

第18章： 概率图模型

Probabilistic Graphical Model

刘智勇（zhiyong.liu@ia.ac.cn）

中国科学院自动化研究所

上次内容简要回顾

- 特征提取
 - PCA
 - ICA
 - LDA
- 特征选择
 - Filter
 - Wrapper

提纲

- 概述
- 有向概率图模型
- 无向概率图模型
- 模型的推理

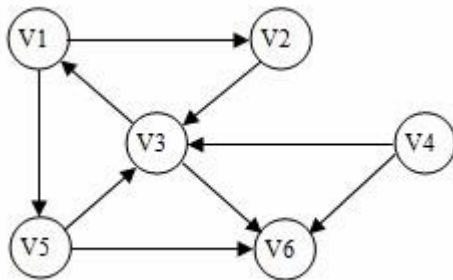
参考资料:

1. Bishop, C., Pattern Recognition and Machine Learning, Chapter 8
2. Murphy, K., An introduction to graphical models
3. Jordan M.I., An introduction to probabilistic graphical models

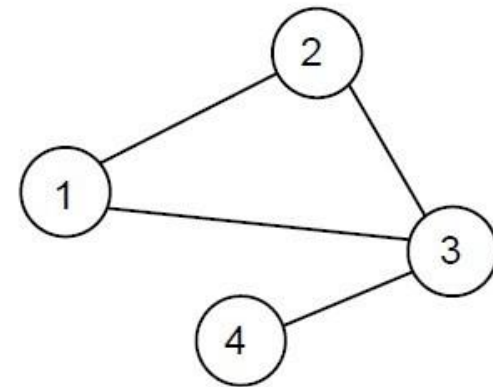
概述

图 (graph)

- A graph comprises *nodes* (also called *vertices*) connected by *links* (also known as *edges* or *arcs*)
- Mathematically, a graph is denoted by $G = (V, E)$ where V is the set of nodes, and $E \subseteq V \times V$ denotes the set of edges
- 图一般分为有向图和无向图



$$V = \{V1, \dots, V6\}, E = \{\{1,2\}, \{1,5\}, \dots, \{5,6\}\}$$



$$V = \{1,2,3,4\}, E = \{\{1,2\}, \{1,3\}, \dots, \{3,4\}\}$$

概述

概率图（Probabilistic Graph）

- 概率图 = 概率+图
- 提供了处理两类问题的合适手段：不确定性+复杂性（uncertainty and complexity）
 - 图提供了模块：简单模块可以组合成复杂模型
 - 概率则保证各部分可以协调一致工作
 - 提供了一个方便直观的手段来描述随机变量间的 dependency 关系
- 很多模型可以看成是通用概率图模型的一个特例，包括 Mixture Model, Factor Analysis, Hidden Markov Model, Kalman Filter, Markov Random Fields

概述

概率图（Probabilistic Graph）的表示（Representation）

- 图的节点表示随机变量
- 图的边（或者说变量之间没有边）意味着变量间的条件独立
- 为变量间的联合概率分布（joint probability distribution）提供了一个紧致（compact）的描述
 - 例如，表述 N 个二值变量的联合概率分布需要 $O(2^N)$ 个参数，如果利用概率图的条件独立性把这些变量分解（factorization），参数的数量将指数级减少
- 对应于图模型，概率图模型也分为有向概率图和无向概率图
 - 有向概率图（directed graphical models），又称为贝叶斯网（Bayesian Networks），belief networks, generative models等，利用贝叶斯规则推理，在机器学习和AI等领域较多使用
 - 无向概率图（undirected graphical models）：马尔科夫随机场（Markov random field），在计算机视觉领域较多使用

有向概率图

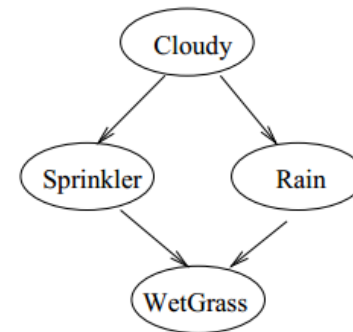
- 在变量（节点） A 和 B 之间存在 $A \rightarrow B$ 的边，称 A 为 B 的父节点， B 为 A 的子节点。可以近似理解为 A “导致”了 B ，因此不能有存在“循环”路径

一个简单的例子

- 每个节点代表一个二值变量
- “grass is wet”这个事件和“sprinkler”还有“rain”这两个事情相关，最下方表格它们之间的强度联系，这个表格称为“条件概率表格”（conditional probability table: CPT）；类似的还有“sprinkler”和“rain”这两个事件
- “cloudy”这个事件没有父节点，因此它的CPT被称为先验概率（prior probability）

	P(C=F)	P(C=T)
	0.5	0.5

C	P(S=F)	P(S=T)
F	0.5	0.5
T	0.9	0.1



C	P(R=F)	P(R=T)
F	0.8	0.2
T	0.2	0.8

S	R	P(W=F)	P(W=T)
F	F	1.0	0.0
T	F	0.1	0.9
F	T	0.1	0.9
T	T	0.01	0.99

有向概率图

- 贝叶斯网中的条件独立关系可以描述如下：给定一个贝叶斯网，给定一个节点 A 的父（parent）节点集合（ $S(A)$ ）， A 独立于除 $S(A)$ 外其它的祖先（ancestor）节点
- 利用上述原理来简化前面的例子
 - 利用chain rule
$$P(C, S, R, W) = P(C)P(S|C)P(R|C, S)P(W|C, S, R)$$
 - 利用上述条件独立关系
$$P(C, S, R, W) = P(C)P(S|C)P(R|C)P(W|S, R)$$
 - 给定 C ， R 独立于 S ，记为 $R \perp S|C$ （有两竖），同样，有 $W \perp C|S, R$
 - 一般来讲，利用条件独立关系可以更为紧致地描述联合概率分布， $O(2^N)$ v.s. $O(n2^k)$ ，这里 k 表示图中所有节点具有的最多父节点数目
- **Factorization: 概率图最核心的性质！**
 - 给定一个 K 个节点的有向概率图，它的联合概率分布可以表示为

$$p(x) = \prod_{k=1}^K p(x_k | pa_k)$$

pa_k 表示 x_k 的父节点集合

有向概率图

条件独立 (Conditional Independence)

- a is independent of b given c

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

- Equivalently

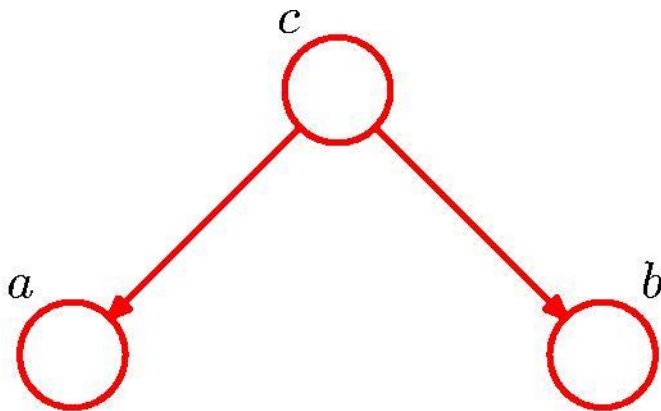
$$\begin{aligned} p(a, b|c) &= p(a|b, c)p(b|c) \\ &= p(a|c)p(b|c) \end{aligned}$$

- Notation

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

有向概率图

- 条件独立：例子1



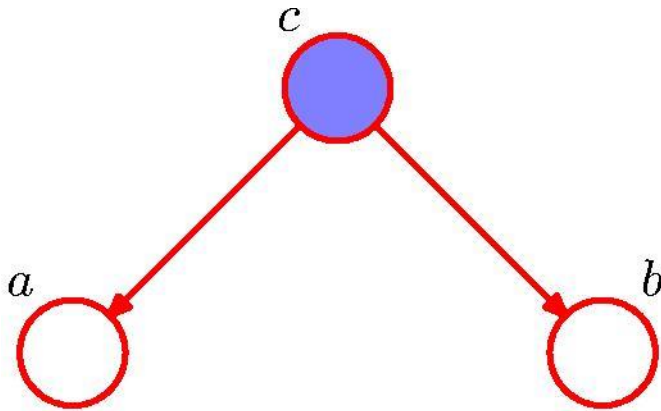
$$p(a, b, c) = p(a|c)p(b|c)p(c)$$

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

$$a \not\perp b \mid \emptyset$$

有向概率图

- 条件独立：例子1

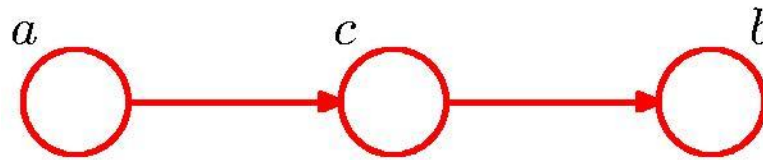


$$\begin{aligned} p(a, b|c) &= \frac{p(a, b, c)}{p(c)} \\ &= p(a|c)p(b|c) \end{aligned}$$

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

有向概率图

- 条件独立：例子2



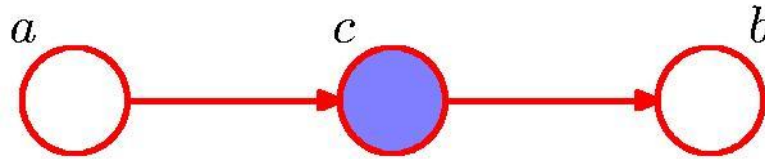
$$p(a, b, c) = p(a)p(c|a)p(b|c)$$

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

$$a \not\perp b \mid \emptyset$$

有向概率图

- 条件独立：例子2

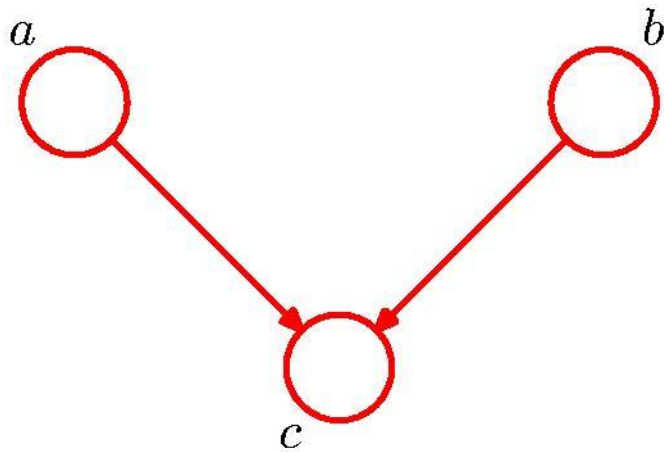


$$\begin{aligned} p(a, b|c) &= \frac{p(a, b, c)}{p(c)} \\ &= \frac{p(a)p(c|a)p(b|c)}{p(c)} \\ &= p(a|c)p(b|c) \end{aligned}$$

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

有向概率图

- 条件独立：例子3



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

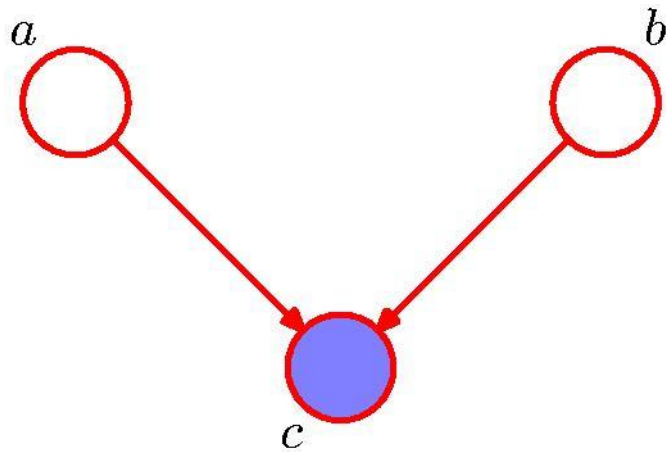
$$p(a, b) = p(a)p(b)$$

$$a \perp\!\!\!\perp b \mid \emptyset$$

Note: this is the opposite of Example 1, with c unobserved.

