CS 3750 Machine Learning
Lecture 3

Graphical models: inference

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Factors

- **Factor:** is a function that maps value assignments for a subset of random variables to $\Re$ (reals)
- **The scope of the factor:**
  - a set of variables defining the factor
- **Example:**
  - Assume discrete random variables x (with values a1,a2, a3) and y (with values b1 and b2)
  - Factor:

$$\phi(x, y) \rightarrow$$

a1	b1	0.5
a1	b2	0.2
a2	b1	0.1
a2	b2	0.3
a3	b1	0.2
a3	b2	0.4

  - Scope of the factor:

$$\{x, y\}$$

Factor Product

Variables: A,B,C

$\phi(A,B,C) = \phi(B,C) \circ \phi(A,B)$

$\phi(B,C)$

b1	c1	0.1
b1	c2	0.6
b2	c1	0.3
b2	c2	0.4

$\phi(A,B)$

a1	b1	0.5
a1	b2	0.2
a2	b1	0.1
a2	b2	0.3
a3	b1	0.2
a3	b2	0.4

$\phi(A,B,C)$

a1	b1	c1	0.5*0.1
a1	b1	c2	0.5*0.6
a1	b2	c1	0.2*0.3
a1	b2	c2	0.2*0.4
a2	b1	c1	0.1*0.1
a2	b1	c2	0.1*0.6
a2	b2	c1	0.3*0.3
a2	b2	c2	0.3*0.4
a3	b1	c1	0.2*0.1
a3	b1	c2	0.2*0.6
a3	b2	c1	0.4*0.3
a3	b2	c2	0.4*0.4

Factor Marginalization

Variables: A,B,C

$\phi(A,C) = \sum_B \phi(A,B,C)$

a1	b1	c1	0.2
a1	b1	c2	0.35
a1	b2	c1	0.4
a1	b2	c2	0.15
a2	b1	c1	0.5
a2	b1	c2	0.1
a2	b2	c1	0.3
a2	b2	c2	0.2
a3	b1	c1	0.25
a3	b1	c2	0.45
a3	b2	c1	0.15
a3	b2	c2	0.25

a1	c1	0.2+0.4=0.6
a1	c2	0.35+0.15=0.5
a2	c1	0.8
a2	c2	0.3
a3	c1	0.4
a3	c2	0.7

Factor division

A=1	B=1	0.5
A=1	B=2	0.4
A=2	B=1	0.8
A=2	B=2	0.2
A=3	B=1	0.6
A=3	B=2	0.5

A=1	0.4
A=2	0.4
A=3	0.5

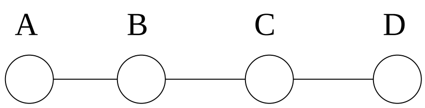
A=1	B=1	0.5/0.4=1.25
A=1	B=2	0.4/0.4=1.0
A=2	B=1	0.8/0.4=2.0
A=2	B=2	0.2/0.4=2.0
A=3	B=1	0.6/0.5=1.2
A=3	B=2	0.5/0.5=1.0

Inverse of a factor product

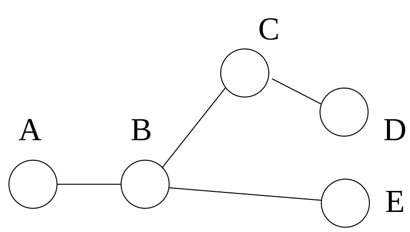
Inferences

We have already seen VE inferences on both BBNs and MRFs
- Inference on chains and trees structures can be often done efficiently in time: **linear in the number of nodes in the tree**

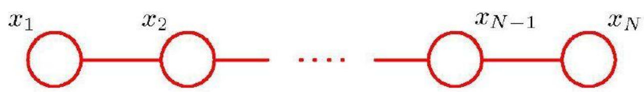
chain



tree



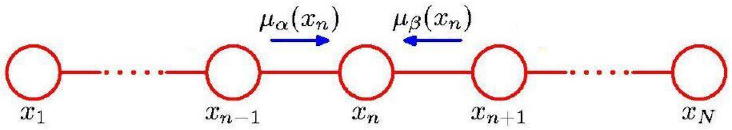
Inference on a Chain



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

$$p(x_n) = \sum_{x_1}\cdots\sum_{x_{n-1}}\sum_{x_{n+1}}\cdots\sum_{x_N} p(\mathbf{x})$$

