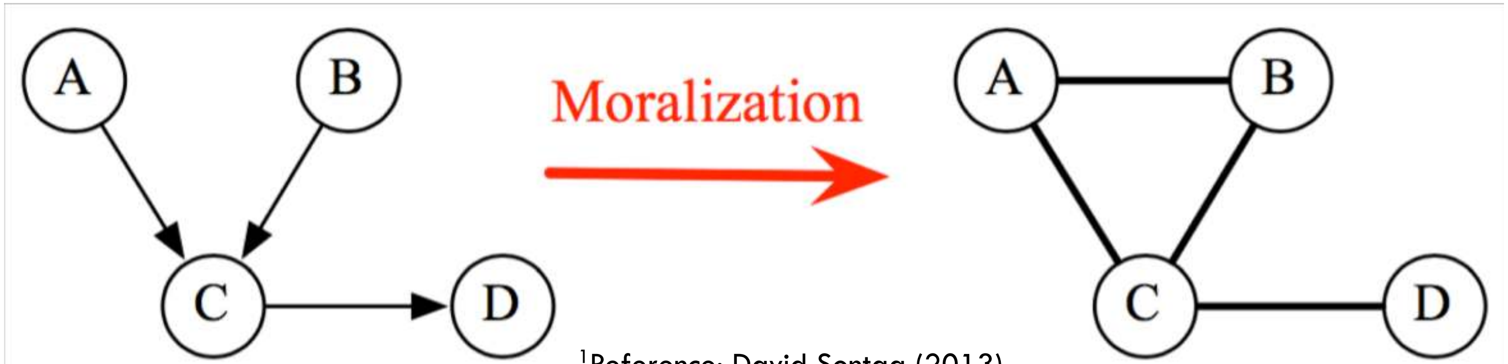
Sample Exam Questions

- What is a factor graph? How is it related to DPGMs? How is it related to UPGMs?
- What are the key steps of Belief propagation?
- What is the use of BP? Can it be used for inference over general factor graphs?
- How would one use sampling for inference?
- Why is Gibbs sampling a MCMC technique?
- Why does BP do better than variable elimination?

DPGMs and UPGMs

- Inference algorithms can typically run on both graphs
- For convenience, we will construct a UPGM from a DPGM and discuss inference on UPGM

- The construction is straightforward
  - For each factor in DPGM, call it a potential now
  - Moralize the DPGM and remove directions
    - (We lose some information in the graph)



[1]Reference: David Sontag (2013)

BP: Computing Maximal State

- BP variant can also solve for the maximal state $\bar{x}^* \in \operatorname{argmax}_{\bar{x}\in\mathfrak{X}} P(\bar{x})$
- This version is called Max-Product Belief Propagation

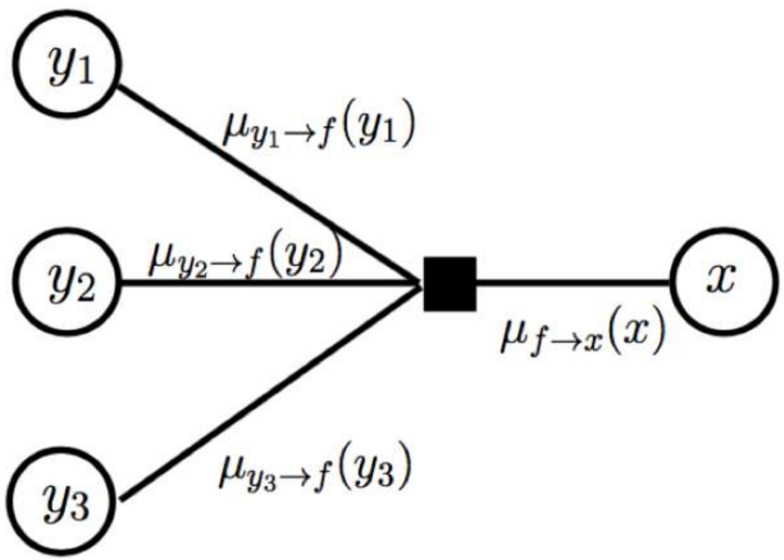
- Has three ingredients just as before
  - Initialization (same as before)
  - Variable to factor message (same as before)
  - Factor to variable message

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP: Computing Maximal State

- Factor to variable message is different from Sum-Product

$$\mu_{f\to x}(x) = \max_{y\in\mathcal{X}_f\setminus x} \phi_f(\mathcal{X}_f) \prod_{y\in\{\mathrm{ne}(f)\setminus x\}} \mu_{y\to f}(y)$$



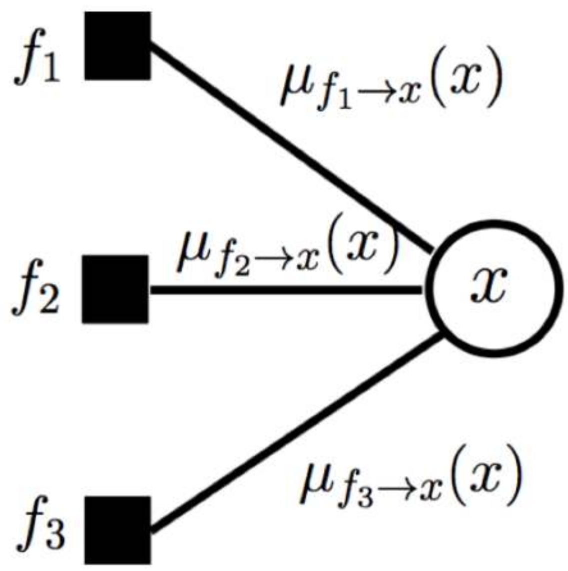
- Additionally, we need to track values achieving maximums as well

[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)

BP: Computing Maximal State

- Maximal state of a variable is

$$x^* = \underset{x}{\operatorname{argmax}} \prod_{f \in \mathsf{ne}(x)} \mu_{f \to x}(x)$$



[1]Reference: Bjoern Andres and Bernt Schiele, MPI (2016)