有向概率图

- 条件独立：例子3



$$\begin{aligned} p(a, b|c) &= \frac{p(a, b, c)}{p(c)} \\ &= \frac{p(a)p(b)p(c|a, b)}{p(c)} \end{aligned}$$

$$a \not\perp b \mid c$$

Note: this is the opposite of Example 1, with c observed.

有向概率图

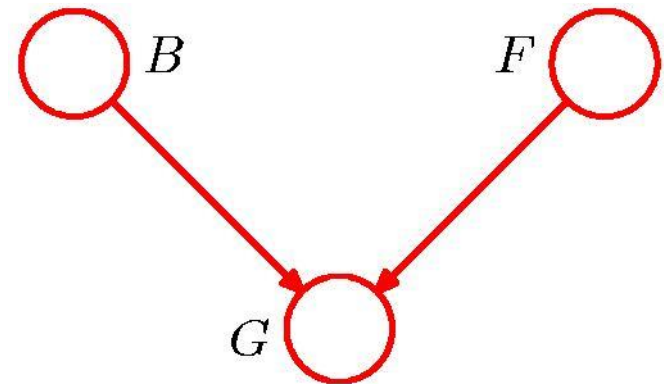
例子3，一个具体的案例

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

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

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

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



$$p(B = 1) = 0.9$$

$$p(F = 1) = 0.9$$

and hence

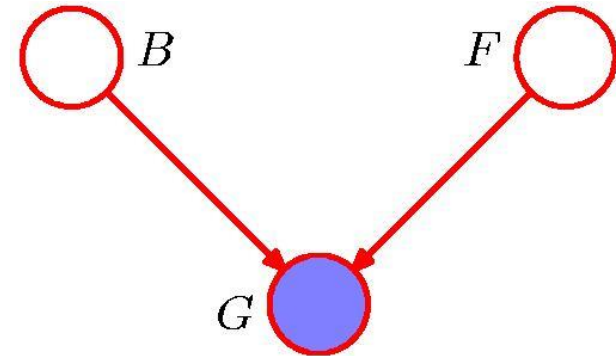
$$p(F = 0) = 0.1$$

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

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

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

有向概率图



例子3，一个具体的案例

- 当观察到 $G = 0$ 时

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

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

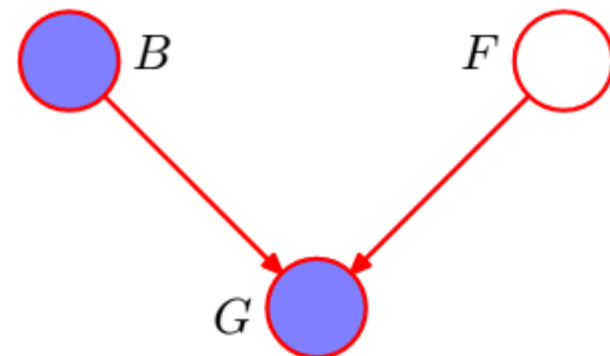
$$= \sum_{B=\{0,1\}} \sum_{F=\{0,1\}} p(G = 0|B, F)p(B)p(F) = 0.315$$

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

$$p(F = 0|G = 0) = \frac{0.81 * 0.1}{0.315} \approx 0.257 > p(F = 0)!!$$

Probability of an empty tank increased by observing $G = 0$.

有向概率图



例子3，一个具体的案例

- 假如我们同时又观察到 $B = 0$

$$\begin{aligned} p(F = 0 | G = 0, B = 0) &= \frac{p(F = 0, G = 0, B = 0)}{p(G = 0 | B = 0)p(B = 0)} \\ &= \frac{p(G = 0 | B = 0, F = 0)p(F = 0)p(B = 0)}{\sum_{F=\{0,1\}} p(G = 0 | B = 0, F)p(F)p(B = 0)} \\ &= \frac{0.9 * 0.1}{0.9 * 0.1 + 0.8 * 0.9} \approx 0.111 \quad \begin{cases} > p(F = 0) \\ < p(F = 0 | G = 0) \end{cases} \end{aligned}$$

有向概率图

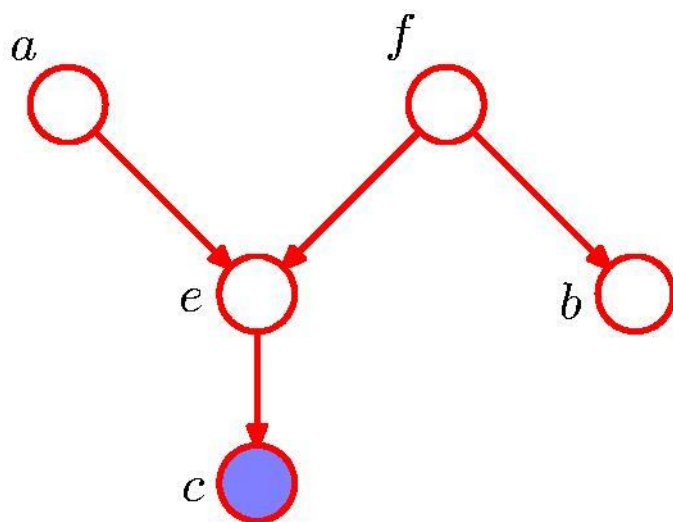
D-separation (D代表“directed”, Pearl 1988)

- A, B, and C are non-intersecting subsets of nodes in a directed graph.
- A path from A to B is blocked if it contains a node such that either
 - a) the arrows on the path meet either head-to-tail or tail-to-tail at the node, and the node is in the set C, or
 - b) the arrows meet head-to-head at the node, and neither the node, nor any of its descendants, are in the set C.
- If all paths from A to B are blocked, A is said to be d-separated from B by C.
- If A is d-separated from B by C, the joint distribution over all variables in the graph satisfies

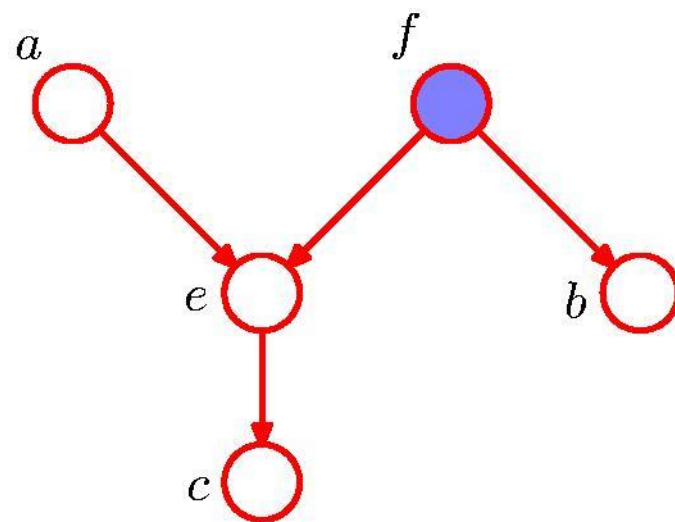
$$A \perp\!\!\!\perp B \mid C$$

有向概率图

D-separation 例子



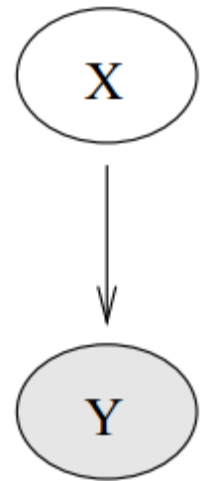
$$a \not\perp b \mid c$$



$$a \perp b \mid f$$

有向概率图

- 当节点中的变量是连续变量时，连续贝叶斯网，它的条件概率被称为条件概率分布（conditional probability distribution: CPD）
 - 右边这个最简单的例子，它代表了 $P(X,Y) = P(X)P(Y|X)$ ，此时CPD一般为参数模型，例如常见的高斯分布等
 - Y 是灰色表示 Y 是被观察到的变量，而 X 则是隐变量
- 概率图模型最常见的两类任务：
 - 推理（inference），给定观察变量数据（ Y ），如何推断出隐藏变量的状态？
 - 学习（learning），如何学习模型中的参数（例如高斯分布的参数）和结构（本课程不涉及）



有向概率图

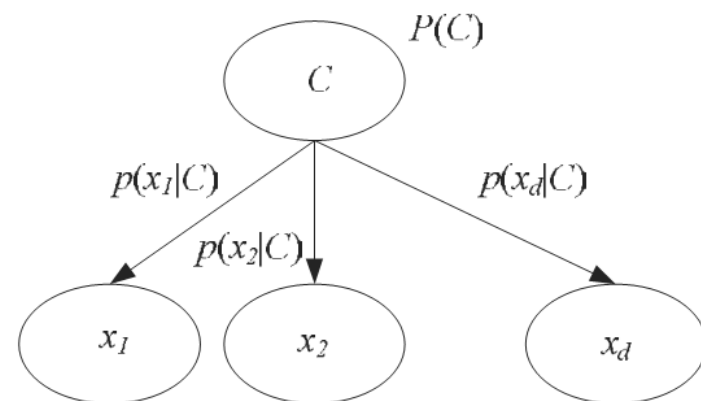
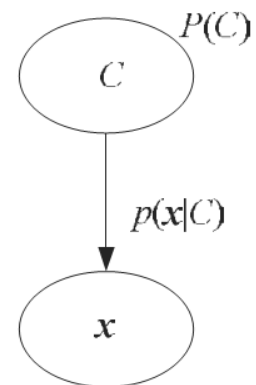
几个典型例子

- 朴素贝叶斯 (Naïve Bayes)
 - 模型, x : 输入, C : 类别, 一般为多项式分布
 - 分类: 求 $p(C|x)$, 利用 Bayes 公式

$$p(C|x) = \frac{p(x|C)p(C)}{p(x)}$$

- 如果 x 维数很高, 那么 $p(x|C)$ 难以估计 (维数灾)
- Naïve Bayes (NB) 假设给定 C , $x = \{x_1, x_2, \dots, x_d\}$ 之间相互独立

$$p(x|C) = \prod_{i=1}^d p(x_i|C)$$



有向概率图

几个典型例子

- 混合高斯模型 (Mixture of Gaussian)
 - 隐藏变量 $q = [q_1, q_2, \dots, q_k]^T$ 为多项式随机变量, 先验概率为

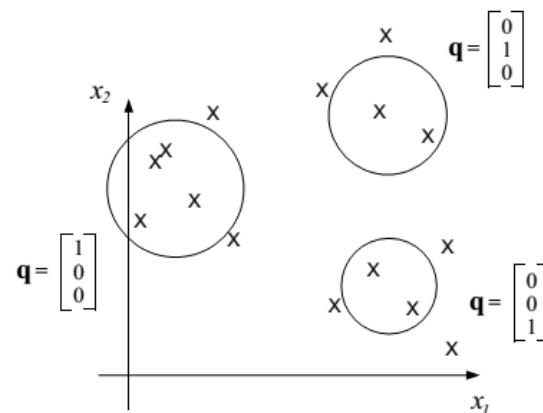
$$\pi_i = P(q_i = 1)$$

- 观察 (输出) 量 x 为高斯变量, 类-条件 (class-conditional) 概率为高斯分布

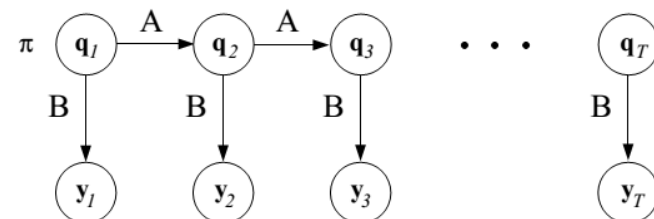
$$p(x|q_i = 1) = G(x|\Sigma_i, \mu_i)$$

- 推理问题 (聚类问题) 为

$$p(q_i = 1|x) = \frac{\pi_i G(x|\Sigma_i, \mu_i)}{\sum_i \pi_i G(x|\Sigma_i, \mu_i)}$$