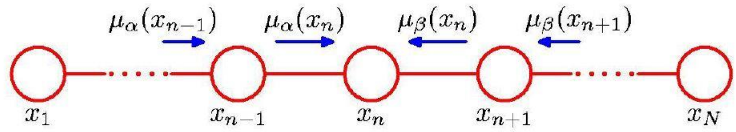
Inference on a Chain



$$p(x_n) = \frac{1}{Z}\underbrace{\left[\sum_{x_{n-1}}\psi_{n-1,n}(x_{n-1}, x_n)\cdots\left[\sum_{x_1}\psi_{1,2}(x_1, x_2)\right]\cdots\right]}_{\mu_\alpha(x_n)}$$

$$\underbrace{\left[\sum_{x_{n+1}}\psi_{n,n+1}(x_n, x_{n+1})\cdots\left[\sum_{x_N}\psi_{N-1,N}(x_{N-1}, x_N)\right]\cdots\right]}_{\mu_\beta(x_n)}$$

Inference on a Chain

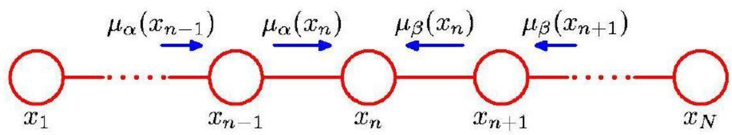


$$\begin{aligned}\mu_\alpha(x_n) &= \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \left[\sum_{x_{n-2}} \cdots\right] \\ &= \sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n)\mu_\alpha(x_{n-1}).\end{aligned}$$

$$\begin{aligned}\mu_\beta(x_n) &= \sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \left[\sum_{x_{n+2}} \cdots\right] \\ &= \sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1})\mu_\beta(x_{n+1}).\end{aligned}$$

Slides by C. Bishop

Inference on a Chain



$$\mu_\alpha(x_2) = \sum_{x_1} \psi_{1,2}(x_1, x_2) \qquad \mu_\beta(x_{N-1}) = \sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)$$

$$Z = \sum_{x_n} \mu_\alpha(x_n)\mu_\beta(x_n)$$

Slides by C. Bishop

Inference on a Chain

To compute local marginals:

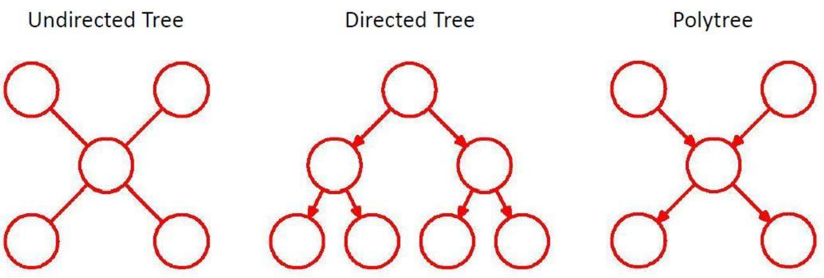
- Compute and store all forward messages, $\mu_\alpha(x_n)$.
- Compute and store all backward messages, $\mu_\beta(x_n)$.
- Compute Z at any node x_m
- Compute

$$p(x_n) = \frac{1}{Z}\mu_\alpha(x_n)\mu_\beta(x_n)$$

for all variables required.

Slides by C. Bishop

Trees



Slides by C. Bishop

Factor graph

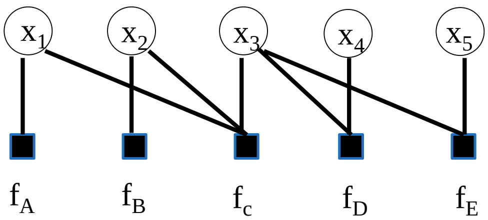
A graphical representation that lets us express a factorization of a function over a set of variables

A factor graph is bipartite graph where:

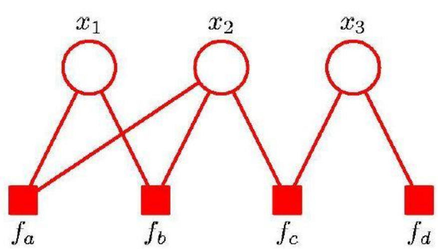
- One layer is formed by variables
- Another layer is formed by factors or functions on subsets of variables