有向概率图



几个典型例子

- 隐马尔科夫模型（Hidden Markov Model）
 - 一类著名的时序模型，也可称为动态混合模型（mixture model with dynamics）
 - T 为时间， \mathbf{q}_t 是一个多项式变量（ M 个状态），状态之间的转移矩阵 $A = P(\mathbf{q}_{t+1}|\mathbf{q}_t)$ ，最初状态为 $\pi = P(\mathbf{q}_1)$
 - A 不随时间变化，则称为齐次马尔科夫链
 - Y 可以为离散的（多项式变量），也可以是连续的，如高斯（混合高斯） $p(y_t = y|\mathbf{q}_t = i) = G(y|\Sigma_i, \mu_i)$
 - 马尔科夫性质如 $P(\mathbf{q}_{t+1}|\mathbf{q}_t, \mathbf{q}_{t-1}) = P(\mathbf{q}_{t+1}|\mathbf{q}_t)$ ，它的联合概率分布为

$$p(\mathbf{q}, \mathbf{y}) = p(\mathbf{q}_1)p(y_1|\mathbf{q}_1) \prod_{t=2}^T p(\mathbf{q}_t|\mathbf{q}_{t-1})p(y_t|\mathbf{q}_t)$$

- 推理问题为给出观察数据 \mathbf{y}_t ，如何计算隐藏节点（状态）的后验概率

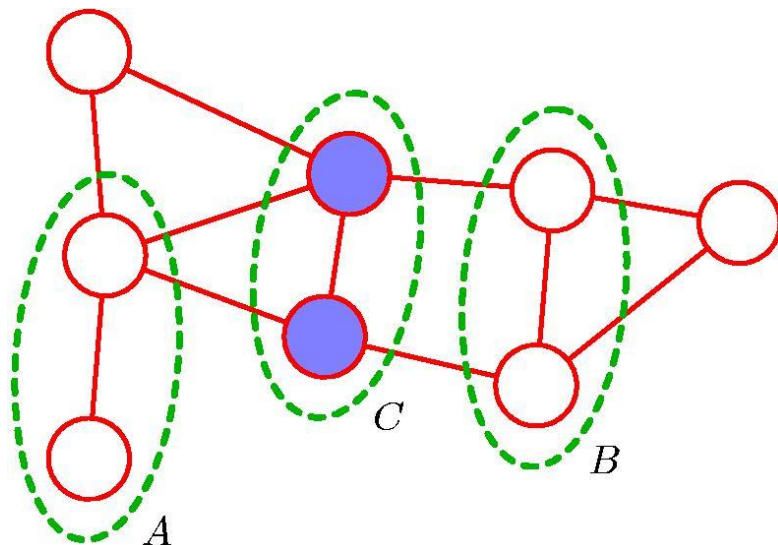
无向概率图（MRF）

- 也称为马尔科夫随机场（Markov random field: MRF）或者马尔科夫网（Markov networks），处理图像的有效工具！

- MRF的条件独立性相对容易判断

$$A \perp\!\!\!\perp B | C$$

- ✓ 将 C 以及和它相连的边去除， A ， B 之间不再存在通路
- ✓ A ， B 的任何通路之间必须经过至少 C 中的一个节点



无向概率图（MRF）

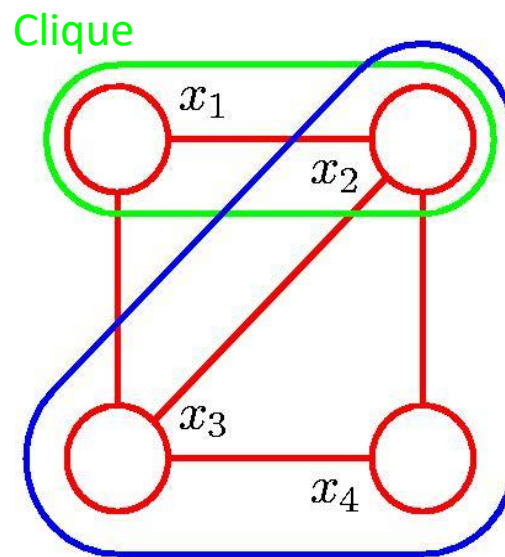
Factorization

- 不直接相连的两个节点 x_i, x_j 不应该出现在一起，因为

$$p(x_i, x_j | \mathbf{x} \setminus \{x_i, x_j\})$$

$$= p(x_i | \mathbf{x} \setminus \{x_i, x_j\}) p(x_j | \mathbf{x} \setminus \{x_i, x_j\})$$

- 定义“簇”（clique）
 - 图的一个子集，其中所有的节点都两两相连
 - 右图包含5个簇
 - 最大簇，“规则”图中只需要考虑最大簇



Maximal Clique

无向概率图 (MRF)

Joint distribution

$$p(\mathbf{x}) = \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C)$$

- where $\psi_C(\mathbf{x}_C)$ is the potential over clique \mathbf{C} and

$$Z = \sum_{\mathbf{x}} \prod_C \psi_C(\mathbf{x}_C)$$

is the normalization coefficient; 相对于有向图则没有此项

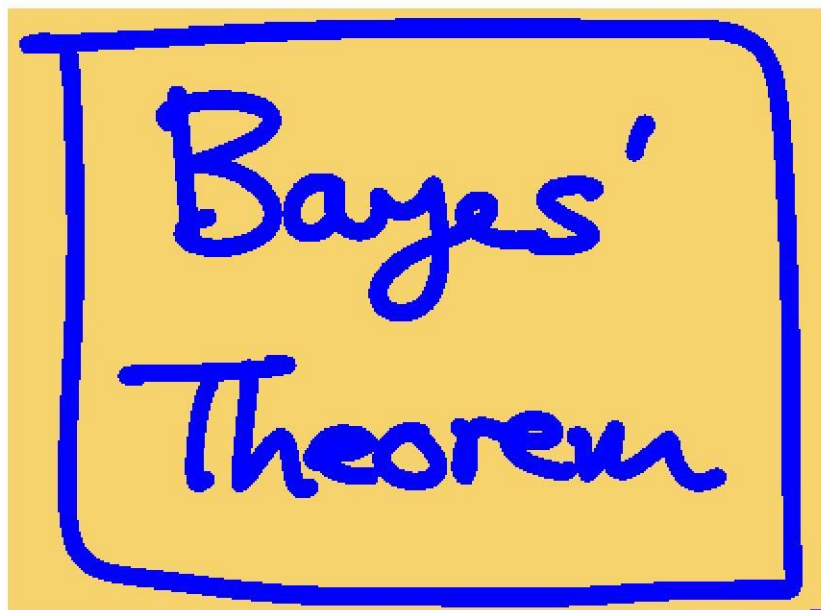
- **Energies** and the Boltzmann distribution

$$\psi_C(\mathbf{x}_C) = \exp \{-E(\mathbf{x}_C)\}$$

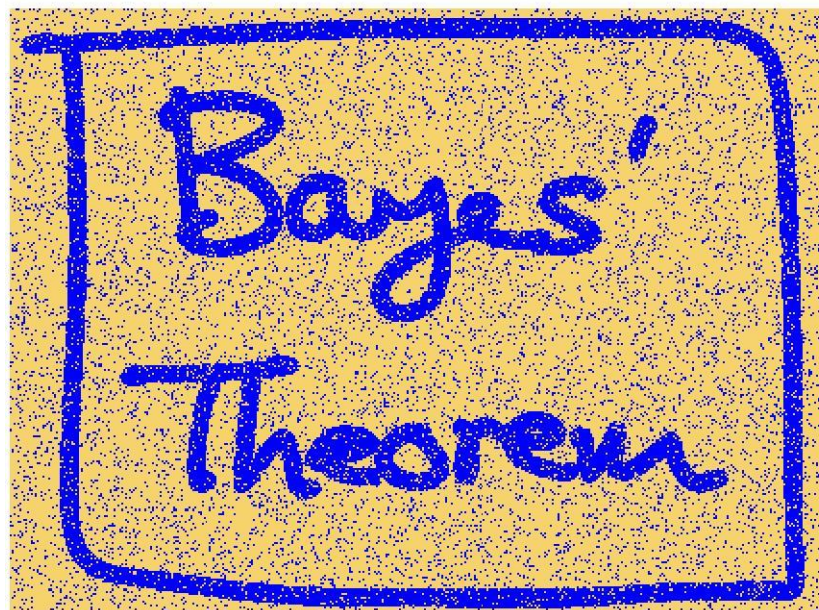
- Distribution的相乘 = 能量的相加 ---- 计算机视觉中的能量最小化方法
- 如何设计 E 是一个关键问题!

无向概率图 (MRF)

一个简单的例子：图像去噪



Original Image

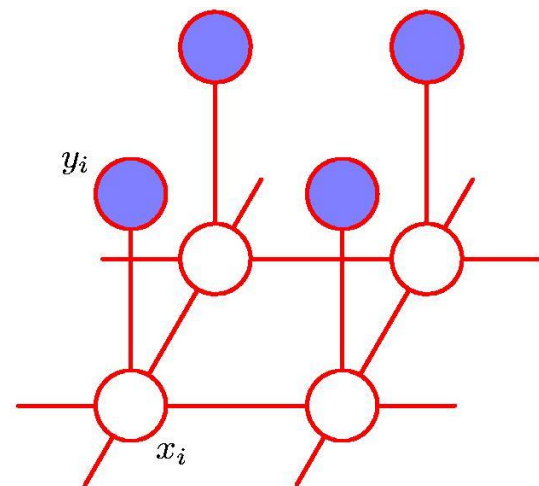


Noisy Image

无向概率图（MRF）

一个简单的例子：图像去噪

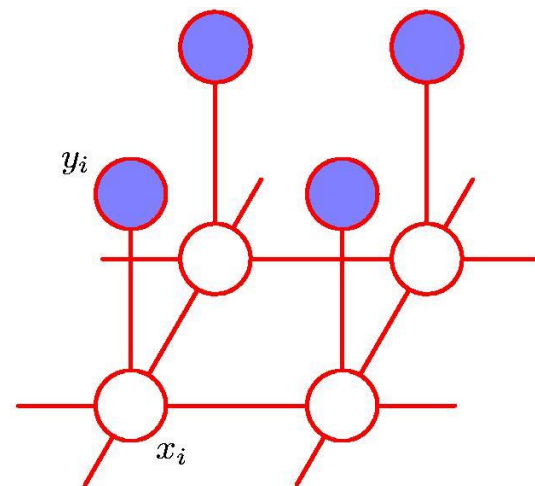
- Observed image $y_i = \{-1, 1\}, i = 1, 2, \dots, N, N$ 为总的像素数目
- y 是被噪声污染观察到的变量（图像）， x 是隐藏变量（待复原的图像）
- 相邻像素间的值相关，也即相邻的 x 值相关； x 和对应的 y 值相关
- 构建一个MRF，包含两类clique，即 $\{x_i, x_j\}, \{x_i, y_i\}$



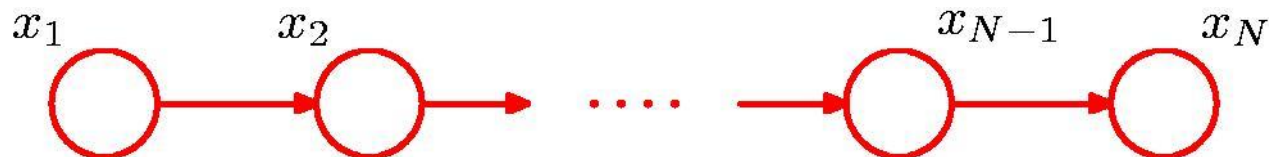
无向概率图（MRF）

一个简单的例子：图像去噪

- 针对两类clique设计能量函数
 - $\{x_i, x_j\}$: $V_{ij} = x_i x_j$, 使得两者尽量相同, smooth先验
 - $\{x_i, y_i\}$: $d_i = x_i y_i$, 使得两者尽量相同, likelihood
 - $E(X, Y) = \beta \sum_{\{i,j\}} V_{ij} + \eta \sum_i d_i$
 - $p(X, Y) = \frac{1}{Z} e^{-E(X, Y)}$
- Y 为观察到的变量, $p(X, Y)$ 某种意义上反映了 $p(X|Y)$, 因此, 我们最大化 $p(X, Y)$, 也就是最小化能量函数
- 计算机视觉中的能量最小化方法! [A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors, TPAMI2008]

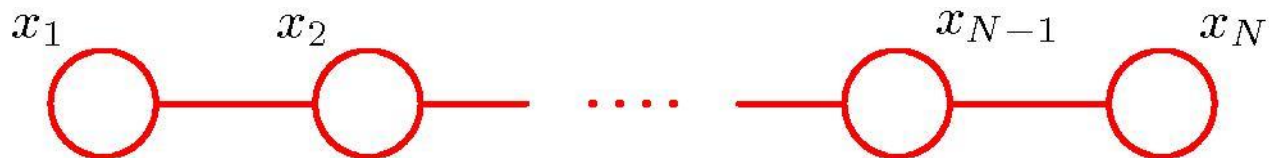


有向 v.s. 无向



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

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



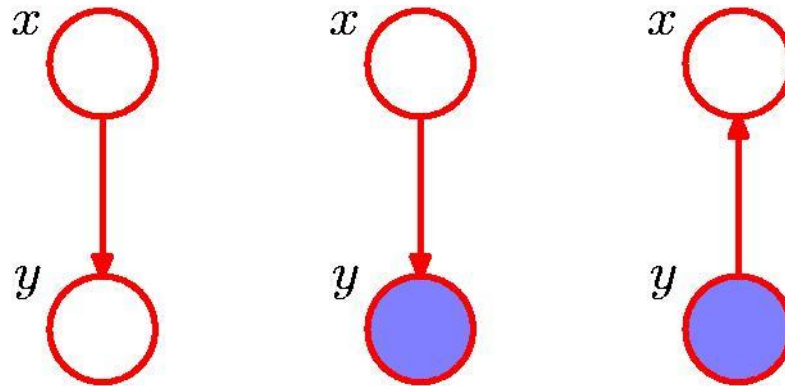
无向 v.s. 有向

有向图转换成无向图

- Chain（每个节点拥有至多一个父节点）
- 将无向图两两节点之间的potential设置成有向图的条件分布（没有父节点 $p(x_1)$ 放入第一个potential的），那么这两个图完全等价
- Implying $Z = 1$
- 其它类型的有向图转换成无向图（略）
 - 不是所有的图都可以相互转换的

图的推理 (inference on a graph)

Inference: 基于某些观察节点（具有观察值），计算剩下的某个或某些节点的后验概率



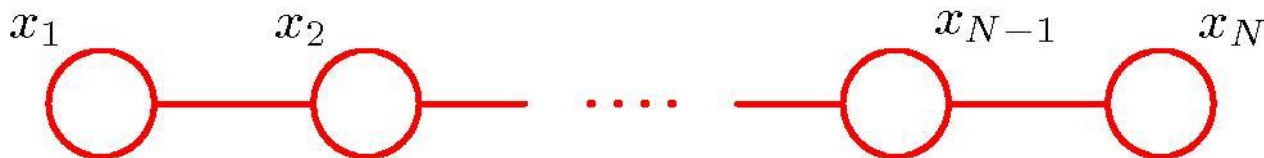
$$p(y) = \sum_{x'} p(y|x')p(x')$$

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

图的推理 (inference on a graph)

Inference on a chain

- 考虑一个无向链图（有向链图可以等价转换）



- 假如我们的任务要求得某个节点 n 的边际分布 $p(x_n)$

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

- 假如每个样本有 K 个状态，那么需要 K^N 存储和计算复杂度（这是传统方法）

图的推理 (inference on a graph)

Inference on a chain

- 如果考虑条件独立性

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

$$\begin{aligned} - p(x_n) &= \sum_{x_1} \dots \sum_{x_{n-1}} \sum_{x_{n+1}} \dots \sum_{x_N} \frac{1}{Z} \psi_{1,2}(x_1, x_2) \psi_{2,3}(x_2, x_3) \dots \psi_{N-1,N}(x_{N-1}, x_N) = \\ &= \frac{1}{Z} \sum_{x_1} \dots \sum_{x_{n-1}} \sum_{x_{n+1}} \dots \psi_{1,2}(x_1, x_2) \psi_{2,3}(x_2, x_3) \dots \sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N) = \dots \end{aligned}$$

$$\begin{aligned} p(x_n) &= \frac{1}{Z} \underbrace{\left[\sum_{x_{n-1}} \psi_{n-1,n}(x_{n-1}, x_n) \dots \left[\sum_{x_1} \psi_{1,2}(x_1, x_2) \right] \dots \right]}_{\mu_\alpha(x_n)} \\ &\quad \underbrace{\left[\sum_{x_{n+1}} \psi_{n,n+1}(x_n, x_{n+1}) \dots \left[\sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N) \right] \dots \right]}_{\mu_\beta(x_n)} \end{aligned}$$

图的推理 (inference on a graph)

Inference on a chain

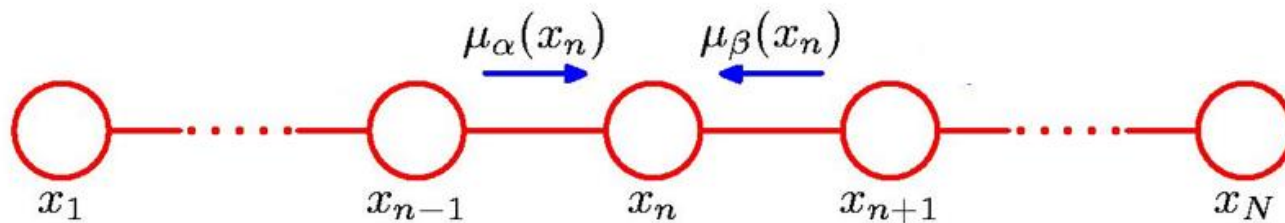
- 如果考虑条件独立性
- 复杂度: $O(NK^2)$ – 线性
- 形式: (big scale)sum \rightarrow (small scale)sum + (small scale)product

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

图的推理（inference on a graph）

Inference on a chain

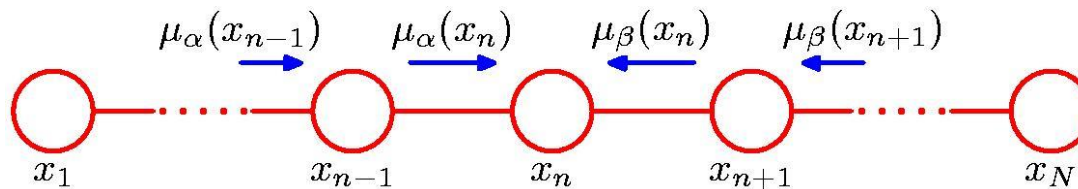
- 另一种解释：局部消息在图中的传递（local message passing around the graph）
 - $p(x_n) = \mu_\alpha(x_n)\mu_\beta(x_n)$
 - 将 $\mu_\alpha(x_n)$ 解读为沿着链表从 x_{n-1} 向 x_n 正向传播的消息（forward message）， $\mu_\beta(x_n)$ 解读为沿着链表从 x_{n+1} 向 x_n 反向传播的消息（backward message）



图的推理 (inference on a graph)

Inference on a chain

- $\mu_\alpha(x_n)$, $\mu_\beta(x_n)$ 可以迭代求解



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

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

$$\mu_\alpha(x_2) = \sum_{x_1} \psi_{1,2}(x_1, x_2)$$

$$\mu_\beta(x_{N-1}) = \sum_{x_N} \psi_{N-1,N}(x_{N-1}, x_N)$$

图的推理 (inference on a graph)

Inference on a chain

- To compute all local marginals:
 - Compute and store all forward messages, .
 - Compute and store all backward messages, .
 - Compute Z at any node x_n
 - Compute

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

for all variables required.

图的推理（inference on a graph）

Inference on a chain

- 上述讨论的节点皆为不可见的，如果某个节点 n_j 是可见的（观察值），那么对于 n_j 则无需求和
- 对于求邻域节点的联合概率分布也同样可以直接求取

$$p(x_{n-1}, x_n) = \frac{1}{Z} \mu_{\alpha}(x_{n-1}) \psi_{n-1,n}(x_{n-1}, x_n) \mu_{\beta}(x_n)$$

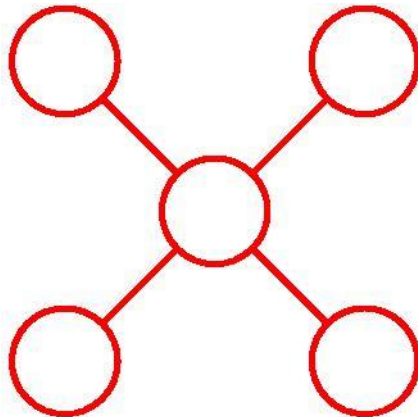
- 可直接用于potential或者参数化条件分布函数的参数估计（直接是EM算法中的E-step所需）

图的推理 (inference on a graph)

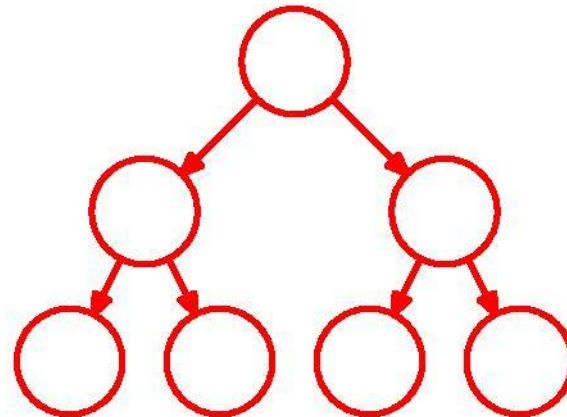
Inference on a tree

- 树 (tree) 的内涵：
 - 无向图：树意味着任意两个节点之间仅有一条通路 (path)
 - 有向图：树意味着有且仅有一个节点没有父节点，剩余的节点有且仅有一个父节点
 - 没有loop
 - 有向树可以直接转换成无向树 (拓扑结构不变)， why?