Example: a function over variables $x_1, x_2, \ldots x_5$

$$g(x_1, x_2, \ldots x_5) = f_A(x_1)\, f_B(x_2)\, f_C(x_1, x_2, x_3)\, f_D(x_3, x_4)\, f_E(x_3, x_5)$$



Factor Graphs



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

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

Inferences on factor graphs

- Efficient inference algorithms for factor graphs built for trees [Frey, 1998; Kschischnang *et al.*, 2001] **:**
  - **Sum-product algorithm**
  - **Max product algorithm**

The Sum-Product Algorithm (1)

Objective:

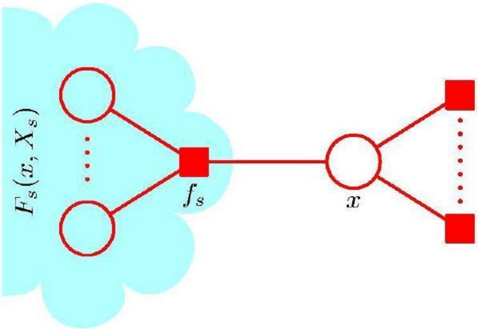
i. to obtain an efficient, exact inference algorithm for finding marginals;

ii. in situations where several marginals are required, to allow computations to be shared efficiently.

Key idea: Distributive Law

$$ab + ac = a(b + c)$$

The Sum-Product Algorithm (2)

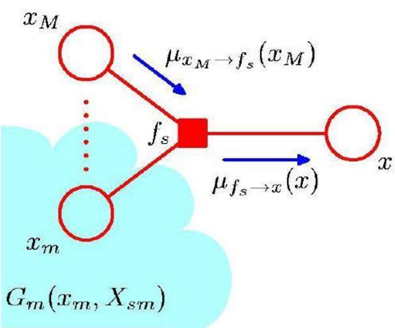


$$p(x) = \sum_{\mathbf{x}\setminus x} p(\mathbf{x})$$

$$p(\mathbf{x}) = \prod_{s\in\mathrm{ne}(x)} F_s(x, X_s)$$

Slides by C. Bishop

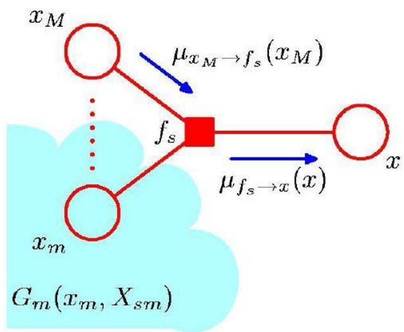
The Sum-Product Algorithm (4)



$$F_s(x, X_s) = f_s(x, x_1, \ldots, x_M) G_1(x_1, X_{s1}) \ldots G_M(x_M, X_{sM})$$

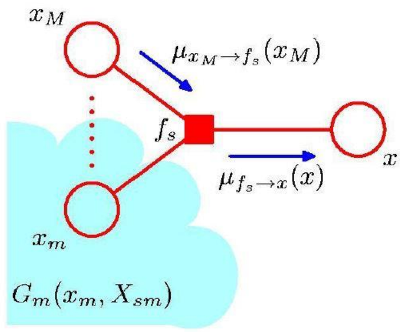
Slides by C. Bishop

The Sum-Product Algorithm (5)



$$\begin{aligned}
\mu_{f_s \to x}(x) &= \sum_{x_1} \ldots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \left[\sum_{X_{sm}} G_m(x_m, X_{sm}) \right] \\
&= \sum_{x_1} \ldots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x_m)
\end{aligned}$$

The Sum-Product Algorithm (6)



$$\begin{aligned}
\mu_{x_m \to f_s}(x_m) \equiv \sum_{X_{sm}} G_m(x_m, X_{sm}) &= \sum_{X_{sm}} \prod_{l \in \mathrm{ne}(x_m) \setminus f_s} F_l(x_m, X_{ml}) \\
&= \prod_{l \in \mathrm{ne}(x_m) \setminus f_s} \mu_{f_l \to x_m}(x_m)
\end{aligned}$$

The Sum-Product Algorithm (7)

Initialization

Slides by C. Bishop

The Sum-Product Algorithm (8)

To compute local marginals:

- Pick an arbitrary node as root
- Compute and propagate messages from the leaf nodes to the root, storing received messages at every node.
- Compute and propagate messages from the root to the leaf nodes, storing received messages at every node.
- Compute the product of received messages at each node for which the marginal is required, and normalize if necessary.

Slides by C. Bishop