Undirected Tree



Directed Tree

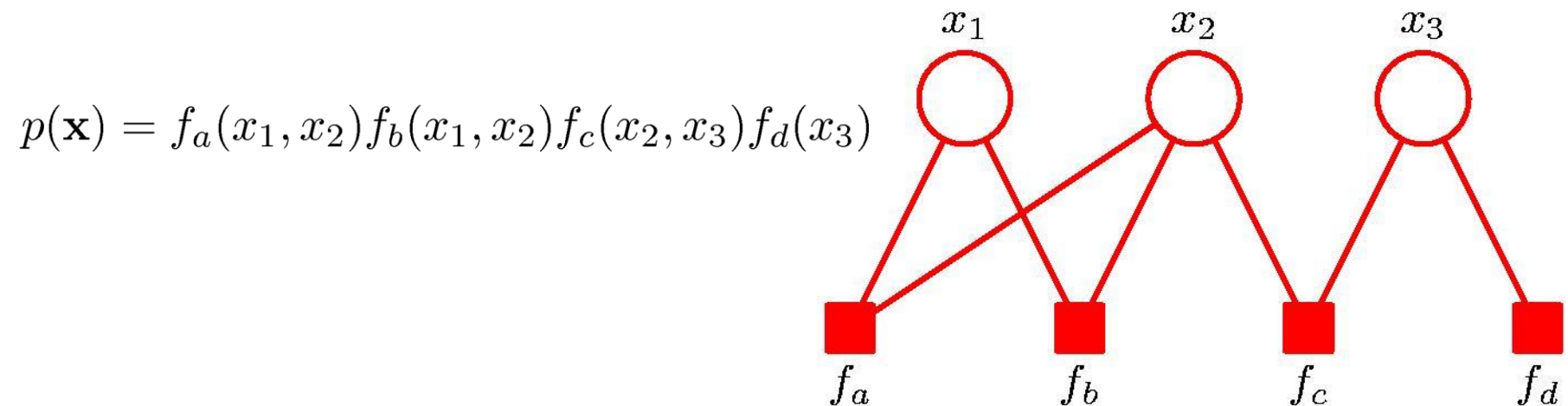


图的推理 (inference on a graph)

Inference on a tree: factor graphs

- Factor graphs: 在已有节点的基础上, 对potential (无向) 或conditional probability (有向) 引入新的节点, 使得概率图的factorization过程更为清晰

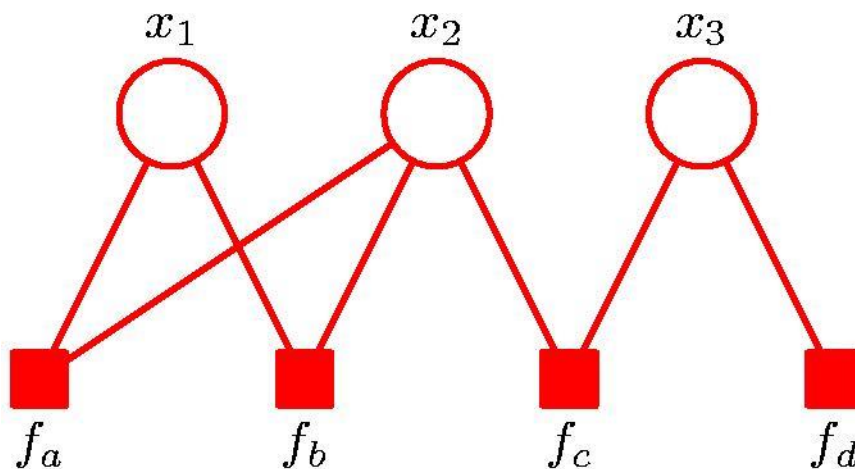
$$p(\mathbf{x}) = \prod_s f_s(\mathbf{x}_s)$$



图的推理 (inference on a graph)

Inference on a tree: factor graphs

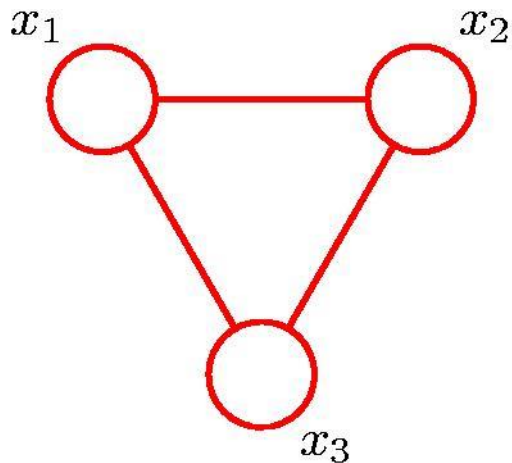
- Factor graph是一个二分图 (bipartite)
- 原来的节点不再连接, 而是通过新增的节点 f_s 来连接
- 有向树: 每个 f_s 可以是每个节点的条件概率
- 无向树: 每个 f_s 可以是每个clique的potential
- 树对应的factor graph还保持是树



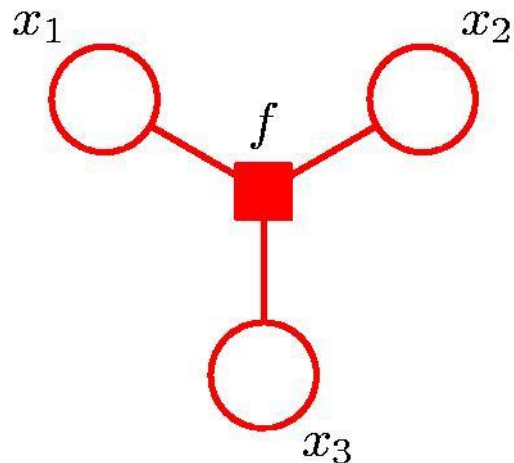
图的推理 (inference on a graph)

Inference on a tree: factor graphs

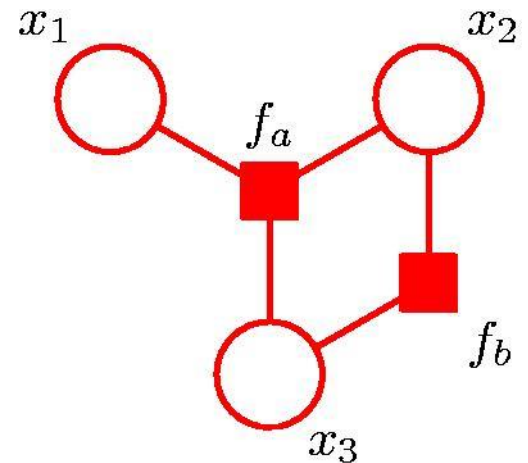
– 无向图 (分解不唯一)



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



$$\begin{aligned} f(x_1, x_2, x_3) \\ = \psi(x_1, x_2, x_3) \end{aligned}$$

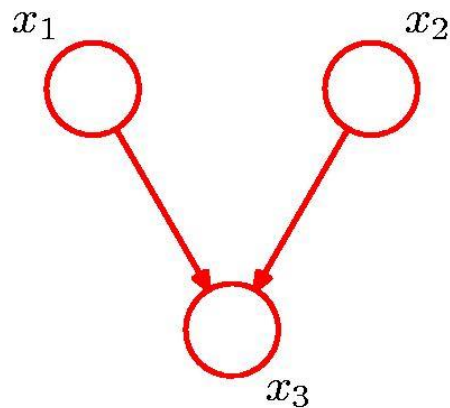


$$\begin{aligned} f_a(x_1, x_2, x_3) f_b(x_2, x_3) \\ = \psi(x_1, x_2, x_3) \end{aligned}$$

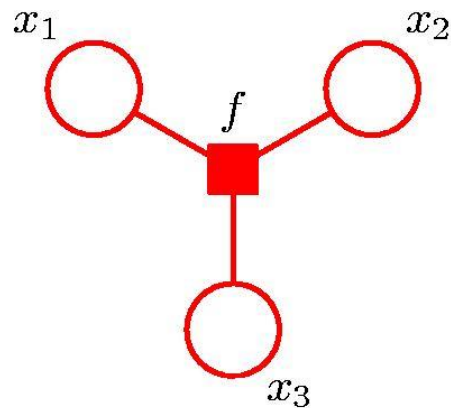
图的推理 (inference on a graph)

Inference on a tree: factor graphs

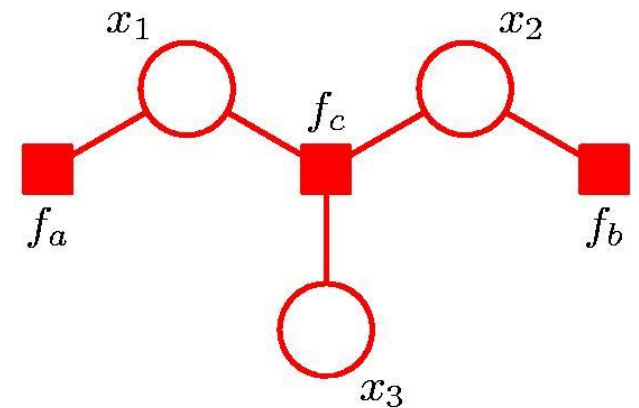
– 有向图 (分解不唯一)



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



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



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

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

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

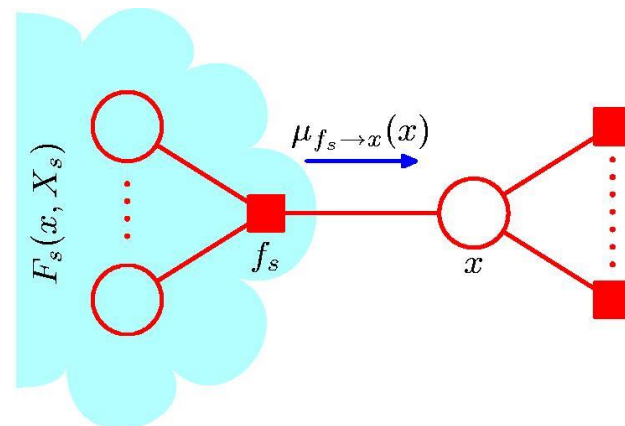
图的推理 (inference on a graph)

- Inference on a tree: **sum-product algorithm**
- 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.
- 有向、无向图都转成了factor graph，因此只需考虑factor graph的推理
- 还是以求某个节点 x 的边际分布 $p(x)$ 为例

图的推理 (inference on a graph)

Inference on a tree: **sum-product algorithm**

- $p(x) = \sum_{\mathbf{x} \setminus x} p(\mathbf{x})$
- $p(\mathbf{x}) = \prod_{s \in ne(x)} F_s(x, X_s)$
 - $ne(x)$ 表示与 x 相邻的factor节点,
 - X_s 表示和factor节点 s 相邻的所有变量 (除了 x)
 - $F_s(x, X_s)$ 表示和factor节点 f_s 相邻所有变量的potential (或者joint distribution)



图的推理 (inference on a graph)

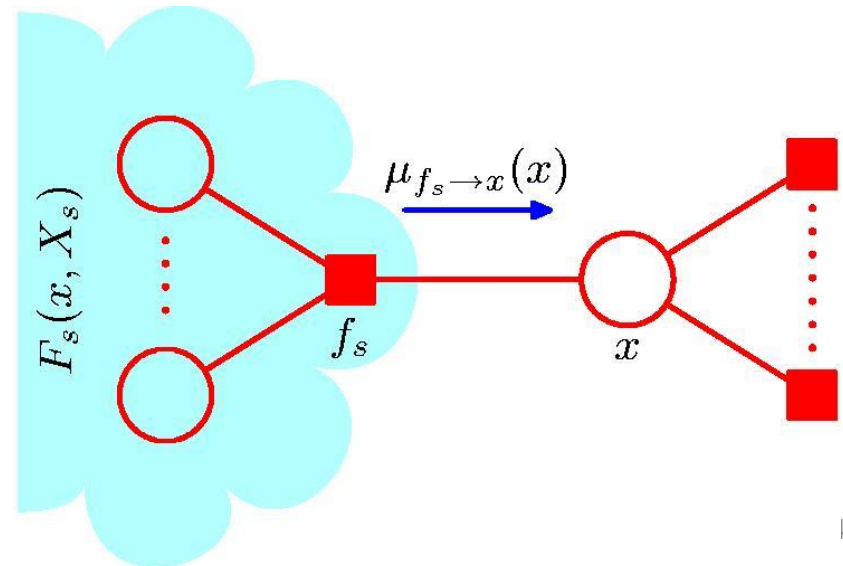
Inference on a tree: **sum-product algorithm**

- 进一步

$$p(x) = \sum_{x \setminus x} \prod_{s \in ne(x)} F_s(x, X_s) = \prod_{s \in ne(x)} \sum_{X_s} F_s(x, X_s) = \prod_{s \in ne(x)} \mu_{f_s \rightarrow x}(x), \text{ where we define}$$

$$\mu_{f_s \rightarrow x}(x) \equiv \sum_{X_s} F_s(x, X_s)$$

- $\mu_{f_s \rightarrow x}(x)$ 可以看成是 **factor 节点 f_s 向变量节点 x 传播的消息 (message)**



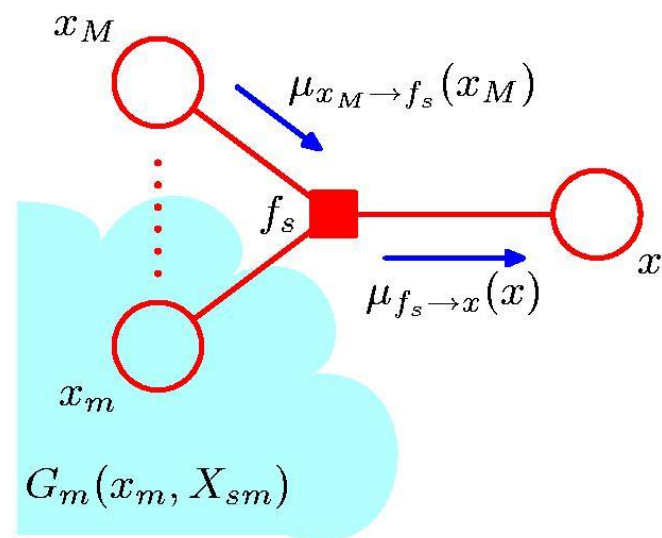
图的推理 (inference on a graph)

Inference on a tree: **sum-product algorithm**

- $F_S(x, X_S) = f_S(x, X_S) \prod_{m \in ne(f_S) \setminus x} \hat{p}(x_m)$
- $\mu_{x_m \rightarrow f_S}(x_m) \equiv \hat{p}(x_m) = \sum_{X_{sm}} G_m(x_m, X_{sm})$
- $\mu_{x_m \rightarrow f_S}(x_m)$ 可以看成是变量节点 x_m 向变量factor节点 f_S 传播的消息 (message)
- $G_m(x_m, X_{sm})$ 内涵上等价于 $F_S(x, X_S)$ (除了 f_S 要排除)

$$G_m(x_m, X_{sm}) = \prod_{l \in ne(x_m) \setminus f_S} F_l(x_m, X_{ml})$$

$$\mu_{x_m \rightarrow f_S}(x_m) = \sum_{X_{sm}} \prod_{l \in ne(x_m) \setminus f_S} F_l(x_m, X_{ml}) = \prod_{l \in ne(x_m) \setminus f_S} \sum_{X_{ml}} F_l(x_m, X_{ml}) = \prod_{l \in ne(x_m) \setminus f_S} \mu_{f_l \rightarrow x_m}(x_m)$$



图的推理 (inference on a graph)

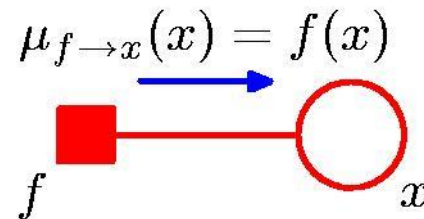
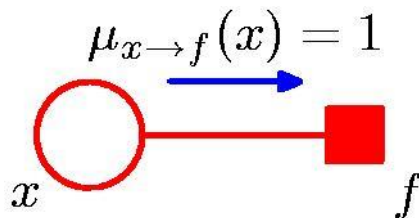
Inference on a tree: **sum-product algorithm**

- 两条公式

$$- \mu_{f_s \rightarrow x}(x) = \sum_{X_S} f_s(x, X_S) \prod_{m \in ne(f_s) \setminus x} \mu_{x_m \rightarrow f_s}(x_m)$$

$$- \mu_{x_m \rightarrow f_s}(x_m) = \prod_{l \in ne(x_m) \setminus f_s} \mu_{f_l \rightarrow x_m}(x_m)$$

- 初始值



图的推理 (inference on a graph)

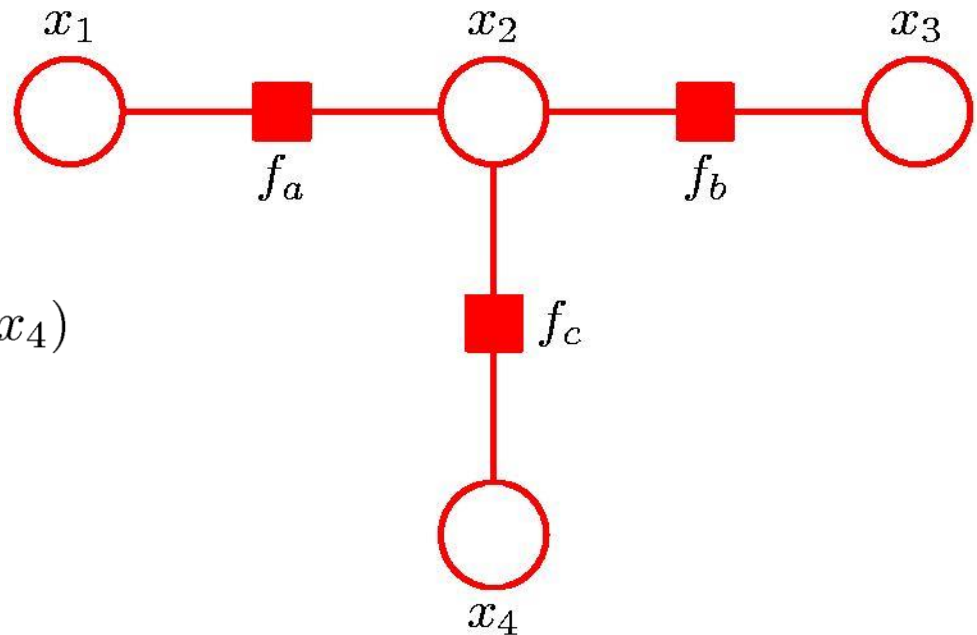
Inference on a tree: **sum-product algorithm**

- 如果要计算很多的margins怎么办?
 - 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.
- NOTE:
 - SUM \rightarrow Sum-Product, 充分利用条件独立 “化整为块”
 - The famous belief propagation is a special case of sum-product

图的推理 (inference on a graph)

Inference on a tree: an example

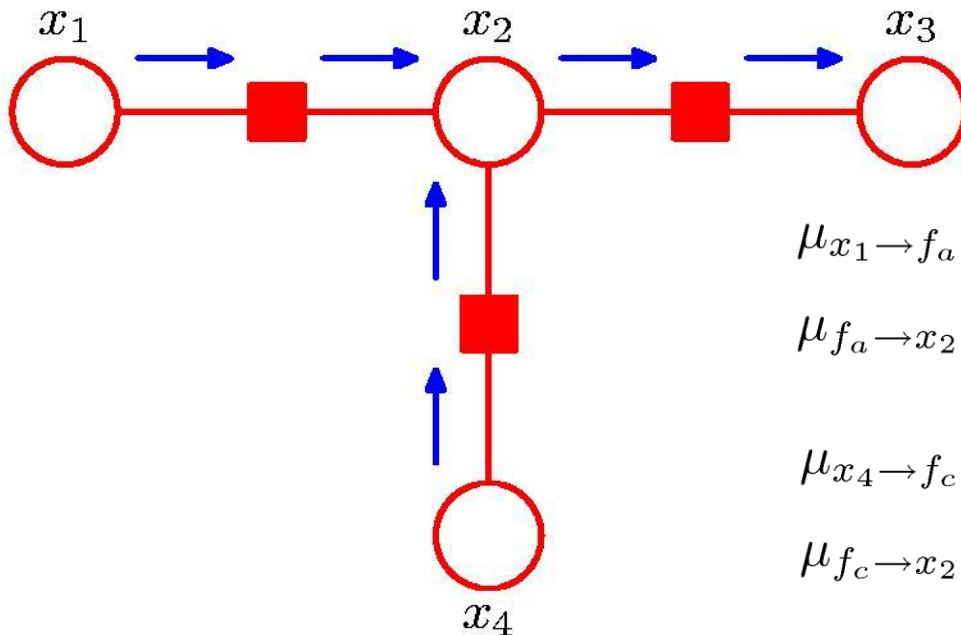
- Calculate $p(x_3)$



$$\tilde{p}(\mathbf{x}) = f_a(x_1, x_2) f_b(x_2, x_3) f_c(x_2, x_4)$$

图的推理 (inference on a graph)

Inference on a tree: an example



$$\mu_{x_1 \rightarrow f_a}(x_1) = 1$$

$$\mu_{f_a \rightarrow x_2}(x_2) = \sum_{x_1} f_a(x_1, x_2)$$

$$\mu_{x_4 \rightarrow f_c}(x_4) = 1$$

$$\mu_{f_c \rightarrow x_2}(x_2) = \sum_{x_4} f_c(x_2, x_4)$$

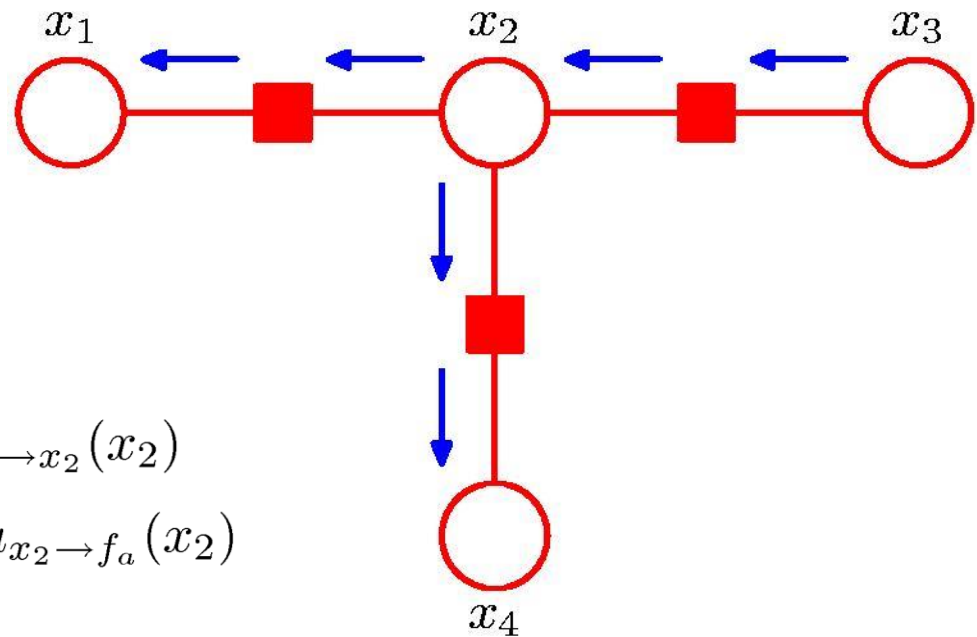
$$\mu_{x_2 \rightarrow f_b}(x_2) = \mu_{f_a \rightarrow x_2}(x_2) \mu_{f_c \rightarrow x_2}(x_2)$$

$$\mu_{f_b \rightarrow x_3}(x_3) = \sum_{x_2} f_b(x_2, x_3) \mu_{x_2 \rightarrow f_b}(x_2)$$

图的推理 (inference on a graph)

Inference on a tree: an example

- 反向传播



$$\mu_{x_3 \rightarrow f_b}(x_3) = 1$$

$$\mu_{f_b \rightarrow x_2}(x_2) = \sum_{x_3} f_b(x_2, x_3)$$

$$\mu_{x_2 \rightarrow f_a}(x_2) = \mu_{f_b \rightarrow x_2}(x_2) \mu_{f_c \rightarrow x_2}(x_2)$$

$$\mu_{f_a \rightarrow x_1}(x_1) = \sum_{x_2} f_a(x_1, x_2) \mu_{x_2 \rightarrow f_a}(x_2)$$

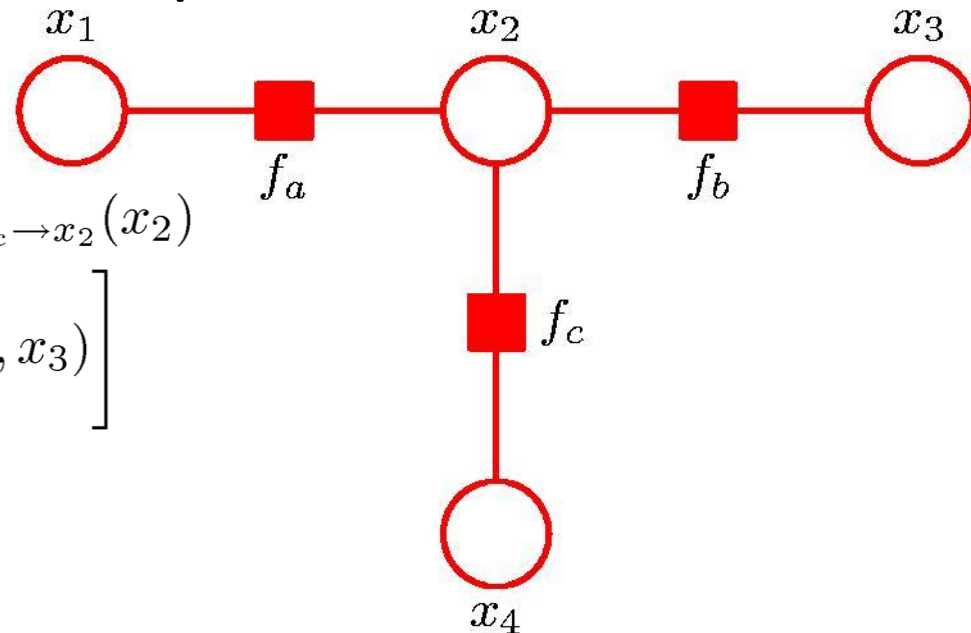
$$\mu_{x_2 \rightarrow f_c}(x_2) = \mu_{f_a \rightarrow x_2}(x_2) \mu_{f_b \rightarrow x_2}(x_2)$$

$$\mu_{f_c \rightarrow x_4}(x_4) = \sum_{x_2} f_c(x_2, x_4) \mu_{x_2 \rightarrow f_c}(x_2)$$

图的推理 (inference on a graph)

Inference on a tree: an example

- Verification on x_2



$$\begin{aligned}\tilde{p}(x_2) &= \mu_{f_a \rightarrow x_2}(x_2) \mu_{f_b \rightarrow x_2}(x_2) \mu_{f_c \rightarrow x_2}(x_2) \\ &= \left[\sum_{x_1} f_a(x_1, x_2) \right] \left[\sum_{x_3} f_b(x_2, x_3) \right] \\ &\quad \left[\sum_{x_4} f_c(x_2, x_4) \right] \\ &= \sum_{x_1} \sum_{x_3} \sum_{x_4} f_a(x_1, x_2) f_b(x_2, x_3) f_c(x_2, x_4) \\ &= \sum_{x_1} \sum_{x_3} \sum_{x_4} \tilde{p}(\mathbf{x})\end{aligned}$$

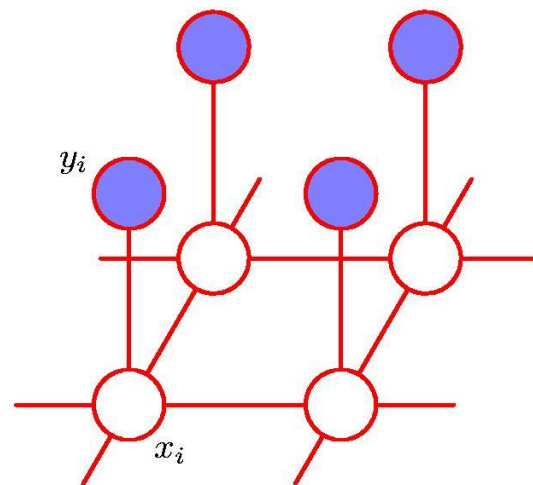
图的推理 (inference on a graph)

- Exact inference (on chain and tree)
 - Max-Sum algorithm: find the value x^{\max} that maximises $p(x)$; the value of $p(x^{\max})$.
- Inference on general graph
 - Much more difficult than on Chain and Tree, (Inference on MRF as an example)
 - Loopy belief propagation
 - Variational Bayesian
 - MCMC(Markov Chain Monte Carlo)

图的推理（inference on a graph）

Inference on graph: MRF

- 用于解决处理视觉中的pixel labeling问题，MRF的结构一般设置为Ising模型，也即grid（网格结构），即一个clique对应于一条边，二阶能量函数
- x 为隐藏变量， y 为观察变量，inference为给定观察值 y ， x 取什么值使得后验 $p(x|y)$ （用 $p(x,y)$ ）最大
- x 的取值范围为标签集合 l
 - Label需要指定
 - 前面的去噪模型标签为 $l = \{-1,1\}$
- 能量函数也即potential需要设计
 - label l 尽可能和 y 吻合
 - 邻域间（clique）的label l 平滑



图的推理 (inference on a graph)

Inference on graph: MRF

- 给定 $l = \{1, 2, \dots, k\}$, $|V| = n$, 那么问题的复杂度为 $O(k^n)$
- 常用的方法
 - ICM (iterated conditional modes): 梯度法的一种, 结果较为粗糙
 - Graph cuts: 要求能量函数为 submodular 函数, 两类问题可在 P 复杂度得到全局最优
 - Loopy belief propagation: 有时候结果很差, 理论上没有保障
 - **Relaxation 方法**: 近几年得到发展, 优势在于对能量函数形式没有任何要求

图推理 (inference on a graph)

Inference on MRF: 另一个角度

- $E = E_d + \lambda E_s = \sum_i d_i(l_i, y_i) + \beta \sum_{\{i,j\} \in C} V_{ij}(l_i, l_j)$
- $d_i(l_i, y_i)$ 称为数据项（似然），表示 x_i 取 l_i 时与 y_i 之间的误差，describing how well label l_i agrees with the data at site i
- $V_{ij}(l_i, l_j)$ 称为平滑项（先验），表示相邻的节点分别取 l_i, l_j 时的惩罚项，describing how well the vertex i with label l_i agrees with the vertex j with label l_j
- 可以改写为

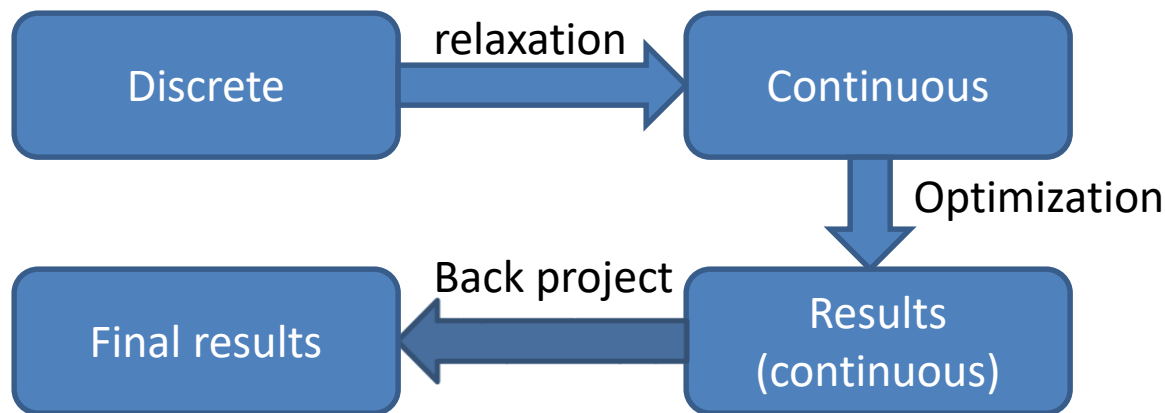
$$E(\mathbf{a}) = \mathbf{d}^T \mathbf{a} + \lambda \mathbf{a}^T \mathbf{V} \mathbf{a}$$

- \mathbf{a} 是一个 $kn \times 1$ 的待求解向量，可以理解为由矩阵 $\mathbf{A} \in R^{k \times n}$ “拉直”为一向量得到，且 \mathbf{A} 的元素只能取值0或1，且每一列有且只有一个1，代表每个像素只能有一个label
- \mathbf{d} 是一个 $kn \times 1$ 数据误差的向量，与 \mathbf{a} 类似，可以理解为矩阵 $\mathbf{D} \in R^{k \times n}$ “拉直”为一向量得到， \mathbf{D}_{ia} 表示第 i 个节点取第 a 个label时的数据误差
- \mathbf{V} 是一个 $kn \times kn$ 对称的平滑惩罚矩阵， $\mathbf{V}_{ia,jb}$ 表示第 i 个节点取第 a 个label和第 j 个节点取第 b 个label之间的平滑惩罚

图推理（inference on a graph）

Inference on MRF: relaxation方法

- 一个典型的离散组合优化（combinatorial optimization）NP难问题
- 松弛法（relaxation）
 - **POINT**: Continuous optimization is usually easier to be approximated than its discrete counterpart
 - A general procedure of relaxation approaches



图推理（inference on a graph）

Inference on MRF: relaxation方法

- Relaxation:
 - 将问题的离散可行域放松为连续可行域，转换成一个连续优化问题
 - 一般为找到问题的凸松弛函数（困难！）
- Optimization
 - 不同的松弛方式需要不同的优化框架，如求特征向量、条件优化等
- Back-projection
 - 将连续解映射回离散解，利用直接投影、Graduated Assignment等确定性退火等方法（困难！）

图推理（inference on a graph）

Inference on MRF: relaxation方法

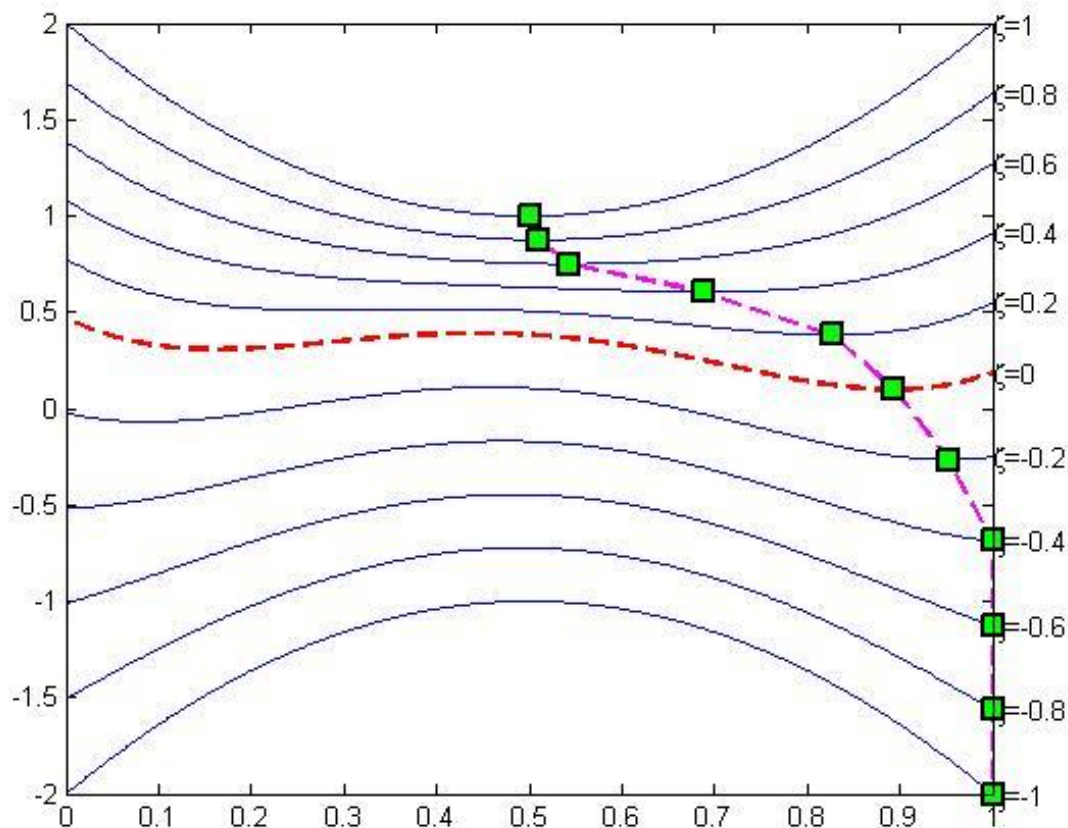
- 渐非凸渐凹化过程 – GNCCP（Graduated NonConvexity and Concavity Procedure） [Liu et al. PAMI 2014]

– 给定一个组合图优化问题，凸、凹松弛函数都可以自主构建，完全彻底解决了凸凹松弛过程的难题

$$F_{\zeta}(\mathbf{X}) = \begin{cases} (1 - \zeta)F(\mathbf{X}) + \zeta \text{tr} \mathbf{X}^{\top} \mathbf{X} & \text{if } 1 \geq \zeta \geq 0, \\ (1 + \zeta)F(\mathbf{X}) + \zeta \text{tr} \mathbf{X}^{\top} \mathbf{X} & \text{if } 0 > \zeta \geq -1, \end{cases}, \mathbf{X} \in \Omega.$$

图推理（inference on a graph）

- GNCCP收敛示意图：凸凹松弛函数隐性自动实现



图推理 (inference on a graph)

- GNCCP based MRF MAP algorithm

Algorithm 3.1: GNCCP MAP ALGORITHM()

```
 $\zeta \leftarrow -1, \mathbf{x} \leftarrow \mathbf{1}/m$   
repeat  
  repeat  
     $\mathbf{y} = \arg \max_{\mathbf{y}} \nabla F_{\zeta}(\mathbf{x})^{\top} \mathbf{y}, \text{ s.t. } \mathbf{y} \in \Omega$   
     $\alpha = \arg \max_{\alpha} F_{\zeta}(\mathbf{x} + \alpha(\mathbf{y} - \mathbf{x})), \text{ s.t. } 0 \leq \alpha \leq 1$   
     $\mathbf{x} \leftarrow \mathbf{x} + \alpha(\mathbf{y} - \mathbf{x})$   
  until converged  
   $\zeta \leftarrow \zeta + d\zeta$   
until  $\zeta > 1 \vee \mathbf{x} \in \Pi$   
return  $(\mathbf{x})$ 
```


图推理（inference on a graph）

GNCCP算法解释

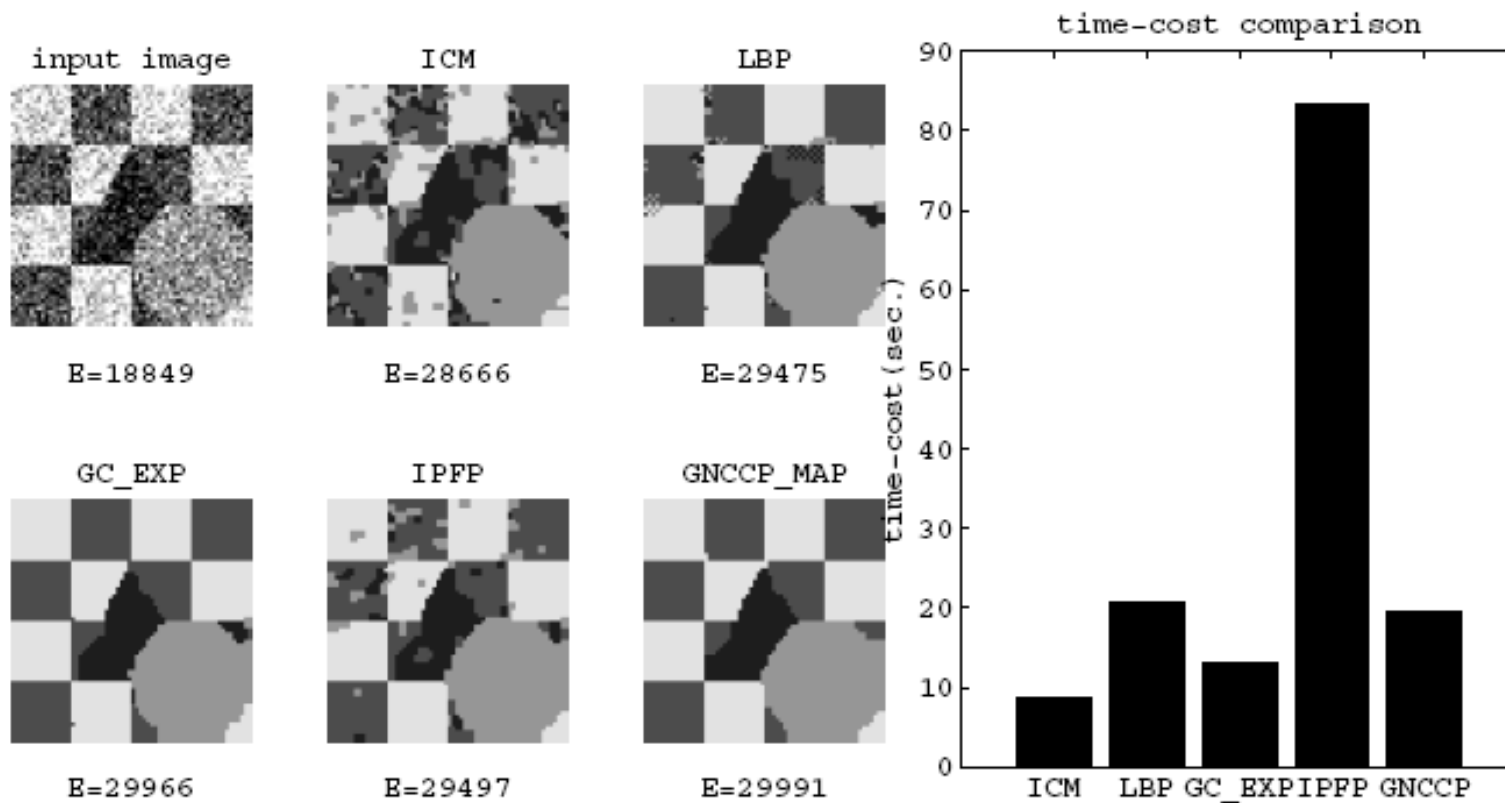
- 内循环采用Frank-Wolfe算法
- **线性指派**步骤由匈牙利算法实现（一对一情况）
或者直接计算每一列的最大值实现（一对多情况）
- **线性搜索**可以解析实现或者利用backtracking算法实现
- 算法中的梯度为
$$\nabla F_{\zeta}(X) = \begin{cases} (1 - \zeta)\nabla F(X) + \zeta X & \text{if } 1 \geq \zeta \geq 0, \\ (1 + \zeta)\nabla F(X) + \zeta X & \text{if } 0 > \zeta \geq -1. \end{cases}$$
- 算法唯一要求的就是求**原函数的梯度！！**

GNCCP特点

- 模型特点：
 - 一种确定性的退火过程（deterministic annealing）
 - 简单来说，它的搜索方向是确定的
 - 更快的收敛性（相对随机退火）
 - 由于引入凹松弛函数，CCRP从定义上和原问题完全等价
 - 求凹松弛函数的极小点等价于求原离散问题的极小点
 - 可能是文献中首个和原离散问题完全等价的松弛法框架
- 性能特点
 - 简单易用：只需要求导数
 - 复杂度低： $\mathcal{O}(|E|k^2)$
 - 精度高：在图割和LBP适用的前提下精度相当
 - 适用范围广：适用于任意的势场函数和任意的图结构

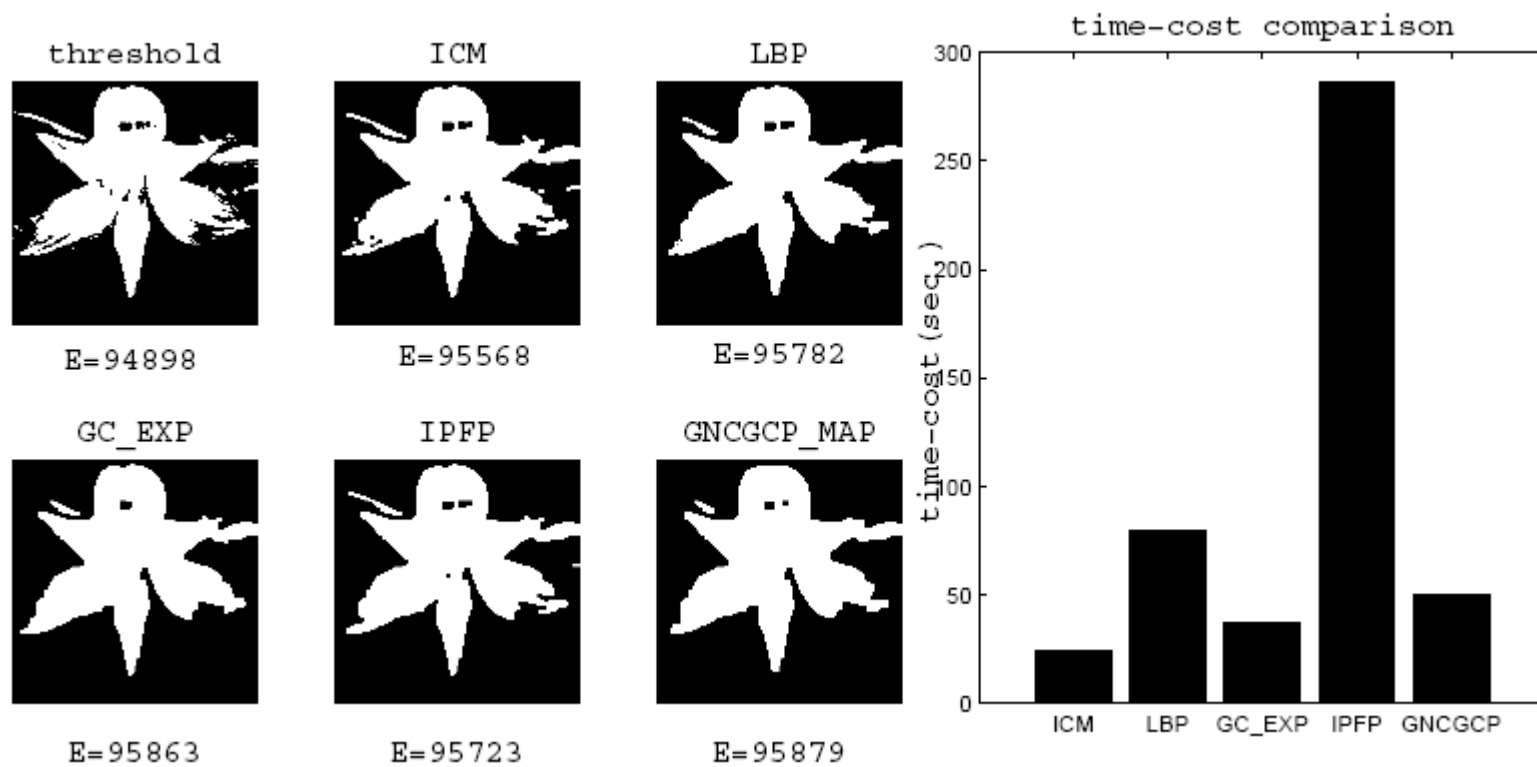
实验对比- GNCCP-EM

- 图像分割



实验对比- GNCCP-EM

- 图像分割



The END, Thanks!